

6man
Internet-Draft
Intended status: Standards Track
Expires: January 25, 2020

Z. Ali
C. Filsfils
Cisco Systems
S. Matsushima
Softbank
D. Voyer
Bell Canada
M. Chen
Huawei
July 24, 2019

Operations, Administration, and Maintenance (OAM) in Segment Routing
Networks with IPv6 Data plane (SRv6)
draft-ali-6man-spring-srv6-oam-03

Abstract

This document defines building blocks for Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Dataplane (SRv6). The document also describes some SRv6 OAM mechanisms.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

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 January 25, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Conventions Used in This Document	3
2.1. Abbreviations	3
2.2. Terminology and Reference Topology	3
3. OAM Building Blocks	5
3.1. O-flag in Segment Routing Header	5
3.1.1. O-flag Processing	6
3.2. OAM Segments	6
3.3. End.OP: OAM Endpoint with Punt	6
3.4. End.OTP: OAM Endpoint with Timestamp and Punt	7
3.5. SRH TLV	7
4. OAM Mechanisms	7
4.1. Ping	8
4.1.1. Classic Ping	8
4.1.2. Pinging a SID Function	9
4.1.3. Error Reporting	12
4.2. Traceroute	12
4.2.1. Classic Traceroute	13
4.2.2. Traceroute to a SID Function	14
4.3. Monitoring of SRv6 Paths	18
5. Security Considerations	19
6. IANA Considerations	19
6.1. ICMPv6 type Numbers RegistrySEC	19
6.2. SRv6 OAM Endpoint Types	19
7. Acknowledgements	20
8. Contributors	20
9. References	21
9.1. Normative References	21
9.2. Informative References	21
Authors' Addresses	22

1. Introduction

This document defines building blocks for Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Dataplane (SRv6). The document also describes some SRv6 OAM mechanisms.

2. Conventions Used in This Document

2.1. Abbreviations

The following abbreviations are used in this document:

SID: Segment ID.

SL: Segment Left.

SR: Segment Routing.

SRH: Segment Routing Header.

SRv6: Segment Routing with IPv6 Data plane.

TC: Traffic Class.

ICMPv6: multi-part ICMPv6 messages [RFC4884].

2.2. Terminology and Reference Topology

This document uses the terminology defined in [I-D.ietf-spring-srv6-network-programming]. The readers are expected to be familiar with the same.

Throughout the document, the following simple topology is used for illustration.

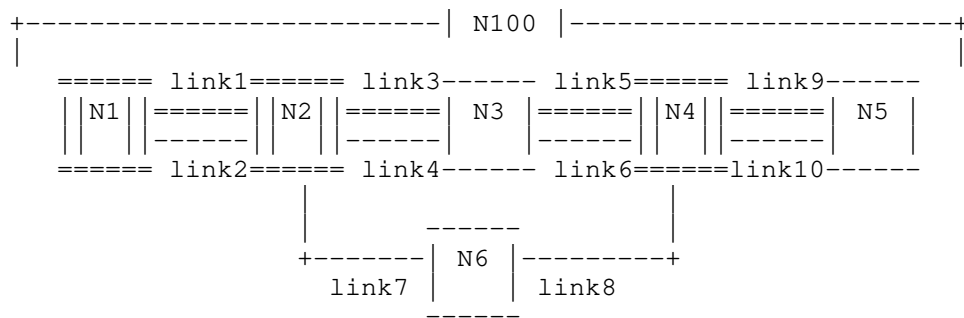


Figure 1 Reference Topology

In the reference topology:

Nodes N1, N2, and N4 are SRv6 capable nodes.

Nodes N3, N5 and N6 are classic IPv6 nodes.

Node N100 is a controller.

Node k has a classic IPv6 loopback address A:k::/128.

A SID at node k with locator block B and function F is represented by B:k:F::.

The IPv6 address of the nth Link between node X and Y at the X side is represented as 2001:DB8:X:Y:Xn::, e.g., the IPv6 address of link6 (the 2nd link) between N3 and N4 at N3 in Figure 1 is 2001:DB8:3:4:32::. Similarly, the IPv6 address of link5 (the 1st link between N3 and N4) at node 3 is 2001:DB8:3:4:31::.

B:k:Cij:: is explicitly allocated as the END.X function at node k towards neighbor node i via jth Link between node i and node j. e.g., B:2:C31:: represents END.X at N2 towards N3 via link3 (the 1st link between N2 and N3). Similarly, B:4:C52:: represents the END.X at N4 towards N5 via link10.

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) (payload) represents an IPv6 packet with:

- * IPv6 header with source address SA, destination addresses DA and SRH as next-header

- * SRH with SID list <S1, S2, S3> with SegmentsLeft = 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.
- * (payload) represents the the payload of the packet.

SRH[SL] represents the SID pointed by the SL field in the first SRH. In our example, SRH[2] represents S1, SRH[1] represents S2 and SRH[0] represents S3.

3. OAM Building Blocks

This section defines the various building blocks for implementing OAM mechanisms in SRv6 networks.

3.1. O-flag in Segment Routing Header

[I-D.ietf-6man-segment-routing-header] describes the Segment Routing Header (SRH) and how SR capable nodes use it. The SRH contains an 8-bit "Flags" field [I-D.draft-ietf-6man-segment-routing-header]. This document defines the following bit in the SRH.Flags to carry the O-flag:

```

      0 1 2 3 4 5 6 7
    +--+--+--+--+--+--+
    |  |  |O|  |  |  |
    +--+--+--+--+--+--+

```

Where:

O-flag: OAM flag. When set, it indicates that this packet is an operations and management (OAM) packet. This document defines the usage of the O-flag in the SRH.Flags.

The document does not define any other flag in the SRH.Flags and meaning and processing of any other bit in SRH.Flags is outside of the scope of this document.

3.1.1. O-flag Processing

Implementation of the O-flag is OPTIONAL. A node MAY ignore SRH.Flags.O-flag. It is also possible that a node is capable of supporting the O-bit but based on a local decision it MAY ignore it during processing on some local SIDs. If a node does not support the O-flag, then upon reception it simply ignores it. If a node supports the O-flag, it can optionally advertise its potential via node capability advertisement in IGP [I-D.ietf-isis-srv6-extensions] and BGP-LS [I-D.ietf-idr-bgpls-srv6-ext].

The SRH.Flags.O-flag implements the "punt a timestamped copy and forward" behavior.

When N receives a packet whose IPv6 DA is S and S is a local SID, N executes the following pseudo-code, before the execution of the local SID S.

1. IF SRH.Flags.O-flag is one and local configuration permits THEN
 - a. Make a copy of the packet.
 - b. Send the copied packet, along with an accurate timestamp to the OAM process. ;; Refl
- Refl: An implementation SHOULD copy and record the timestamp as soon as possible during packet processing. Timestamp is not carried in the packet forwarded to the next hop.

3.2. OAM Segments

OAM Segment IDs (SIDs) is another component of the SRv6 OAM building Blocks. This document defines a couple of OAM SIDs.

3.3. End.OP: OAM Endpoint with Punt

Many scenarios require punting of SRv6 OAM packets at the desired nodes in the network. The "OAM Endpoint with Punt" function (End.OP for short) represents a particular OAM function to implement the punt behavior for an OAM packet. It is described using the pseudocode as follows:

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

1. Send the packet to the OAM process

Please note that in an SRH containing END.OP SID, it is RECOMMENDED to set the SRH.Flags.O-flag = 0.

3.4. End.OTP: OAM Endpoint with Timestamp and Punt

Scenarios demanding performance management of an SR policy/ path requires hardware timestamping before hardware punts the packet to the software for OAM processing. The "OAM Endpoint with Timestamp and Punt" function (End.OTP for short) represents an OAM SID function to implement the timestamp and punt behavior for an OAM packet. It is described using the pseudocode as follows:

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

1. Timestamp the packet ;; Ref1, Ref2
2. Send the packet, along with an accurate timestamp, to the OAM process.

Ref1: Timestamping SHOULD be done in hardware, as soon as possible during the packet processing.

Ref2: An implementation should not generate further ICMP error during local SID S processing. If local SID S processing requires generation of an ICMP error, the error is generated by the local OAM process.

Please note that in an SRH containing END.OTP SID, it is RECOMMENDED to set the SRH.Flags.O-flag = 0.

3.5. SRH TLV

[I-D.ietf-6man-segment-routing-header] defines TLVs of the Segment Routing Header.

SRH TLV plays an important role in carrying OAM and Performance Management (PM) metadata.

4. OAM Mechanisms

This section describes how OAM mechanisms can be implemented using the OAM building blocks described in the previous section. Additional OAM mechanisms will be added in a future revision of the document.

[RFC4443] describes Internet Control Message Protocol for IPv6 (ICMPv6) that is used by IPv6 devices for network diagnostic and error reporting purposes. As Segment Routing with IPv6 data plane (SRv6) simply adds a new type of Routing Extension Header, existing ICMPv6 ping mechanisms can be used in an SRv6 network. This section

describes the applicability of ICMPv6 in the SRv6 network and how the existing ICMPv6 mechanisms can be used for providing OAM functionality.

The document does not propose any changes to the standard ICMPv6 [RFC4443], [RFC4884] or standard ICMPv4 [RFC792].

4.1. Ping

There is no hardware or software change required for ping operation at the classic IPv6 nodes in an SRv6 network. That includes the classic IPv6 node with ingress, egress or transit roles. Furthermore, no protocol changes are required to the standard ICMPv6 [RFC4443], [RFC4884] or standard ICMPv4 [RFC792]. In other words, existing ICMP ping mechanisms work seamlessly in the SRv6 networks.

The following subsections outline some use cases of the ICMP ping in the SRv6 networks.

4.1.1. Classic Ping

The existing mechanism to ping a remote IP prefix, along the shortest path, continues to work without any modification. The initiator may be an SRv6 node or a classic IPv6 node. Similarly, the egress or transit may be an SRv6 capable node or a classic IPv6 node.

If an SRv6 capable ingress node wants to ping an IPv6 prefix via an arbitrary segment list <S1, S2, S3>, it needs to initiate ICMPv6 ping with an SR header containing the SID list <S1, S2, S3>. This is illustrated using the topology in Figure 1. Assume all the links have IGP metric 10 except both links between node2 and node3, which have IGP metric set to 100. User issues a ping from node N1 to a loopback of node 5, via segment list <B:2:C31, B:4:C52>.

Figure 2 contains sample output for a ping request initiated at node N1 to the loopback address of node N5 via a segment list <B:2:C31, B:4:C52>.

```
> ping A:5:: via segment-list B:2:C31, B:4:C52

Sending 5, 100-byte ICMP Echos to B5::, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0.625
/0.749/0.931 ms
```

Figure 2 A sample ping output at an SRv6 capable node

All transit nodes process the echo request message like any other data packet carrying SR header and hence do not require any change. Similarly, the egress node (IPv6 classic or SRv6 capable) does not require any change to process the ICMPv6 echo request. For example, in the ping example of Figure 2:

- o Node N1 initiates an ICMPv6 ping packet with SRH as follows (A:1::, B:2:C31) (A:5::, B:4:C52, B:2:C31, SL=2, NH = ICMPv6) (ICMPv6 Echo Request).
- o Node N2, which is an SRv6 capable node, performs the standard SRH processing. Specifically, it executes the END.X function (B:2:C31) and forwards the packet on link3 to N3.
- o Node N3, which is a classic IPv6 node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on DA B:4:C52 in the IPv6 header.
- o Node N4, which is an SRv6 capable node, performs the standard SRH processing. Specifically, it observes the END.X function (B:4:C52) with PSP (Penultimate Segment POP) on the echo request packet and removes the SRH and forwards the packet across link10 to N5.
- o The echo request packet at N5 arrives as an IPv6 packet without an SRH. Node N5, which is a classic IPv6 node, performs the standard IPv6/ ICMPv6 processing on the echo request and responds, accordingly.

4.1.2. Pinging a SID Function

The classic ping described in the previous section cannot be used to ping a remote SID function, as explained using an example in the following.

Consider the case where the user wants to ping the remote SID function B:4:C52, via B:2:C31, from node N1. Node N1 constructs the ping packet (A:1::, B:2:C31) (B:4:C52, B:2:C31, SL=1; NH=ICMPv6) (ICMPv6 Echo Request). The ping fails because the node N4 receives the ICMPv6 echo request with DA set to B:4:C52 but the next header is ICMPv6, instead of SRH. To solve this problem, the initiator needs to mark the ICMPv6 echo request as an OAM packet.

The OAM packets are identified either by setting the O-flag in SRH or by inserting the END.OP/ END.OTP SIDs at an appropriate place in the SRH. The following illustration uses END.OTP SID but the procedures are equally applicable to the END.OP SID.

In an SRv6 network, the user can exercise two flavors of the ping: end-to-end ping or segment-by-segment ping, as outlined in the following subsection.

4.1.2.1. End-to-end ping using END.OP/ END.OTP

The end-to-end ping illustration uses the END.OTP SID but the procedures are equally applicable to the END.OP SID.

Consider the same example where the user wants to ping a remote SID function B:4:C52, via B:2:C31, from node N1. To force a punt of the ICMPv6 echo request at the node N4, node N1 inserts the END.OTP SID just before the target SID B:4:C52 in the SRH. The ICMPv6 echo request is processed at the individual nodes along the path as follows:

- o Node N1 initiates an ICMPv6 ping packet with SRH as follows (A:1::, B:2:C31) (B:4:C52, B:4:OTP, B:2:C31; SL=2; NH=ICMPv6) (ICMPv6 Echo Request).
- o Node N2, which is an SRv6 capable node, performs the standard SRH processing. Specifically, it executes the END.X function (B:2:C31) on the echo request packet.
- o Node N3 receives the packet as follows (A:1::, B:4:OTP) (B:4:C52, B:4:OTP, B:2:C31 ; SL=1; NH=ICMPv6) (ICMPv6 Echo Request). Node N3, which is a classic IPv6 node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on DA B:4:OTP in the IPv6 header.
- o When node N4 receives the packet (A:1::, B:4:OTP) (B:4:C52, B:4:OTP, B:2:C31 ; SL=1; NH=ICMPv6) (ICMPv6 Echo Request), it processes the END.OTP SID, as described in the pseudocode in Section 3. The packet gets punted to the ICMPv6 process for processing. The ICMPv6 process checks if the next SID in SRH (the target SID B:4:C52) is locally programmed.
- o If the target SID is not locally programmed, N4 responds with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "SID not locally implemented (TBA)"); otherwise a success is returned.

4.1.2.2. Segment-by-segment ping using O-flag (Proof of Transit)

Consider the same example where the user wants to ping a remote SID function B:4:C52, via B:2:C31, from node N1. However, in this ping, the node N1 wants to get a response from each segment node in the SRH as a "proof of transit". In other words, in the segment-by-segment ping case, the node N1 expects a response from node N2 and node N4

for their respective local SID function. When a response to O-bit is desired from the last SID in a SID-list, it is the responsibility of the ingress node to use USP as the last SID. E.g., in this example, the target SID B:4:C52 is a USP SID.

To force a punt of the ICMPv6 echo request at node N2 and node N4, node N1 sets the O-flag in SRH. The ICMPv6 echo request is processed at the individual nodes along the path as follows:

- o Node N1 initiates an ICMPv6 ping packet with SRH as follows (A:1::, B:2:C31) (B:4:C52, B:2:C31; SL=1, Flags.O=1; NH=ICMPv6) (ICMPv6 Echo Request).
- o When node N2 receives the packet (A:1::, B:2:C31) (B:4:C52, B:2:C31; SL=1, Flags.O=1; NH=ICMPv6) (ICMPv6 Echo Request) packet, it processes the O-flag in SRH, as described in the pseudocode in Section 3. A time-stamped copy of the packet gets punted to the ICMPv6 process for processing. Node N2 continues to apply the B:2:C31 SID function on the original packet and forwards it, accordingly. As B:4:C52 is a USP SID, N2 does not remove the SRH. The ICMPv6 process at node N2 checks if its local SID (B:2:C31) is locally programmed or not and responds to the ICMPv6 Echo Request.
- o If the target SID is not locally programmed, N4 responds with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "SID not locally implemented (TBA)"); otherwise a success is returned. Please note that, as mentioned in Section 3, if node N2 does not support the O-flag, it simply ignores it and process the local SID, B:2:C31.
- o Node N3, which is a classic IPv6 node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on DA B:4:C52 in the IPv6 header.
- o When node N4 receives the packet (A:1::, B:4:C52) (B:4:C52, B:2:C31; SL=0, Flags.O=1; NH=ICMPv6) (ICMPv6 Echo Request), it processes the O-flag in SRH, as described in the pseudocode in Section 3. A time-stamped copy of the packet gets punted to the ICMPv6 process for processing. The ICMPv6 process at node N4 checks if its local SID (B:2:C31) is locally programmed or not and responds to the ICMPv6 Echo Request. If the target SID is not locally programmed, N4 responds with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "SID not locally implemented (TBA)"); otherwise a success is returned.

Support for O-flag is part of node capability advertisement. That enables node N1 to know which segment nodes are capable of responding to the ICMPv6 echo request. Node N1 processes the echo responses and presents data to the user, accordingly.

Please note that segment-by-segment ping can be used to address proof of transit use-case.

4.1.3. Error Reporting

Any IPv6 node can use ICMPv6 control messages to report packet processing errors to the host that originated the datagram packet. To name a few such scenarios:

- o If the router receives an undeliverable IP datagram, or
- o If the router receives a packet with a Hop Limit of zero, or
- o If the router receives a packet such that if the router decrements the packet's Hop Limit it becomes zero, or
- o If the router receives a packet with problem with a field in the IPv6 header or the extension headers such that it cannot complete processing the packet, or
- o If the router cannot forward a packet because the packet is larger than the MTU of the outgoing link.

In the scenarios listed above, the ICMPv6 response also contains the IP header, IP extension headers and leading payload octets of the "original datagram" to which the ICMPv6 message is a response. Specifically, the "Destination Unreachable Message", "Time Exceeded Message", "Packet Too Big Message" and "Parameter Problem Message" ICMPV6 messages can contain as much of the invoking packet as possible without the ICMPv6 packet exceeding the minimum IPv6 MTU [RFC4443], [RFC4884]. In an SRv6 network, the copy of the invoking packet contains the SR header. The packet originator can use this information for diagnostic purposes. For example, traceroute can use this information as detailed in the following subsection.

4.2. Traceroute

There is no hardware or software change required for traceroute operation at the classic IPv6 nodes in an SRv6 network. That includes the classic IPv6 node with ingress, egress or transit roles. Furthermore, no protocol changes are required to the standard traceroute operations. In other words, existing traceroute mechanisms work seamlessly in the SRv6 networks.

The following subsections outline some use cases of the traceroute in the SRv6 networks.

4.2.1. Classic Traceroute

The existing mechanism to traceroute a remote IP prefix, along the shortest path, continues to work without any modification. The initiator may be an SRv6 node or a classic IPv6 node. Similarly, the egress or transit may be an SRv6 node or a classic IPv6 node.

If an SRv6 capable ingress node wants to traceroute to IPv6 prefix via an arbitrary segment list <S1, S2, S3>, it needs to initiate traceroute probe with an SR header containing the SID list <S1, S2, S3>. That is illustrated using the topology in Figure 1. Assume all the links have IGP metric 10 except both links between node2 and node3, which have IGP metric set to 100. User issues a traceroute from node N1 to a loopback of node 5, via segment list <B:2:C31, B:4:C52>. Figure 3 contains sample output for the traceroute request.

```
> traceroute A:5:: via segment-list B:2:C31, B:4:C52

Tracing the route to B5::
 1  2001:DB8:1:2:21:: 0.512 msec 0.425 msec 0.374 msec
    SRH: (A:5::, B:4:C52, B:2:C31, SL=2)
 2  2001:DB8:2:3:31:: 0.721 msec 0.810 msec 0.795 msec
    SRH: (A:5::, B:4:C52, B:2:C31, SL=1)
 3  2001:DB8:3:4::41:: 0.921 msec 0.816 msec 0.759 msec
    SRH: (A:5::, B:4:C52, B:2:C31, SL=1)
 4  2001:DB8:4:5::52:: 0.879 msec 0.916 msec 1.024 msec
```

Figure 3 A sample traceroute output at an SRv6 capable node

Please note that information for hop2 is returned by N3, which is a classic IPv6 node. Nonetheless, the ingress node is able to display SR header contents as the packet travels through the IPv6 classic node. This is because the "Time Exceeded Message" ICMPv6 message can contain as much of the invoking packet as possible without the ICMPv6 packet exceeding the minimum IPv6 MTU [RFC4443]. The SR header is also included in these ICMPv6 messages initiated by the classic IPv6 transit nodes that are not running SRv6 software. Specifically, a node generating ICMPv6 message containing a copy of the invoking packet does not need to understand the extension header(s) in the invoking packet.

The segment list information returned for hop1 is returned by N2, which is an SRv6 capable node. Just like for hop2, the ingress node is able to display SR header contents for hop1.

There is no difference in processing of the traceroute probe at an IPv6 classic node and an SRv6 capable node. Similarly, both IPv6 classic and SRv6 capable nodes may use the address of the interface on which probe was received as the source address in the ICMPv6 response. ICMP extensions defined in [RFC5837] can be used to also display information about the IP interface through which the datagram would have been forwarded had it been forwardable, and the IP next hop to which the datagram would have been forwarded, the IP interface upon which a datagram arrived, the sub-IP component of an IP interface upon which a datagram arrived.

The information about the IP address of the incoming interface on which the traceroute probe was received by the reporting node is very useful. This information can also be used to verify if SID functions B:2:C31 and B:4:C52 are executed correctly by N2 and N4, respectively. Specifically, the information displayed for hop2 contains the incoming interface address 2001:DB8:2:3:31:: at N3. This matches with the expected interface bound to END.X function B:2:C31 (link3). Similarly, the information displayed for hop5 contains the incoming interface address 2001:DB8:4:5::52:: at N5. This matches with the expected interface bound to the END.X function B:4:C52 (link10).

4.2.2. Traceroute to a SID Function

The classic traceroute described in the previous section cannot be used to traceroute a remote SID function, as explained using an example in the following.

Consider the case where the user wants to traceroute the remote SID function B:4:C52, via B:2:C31, from node N1. The trace route fails at N4. This is because the node N4 trace route probe where next header is UDP or ICMPv6, instead of SRH (even though the hop limit is set to 1). To solve this problem, the initiator needs to mark the ICMPv6 echo request as an OAM packet.

The OAM packets are identified either by setting the O-flag in SRH or by inserting the END.OP or END.OTP SID at an appropriate place in the SRH.

In an SRv6 network, the user can exercise two flavors of the traceroute: hop-by-hop traceroute or overlay traceroute.

- o In hop-by-hop traceroute, user gets responses from all nodes including classic IPv6 transit nodes, SRv6 capable transit nodes as well as SRv6 capable segment endpoints. E.g., consider the example where the user wants to traceroute to a remote SID function B:4:C52, via B:2:C31, from node N1. The traceroute

output will also display information about node3, which is a transit (underlay) node.

- o The overlay traceroute, on the other hand, does not trace the underlay nodes. In other words, the overlay traceroute only displays the nodes that acts as SRv6 segments along the route. I.e., in the example where the user wants to traceroute to a remote SID function B:4:C52, via B:2:C31, from node N1, the overlay traceroute would only display the traceroute information from node N2 and node N4; it will not display information from node 3.

4.2.2.1. Hop-by-hop traceroute using END.OP/ END.OTP

In this section, hop-by-hop traceroute to a SID function is exemplified using UDP probes. However, the procedure is equally applicable to other implementation of traceroute mechanism. Furthermore, the illustration uses the END.OTP SID but the procedures are equally applicable to the END.OP SID.

Consider the same example where the user wants to traceroute to a remote SID function B:4:C52, via B:2:C31, from node N1. To force a punt of the traceroute probe only at the node N4, node N1 inserts the END.OTP SID just before the target SID B:4:C52 in the SRH. The traceroute probe is processed at the individual nodes along the path as follows:

- o Node N1 initiates a traceroute probe packet with a monotonically increasing value of hop count and SRH as follows (A:1::, B:2:C31) (B:4:C52, B:4:OTP, B:2:C31; SL=2; NH=UDP) (Traceroute probe).
- o When node N2 receives the packet with hop-count = 1, it processes the hop count expiry. Specifically, the node N2 responds with the ICMPv6 message (Type: "Time Exceeded", Code: "Time to Live exceeded in Transit").
- o When Node N2 receives the packet with hop-count > 1, it performs the standard SRH processing. Specifically, it executes the END.X function (B:2:C31) on the traceroute probe.
- o When node N3, which is a classic IPv6 node, receives the packet (A:1::, B:4:OTP) (B:4:C52, B:4:OTP, B:2:C31 ; HC=1, SL=1; NH=UDP) (Traceroute probe) with hop-count = 1, it processes the hop count expiry. Specifically, the node N3 responds with the ICMPv6 message (Type: "Time Exceeded", Code: "Time to Live exceeded in Transit").

- o When node N3, which is a classic IPv6 node, receives the packet with hop-count > 1, it performs the standard IPv6 processing. Specifically, it forwards the traceroute probe based on DA B:4:OTP in the IPv6 header.
- o When node N4 receives the packet (A:1::, B:4:OTP) (B:4:C52, B:4:OTP, B:2:C31 ; SL=1; HC=1, NH=UDP) (Traceroute probe), it processes the END.OTP SID, as described in the pseudocode in Section 3. The packet gets punted to the traceroute process for processing. The traceroute process checks if the next SID in SRH (the target SID B:4:C52) is locally programmed. If the target SID B:4:C52 is locally programmed, node N4 responds with the ICMPv6 message (Type: Destination unreachable, Code: Port Unreachable). If the target SID B:4:C52 is not a local SID, node N4 silently drops the traceroute probe.

Figure 4 displays a sample traceroute output for this example.

```
> traceroute srv6 B:4:C52 via segment-list B:2:C31

Tracing the route to SID function B:4:C52
 1  2001:DB8:1:2:21 0.512 msec 0.425 msec 0.374 msec
    SRH: (B:4:C52, B:4:OTP, B:2:C31; SL=2)
 2  2001:DB8:2:3:31 0.721 msec 0.810 msec 0.795 msec
    SRH: (B:4:C52, B:4:OTP, B:2:C31; SL=1)
 3  2001:DB8:3:4::41 0.921 msec 0.816 msec 0.759 msec
    SRH: (B:4:C52, B:4:OTP, B:2:C31; SL=1)
```

Figure 4 A sample output for hop-by-hop traceroute to a SID function

4.2.2.2. Tracing SRv6 Overlay

The overlay traceroute does not trace the underlay nodes, i.e., only displays the nodes that acts as SRv6 segments along the path. This is achieved by setting the SRH.Flags.0 bit.

In this section, overlay traceroute to a SID function is exemplified using UDP probes. However, the procedure is equally applicable to other implementation of traceroute mechanism.

Consider the same example where the user wants to traceroute to a remote SID function B:4:C52, via B:2:C31, from node N1.

- o Node N1 initiates a traceroute probe with SRH as follows (A:1::, B:2:C31) (B:4:C52, B:2:C31; HC=64, SL=1, Flags.0=1; NH=UDP) (Traceroute Probe). Please note that the hop-count is set

to 64 to skip the underlay nodes from tracing. The O-flag in SRH is set to make the overlay nodes (nodes processing the SRH) respond.

- o When node N2 receives the packet (A:1::, B:2:C31) (B:4:C52, B:2:C31; SL=1, HC=64, Flags.O=1; NH=UDP) (Traceroute Probe), it processes the O-flag in SRH, as described in the pseudocode in Section 3. A time-stamped copy of the packet gets punted to the traceroute process for processing. Node N2 continues to apply the B:2:C31 SID function on the original packet and forwards it, accordingly. The traceroute process at node N2 checks if its local SID (B:2:C31) is locally programmed. If the SID is not locally programmed, it silently drops the packet. Otherwise, it performs the egress check by looking at the SL value in SRH.
- o As SL is not equal to zero (i.e., it's not egress node), node N2 responses with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "O-flag punt at Transit (TBA)"). Please note that, as mentioned in Section 3, if node N2 does not support the O-flag, it simply ignores it and processes the local SID, B:2:C31.
- o When node N3 receives the packet (A:1::, B:4:C52) (B:4:C52, B:2:C31; SL=0, HC=63, Flags.O=1; NH=UDP) (Traceroute Probe), performs the standard IPv6 processing. Specifically, it forwards the traceroute probe based on DA B:4:C52 in the IPv6 header. Please note that there is no hop-count expiration at the transit nodes.
- o When node N4 receives the packet (A:1::, B:4:C52) (B:4:C52, B:2:C31; SL=0, HC=62, Flags.O=1; NH=UDP) (Traceroute Probe), it processes the O-flag in SRH, as described in the pseudocode in Section 3. A time-stamped copy of the packet gets punted to the traceroute process for processing. The traceroute process at node N4 checks if its local SID (B:2:C31) is locally programmed. If the SID is not locally programmed, it silently drops the packet. Otherwise, it performs the egress check by looking at the SL value in SRH. As SL is equal to zero (i.e., N4 is the egress node), node N4 tries to consume the UDP probe. As UDP probe is set to access an invalid port, the node N4 responses with the ICMPv6 message (Type: Destination unreachable, Code: Port Unreachable)

Figure 5 displays a sample overlay traceroute output for this example. Please note that the underlay node N3 does not appear in the output.

```
Tracing the route to SID function B:4:C52
 1  2001:DB8:1:2:21:: 0.512 msec 0.425 msec 0.374 msec
    SRH: (B:4:C52, B:4:OTP, B:2:C31; SL=2)
 2  2001:DB8:3:4::41:: 0.921 msec 0.816 msec 0.759 msec
    SRH: (B:4:C52, B:4:OTP, B:2:C31; SL=1)
```

Figure 5 A sample output for overlay traceroute to a SID function

4.3. Monitoring of SRv6 Paths

In the recent past, network operators are interested in performing network OAM functions in a centralized manner. Various data models like YANG are available to collect data from the network and manage it from a centralized entity.

SR technology enables a centralized OAM entity to perform path monitoring from centralized OAM entity without control plane intervention on monitored nodes. [RFC 8403] describes such a centralized OAM mechanism. Specifically, the draft describes a procedure that can be used to perform path continuity check between any nodes within an SR domain from a centralized monitoring system, with minimal or no control plane intervene on the nodes. However, the draft focuses on SR networks with MPLS data plane. The same concept applies to the SRv6 networks. This document describes how the concept can be used to perform path monitoring in an SRv6 network. This document describes how the concept can be used to perform path monitoring in an SRv6 network as follows.

In the above reference topology, N100 is the centralized monitoring system implementing an END function B:100:1::.. In order to verify a segment list <B:2:C31, B:4:C52>, N100 generates a probe packet with SRH set to (B:100:1::, B:4:C52, B:2:C31, SL=2). The controller routes the probe packet towards the first segment, which is B:2:C31. N2 performs the standard SRH processing and forward it over link3 with the DA of IPv6 packet set to B:4:C52. N4 also performs the normal SRH processing and forward it over link10 with the DA of IPv6 packet set to B:100:1::.. This makes the probe loops back to the centralized monitoring system.

In the reference topology in Figure 1, N100 uses an IGP protocol like OSPF or ISIS to get the topology view within the IGP domain. N100 can also use BGP-LS to get the complete view of an inter-domain topology. In other words, the controller leverages the visibility of the topology to monitor the paths between the various endpoints without control plane intervention required at the monitored nodes.

5. Security Considerations

This document does not define any new protocol extensions and relies on existing procedures defined for ICMP. This document does not impose any additional security challenges to be considered beyond security considerations described in [RFC4884], [RFC4443], [RFC792], RFCs that updates these RFCs, [I-D.ietf-6man-segment-routing-header] and [I-D.ietf-spring-srv6-network-programming].

6. IANA Considerations

6.1. ICMPv6 type Numbers RegistrySEC

This document defines one ICMPv6 Message, a type that has been allocated from the "ICMPv6 'type' Numbers" registry of [RFC4443]. Specifically, it requests to add the following to the "ICMPv6 Type Numbers" registry:

TBA (suggested value: 162) SRv6 OAM Message.

The document also requests the creation of a new IANA registry to the "ICMPv6 'Code' Fields" against the "ICMPv6 Type Numbers TBA - SRv6 OAM Message" with the following codes:

Code	Name	Reference
0	No Error	This document
1	SID is not locally implemented	This document
2	O-flag punt at Transit	This document

6.2. SRv6 OAM Endpoint Types

This I-D requests to IANA to allocate, within the "SRv6 Endpoint Behaviors Registry" sub-registry belonging to the top-level "Segment-routing with IPv6 dataplane (SRv6) Parameters" registry [I-D.ietf-spring-srv6-network-programming], the following allocations:

Value (Suggested Value)	Endpoint Behavior	Reference
TBA (40)	End.OP	[This.ID]
TBA (41)	End.OTP	[This.ID]

7. Acknowledgements

The authors would like to thank Gaurav Naik for his review comments.

8. Contributors

The following people have contributed to this document:

Robert Raszuk
Bloomberg LP
Email: robert@raszuk.net

John Leddy
Individual
Email: john@leddy.net

Gaurav Dawra
LinkedIn
Email: gdawra.ietf@gmail.com

Bart Peirens
Proximus
Email: bart.peirens@proximus.com

Nagendra Kumar
Cisco Systems, Inc.
Email: naikumar@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
Email: cpignata@cisco.com

Rakesh Gandhi
Cisco Systems, Inc.
Canada
Email: rgandhi@cisco.com

Frank Brockners
Cisco Systems, Inc.
Germany
Email: fbrockne@cisco.com

Darren Dukes
Cisco Systems, Inc.
Email: ddukes@cisco.com

Cheng Li
Huawei
Email: chengli13@huawei.com

Faisal Iqbal
Individual
Email: faisal.ietf@gmail.com

9. References

9.1. 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-21 (work in progress), June 2019.
- [I-D.ietf-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-01 (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>>.

9.2. Informative References

- [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <<https://www.rfc-editor.org/info/rfc792>>.

- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/info/rfc4443>>.
- [RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <<https://www.rfc-editor.org/info/rfc4884>>.
- [RFC5837] Atlas, A., Ed., Bonica, R., Ed., Pignataro, C., Ed., Shen, N., and JR. Rivers, "Extending ICMP for Interface and Next-Hop Identification", RFC 5837, DOI 10.17487/RFC5837, April 2010, <<https://www.rfc-editor.org/info/rfc5837>>.
- [RFC8403] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC 8403, DOI 10.17487/RFC8403, July 2018, <<https://www.rfc-editor.org/info/rfc8403>>.

Authors' Addresses

Zafar Ali
Cisco Systems

Email: zali@cisco.com

Clarence Filsfils
Cisco Systems

Email: cfilsfil@cisco.com

Satoru Matsushima
Softbank

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

Daniel Voyer
Bell Canada

Email: daniel.voyer@bell.ca

Mach Chen
Huawei

Email: mach.chen@huawei.com

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2020

Z. Ali
R. Gandhi
C. Filsfils
F. Brockners
N. Nainar
C. Pignataro
Cisco Systems, Inc.
C. Li
M. Chen
Huawei
G. Dawra
LinkedIn
July 8, 2019

Segment Routing Header encapsulation for In-situ OAM Data
draft-ali-spring-ioam-srv6-01

Abstract

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records operational and telemetry information in the data packet while the packet traverses a path between two points in the network. This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header.

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 <http://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."

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions	3
2.1. Requirement Language	3
2.2. Abbreviations	3
3. OAM Metadata Piggybacked in Data Packets	4
3.1 IOAM Data Field Encapsulation in SRH	4
4. Procedure	5
4.1. Ingress Node	5
4.2. SR Segment Endpoint Node	5
4.3. Egress Node	6
5. IANA Considerations	6
6. Security Considerations	6
7. Acknowledgements	6
8. References	7
8.1. Normative References	7
8.2. Informative References	7
Authors' Addresses	8

1. Introduction

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within probe packets specifically dedicated to OAM.

This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header [I-D.6man-segment-routing-header].

The IOAM data fields carried are defined in [I-D.ietf-ippm-ioam-data], and can be used for various use-cases including Performance Measurement (PM) and Proof-of-Transit (PoT).

2. Conventions

2.1. Requirement Language

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

2.2. Abbreviations

Abbreviations used in this document:

IOAM In-situ Operations, Administration, and Maintenance

OAM Operations, Administration, and Maintenance

PM Performance Measurement

PoT Proof-of-Transit

SR Segment Routing

SRH SRv6 Header

SRv6 Segment Routing with IPv6 Data plane

3. OAM Metadata Piggybacked in Data Packets

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). This section describes IOAM functionality in SRv6 network.

The IOAM data is carried in SRH.TLV. This enables the IOAM mechanism to build on the network programmability capability of SRv6. The ability for an SRv6 endpoint to determine whether to process or ignore some specific SRH TLVs is based on the SID function. This enables collection of the IOAM information from the intermediate endpoint nodes of choice. The nodes that are not capable of supporting the IOAM functionality does not have to look or process SRH TLV (i.e., such nodes can simply ignore the SRH IOAM TLV).

3.1 IOAM Data Field Encapsulation in SRH

The SRv6 encapsulation header (SRH) is defined in [I-D.ietf-6man-segment-routing-header]. IOAM data fields are carried in the SRH, using a single pre-allocated SRH TLV. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as sub-TLVs.

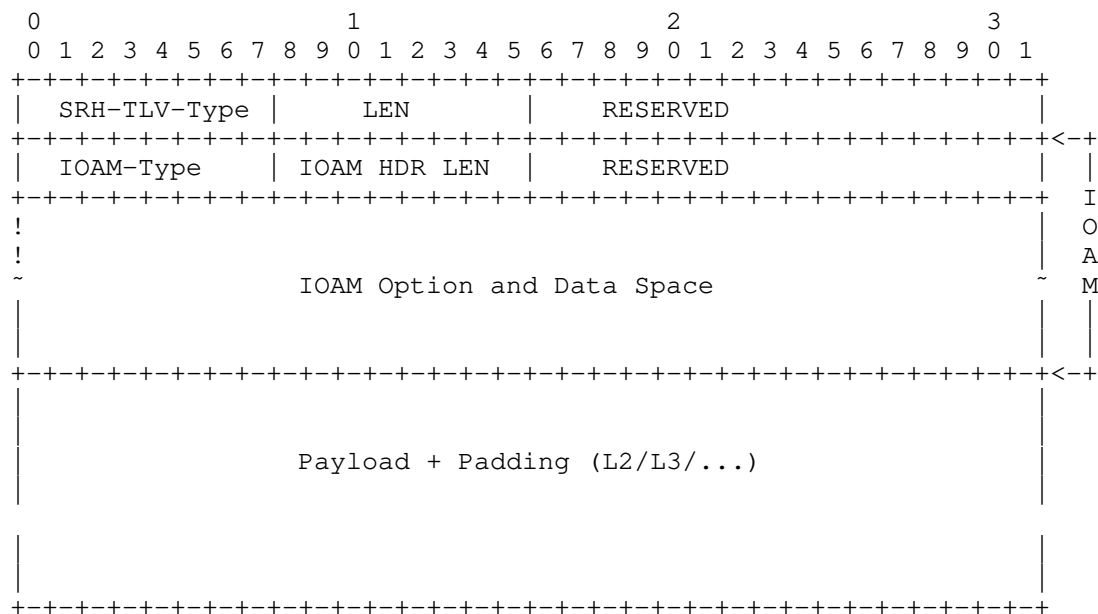


Figure 1: IOAM data encapsulation in SRH

SRH-TLV-Type: IOAM TLV Type for SRH is defined as TBA1.

The fields related to the encapsulation of IOAM data fields in the SRH are defined as follows:

IOAM-Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR LEN: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

RESERVED: 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.

IOAM Option and Data Space: IOAM option header and data is present as defined by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

4. Procedure

This section summarizes the procedure for IOAM data encapsulation in SRv6 SRH. The SR nodes implementing the IOAM functionality follows the MTU and other considerations outlined in [I-D.6man-extension-header-insertion].

4.1. Ingress Node

As part of the SRH encapsulation, the ingress node of an SR domain or an SR Policy [I-D.ietf-spring-segment-routing-policy] MAY add the IOAM TLV in the SRH of the data packet. If an ingress node supports IOAM functionality and, based on a local configuration, wants to collect IOAM data, it adds IOAM TLV in the SRH. Based on the size of the segment list (SL), the ingress node preallocates space in the IOAM TLV.

If IOAM data from the last node in the segment-list (Egress node) is desired, the ingress uses an Ultimate Segment Pop (USP) SID advertised by the Egress node.

The ingress node may also insert the IOAM data about the local information in the IOAM TLV in the SRH at index 0 of the preallocated IOAM TLV.

4.2. Intermediate SR Segment Endpoint Node

The SR segment endpoint node is any node receiving an IPv6 packet where the destination address of that packet is a local SID. As part of the SR Header processing as described in [I-D.ietf-6man-segment-routing-header] and [I-D.ietf-spring-srv6-network-programming], the SR Segment Endpoint node performs the following IOAM operations.

If an intermediate SR segment endpoint node is not capable of processing IOAM TLV, it simply ignores it. I.e., it does not have to look or process SRH TLV.

If an intermediate SR segment endpoint node is capable of processing IOAM TLV and the local SID supports IOAM data recording, it checks if any SRH TLV is present in the packet using procedures defined in [I-D.ietf-6man-segment-routing-header]. If the node finds IOAM TLV in the SRH it finds the local index at which it is expected to record the IOAM data. The local index is found using the SRH.SL field. The node records the IOAM data at the desired preallocated space.

4.3. Egress Node

The Egress node is the last node in the segment-list of the SRH. When IOAM data from the Egress node is desired, a USP SID advertised by the Egress node is used by the Ingress node.

The processing of IOAM TLV at the Egress node is similar to the processing of IOAM TLV at the SR Segment Endpoint Node. The only difference is that the Egress node may telemeter the IOAM data to an external entity.

5. IANA Considerations

IANA is requested to allocate a mutable SRH TLV Type for IOAM TLV data fields under registry name "Segment Routing Header TLVs" requested by [I-D.6man-segment-routing-header].

SRH TLV Type	Description	Reference
TBA1 Greater than 128	TLV for IOAM Data Fields	This document

6. Security Considerations

The security considerations of SRv6 are discussed in [I-D.spring-srv6-network-programming] and [I-D.6man-segment-routing-header], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

7. Acknowledgements

The authors would like to thank Shwetha Bhandari and Vengada Prasad Govindan for the discussions on IOAM.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.
- [I-D.spring-srv6-network-programming] Filsfils, C. et al. "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming, work in progress.
- [I-D.6man-segment-routing-header] Previdi, S., Filsfils, C. et al, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header, work in progress.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and Bernier, D., "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data, work in progress.
- [I-D.spring-segment-routing-policy] Filsfils, C., et al., "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy, work in progress.

8.2. Informative References

- [I-D.6man-extension-header-insertion] D. Voyer, et al., "Insertion of IPv6 Segment Routing Headers in a Controlled Domain", draft-voyer-6man-extension-header-insertion, work in progress.

Authors' Addresses

Zafar Ali
Cisco Systems, Inc.

Email: zali@cisco.com

Rakesh Gandhi
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Frank Brockners
Cisco Systems, Inc.
Germany

Email: fbrockne@cisco.com

Nagendra Kumar Nainar
Cisco Systems, Inc.

Email: naikumar@cisco.com

Carlos Pignataro
Cisco Systems, Inc.

Email: cpignata@cisco.com

Cheng Li
Huawei

Email: chengli13@huawei.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Gaurav Dawra
LinkedIn

Email: gdawra.ietf@gmail.com

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 12, 2022

Z. Ali
R. Gandhi
C. Filsfils
F. Brockners
N. Nainar
C. Pignataro
Cisco Systems, Inc.
C. Li
M. Chen
Huawei
G. Dawra
LinkedIn
January 12, 2022

Segment Routing Header encapsulation for In-situ OAM Data
draft-ali-spring-ioam-srv6-05

Abstract

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records operational and telemetry information in the data packet while the packet traverses a path between two points in the network. This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header.

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 July 12, 2022.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions	3
2.1. Requirement Language	3
2.2. Abbreviations	3
3. OAM Metadata Piggybacked in Data Packets	4
3.1 IOAM Data Field Encapsulation in SRH	4
4. Procedure	5
4.1. Ingress Node	5
4.2. SR Segment Endpoint Node	5
4.3. Egress Node	6
5. IANA Considerations	6
6. Security Considerations	6
7. Acknowledgements	6
8. References	7
8.1. Normative References	7
8.2. Informative References	7
Authors' Addresses	8

1. Introduction

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). IOAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within probe packets specifically dedicated to OAM.

This document defines how IOAM data fields are transported as part of the Segment Routing with IPv6 data plane (SRv6) header [I-D.6man-segment-routing-header].

The IOAM data fields carried are defined in [I-D.ietf-ippm-ioam-data], and can be used for various use-cases including Performance Measurement (PM) and Proof-of-Transit (PoT).

2. Conventions

2.1. Requirement Language

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

2.2. Abbreviations

Abbreviations used in this document:

IOAM	In-situ Operations, Administration, and Maintenance
OAM	Operations, Administration, and Maintenance
PM	Performance Measurement
PoT	Proof-of-Transit
SR	Segment Routing
SRH	SRv6 Header
SRv6	Segment Routing with IPv6 Data plane

3. OAM Metadata Piggybacked in Data Packets

OAM and PM information from the SR endpoints can be piggybacked in the data packet. The OAM and PM information piggybacking in the data packets is also known as In-situ OAM (IOAM). This section describes iOAM functionality in SRv6 network.

The IOAM data is carried in SRH.TLV. This enables the IOAM mechanism to build on the network programmability capability of SRv6. Specifically, the ability for an SRv6 endpoint to determine whether to process or ignore some specific SRH TLVs is based on the SID function. This enables collection of the IOAM information hardware friendly based on the intermediate endpoint capability. The nodes that are not capable of supporting the IOAM functionality does not have to look or process SRH TLV (i.e., such nodes can simply ignore the SRH IOAM TLV). This also enable collection of IOAM data only from segment endpoint.

3.1 IOAM Data Field Encapsulation in SRH

The SRv6 encapsulation header (SRH) is defined in [I-D.ietf-6man-segment-routing-header]. IOAM data fields are carried in the SRH, using a single pre-allocated SRH TLV. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as sub-TLVs.

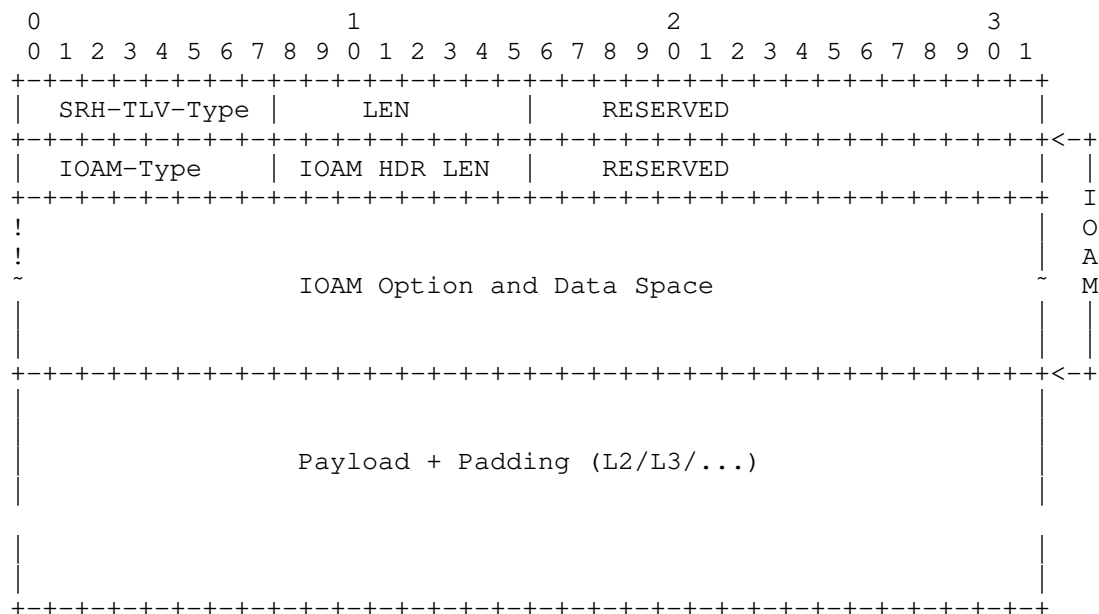


Figure 1: IOAM data encapsulation in SRH

SRH-TLV-Type: IOAM TLV Type for SRH is defined as TBA1.

The fields related to the encapsulation of IOAM data fields in the SRH are defined as follows:

IOAM-Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR LEN: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

RESERVED: 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.

IOAM Option and Data Space: IOAM option header and data is present as defined by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

4. Procedure

This section summarizes the procedure for IOAM data encapsulation in SRv6 SRH. The SR nodes implementing the IOAM functionality follows the MTU and other considerations outlined in [I-D.6man-extension-header-insertion].

4.1. Ingress Node

As part of the SRH encapsulation, the ingress node of an SR domain or an SR Policy [I-D.ietf-spring-segment-routing-policy] MAY add the IOAM TLV in the SRH of the data packet. If an ingress node supports IOAM functionality and, based on a local configuration, wants to collect IOAM data, it adds IOAM TLV in the SRH. Based on the size of the segment list (SL), the ingress node preallocates space in the IOAM TLV.

If IOAM data from the last node in the segment-list (Egress node) is desired, the ingress uses an Ultimate Segment Pop (USP) SID advertised by the Egress node.

The ingress node MAY also insert the IOAM data about the local information in the IOAM TLV in the SRH at index 0 of the preallocated IOAM TLV.

4.2. Intermediate SR Segment Endpoint Node

The SR segment endpoint node is any node receiving an IPv6 packet where the destination address of that packet is a local SID. As part of the SR Header processing as described in [I-D.ietf-6man-segment-routing-header] and [I-D.ietf-spring-srv6-network-programming], the SR Segment Endpoint node performs the following IOAM operations.

If an intermediate SR segment endpoint node is not capable of processing IOAM TLV, it simply ignores it. I.e., it does not have to look or process SRH TLV.

If an intermediate SR segment endpoint node is capable of processing IOAM TLV and the local SID supports IOAM data recording, it checks if any SRH TLV is present in the packet using procedures defined in [I-D.ietf-6man-segment-routing-header]. If the node finds IOAM TLV in the SRH it finds the local index at which it is expected to record the IOAM data. The local index is found using the SRH.SL field. The node records the IOAM data at the desired preallocated space.

4.3. Egress Node

The Egress node is the last node in the segment-list of the SRH. When IOAM data from the Egress node is desired, a USP SID advertised by the Egress node is used by the Ingress node.

The processing of IOAM TLV at the Egress node is similar to the processing of IOAM TLV at the SR Segment Endpoint Node. The only difference is that the Egress node may telemeter the IOAM data to an external entity.

5. IANA Considerations

IANA is requested to allocate a mutable SRH TLV Type for IOAM TLV data fields under registry name "Segment Routing Header TLVs" requested by [I-D.6man-segment-routing-header].

SRH TLV Type	Description	Reference
TBA1 Greater than 128	TLV for IOAM Data Fields	This document

6. Security Considerations

The security considerations of SRv6 are discussed in [I-D.spring-srv6-network-programming] and [I-D.6man-segment-routing-header], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

7. Acknowledgements

The authors would like to thank Shwetha Bhandari and Vengada Prasad Govindan for the discussions on IOAM.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.
- [I-D.spring-srv6-network-programming] Filsfils, C. et al. "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming, work in progress.
- [I-D.6man-segment-routing-header] Previdi, S., Filsfils, C. et al, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header, work in progress.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and Bernier, D., "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data, work in progress.
- [I-D.spring-segment-routing-policy] Filsfils, C., et al., "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy, work in progress.

8.2. Informative References

- [I-D.6man-extension-header-insertion] D. Voyer, et al., "Insertion of IPv6 Segment Routing Headers in a Controlled Domain", draft-voyer-6man-extension-header-insertion, work in progress.

Internet-Draft

In-situ OAM SRv6 encapsulation

Authors' Addresses

Zafar Ali
Cisco Systems, Inc.

Email: zali@cisco.com

Rakesh Gandhi
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Frank Brockners
Cisco Systems, Inc.
Germany

Email: fbrockne@cisco.com

Nagendra Kumar Nainar
Cisco Systems, Inc.

Email: naikumar@cisco.com

Carlos Pignataro
Cisco Systems, Inc.

Email: cpignata@cisco.com

Cheng Li
Huawei

Email: chenglil13@huawei.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Gaurav Dawra
LinkedIn

Email: gdawra.ietf@gmail.com

Routing area
Internet-Draft
Intended status: Standards Track
Expires: August 25, 2019

K. Arora
S. Hegde
Juniper Networks Inc.
S. Aldrin
Google
S. Litkowski
Orange Business Service
M. Durrani
Equinix
February 21, 2019

TTL Procedures for SR-TE Paths in Label Switched Path Traceroute
Mechanisms
draft-arora-mpls-spring-ttl-procedures-srte-paths-01

Abstract

Segment routing supports the creation of explicit paths using adjacency-sids, node-sids, and anycast-sids. The SR-TE paths are built by stacking the labels that represent the nodes and links in the explicit path. A very useful Operations And Maintenance requirement is to be able to trace these paths as defined in [RFC8029]. This document specifies a uniform mechanism to support MPLS traceroute for the SR-TE paths when the nodes in the network are following uniform mode or short-pipe mode [RFC3443].

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 August 25, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Problem with SR-TE Paths	3
2.1. Short Pipe model	4
2.2. Uniform Model	4
3. Detailed Solution For TTL procedures for SR-TE paths	5
3.1. P bit in DDMT TLV	5
3.1.1. Procedures for a PHP router of the tunnel being traced	5
3.1.2. Procedures for a egress router of the tunnel being traced	5
3.1.3. Procedures for a ingress router of the SR-TE path	5
3.1.4. Example describing the solution	6
3.2. Procedures for handling binding-sids	7
3.2.1. Uniform Model	7
3.2.2. Shortpipe Model	8
4. Backward Compatibility	8
5. Security Considerations	8
6. IANA Considerations	8
7. Acknowledgements	9
8. References	9
8.1. Normative References	9
8.2. Informative References	9
Authors' Addresses	10

1. Introduction

The mechanisms to handle TTL procedures for SR-TE paths are described in ([RFC8287]). Section 7.5 of ([RFC8287]) defines the TTL manipulation procedures for short pipe model as below. The LSR

initiating the traceroute SHOULD start by setting the TTL to 1 for the tunnel in the LSP's label stack it wants to start the tracing from, the TTL of all outer labels in the stack to the max value, and the TTL of all the inner labels in the stack to zero. However this mechanism has issues when the constituent tunnels are penultimate-hop-popping(PHP). This document does not propose any change to ([RFC8287]) if the constituent tunnels are ultimate-hop-popping (UHP) or Egress LSR advertizes explicit NULL.

Section 2 describes problems in tracing SR-TE paths and the need for a specialized mechanism to trace SR-TE paths. Section 3 describes the solution applied to mpls echo request/response to trace adjacency-sids and node-sids trace SR-TE path in uniform model and short pipe model.

2. Problem with SR-TE Paths

The topology shown in Figure 1. illustrates a example network topology with SPRING enabled on each node.

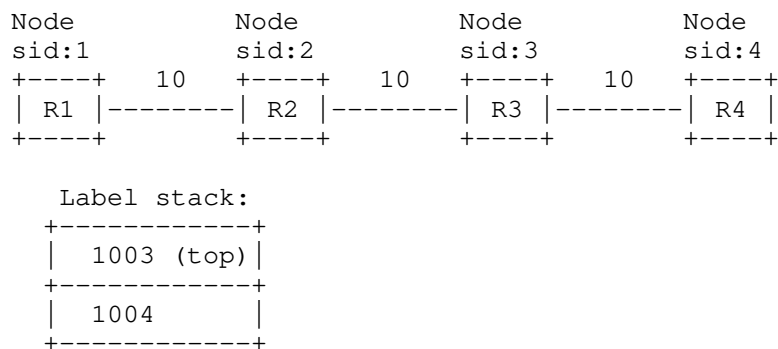


Figure 1: Example topology with SRGB 1000-2000

Consider an explicit path in the topology in Figure 1 from R1->R4 via R1->R2->R3->R4. The label stack to instantiate this path contains two node-sids 1003 and 1004. The 1003 label will take the packet from R1 to R3. The next label in the stack 1004 will take the packet from R3 to the destination R4. consider the mechanism below for the TTL procedures specified in RFC 8287 for short pipe model and uniform model for PHP LSPs.

Notation: ((X,Y),(Z,W)) refers to a label stack whose top label stack entry has the label corresponding to the node-SID of X, with TTL Y, and whose second label stack entry has the label corresponding to the node-SID of Z, with TTL W.

According to the procedure in Section 7.5 of [RFC8287], the LSP traceroute is done as follows in short pipe model and uniform model:

2.1. Short Pipe model

Refer the diagram in Figure 1.

1. Ingress R1 sends mpls LSP Echo Request with label stack of ((1003,1),(1004,0)) to R2.
2. Since R2 receives mpls LSP Echo Request with TTL as 1 for outer most label, R2's local software processes the Lsp traceroute packet and R2 sends an echo reply to R1 with return code as 'transit'.
3. R1 receives the LSP Echo Reply from R2, and then sends next LSP Echo Request with label stack ((1003,2),(1004,0)).
4. R2 forwards packet to R3 as ((1004,0)) (i.e. R2 being PHP, pops the label 1003 and does not propagate TTL)
5. R3 receives a packet with TTL=0 at the top of the stack. Receipt of a packet with TTL=0 may cause R3 to drop the packet or rate limit it.
6. Even if R3's local software processes the packet and validates the FEC for 1003 and sends egress code in echo-reply, the next packet will have ((1003,255), (1004, 1)) which causes TTL to expire again on R3 as the 1003 label is popped at the penultimate.

RFC 8287 suggests that when R1's LSP Echo Request has reached the egress of the outer tunnel, R1 should begin to trace the inner tunnel by sending a LSP Echo Request with label stack ((1003,255),(1004,1)). However, as explained in step 6, the traceroute procedure does not work correctly.

2.2. Uniform Model

1. Ingress R1 sends mpls LSP Echo Request with label stack of ((1003,1),(1004,0)) to R2.
2. Since R2 receives mpls LSP Echo Request with TTL as 1 for outer most label, R2's local software processes the Lsp ping packet and R2 sends an echo reply to R1 with return code as 'transit'.
3. R1 receives the LSP Echo Reply from R2, and then sends next LSP Echo Request with label stack ((1003,2),(1004,0)).

4. It is expected that R2 should propagate the TTL of outer label to inner label before forwarding the packet to R3. However most of the PFEs implementations generally do not increase a label stack entry's TTL when they do TTL propagation. So when (1003,2) is popped, we might still end up with (1004,0) at R3, even if we have TTL propagation configured. Increasing the TTL of a packet is not a good practice as it can result in forwarding loops.

5. R3 receives a packet with TTL=0 at the top of the stack. Receipt of a packet with TTL=0 will cause R3 to drop the packet or rate limit it.

6. Even if R3's local software processes the packet and validates the FEC for 1003 and sends egress code in echo-reply, the next packet will have ((1003,255), (1004, 1)) which causes TTL to expire again on R3 as the 1003 label is popped at the penultimate.

So in either case (uniform model or short pipe model) traceroute may not work for SR-TE paths with PHP Lsps.

3. Detailed Solution For TTL procedures for SR-TE paths

3.1. P bit in DDMT TLV

DS flags has 4 unused bits from position '0' to '3'. This document uses bit '3' in DS flags of downstream mapping TLV.

3.1.1. Procedures for a PHP router of the tunnel being traced

When a LSR receives an echo request it MUST validate the outermost FEC in the echo request. LSR SHOULD set the 'P' bit in the DS flags of downstream mapping TLV if its a PHP router for the outermost FEC. Other cases it should work as explained in [RFC8029] and [RFC8287].

3.1.2. Procedures for a egress router of the tunnel being traced

When a LSR receives an echo request it MUST validate the outermost FEC in the echo request. Egress cases should work as explained in [RFC8029] and [RFC8287].

3.1.3. Procedures for a ingress router of the SR-TE path

When an ingress LSR receives an echo response it MUST behave as defined below depending on the return code in the echo response.

1. When an ingress LSR receives an echo response with return code as 8 (Label switched at stack-depth), Ingress LSR MUST check if the LSR that sent the echo response is PHP for the outermost FEC in the FEC

stack. If the LSR that sent the echo response is PHP for the outermost FEC then while sending next echo request Ingress LSR MUST increase the TTL value of inner label also (if exists) in addition to increasing the TTL value of the tunnel it is tracing. Ingress LSR can detect that LSR that sent the echo response is a PHP router for the outermost FEC, either by looking at 'P' bit set in the DS flags of downstream mapping TLV or if Ingress LSR has received LABEL '3' in the label stack TLV of downstream detailed mapping TLV. For all other cases ingress should work as explained in [RFC8029] and [RFC8287].

2. When an Ingress LSR receives an echo response with return code as 3 (Replying router is an egress for the FEC at stack-depth) for the outermost FEC and this is not the only FEC in the FEC stack, then ingress LSR SHOULD remove the outermost FEC from the FEC stack and send the next traceroute request with the same TTL value for all the labels in the label stack as the previous echo request. This will ensure the egress of the tunnel is visited twice, once as egress for top label and again as a transit for next tunnel.

3.1.4. Example describing the solution

This section provides a detailed description of how PHP router helps ingress in handling TTL procedures for SR-TE paths. Below are the procedures performed by PHP router and ingress router to perform TTL procedure for mpls traceroute for SR-TE paths. Below solution works for both uniform model and short pipe model.

1. Ingress R1 sends mpls LSP Echo Request with label stack of ((1003,1),(1004,0)) to R2.
2. Since R2 receives mpls LSP Echo Request with TTL as 1 for outer most label, R2's local software processes the Lsp ping packet. R2's local software validates the outermost FEC and looking at the FEC R2 knows that its the PHP router for outermost FEC (Node-Sid R3).
3. R2 sets a bit in the DS flags in the DDMT TLV in echo response (P bit, One of the reserved bits).
4. When R1 looks at the echo response from R2 it sees P bit in DDMT TLV.
5. So R1 increments the TTL value of Node-R3 by 1 (make it 2) and TTL value of next element in the label stack also
6. R1 should send the next mpls LSP Echo Request with label stack ((1003,2),(1004,1)).

7. R2 being PHP pops the outermost label from the label stack and forwards the packet to R3 with with label (1004, 1)
8. R3 receives mpls LSP Echo Request with TTL as 1 for outer most label, R3's local software processes the echo request.
9. R3 validates the outermost FEC and sends echo response to R1 with return code as the egress for outermost FEC (Node-Sid R3).
10. When R1 receives echo response with return code as egress, R1 should remove outermost FEC (Node-Sid R3) from the FEC stack and send the next echo request with the same TTL value as the previous one i.e ((1003,2),(1004,1)).
11. Since R3 is the PHP router for FEC (Node-Sid R4) in the label stack. R3 should set 'P' bit in the in the DS flags in the DDMT TLV in echo response with return code as Transit.
12. R1 should send the next mpls LSP Echo Request with label stack ((1003,2),(1004,2)) with FEC Node-Sid-R4 .
13. R2 pops the first label from the label stack and R3 pops the second label from the label stack.
14. R4 receives an unlabelled packet with RA bit set in ip options. R4 delivers the packet to local software for processing.
15. R4's local software validates the ouetmost FEC as 'egress' and sends an echo reply with return code as egress.
17. R1 receives an echo reply with return code as egress for the last FEC in the FEC stack TLV and completes the traceroute.

3.2. Procedures for handling binding-sids

Inorder to provide greater scalability, network opacity, and service independence, SR architecture [RFC8402] defines a Binding SID (BSID). A Binding SID is bound to an SR policy which typically involves a list of SIDs. These Binding SIDs may appear in another SR Policy or may be used to steer service traffic from the service origin. The TTL handling mechanisms for MPLS traceroute procedures involving Binding SIDs is described below.

3.2.1. Uniform Model

When the node advertising the Binding SID is operating in uniform mode [RFC3443], it SHOULD send FEC stack change sub-TLV as in sec 4.5.1 of [RFC8029]. The ingress node SHOULD increment the TTL of

Binding SID label at every step until "egress" return code is sent for all the new FECs included due to FEC stack change and all the Tunnels replaced by the Binding SID are completely traced. It is required that all the label popping nodes involved in these tunnels MUST support uniform model and copy the TTL to bottom label when the label is popped.

3.2.2. Shortpipe Model

When the node advertising the Binding SID is operating in short pipe model [RFC3443], it SHOULD not send FEC stack change sub-TLV. The Binding SID is treated as single hop and the nodes internal to the Tunnel represented by Binding SID SHOULD NOT be traced.

4. Backward Compatibility

The extension proposed in this document is backward compatible with procedures described in [RFC8029] and [RFC8287]. If the LSR with the proposed solution is the Ingress and all other LSR in the SR tunnel are not with the extension, Then no LSR is going to set 'P' bit so ingress LSR with new extension will work as per [RFC8029] and [RFC8287]. If the LSR with the proposed extension is the one of the transit router and if its the PHP then it may set 'P' bit based on the section 3. Ingress may not react to the 'P' bit and traceroute will continue to work as per [RFC8029] and [RFC8287].

5. Security Considerations

TBD

6. IANA Considerations

IANA has created and now maintains a registry entitled "DS Flags". The registration policy for this registry is Standards Action [RFC5226]. IANA has made the following assignments:

Bit Number Name Reference

7 N: Treat as a Non-IP Packet [RFC8029]

6 I: Interface and Label Stack Object Request [RFC8029]

5 E: ELI/EL push indicator [RFC8012]

4 L: Label-based load balance indicator [RFC8012]

3 P: Penultimate Hop router

2-0 Unassigned

7. Acknowledgements

Thanks to Przemyslaw Krol for careful review and comments.

8. References

8.1. Normative References

- [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>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <<https://www.rfc-editor.org/info/rfc5226>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.
- [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>>.

8.2. Informative References

- [RFC3443] Agarwal, P. and B. Akyol, "Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks", RFC 3443, DOI 10.17487/RFC3443, January 2003, <<https://www.rfc-editor.org/info/rfc3443>>.

Authors' Addresses

Kapil Arora
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: kapilaro@juniper.net

Shraddha Hegde
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Sam Aldrin
Google

Email: aldrin.ietf@gmail.com

Stephane Litkowski
Orange Business Service

Email: stephane.litkowski@orange.com

Muhammad Durrani
Equinix

Email: mdurrani@equinix.com

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 17, 2020

R. Bonica
S. Hegde
Juniper Networks
Y. Kamite
NTT Communications Corporation
A. Alston
D. Henriques
Liquid Telecom
L. Jalil
Verizon
J. Halpern
Ericsson
J. Linkova
Google
G. Chen
Baidu
October 15, 2019

Segment Routing Mapped To IPv6 (SRm6)
draft-bonica-spring-srv6-plus-06

Abstract

This document describes Segment Routing mapped to IPv6 (SRm6). SRm6 is a Segment Routing (SR) solution that leverages IPv6. It supports a wide variety of use-cases while remaining in strict compliance with IPv6 specifications. SRm6 is optimized for ASIC-based forwarding devices that operate at high data rates.

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 April 17, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Overview	3
2. Requirements Language	4
3. Paths, Segments And Instructions	5
4. Segment Types	6
4.1. Adjacency Segments	6
4.2. Node Segments	7
4.3. Binding Segments	7
5. Segment Identifiers (SID)	8
5.1. Range	9
5.2. Assigning SIDs to Adjacency Segments	10
5.3. Assigning SIDs to Node Segments	11
5.4. Assigning SIDs to Binding Segments	11
6. Service Instructions	11
6.1. Per-Segment	11
6.2. Per-Path	12
7. The IPv6 Data Plane	12
7.1. The Routing Header	13
7.2. The Destination Options Header	14
8. Control Plane	15
9. Differences Between SRv6 and SRv6+	15
9.1. Routing Header Size	15
9.2. Decoupling of Topological and Service Instructions	17
9.3. Authentication	17
9.4. Traffic Engineering Capability	18
9.5. IP Addressing Architecture	18
10. Compliance	18
11. Operational Considerations	19
11.1. Ping and Traceroute	19
11.2. ICMPv6 Rate Limiting	19
12. IANA Considerations	19
13. Security Considerations	19

14. Acknowledgements	19
15. References	19
15.1. Normative References	19
15.2. Informative References	21
Authors' Addresses	22

1. Overview

Network operators deploy Segment Routing (SR) [RFC8402] so that they can forward packets through SR paths. An SR path provides unidirectional connectivity from its ingress node to its egress node. While an SR path can follow the least cost path from ingress to egress, it can also follow any other path.

An SR path contains one or more segments. A segment provides unidirectional connectivity from its ingress node to its egress node. It includes a topological instruction that controls its behavior.

The topological instruction is executed on the segment ingress node. It determines the segment egress node and the method by which the segment ingress node forwards packets to the segment egress node.

Per-segment service instructions can augment a segment. Per-segment service instructions, if present, are executed on the segment egress node.

Likewise, a per-path service instruction can augment a path. The per-path service instruction, if present, is executed on the path egress node. Section 3 of this document illustrates the relationship between SR paths, segments and instructions.

A Segment Identifier (SID) identifies each segment. Because there is a one-to-one relationship between segments and the topological instructions that control them, the SID that identifies a segment also identifies the topological instruction that controls it.

A SID is different from the topological instruction that it identifies. While a SID identifies a topological instruction, it does not contain the topological instruction that it identifies. Therefore, a SID can be encoded in relatively few bits, while the topological instruction that it identifies may require many more bits for encoding.

An SR path can be represented by its ingress node as an ordered sequence of SIDs. In order to forward a packet through an SR path, the SR ingress node encodes the SR path into the packet as an ordered sequence of SIDs. It can also augment the packet with service instructions.

Because the SR ingress node is also the first segment ingress node, it executes the topological instruction associated with the first segment. This causes the packet to be forwarded to the first segment egress node. When the first segment egress node receives the packet, it executes any per-segment service instructions that augment the first segment.

If the SR path contains exactly one segment, the first segment egress node is also the path egress node. In this case, that node executes any per-path service instruction that augments the path, and SR forwarding is complete.

If the SR path contains multiple segments, the first segment egress node is also the second segment ingress node. In this case, that node executes the topological instruction associated with the second segment. The above-described procedure continues until the packet arrives at the SR egress node.

In the above-described procedure, only the SR ingress node maintains path information. Segment ingress and egress nodes maintain information regarding the segments in which they participate, but they do not maintain path information.

The SR architecture, described above, can leverage either an MPLS [RFC3031] data plane or an IPv6 [RFC8200] data plane. SR-MPLS [I-D.ietf-spring-segment-routing-mpls] leverages MPLS. SRv6 [I-D.ietf-spring-srv6-network-programming] [I-D.ietf-6man-segment-routing-header] leverages IPv6.

This document describes Segment Routing mapped to IPv6 (SRm6). SRm6 is an SR variant that leverages IPv6. It supports a wide variety of use-cases while remaining in strict compliance with IPv6 specifications. SRm6 is optimized for ASIC-based forwarding devices that operate at high data rates. Section 9 of this document highlights differences between SRv6 and SRm6.

2. 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.

3. Paths, Segments And Instructions

An SRm6 path is determined by the segments that it contains. It can be represented by its ingress node as an ordered sequence of SIDs.

A segment is determined by its ingress node and by the topological instruction that controls its behavior. The topological instruction determines the segment egress node and the method by which the segment ingress node forwards packets to the segment egress node.

Per-segment service instructions augment, but do not determine, segments. A segment ingress node can:

- o Send one packet through a segment with one per-segment service instruction.
- o Send another packet through the same segment with a different per-segment service instruction.
- o Send another packet through the same segment without any per-segment service instructions.

Likewise, per-path service instructions augment, but do not determine, paths.

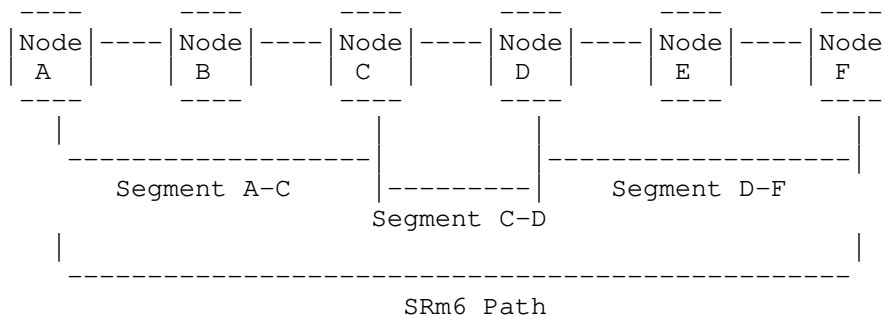


Figure 1: Paths, Segments And Instructions

Figure 1 depicts an SRm6 path. The path provides unidirectional connectivity from its ingress node (i.e., Node A) to its egress node (i.e., Node F). It contains Segment A-C, Segment C-D and Segment D-F.

In Segment A-C, Node A is the ingress node, Node B is a transit node, and Node C is the egress node. Therefore, the topological instruction that controls the segment is executed on Node A, while

per-segment service instructions that augment the segment (if any exist) are executed on Node C.

In Segment C-D, Node C is the ingress node and Node D is the egress node. Therefore, the topological instruction that controls the segment is executed on Node C, while per-segment service instructions that augment the segment (if any exist) are executed on Node D.

In Segment D-F, Node D is the ingress node, Node E is a transit node, and Node F is the egress node. Therefore, the topological instruction that controls the segment is executed on Node D, while per-segment service instructions that augment the segment (if any exist) are executed on Node F.

Node F is also the path egress node. Therefore, if a per-path service instruction augments the path, it is executed on Node F.

Segments A-C, C-D and D-F are also contained by other paths that are not included in the figure.

4. Segment Types

SRm6 supports the following segment types:

- o Adjacency.
- o Node.
- o Binding.

Adjacency segments forward packets through a specified link that connects the segment ingress node to the segment egress node. Node segments forward packets through the least cost path from the segment ingress node to the segment egress node. Binding segments facilitate recursive application of SRm6. They cause SRm6 paths to be nested in a hierarchy.

Each segment type is described below.

4.1. Adjacency Segments

When a packet is submitted to an adjacency segment, the topological instruction associated with that segment operates upon the packet. The topological instruction executes on the segment ingress node and receives the following parameters:

- o An IPv6 address that identifies an interface on the segment egress node.

- o An interface identifier.

The topological instruction behaves as follows:

- o If the interface that was received as a parameter is not operational, discard the packet and send an ICMPv6 [RFC4443] Destination Unreachable message (Code: 5, Source Route Failed) to the packet's source node.
- o Overwrite the packet's Destination Address with the IPv6 address that was received as a parameter.
- o Forward the packet through the above-mentioned interface.

For further processing details, see [I-D.bonica-6man-comp-rtg-hdr].

4.2. Node Segments

When a packet is submitted to a node segment, the topological instruction associated with that segment operates upon the packet. The topological instruction executes on the segment ingress node and receives an IPv6 address as a parameter. The IPv6 address identifies an interface on the segment egress node.

The topological instruction behaves as follows:

- o If the segment ingress node does not have a viable route to the IPv6 address received as a parameter, discard the packet and send an ICMPv6 Destination Unreachable message (Code:1 Net Unreachable) to the packet's source node.
- o Overwrite the packet's Destination Address with the destination address that was received as a parameter.
- o Forward the packet to the next hop along the least cost path to the segment egress node. If there are multiple least cost paths to the segment egress node (i.e., Equal Cost Multipath), execute procedures so that all packets belonging to a flow are forwarded through the same next hop.

For further processing details, see [I-D.bonica-6man-comp-rtg-hdr].

4.3. Binding Segments

When a packet is submitted to a binding segment, the topological instruction associated with that segment operates upon the packet. The topological instruction executes on the segment ingress node and receives the following parameters:

- o An IPv6 address.
- o A SID list length.
- o A SID list.

The topological instruction behaves as follows:

- o If the segment ingress node does not have a viable route to the IPv6 address received as a parameter, discard the packet and send an ICMPv6 Destination Unreachable message (Code:1 Net Unreachable) to the packet's source node.
- o Prepend a Compressed Routing Header (CRH) [I-D.bonica-6man-comp-rtg-hdr] to the packet. Copy the SID list length, received as a parameter, to the CRH Segments Left field. Also copy the SID list, received as a parameter, to the CRH SID list.
- o Prepend an IPv6 header to the packet. Copy the IPv6 address, received as a parameter, to the IPv6 Destination Address.
- o Forward the packet to the next hop along the least cost path to the IPv6 address received as a parameter. If there are multiple least cost paths to the IPv6 address received as a parameter (i.e., Equal Cost Multipath), execute procedures so that all packets belonging to a flow are forwarded through the same next hop.

For further processing details, see [I-D.bonica-6man-comp-rtg-hdr].

5. Segment Identifiers (SID)

A Segment Identifier (SID) is an unsigned integer that identifies a segment. Because there is a one-to-one relationship between segments and the topological instructions that control them, the SID that identifies a segment also identifies the topological instruction that controls it.

A SID is different from the topological instruction that it identifies. While a SID identifies a topological instruction, it does not contain the topological instruction that it identifies. Therefore, a SID can be encoded in relatively few bits, while the topological instruction that it identifies may require many more bits for encoding.

SIDs have node-local significance. This means that a segment ingress node MUST identify each segment that it originates with a unique SID.

However, a SID that is used by one segment ingress node to identify a segment that it originates can be used by another segment ingress node to identify another segment. For example, SID S can identify both of the following:

- o A segment whose ingress is Node A and whose egress is Node Z.
- o Another segment whose ingress is Node B and whose egress is also node Z.

Although SIDs have node-local significance, an SRm6 path can be uniquely identified by its ingress node and an ordered sequence of SIDs. This is because the topological instruction associated with each segment determines the ingress node of the next segment (i.e., the node upon which the next SID has significance.)

SIDs can be assigned in a manner that simplifies network operations. See Section 5.2 and Section 5.3 for details.

5.1. Range

SID values range from 0 to a configurable Maximum SID Value (MSV). The values 0 through 15 are reserved for future use. The following are valid MSVs:

- o 65,535 (i.e., $2^{16} - 1$).
- o 4,294,967,295 (i.e., $2^{32} - 1$).

In order to optimize packet encoding (Section 7.1), network operators can configure all nodes within an SRm6 domain to have the smallest feasible MSV. The following paragraphs explain how an operator determines the smallest feasible MSV.

Consider an SRm6 domain that contains 5,000 nodes connected to one another by point-to-point infrastructure links. The network topology is not a full-mesh. In fact, each node supports 200 point-to-point infrastructure links or fewer. Given this SRm6 domain, we will determine the smallest feasible MSV under the following conditions:

- o The SRm6 domain contains adjacency segments only.
- o The SRm6 domain contains node segments only.
- o The SRm6 domain contains both adjacency and node segments.

If an SRm6 domain contains adjacency segments only, and each node creates a adjacency segment to each of its neighbors, each node will

create 200 segments or fewer and consume 200 SIDs or fewer. This is because each node has 200 neighbors or fewer. Because SIDs have node-local significance (i.e., they can be reused across nodes), the smallest feasible MSV is 65,535.

Adding nodes to this SRm6 domain will not increase the smallest feasible MSV, so long as each node continues to support 65,519 point-to-point infrastructure links or fewer. If a single node is added to the domain and that node supports 65,520 infrastructure links, the smallest feasible MSV will increase to 4,294,967,295.

If an SRm6 domain contains node segments only, and every node creates a node segment to every other node, every node will create 4,999 segments and consume 4,999 SIDs. This is because the domain contains 5,000 nodes. Because SIDs have node-local significance (i.e., they can be reused across nodes), the smallest feasible MSV is 65,535.

Adding nodes to this SRm6 domain will not increase the smallest feasible MSV until the number of nodes exceeds 65,519. When the smallest feasible MSV increases, it becomes 4,294,967,295.

If an SRm6 domain contains both adjacency and node segments, each node will create 5,199 segments or fewer and consume 5,199 SIDs or fewer. This value is the sum of the following:

- o The number of node segments that each node will create, given that every node creates a node segment to every other node (i.e., 4,999).
- o The number of adjacency segments that each node will create, given that each node creates a adjacency segment to each of its neighbors (i.e., 200 or fewer).

Because SIDs have node-local significance (i.e., they can be reused across nodes), the smallest feasible MSV is 65,535.

Adding nodes to this SRm6 domain will not increase the smallest feasible MSV until the number of nodes plus the maximum number of infrastructure links per node exceeds 65,519. When the smallest feasible MSV increases, it becomes 4,294,967,295.

5.2. Assigning SIDs to Adjacency Segments

Network operators can establish conventions by which they assign SIDs to adjacency segments. These conventions can simplify network operations.

For example, a network operator can reserved a range of SIDs for adjacency segments. It can further divide that range into subranges, so that all segments sharing a common egress node are identified by SIDs from the same subrange.

5.3. Assigning SIDs to Node Segments

In order to simplify network operations, all node segments that share a common egress node are identified by the same SID. In order to maintain this discipline, network wide co-ordination is required.

For example, assume that an SRm6 domain contains N nodes. Network administrators reserve a block of N SIDs and configure one of those SIDs on each node. Each node advertises its SID into the control plane. When another node receives that advertisement, it creates a node segment between itself and the advertising node. It also associates the SID that it received in the advertisement with the newly created segment. See [I-D.bonica-lsr-crh-isis-extensions] for details.

5.4. Assigning SIDs to Binding Segments

Network operators can establish conventions by which they assign SIDs to binding segments. These conventions can simplify network operations.

For example, a network operator can reserve a range of SIDs for binding segments. It can further divide that range into subranges, so that all segments sharing a common egress node are identified by SIDs from the same subrange.

6. Service Instructions

SRm6 supports the following service instruction types:

- o Per-segment.
- o Per-path.

Each is described below.

6.1. Per-Segment

Per-segment service instructions can augment a segment. Per-segment service instructions, if present, are executed on the segment egress node. Because the path egress node is also a segment egress node, it can execute per-segment service instructions.

The following are examples of per-segment service instructions:

- o Expose a packet to a firewall policy.
- o Expose a packet to a sampling policy.

Per-segment Service Instruction Identifiers identify a set of service instructions. Per-segment Service Instruction Identifiers are allocated and distributed by a controller. They have domain-wide significance.

6.2. Per-Path

A per-path service instruction can augment a path. The per-path service instruction, if present, is executed on the path egress node.

The following are examples of per-path service instructions:

- o De-encapsulate a packet and forward its newly exposed payload through a specified interface.
- o De-encapsulate a packet and forward its newly exposed payload using a specified routing table.

Per-path Service Instruction Identifiers identify per-path service instructions. Per-path Service Instruction Identifiers are allocated and distributed by the processing node (i.e., the path egress node). They have node-local significance. This means that the path egress node MUST allocate a unique Per-path Service Instruction Identifier for each per-path service instruction that it instantiates.

7. The IPv6 Data Plane

SRm6 ingress nodes generate IPv6 header chains that represent SRm6 paths. An IPv6 header chain contains an IPv6 header. It can also contain one or more extension headers.

An extension header chain that represents an SRm6 path can contain any valid combination of IPv6 extension headers. The following bullet points describe how SRm6 leverages IPv6 extension headers:

- o If an SRm6 path contains multiple segments, the IPv6 header chain that represents it MUST contain a Routing header. The SRm6 path MUST be encoded in the Routing header as an ordered sequence of SIDs.
- o If an SRm6 path is augmented by a per-path service instruction, the IPv6 header chain that represents it MUST contain a

Destination Options header. The Destination Options header MUST immediately precede an upper-layer header and it MUST include a Per-Path Service Instruction Identifier.

- o If an SRm6 path contains a segment that is augmented by a per-segment service instruction, the IPv6 chain that represents it MUST contain a Routing header and a Destination Options header. The Destination Options header MUST immediately precede a Routing header and it MUST include the Per-Segment Service Instruction Identifier.

The following subsections describe how SRm6 uses the Routing header and the Destination Options header.

7.1. The Routing Header

SRm6 defines two new Routing header types. Generically, they are called the Compressed Routing Header (CRH) [I-D.bonica-6man-comp-rtg-hdr]. More specifically, the 16-bit version of the CRH is called the CRH-16, while the 32-bit version of the CRH is called the CRH-32.

Both CRH versions contain the following fields:

- o Next Header - Identifies the header immediately following the CRH.
- o Hdr Ext Len - Length of the CRH.
- o Routing Type - Identifies the Routing header variant (i.e., CRH-16 or CRH-32).
- o Segments Left - The number of segments still to be traversed before reaching the path egress node.
- o SID List - Represents the SRm6 path as an ordered list of SIDs. SIDs are listed in reverse order, with SID[0] representing the final segment, SID[1] representing the penultimate segment, and so forth. SIDs are listed in reverse order so that Segments Left can be used as an index to the SID List. The SID indexed by Segments Left is called the current SID.

In the CRH-16, each SID list entry is encoded in 16-bits. In the CRH-32, each SID list entry is encoded in 32-bits. In networks where the smallest feasible MSV (Section 5.1) is greater than 65,635, CRH-32 is required. Otherwise, CRH-16 is preferred.

As per [RFC8200], when an IPv6 node receives a packet, it examines the packet's destination address. If the destination address

represents an interface belonging to the node, the node processes the next header. If the node encounters and recognizes the CRH, it processes the CRH as follows:

- o If Segments Left equal 0, skip over the CRH and process the next header in the packet.
- o Decrement Segments Left.
- o Search for the current SID in a local table that maps SID's to topological instructions. If the current SID cannot be found in that table, send an ICMPv6 Parameter Problem message to the packet's Source Address and discard the packet.
- o Execute the topological instruction found in the table as described in Section 4. This causes the packet to be forwarded to the segment egress node.

When the packet arrives at the segment egress node, the above-described procedure is repeated. For further processing details, see [I-D.bonica-6man-comp-rtg-hdr].

7.2. The Destination Options Header

According to [RFC8200], the Destination Options header contains one or more IPv6 options. It can occur twice within a packet, once before a Routing header and once before an upper-layer header. The Destination Options header that occurs before a Routing header is processed by the first destination that appears in the IPv6 Destination Address field plus subsequent destinations that are listed in the Routing header. The Destination Options header that occurs before an upper-layer header is processed by the packet's final destination only.

Therefore, SRm6 defines the following new IPv6 options:

- o The SRm6 Per-Segment Service Instruction Option
[I-D.bonica-6man-seg-end-opt]
- o The SRm6 Per-Path Service Instruction Option
[I-D.bonica-6man-vpn-dest-opt]

The SRm6 Per-Segment Service Instruction Option is encoded in a Destination Options header that precedes the CRH. Therefore, it is processed by every segment egress node. It includes a Per-Segment Service Instruction Identifier and causes segment egress nodes to execute per-segment service instructions.

The SRm6 Per-Path Service Instruction Option is encoded in a Destination Options header that precedes the upper-layer header. Therefore, it is processed by the path egress node only. It includes a Per-Path Service Instruction Identifier and causes the path egress node to execute a per-path service instruction.

8. Control Plane

IS-IS extensions [I-D.bonica-lsr-crh-isis-extensions] have been defined for the following purposes:

- o So that SRm6 segment ingress nodes can flood information regarding adjacency segments that they originate.
- o So that SRm6 segment egress nodes can flood information regarding node segments that they terminate.

BGP extensions [I-D.ssangli-idr-bgp-vpn-srv6-plus] are defined so that SRm6 path egress nodes can associate path-terminating service instructions with Network Layer Reachability Information (NLRI). Additional BGP extensions [I-D.alston-spring-crh-bgp-signalling] are defined so that SIDs can be mapped to the IPv6 addresses that they represent.

9. Differences Between SRv6 and SRv6+

9.1. Routing Header Size

SRv6 defines a Routing header type, called the Segment Routing Header (SRH). The SRH contains a field that represents the SRv6 path as an ordered sequence of SIDs. Each SID contained by that field is 128 bits long.

Likewise, SRm6 defines two Routing Header Types, called CRH-16 and CRH-32. Both contain a field that represents the SRv6 path as an ordered sequence of SIDs. In the CRH-16, each SID is 16 bits long. In the CRH-32, each SID is 32 bits long.

SIDs	SRv6 SRH (128-bit SID)	SRm6 CRH-16	SRm6 CRH-32
1	24	8	8
2	40	8	16
3	56	16	16
4	72	16	24
5	88	16	24
6	104	16	32
7	120	24	32
8	136	24	40
9	152	24	40
10	168	24	N/A
11	184	32	N/A
12	200	32	N/A
13	216	32	N/A
14	232	32	N/A
15	248	40	N/A
16	264	40	N/A
17	280	40	N/A
18	296	40	N/A

Table 1: Routing Header Size (in Bytes) As A Function Of Routing Header Type and Number Of SIDs

Table 1 reflects Routing header size as a function of Routing header type and number of SIDs contained by the Routing header. Due to their relative immaturity, [I-D.filsfils-spring-net-pgm-extension-srv6-usid], [I-D.li-spring-compressed-srv6-np] and [I-D.mirsky-6man-unified-id-sr] are omitted from this analysis.

Large Routing headers are undesirable for the following reasons:

- o Many ASIC-based forwarders copy the entire IPv6 extension header chain from buffer memory to on-chip memory. As the size of the IPv6 extension header chain increases, so does the cost of this copy.
- o Because Path MTU Discovery (PMTUD) [RFC8201] is not entirely reliable, many IPv6 hosts refrain from sending packets larger than the IPv6 minimum link MTU (i.e., 1280 bytes). When packets are small, the overhead imposed by large Routing headers becomes pronounced.

9.2. Decoupling of Topological and Service Instructions

SRm6 decouples topological instructions from service instructions. Topological instructions are invoked at the segment ingress node, as a result of CRH processing, while service instructions are invoked at the segment egress node, as a result of Destination Option processing. Therefore, network operators can use SRm6 mechanisms to support topological instructions, service instructions, or both.

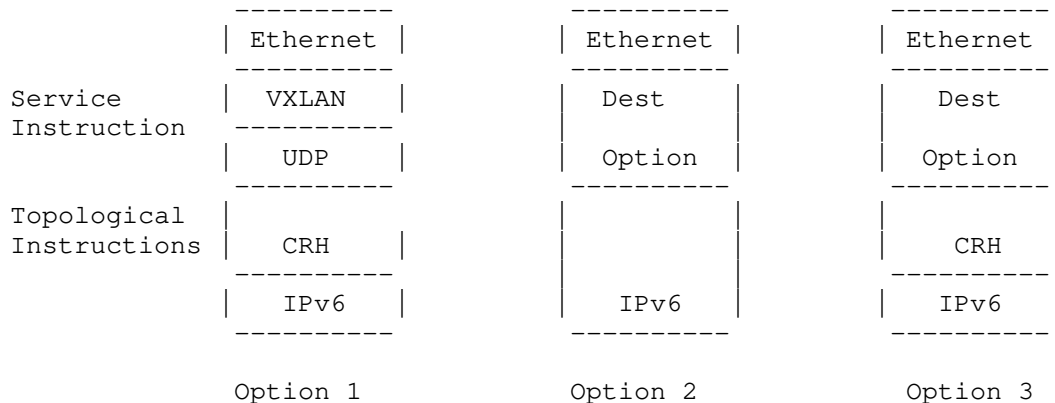


Figure 2: EVPN Design Alternatives

Figure 2 illustrates this point by depicting design options available to network operators offering Ethernet Virtual Private Network [RFC7432] services over Virtual eXtensible Local Area Network (VXLAN) [RFC7348]. In Option 1, the network operator encodes topological instructions in the CRH, while encoding service instructions in a VXLAN header. In Option 2, the network operator encodes service instructions in a Destination Options header, while allowing traffic to traverse the least cost path between the ingress and egress Provider Edge (PE) routers. In Option 3, the network operator encodes topological instructions in the CRH, and encodes service instructions in a Destination Options header.

9.3. Authentication

The IPv6 Authentication Header (AH) [RFC4302] can be used to authenticate SRm6 packets. However, AH processing is not defined in SRv6.

9.4. Traffic Engineering Capability

SRm6 supports traffic engineering solutions that rely exclusively upon adjacency segments. For example, consider an SRm6 network whose diameter is 12 hops and whose minimum feasible MSV is 65,525. In that network, in the worst case, SRm6 overhead is 72 bytes (i.e., a 40-byte IPv6 header and a 32-byte CRH-16).

SRv6 also supports traffic engineering solutions that rely exclusively upon adjacency segments (i.e., END.X SIDs). However, SRv6 overhead may be prohibitive. For example, consider an SRv6 network whose diameter is 12 hops. In the worst case, SRv6 overhead is 240 bytes (i.e., a 40 byte IPv6 header and a 200-byte SRH).

9.5. IP Addressing Architecture

In SRv6, an IPv6 address can represent either of the following:

- o A network interface
- o An instruction instantiated on a node (i.e., an SRv6 SID)

In SRm6 an IPv6 address always represents a network interface, as per [RFC4291].

10. Compliance

In order to be compliant with this specification, an SRm6 implementation MUST:

- o Be able to process IPv6 options as described in Section 4.2 of [RFC8200].
- o Be able to process the Routing header as described in Section 4.4 of [RFC8200].
- o Be able to process the Destination Options header as described in Section 4.6 of [RFC8200].
- o Support the CRH-16 and the CRH-32

Additionally, an SRm6 implementation MAY:

- o Recognize the Per-Segment Service Instruction Option.
- o Recognize the Per-Path Service Instruction Option.

11. Operational Considerations

11.1. Ping and Traceroute

Ping and Traceroute [RFC2151] both operate correctly in SRm6 (i.e., in the presence of the CRH).

11.2. ICMPv6 Rate Limiting

As per [RFC4443], SRm6 nodes rate limit the ICMPv6 messages that they emit.

12. IANA Considerations

SID values 0-15 are reserved for future use. They may be assigned by IANA, based on IETF Consensus.

IANA is requested to establish a "Registry of SRm6 Reserved SIDs". Values 0-15 are reserved for future use.

13. Security Considerations

SRm6 domains MUST NOT span security domains. In order to enforce this requirement, security domain edge routers MUST do one of the following:

- o Discard all inbound SRm6 packets whose IPv6 destination address represents domain infrastructure.
- o Authenticate [RFC4302] [RFC4303] all inbound SRm6 packets whose IPv6 destination address represents domain infrastructure.

14. Acknowledgements

The authors wish to acknowledge Dr. Vanessa Ameen, Reji Thomas, Parag Kaneriy, Rejesh Shetty, Nancy Shaw, and John Scudder.

15. References

15.1. Normative References

[I-D.alston-spring-crh-bgp-signalling]

Alston, A., Henriques, D., and R. Bonica, "BGP Extensions for IPv6 Compressed Routing Header (CRH)", draft-alston-spring-crh-bgp-signalling-01 (work in progress), July 2019.

- [I-D.bonica-6man-comp-rtg-hdr]
Bonica, R., Kamite, Y., Niwa, T., Alston, A., Henriques, D., So, N., Xu, F., Chen, G., Zhu, Y., Yang, G., and Y. Zhou, "The IPv6 Compressed Routing Header (CRH)", draft-bonica-6man-comp-rtg-hdr-07 (work in progress), September 2019.
- [I-D.bonica-6man-seg-end-opt]
Bonica, R., Halpern, J., Kamite, Y., Niwa, T., So, N., Xu, F., Chen, G., Zhu, Y., Yang, G., and Y. Zhou, "The Per-Segment Service Instruction (PSSI) Option", draft-bonica-6man-seg-end-opt-04 (work in progress), July 2019.
- [I-D.bonica-6man-vpn-dest-opt]
Bonica, R., Kamite, Y., Lenart, C., So, N., Xu, F., Presbury, G., Chen, G., Zhu, Y., Yang, G., and Y. Zhou, "The Per-Path Service Instruction (PPSI) Option", draft-bonica-6man-vpn-dest-opt-06 (work in progress), July 2019.
- [I-D.bonica-lsr-crh-isis-extensions]
Kaneriya, P., Shetty, R., Hegde, S., and R. Bonica, "IS-IS Extensions To Support The IPv6 Compressed Routing Header (CRH)", draft-bonica-lsr-crh-isis-extensions-00 (work in progress), May 2019.
- [I-D.ssangli-idr-bgp-vpn-srv6-plus]
Ramachandra, S. and R. Bonica, "BGP based Virtual Private Network (VPN) Services over SRv6+ enabled IPv6 networks", draft-ssangli-idr-bgp-vpn-srv6-plus-02 (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>>.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <<https://www.rfc-editor.org/info/rfc4443>>.

- [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>>.
- [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>>.

15.2. Informative References

- [I-D.filsfils-spring-net-pgm-extension-srv6-usid]
Filsfils, C., Camarillo, P., Cai, D., Jiang, Z., daniel.voyer@bell.ca, d., Shawky, A., Leymann, N., Steinberg, D., Zandi, S., Dawra, G., Meilik, I., Uttaro, J., Jalil, L., So, N., Fiumano, M., and M. Khaddam, "Network Programming extension: SRv6 uSID instruction", draft-filsfils-spring-net-pgm-extension-srv6-usid-02 (work in progress), August 2019.
- [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-24 (work in progress), October 2019.
- [I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-22 (work in progress), May 2019.
- [I-D.ietf-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-04 (work in progress), October 2019.
- [I-D.li-spring-compressed-srv6-np]
Li, Z., Li, C., Peng, S., Wang, Z., and B. Liu, "Compressed SRv6 Network Programming", draft-li-spring-compressed-srv6-np-00 (work in progress), July 2019.

- [I-D.mirsky-6man-unified-id-sr]
Cheng, W., Mirsky, G., Peng, S., Aihua, L., Wan, X., and C. Wei, "Unified Identifier in IPv6 Segment Routing Networks", draft-mirsky-6man-unified-id-sr-03 (work in progress), July 2019.
- [RFC2151] Kessler, G. and S. Shepard, "A Primer On Internet and TCP/IP Tools and Utilities", FYI 30, RFC 2151, DOI 10.17487/RFC2151, June 1997, <<https://www.rfc-editor.org/info/rfc2151>>.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <<https://www.rfc-editor.org/info/rfc3031>>.
- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<https://www.rfc-editor.org/info/rfc4302>>.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <<https://www.rfc-editor.org/info/rfc4303>>.
- [RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <<https://www.rfc-editor.org/info/rfc7348>>.
- [RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <<https://www.rfc-editor.org/info/rfc7432>>.
- [RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <<https://www.rfc-editor.org/info/rfc8201>>.

Authors' Addresses

Ron Bonica
Juniper Networks
Herndon, Virginia 20171
USA

Email: rbonica@juniper.net

Shraddha Hegde
Juniper Networks
Embassy Business Park
Bangalore, KA 560093
India

Email: shraddha@juniper.net

Yuji Kamite
NTT Communications Corporation
3-4-1 Shibaura, Minato-ku
Tokyo 108-8118
Japan

Email: y.kamite@ntt.com

Andrew Alston
Liquid Telecom
Nairobi
Kenya

Email: Andrew.Alston@liquidtelecom.com

Daniam Henriques
Liquid Telecom
Johannesburg
South Africa

Email: daniam.henriques@liquidtelecom.com

Luay Jalil
Verizon
Richardson, Texas
USA

Email: luay.jalil@one.verizon.com

Joel Halpern
Ericsson
P. O. Box 6049
Leesburg, Virginia 20178
USA

Email: joel.halpern@ericsson.com

Jen Linkova
Google
Mountain View, California 94043
USA

Email: furry@google.com

Gang Chen
Baidu
No.10 Xibeiwang East Road Haidian District
Beijing 100193
P.R. China

Email: phdgang@gmail.com

SPRING
Internet-Draft
Intended status: Standards Track
Expires: January 9, 2020

C. Filsfils, Ed.
Cisco Systems, Inc.
D. Cai
Alibaba
Z. Jiang
Tencent
D. Voyer
Bell Canada
A. Shawky
Saudi Telecom Company
N. Leymann
Deutsche Telekom
D. Steinberg
Lapishills Consulting Limited
S. Zandi
G. Dawra
LinkedIn
I. Meilik
Broadcom
J. Uttaro
AT&T
L. Jalil
Verizon
N. So
Reliance
M. Fiumano
Sprint
M. Khaddam
Cox
J. Ma
China Unicom
S. Matsushima
Softbank
F. Ferguson
CenturyLink
T. Miyasaka
KDDI
K. Ebisawa
Toyota Motor Corporation
Y. Ueno
NTT Communications Corporation
W. Henderickx
Nokia
P. Jonnalagadda
Barefoot Networks
J. Bhattacharya
K. Raza
P. Camarillo, Ed.
Cisco Systems, Inc.
July 8, 2019

Network Programming extension: SRv6 uSID instruction
draft-filsfils-spring-net-pgm-extension-srv6-usid-00

Abstract

The SRv6 "micro segment" (SRv6 uSID or uSID for short) instruction is defined and illustrated.

It is a straightforward extension to the SRv6 Network Programming model and its SRH encapsulation.

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 January 9, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
2. Terminology	4
2.1. Notation for human readability	5
3. SRv6 behaviors associated with a uSID	5
3.1. uN	5
4. Routing	6
5. Illustration	6
5.1. Reference diagram	6
5.2. SRv6 overlay with underlay optimization	6
6. Benefits	8
7. Security	9
8. Acknowledgements	9
9. Contributors	9
10. References	9
10.1. Normative References	9
10.2. Informative References	10
Authors' Addresses	10

1. Introduction

SRv6 Network Programming [I-D.ietf-spring-srv6-network-programming] defines a mechanism to build a network program with topological and service segments. It leverages the SRH [I-D.ietf-6man-segment-routing-header] to encode a network program together with optional metadata shared among the different SIDs.

This draft extends SRv6 Network Programming with a new type of SRv6 SID behavior: SRv6 uN. This is combined with the rest of instructions of the network program and the SRH encapsulation to build programs in a scalable and efficient way.

2. Terminology

The SRv6 Network Programming [I-D.ietf-spring-srv6-network-programming] and SRH [I-D.ietf-6man-segment-routing-header] terminology is leveraged and extended with the following terms:

uSID carrier: a 128bit SRv6 SID of format <uSID-Block><Active-uSID><Next-uSID>...<Last-uSID><End-of-Carrier>...<End-of-Carrier>.

uSID block: A block of uSID's

It can be any IPv6 prefix allocated to the provider (e.g. /40 or /48), or it can be any block generally available for private use. An SR domain may have multiple uSID blocks.

In this document we leverage FC00::/8 block reserved for private use as ULA space (RFC4193). Throughout this document we use FC00::/16 as the illustrated uSID block. ULA space allows for up to 256 uSID blocks in FC00::/8.

uSID: in this document a 16-bit ID. A different length may be used.

Active uSID: first uSID after the uSID block

Next uSID: next uSID after the Active uSID

Last uSID: from left to right, the last uSID before the first End-of-Carrier uSID

End-of-Carrier: reserved ID used to mark the end of a uSID carrier. The value 0000 is selected as End-of-Carrier. All of the empty uSID carrier positions must be filled with the End-of-Carrier ID. Hence, the End-of-Carrier can be present more than once in a uSID carrier.

Parent (node): the node at which an uSID is instantiated. The uSIDs are instantiated on a per-parent node basis.

Behavior of an uSID: the SRv6 function associated with a given ID. Section 3 defines them.

2.1. Notation for human readability

For human readability, the example in this document follow this notation:

FC00::/16 is the uSID block used in the SR domain

0N00: uN behavior bound to node N

3. SRv6 behaviors associated with a uSID

The SRv6 SRH encapsulation and its network programming model are extended with the following functions:

3.1. uN

The uN behavior is a variant of the endpoint behavior.

This behavior takes a 96b argument, "Arg", which contains the next uSIDs in the uSID carrier.

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

1. IF DA[32..47] != 0 ;; Ref1
2. Copy DA[32..127] into DA[16..111]
3. Set DA[112..127] to 0x0000
4. Forward the packet to the new DA
5. ELSE
6. Execute the End pseudocode ;; Ref2

Ref 1: DA[X..Y] refers to the bits from position X to Y (included) in the IPv6 Destination Address of the received packet. The bit 0 is the MSB, while the bit 127 is the LSB.

Ref 2: This refers to the End behavior as defined in Section 4.1 of [I-D.ietf-spring-srv6-network-programming]. The End behavior may be combined with the PSP, USP and USD flavours.

4. Routing

If N is configured with a uN SID FC00:0N00::/32 then the operator must ensure that N advertises FC00:0N00::/32 in routing.

5. Illustration

This section extends the illustrations for SRv6 Network Programming [I-D.filsfils-spring-srv6-net-pgm-illustration] to cover uSID. The reference topology is the same with the addition of link 6-8.

5.1. Reference diagram

Nodes 1 to 8 are considered within the network domain.

Nodes X and Y are outside the domain.

Nodes 1 and 8 act as PE respectively to nodes X and Y.

All the links within the domain have the same IGP metric. The IGP-metric shortest-path from 1 to 8 is 1-2-7-8 while the latency-metric shortest-path from 1 to 8 is 1-2-3-4-5-6-7-8.

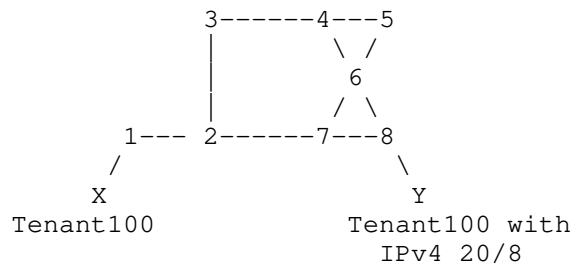


Figure 1: Reference topology

5.2. SRv6 overlay with underlay optimization

Let us illustrate a low-latency SR-L3VPN service delivered to a packet (X,Y).

PE 1 encapsulates (X, Y) in an outer IPv6 header with DA = FC00:0300:0500:0700:: and SRH (B:8:D0::; SL=1; NH=4). Leveraging the illustration conventions from SRv6 network programming, the following resulting packet leaves node 1 in the direction of node 3:

(A1::, FC00:0300:0500:0700::) (B:8:D0::; SL=1; NH=4) (X, Y)

FC00:0300:0500:0700:: is a uSID carrier encoding a source routed stateless path via node 3 then 5 then 7.

B:8:D0:: is an End.DT4 SID instantiated at node 8.

1 sends this packet to 2, as 2 is on the shortest-path to FC00:0300::/32 advertised by 3.

When 2 receives the packet, 2 performs a regular IPv6 FIB lookup. It finds a FIB entry for FC00:0300::/32 and forwards along the shortest path to 3.

When 3 receives the packet, 3 matches FC00:0300::/32 in its "My SID Table" and executes the uN behavior. The updated DA becomes FC00:0500:0700::. Node 3 then performs a lookup on the updated DA and forwards the packet to 5 along the shortest path to FC00:0500::/32.

The following packet leaves node 3:

(A1::, FC00:0500:0700::) (B:8:D0::; SL=1; NH=4) (X, Y)

4 forwards along the shortest path to FC00:0500::/32.

When 5 receives the packet, 5 matches FC00:0500::/32 in its "My SID Table" and executes the uN behavior. The updated DA becomes FC00:0700::. 5 performs a lookup on the updated DA and forwards the packet to 7 along the shortest path to FC00:0700::/32.

The following packet leaves node 5:

(A1::, FC00:0700::) (B:8:D0::; SL=1; NH=4) (X, Y)

6 forwards along the shortest path to FC00:0700::/32.

When 7 receives the packet, 7 matches FC00:0700::/32 in its "My SID Table" and finds the bound function uN. As a result, Node 7 executes the "End with PSP and USD support" pseudocode, decrementing the SL value in the SRH, and updating the DA with the next SID B:8:D0::. Since the SL value is zero the SRH is removed. Node 7 performs a lookup on the updated DA and forwards along the shortest path.

The following packet leaves node 7:

(A1::, B:8:D0::) (X, Y)

8 receives it, performs the End.DT4 function and sends the IP packet (X, Y) towards its VPN destination.

This example illustrates the benefits highlighted in the next section.

6. Benefits

Perfect integration with SRv6 Network Programming

SRv6 uSID is an instruction of the SRv6 network programming model

Perfect integration with SRH

Any SID in DA or SRH can be an SRv6 uSID carrier

Scalable SR Policy

7 uSID' per uSID carrier

21 source routing waypoints in solely 40bytes of overhead

T.Encaps.Red with an SRH of 40 bytes (8 fixed + 2 * 16 bytes)

7 uSID's in DA and 14 in SRH

Efficient MTU overhead

In apple to apple comparison, the SRv6 solution outperforms any alternative (VxLAN with SR-MPLS, CRH).

Scalable number of globally unique nodes in the domain

16-bit uSID: 65k uSIDs per domain block (*256 solely using FC/8)

32-bit uSID: 4.3M uSIDs per domain block (*256 solely using FC/8)

Hardware-friendly:

Leverages mature hardware capabilities (shift)

Avoids any extra lookup in indexed mapping table

Demonstrated by Cisco linerate implementation on Jericho1

Control-Plane friendly

No indexed mapping table is required

No routing extension is required: a simple /32 advertisement suffices

7. Security

The security rules defined in Section 7 of [I-D.ietf-spring-srv6-network-programming], protect intra-domain deployments that includes SRv6 uSID.

8. Acknowledgements

The authors would like to acknowledge Francois Clad, Peter Psenak, Ketan Talaulikar, Swadesh Agrawal, Zafar Ali, Darren Dukes, Kiran Sadshiran, Junaid Israr, Lakshmanan Srikanth, Asif Islam, Saleem Hafeez, Michael MacKenzie, Sushek Shekar, YuanChao Su, Alexander Preusche, Alberto Donzelli, Miya Kohno, David Smith, Ianik Semco, Bertrand Duvivier, Frederic Trate, Kris Michielsen, Eyal Dagan, Eli Stein, Ofer Iny, Elad Naor, Aviad Behar, Joseph Chin.

9. Contributors

Tomonobu Niwa
KDDI
Japan

Email: to-niwa@kddi.com

10. References

10.1. 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-21 (work in progress), June 2019.

[I-D.ietf-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-01 (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>>.

10.2. Informative References

- [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-00 (work in progress), February 2019.

Authors' Addresses

Clarence Filsfils (editor)
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Dennis Cai
Alibaba
China

Email: d.cai@alibaba-inc.com

Zhichun Jiang
Tencent
China

Email: zcjiang@tencent.com

Daniel Voyer
Bell Canada
Canada

Email: daniel.voyer@bell.ca

Ahmed Shawky
Saudi Telecom Company
Saudi Arabia

Email: ashawky@stc.com.sa

Nic Leymann
Deutsche Telekom
Germany

Email: N.Leymann@telekom.de

Dirk Steinberg
Lapishills Consulting Limited
Cyprus

Email: dirk@lapishills.com

Shawn Zandi
LinkedIn
United States of America

Email: szandi@linkedin.com

Gaurav Dawra
LinkedIn
United States of America

Email: gdawra@linkedin.com

Israel Meilik
Broadcom
Israel

Email: israel.meilik@broadcom.com

Jim Uttaro
AT&T
United States of America

Email: jul738@att.com

Luay Jalil
Verizon
United States of America

Email: luay.jalil@one.verizon.com

Ning So
Reliance
United States of America

Email: Ning.So@ril.com

Michael Fiumano
Sprint
United States of America

Email: michael.f.fiumano@sprint.com

Mazen Khaddam
Cox
United States of America

Email: Mazen.Khaddam@cox.com

Jichun Ma
China Unicom
China

Email: majc16@chinaunicom.cn

Satoru Matsushima
Softbank
Japan

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

Francis Ferguson
CenturyLink
United States of America

Email: Francis.Ferguson@centurylink.com

Takuya Miyasaka
KDDI
Japan

Email: ta-miyasaka@kddi.com

Kentaro Ebisawa
Toyota Motor Corporation
Japan

Email: ebisawa@toyota-tokyo.tech

Yukito Ueno
NTT Communications Corporation
Japan

Email: yukito.ueno@ntt.com

Wim Henderickx
Nokia
Belgium

Email: wim.henderickx@nokia.com

Prem Jonnalagadda
Barefoot Networks
United States of America

Email: prem@barefootnetworks.com

Jisu Bhattacharya
Cisco Systems, Inc.
United States of America

Email: jisu@cisco.com

Kamran Raza
Cisco Systems, Inc.
Canada

Email: skraza@cisco.com

Internet-Draft Network Programming extension: SRv6 uSID instruct July 2019

Pablo Camarillo (editor)
Cisco Systems, Inc.
Spain

Email: pcamaril@cisco.com

SPRING
Internet-Draft
Intended status: Standards Track
Expires: 16 June 2022

C. Filsfils, Ed.
P. Camarillo, Ed.
Cisco Systems, Inc.
D. Cai
Alibaba
D. Voyer
Bell Canada
I. Meilik
Broadcom
K. Patel
Arrcus, Inc.
W. Henderickx
Nokia
P. Jonnalagadda
Barefoot Networks
D. Melman
Marvell
Y. Liu
China Mobile
J. Guichard
Futurewei
13 December 2021

Network Programming extension: SRv6 uSID instruction
draft-filsfils-spring-net-pgm-extension-srv6-usid-12

Abstract

The SRv6 "micro segment" (SRv6 uSID or uSID for short) instruction is a straightforward extension of the SRv6 Network Programming model:

- * The SRv6 Control Plane is leveraged without any change
- * The SRH dataplane encapsulation is leveraged without any change
- * Any SID in the SID list can carry micro segments
- * Based on the Compressed SRv6 Segment List Encoding in SRH [I-D.filsfilscheng-spring-srv6-srh-compression] framework

This enables:

- * ultra-scale (e.g. multi-domain 5G deployments)
- * minimum MTU overhead
- * installed-base reuse

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 16 June 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
2. Terminology	4
3. uSID Allocation within a uSID Block	5
3.1. GIB, LIB, global uSID and local uSID	5
3.1.1. Global uSID	5
3.1.2. Local uSID	5
3.1.3. Reference Illustration	5

4.	SRv6 behaviors associated with a uSID	6
4.1.	uSID behaviors related to the IGP	6
4.1.1.	uN	6
4.1.2.	uA	7
4.2.	uSID Behaviors related to BGP	8
4.2.1.	uDT	8
4.2.2.	uDX	9
5.	FIB entry at originating node for performant support of global-local sequence	10
6.	Routing	10
7.	Benefits	10
8.	Running code	12
8.1.	NANOG78 interoperability testing	12
8.2.	L3VPN interoperability testing with control-plane	13
8.3.	Dataplane traffic engineering interoperability testing .	13
9.	Security	14
10.	IANA Considerations	14
11.	Acknowledgements	15
12.	Contributors	15
13.	References	17
13.1.	Normative References	17
13.2.	Informative References	18
	Authors' Addresses	18

1. Introduction

SRv6 Network Programming [RFC8986] defines a mechanism to build a network program with topological and service segments. It leverages the SRH [RFC8754] to encode a network program together with optional metadata shared among the different SIDs.

This draft extends SRv6 Network Programming with a new type of SRv6 SID behaviors: SRv6 uN, uA, uDT, uDX.

This extension fully leverages the SRv6 network programming solution:

- * The SRv6 Control Plane is leveraged without any change
- * The SRH dataplane encapsulation is leveraged without any change
- * Any SID in the SID list can carry micro segments
- * Based on the Compressed SRv6 Segment List Encoding in SRH [I-D.filsfilscheng-spring-srv6-srh-compression] framework

This enables:

- * ultra-scale (e.g. multi-domain 5G deployments)

- * minimum MTU overhead
- * installed-base reuse

2. Terminology

The SRv6 Network Programming, SRH and Compressed SRv6 Segment List Encoding in SRH terminology is leveraged and extended with the following terms:

Term	Definition
uSID block	A block of uSID's. It can be any IPv6 prefix available to the provider.
uSID	A Compressed-SID. In this document a 16-bit ID. A different uSID length may be used.
Active uSID	First uSID after the uSID block.
Next uSID	Next uSID after the Active uSID.
Last uSID	From left to right, the last uSID before the first End-of-Container uSID.
End-of-Container	Reserved uSID used to mark the end of a uSID container. The value 0000 is selected as End-of-Container. All of the empty uSID container positions must be filled with the End-of-Container ID. Hence, the End-of-Container can be present more than once in a uSID container.
uSID container	A CSID container. A 128bit SRv6 SID of format <uSID-Block><Active-uSID><Next-uSID>...<Last-uSID><End-of-Container>...<End-of-Container>. A uSID container can be encoded in the Destination Address of an IPv6 header or at any position in the Segment List of an SRH.

Table 1

3. uSID Allocation within a uSID Block

3.1. GIB, LIB, global uSID and local uSID

GIB: The set of IDs available for global uSID allocation.

LIB: The set of IDs available for local uSID allocation.

3.1.1. Global uSID

A uSID from the GIB.

A Global uSID typically identifies a shortest-path to a node in the SR domain. An IP route (e.g., /64) is advertised by the parent node to each of its global uSID's, under the associated uSID block. The parent node executes a variant of the END behavior.

A node can have multiple global uSID's under the same uSID blocks (e.g. one per IGP flex-algorithm). Multiple nodes may share the same global uSID (anycast).

3.1.2. Local uSID

A uSID from the LIB.

A local uSID may identify a cross-connect to a direct neighbor over a specific interface or a VPN context.

No IP route is advertised by a parent node for its local uSID'.

If N1 and N2 are two different physical nodes of the uSID domain and I is a local uSID value, then N1 and N2 may bind two different behaviors to I.

3.1.3. Reference Illustration

For illustration simplicity, we will use:

- * uSID block length: 48 bits
- * uSID block: 2001:db8:0::/48
- * uSID length: 16 bits
- * uSID: 2001:db8:0:XYZW::/64
- * GIB: nibble X from hexa(0) to hexa(D)

- * LIB: nibble X hexa(E) or hexa(F)

Leveraging our reference illustration,

- * A uSID 2001:db8:0:XYZW::/64 is said to be allocated from its block (2001:db8:0::/48).
- * More specifically, a uSID is allocated from the GIB or LIB of block 2001:db8:0::/48 depending on the value of the "X" nibble: 0-D for GIB, and E-F for LIB.
- * With the above allocation scheme, the uSID Block 2001:db8:0::/48 supports up to 57k global uSID's (e.g. routers) while each router would support up to 8k local uSID's.

Another illustration could assume a 32-bit uSID length and a LIB restricted to the uSIDs with the first byte set to FF. In this context, the network as a whole would support $2^{32}-2^{24}$ global uSID's (e.g. routers) while each router would support up to 2^{24} local uSID's.

4. SRv6 behaviors associated with a uSID

The SRv6 SRH encapsulation and its network programming model are extended with the following functions:

4.1. uSID behaviors related to the IGP

4.1.1. uN

The uN is a short notation for the End behavior with NEXT-CSID, PSP and USD flavors as defined in [I-D.filsfilscheng-spring-srv6-srh-compression].

As a reminder the pseudo-code of the End behavior with NEXT-CSID flavor, when applied to a 48b uSID block and a 16b uSID length is as follows:

```
2001:db8:0:0N00::/64 bound to the pseudocode shift-and-lookup:
  1. Copy DA[64..127] into DA[48..111]                ;; Ref1
  2. Set DA[112..127] to 0x0000
  3. Forward the packet to the new DA
```

2001:db8:0:0N00::/80 bound to the End behavior with PSP & USD flavors

Ref 1: DA[X..Y] refers to the bits from position X to Y (included) in the IPv6 Destination Address of the received packet. The bit 0 is the MSB, while the bit 127 is the LSB.

4.1.1.1. Control-plane representation

In ISIS [I-D.ietf-lsr-isis-srv6-extensions], a uN is advertised with the following information:

- * Value = 2001:db8:0:0N00::
- * Behavior = uN
- * Structure =
 - LBL = 48
 - LNL = 16
 - FL = 0
 - AL = 64
- * Algorithm = 0 (or other)

4.1.2. uA

The uA local behavior is a short notation for the End.X behavior with NEXT-CSID, PSP and USD flavors [I-D.filsfilscheng-spring-srv6-srh-compression].

An instance of the uA SRv6 uSID behavior is associated with a set, J, of one or more Layer-3 adjacencies.

As a reminder the pseudo-code of the End.X behavior with NEXT-CSID flavor, when applied to a 48b uSID block and a 16b uSID length is as follows:

```
2001:db8:0:FNAJ::/64 bound to the pseudocode shift-and-xconnect:
1.  Copy DA[64..127] into DA[48..111]                ;; Ref1
2.  Set DA[112..127] to 0x0000
3.  Forward to layer-3 adjacency J
```

2001:db8:0:FNAJ::/80 bound to the End.X behavior w PSP & USD flavors

Ref 1: DA[X..Y] refers to the bits from position X to Y (included) in the IPv6 Destination Address of the received packet. The bit 0 is the MSB, while the bit 127 is the LSB.

4.1.2.1. Control-plane representation

In ISIS [I-D.ietf-lsr-isis-srv6-extensions], a uA is advertised with the following information:

- * Value = 2001:db8:0:0N00:FNAJ::
- * Behavior = uA
- * Structure =
 - LBL = 48
 - LNL = 16
 - FL = 16
 - AL = 48
- * Algorithm = 0 (or other)

Note: From a formal viewpoint, a uA SID of node N is defined by the local FIB entry B:uA/64 of N (i.e. this definition is independent from any uN SID of node N). In order to signal in ISIS a container SID with the same routable semantics as End.X, the ISIS advertisement of a uA SID is done as uN+uA. uN provides the global route to the node like the End behavior. uA provides the cross-connect function like the "X" of the End.X.

4.2. uSID Behaviors related to BGP

4.2.1. uDT

A local uDT behavior of Node D 2001:db8:0:FNVT:: is defined by the following single FIB entry and pseudo-code:

2001:db8:0:FNVT::/80 bound to the same pseudocode as End.DT4/End.DT6/
End.DT2*

4.2.1.1. Control-plane representation

In BGP [I-D.ietf-bess-srv6-services], a uDT is advertised with the following information:

- * Value = 2001:db8:0:0N00:FNVT::
- * Behavior = uDT

* Structure =

- LBL = 48
- LNL = 16
- FL = 16
- AL = 0
- TL = 16
- TO = 64

* Algorithm = 0 (or other)

Note: the advertised SID value includes the uN SRv6 uSID of the parent.

4.2.2. uDX

A local uDX behavior of Node D 2001:db8:0:FNXJ:: is defined by the following single FIB entry and pseudo-code:

2001:db8:0:FNXJ::/80 bound to the same pseudocode as End.DX4/End.DX6/
End.DX2

4.2.2.1. Control-plane representation

In BGP [I-D.ietf-bess-srv6-services], a uDX is advertised with the following information:

* Value = 2001:db8:0:0N00:FNXJ::

* Behavior = uDX

* Structure =

- LBL = 48
- LNL = 16
- FL = 16
- AL = 0
- TL = 16

- TO = 64

- * Algorithm = 0 (or other)

Note: the advertised SID value includes the uN SRv6 uSID of the parent.

5. FIB entry at originating node for performant support of global-local sequence

Any originating parent node may install the sequence of <Global, Local> uSID to perform more efficient processing given the LPM lookup.

For example, a parent node N that has the following FIB entries:

- * 2001:db8:0:0N00::/64 bound to the pseudocode shift-and-lookup
- * 2001:db8:0:0N00:0000::/80 bound to the End behavior with PSP&USD flavors
- * 2001:db8:0:FNAJ::/64 bound to the pseudocode shift-and-xconnect
- * 2001:db8:0:FNAJ:0000::/80 bound to the End.X behavior with PSP&USD flavors

may install the following additional FIB entries:

- * 2001:db8:0:0N00:FNAJ::/80 bound to the pseudocode shift-and-xconnect (with 32b shifting)
- * 2001:db8:0:0N00:FNAJ:0000::/96 bound to the End.X behavior with PSP&USD flavors

6. Routing

If Node 1 is configured with a uN SID 2001:db8:0:0100::/64 then the operator must ensure that Node 1 advertises 2001:db8:0:0100::/64 in the routing protocol.

7. Benefits

- * Leverages SRv6 Network Programming with NO change
 - SRv6 uSID is a flavor of the SRv6 network programming model
- * Leverages SRv6 dataplane (SRH) with NO change

- Any SID in DA or SRH can be an SRv6 uSID container
- * Leverages SRv6 Control-Plane with NO change
- * Ultra-Scale
 - 6 uSID's per uSID container
 - 18 source routing waypoints in only 40bytes of overhead
 - o H.Encaps.Red with an SRH of 40 bytes (8 fixed + 2 * 16 bytes)
 - o 6 uSID's in DA and 12 in SRH
- * Lowest MTU overhead
 - In apple to apple comparison, the SRv6 solution outperforms any alternative (VxLAN with SR-MPLS, CRH).
- * Scalable number of globally unique nodes in the domain
 - 16-bit uSID: 65k uSIDs per domain block
 - 32-bit uSID: 4.3M uSIDs per domain block
- * Proven Hardware-friendliness
 - Leverages mature hardware capabilities (Inline DA edit, DA longest match)
 - Avoids any extra lookup in indexed mapping table
 - Demonstrated by the number of linerate interoperable hardware implementations at the first Interop report in February 2020, less than 9 months after the first public version of this document.
 - Public operator report of leverage of installed base
 - A micro-program which requires less than 6 uSID's only requires legacy IPinIP encapsulation behavior
- * Scalable Control-Plane
 - No indexed mapping table is required

- 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: i.e. the container holds a single uSID
 - The inner structure of an SR Policy can stay opaque to the source: i.e. a container with uSID's is just seen as a SID by the policy headend
- * Security
 - Leverages SRv6's native SR domain security
- * Large-Scale DC
 - SID's may be used to address applications on hosts (scale in 2^{128})
 - Hardware friendliness of uSID's may be used to specify billions of waypoints in cost/power-optimized DC fabric

8. Running code

8.1. NANOG78 interoperability testing

The hardware and software platforms listed have participated in a joint interoperability testing of the uN instruction defined in this document.

Hardware implementations (in alphabetical order):

- * Arrcus ArcOS (based on Broadcom Jericho2)
- * Barefoot Tofino P4-programmable Ethernet switch ASIC
- * Cisco 8000 Series Routers (based on Cisco Silicon One Q100)
- * Cisco ASR9000 platform (with 3rd gen Tomahawk and 4th gen Lightspeed line-cards)
- * Cisco NCS5500 platform (based on Broadcom Jericho/Jericho+)

- * Marvell Prestera Packet Processor

Software open-source implementations (in alphabetical order):

- * FD.io VPP
- * Linux Kernel

Further details are available in the [NANOG78].

8.2. L3VPN interoperability testing with control-plane

In December 2020 the following routing platforms have participated in a successful interoperability testing including the uDT instruction and its BGP control-plane signalling.

- * Arrcus ArcOS
- * Cisco ASR9000 with IOS-XR
- * Cisco NCS5500 with IOS-XR
- * Cisco XRV9k with IOS-XR
- * FD.io VPP with GoBGP

Further details are available in [L3VPN-INTEROP].

8.3. Dataplane traffic engineering interoperability testing

In November 2020, the following hardware and software platforms have participated in a joint interoperability testing of the uN instruction defined in this document. This interoperability testing was hosted by China Mobile.

- * Hardware implementation in Cisco ASR 9000 running IOS XR
- * Software implementation in Cisco IOS XRV9000 virtual appliance
- * Hardware implementation in Huawei NE40E running VRP
- * Hardware implementation in Huawei NE5000E running VRP

Further details are available in [I-D.filsfilscheng-spring-srv6-srh-compression] Section 11.

9. Security

The security rules defined in Section 7 of [RFC8986], protect intra-domain deployments that includes SRv6 uSID.

10. IANA Considerations

This document requests IANA to allocate the following codepoints within the "SRv6 Endpoint Behaviors" sub-registry under the top-level "Segment Routing Parameters" registry.

Value	Hex	Endpoint behavior	Reference
42	0x002A	End with NEXT-ONLY-CSID	[This.ID]
43	0x002B	End with NEXT-CSID	[This.ID]
44	0x002C	End with NEXT-CSID & PSP	[This.ID]
45	0x002D	End with NEXT-CSID & USP	[This.ID]
46	0x002E	End with NEXT-CSID, PSP & USP	[This.ID]
47	0x002F	End with NEXT-CSID & USD	[This.ID]
48	0x0030	End with NEXT-CSID, PSP & USD	[This.ID]
49	0x0031	End with NEXT-CSID, USP & USD	[This.ID]
50	0x0032	End with NEXT-CSID, PSP, USP & USD	[This.ID]
51	0x0033	End.X with NEXT-ONLY-CSID	[This.ID]
52	0x0034	End.X with NEXT-CSID	[This.ID]
53	0x0035	End.X with NEXT-CSID & PSP	[This.ID]
54	0x0036	End.X with NEXT-CSID & USP	[This.ID]
55	0x0037	End.X with NEXT-CSID, PSP & USP	[This.ID]
56	0x0038	End.X with NEXT-CSID & USD	[This.ID]
57	0x0039	End.X with NEXT-CSID, PSP & USD	[This.ID]
58	0x003A	End.X with NEXT-CSID, USP & USD	[This.ID]

59	0x003B	End.X with NEXT-CSID, PSP, USP & USD	[This.ID]
60	0x003C	End.DX6 with NEXT-CSID	[This.ID]
61	0x003D	End.DX4 with NEXT-CSID	[This.ID]
62	0x003E	End.DT6 with NEXT-CSID	[This.ID]
63	0x003F	End.DT4 with NEXT-CSID	[This.ID]
64	0x0040	End.DT46 with NEXT-CSID	[This.ID]
65	0x0041	End.DX2 with NEXT-CSID	[This.ID]
66	0x0042	End.DX2V with NEXT-CSID	[This.ID]
67	0x0043	End.DT2U with NEXT-CSID	[This.ID]
68	0x0044	End.DT2M with NEXT-CSID	[This.ID]

Table 2: IETF - SRv6 Endpoint Behaviors

11. Acknowledgements

The authors would like to acknowledge Francois Clad, Peter Psenak, Ketan Talaulikar, Jakub Horn, Swadesh Agrawal, Zafar Ali, Darren Dukes, Kiran Sasidharan, Junaid Israr, Lakshmanan Srikanth, Asif Islam, Saleem Hafeez, Michael MacKenzie, Sushek Shekar, YuanChao Su, Alexander Preusche, Alberto Donzelli, Miya Kohno, David Smith, Ianik Semco, Bertrand Duvivier, Frederic Trate, Kris Michielsen, Eyal Dagan, Eli Stein, Ofer Iny, Elad Naor, Guy Caspari, Mel Tsai, Anand Sridharan, Aviad Behar, Joseph Chin.

12. Contributors

Jisu Bhattacharyaa Cisco Systems, Inc. United States of America

Email: jisu@cisco.com

Kamran Raza Cisco Systems, Inc. Canada

Email: skraza@cisco.com

John Bettink Cisco Systems, Inc. United States of America

Email: jbettink@cisco.com

Tomonobu Niwa KDDI Japan

Email: to-niwa@kddi.com

Luay Jalil Verizon United States of America

Email: luay.jalil@one.verizon.com

Zhichun Jiang Tencent China

Email: zcjiang@tencent.com

Ahmed Shawky Saudi Telecom Company Saudi Arabia

Email: ashawky@stc.com.sa

Nic Leymann Deutsche Telekom Germany

Email: N.Leymann@telekom.de

Dirk Steinberg Lapishills Consulting Limited Cyprus

Email: dirk@lapishills.com

Shawn Zandi LinkedIn United States of America

Email: szandi@linkedin.com

Gaurav Dawra LinkedIn United States of America

Email: gdawra@linkedin.com

Jim Uttaro AT&T United States of America

Email: jul738@att.com

Ning So Reliance United States of America

Email: Ning.So@ril.com

Michael Fiumano Sprint United States of America

Email: michael.f.fiumano@sprint.com

Mazen Khaddam Cox United States of America

Email: Mazen.Khaddam@cox.com

Jichun Ma China Unicom China

Email: majc16@chinaunicom.cn

Satoru Matsushima Softbank Japan

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

Francis Ferguson CenturyLink United States of America

Email: Francis.Ferguson@centurylink.com

Takuya Miyasaka KDDI Japan

Email: ta-miyasaka@kddi.com

Kentaro Ebisawa Toyota Motor Corporation Japan

Email: ebisawa@toyota-tokyo.tech

Yukito Ueno NTT Communications Corporation Japan

Email: yukito.ueno@ntt.com

13. References

13.1. Normative References

- [I-D.filsfilscheng-spring-srv6-srh-compression]
Cheng, W., Filsfils, C., Li, Z., Decraene, B., Cai, D.,
Voyer, D., Clad, F., Zadok, S., Guichard, J. N., Aihua,
L., Raszuk, R., and C. Li, "Compressed SRv6 Segment List
Encoding in SRH", Work in Progress, Internet-Draft, draft-
filsfilscheng-spring-srv6-srh-compression-02, 28 July
2021, <[https://www.ietf.org/archive/id/draft-
filsfilscheng-spring-srv6-srh-compression-02.txt](https://www.ietf.org/archive/id/draft-filsfilscheng-spring-srv6-srh-compression-02.txt)>.
- [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>>.

- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", RFC 8986, DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.

13.2. Informative References

- [I-D.ietf-bess-srv6-services]
Dawra, G., Filsfils, C., Talaulikar, K., Raszuk, R., Decraene, B., Zhuang, S., and J. Rabadan, "SRv6 BGP based Overlay Services", Work in Progress, Internet-Draft, draft-ietf-bess-srv6-services-08, 10 November 2021, <<https://www.ietf.org/archive/id/draft-ietf-bess-srv6-services-08.txt>>.
- [I-D.ietf-lsr-isis-srv6-extensions]
Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extensions to Support Segment Routing over IPv6 Dataplane", Work in Progress, Internet-Draft, draft-ietf-lsr-isis-srv6-extensions-18, 20 October 2021, <<https://www.ietf.org/archive/id/draft-ietf-lsr-isis-srv6-extensions-18.txt>>.
- [L3VPN-INTEROP]
Cisco Systems, Inc. and Arrcus, "SRv6 uSID L3VPN Interopability Testing", L3VPN Interop , December 2020, <<https://www.segment-routing.net/demos/2020-12-22-SRv6-uSID-L3VPN-interopability/>>.
- [NANOG78] Filsfils, C., "SRv6 Technology and Deployment Use-cases", NANOG78 , February 2020, <https://storage.googleapis.com/site-media-prod/meetings/NANOG78/2097/20200212_Mcdougall_Srv6_Technology_And_v1.pdf>.

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

Dennis Cai
Alibaba
China

Email: d.cai@alibaba-inc.com

Daniel Voyer
Bell Canada
Canada

Email: daniel.voyer@bell.ca

Israel Meilik
Broadcom
Israel

Email: israel.meilik@broadcom.com

Keyur Patel
Arrcus, Inc.
United States of America

Email: keyur@arrcus.com

Wim Henderickx
Nokia
Belgium

Email: wim.henderickx@nokia.com

Prem Jonnalagadda
Barefoot Networks
United States of America

Email: prem@barefootnetworks.com

David Melman
Marvell
Israel

Email: davidme@marvell.com

Yisong Liu
China Mobile
China

Email: liuyisong@chinamobile.com

James Guichard
Futurewei
United States of America

Email: james.n.guichard@futurewei.com

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 27, 2019

R. Gandhi, Ed.
Z. Ali
C. Filsfils
F. Brockners
Cisco Systems, Inc.
B. Wen
V. Kozak
Comcast
April 25, 2019

Segment Routing with MPLS Data Plane Encapsulation
for In-situ OAM Data
draft-gandhi-spring-ioam-sr-mpls-01

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the data packet while the packet traverses a path between two points in the network. This document defines how IOAM data fields are transported with the Segment Routing with MPLS data plane (SR-MPLS) encapsulation.

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 <http://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."

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions	3
2.1. Requirement Language	3
2.2. Abbreviations	3
3. IOAM Data Field Encapsulation in SR-MPLS Header	3
4. Procedure	4
4.1. IOAM Indicator Label	5
4.2. Hashing Function	5
4.3. Node Capability	5
5. IANA Considerations	5
6. Security Considerations	6
7. Acknowledgements	6
8. Normative References	7
Contributors	7
Authors' Addresses	7

1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within probe packets specifically dedicated to OAM.

Segment Routing (SR) leverages the source routing paradigm [I-D.ietf-spring-segment-routing-mpls]. A node steers a packet through a controlled set of instructions, called segments, by prepending the packet with an SR header. In the MPLS data plane, the SR header is instantiated through a label stack. This document defines how IOAM data fields are transported with the SR with MPLS data plane (SR-MPLS) encapsulation.

The IOAM data fields carried are defined in [I-D.ietf-ippm-ioam-data], and can be used for various use-cases including Performance Measurement (PM).

2. Conventions

2.1. Requirement Language

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

2.2. Abbreviations

Abbreviations used in this document:

IOAM	In-situ Operations, Administration, and Maintenance
OAM	Operations, Administration, and Maintenance
PM	Performance Measurement
PoT	Proof-of-Transit
SR	Segment Routing
SR-MPLS	Segment Routing with MPLS Data plane

3. IOAM Data Field Encapsulation in SR-MPLS Header

SR-MPLS encapsulation is defined in [I-D.ietf-spring-segment-routing-mpls]. IOAM data fields are carried in the SR-MPLS header, as IOAM data fields. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as TLVs. More than one TLVs can be present in the IOAM data fields. The IOAM Indicator Label is added at the bottom of the MPLS label stack (S flag set to 1) to indicate the presence of the IOAM data fields in the MPLS header.

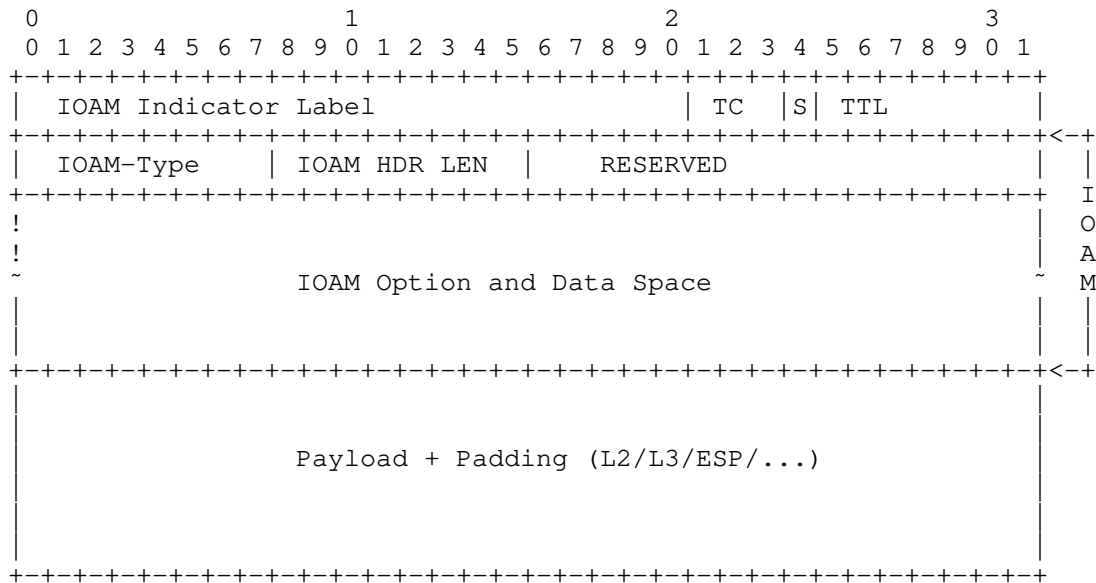


Figure 1: IOAM data encapsulation in SR-MPLS Header

IOAM Indicator Label as defined in Section 4.1.

The fields related to the encapsulation of IOAM data fields in the SR-MPLS header are defined as follows:

IOAM-Type: 8-bit field defining the IOAM Option type, as defined in Section 4 of [I-D.ietf-ippm-ioam-data].

IOAM HDR LEN: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

RESERVED: 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.

IOAM Option and Data Space: IOAM option header and data is present as defined by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

4. Procedure

This section summarizes the procedure for IOAM data encapsulation in SR-MPLS.

- o The ingress node inserts the IOAM Indicator Label and one or more

IOAM TLV(s) in the MPLS header.

- o The egress node "forwards and punts the timestamped copy" of the data packet including IOAM TLVs when the node recognizes the IOAM Indicator Label.
- o The egress node also pops the IOAM Indicator Label and the IOAM TLVs from the MPLS header.

4.1. IOAM Indicator Label

The IOAM Indicator Label can be allocated using one of the following methods:

- o Label assigned by IANA with value TBA1.
- o Label allocated by a controller from the global table of the egress node. The controller provisions the label on both ingress and egress nodes.
- o Label allocated by the egress node from the global label table of the egress node. The signaling extension for this is outside the scope of this document.

4.2. Hashing Function

The ingress node needs to make sure the IOAM TLV does not start with a well known protocol type (e.g. 0x4 for IPv4 and 0x6 for IPv6) and does not alter the hashing function that uses the IP header.

Note that the hashing function that uses the label values from the MPLS header may also now include the IOAM Indicator Label.

4.3. Node Capability

The egress node that has to pop the IOAM Indicator Label, TLVs, and perform the IOAM function may not be capable of supporting it. The ingress node needs to know if the egress node can support the IOAM function. The signaling extension for this capability exchange is outside the scope of this document.

5. IANA Considerations

IANA maintains the "Special-Purpose Multiprotocol Label Switching (MPLS) Label Values" registry (see <https://www.iana.org/assignments/mpls-label-values/mpls-label-values.xml>). IANA is requested to allocate IOAM Indicator Label

value from the "Special-Purpose MPLS Label Values" registry:

Value	Description	Reference
TBA1	IOAM Indicator Label	This document

6. Security Considerations

The security considerations of SR-MPLS are discussed in [I-D.ietf-spring-segment-routing-mpls], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

7. Acknowledgements

The authors would like to thank Shwetha Bhandari and Vengada Prasad Govindan for the discussions on IOAM.

8. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.
- [I-D.ietf-spring-segment-routing-mpls] Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls, work in progress.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and Bernier, D., "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data, work in progress.

Contributors

Sagar Soni
Cisco Systems, Inc.
Email: sagsoni@cisco.com

Patrick Khordoc
Cisco Systems, Inc.
Email: pkhordoc@cisco.com

Authors' Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Zafar Ali
Cisco Systems, Inc.

Email: zali@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

Voitek Kozak
Comcast

Email: Voitek_Kozak@comcast.com

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 23, 2020

R. Gandhi, Ed.
Z. Ali
C. Filsfils
F. Brockners
Cisco Systems, Inc.
B. Wen
V. Kozak
Comcast
August 22, 2019

Segment Routing with MPLS Data Plane Encapsulation
for In-situ OAM Data
draft-gandhi-spring-ioam-sr-mpls-02

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the data packet while the packet traverses a path between two nodes in the network. Segment Routing (SR) technology leverages the source routing paradigm. This document defines how IOAM data fields are transported with the Segment Routing with MPLS data plane (SR-MPLS) encapsulation. The procedures defined are also equally applicable to all other MPLS data plane encapsulations.

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 <http://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."

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Conventions	3
2.1. Requirement Language	3
2.2. Abbreviations	3
3. IOAM Data Field Encapsulation in SR-MPLS Header	3
4. Procedure for Edge-to-Edge IOAM	5
4.1. Edge-to-Edge IOAM Indicator Labels	6
5. Procedure for Hop-by-Hop IOAM	7
6. Considerations for ECMP	7
7. Node Capability	7
8. IANA Considerations	7
9. Security Considerations	8
10. Acknowledgements	8
11. References	8
11.1. Normative References	8
11.2. Informative References	9
Contributors	9
Authors' Addresses	9

1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within the probe packets specifically dedicated to OAM or Performance Measurement (PM). The IOAM data fields are defined in [I-D.ietf-ippm-ioam-data], and can be used for various use-cases for OAM and PM.

Segment Routing (SR) technology leverages the source routing paradigm [I-D.ietf-spring-segment-routing-mpls]. A node steers a packet through a controlled set of instructions, called segments, by pre-pending the packet with an SR header. In the MPLS data plane,

the SR header is instantiated through a label stack.

This document defines how IOAM data fields are transported with the SR with MPLS data plane (SR-MPLS) encapsulation. The procedures defined are also equally applicable to all other MPLS data plane encapsulations.

2. Conventions

2.1. Requirement Language

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

2.2. Abbreviations

Abbreviations used in this document:

ECMP	Equal Cost Multi-Path
IOAM	In-situ Operations, Administration, and Maintenance
MPLS	Multiprotocol Label Switching
OAM	Operations, Administration, and Maintenance
PBT	Postcard Based Telemetry
PM	Performance Measurement
PoT	Proof-of-Transit
SR	Segment Routing
SR-MPLS	Segment Routing with MPLS Data plane

3. IOAM Data Field Encapsulation in SR-MPLS Header

SR-MPLS encapsulation is defined in [I-D.ietf-spring-segment-routing-mpls]. The IOAM data fields are defined in [I-D.ietf-ippm-ioam-data]. IOAM data fields are carried in the SR-MPLS header as shown in Figure 1 and Figure 2. More than one trace options can be present in the IOAM data fields. The Indicator Label is added at the bottom of the MPLS label stack (S

flag set to 1) to indicate the presence of the IOAM data field(s) in the MPLS header.

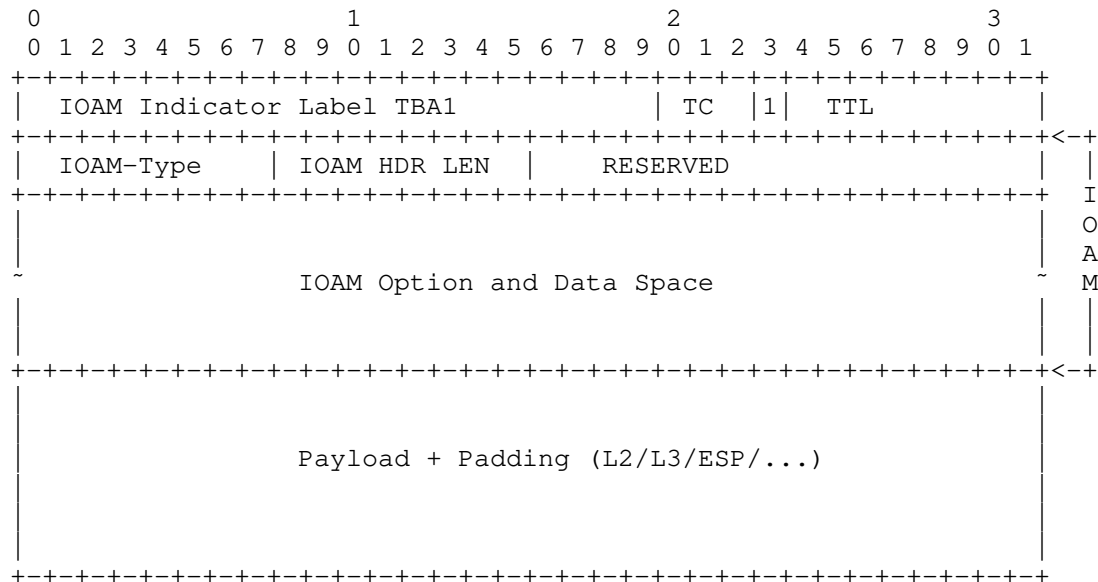
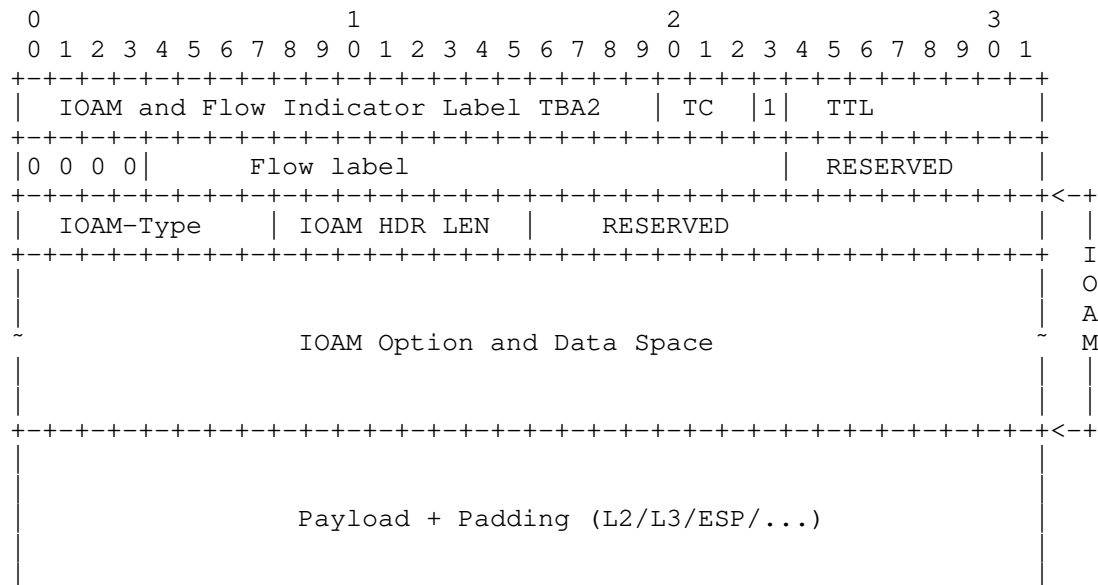


Figure 1: IOAM data encapsulation in SR-MPLS Header



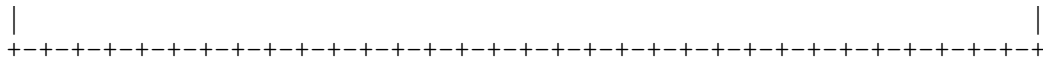


Figure 2: IOAM data encapsulation with Flow Label in SR-MPLS Header

Indicator Label and Flow Label as defined in this document.

The fields related to the encapsulation of IOAM data fields in the SR-MPLS header are defined as follows:

IOAM-Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR LEN: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

RESERVED: 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.

IOAM Option and Data Space: IOAM option header and data is present as defined by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

4. Procedure for Edge-to-Edge IOAM

This section summarizes the procedure for data encapsulation and decapsulation for IOAM Edge-to-Edge Option Type [I-D.ietf-ippm-ioam-data] in SR-MPLS header.

- o The encapsulating node inserts the IOAM Indicator Label or IOAM Flow Indicator Label with Flow Label and one or more IOAM data field(s) in the MPLS header. The procedure to generate the Flow Label is outside the scope of this document.
- o The decapsulating node "forwards and punts the timestamped copy" of the data packet including IOAM data fields when the node recognizes the IOAM Indicator Label and IOAM Flow Indicator Label. The copy of the data packet is punted to the slow path for OAM processing and is not necessarily punted to the control-plane. The receive timestamp is required by various OAM use-cases.
- o The decapsulating node processes the IOAM data field(s) using the procedures defined in [I-D.ietf-ippm-ioam-data]. An example of IOAM processing may be to export the data fields, send data fields via Telemetry, etc.

- o The decapsulating node also pops the Indicator Label and the IOAM data fields from the MPLS header.

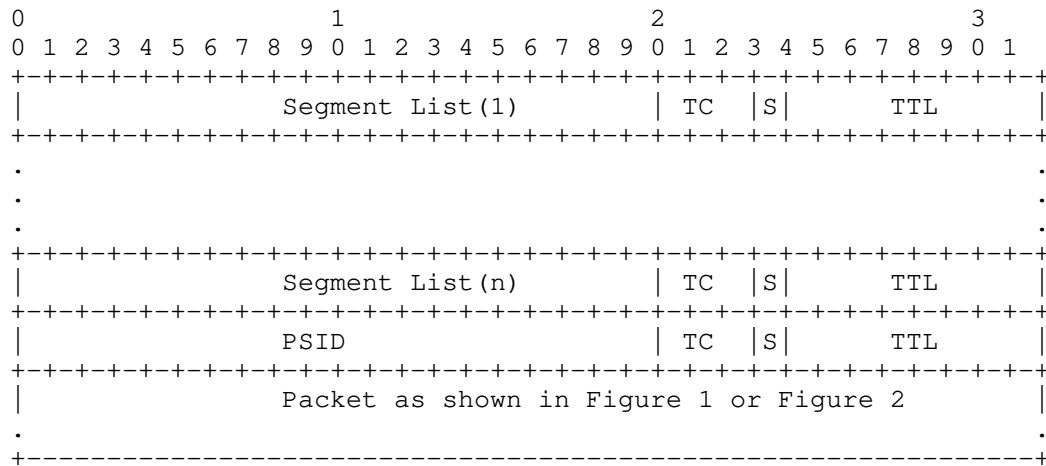


Figure 3: Data Packet over SR-MPLS Policy

4.1. Edge-to-Edge IOAM Indicator Labels

IOAM Indicator Label (value TBA1) and IOAM and Flow Indicator Label (value TBA2) are used to indicate the presence of the IOAM data field in the MPLS header.

The Indicator Label with value TBA2 is used to carry a second label underneath with protocol value 0000b and 20-bit Flow Label. The protocol value 0000b allows to avoid incorrect IP header based hashing over ECMP paths that uses the value 0x4 (for IPv4) and value 0x6 (for IPv6) [RFC4928]. The Flow Label identifies the traffic flow that can be used for IOAM purpose as well as for hashing over ECMP paths.

The IOAM Indicator Label and IOAM and Flow Indicator Label can be allocated using one of the following methods:

- o Labels assigned by IANA with value TBA1 and TBA2 from the Extended Special-Purpose MPLS Values [mpls-spl-terminology].
- o Labels allocated by a controller from the global table of the decapsulating node. The controller provisions the label on both encapsulating and decapsulating nodes.
- o Labels allocated by the decapsulating node. The signaling

extension for this is outside the scope of this document.

5. Procedure for Hop-by-Hop IOAM

The hop-by-hop IOAM includes IOAM-Types IOAM Pre-allocated Trace Option Type, IOAM Incremental Trace Option Type and IOAM POT Option Type.

Different Indicator Labels (TBA3 and TBA4) are used for hop-by-hop IOAM.

The details for hop-by-hop IOAM will be added in a future version of the document.

6. Considerations for ECMP

The encapsulating node needs to make sure the IOAM data field does not start with a well known IP protocol value (e.g. 0x4 for IPv4 and 0x6 for IPv6) as it can alter the hashing function for ECMP that uses the IP header. This can be achieved by using the IOAM and Flow Indicator Label (value TBA2 and TBA4) that follows by protocol value 0000b. This approach is consistent with the use of utilizing 0000b as the first nibble after the MPLS label stack, as described in [RFC4928] [RFC4385].

Note that the hashing function for ECMP that uses the labels from the MPLS header may also now include the Indicator Label.

The entropy label can be used for hashing function for ECMP as defined in [RFC6790].

7. Node Capability

The decapsulating node that has to pop the Indicator Label, data fields, and perform the IOAM function may not be capable of supporting it. The encapsulating node needs to know if the decapsulating node can support the IOAM function. The signaling extension for this capability exchange is outside the scope of this document.

8. IANA Considerations

IANA maintains the "Special-Purpose Multiprotocol Label Switching (MPLS) Label Values" registry (see

<<https://www.iana.org/assignments/mpls-label-values/mpls-label-values.xml>>). IANA is requested to allocate IOAM Indicator Label value and IOAM and Flow Indicator value from the "Extended Special-Purpose MPLS Label Values" registry:

Value	Description	Reference
TBA1	E2E IOAM Indicator Label	This document
TBA2	E2E IOAM and Flow Indicator Label	This document
TBA3	HbH IOAM Indicator Label	This document
TBA4	HbH IOAM and Flow Indicator Label	This document

9. Security Considerations

The security considerations of SR-MPLS are discussed in [I-D.ietf-spring-segment-routing-mpls], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

10. Acknowledgements

The authors would like to thank Shwetha Bhandari and Vengada Prasad Govindan for the discussions on IOAM. The authors would also like to thank Tarek Saad, Loa Andersson and Cheng Li for providing many useful comments.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.

- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.
- [I-D.ietf-spring-segment-routing-mpls] Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls, work in progress.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and Bernier, D., "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data, work in progress.

11.2. Informative References

- [RFC4385] Bryant, S., Swallow, G., Martini, L., and D. McPherson, "Pseudowire Emulation Edge-to-Edge (PWE3) Control Word for Use over an MPLS PSN", RFC 4385, February 2006.
- [RFC4928] Swallow, G., Bryant, S., and L. Andersson, "Avoiding Equal Cost Multipath Treatment in MPLS Networks", BCP 128, RFC 4928, June 2007.
- [RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, November 2012.
- [mpls-spl-terminology] L. Andersson, et al. "Special Purpose Label terminology", draft-ietf-mpls-spl-terminology, work in progress.

Contributors

Sagar Soni
Cisco Systems, Inc.
Email: sagsoni@cisco.com

Patrick Khordoc
Cisco Systems, Inc.
Email: pkhordoc@cisco.com

Authors' Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Zafar Ali
Cisco Systems, Inc.

Email: zali@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Bin Wen
Comcast

Email: Bin_Wen@cable.comcast.com

Voitek Kozak
Comcast

Email: Voitek_Kozak@comcast.com

SPRING
Internet-Draft
Intended status: Standards Track
Expires: October 21, 2019

S. Hegde
S. Sangli
Juniper Networks Inc.
X. Xu
Alibaba Inc.
April 19, 2019

BGP-LS Extensions for Inter-AS TE using EPE based mechanisms
draft-hegde-idr-bgp-ls-epe-inter-as-01

Abstract

In certain network deployments, a single operator has multiple Autonomous Systems (AS) to facilitate ease of management. A multiple AS network design could also be a result of network mergers and acquisitions. In such scenarios, a centralized Inter-domain TE approach could provide most optimal allocation of resources and the most controlled path placement. BGP-LS-EPE [I-D.ietf-idr-bgp-ls-segment-routing-epe] describes an extension to BGP Link State (BGP-LS) for the advertisement of BGP Peering Segments along with their BGP peering node and inter-AS link information, so that efficient BGP Egress Peer Engineering (EPE) policies and strategies can be computed based on Segment Routing. This document describes extensions to the BGP-LS EPE to enable it to be used for inter-AS Traffic-Engineering (TE) purposes.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 October 21, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Reference Topology	3
3. Fast Reroute Label	4
4. TE Link attributes of PeerNode SID	5
5. Example Advertisements	6
6. Backward Compatibility	8
7. Security Considerations	8
8. IANA Considerations	8
9. Acknowledgements	8
10. References	8
10.1. Normative References	8
10.2. Informative References	9
Authors' Addresses	9

1. Introduction

Segment Routing (SR) leverages source routing. A node steers a packet through a controlled set of instructions, called segments, by prepending the packet with an SR header with segment identifiers (SID). A SID can represent any instruction, topological or service-based. SR segments allows to enforce a flow through any topological path or service function while maintaining per-flow state only at the ingress node of the SR domain.

As there is no per-path state in the network, the bandwidth management for the paths is expected to be handled by a centralized entity which has a complete view of:

1. Up-to-date topology of the network

2. Resources, States and Attributes of links and nodes of the network

3. Run-time utilization/availability of resources

The BGP Link-State extensions provide mechanisms whereby link-state and TE information can be propagated in a network and a consumer of such BGP LS updates may build topology, provide bandwidth calendaring and other traffic engineering services. The centralized entity can be such a consumer (also referred to as controller). In the case of multi-AS networks, the controller needs to learn the per-AS network information and the inter-AS link information thus arriving at a consolidated Traffic Engineering Database which can be used to compute end-to-end Traffic Engineering Path. The controller can learn the topology, link-state and TE information from each of the AS networks either by participating in their IGP or by listening to BGP LS updates [RFC7752]. Similar information about the inter-AS links can be learnt via BGP-LS EPE [I-D.ietf-idr-bgpls-segment-routing-epe] along with extensions defined in this document.

2. Reference Topology

The controller learns TE attributes of all the links, including the inter-AS links and uses the attributes to compute constrained paths. The controller should be able to correlate the inter-AS links for bidirectional connectivity from both ASes.

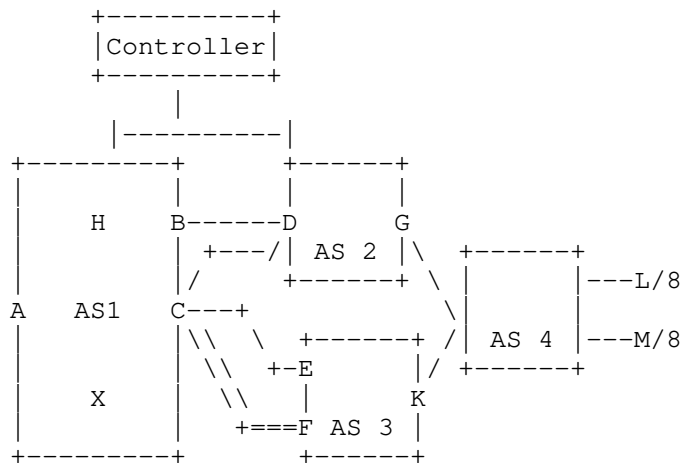


Figure 1: Reference Diagram

The reference diagram from [I-D.ietf-spring-segment-routing-central-epe] represents multiple Autonomous Systems connected to each other. When the Multiple ASes belong to same operator and are organised into separate domains for operational purposes, it is advantageous to support Traffic-Engineering across the ASes including the inter-AS links. The controller has visibility of all of the ASes by means of IGP topology exported via BGP-LS [RFC7752], or other means. In addition, the inter-AS links and the labels associated with the inter-AS links are exported via [I-D.ietf-idr-bgpls-segment-routing-epe]. The controller needs to correlate the information acquired from all of the ASes, including the inter-AS links in order to get a view of the unified topology so that it can build end-to-end Traffic-Engineered paths.

3. Fast Reroute Label

[I-D.ietf-spring-segment-routing-central-epe] section 3.6 describes mechanisms to provide Fast Reroute (FRR) protection for the EPE Labels. The BGP-LS EPE [I-D.ietf-idr-bgpls-segment-routing-epe] describes "B" bit to indicate that a PeerNodeSID or PeerAdjSID is eligible for backup. However, it does not specify what is the behaviour when the failure kicks-in. The controller needs to know which links are used for protection so that admission control and failure simulation can be done effectively and appropriate inter-AS links used for path construction.

This document defines a new flag "F" in the Peering SIDs TLV to indicate a SID as an FRR SID. With the "F" flag set, the protection for any peering SID can be specified using another PeerAdjSID, PeerNodeSID or PeerSetSID to the controller. If the protection is achieved by fallback to local IP lookup, FRR SID SHOULD not be advertised. The link(s) represented by the FRR SID will carry the traffic when there is a failure. These SIDs are included as an FRR SIDs in the peerAdjSID, PeerNodeSID and PeerSetSID advertisements.

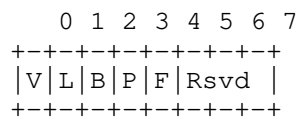


Figure 2: Peering SID TLV Flags Format

* F-Flag: FRR Label Flag: If set, the peer SID where the FRR Label appears is using backup links represented by FRR Label.

4. TE Link attributes of PeerNode SID

In any eBGP deployment, the peering session can be either single-hop or multi-hop. For single-hop eBGP sessions, the peering address is that of the directly attached interface to which the session is pinned down. For multi-hop eBGP session, the peering address is reachable over more than one interface and that the peering session is not pinned down to any of the directly attached interfaces.

A Peer Node Segment is a segment describing a peer, including the SID (PeerNodeSID) allocated to it. The link descriptors for the PeerNodeSID include the addresses used by the BGP session encoded using TLVs as defined in [RFC7752]. Since the eBGP session can be either single-hop or multi-hop, the IP address used by BGP session as local/neighbour is not sufficient to identify the underlying interface(s). Also, the controller needs to know the links associated with the PeerNodeSID, to be able derive TE link attributes. This can be achieved by including the interface local and remote addresses in the Link attributes in PeerNodeSID NLRI.

PeerAdjSID MUST be advertised for each inter-AS link for the purposes of inter-AS TE. The PeerAdjSID should contain link TE attributes such as bandwidth, admin-group etc. The PeerAdjSID should also contain the local and remote interface IPv4/IPv6 addresses which is used for correlating the links. PeerNodeSID SHOULD contain the additional attribute of link local address which is used by the controller to find corresponding PeerAdjSID and hence the corresponding link TE attributes.

A peerAdj segment carries mandatory link descriptors as local and remote link id. Remote link id of the neighboring ASBR is not readily available. [I-D.ietf-idr-bgpls-segment-routing-epe] suggests to carry the value '0' for the remote link id. The Controller needs to associate the links in both directions to effectively handle failure notifications and for this purpose a unique remote link is necessary. The remote link ID cannot be manually configured on the router as the link-ids generally change over router reboot etc and hence manual configuration is operationally very difficult to manage. This document mandates advertisement of local and remote interface addresses for the inter-AS TE purposes.

The Unnumbered interface is not in the scope of this document.

TLV Code Point	Description	IS-IS TLV /Sub-TLV	Reference (RFC/Section)
259	IPv4 Local interface Address	22/6	[RFC5305]/3.2
261	IPv6 Local interface Address	22/12	[RFC6119]/4.2

Figure 3: Link Addresses carried as attributes

5. Example Advertisements

The below diagram represents two ASBR routers and inter-AS links between them. The inter-AS links could be connected via switches L1 and L2 as shown in the diagram or via Point-to-point links A2->B2, A3->B3 as shown in the diagram below. In the below example, lets assume peerNodeSID 1 is configured to use peerAdjSID 10002 then PeerNodeSID 1 will have the B bit set which means the PeerNodeSID 1 is eligible for backup. Label 10002 is added to the PeerNodeSID with a "F" bit set, which means 10002 is a backup for PeerNodeSID 1.

PeerNodeSID				PeerAdjSID			
N	Loc Node Descr:	AS1:A		N	Loc Node Descr:	AS1:A	
L	Rmt Node Descr:	AS2:B		L	Rmt Node Descr:	AS2:B	
R	Link Descr:	lo1:lo1		R	Link Descr LinkLocRmtID:	1:0	
I				I	Link IP (mandatory):	A1:B1	
A	Link IP (new):	A1:B1		A	PeerAdjSID:	10001	
T	Link IP (new):	A2:B2		T	SRLG		
T	PeerNodeSID:	1		T	affinity group		
R	PeerSetSID (optional)			R	MaxB/W		
N	Loc Node Descr:	AS1:A		N	Loc Node Descr:	AS1:A	
L	Rmt Node Descr:	AS2:B		L	Rmt Node Descr:	AS2:B	
R	Link Descr:	lo2:lo2		R	Link Descr LinkLocRmtID:	2:0	
I				I	Link IP (mandatory):	A2:B2	
A	Link IP (new):	A1:B1		A	PeerAdjSID:	10002	
T	Link IP (new):	A3:B3		T	SRLG		

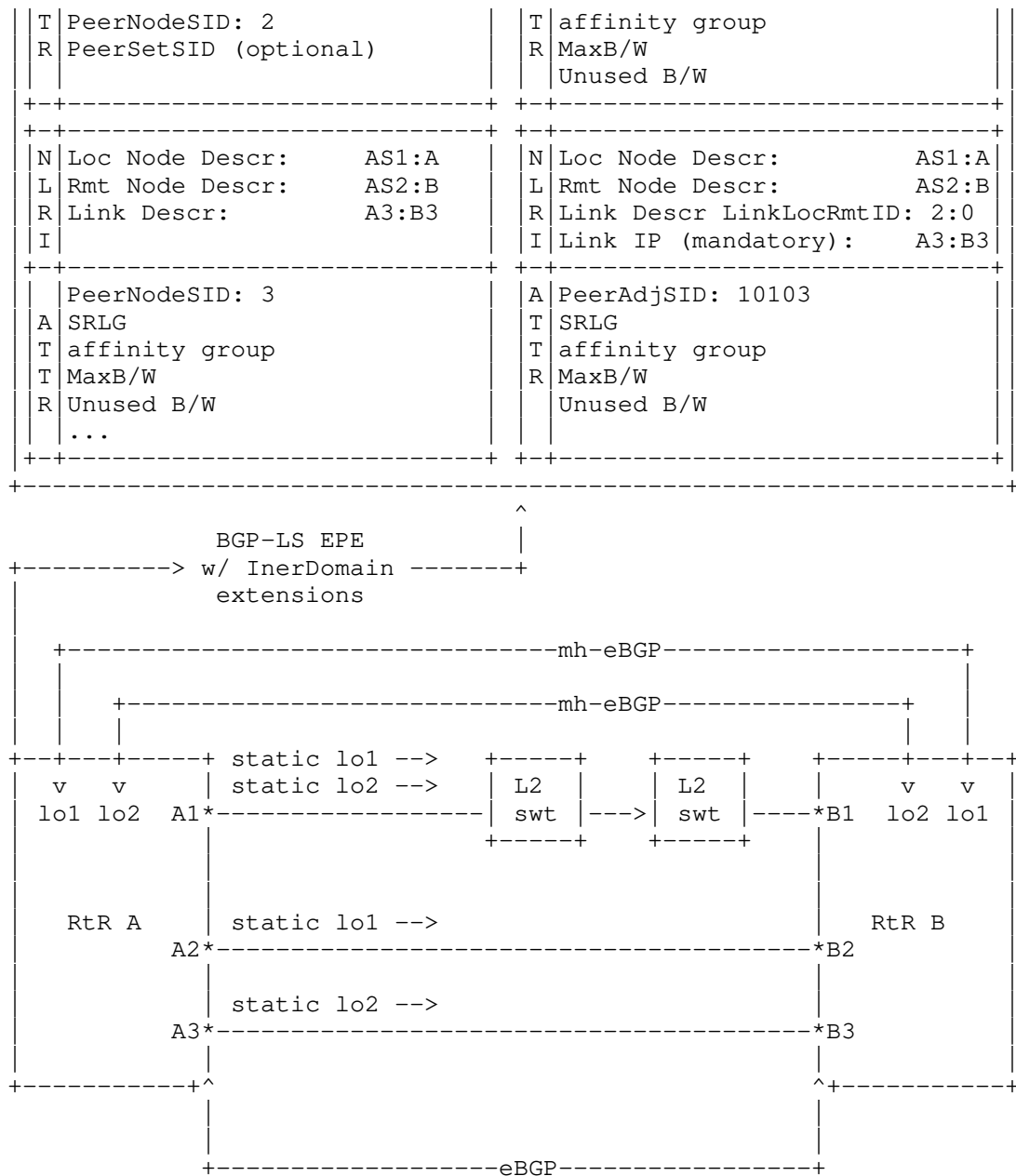


Figure 4: Example Advertisements

6. Backward Compatibility

The extension proposed in this document is backward compatible with procedures described in [I-D.ietf-idr-bgpls-segment-routing-epe] and [I-D.ietf-spring-segment-routing-central-epe]

7. Security Considerations

TBD

8. IANA Considerations

No new TLV code points are needed.

9. Acknowledgements

Thanks to Julian Lucek and Rafal Jan Szarecki for careful review and suggestions.

10. References

10.1. Normative References

- [I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-18 (work in progress), March 2019.
- [I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.
- [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>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<https://www.rfc-editor.org/info/rfc7752>>.

10.2. Informative References

[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

Shraddha Hegde
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Srihari Sangli
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: ssangli@juniper.net

Xiaohu Xu
Alibaba Inc.
Beijing
China

Email: xiaohu.xxh@alibaba-inc.com

SPRING
Internet-Draft
Intended status: Standards Track
Expires: December 4, 2020

S. Hegde
S. Sangli
M. Srivastava
Juniper Networks Inc.
X. Xu
Alibaba Inc.
June 2, 2020

BGP-LS Extensions for Inter-AS TE using EPE based mechanisms
draft-hegde-idr-bgp-ls-epe-inter-as-03

Abstract

In certain network deployments, a single operator has multiple Autonomous Systems(AS) to facilitate ease of management. A multiple AS network design could also be a result of network mergers and acquisitions. In such scenarios, a centralized Inter-domain TE approach could provide most optimal allocation of resources and the most controlled path placement. BGP-LS-EPE [I-D.ietf-idr-bgp-ls-segment-routing-epe] describes an extension to BGP Link State (BGP-LS) for the advertisement of BGP Peering Segments along with their BGP peering node and inter-AS link information, so that efficient BGP Egress Peer Engineering (EPE) policies and strategies can be computed based on Segment Routing. This document describes extensions to the BGP-LS EPE to enable it to be used for inter-AS Traffic-Engineering (TE) purposes.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 December 4, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Reference Topology	3
3. Fast Reroute Label	4
4. TE Link attributes of PeerNode SID	5
5. TE Link attributes of PeerAdj SID	5
6. Link address TLV	6
7. Example Advertisements	7
8. Backward Compatibility	9
9. Security Considerations	9
10. IANA Considerations	9
11. Acknowledgements	9
12. References	10
12.1. Normative References	10
12.2. Informative References	10
Authors' Addresses	10

1. Introduction

Segment Routing (SR) leverages source routing. A node steers a packet through a controlled set of instructions, called segments, by prepending the packet with an SR header with segment identifiers (SID). A SID can represent any instruction, topological or service-based. SR segments allows to enforce a flow through any topological path or service function while maintaining per-flow state only at the ingress node of the SR domain.

As there is no per-path state in the network, the bandwidth management for the paths is expected to be handled by a centralized entity which has a complete view of:

1. Up-to-date topology of the network
2. Resources, States and Attributes of links and nodes of the network
3. Run-time utilization/availability of resources

The BGP Link-State extensions provide mechanisms whereby link-state and TE information can be propagated in a network and a consumer of such BGP LS updates may build topology, provide bandwidth calendaring and other traffic engineering services. The centralized entity can be such a consumer (also referred to as controller). In the case of multi-AS networks, the controller needs to learn the per-AS network information and the inter-AS link information thus arriving at a consolidated Traffic Engineering Database which can be used to compute end-to-end Traffic Engineering Path. The controller can learn the topology, link-state and TE information from each of the AS networks either by participating in their IGPs or by listening to BGP LS updates [RFC7752]. Similar information about the inter-AS links can be learnt via BGP-LS EPE [I-D.ietf-idr-bgppls-segment-routing-epe] along with extensions defined in this document.

2. Reference Topology

The controller learns TE attributes of all the links, including the inter-AS links and uses the attributes to compute constrained paths. The controller should be able to correlate the inter-AS links for bidirectional connectivity from both ASes.

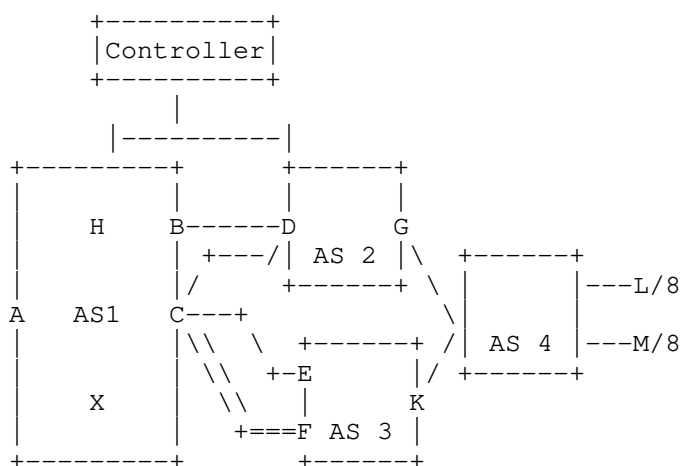


Figure 1: Reference Diagram

The reference diagram from [I-D.ietf-spring-segment-routing-central-epe] represents multiple Autonomous Systems connected to each other. When the Multiple ASes belong to same operator and are organised into separate domains for operational purposes, it is advantageous to support Traffic-Engineering across the ASes including the inter-AS links. The controller has visibility of all of the ASes by means of IGP topology exported via BGP-LS [RFC7752], or other means. In addition, the inter-AS links and the labels associated with the inter-AS links are exported via [I-D.ietf-idr-bgppls-segment-routing-epe]. The controller needs to correlate the information acquired from all of the ASes, including the inter-AS links in order to get a view of the unified topology so that it can build end-to-end Traffic-Engineered paths.

3. Fast Reroute Label

[I-D.ietf-spring-segment-routing-central-epe] section 3.6 describes mechanisms to provide Fast Reroute (FRR) protection for the EPE Labels. The BGP-LS EPE [I-D.ietf-idr-bgppls-segment-routing-epe] describes "B" bit to indicate that a PeerNodeSID or PeerAdjSID is eligible for backup. However, it does not specify what is the behaviour when the failure kicks-in. The controller needs to know which links are used for protection so that admission control and failure simulation can be done effectively and appropriate inter-AS links used for path construction.

This document defines a new flag "F" in the Peering SIDs TLV to indicate a SID as an FRR SID. With the "F" flag set, the protection for any peering SID can be specified using another PeerAdjSID, PeerNodeSID or PeerSetSID to the controller. If the protection is achieved by fallback to local IP lookup, FRR SID SHOULD not be advertised. The link(s) represented by the FRR SID will carry the traffic when there is a failure. These SIDs are included as an FRR SIDs in the peerAdjSID, PeerNodeSID and PeerSetSID advertisements.

```

      0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|V|L|B|P|F|Rsvd|
+---+---+---+---+---+---+

```

Figure 2: Peering SID TLV Flags Format

* F-Flag: FRR Label Flag: If set, the peer SID where the FRR Label appears is using backup links represented by FRR Label.

4. TE Link attributes of PeerNode SID

In any eBGP deployment, the peering session can be either single-hop or multi-hop. For single-hop eBGP sessions, the peering address is that of the directly attached interface to which the session is pinned down. For multi-hop eBGP session, the peering address is reachable over more than one interface and that the peering session is not pinned down to any of the directly attached interfaces.

A Peer Node Segment is a segment describing a peer, including the SID (PeerNodeSID) allocated to it. The link descriptors for the PeerNodeSID include the addresses used by the BGP session encoded using TLVs as defined in [RFC7752]. Since the eBGP session can be either single-hop or multi-hop, the IP address used by BGP session as local/neighbour is not sufficient to identify the underlaying interface(s). Also, the controller needs to know the links associated with the PeerNodeSID, to be able derive TE link attributes. This can be achieved by including the interface local and remote addresses in the Link attributes in PeerNodeSID NLRI. This document defines a new link attribute TLV name Interface Address TLV. PeerNodeSID NLRI MAY optionally include Interface Address TLV.

5. TE Link attributes of PeerAdj SID

PeerAdjSID MUST be advertised for each inter-AS link for the purposes of inter-AS TE. The PeerAdjSID should contain link TE attributes such as bandwidth, admin-group etc. The PeerAdjSID should also

contain the local and remote interface IPv4/IPv6 addresses which is used for correlating the links. PeerNodeSID SHOULD contain the additional attribute of link local address which is used by the controller to find corresponding PeerAdjSID and hence the corresponding link TE attributes.

A peerAdj segment carries mandatory link descriptors as local and remote link id. Remote link id of the neighboring ASBR is not readily available. [I-D.ietf-idr-bgppls-segment-routing-epe] suggests to carry the value '0' for the remote link id. The Controller needs to associate the links in both directions to effectively handle failure notifications and for this purpose a unique remote link is necessary. The remote link ID cannot be manually configured on the router as the link-ids generally change over router reboot etc and hence manual configuration is operationally very difficult to manage. This document mandates advertisement of local and remote interface addresses for the inetr-AS TE purposes.

The Unnumbered interface is not in the scope of this document.

6. Link address TLV

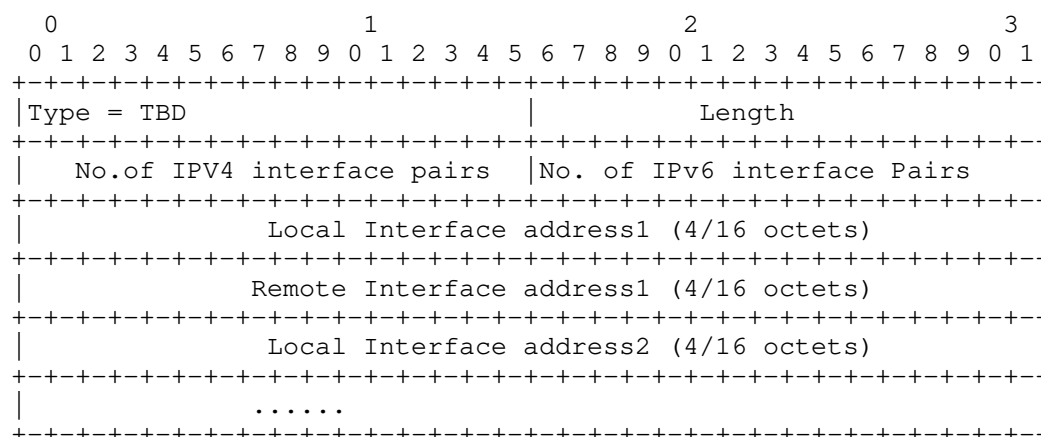


Figure 3: Link Address TLV carried as attribute

Type : TBD

Length : variable based on ipv4/ipv6 interface address

Number of IPv4 interface pairs:

Number of IPv6 interface pairs:

There may be a number of parallel interfaces and few or all of them may be used for the PeerNodeSID. These interfaces may have both IPV4 and IPV6 address or some interfaces may be IPv4 only and some IPv6 only. The total number of IPv4 and IPv6 interface address count is carried separately in above fields.

Local Interface Address :

The interface local address ipv4/ipv6 which corresponds to the PeerNodeSID MUST be specified. For IPv4, this field is 4 octets; for IPv6, this field is 16 octets.

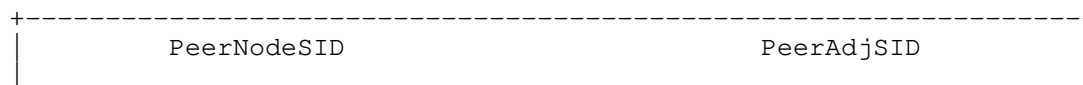
Remote Interface Address :

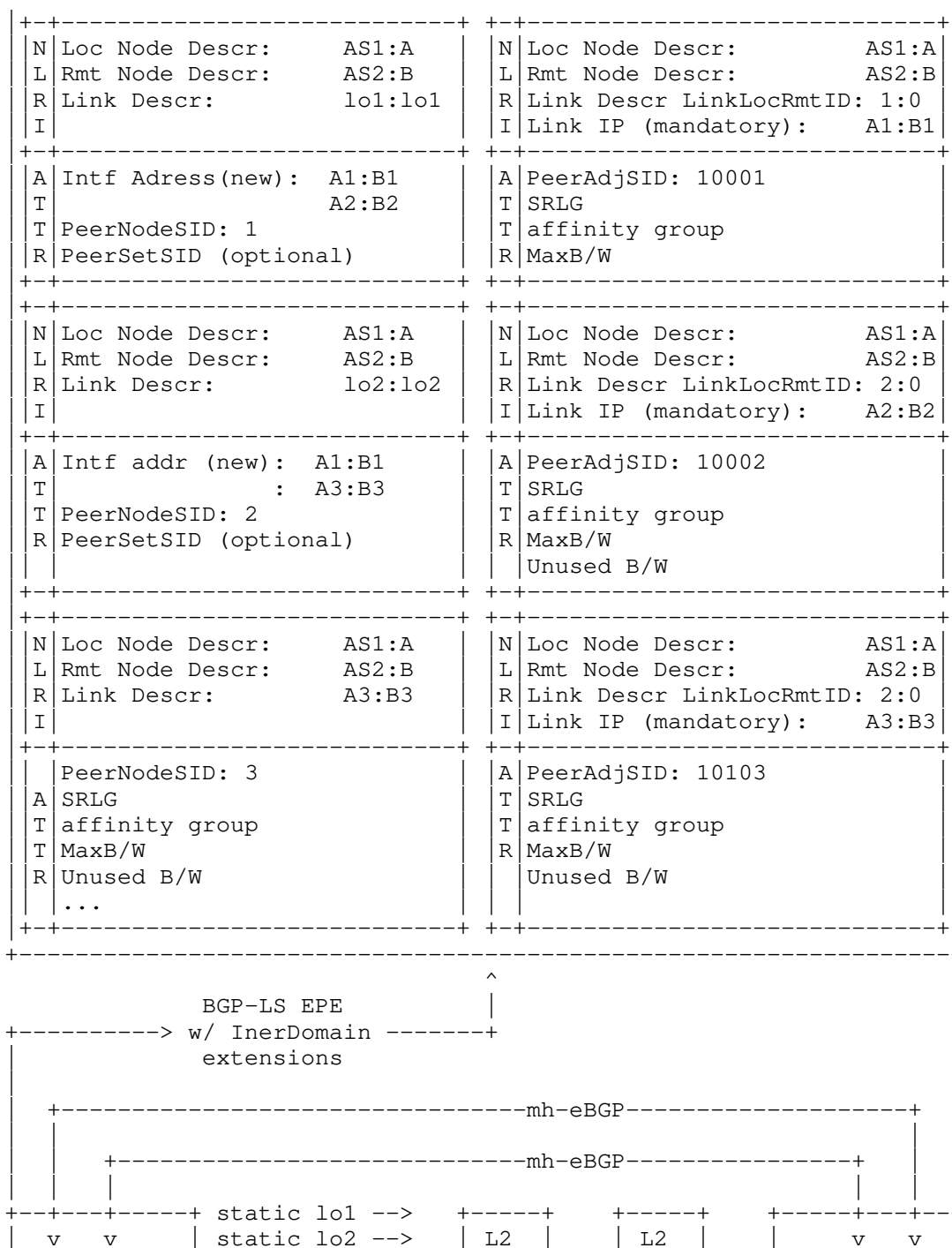
The interface remote address ipv4/ipv6 which corresponds to the PeerNodeSID MUST be specified. For IPv4, this field is 4 octets; for IPv6, this field is 16 octets.

There can be multiple Layer 3 interfaces on which a peerNodeSID loadbalances the traffic. All such interfaces local/remote address MUST be included when a Link address TLV is added. When a single Layer 3 interface consists of multiple addresses or when a link has both IPv4 and IPv6 addresses configured, It is sufficient to include one such pair (either IPV4 or IPV6) address for the PeerNodeSID advertisement. When a PeerNodeSID load-balances over few interfaces with IPv4 only address and few interfaces with IPv6 address then the Link address TLV should list all IPv4 address pairs together followed by IPv6 address pairs.

7. Example Advertisements

The below diagram represents two ASBR routers and inter-AS links between them. The inter-AS links could be connected via switches L1 and L2 as shown in the diagram or via Point-to-point links A2->B2, A3->B3 as shown in the diagram below. In the below example, lets assume peerNodeSID 1 is configured to use peerAdjSID 10002 then PeerNodeSID 1 will have the B bit set which means the PeerNodeSID 1 is eligible for backup. Label 10002 is added to the PeerNodeSID with a "F" bit set, which means 10002 is a backup for PeerNodeSID 1.





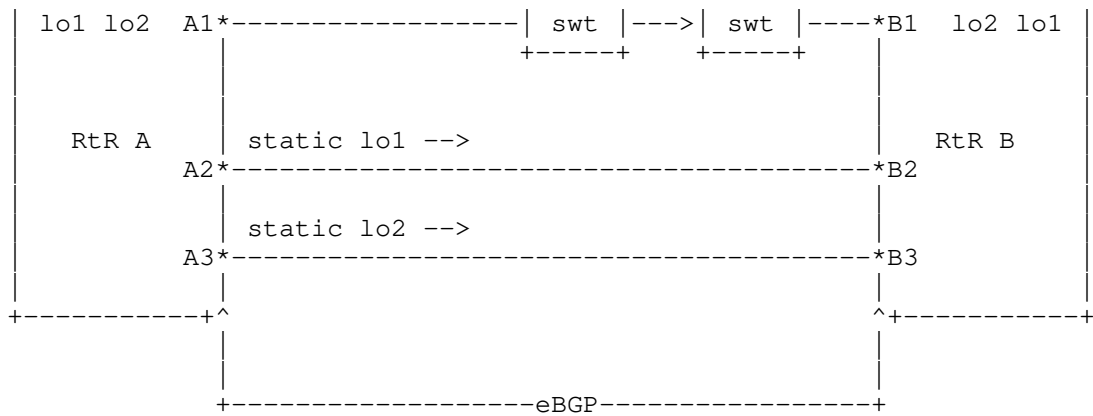


Figure 4: Example Advertisements

8. Backward Compatibility

The extension proposed in this document is backward compatible with procedures described in [I-D.ietf-idr-bgppls-segment-routing-epe] and [I-D.ietf-spring-segment-routing-central-epe]

9. Security Considerations

TBD

10. IANA Considerations

New attribute TLV in BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs registry

TLV Code Point	Description	IS-IS TLV /Sub-TLV	Reference (RFC/Section)
TBD	Link address TLV	NA	This draft

Figure 5: IANA code point

11. Acknowledgements

Thanks to Julian Lucek and Rafal Jan Szarecki for careful review and suggestions.

12. References

12.1. Normative References

- [I-D.ietf-idr-bgppls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgppls-segment-routing-epe-19 (work in progress), May 2019.
- [I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.
- [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>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<https://www.rfc-editor.org/info/rfc7752>>.

12.2. Informative References

- [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

Shraddha Hegde
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Srihari Sangli
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: ssangli@juniper.net

Mukul Srivastava
Juniper Networks Inc.

Email: msri@juniper.net

Xiaohu Xu
Alibaba Inc.
Beijing
China

Email: xiaohu.xxh@alibaba-inc.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 27, 2019

Z. Hu
Huawei Technologies
D. Ye
Cisco Systems
Y. Qu
J. Dong
Huawei Technologies
March 26, 2019

YANG Data Model for IS-IS SRv6
draft-hu-isis-srv6-yang-01

Abstract

This document defines a YANG data model that can be used to configure and manage IS-IS SRv6 [I-D.bashandy-isis-srv6-extensions].

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 September 27, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Overview	2
2. IS-IS SRv6	2
3. IS-IS SRv6 configuration	5
3.1. SRv6 activation	5
3.2. Locator setting	5
3.3. IP Fast reroute	5
4. IS-IS SRv6 YANG Module	5
5. Security Considerations	18
6. Contributors	18
7. Acknowledgements	18
8. IANA Considerations	18
9. References	18
Authors' Addresses	19

1. Overview

YANG[RFC6020][RFC7950] is a data definition language used to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF[RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g., ReST) and encodings other than XML (e.g., JSON) are being defined. Furthermore, YANG data models can be used as the basis for implementation of other interfaces, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage IS-IS SRv6 and it is an augmentation to the IS- IS YANG data model.

2. IS-IS SRv6

This document defines a model for IS-IS SRv6 feature. It is an augmentation of the IS-IS base model.

The IS-IS SRv6 YANG module requires support for the base srv6 module[I-D.raza-spring-srv6-yang], which defines the global srv6 configuration independent of any specific routing protocol configuration, and support of IS-IS base model

[I-D.ietf-isis-yang-isis-cfg] which defines basic IS-IS configuration and state.

The figure below describes the overall structure of the isis-srv6 YANG module:

```

module: ietf-isis-srv6
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/isis:isis:
      +--rw srv6-cfg
        +--rw enable?                boolean
        +--rw default-locator?      boolean
        +--rw locator-name?         -> /rt:routing/srv6:srv6
                                      /locators/locator/name
        +--rw persistent-end-x-sid? boolean
      augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/isis:isis
          /isis:interfaces/isis:interface/isis:fast-reroute:
            +--rw srv6-ti-lfa {srv6-ti-lfa}?
              +--rw enable?    boolean
      augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/isis:isis/isis:database
          /isis:level-db/isis:lsp/isis:router-capabilities:
            +--ro v6-capability
              | +--ro flags?    bits
            +--ro v6-msd
              +--ro max-segments-left?  uint8
              +--ro max-end-pop?         uint8
              +--ro max-t-insert?        uint8
              +--ro max-t-encap?         uint8
              +--ro max-end-d?           uint8
      augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/isis:isis/isis:database
          /isis:level-db/isis:lsp/isis:extended-is-neighbor
            /isis:neighbor:
              +--ro srv6-end-x-sids
                +--ro end-x-sid* [sid]
                  +--ro func-flags?    bits
                  +--ro algorithm?     uint8
                  +--ro weight?        uint8
                  +--ro endpoint-func
                    | +--ro flags?                uint8
                    | +--ro endpoint-func?        identityref
                    | +--ro undefined-endpoint-func? uint16
                  +--ro sid              srv6-sid-value
                  +--ro neighbor-id?     isis:system-id
      augment /rt:routing/rt:control-plane-protocols
        /rt:control-plane-protocol/isis:isis/isis:database

```

```

    /isis:level-db/isis:lsp/isis:mt-is-neighbor/isis:neighbor:
+--ro srv6-end-x-sids
  +--ro end-x-sid* [sid]
    +--ro func-flags?      bits
    +--ro algorithm?      uint8
    +--ro weight?         uint8
    +--ro endpoint-func
      | +--ro flags?              uint8
      | +--ro endpoint-func?      identityref
      | +--ro undefined-endpoint-func? uint16
    +--ro sid              srv6-sid-value
    +--ro neighbor-id?     isis:system-id
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/isis:isis/isis:database
  /isis:level-db/isis:lsp:
+--ro srv6-locators
  +--ro locator* [locator]
    +--ro mt-id?          uint16
    +--ro flags?          bits
    +--ro metric?         uint32
    +--ro algorithm?      uint8
    +--ro loc-size?       uint8
    +--ro locator         inet:ipv6-address-no-zone
    +--ro srv6-end-sids
      | +--ro end-sid* [sid]
      |   +--ro flags?      uint8
      |   +--ro endpoint-func
      |     | +--ro flags?          uint8
      |     | +--ro endpoint-func?  identityref
      |     | +--ro undefined-endpoint-func? uint16
      |   +--ro sid        srv6-sid-value
    +--ro external-prefix-flag? boolean
    +--ro readvertisement-flag? boolean
    +--ro node-flag?          boolean
    +--ro ipv4-source-router-id? inet:ipv4-address
    +--ro ipv6-source-router-id? inet:ipv6-address
augment /rt:routing/rt:control-plane-protocols
  /rt:control-plane-protocol/isis:isis/isis:interfaces
  /isis:interface/isis:adjacencies/isis:adjacency:
+--ro end-x-sid* [value]
  +--ro value              srv6-sid-value
  +--ro weight?            uint8
  +--ro protection-requested? boolean
  +--ro persistent?        boolean
  +--ro algorithm?         uint8
  +--ro endpoint-func
    +--ro flags?            uint8
    +--ro endpoint-func?    identityref

```


+++ro undefined-endpoint-func? uint16

3. IS-IS SRv6 configuration

3.1. SRv6 activation

Activation of IS-IS SRv6 is done by setting the "enable" leaf to true. This triggers advertisement of SRv6 extensions based on the configuration parameters that have been setup using the base SRv6 module.

3.2. Locator setting

The basic SRv6 module defines the related locator leafs. When the IS-IS SRv6 module is enabled, set the locator by using the following strategy: firstly, it is reasonable to check whether the default locator is used, if not, to use the specified locator. The strategy is realized by adding the leaf "default-locator", "locator-name" .

3.3. IP Fast reroute

IS-IS SRv6 model augments the fast-reroute container under interface. It brings the ability to activate ipv6 TI-LFA (topology independent LFA).

4. IS-IS SRv6 YANG Module

```
<CODE BEGINS> file "ietf-isis-srv6@2019-03-22.yang"
module ietf-isis-srv6 {
  namespace "urn:ietf:params:xml:ns:"
    + "yang:ietf-isis-srv6";
  prefix isis-srv6;

  import ietf-routing {
    prefix "rt";
  }

  import ietf-isis {
    prefix "isis";
  }

  import ietf-srv6-base {
    prefix "srv6";
  }

  import iana-routing-types {
    prefix "iana-rt-types";
  }
}
```

```
}

import ietf-inet-types {
  prefix "inet";
}

organization
  "IETF ISIS Working Group";

contact
  "WG List: <mailto:spring@ietf.org>
  Author:      Zhibo Hu
               <mailto:huzhibo@huawei.com>
  Author:      Dan Ye
               <mailto:daye@cisco.com>
  Author:      Yingzhen Qu
               <mailto:yingzhen.qu@huawei.com>
  Author:      Jiajia Dong
               <mailto:dongjiajia@huawei.com>
  ";

description
  "The YANG module defines a generic configuration model for
  Segment IPV6 routing ISIS extensions common across all of
  the vendor implementations.";

revision 2019-03-26 {
  description
    "Initial revision.";
  reference "draft-bashandy-isis-srv6-extensions-04";
}

/* Identities */
identity SRV6_END_FUNC_TYPE {
  description
    "Base identity type for srv6 endpoint function code points.";
}

identity SRV6_END_FUNC_PSP_USP_USD {
  base "SRV6_END_FUNC_TYPE";
  description
    "End (May support PSP, USP, USD).";
}

identity SRV6_END_X_FUNC_PSP_USP_USD {
  base "SRV6_END_FUNC_TYPE";
  description
    "End.X(May support PSP, USP, USD)";
}
```

```
}

identity SRV6_END_T_FUNC_PSP_USP_USD {
  base "SRV6_END_FUNC_TYPE";
  description
    "END (May support PSP, USP, USD)";
}

identity SRV6_END_FUNC_DX6 {
  base "SRV6_END_FUNC_TYPE";
  description
    "End.DX6.";
}

identity SRV6_END_FUNC_DX4 {
  base "SRV6_END_FUNC_TYPE";
  description
    "End.DX4.";
}

identity SRV6_END_FUNC_DT6 {
  base "SRV6_END_FUNC_TYPE";
  description
    "End.DT6.";
}

identity SRV6_END_FUNC_DT4 {
  base "SRV6_END_FUNC_TYPE";
  description
    "End.DT4.";
}

identity SRV6_END_FUNC_DT64 {
  base "SRV6_END_FUNC_TYPE";
  description
    "End.DT64.";
}

identity SRV6_END_FUNC_OP {
  base "SRV6_END_FUNC_TYPE";
  description
    "END.OP .";
}

identity SRV6_END_FUNC_OTP {
  base "SRV6_END_FUNC_TYPE";
  description
    "END.OTP .";
}
```

```
}

/* typedef */
typedef srv6-sid-value {
  type inet:ipv6-address-no-zone;
  description
    "16 Octets encoded sid value.";
}

/* Features */
feature srv6-ti-lfa {
  description
    "Enhance SRv6 FRR with ti-lfa
    support";
}

/* Groupings */
grouping srv6-msd {
  description
    "means to advertise to advertise node/link specific
    values for Maxium Sid Depths(MSD) of various types";
  container v6-msd {
    description
      "Maximum SRv6 SID Depths.";
    leaf max-segments-left {
      type uint8;
      description
        "The maximum value of 'SL' field in the SRH of a
        recevied packet.";
    }
    leaf max-end-pop {
      type uint8;
      description
        "The maximum number of SIDS in the top SRH in an SRH
        stack to which the router can apply 'PSP' or 'USP'.";
    }
    leaf max-t-insert {
      type uint8;
      description
        "The maximum number of SIDs can be inserted as port of
        the 'T.insert' behavior.";
    }
    leaf max-t-encap {
      type uint8;
      description
        "The maximum number of SIDs can be included as part of
        the 'T.Encap' behavior.";
    }
    leaf max-end-d {
```

```
    type uint8;
    description
      "The maximum number of SIDs in an SRH when performing
      decapsulation associated with 'End.Dx' functions
      (e.g., 'End.DX6' and 'End.DT6').";
  }
}

grouping srv6-capability {
  description
    "SRV6 capability grouping.";
  container v6-capability {
    description
      "SRv6 capability.";
    leaf flags {
      type bits {
        bit o-flag {
          position 1;
          description
            "If set, then the router is able to
            use of the O-bit in the Segment Routing Header (SRH)
            as defined in [draft-ietf-6man-segment-routing-header].";
        }
      }
      description
        "Flags.";
    }
  }
}

grouping srv6-endpoint-func {
  description
    "This group defines srv6 endpoint function";
  container endpoint-func {
    description
      "Srv6 Endpoint function Descriptor.";
    leaf flags {
      type uint8;
      description
        "No flags are currently being defined.";
    }
    leaf endpoint-func {
      type identityref {
        base isis-srv6:SRV6_END_FUNC_TYPE;
      }
      description
        "The endpoint function.";
    }
  }
}
```

```
    }
    leaf undefined-endpoint-func {
        type uint16;
        description
            "Unknown endpoint func value.";
    }
}

grouping srv6-end-sid {
    description
        "SRv6 Segment Identifier(SID) with Endpoint functions.";
    leaf flags {
        type uint8;
        description
            "NO flags are currently being defined.";
    }

    uses srv6-endpoint-func;

    leaf sid {
        type srv6-sid-value;
        description
            "SRV6 sid value.";
    }
    // sub-sub-tlvs not yet defined
}

grouping srv6-locator {
    description
        "This group defines srv6 locator tlv.";
    leaf mt-id {
        type uint16 {
            range "0..4095";
        }
        description
            "Multitopology Identifier as defined in [RFC5120].";
    }
    leaf flags {
        type bits {
            bit d-flag {
                position 0;
                description
                    "When the locator is leaked from level-2 to leve-1,
                     the d-flag must be set.";
            }
            bit a-flag {
                position 1;
            }
        }
    }
}
```

```
        description
            "When the Locator is associated with anycast destinations, the A bit
            SHOULD be set. Otherwise, this bit MUST be clear.";
    }
}
description
    "Flags for srv6 locator tlv.";
}

leaf metric {
    type uint32;
    description
        "Metric value.";
}
leaf algorithm {
    type uint8;
    description
        "Associated algorithm.";
}

leaf loc-size {
    type uint8;
    description
        "Number of bits in the locator field.";
}
leaf locator {
    type inet:ipv6-address-no-zone;
    description
        "Advertised SRV6 locator.";
}
container srv6-end-sids {
    description
        "This contains list of srv6 end sids.";
    list end-sid {
        key "sid";
        description
            "List of SRV6 SRv6 Segment Identifiers (SID)
            with Endpoint functions.";
        uses srv6-end-sid;
    }
}
uses isis:prefix-reachability-attributes;
uses isis:prefix-ipv4-source-router-id;
uses isis:prefix-ipv6-source-router-id;
}

grouping srv6-end-x-sid {
    description
```

```
"SRv6 sid associated with an adjacency.";

leaf func-flags {
  type bits {
    bit b-flag {
      position 0;
      description
        "Backup flag. If set, the End.X sid is
         eligible for protection.";
    }

    bit s-flag {
      position 1;
      description
        "Set flag. When set, the End.X sid refers to
         a set of adjacencies (and therefore May be assigned
         to other adjacencies as well.";
    }

    bit p-flag {
      position 2;
      description
        "Persistent flag. When set, the End.X sid is persistently
         allocated, i.e., the End.x sid value remains consistent
         across router restart and/or interface flap.";
    }
  }
  description
    "Flags for srv6 end x sid.";
}

leaf algorithm {
  type uint8;
  description
    "Associated algorithm.";
}

leaf weight {
  type uint8;
  description
    "The value represents the weight of the End.X sid
     for the purpose of load balancing.";
}

uses srv6-endpoint-func;

leaf sid {
  type srv6-sid-value;
```



```
        description
            "SRV6 sid value.";
    }

    leaf neighbor-id {
        type isis:system-id;
        description
            "Describes the system ID of the neighbor
             associated with the SID value. This is only
             used on LAN adjacencies.";
    }

    // sub-sub-tlvs
}

grouping srv6-adjacency-state {
    description
        "This group will extend adjacency state.";
    list end-x-sid {
        key value;
        config false;
        leaf value {
            type srv6-sid-value;
            description
                "Value of the Adj-SID.";
        }
        leaf weight {
            type uint8;
            description
                "Weight associated with
                 the End.X SID.";
        }
        leaf protection-requested {
            type boolean;
            description
                "Set to True if the End.X SID
                 must be protected.";
        }
        leaf persistent {
            type boolean;
            description
                "Persistent flag. When set, the End.X sid is persistently
                 allocated, i.e., the End.X sid value remains consistent
                 across router restart and/or interface flap.";
        }
        leaf algorithm {
            type uint8;
            description
```

```
        "Associated algorithm.";
    }
    uses srv6-endpoint-func;

    description
        "List of End.X Segment IDs.";
    }
}
/* Cfg */
augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis" {
    when "/rt:routing/rt:control-plane-protocols/" +
        "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
            "This augment ISIS routing protocol when used";
    }
    description
        "This augments ISIS protocol configuration
        with SRv6.";

    container srv6-cfg{
        leaf enable{
            type boolean;
            default "false";
            description
                "Enables SRv6
                protocol extensions.";
        }

        leaf default-locator {
            type boolean;
            default "false";
            description
                "Enable ISIS segment-routing IPv6 with default Locator.";
        }

        leaf locator-name {
            when "not(..../default-locator='true')";
            type leafref {
                path "/rt:routing/srv6:srv6/srv6:locators/srv6:locator/srv6:name";
            }
            description
                "Enable ISIS segment-routing IPv6 with specified Locator.";
        }

        leaf persistent-end-x-sid{
```

```
        type boolean;
        default "false";
        description
            "Enable the persistent nature of End.X sid";
    }
    description
        "Configuration about ISIS segment-routing IPv6.";
}

augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:interfaces/isis:interface"+
    "/isis:fast-reroute"{
when "/rt:routing/rt:control-plane-protocols/"+
    "rt:control-plane-protocol/rt:type = 'isis:isis'"{
    description
        "This augment ISIS routing protocol when used";
}
description
    "This augments ISIS IPFRR with IPV6 TILFA.";

container srv6-ti-lfa {

    if-feature srv6-ti-lfa;
    leaf enable {
        type boolean;
        description
            "Enables SRv6 TI-LFA computation.";
    }

    description
        "SRv6 TILFA configuration.";
}

/* Operational states */
augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:database/isis:level-db/isis:lsp"+
    "/isis:router-capabilities" {
when "/rt:routing/rt:control-plane-protocols/"+
    "rt:control-plane-protocol/rt:type = 'isis:isis' " {
    description
        "This augment ISIS routing protocol when used";
}
description
```

```
"This augments ISIS protocol router capability.";
uses srv6-capability;
uses srv6-msd;
}

augment "/rt:routing/" +
  "rt:control-plane-protocols/rt:control-plane-protocol"+
  "/isis:isis/isis:database/isis:level-db/isis:lsp"+
  "/isis:extended-is-neighbor/isis:neighbor" {
  when "/rt:routing/rt:control-plane-protocols/"+
    "rt:control-plane-protocol/rt:type = 'isis:isis'" {
    description
      "This augment ISIS routing protocol when used.";
  }
  description
    "This augments ISIS protocol neighbor.";
  container srv6-end-x-sids {
    description
      "This defines svr6 end-x sids for the adjacency.";
    list end-x-sid {
      key "sid";
      uses srv6-end-x-sid;
      description
        "List of end-x sids.";
    }
  }
}

augment "/rt:routing/" +
  "rt:control-plane-protocols/rt:control-plane-protocol"+
  "/isis:isis/isis:database/isis:level-db/isis:lsp"+
  "/isis:mt-is-neighbor/isis:neighbor" {
  when "/rt:routing/rt:control-plane-protocols/"+
    "rt:control-plane-protocol/rt:type = 'isis:isis'" {
    description
      "This augment ISIS routing protocol when used.";
  }
  description
    "This augments ISIS protocol neighbor.";
  container srv6-end-x-sids {
    description
      "This defines svr6 end-x sids for the adjacency.";
    list end-x-sid {
      key "sid";
      uses srv6-end-x-sid;
      description
        "List of end-x sids.";
    }
  }
}
```

```
}

augment "/rt:routing/" +
  "rt:control-plane-protocols/rt:control-plane-protocol"+
  "/isis:isis/isis:database/isis:level-db/isis:lsp" {
  when "/rt:routing/rt:control-plane-protocols/"+
  "rt:control-plane-protocol/rt:type = 'isis:isis'" {
    description
      "This augment ISIS routing protocol when used.";
  }
  description
    "This augments ISIS protocol LSDB.";
  container srv6-locators {
    description
      "This defines srv6 locator tlvs.";
    list locator {
      key "locator";
      uses srv6-locator;
      description
        "List of srv6 locators.";
    }
  }
}

augment "/rt:routing/" +
  "rt:control-plane-protocols/rt:control-plane-protocol"+
  "/isis:isis/isis:interfaces/isis:interface" +
  "/isis:adjacencies/isis:adjacency" {
  when "/rt:routing/rt:control-plane-protocols/"+
  "rt:control-plane-protocol/rt:type = 'isis:isis'" {
    description
      "This augment ISIS routing protocol when used.";
  }
  description
    "This augments ISIS protocol operational state
    with segment routing.";

  uses srv6-adjacency-state;
}

/* Notifications */
}
<CODE ENDS>
```

5. Security Considerations

Configuration and state data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241].

As IS-IS is an IGP protocol (critical piece of the network), ensuring stability and security of the protocol is mandatory for the network service.

Authors recommends to implement NETCONF access control model ([RFC6536]) to restrict access to all or part of the configuration to specific users.

6. Contributors

TBD.

7. Acknowledgements

TBD.

8. IANA Considerations

The IANA is requested to assign two new URIs from the IETF XML registry ([RFC3688]). Authors are suggesting the following URI:

URI: urn:ietf:params:xml:ns:yang:ietf-isis-srv6
Registrant Contact: IS-IS WG
XML: N/A, the requested URI is an XML namespace

This document also requests one new YANG module name in the YANG Module Names registry ([RFC6020]) with the following suggestion :

name: ietf-isis-srv6
namespace: urn:ietf:params:xml:ns:yang:ietf-isis-srv6 prefix: isis-srv6
reference: RFC XXXX

9. References

[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.ietf-isis-yang-isis-cfg]
Litkowski, S., Yeung, D., Lindem, A., Zhang, Z., and L. Lhotka, "YANG Data Model for IS-IS Protocol", draft-ietf-isis-yang-isis-cfg-35 (work in progress), March 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-02 (work in progress), October 2018.
- [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>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.

Authors' Addresses

Zhibo Hu
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: huzhibo@huawei.com

Dan Ye
Cisco Systems
170 W. Tasman Dr. San Jose,
California 95134
USA

Email: daye@cisco.com

Yingzhen Qu
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: yingzhen.qu@huawei.com

Jiajia Dong
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: dongjiajia@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 26, 2021

Z. Hu
Huawei
D. Ye
Cisco
Y. Qu
Futurewei
X. Geng
Q. Ma
Huawei
June 24, 2021

YANG Data Model for IS-IS SRv6
draft-hu-isis-srv6-yang-06

Abstract

This document defines a YANG data model that can be used to configure and manage IS-IS SRv6 [I-D.ietf-lsr-isis-srv6-extensions].

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 December 26, 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Overview	2
2. IS-IS SRv6	2
3. IS-IS SRv6 configuration	5
3.1. SRv6 activation	5
3.2. Locator setting	5
3.3. IP Fast reroute	5
3.4. Microloop avoidance	6
4. IS-IS SRv6 YANG Module	6
5. Security Considerations	19
6. Contributors	20
7. Acknowledgements	20
8. IANA Considerations	20
9. References	20
Appendix A. Configuration examples	22
Authors' Addresses	23

1. Overview

YANG [RFC6020] [RFC7950] is a data definition language used to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF [RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces (e.g., REST) and encodings other than XML (e.g., JSON) are being defined. Furthermore, YANG data models can be used as the basis for implementation of other interfaces, such as CLI and programmatic APIs.

This document defines a YANG data model that can be used to configure and manage IS-IS SRv6 and it is an augmentation to the IS-IS YANG data model.

2. IS-IS SRv6

This document defines a model for IS-IS SRv6 feature. It is an augmentation of the IS-IS base model.

The IS-IS SRv6 YANG module requires support for the base srv6 module [I-D.ietf-spring-srv6-yang], which defines the global srv6 configuration independent of any specific routing protocol configuration, and support of IS-IS base model [I-D.ietf-isis-yang-isis-cfg] which defines basic IS-IS configuration and state. This module uses types defined in [RFC6991]. It also references [RFC8349], [I-D.ietf-spring-srv6-yang], [I-D.ietf-isis-yang-isis-cfg], [I-D.ietf-spring-sr-yang] and [I-D.ietf-spring-srv6-yang].

The figure below describes the overall structure of the isis-srv6 YANG module:

```

module: ietf-isis-srv6
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/isis:isis:
      +--rw srv6-cfg
      |   +--rw enable?                boolean
      |   +--rw default-locator?      boolean
      |   +--rw locator-name*         -> /rt:routing/sr:segment-routing
      |                                   /srv6:srv6/locators/locator/name
      |   +--rw persistent-end-x-sid?  boolean
      +--rw micro-loop-avoidance
      |   +--rw srv6-enable?           boolean
      |   +--rw srv6-rib-update-delay? uint16
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/isis:isis/isis:fast-reroute:
      +--rw srv6-ti-lfa {srv6-ti-lfa}?
      |   +--rw enable?               boolean
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/isis:isis/isis:database
      /isis:levels/isis:lsp/isis:router-capabilities:
      +--ro v6-capability
      |   +--ro flags?                bits
      +--ro srv6-msd
      |   +--ro max-segments-left?    uint8
      |   +--ro max-end-pop?          uint8
      |   +--ro max-h-encaps?         uint8
      |   +--ro max-end-d?            uint8
  augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/isis:isis/isis:database
      /isis:levels/isis:lsp/isis:extended-is-neighbor/isis:neighbor:
      +--ro srv6-adjacency-sids
      |   +--ro end-x-sid* [sid]
      |   |   +--ro func-flags?       bits
      |   |   +--ro algorithm?        uint8
      |   |   +--ro weight?           uint8
      |   +--ro endpoint-func

```

```

    |   +---ro flags?                               uint8
    |   +---ro endpoint-func?                       identityref
    |   +---ro undefined-endpoint-func?             uint16
+---ro sid                               srv6-sid-value
+---ro neighbor-id?                      isis:system-id
+---ro srv6-sid-structure
    |   +---ro lb-length?      uint8
    |   +---ro ln-length?      uint8
    |   +---ro fun-length?     uint8
    |   +---ro arg-length?     uint8
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/isis:isis/isis:database
    /isis:levels/isis:lsp/isis:mt-is-neighbor/isis:neighbor:
+---ro srv6-adjacency-sids
    +---ro end-x-sid* [sid]
        +---ro func-flags?          bits
        +---ro algorithm?           uint8
        +---ro weight?              uint8
        +---ro endpoint-func
            |   +---ro flags?                uint8
            |   +---ro endpoint-func?        identityref
            |   +---ro undefined-endpoint-func?  uint16
        +---ro sid                    srv6-sid-value
        +---ro neighbor-id?           isis:system-id
        +---ro srv6-sid-structure
            |   +---ro lb-length?      uint8
            |   +---ro ln-length?      uint8
            |   +---ro fun-length?     uint8
            |   +---ro arg-length?     uint8
augment /rt:routing/rt:control-plane-protocols
    /rt:control-plane-protocol/isis:isis/isis:database
    /isis:levels/isis:lsp:
+---ro srv6-locators
    +---ro locator* [locator]
        +---ro mt-id?                uint16
        +---ro flags?                 bits
        +---ro metric?                uint32
        +---ro algorithm?             uint8
        +---ro loc-size?              uint8
        +---ro locator                inet:ipv6-address-no-zone
    +---ro srv6-end-sids
        |   +---ro end-sid* [sid]
        |   |   +---ro flags?                uint8
        |   |   +---ro endpoint-func
        |   |   |   +---ro flags?                uint8
        |   |   |   +---ro endpoint-func?        identityref
        |   |   |   +---ro undefined-endpoint-func?  uint16
        |   +---ro sid                    srv6-sid-value

```

```

    +--ro srv6-sid-structure
      +--ro lb-length?      uint8
      +--ro ln-length?      uint8
      +--ro fun-length?     uint8
      +--ro arg-length?     uint8
    +--ro external-prefix-flag?  boolean
    +--ro readvertisement-flag?  boolean
    +--ro node-flag?            boolean
    +--ro ipv4-source-router-id? inet:ipv4-address
    +--ro ipv6-source-router-id? inet:ipv6-address
augment /rt:routing/rt:control-plane-protocols
/rt:control-plane-protocol/isis:isis/isis:interfaces
/isis:interface/isis:adjacencies/isis:adjacency:
+--ro end-x-sid* [value]
  +--ro value                srv6-sid-value
  +--ro weight?              uint8
  +--ro protection-requested? boolean
  +--ro persistent?          boolean
  +--ro algorithm?           uint8
  +--ro endpoint-func
    +--ro flags?              uint8
    +--ro endpoint-func?      identityref
    +--ro undefined-endpoint-func? uint16

```

3. IS-IS SRv6 configuration

3.1. SRv6 activation

Activation of IS-IS SRv6 is done by setting the "enable" leaf to true. This triggers advertisement of SRv6 extensions based on the configuration parameters that have been setup using the base SRv6 module.

3.2. Locator setting

The basic SRv6 module defines the related locator leafs. When the IS-IS SRv6 module is enabled, set the locator by using the following strategy: firstly, it is reasonable to check whether the default locator is used, if not, to use the specified locator. The strategy is realized by adding the leaf "default-locator", "locator-name" .

3.3. IP Fast reroute

IS-IS SRv6 model augments the fast-reroute container. It brings the ability to activate ipv6 TI-LFA (topology independent LFA).

3.4. Microloop avoidance

IS-IS SRv6 model augments the micro-loop-avoidance container, this container including the leaf "srv6-enable" brings the ability to activate SRv6 avoid-microloop.

4. IS-IS SRv6 YANG Module

```
<CODE BEGINS> file "ietf-isis-srv6@2020-07-13.yang"
module ietf-isis-srv6 {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:"
    + "yang:ietf-isis-srv6";
  prefix isis-srv6;

  import ietf-routing {
    prefix "rt";
    reference "RFC8349: A YANG Data Model for
              Routing Management (NMDA Version)";
  }

  import ietf-isis {
    prefix "isis";
    reference "draft-ietf-isis-yang-isis-cfg: YANG
              Data Model for IS-IS Protocol";
  }

  import ietf-inet-types {
    prefix "inet";
    reference "RFC 6991:Common YANG Data Types";
  }

  import ietf-segment-routing {
    prefix sr;
    reference "draft-ietf-spring-sr-yang: YANG Data
              Model for Segment Routing";
  }

  import ietf-srv6-base {
    prefix "srv6";
    reference "draft-ietf-spring-srv6-yang: YANG Data
              Model for SRv6 Base and Static";
  }

  organization
    "IETF LSR Working Group";

  contact
```

```
"WG List: <mailto:spring@ietf.org>
Author:   Zhibo Hu
          <mailto:huzhibo@huawei.com>
Author:   Dan Ye
          <mailto:daye@cisco.com>
Author:   Yingzhen Qu
          <mailto:yingzhen.qu@futurewei.com>
Author:   Qiufang Ma
          <mailto:maqiufang1@huawei.com>
";
description
  "The YANG module defines a generic configuration model for
  Segment IPV6 routing ISIS extensions common across all of
  the vendor implementations.";

revision 2020-07-13 {
  description
    "Initial revision.";
  reference "draft-ietf-lsr-isis-srv6-extensions-08";
}

/* Identities */
identity SRV6_END_FUNC_TYPE {
  description
    "Base identity type for srv6 endpoint function code points.";
}

identity SRV6_END_FUNC_PSP_USP_USD {
  base "SRV6_END_FUNC_TYPE";
  description
    "End (May support PSP, USP, USD).";
}

identity SRV6_END_X_FUNC_PSP_USP_USD {
  base "SRV6_END_FUNC_TYPE";
  description
    "End.X(May support PSP, USP, USD)";
}

identity SRV6_END_T_FUNC_PSP_USP_USD {
  base "SRV6_END_FUNC_TYPE";
  description
    "END (May support PSP, USP, USD)";
}

identity SRV6_END_FUNC_DX6 {
  base "SRV6_END_FUNC_TYPE";
  description
```

```
    "End.DX6.";
}

identity SRV6_END_FUNC_DX4 {
    base "SRV6_END_FUNC_TYPE";
    description
        "End.DX4.";
}

identity SRV6_END_FUNC_DT6 {
    base "SRV6_END_FUNC_TYPE";
    description
        "End.DT6.";
}

identity SRV6_END_FUNC_DT4 {
    base "SRV6_END_FUNC_TYPE";
    description
        "End.DT4.";
}

identity SRV6_END_FUNC_DT64 {
    base "SRV6_END_FUNC_TYPE";
    description
        "End.DT64.";
}

identity SRV6_END_FUNC_OP {
    base "SRV6_END_FUNC_TYPE";
    description
        "END.OP .";
}

identity SRV6_END_FUNC_OTP {
    base "SRV6_END_FUNC_TYPE";
    description
        "END.OTP .";
}

/* typedef */
typedef srv6-sid-value {
    type inet:ipv6-address-no-zone;
    description
        "16 Octets encoded sid value.";
}

/* Features */
feature srv6-ti-lfa {
```



```
description
  "Enhance SRv6 FRR with ti-lfa
  support";
}
/* Groupings */
grouping srv6-msds {
  description
    "means to advertise to advertise node/link specific
    values for Maximum Sid Depths(MSD) of various types";
  container srv6-msd {
    description
      "Maximum SRv6 SID Depths.";
    leaf max-segments-left {
      type uint8;
      description
        "The Maximum Segments Left MSD Type specifies
        the maximum value of the 'SL' field in the SRH
        of a received packet before applying the
        Endpoint behavior associated with a SID.";
    }
    leaf max-end-pop {
      type uint8;
      description
        "The Maximum End Pop MSD Type specifies the maximum
        number of SIDs in the SRH to which the router can
        apply 'PSP' or 'USP' behavior, as defined in flavors.";
    }
    leaf max-h-encaps {
      type uint8;
      description
        "The Maximum H.Encaps MSD Type specifies the maximum number
        of SIDs that can be included as part of the 'H.Encaps'
        behavior";
    }
    leaf max-end-d {
      type uint8;
      description
        "The maximum number of SIDs in an SRH when performing
        decapsulation associated with 'End.Dx' functions
        (e.g., 'End.DX6' and 'End.DT6').";
    }
  }
}

grouping srv6-sid-structures {
  description
    "This group defines SRv6 SID Structure sub-sub-TLV.";
  container srv6-sid-structure {
```

```
description
  "SRv6 SID Structure sub-sub-TLV is used to advertise
   the length of each individual part of the SRv6 SID
   as defined in [I-D.ietf-spring-srv6-network-programming]";
leaf lb-length {
  type uint8;
  description
    "SRv6 SID Locator Block length in bits.";
}

leaf ln-length {
  type uint8;
  description
    "SRv6 SID Locator Node length in bits.";
}

leaf fun-length {
  type uint8;
  description
    "SRv6 SID Function length in bits.";
}

leaf arg-length {
  type uint8;
  description
    "SRv6 SID Argument length in bits.";
}
}
}
grouping srv6-capability {
  description
    "SRV6 capability grouping.";
  container v6-capability {
    description
      "SRv6 capability.";
    leaf flags {
      type bits {
        bit o-flag {
          position 1;
          description
            "If set, then the router is able to
             use of the O-bit in the Segment Routing Header(SRH)
             as defined in [draft-ietf-6man-segment-routing-header].";
        }
      }
    }
    description
      "Flags.";
  }
}
```

```
    }
  }

  grouping srv6-endpoint-func {
    description
      "This group defines srv6 endpoint function";
    container endpoint-func {
      description
        "Srv6 Endpoint function Descriptor.";
      leaf flags {
        type uint8;
        description
          "No flags are currently being defined.";
      }
      leaf endpoint-func {
        type identityref {
          base isis-srv6:SRV6_END_FUNC_TYPE;
        }
        description
          "The endpoint function.";
      }
      leaf undefined-endpoint-func {
        type uint16;
        description
          "Unknown endpoint func value.";
      }
    }
  }

  grouping srv6-end-sid {
    description
      "SRv6 Segment Identifier(SID) with Endpoint functions.";
    leaf flags {
      type uint8;
      description
        "NO flags are currently being defined.";
    }

    uses srv6-endpoint-func;

    leaf sid {
      type srv6-sid-value;
      description
        "SRV6 sid value.";
    }
    // sub-sub-tlvs
    uses srv6-sid-structures;
  }
```

```
grouping srv6-locator {
  description
    "This group defines srv6 locator tlv.";
  leaf mt-id {
    type uint16 {
      range "0..4095";
    }
    description
      "Multitopology Identifier as defined in [RFC5120].";
  }
  leaf flags {
    type bits {
      bit d-flag {
        position 0;
        description
          "When the locator is leaked from level-2 to leve-1,
           the d-flag must be set.";
      }
    }
    description
      "Flags for srv6 locator tlv.";
  }

  leaf metric {
    type uint32;
    description
      "Metric value.";
  }
  leaf algorithm {
    type uint8;
    description
      "Associated algorithm.";
  }

  leaf loc-size {
    type uint8;
    description
      "Number of bits in the locator field.";
  }
  leaf locator {
    type inet:ipv6-address-no-zone;
    description
      "Advertised SRV6 locator.";
  }
  container srv6-end-sids {
    description
      "This contains list of srv6 end sids.";
    list end-sid {
```

```
        key "sid";
        description
            "List of SRV6 SRv6 Segment Identifiers (SID)
             with Endpoints.";
        uses srv6-end-sid;
    }
}
uses isis:prefix-reachability-attributes;
uses isis:prefix-ipv4-source-router-id;
uses isis:prefix-ipv6-source-router-id;
}

grouping srv6-adjacency-sid {
    description
        "SRv6 sid associated with an adjacency.";

    leaf func-flags {
        type bits {
            bit b-flag {
                position 0;
                description
                    "Backup flag. If set, the End.X sid is
                     eligible for protection.";
            }

            bit s-flag {
                position 1;
                description
                    "Set flag. When set, the End.X sid refers to
                     a set of adjacencies (and therefore May be assigned
                     to other adjacencies as well.";
            }

            bit p-flag {
                position 2;
                description
                    "Persistent flag. When set, the End.X sid is persistently
                     allocated, i.e., the End.x sid value remains consistent
                     across router restart and/or interface flap.";
            }
        }
        description
            "Flags for srv6 end x sid.";
    }

    leaf algorithm {
        type uint8;
        description
```

```
        "Associated algorithm.";
    }

    leaf weight {
        type uint8;
        description
            "The value represents the weight of the End.X sid
            for the purpose of load balancing.";
    }

    uses srv6-endpoint-func;

    leaf sid {
        type srv6-sid-value;
        description
            "SRV6 sid value.";
    }

    leaf neighbor-id {
        type isis:system-id;
        description
            "Describes the system ID of the neighbor
            associated with the SID value. This is only
            used on LAN adjacencies.";
    }

    // sub-sub-tlvs
    uses srv6-sid-structures;
}

grouping srv6-adjacency-state {
    description
        "This group will extend adjacency state.";
    list end-x-sid {
        key value;
        config false;
        leaf value {
            type srv6-sid-value;
            description
                "Value of the Adj-SID.";
        }
        leaf weight {
            type uint8;
            description
                "Weight associated with
                the End.X SID.";
        }
        leaf protection-requested {
```

```
        type boolean;
        description
            "Set to True if the End.X SID
            must be protected.";
    }
    leaf persistent {
        type boolean;
        description
            "Persistent flag. When set, the End.X sid is persistently
            allocated, i.e., the End.X sid value remains consistent
            across router restart and/or interface flap.";
    }
    leaf algorithm {
        type uint8;
        description
            "Associated algorithm.";
    }
    uses srv6-endpoint-func;

    description
        "List of End.X Segment IDs.";
}
}
/* Cfg */
augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis" {
    when "/rt:routing/rt:control-plane-protocols/" +
        "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
            "This augment ISIS routing protocol when used";
    }
    description
        "This augments ISIS protocol configuration
        with SRv6.";

    container srv6-cfg{
        leaf enable{
            type boolean;
            default "false";
            description
                "Enables SRv6
                protocol extensions.";
        }

        leaf default-locator {
            type boolean;
            default "false";
        }
    }
}
```

```
    description
      "Enable ISIS segment-routing IPv6 with default Locator.";
  }

  leaf-list locator-name {
    when "../default-locator = 'false'" {
      description
        "Only applies to non default locator.";
    }
    type leafref {
      path "/rt:routing/sr:segment-routing/srv6:srv6" +
        "/srv6:locators/srv6:locator/srv6:name";
    }
    description
      "Enable ISIS segment-routing IPv6 with specified Locator.";
  }

  leaf persistent-end-x-sid{
    type boolean;
    default "false";
    description
      "Enable the persistent nature of End.X sid";
  }
  description
    "Configuration about ISIS segment-routing IPv6.";
}
container micro-loop-avoidance {
  leaf srv6-enable {
    type boolean;
    default "false";
    description
      "Enable SRv6 avoid-microloop.Depend on SR IPv6 Enable.";
  }

  leaf srv6-rib-update-delay {
    type uint16 {
      range "1000..10000";
    }
    units "ms";
    default "5000";
    description
      "Set the route delivery delay for SRv6 avoid-microloop.
      Depend on SR IPv6 Enable.";
  }

  description
```



```
        "Enable IS-IS avoid-microloop.";
    }
}

augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:fast-reroute"{
    when "/rt:routing/rt:control-plane-protocols/"+
        "rt:control-plane-protocol/rt:type = 'isis:isis'"{
        description
            "This augment ISIS routing protocol when used";
    }
    description
        "This augments ISIS IPFRR with IPV6 TILFA.";

    container srv6-ti-lfa {

        if-feature srv6-ti-lfa;
        leaf enable {
            type boolean;
            description
                "Enables SRv6 TI-LFA computation.";
        }

        description
            "SRv6 TILFA configuration.";
    }
}

/* Operational states */
augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:database/isis:levels/isis:lsp"+
    "/isis:router-capabilities" {
    when "/rt:routing/rt:control-plane-protocols/"+
        "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
            "This augment ISIS routing protocol when used";
    }
    description
        "This augments ISIS protocol router capability.";
    uses srv6-capability;
    uses srv6-msds;
}

augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:database/isis:levels/isis:lsp"+
```

```
    "/isis:extended-is-neighbor/isis:neighbor" {
when "/rt:routing/rt:control-plane-protocols/"+
    "rt:control-plane-protocol/rt:type = 'isis:isis'" {
    description
    "This augment ISIS routing protocol when used.";
}
description
    "This augments ISIS protocol neighbor.";
container srv6-adjacency-sids {
    description
    "This defines svr6 end-x sids for the adjacency.";
    list end-x-sid {
        key "sid";
        uses srv6-adjacency-sid;
        description
        "List of end-x sids.";
    }
}
}
augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:database/isis:levels/isis:lsp"+
    "/isis:mt-is-neighbor/isis:neighbor" {
    when "/rt:routing/rt:control-plane-protocols/"+
    "rt:control-plane-protocol/rt:type = 'isis:isis'" {
    description
    "This augment ISIS routing protocol when used.";
}
description
    "This augments ISIS protocol neighbor.";
container srv6-adjacency-sids {
    description
    "This defines svr6 end-x sids for the adjacency.";
    list end-x-sid {
        key "sid";
        uses srv6-adjacency-sid;
        description
        "List of end-x sids.";
    }
}
}

augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:database/isis:levels/isis:lsp" {
    when "/rt:routing/rt:control-plane-protocols/"+
    "rt:control-plane-protocol/rt:type = 'isis:isis'" {
    description
```

```

        "This augment ISIS routing protocol when used.";
    }
    description
        "This augments ISIS protocol LSDB.";
    container srv6-locators {
        description
            "This defines srv6 locator tlvs.";
        list locator {
            key "locator";
            uses srv6-locator;
            description
                "List of srv6 locators.";
        }
    }
}

augment "/rt:routing/" +
    "rt:control-plane-protocols/rt:control-plane-protocol"+
    "/isis:isis/isis:interfaces/isis:interface" +
    "/isis:adjacencies/isis:adjacency" {
    when "/rt:routing/rt:control-plane-protocols/"+
        "rt:control-plane-protocol/rt:type = 'isis:isis'" {
        description
            "This augment ISIS routing protocol when used.";
    }
    description
        "This augments ISIS protocol operational state
        with segment routing.";
    uses srv6-adjacency-state;
}
/* Notifications */
}
<CODE ENDS>

```

5. Security Considerations

Configuration and state data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241].

As IS-IS is an IGP protocol (critical piece of the network), ensuring stability and security of the protocol is mandatory for the network service.

Authors recommends to implement NETCONF access control model ([RFC6536]) to restrict access to all or part of the configuration to specific users.

6. Contributors

Jiajia Dong
Huawei
China

Email: dongjiajia@huawei.com

7. Acknowledgements

TBD.

8. IANA Considerations

The IANA is requested to assign two new URIs from the IETF XML registry ([RFC3688]). Authors are suggesting the following URI:

URI: urn:ietf:params:xml:ns:yang:ietf-isis-srv6
Registrant Contact: IS-IS WG
XML: N/A, the requested URI is an XML namespace

This document also requests one new YANG module name in the YANG Module Names registry ([RFC6020]) with the following suggestion :

name: ietf-isis-srv6
namespace: urn:ietf:params:xml:ns:yang:ietf-isis-srv6 prefix: isis-srv6
reference: RFC XXXX

9. References

- [I-D.ietf-isis-yang-isis-cfg]
Litkowski, S., Yeung, D., Lindem, A., Zhang, J., and L. Lhotka, "YANG Data Model for IS-IS Protocol", draft-ietf-isis-yang-isis-cfg-42 (work in progress), October 2019.
- [I-D.ietf-lsr-isis-srv6-extensions]
Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extension to Support Segment Routing over IPv6 Dataplane", draft-ietf-lsr-isis-srv6-extensions-14 (work in progress), April 2021.
- [I-D.ietf-spring-sr-yang]
Litkowski, S., Qu, Y., Lindem, A., Sarkar, P., and J. Tantsura, "YANG Data Model for Segment Routing", draft-ietf-spring-sr-yang-30 (work in progress), January 2021.

- [I-D.ietf-spring-srv6-yang]
Raza, K., Agarwal, S., Liu, X., Hu, Z., Hussain, I., Shah, H., Voyer, D., Matsushima, S., Horiba, K., AbdelSalam, A., and J. Rajamanickam, "YANG Data Model for SRv6 Base and Static", draft-ietf-spring-srv6-yang-00 (work in progress), September 2020.
- [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>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.

Appendix A. Configuration examples

The following is an XML example using IS-IS SRv6 YANG module.

```
<routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
  <control-plane-protocols>
    <control-plane-protocol>
      <isis xmlns="urn:ietf:params:xml:ns:yang:ietf-isis">
        <srv6-cfg>
          <enable>true</enable>
        </srv6-cfg>
        <default-locator>false</default-locator>
        <locator-name>DOM0_ALG0</locator-name>
        <persistent-end-x-sid>true</persistent-end-x-sid>
        </srv6-cfg>
        <micro-loop-avoidance>
          <srv6-enable>true</srv6-enable>
        </micro-loop-avoidance>
        <srv6-rib-update-delay>2000</srv6-rib-update-delay>
      </isis>
    </control-plane-protocol>
  </control-plane-protocols>
</routing>

<routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
  <control-plane-protocols>
    <control-plane-protocol>
      <isis xmlns="urn:ietf:params:xml:ns:yang:ietf-isis">
        <fast-reroute>
          <srv6-ti-lfa>
            <enable>true</enable>
          </srv6-ti-lfa>
        </fast-reroute>
      </isis>
    </control-plane-protocol>
  </control-plane-protocols>
</routing>
```

The following is the corresponding example using JSON format.

```
{
  "control-plane-protocols": {
    "control-plane-protocol": {
      "isis": {
        "srv6-cfg": {
          "enable": "true",
          "default-locator": "false",
          "locator-name": "DOM0_ALG0",
          "persistent-end-x-sid": "true"
        },
        "micro-loop-avoidance": {
          "srv6-enable": "true",
          "srv6-rib-update-delay": "2000"
        }
      }
    }
  }
}

{
  "control-plane-protocols": {
    "control-plane-protocol": {
      "isis": {
        "fast-reroute": {
          "srv6-ti-lfa": {
            "enable": "true"
          }
        }
      }
    }
  }
}
```

Authors' Addresses

Zhibo Hu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: huzhibo@huawei.com

Dan Ye
Cisco
170 W. Tasman Dr. San Jose,
California 95134
USA

Email: daye@cisco.com

Yingzhen Qu
Futurewei
2330 Central Express Way
Santa Clara 950950
USA

Email: yingzhen.qu@futurewei.com

Xuesong Geng
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: gengxuesong@huawei.com

Qiufang Ma
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: maqiufang1@huawei.com

SPRING
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2020

C. Filsfils
P. Camarillo, Ed.
Cisco Systems, Inc.
J. Leddy
Individual Contributor
D. Voyer
Bell Canada
S. Matsushima
SoftBank
Z. Li
Huawei Technologies
July 3, 2019

SRv6 Network Programming
draft-ietf-spring-srv6-network-programming-01

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 January 4, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	4
3. SRv6 Segment	5
4. Functions associated with a SID	7
4.1. End: Endpoint	8
4.2. End.X: Layer-3 cross-connect	9
4.3. End.T: Specific IPv6 table lookup	10
4.4. End.DX2: Decapsulation and L2 cross-connect	10
4.5. End.DX2V: Decapsulation and VLAN L2 table lookup	11
4.6. End.DT2U: Decapsulation and unicast MAC L2 table lookup	12
4.7. End.DT2M: Decapsulation and L2 table flooding	12
4.8. End.DX6: Decapsulation and IPv6 cross-connect	13
4.9. End.DX4: Decapsulation and IPv4 cross-connect	14
4.10. End.DT6: Decapsulation and specific IPv6 table lookup	15
4.11. End.DT4: Decapsulation and specific IPv4 table lookup	15
4.12. End.DT46: Decapsulation and specific IP table lookup	16
4.13. End.B6.Insert: Endpoint bound to an SRv6 policy	17
4.14. End.B6.Insert.Red: [...] with reduced SRH insertion	18
4.15. End.B6.Encaps: Endpoint bound to an SRv6 policy w/ encaps	18
4.16. End.B6.Encaps.Red: [...] with reduced SRH insertion	19
4.17. End.BM: Endpoint bound to an SR-MPLS policy	19
4.18. End.S: Endpoint in search of a target in table T	20
4.19. SR-aware application	21
4.20. Non SR-aware application	21
4.21. Flavours	21
4.21.1. PSP: Penultimate Segment Pop of the SRH	21
4.21.2. USP: Ultimate Segment Pop of the SRH	22
4.21.3. USD: Ultimate Segment Decapsulation	23
5. Transit behaviors	24
5.1. T: Transit behavior	24
5.2. T.Insert: Transit with insertion of an SRv6 Policy	24

5.3.	T.Insert.Red: Transit with reduced insertion	25
5.4.	T.Encaps: Transit with encapsulation in an SRv6 Policy .	25
5.5.	T.Encaps.Red: Transit with reduced encapsulation	26
5.6.	T.Encaps.L2: Transit with encapsulation of L2 frames . .	27
5.7.	T.Encaps.L2.Red: Transit with reduced encaps of L2 frames	27
6.	Operation	28
6.1.	Counters	28
6.2.	Flow-based hash computation	28
6.3.	OAM	28
7.	Basic security for intra-domain deployment	29
7.1.	SEC-1	29
7.2.	SEC-2	30
7.3.	SEC-3	30
8.	Control Plane	31
8.1.	IGP	31
8.2.	BGP-LS	31
8.3.	BGP IP/VPN/EVPN	31
8.4.	Summary	32
9.	IANA Considerations	33
10.	Work in progress	35
11.	Acknowledgements	35
12.	Contributors	35
13.	References	38
13.1.	Normative References	38
13.2.	Informative References	38
	Authors' Addresses	40

1. Introduction

Segment Routing leverages the source routing paradigm. An ingress node steers a packet through a 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 functions 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

SRH is the abbreviation for the Segment Routing Header. We assume that the SRH may be present multiple times inside each packet.

NH is the abbreviation of the IPv6 next-header field.

NH=SRH means that the next-header field is 43 with routing type 4.

When there are multiple SRHs, they must follow each other: the next-header field of all SRH, except the last one, must be SRH.

The effective next-header (ENH) is the next-header field of the IP header when no SRH is present, or is the next-header field of the last SRH.

In this version of the document, we assume that there are no other extension headers than the SRH. These will be lifted in future versions of the document.

SID: A Segment Identifier which represents a specific segment in segment routing domain. The SID type used in this document is IPv6 address (also referenced as SRv6 Segment or SRv6 SID).

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:

- IPv6 header with source address SA, destination addresses DA and SRH as next-header
- SRH with SID list <S1, S2, S3> with SegmentsLeft = 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.

SRH[SL] represents the SID pointed by the SL field in the first SRH. In our example, SRH[2] represents S1, SRH[1] represents S2 and SRH[0] represents S3.

FIB is the abbreviation for the forwarding table. A FIB lookup is a lookup in the forwarding table.

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

3. SRv6 Segment

An SRv6 Segment is a 128-bit value. "SID" (abbreviation for Segment Identifier) is often used as a shorter reference for "SRv6 Segment".

An SRv6-capable node N maintains a "My SID Table". This table contains all the SRv6 segments explicitly instantiated at node N. N is the parent node for these SIDs.

A local SID of N can be an IPv6 address associated to a local interface of N but it is not mandatory. Nor is the "My SID table" populated by default with all IPv6 addresses defined on node N.

In most use-cases, a local SID will NOT be an address associated to a local interface of N.

A local SID of N could be routed to N but it does not have to be. Most often, it is routed to N via a shorter-mask prefix.

Let's provide a classic illustration.

Node N is configured with a loopback0 interface address of A:1::/32 originated in its IGP. Node N is configured with two SIDs: B:1:100:: and B:2:101::.

The entry A:1:: is not defined explicitly as an SRv6 SID and hence does not appear in the "My SID Table". The entries B:1:100:: and B:2:101:: are defined explicitly as SRv6 SIDs and hence appear in the "My SID Table".

The network learns about a path to B:1::/32 via the IGP and hence a packet destined to B:1:100:: would be routed up to N. The network does not learn about a path to B:2::/32 via the IGP and hence a packet destined to B:2:101:: would not be routed up to N.

A packet could be steered to a non-routed SID B:2:101:: by using a SID list <...,B:1:100::,B:2:101::,...> where the non-routed SID is preceded by a routed SID to the same node. This is similar to the local vs global segments in SR-MPLS.

Every SRv6 SID instantiated has a specific instruction bound to it. This information is stored in the "My SID Table". The "My SID Table" has three main purposes:

- Define which SIDs are explicitly instantiated on that node
- Specify which instruction is bound to each of the instantiated SIDs
- Store the parameters associated with such instruction (i.e. OIF, NextHop, VRF,...)

We represent an SRv6 SID as LOC:FUNCT where LOC (locator) is the L most significant bits and FUNCT (function) is the 128-L least significant bits. 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 function bound to the SID. The FUNCT value zero is invalid.

Often, for simplicity of illustration, we will use a locator length of 32 bits. This is just an example. Implementations must not assume any a priori prefix length.

A function 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 "My SID Table" matches on a per longest-prefix-match basis.

These arguments may vary on a per-packet basis and may contain information related to the flow, service, or any other information required by the function associated to the SRv6 SID.

A node may receive a packet with an SRv6 SID in the DA without an SRH. In such case the packet should still be processed by the Segment Routing engine.

4. Functions associated with a SID

Each entry of the "My SID Table" indicates the function associated with the local SID and its parameters.

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

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

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 function on the packet.

We call N the node who has an explicitly instantiated SID S and we detail the function that N binds to S.

At the end of this section we also present some flavours of these well-known functions.

4.1. End: Endpoint

The Endpoint function ("End" for short) is the most basic function.

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

1. IF NH=SRH and SL > 0
2. decrement SL
3. update the IPv6 DA with SRH[SL]
4. FIB lookup on the updated DA ;; Ref1
5. forward accordingly to the matched entry ;; Ref2
6. ELSE IF NH!=SRH
7. Send an ICMP parameter problem message; drop the packet ;; Ref3
8. ELSE
9. drop the packet

Ref1: The End function performs the FIB lookup in the forwarding table associated to the ingress interface

Ref2: If the FIB lookup matches a multicast state, then the related RPF check must be considered successful

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

A local SID could be bound to a function which authorizes the decapsulation of an outer header (e.g. IPinIP) or the punting of the packet to TCP, UDP or any other protocol. This however needs to be explicitly defined in the function bound to the local SID. By default, a local SID bound to the well-known function "End" neither allows the decapsulation of an outer header nor the cleanup of an SRH. As a consequence, an End SID is not the last SID of an SRH or the DA of a packet without SRH, unless combined with the flavours defined in Section 4.21.

This is the SRv6 instantiation of a Prefix SID [RFC8402].

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

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

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

1. IF NH=SRH and SL > 0
2. decrement SL
3. update the IPv6 DA with SRH[SL]
4. forward to layer-3 adjacency bound to the SID S ;; Ref1
6. ELSE IF NH!=SRH
7. Send an ICMP parameter problem message; drop the packet ;; Ref2
8. ELSE
9. drop the packet

Ref1: If an array of adjacencies is bound to the End.X SID, then one entry of the array is selected based on a hash of the packet's header.

Ref2: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

The End.X function is required to express any traffic-engineering policy.

This is the SRv6 instantiation of an Adjacency SID [RFC8402].

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 with SR-MPLS, an AdjSID is typically preceded by a PrefixSID. This is unlikely in SRv6 as most likely an End.X SID is globally routed to N.

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: one for the bundle itself and then up to one for each member link. This is the equivalent of the L2-Link Adj SID in SR-MPLS [I-D.ietf-isis-l2bundles].

4.3. End.T: Specific IPv6 table lookup

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

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

1. IF NH=SRH and SL > 0 ;; Ref1
2. decrement SL
3. update the IPv6 DA with SRH[SL]
4. lookup the next segment in IPv6 table T associated with the SID
5. forward via the matched table entry
6. ELSE IF NH!=SRH
7. Send an ICMP parameter problem message; drop the packet ;; Ref2
8. ELSE
9. drop the packet

Ref1: The End.T SID must not be the last SID

Ref2: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

The End.T is used for multi-table operation in the core.

4.4. End.DX2: Decapsulation and L2 cross-connect

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

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

1. IF NH=SRH and SL > 0
2. drop the packet ;; Ref1
3. ELSE IF ENH=59 ;; Ref2
4. pop the (outer) IPv6 header and its extension headers
5. forward the resulting frame to OIF bound to the SID S
6. ELSE
7. Send an ICMP parameter problem message ;; Ref3
8. drop the packet

Ref1: An End.DX2 SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: The next-header value 59 (IPv6 No Next Header [RFC8200]) identifies that there is no further Internet Protocol header to be processed in the packet. When the SID corresponds to function

End.DX2 and the Next-Header value is 59, we know that an Ethernet frame is in the payload without any further header.

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

An End.DX2 function could be customized to expect a specific VLAN format and rewrite the egress VLAN header before forwarding on the outgoing interface.

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

4.5. End.DX2V: Decapsulation and VLAN L2 table lookup

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

When N receives a packet destined to S and S is a local End.DX2V SID, N does:

1. IF NH=SRH and SL > 0
2. drop the packet ;; Ref1
3. ELSE IF ENH = 59 ;; Ref2
4. pop the (outer) IPv6 header and its extension headers
5. lookup the exposed inner VLANs in L2 table T
6. forward via the matched table entry
7. ELSE
8. Send an ICMP parameter problem message ;; Ref3
9. drop the packet

Ref1: An End.DX2V SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: The next-header value 59 (IPv6 No Next Header [RFC8200]) identifies that there is no further Internet Protocol header to be processed in the packet. When the SID corresponds to function End.DX2 and the Next-Header value is 59, we know that an Ethernet frame is in the payload without any further header.

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

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

The End.DX2V is used for EVPN Flexible cross-connect use-cases.

4.6. End.DT2U: Decapsulation and unicast MAC L2 table lookup

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

When N receives a packet destined to S and S is a local End.DT2U SID, N does:

```
1.  IF NH=SRH and SL > 0
2.      drop the packet                                ;; Ref1
3.  ELSE IF ENH = 59                                    ;; Ref2
4.      pop the (outer) IPv6 header and its extension headers
5.      learn the exposed inner MAC SA in L2 table T    ;; Ref3
6.      lookup the exposed inner MAC DA in L2 table T
7.      IF matched entry in table T
8.          forward via the matched table T entry
9.      ELSE
10.         forward via all L2OIF entries in table T
11.  ELSE
12.      Send an ICMP parameter problem message          ;; Ref4
13.      drop the packet
```

Ref1: An End.DT2U SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: The next-header value 59 (IPv6 No Next Header [RFC8200]) identifies that there is no further Internet Protocol header to be processed in the packet. When the SID corresponds to function End.DX2 and the Next-Header value is 59, we know that an Ethernet frame is in the payload without any further header.

Ref3: In EVPN, the learning of the exposed inner MAC SA is done via control plane.

Ref4: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

The End.DT2U is used for EVPN Bridging unicast use cases.

4.7. End.DT2M: Decapsulation and L2 table flooding

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

This function may take an argument: "Arg.FE2". It is an argument specific to EVPN ESI filtering. It is used to exclude a specific OIF (or set of OIFs) from L2 table T flooding.

When N receives a packet destined to S and S is a local End.DT2M SID, N does:

1. IF NH=SRH and SL > 0
2. drop the packet ;; Ref1
3. ELSE IF ENH = 59 ;; Ref2
4. pop the (outer) IPv6 header and its extension headers
3. learn the exposed inner MAC SA in L2 table T ;; Ref3
4. forward on all L2OIF excluding the one specified in Arg.FE2
5. ELSE
6. Send an ICMP parameter problem message ;; Ref4
7. drop the packet

Ref1: An End.DT2M SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: The next-header value 59 (IPv6 No Next Header [RFC8200]) identifies that there is no further Internet Protocol header to be processed in the packet. When the SID corresponds to function End.DX2 and the Next-Header value is 59, we know that an Ethernet frame is in the payload without any further header.

Ref3: In EVPN, the learning of the exposed inner MAC SA is done via control plane

Ref4: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

The End.DT2M is used for EVPN Bridging BUM use-case with ESI filtering capability.

4.8. End.DX6: Decapsulation and IPv6 cross-connect

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

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

1. IF NH=SRH and SL > 0
2. drop the packet ;; Ref1
3. ELSE IF ENH = 41 ;; Ref2
4. pop the (outer) IPv6 header and its extension headers
5. forward to layer-3 adjacency bound to the SID S ;; Ref3
6. ELSE
7. Send an ICMP parameter problem message ;; Ref4
8. drop the packet

Ref1: The End.DX6 SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: 41 refers to IPv6 encapsulation as defined by IANA allocation for Internet Protocol Numbers

Ref3: Selected based on a hash of the packet's header (at least SA, DA, Flow Label)

Ref4: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

One of the applications of the End.DX6 function is the L3VPNV6 use-case where a FIB lookup in a specific tenant table at the egress PE is not required. This would be equivalent to the per-CE VPN label in MPLS [RFC4364].

4.9. End.DX4: Decapsulation and IPv4 cross-connect

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

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

```
1.  IF NH=SRH and SL > 0
2.      drop the packet                                ;; Ref1
3.  ELSE IF ENH = 4                                     ;; Ref2
4.      pop the (outer) IPv6 header and its extension headers
5.      forward to layer-3 adjacency bound to the SID S    ;; Ref3
6.  ELSE
7.      Send an ICMP parameter problem message            ;; Ref4
8.      drop the packet
```

Ref1: The End.DX4 SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: 4 refers to IPv4 encapsulation as defined by IANA allocation for Internet Protocol Numbers

Ref3: Selected based on a hash of the packet's header (at least SA, DA, Flow Label)

Ref4: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

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

4.10. End.DT6: Decapsulation and specific IPv6 table lookup

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

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

1. IF NH=SRH and SL > 0
2. drop the packet ;; Ref1
3. ELSE IF ENH = 41 ;; Ref2
4. pop the (outer) IPv6 header and its extension headers
5. lookup the exposed inner IPv6 DA in IPv6 table T
6. forward via the matched table entry
7. ELSE
8. Send an ICMP parameter problem message ;; Ref3
9. drop the packet

Ref1: the End.DT6 SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: 41 refers to IPv6 encapsulation as defined by IANA allocation for Internet Protocol Numbers

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

One of the applications of the End.DT6 function 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 End.DT6 supports the equivalent of an IPv6inIPv6 decaps (without VPN/tenant implication).

4.11. End.DT4: Decapsulation and specific IPv4 table lookup

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

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

```
1.  IF NH=SRH and SL > 0
2.      drop the packet                                ;; Ref1
3.  ELSE IF ENH = 4                                    ;; Ref2
4.      pop the (outer) IPv6 header and its extension headers
5.      lookup the exposed inner IPv4 DA in IPv4 table T
6.      forward via the matched table entry
7.  ELSE
8.      Send an ICMP parameter problem message          ;; Ref3
9.      drop the packet
```

Ref1: the End.DT4 SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: 4 refers to IPv4 encapsulation as defined by IANA allocation for Internet Protocol Numbers

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

One of the applications of the End.DT4 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 and End.DT4 supports the equivalent of an IPv4inIPv6 decaps (without VPN/tenant implication).

4.12. End.DT46: Decapsulation and specific IP table lookup

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

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


```
1.  IF NH=SRH and SL > 0
2.      drop the packet                                ;; Ref1
3.  ELSE IF ENH = 4                                     ;; Ref2
4.      pop the (outer) IPv6 header and its extension headers
5.      lookup the exposed inner IPv4 DA in IPv4 table T
6.      forward via the matched table entry
7.  ELSE IF ENH = 41                                    ;; Ref2
8.      pop the (outer) IPv6 header and its extension headers
9.      lookup the exposed inner IPv6 DA in IPv6 table T
10.     forward via the matched table entry
11.  ELSE
12.      Send an ICMP parameter problem message        ;; Ref3
13.      drop the packet
```

Ref1: the End.DT46 SID must always be the last SID, or it can be the Destination Address of an IPv6 packet with no SRH header.

Ref2: 4 and 41 refer to IPv4 and IPv6 encapsulation respectively as defined by IANA allocation for Internet Protocol Numbers

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

One of the applications of the End.DT46 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 the per-VRF VPN label in MPLS [RFC4364].

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

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

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

When N receives a packet destined to S and S is a local End.B6.Insert SID, N does:

1. IF NH=SRH and SL > 0 ;; Ref1
2. do not decrement SL nor update the IPv6 DA with SRH[SL]
3. insert a new SRH, in between the IPv6 header and the received SRH ;; Ref2
4. set the IPv6 DA to the first segment of the SRv6 Policy
5. forward according to the first segment of the SRv6 Policy
6. ELSE
7. Send an ICMP parameter problem message ;; Ref3
8. drop the packet

Ref1: An End.B6.Insert SID, by definition, is never the last SID.

Ref2: [I-D.voyer-6man-extension-header-insertion]

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

Note: Instead of the term "insert", "push" may also be used.

The End.B6.Insert function is required to express scalable traffic-engineering policies across multiple domains. This is the SRv6 instantiation of a Binding SID [RFC8402].

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

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

End.B6.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.

Note that SL value is initially pointing to a non-existing segment in the SRH.

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

This is a variation of the End.B6.Insert behavior where the SRv6 Policy also includes an IPv6 Source Address A.

When N receives a packet destined to S and S is a local End.B6.Encaps SID, N does:

1. IF NH=SRH and SL > 0
2. decrement SL and update the IPv6 DA with SRH[SL]
3. push an outer IPv6 header with its own SRH
4. set the outer IPv6 SA to A
5. set the outer IPv6 DA to the first segment of the SRv6 Policy
6. set outer payload length, traffic class and flow label ;; Ref1,2
7. update the Next-Header value ;; Ref1
8. decrement inner Hop Limit or TTL ;; Ref1
9. forward according to the first segment of the SRv6 Policy
10. ELSE
11. Send an ICMP parameter problem message ;; Ref3
12. drop the packet

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

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

Ref3: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

Instead of simply inserting an SRH with the policy (End.B6), this behavior also adds an outer IPv6 header. The source address defined for the outer header does not have to be a local SID of the node.

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.

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

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

End.B6.Encaps.Red will reduce the size of the SRH by one segment 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.

Note that SL value is initially pointing to a non-existing segment in the SRH.

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.B6 function.

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

1. IF NH=SRH and SL > 0 ;; Ref1
2. decrement SL and update the IPv6 DA with SRH[SL]
3. push an MPLS label stack <L1, L2, L3> on the received packet
4. forward according to L1
5. ELSE
6. Send an ICMP parameter problem message ;; Ref2
7. drop the packet

Ref1: an End.BM SID, by definition, is never the last SID.

Ref2: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

The End.BM function 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].

4.18. End.S: Endpoint in search of a target in table T

The "Endpoint in search of a target in Table T" function (End.S for short) is a variant of the End function.

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

1. IF NH=SRH and SL = 0 ;; Ref1
2. Send an ICMP parameter problem message ;; Ref2
3. drop the packet
4. ELSE IF match(last SID) in specified table T
5. forward accordingly
6. ELSE
7. apply the End behavior

Ref1: By definition, an End.S SID cannot be the last SID, as the last SID is the targeted object.

Ref2: ICMP error is sent to the source address with error code (TBD by IANA) "SR Upper-layer Header Error" and pointer set to the NH.

The End.S function is required in information-centric networking (ICN) use-cases where the last SID in the SRv6 SID list represents a targeted object. If the identification of the object would require more than 128 bits, then obvious customization of the End.S function

may either use multiple SIDs or a TLV of the SR header to encode the searched object ID.

4.19. SR-aware application

Generally, any SR-aware application can be bound to an SRv6 SID. This application could represent anything from a small piece of code focused on topological/tenant function to a larger process focusing on higher-level applications (e.g. video compression, transcoding etc.).

The ways in which an SR-aware application binds itself on a local SID depends on the operating system. Let us consider an SR-aware application running on a Linux operating system. A possible approach is to associate an SRv6 SID to a target (virtual) interface, so that packets with IP DA corresponding to the SID will be sent to the target interface. In this approach, the SR-aware application can simply listen to all packets received on the interface.

A different approach for the SR-aware app is to listen to packets received with a specific SRv6 SID as IPv6 DA on a given transport port (i.e. corresponding to a TCP or UDP socket). In this case, the app can read the SRH information with a `getsockopt` Linux system call and can set the SRH information to be added to the outgoing packets with a `setsockopt` system call.

4.20. Non SR-aware application

[I-D.xuclad-spring-sr-service-programming] defines a set of additional functions in order to enable non SR-aware applications to be associated with an SRv6 SID.

4.21. Flavours

We present the PSP, USP and USD variants of the functions `End`, `End.X` and `End.T`. For each of these functions these variants can be enabled or disabled either individually or together.

4.21.1. PSP: Penultimate Segment Pop of the SRH

After the instruction 'update the IPv6 DA with SRH[SL]' is executed, the following instructions must be added:

1. IF updated SL = 0 & PSP is TRUE
2. pop the top SRH ;; Ref1

Ref1: The received SRH had SL=1. When the last SID is written in the DA, the `End`, `End.X` and `End.T` functions with the PSP flavour pop the

first (top-most) SRH. Subsequent stacked SRH's may be present but are not processed as part of the function.

4.21.2. USP: Ultimate Segment Pop of the SRH

We insert at the beginning of the pseudo-code the following instructions:

1. IF NH=SRH & SL = 0 & USP=TRUE ;; Ref1
2. pop the top SRH
3. restart the function processing on the modified packet ;; Ref2

Ref1: The next header is an SRH header

Ref2: Typically SL of the exposed SRH is > 0 and hence the restarting of the complete function would lead to decrement SL, update the IPv6 DA with SRH[SL], FIB lookup on updated DA and forward accordingly to the matched entry.

4.21.3. USD: Ultimate Segment Decapsulation

We insert at the beginning of the pseudo-code the following instructions:

1. IF (NH=41) or (NH = SRH and SL = 0 and NNH = 41)
2. pop the (outer) IPv6 header and its extension headers
3. lookup the exposed inner IP DA and forward ;; Ref1
4. forward via the matched table entry

Ref1: In case that the USD flavor is applied on an End.X function, the packet is forwarded to the layer-3 adjacency bound to SID S without any lookup.

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, Reflbis
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.

Reflbis: 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.

Note that SL value is initially pointing to a non-existing segment in
the SRH.

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.

Note that SL value is initially pointing to a non-existing segment in the SRH.

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.

Note that SL value is initially pointing to a non-existing segment in the SRH.

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 entry of the "My SID Table", 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].

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 functions.

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 function.

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 functions and hence are expected to be signaled in the IGP together with the flavours 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 functions 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.Insert 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.DX2		X	X
End.DX2V		X	X
End.DT2U		X	X
End.DT2M		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.B6.Insert		X	
End.B6.Insert.Red		X	
End.B6.Encaps		X	
End.B6.Encaps.Red		X	
End.B6.BM		X	
End.S		X	

Table 1: SRv6 locally instantiated 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	Opaque
65535	0xFFFF	Reserved	

Table 3: SRv6 Endpoint Behaviors Registry

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

Value	Hex	Endpoint function	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	[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	End.S	[This.ID]
26	0x001A	End.B6.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-21 (work in progress), June 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-05 (work in progress), January 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-00 (work in progress), February 2019.

[I-D.ietf-isis-l2bundles]

Ginsberg, L., Bashandy, A., Filsfils, C., Nanduri, M., and E. Aries, "Advertising L2 Bundle Member Link Attributes in IS-IS", draft-ietf-isis-l2bundles-07 (work in progress), May 2017.

[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-03 (work in progress), May 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
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

SPRING
Internet-Draft
Intended status: Standards Track
Expires: July 2, 2021

C. Filsfils, Ed.
P. Camarillo, Ed.
Cisco Systems, Inc.
J. Leddy
Individual Contributor
D. Voyer
Bell Canada
S. Matsushima
SoftBank
Z. Li
Huawei Technologies
December 29, 2020

SRv6 Network Programming
draft-ietf-spring-srv6-network-programming-28

Abstract

The SRv6 Network Programming framework 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 in the packet.

This document defines the SRv6 Network Programming concept and specifies the base set of SRv6 behaviors that enables the creation of interoperable overlays with underlay optimization.

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 July 2, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
2. Terminology	4
2.1. Requirements Language	5
3. SRv6 SID	5
3.1. SID Format	6
3.2. SID Allocation within an SR domain	7
3.3. SID Reachability	9
4. SR Endpoint Behaviors	10
4.1. End: Endpoint	12
4.1.1. Upper-Layer Header	12
4.2. End.X: Layer-3 Cross-Connect	13
4.3. End.T: Specific IPv6 Table Lookup	14
4.4. End.DX6: Decapsulation and IPv6 Cross-Connect	14
4.5. End.DX4: Decapsulation and IPv4 Cross-Connect	15
4.6. End.DT6: Decapsulation and Specific IPv6 Table Lookup	16
4.7. End.DT4: Decapsulation and Specific IPv4 Table Lookup	17
4.8. End.DT46: Decapsulation and Specific IP Table Lookup	18
4.9. End.DX2: Decapsulation and L2 Cross-Connect	19
4.10. End.DX2V: Decapsulation and VLAN L2 Table Lookup	20
4.11. End.DT2U: Decapsulation and Unicast MAC L2 Table Lookup	21
4.12. End.DT2M: Decapsulation and L2 Table Flooding	22
4.13. End.B6.Encaps: Endpoint Bound to an SRv6 Policy w/ Encaps	22
4.14. End.B6.Encaps.Red: End.B6.Encaps with Reduced SRH	24
4.15. End.BM: Endpoint Bound to an SR-MPLS Policy	24
4.16. Flavors	25
4.16.1. PSP: Penultimate Segment Pop of the SRH	25
4.16.2. USP: Ultimate Segment Pop of the SRH	28
4.16.3. USD: Ultimate Segment Decapsulation	28
5. SR Policy Headend Behaviors	29
5.1. H.Encaps: SR Headend with Encapsulation in an SRv6 Policy	30
5.2. H.Encaps.Red: H.Encaps with Reduced Encapsulation	31

5.3.	H.Encaps.L2: H.Encaps Applied to Received L2 Frames . . .	31
5.4.	H.Encaps.L2.Red: H.Encaps.Red Applied to Received L2 frames	31
6.	Counters	32
7.	Flow-based Hash Computation	32
8.	Control Plane	32
8.1.	IGP	33
8.2.	BGP-LS	33
8.3.	BGP IP/VPN/EVPN	33
8.4.	Summary	33
9.	Security Considerations	35
10.	IANA Considerations	35
10.1.	Ethernet Next Header Type	35
10.2.	SRv6 Endpoint Behaviors Registry	36
10.2.1.	Initial Registrations	36
11.	Acknowledgements	38
12.	Contributors	38
13.	References	41
13.1.	Normative References	41
13.2.	Informative References	42
	Authors' Addresses	43

1. Introduction

Segment Routing [RFC8402] 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. Network programming combines 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 specifies the main segment routing behaviors to enable the creation of interoperable overlays with underlay optimization.

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

Familiarity with the Segment Routing Header [RFC8754] is expected.

2. Terminology

The following terms used within this document are defined in [RFC8402]: Segment Routing, SR Domain, Segment ID (SID), SRv6, SRv6 SID, SR Policy, Prefix-SID, and Adj-SID.

The following terms used within this document are defined in [RFC8754]: SRH, SR Source Node, Transit Node, SR Segment Endpoint Node, Reduced SRH, Segments Left and Last Entry.

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

SRv6 SID function: The function part of the SID is an opaque identification of a local behavior bound to the SID. It is formally defined in Section 3.1 of this document.

SRv6 Segment Endpoint behavior: A packet processing behavior executed at an SRv6 Segment Endpoint Node. Section 4 of this document defines SRv6 Segment Endpoint behaviors related to traffic-engineering and

overlay use-cases. Other behaviors (e.g. service programming) are outside the scope of this document.

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.

Per-VRF VPN label: a single label for the entire VRF that is shared by all routes from that VRF ([RFC4364] Section 4.3.2)

Per-CE VPN label: a single label for each attachment circuit that is shared by all routes with the same "outgoing attachment circuit" ([RFC4364] Section 4.3.2)

2.1. 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.

3. SRv6 SID

RFC8402 defines an SRv6 Segment Identifier as an IPv6 address explicitly associated with the segment.

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

Its processing is defined in [RFC8754] 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 packet's 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 nonlocal route
- * No Match

Section 4 of this document defines a new set of SRv6 SID behaviors in addition to that defined in [RFC8754] Section 4.3.1.

3.1. SID Format

This document defines an SRv6 SID as consisting of LOC:FUNCT:ARG, where a locator (LOC) is encoded in the L most significant bits of the SID, followed by F bits of function (FUNCT) and A bits of arguments (ARG). L, the locator length, is flexible, and an operator is free to use the locator length of their choice. F and A may be any value as long as $L+F+A \leq 128$. When $L+F+A$ is less than 128 then the remaining bits of the SID MUST be zero.

A locator may be represented as B:N where B is the SRv6 SID block (IPv6 prefix allocated for SRv6 SIDs by the operator) and N is the identifier of the parent node instantiating the SID.

When the LOC part of the SRv6 SIDs is routable, it leads to the node which instantiates the SID.

The FUNCT is an opaque identification of a local behavior bound to the SID.

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

An SRv6 Segment Endpoint Behavior may require additional information for its processing (e.g. related to the flow or service). This information may be encoded in the ARG bits of the SID.

In such a case, the semantics and format of the ARG bits are defined as part of the SRv6 endpoint behavior specification.

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 Allocation within an SR domain

Locators are assigned consistent with IPv6 infrastructure allocation. For example, a network operator may:

- o Assign block B::/48 to the SR domain
- o Assign a unique B:N::/64 block to each SRv6-enabled node in the domain

As an example, one mobile service provider has commercially deployed SRv6 across more than 1000 commercial routers and 1800 whitebox routers. All these devices are enabled for SRv6 and advertise SRv6 SIDs. The provider historically deployed IPv6 and assigned infrastructure addresses from ULA space [RFC4193]. They specifically allocated three /48 prefixes (Country X, Country Y, Country Z) to support their SRv6 infrastructure. From those /48 prefixes each router was assigned a /64 prefix from which all SIDs of that router are allocated.

In another example, a large mobile and fixed-line service provider has commercially deployed SRv6 in their country-wide network. This provider is assigned a /20 prefix by an RIR (Regional Internet Registry). They sub-allocated a few /48 prefixes to their infrastructure to deploy SRv6. Each router is assigned a /64 prefix from which all SIDs of that router are allocated.

IPv6 address consumption in both these examples is minimal, representing less than one billionth and one millionth of the available address space, respectively.

A service provider receiving the current minimum allocation of a /32 from an RIR may assign a /48 prefix to their infrastructure deploying

SRv6, and subsequently allocate /64 prefixes for SIDs at each SRv6 node. The /48 assignment is one sixty-five thousandth ($1/2^{16}$) of the usable IPv6 address space available for assignment by the provider.

When an operator instantiates a SID at a node, they specify a SID value B:N:FUNCT and the behavior bound to the SID using one of the SRv6 Endpoint Behavior codepoint of the registry defined in this document (see Table 4).

The node advertises the SID, B:N:FUNCT, in the control-plane (see Section 8) together with the SRv6 Endpoint Behavior codepoint identifying the behavior of the SID.

An SR Source Node cannot infer the behavior by examination of the FUNCT value of a SID.

Therefore, the SRv6 Endpoint Behavior codepoint is advertised along with the SID in the control plane.

An SR Source Node uses the SRv6 Endpoint Behavior codepoint to map the received SID (B:N:FUNCT) to a behavior.

An SR Source Node selects a desired behavior at an advertising node by selecting the SID (B:N:FUNCT) advertised with the desired behavior.

As an example, a network operator may:

- o Assign an SRv6 SID block 2001:db8:bbbb::/48 from their in-house operation block for their SRv6 infrastructure
- o Assign an SRv6 Locator 2001:db8:bbbb:3::/64 to one particular router, for example Router 3, in their SR Domain
- o At Router 3, within the locator 2001:db8:bbbb:3::/64, the network operator or the router performs dynamic assignment for:
 - * Function 0x0100 associated with the behavior End.X (Endpoint with cross-connect) between router 3 and its connected neighbor router, for example Router 4. This function is encoded as 16-bit value and has no arguments (F=16, A=0). This SID is advertised in the control plane as 2001:db8:bbbb:3:100:: with SRv6 Endpoint Behavior codepoint value of 5.
 - * Function 0x0101 associated with the behavior End.X (Endpoint with cross-connect) between router 3 and its connected neighbor

router, for example Router 2. This function is encoded as 16-bit value and has no arguments (F=16, A=0). This SID is advertised in the control plane as 2001:db8:bbbb:3:101:: with SRv6 Endpoint Behavior codepoint value of 5.

These examples do not preclude any other IPv6 addressing allocation scheme.

3.3. 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 specific to the routing protocol and are outside of 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 explicitly 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 through 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. A packet could also be steered to a node instantiating a non-routed SID by preceding it in the SID-list with an Adjacency SID to that node. Routed and non-routed SRv6 SIDs are the SRv6 instantiation of global and local segments, respectively [RFC8402].

4. SR Endpoint Behaviors

Following is a set of well-known behaviors that can be associated with a SID.

End	Endpoint function
End.X	The SRv6 instantiation of a Prefix SID [RFC8402]
End.T	Endpoint with Layer-3 cross-connect
End.DX6	The SRv6 instantiation of an Adj SID [RFC8402]
End.DX4	Endpoint with specific IPv6 table lookup
End.DT6	Endpoint with decapsulation and IPv6 cross-connect
End.DT4	e.g. IPv6-L3VPN (equivalent to per-CE VPN label)
End.DT46	Endpoint with decaps and IPv4 cross-connect
End.DX2	e.g. IPv4-L3VPN (equivalent to per-CE VPN label)
End.DX2V	Endpoint with decapsulation and IPv6 table lookup
End.DT2U	e.g. IPv6-L3VPN (equivalent to per-VRF VPN label)
End.DT2M	Endpoint with decapsulation and IPv4 table lookup
End.B6.Encaps	e.g. IPv4-L3VPN (equivalent to per-VRF VPN label)
End.B6.Encaps.Red	Endpoint with decapsulation and IP table lookup
End.BM	e.g. IP-L3VPN (equivalent to per-VRF VPN label)
	Endpoint with decapsulation and L2 cross-connect
	e.g. L2VPN use-case
	Endpoint with decaps and VLAN L2 table lookup
	e.g. EVPN Flexible cross-connect use-case
	Endpoint with decaps and unicast MAC L2 table lookup
	e.g. EVPN Bridging unicast use-case
	Endpoint with decapsulation and L2 table flooding
	e.g. EVPN Bridging BUM use-case with ESI filtering
	Endpoint bound to an SRv6 policy with encapsulation
	SRv6 instantiation of a Binding SID
	End.B6.Encaps with reduced SRH
	SRv6 instantiation of a Binding SID
	Endpoint bound to an SR-MPLS Policy
	SRv6 instantiation of an SR-MPLS Binding SID

The list is not exhaustive. In practice, any behavior 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, provided there is a behavior codepoint allocated for the processing.

When an SRv6-capable node (N) receives an IPv6 packet whose destination address matches a FIB entry that represents a locally instantiated SRv6 SID (S), the IPv6 header chain is processed as defined in Section 4 of [RFC8200]. For SRv6 SIDs associated with an Endpoint Behavior defined in this document, the SRH and Upper-layer Header are processed as defined in the following subsections.

The pseudocode describing these behaviors details local processing at a node. An implementation of the pseudocode is compliant as long as the externally observable wire protocol is as described by the pseudocode.

Section 4.16 defines flavors of some of these behaviors.

Section 10.2 of this document defines the IANA Registry used to maintain all these behaviors as well as future ones defined in other documents.

4.1. End: Endpoint

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

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.     Stop processing the SRH, and proceed to process the next
        header in the packet, whose type is identified by
        the Next Header field in the routing header.
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 IPv6 Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Update IPv6 DA with Segment List[Segments Left]
S15.   Submit 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. identified by VRF or 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.

4.1.1. Upper-Layer Header

When processing the Upper-layer Header of a packet matching a FIB entry locally instantiated as an End SID, N does:

```
S01. If (Upper-Layer Header type is allowed by local configuration) {
S02.   Proceed to process the Upper-layer Header
S03. } Else {
S04.   Send an ICMP Parameter Problem to the Source Address,
       Code 4 (SR Upper-layer Header Error),
       Pointer set to the offset of the Upper-layer Header,
       Interrupt packet processing and discard the packet.
S05 }
```

Allowing processing of specific Upper-Layer Headers types is useful for OAM. As an example, an operator might permit pinging of SIDs. To do this they may enable local configuration to allow Upper-layer Header type 58 (ICMPv6).

It is RECOMMENDED that an implementation of local configuration only allows Upper-layer Header processing of types that do not result in the packet being forwarded (e.g. ICMPv6).

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 its main use is for traffic-engineering policies.

Any SID instance of this behavior is associated with a set, 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.   Submit the packet to the IPv6 module for transmission
       to the new destination via a member of J
```

Notes:

S15. If the set J contains several L3 adjacencies, then one element of the set is selected based on a hash of the packet's header (see Section 7).

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 might allocate up to 11 End.X local SIDs: one for the bundle itself and then up to one for each Layer-2 member link. The flows steered using the End.X SID corresponding to the bundle itself get load balanced across the member links via hashing while the flows steered using the End.X SID corresponding to a member link get steered over that specific member link alone.

When the End.X behavior is associated with a BGP Next-Hop, it is the SRv6 instantiation of the BGP Peering Segments [RFC8402].

When processing the Upper-layer Header of a packet matching a FIB entry locally instantiated as an End.X SID, process the packet as per Section 4.1.1.

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 is 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. Submit 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 End.T SID, process the packet as per Section 4.1.1.

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

Provider Edge (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 is 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:

```
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 End.DX6 SID, N does:

```
S01. If (Upper-Layer Header type == 41(IPv6) ) {
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.   Process as per Section 4.1.1
S06. }
```

Notes:

S01. 41 refers to IPv6 encapsulation as defined by IANA allocation for Internet Protocol Numbers.

S03. 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 (see Section 7).

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 is 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:

```
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 End.DX4 SID, N does:

```
S01. If (Upper-Layer Header type == 4(IPv4) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Forward the exposed IPv4 packet to the L3 adjacency J
S04. } Else {
S05.   Process as per Section 4.1.1
S06. }
```

Notes:

S01. 4 refers to IPv4 encapsulation as defined by IANA allocation for Internet Protocol Numbers

S03. 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 (see Section 7).

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 is 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 an 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 is 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:

```
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 End.DT6 SID, N does:

```
S01. If (Upper-Layer Header type == 41(IPv6) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Set the packet's associated FIB table to T
S04.   Submit the packet to the egress IPv6 FIB lookup and
           transmission to the new destination
S05. } Else {
S06.   Process as per Section 4.1.1
S07. }
```

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.T 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 is 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 is 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:

```
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 End.DT4 SID, N does:

```
S01. If (Upper-Layer Header type == 4(IPv4) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Set the packet's associated FIB table to T
S04.   Submit the packet to the egress IPv4 FIB lookup and
        transmission to the new destination
S05. } Else {
S06.   Process as per Section 4.1.1
S07. }
```

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 is 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 is 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:

```
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 End.DT46 SID, N does:

```
S01. If (Upper-layer Header type == 4(IPv4) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Set the packet's associated FIB table to T4
S04.   Submit the packet to the egress IPv4 FIB lookup and
        transmission to the new destination
S05. } Else if (Upper-layer Header type == 41(IPv6) ) {
S06.   Remove the outer IPv6 Header with all its extension headers
S07.   Set the packet's associated FIB table to T6
S08.   Submit the packet to the egress IPv6 FIB lookup and
        transmission to the new destination
S09. } Else {
S10.   Process as per Section 4.1.1
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 [RFC4664] / EVPN VPWS [RFC7432] [RFC8214] use-case.

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

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 End.DX2 SID, N does:

```
S01. If (Upper-Layer Header type == 143(Ethernet) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Forward the Ethernet frame to the OIF I
S04. } Else {
S05.   Process as per Section 4.1.1
S06. }
```

Notes:

S01. IANA has allocated the Internet Protocol number 143 to Ethernet [IEEE.802.3_2018] (see Section 10.1).
S03. An End.DX2 behavior could be customized to expect a specific IEEE header (e.g. VLAN tag) and rewrite the egress IEEE header before forwarding on the outgoing interface.

Note that an End.DX2 SID may also be associated with a bundle of outgoing interfaces.

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 this behavior is 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:

S03. Lookup the exposed VLANs in L2 table T, and forward via the matched table entry.

Notes:

S03. 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 [RFC7432]. Any SID instance of the End.DT2U behavior is 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 == 143(Ethernet) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Learn the exposed MAC Source Address in L2 Table T
S04.   Lookup the exposed MAC Destination Address in L2 Table T
S05.   If (matched entry in T) {
S06.     Forward via the matched table T entry
S07.   } Else {
S08.     Forward via all L2 OIFs entries in table T
S09.   }
S10. } Else {
S11.   Process as per Section 4.1.1
S12. }
```

Notes:

S01. IANA has allocated the Internet Protocol number 143 to Ethernet (see Section 10.1).

S03. In EVPN [RFC7432], the learning of the exposed MAC Source Address is done via control plane. In L2VPN VPLS [RFC4761] [RFC4762] reachability is obtained by standard learning bridge functions in the data 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.

Two of the applications of the End.DT2M behavior are the EVPN Bridging of broadcast, unknown and multicast (BUM) traffic with Ethernet Segment Identifier (ESI) filtering [RFC7432] and the EVPN ETREE [RFC8317] use-cases.

Any SID instance of this behavior is associated with a L2 table T. The behavior also takes an argument: "Arg.FE2". This argument provides a local mapping to ESI for split-horizon filtering of the received traffic to exclude specific OIF (or set of OIFs) from L2 table T flooding. The allocation of the argument values is local to the SR Endpoint Node instantiating this behavior and the signaling of the argument to other nodes for the EVPN functionality occurs via control plane.

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.DX2 behavior except for the Upper-layer header processing which is as follows:

```
S01. If (Upper-Layer Header type == 143(Ethernet) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Learn the exposed MAC Source Address in L2 Table T
S04.   Forward via all L2OIFs excluding those associated by the
       identifier Arg.FE2
S05. } Else {
S06.   Process as per Section 4.1.1
S07. }
```

Notes:

S01. IANA has allocated the Internet Protocol number 143 to Ethernet (see Section 10.1).

S03. In EVPN [RFC7432], the learning of the exposed MAC Source Address is done via control plane. In L2VPN VPLS [RFC4761] [RFC4762] reachability is obtained by standard learning bridge functions in the data plane.

4.13. 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 one of the SRv6 instantiations of a Binding SID [RFC8402].

Any SID instance of this behavior is associated with an SR Policy B and a source address A.

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

```
S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.     Stop processing the SRH, and proceed to process the next
        header in the packet, whose type is identified by
        the Next Header field in the routing header.
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 IPv6 Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Update IPv6 DA with Segment List[Segments Left]
S15.   Push a new IPv6 header with its own SRH containing B
S16.   Set the outer IPv6 SA to A
S17.   Set the outer IPv6 DA to the first SID of B
S18.   Set the outer Payload Length, Traffic Class, Flow Label,
        Hop Limit and Next-Header fields
S19.   Submit the packet to the egress IPv6 FIB lookup and
        transmission to the new destination
S20. }
```

Notes:

S15. 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.

S18. The Payload Length, Traffic Class, Hop Limit 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 End.B6.Encaps SID, process the packet as per Section 4.1.1.

4.14. End.B6.Encaps.Red: End.B6.Encaps 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 excluding the first SID in the SRH of the new IPv6 header. Thus, the first segment is only placed in the IPv6 Destination Address of the new IPv6 header and the packet is forwarded according to it.

The SRH Last Entry field is set as defined in Section 4.1.1 of [RFC8754].

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

4.15. 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].

Any SID instance of this behavior is associated with an SR-MPLS Policy B.

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

```
S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.     Stop processing the SRH, and proceed to process the next
           header in the packet, whose type is identified by
           the Next Header field in the routing header.
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 IPv6 Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Update IPv6 DA with Segment List[Segments Left]
S15.   Push the MPLS label stack for B
S16.   Submit the packet to the MPLS engine for transmission
S17. }
```

When processing the Upper-layer header of a packet matching a FIB entry locally instantiated as an End.BM SID, process the packet as per Section 4.1.1.

4.16. Flavors

The Penultimate Segment Pop of the SRH (PSP), Ultimate Segment Pop of the SRH (USP) and Ultimate Segment Decapsulation (USD) flavors are variants of the End, End.X and End.T behaviors. The End, End.X and End.T behaviors can support these flavors either individually or in combinations.

4.16.1. PSP: Penultimate Segment Pop of the SRH

4.16.1.1. Guidelines

SR Segment Endpoint Nodes advertise the SIDs instantiated on them via control plane protocols as described in Section 8. Different behavior ids are allocated for flavored and unflavored SIDs (see Table 4).

An SR Segment Endpoint Node that offers both PSP and non-PSP flavored behavior advertises them as two different SIDs.

The SR Segment Endpoint Node only advertises the PSP flavor if the operator enables this capability at the node.

The PSP operation is deterministically controlled by the SR Source Node.

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.

4.16.1.2. Definition

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 only takes place at a penultimate SR Segment Endpoint Node and does not happen at any Transit Node. When a SID of PSP-flavor is processed at a non-penultimate SR Segment Endpoint Node, the PSP behavior is not performed as described in the pseudocode below since Segments Left would not be zero.

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 (Segments Left == 0) {
S14.2.      Update the Next Header field in the preceding header to the
           Next Header value from the SRH
S14.3.      Decrease the IPv6 header Payload Length by 8*(Hdr Ext Len+1)
S14.4.      Remove the SRH from the IPv6 extension header chain
S14.5.  }
```

The usage of PSP does not increase the MTU of the IPv6 packet and hence does not have any impact on the PMTU discovery mechanism.

As a reminder, [RFC8754] defines in section 5 the SR Deployment Model within the SR Domain [RFC8402]. Within this framework, the Authentication Header (AH) is not used to secure the SRH as described

in Section 7.5 of [RFC8754]. Hence, the discussion of applicability of PSP along with AH usage is beyond the scope of this document.

In the context of this specification, the End, End.X and End.T behaviors with PSP do not contravene Section 4 of [RFC8200] because the destination address of the incoming packet is the address of the node executing the behavior.

4.16.1.3. Use-case

One use-case for the PSP functionality is streamlining the operation of an egress border router.

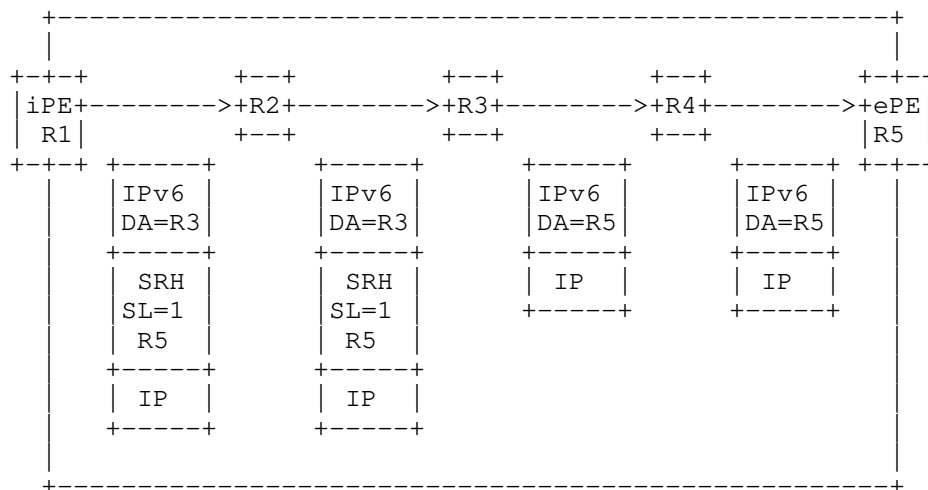


Figure 1: PSP use-case topology

In the above illustration, for a packet sent from iPE to ePE, node R3 is an intermediate traffic engineering waypoint and is the penultimate segment endpoint router; the node that copies the last segment from the SRH into the IPv6 Destination Address and decrements segments left to 0. The SDN controller knows that no other node after R3 needs to inspect the SRH, and it instructs R3 to remove the exhausted SRH from the packet by using a PSP-flavored SID.

The benefits for the egress PE are straightforward:

- as part of the decapsulation process the egress PE is required to parse and remove fewer bytes from the packet.
- if a lookup on an upper-layer IP header is required (e.g. per-VRF VPN), the header is more likely to be within the memory accessible

to the lookup engine in the forwarding ASIC (Application-specific integrated circuit).

4.16.2. 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) {
S03.1.      Update the Next Header field in the preceding header to the
              Next Header value of the SRH
S03.2.      Decrease the IPv6 header Payload Length by 8*(Hdr Ext Len+1)
S03.3.      Remove the SRH from the IPv6 extension header chain
S03.4.      Proceed to process the next header in the packet
S04.    }
```

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.

4.16.3. USD: Ultimate Segment Decapsulation

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(IPv6) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Submit the packet to the egress IPv6 FIB lookup and
        transmission to the new destination
S04. } Else if (Upper-layer Header type == 4(IPv4) ) {
S05.   Remove the outer IPv6 Header with all its extension headers
S06.   Submit the packet to the egress IPv4 FIB lookup and
        transmission to the new destination
S07. Else {
S08.   Process as per Section 4.1.1
S09. }
```

```

End.T:
S01. If (Upper-layer Header type == 41(IPv6) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Set the packet's associated FIB table to T
S04.   Submit the packet to the egress IPv6 FIB lookup and
       transmission to the new destination
S05. } Else if (Upper-layer Header type == 4(IPv4) ) {
S06.   Remove the outer IPv6 Header with all its extension headers
S07.   Set the packet's associated FIB table to T
S08.   Submit the packet to the egress IPv4 FIB lookup and
       transmission to the new destination
S09. Else {
S10.   Process as per Section 4.1.1
S11. }

End.X:
S01. If (Upper-layer Header type == 41(IPv6) ||
       Upper-layer Header type == 4(IPv4) ) {
S02.   Remove the outer IPv6 Header with all its extension headers
S03.   Forward the exposed IP packet to the L3 adjacency J
S04. } Else {
S05.   Process as per Section 4.1.1
S06. }

```

One of the applications of the USD flavor is the case of TI-LFA in P routers with encapsulation. 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.

5. SR Policy Headend Behaviors

This section describes a set of SR Policy Headend [RFC8402] behaviors.

H.Encaps	SR Headend Behavior with Encapsulation in an SR Policy
H.Encaps.Red	H.Encaps with Reduced Encapsulation
H.Encaps.L2	H.Encaps Applied to Received L2 Frames
H.Encaps.L2.Red	H.Encaps.Red Applied to Received L2 Frames

This list is not exhaustive and future documents may define additional behaviors.

5.1. H.Encaps: SR Headend 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.

Node N is configured with an IPv6 Address T (e.g. assigned to its loopback).

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

The H.Encaps encapsulation behavior is defined as follows:

- S01. Push an IPv6 header with its own SRH
- S02. Set outer IPv6 SA = T and outer IPv6 DA to the first SID in the segment list
- S03. Set outer Payload Length, Traffic Class, Hop Limit and Flow Label fields
- S04. Set the outer Next-Header value
- S05. Decrement inner IPv6 Hop Limit or IPv4 TTL
- S06. Submit the packet to the IPv6 module for transmission to S1

Note:

S03: As described in [RFC2473] and [RFC6437].

After the H.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 received packet is encapsulated unmodified (with the exception of the IPv4 TTL or IPv6 Hop Limit that is decremented as described in [RFC2473]).

The H.Encaps behavior is valid for any kind of Layer-3 traffic. This behavior is commonly used for L3VPN with IPv4 and IPv6 deployments. It may be also used for TI-LFA [I-D.ietf-rtgwg-segment-routing-ti-lfa] at the point of local repair.

The push of 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.

5.2. H.Encaps.Red: H.Encaps with Reduced Encapsulation

The H.Encaps.Red behavior is an optimization of the H.Encaps behavior.

H.Encaps.Red reduces the length of the SRH by excluding the first SID in the SRH of the pushed IPv6 header. The first SID is only placed in the Destination Address field of the pushed IPv6 header.

After the H.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 push of 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.

5.3. H.Encaps.L2: H.Encaps Applied to Received L2 Frames

The H.Encaps.L2 behavior encapsulates a received Ethernet [IEEE.802.3_2018] frame and its attached VLAN header, if present, in an IPv6 packet with an SRH. The Ethernet frame becomes the payload of the new IPv6 packet.

The Next Header field of the SRH MUST be set to 143.

The push of 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.

The encapsulating node MUST remove the preamble (if any) and frame check sequence (FCS) from the Ethernet frame upon encapsulation and the decapsulating node MUST regenerate, as required, the preamble and FCS before forwarding Ethernet frame.

5.4. H.Encaps.L2.Red: H.Encaps.Red Applied to Received L2 frames

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

H.Encaps.L2.Red reduces the length of the SRH by excluding the first SID in the SRH of the pushed IPv6 header. The first SID is only placed in the Destination Address field of the pushed IPv6 header.

The push of 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. Counters

A node supporting this document SHOULD implement a pair of traffic counters (one for packets and one for bytes) per local SID entry, for traffic that matched that SID and was processed successfully (i.e. packets which generate ICMP Error Messages or are dropped are not counted). The retrieval of these counters from MIB, NETCONF/YANG or any other data structure is outside the scope of this document.

7. Flow-based Hash Computation

When a flow-based selection within a set needs to be performed, the IPv6 Source Address, the IPv6 Destination Address and the IPv6 Flow Label of the outer IPv6 header MUST be included in the flow-based hash.

This occurs when 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.

Additionally, any transit router in an SRv6 domain includes the outer flow label in its ECMP flow-based hash [RFC6437].

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.

While not necessary for an SDN control plane, the remainder of this section provides a high-level illustrative overview of how control-plane protocols may be involved with SRv6. Their specification is outside the scope of this document.

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. The IGP should also advertise the maximum SRv6 SID depth (MSD) capability of the node for each type of SRv6 operation - in particular, the SR source (e.g. H.Encaps), intermediate endpoint (e.g. End, End.X) and final endpoint (e.g. End.DX4, End.DT6) behaviors. These capabilities are factored in by an SR Source Node (or a controller) during the SR Policy computation.

The presence of SIDs in the IGP does not imply any routing semantics to the addresses represented by these SIDs. The routing reachability to an IPv6 address is solely governed by the non-SID-related IGP prefix reachability information that includes locators. Routing is neither governed nor influenced in any way by a SID advertisement in the IGP.

These SIDs provide important topological behaviors for the IGP to build FRR solutions based on TI-LFA [I-D.ietf-rtgwg-segment-routing-ti-lfa] and for TE processes relying on IGP topology database to build SR policies.

8.2. BGP-LS

BGP-LS provides the functionality for topology discovery that includes the SRv6 capabilities of the nodes, their locators and locally instantiated SIDs. This enables controllers or applications to build an inter-domain topology that can be used for computation of SR Policies using the SRv6 SIDs.

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 can be signaled in BGP.

In some scenarios an egress PE advertising a VPN route might wish to abstract the specific behavior bound to the SID from the ingress PE and other routers in the network. In such case, the SID may be advertised using the Opaque SRv6 Endpoint Behavior codepoint defined in Table 4. The details of such control plane signaling mechanisms are out of the scope of this document.

8.4. Summary

The following table summarizes behaviors for SIDs that can be signaled in which each respective control plane 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.Encaps		X	
End.B6.Encaps.Red		X	
End.B6.BM		X	

Table 1: SRv6 locally instantiated SIDs signaling

The following table summarizes which SR Policy Headend capabilities are signaled in which signaling protocol.

	IGP	BGP-LS	BGP IP/VPN/EVPN
H.Encaps	X	X	
H.Encaps.Red	X	X	
H.Encaps.L2		X	
H.Encaps.L2.Red		X	

Table 2: SRv6 Policy Headend behaviors signaling

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

For example, a BGP-LS advertisement of H.Encaps behavior would describe the capability of node N to perform a H.Encaps behavior. Specifically, it would describe how many SIDs could be pushed by N without significant performance degradation.

As a reminder, an SR policy is always assigned a Binding SID [RFC8402]. BSIDs are also advertised in BGP-LS as shown in Table 1.

Hence, the Table 2 only focuses on the generic capabilities related to H.Encaps.

9. Security Considerations

The security considerations for Segment Routing are discussed in [RFC8402]. Section 5 of [RFC8754] describes the SR Deployment Model and the requirements for securing the SR Domain. The security considerations of [RFC8754] also cover topics such as attack vectors and their mitigation mechanisms that also apply the behaviors introduced in this document. Together, they describe the required security mechanisms that allow establishment of an SR domain of trust. Having such a well-defined trust boundary is necessary in order to operate SRv6-based services for internal traffic while preventing any external traffic from accessing or exploiting the SRv6-based services. Care and rigor in IPv6 address allocation for use for SRv6 SID allocations and network infrastructure addresses, as distinct from IPv6 addresses allocated for end-users/systems (as illustrated in Section 5.1 of [RFC8754]), can provide the clear distinction between internal and external address space that is required to maintain the integrity and security of the SRv6 Domain. Additionally, [RFC8754] defines an HMAC TLV permitting SR Endpoint Nodes in the SR domain to verify that the SRH applied to a packet was selected by an authorized party and to ensure that the segment list is not modified after generation, regardless of the number of segments in the segment list. When enabled by local configuration, HMAC processing occurs at the beginning of SRH processing as defined in [RFC8754] Section 2.1.2.1 .

This document introduces SRv6 Endpoint and SR Policy Headend behaviors for implementation on SRv6 capable nodes in the network. The headend policy definition should be consistent with the specific behavior used and any local configuration (as specified in Section 4.1.1). As such, this document does not introduce any new security considerations.

The SID Behaviors specified in this document have the same HMAC TLV handling and mutability properties of the Flags, Tag, and Segment List field as the SID Behavior specified in [RFC8754].

10. IANA Considerations

10.1. Ethernet Next Header Type

This document requests IANA to allocate, in the "Protocol Numbers" registry (<https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>), a new value for "Ethernet" with the following definition: The value 143 in the Next Header field of an IPv6 header

or any extension header indicates that the payload is an Ethernet frame [IEEE.802.3_2018].

IANA has done a temporary allocation of Protocol Number 143.

10.2. SRv6 Endpoint Behaviors Registry

This document requests IANA to create a new top-level registry called "Segment Routing Parameters". This registry is being defined to serve as a top-level registry for keeping all other Segment Routing sub-registries.

Additionally, a new sub-registry "SRv6 Endpoint Behaviors" is to be created under top-level "Segment Routing Parameters" registry. This sub-registry maintains 16-bit identifiers for the SRv6 Endpoint behaviors. This registry is established to provide consistency for control plane protocols which need to refer to these behaviors. These values are not encoded in the function bits within a SID.

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	Not to be allocated
1-32767	0x0001-0x7FFF	First Come First Served [RFC8126]	
32768-34815	0x8000-0x87FF	Private Use [RFC8126]	
34816-65534	0x8800-0xFFFFE	Reserved	Opaque
65535	0xFFFF	Reserved	

Table 3: SRv6 Endpoint Behaviors Registry

10.2.1. Initial Registrations

The initial registrations for the sub-registry are as follows:

Value	Hex	Endpoint behavior	Reference
0	0x0000	Reserved	Not to be allocated [This.ID]
1	0x0001	End	

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	[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	[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]
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]
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]
40-32766		Unassigned	
32767	0x7FFF	The SID defined in RFC8754	[This.ID] [RFC8754]
32768-65534		Reserved	
65535	0xFFFF	Opaque	[This.ID]

Table 4: IETF - SRv6 Endpoint Behaviors

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, Saleem Hafeez and Brian Carpenter.

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
Google
United States of America

Email: helmalky@google.com

Prem Jonnalagadda
Barefoot Networks
United States of America

Email: prem@barefootnetworks.com

Milad Sharif
SambaNova Systems
United States of America

Email: milad.sharif@sambanova.ai

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
Arrcus, Inc
United States of America

Email: arthi@arrcus.com

Satish Mynam
Arrcus, Inc
United States of America

Email: satishm@arrcus.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 Brissette
Cisco Systems, Inc.
Canada

Email: pbrisset@cisco.com

Zafar Ali
Cisco Systems, Inc.
United States of America

Email: zali@cisco.com

Ketan Talaulikar
Cisco Systems, Inc.
India

Email: ketant@cisco.com

13. References

13.1. Normative References

- [IEEE.802.3_2018]
IEEE, "802.3-2018", IEEE 802.3-2018,
DOI 10.1109/IEEESTD.2018.8457469, August 2018,
<<https://ieeexplore.ieee.org/document/8457469>>.
- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J.,
Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header
(SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020,
<<https://www.rfc-editor.org/info/rfc8754>>.

13.2. Informative References

- [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-03 (work in progress), September 2020.
- [I-D.ietf-rtgwg-segment-routing-ti-lfa]
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., and D. Voyer, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-05 (work in progress), November 2020.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <<https://www.rfc-editor.org/info/rfc4193>>.
- [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>>.
- [RFC4664] Andersson, L., Ed. and E. Rosen, Ed., "Framework for Layer 2 Virtual Private Networks (L2VPNs)", RFC 4664, DOI 10.17487/RFC4664, September 2006, <<https://www.rfc-editor.org/info/rfc4664>>.
- [RFC4761] Kompella, K., Ed. and Y. Rekhter, Ed., "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling", RFC 4761, DOI 10.17487/RFC4761, January 2007, <<https://www.rfc-editor.org/info/rfc4761>>.
- [RFC4762] Lasserre, M., Ed. and V. Kompella, Ed., "Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling", RFC 4762, DOI 10.17487/RFC4762, January 2007, <<https://www.rfc-editor.org/info/rfc4762>>.
- [RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <<https://www.rfc-editor.org/info/rfc7432>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.

- [RFC8214] Boutros, S., Sajassi, A., Salam, S., Drake, J., and J. Rabadan, "Virtual Private Wire Service Support in Ethernet VPN", RFC 8214, DOI 10.17487/RFC8214, August 2017, <<https://www.rfc-editor.org/info/rfc8214>>.
- [RFC8317] Sajassi, A., Ed., Salam, S., Drake, J., Uttaro, J., Boutros, S., and J. Rabadan, "Ethernet-Tree (E-Tree) Support in Ethernet VPN (EVPN) and Provider Backbone Bridging EVPN (PBB-EVPN)", RFC 8317, DOI 10.17487/RFC8317, January 2018, <<https://www.rfc-editor.org/info/rfc8317>>.

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

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 8, 2020

K. Kompella
R. Bonica
Juniper Networks
July 7, 2019

Using DHCP to Manage Node and Ring SID Assignment
draft-kompella-spring-dhcp-00

Abstract

Node and ring segment identifiers (SIDs) assignments in a particular domain (such as an IGP area) must follow certain rules: they must be allocated from a configured set of SID blocks; they must be unique; and the values should be sticky, i.e., the same value(s) should be assigned to a node should its assignment expire (as might happen if the node resets). This memo suggests the use of the Dynamic Host Configuration Protocol to handle such assignments.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

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 January 8, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Operational Requirements	2
3. Theory of Operation	3
4. Security Considerations	4
5. IANA Considerations	4
6. References	4
6.1. Normative References	4
6.2. Informative References	5
Authors' Addresses	5

1. Introduction

Fundamental to SPRING forwarding is the notion of Segment Identifiers (SIDs) [RFC8402]. At a high level, there are two types of SIDs: those that are locally assigned by the advertising node, such as adjacency and binding SIDs; and those that are globally unique within a given SPRING domain, such as node and ring SIDs. Node SIDs are often manually configured on routers today; this is not only tedious, but error-prone as well; the addition of ring SIDs which must be managed per ring makes manual assignment even more fraught ([I-D.kompella-spring-rmr]).

This document describes the use of the Dynamic Host Configuration Protocol (DHCP [RFC2132]) for managing global SID allocation. The description is limited to the use of node and ring SIDs for MPLS ([I-D.ietf-spring-segment-routing-mpls]); other types of SID allocation, such as for SRv6+ ([I-D.bonica-spring-srv6-plus]) will be described in a future version.

2. Operational Requirements

Node SID assignments must satisfy the following properties:

A SID allocation is an index within a block. This block is defined by a base value (SRGB) and a range; the SID value MUST fall within the range.

SID assignments MUST be unique. Duplicate assignments can have serious forwarding consequences, such as loops and packet misdelivery.

and should have the following properties:

Assignments SHOULD have a long lease time.

Assignments SHOULD be "sticky", i.e., a node re-requesting a global SID of the same type that it had previously requested SHOULD be assigned the same SID (if possible).

An expired SID SHOULD NOT be re-assigned to another node until sufficient time has passed. This time SHOULD be configurable on the DHCP server.

3. Theory of Operation

A DHCP server to be used for global SID assignment SHOULD be told the following:

The type of SID (node, anycast, ring, ...)

The block or set of blocks for each type: SRGB and range.

The default lease time and hold time (before re-assigning a SID to a different node).

The DHCP server need know nothing about SID semantics; the only thing it needs to know is that ring SIDs are allocated in pairs, and all other SIDs are allocated singly.

A node taking part in a SPRING network MAY be configured to use DHCP to get node SIDs. This configuration should say whether to use DHCP for its loopback address, for anycast SIDs and/or for ring SIDs.

A node configured to use DHCP to obtain a SID for its loopback and/or any other prefix sends a request to the DHCP server including the following information:

the type of SID

the prefix

A node that participates in an RMR ring and is configured to use DHCP to obtain a pair of ring SIDs sends, once ring identification is complete ([I-D.ietf-mpls-rmr]), a DHCP request including:

the type of SID (ring SID)

the ring ID

The DHCP server replies to such requests by:

1. looking up the type of SID request;
2. checking if it has previously allocated a SID for this node and prefix (or pair of SIDs for this node and ring ID);
3. if so, checking if the same SID (or pair of SIDs) is available; if so, allocating that SID (or pair of SIDs) and returning.
4. Otherwise, allocating a new SID/pair of SIDs, noting this in its database, and returning.

4. Security Considerations

DHCP is a very widely used protocol, and thus ensuring its continuing secure and robust operation is vital. When the requirements of DHCP in this context are better understood, this section will be filled out.

5. IANA Considerations

Should this document be deemed useful, relevant IANA code points would be requested.

6. References

6.1. Normative References

[I-D.ietf-mpls-rmr]

Kompella, K. and L. Contreras, "Resilient MPLS Rings", draft-ietf-mpls-rmr-11 (work in progress), June 2019.

[I-D.ietf-spring-segment-routing-mpls]

Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-22 (work in progress), May 2019.

[I-D.kompella-spring-rmr]

Kompella, K., Deshmukh, A., and R. Torvi, "Resilient MPLS Rings", draft-kompella-spring-rmr-00 (work in progress), October 2018.

- [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>>.
- [RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, DOI 10.17487/RFC2132, March 1997, <<https://www.rfc-editor.org/info/rfc2132>>.
- [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>>.

6.2. Informative References

- [I-D.bonica-spring-srv6-plus]
Bonica, R., Hegde, S., Kamite, Y., Alston, A., Henriques, D., Halpern, J., and J. Linkova, "IPv6 Support for Segment Routing: SRv6+", draft-bonica-spring-srv6-plus-03 (work in progress), July 2019.

Authors' Addresses

Kireeti Kompella
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

Email: kireeti.kompella@gmail.com

Ron Bonica
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
US

Email: rbonica@juniper.net

Routing area
Internet-Draft
Intended status: Standards Track
Expires: January 6, 2020

S. Hegde
K. Arora
S. Ninan
M. Srivastava
Juniper Networks Inc.
July 5, 2019

PMS/Head-end based MPLS Ping and Traceroute in Inter-AS SR Networks
draft-ninan-spring-mpls-inter-as-oam-01

Abstract

Segment Routing (SR) architecture leverages source routing and tunneling paradigms and can be directly applied to the use of a Multiprotocol Label Switching (MPLS) data plane. Segment Routing also provides an easy and efficient way to provide inter connectivity in a large scale network as described in [RFC8604]. [RFC8287] illustrates the problem and defines extensions to perform LSP Ping and Traceroute for Segment Routing IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with an MPLS data plane. It is useful to have the LSP Ping and traceroute procedures when an SR end-to-end path spans across multiple ASes. This document describes mechanisms to facilitate LSP ping and traceroute in inter-AS SR networks in an efficient manner with simple OAM protocol extension which uses dataplane forwarding alone for sending Echo-Reply.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 January 6, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Reverse Path label stack TLV	4
2.1. Reverse Path label stack TLV definition	4
2.2. SRv6 Dataplane	5
3. Detailed Procedures	5
3.1. Sending an Echo-Request	5
3.2. Receiving an Echo-Request	6
3.3. Sending an Echo-Reply	6
4. Detailed Example	6
4.1. Procedures for Segment Routing LSP ping	7
4.2. Procedures for Segment Routing LSP Traceroute	7
5. Security Considerations	8
6. IANA Considerations	8
7. Acknowledgments	8
8. References	8
8.1. Normative References	8
8.2. Informative References	8
Authors' Addresses	10

1. Introduction

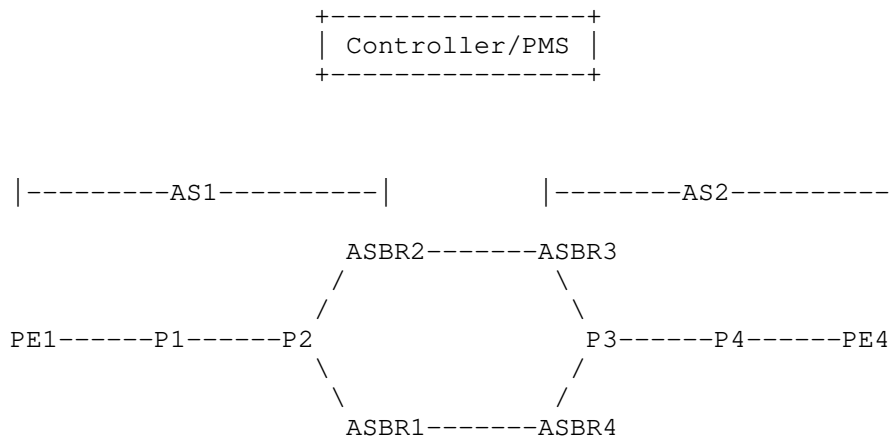


Figure 1: Inter-AS Segment Routing topology

Many network deployments have built their networks consisting of multiple Autonomous Systems either for ease of operations or as a result of network mergers and acquisitions. Segment Routing can be deployed in such scenarios to provide end to end paths, traversing multiple Autonomous systems(AS). These paths consist of Segment Identifiers(SID) of different type as per [RFC8402].

[I-D.ietf-spring-segment-routing-mpls] specifies the forwarding plane behaviour to allow Segment Routing to operate on top of MPLS data plane. [I-D.ietf-spring-segment-routing-central-epe] describes BGP peering SIDs, which will help in steering packet from one Autonomous system to another. Using above SR capabilities, paths which span across multiple Autonomous systems can be created.

For example Figure 1 describes a topology consisting of inter-AS network consisting of ASes AS1 and AS2. Both AS1 and AS2 are Segment Routing enabled and the EPE links have EPE labels configured and advertised via [I-D.ietf-idr-bgpls-segment-routing-epe]. Controller or head-end can build end-to-end Traffic-Engineered path Node-SIDs, Adjacency-SIDs and EPE-SIDs. It is advantageous for operations to be able to perform LSP ping and traceroute procedures on these inter-AS SR paths. LSP ping/traceroute procedures use ip connectivity for Echo-reply to reach the head-end. In inter-AS networks, ip connectivity may not be there from each router in the path. For example in Figure 1 P3 and P4 may not have ip connectivity for PE1.

[RFC8403] describes mechanisms to carry out the MPLS ping/traceroute from a PMS. It is possible to build GRE tunnels or static routes to each router in the network to get IP connectivity for the return

path. This mechanism is operationally very heavy and requires PMS to be capable of building huge number of GRE tunnels, which may not be feasible.

It is not possible to carry out LSP ping and Traceroute functionality on these paths to verify basic connectivity and fault isolation using existing LSP ping and Traceroute mechanism([RFC8287] and [RFC8029]). This is because, there exists no IP connectivity to source address of ping packet, which is in a different AS, from the destination of Ping/Traceroute.

[RFC7743] describes a Echo-relay based solution based on advertising a new Relay Node Address Stack TLV containing stack of Echo-relay ip addresses. This mechanism requires the return ping packet to reach the control plane on every relay node. This document describes a mechanism which is efficient and simple and can be easily deployed in SR networks.

2. Reverse Path label stack TLV

Segment Routing networks statically assign the labels to nodes and PMS/Head-end knows entire database. The return path can be built from PMS/Head-end by stacking labels for the return path. A new TLV "Reverse Path label stack TLV" is defined. Each TLV contains a list of labels which may be a prefix/adjacency/binding SID/EPE SID. MPLS Echo -request should contain this TLV, which defines reverse path to reach source from the destination.

The new Reverse Path label stack TLV is an optional TLV. This TLV is carried in the Echo-Request message. This optional TLV MAY appear in the Echo-request message in any order before or after Target FEC Stack TLV. The Reverse Path label stack TLV is defined as below. Each MPLS Echo-request SHOULD contain this TLV in inter-AS cases, which will enable remote end to send the reply to source.

2.1. Reverse Path label stack TLV definition

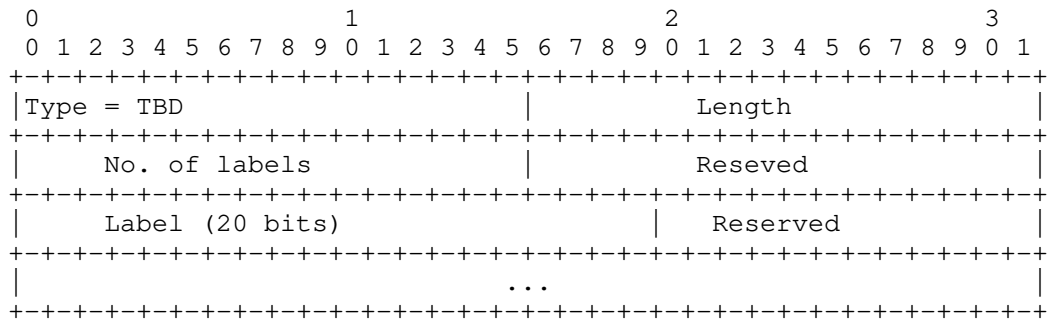


Figure 2: Reverse Path label stack TLV

Type: TBD

Length: Length of TLV including TLV header

No. Of labels:

Ordered set of Labels in the Reverse Path label stack

Label :

Represents 20 bit label. This field should be used to build the return packet. The first label in the label stack represents the top most label that should be encoded in the return packet.

2.2. SRv6 Dataplane

A future version of this document will address the SRv6 Dataplane.

3. Detailed Procedures

3.1. Sending an Echo-Request

LSP ping initiator MUST add a Return Path Label Stack TLV in the Echo-request message. The return label stack MUST correspond to the return path from the egress. The Reverse Path Label Stack TLV is an ordered list of labels. The first label corresponds to the top label that the responder should use to construct the Echo-reply.

3.2. Receiving an Echo-Request

When a receiver does not understand the Reverse Path Label Stack TLV, it SHOULD silently ignore the TLV and proceed with normal processing as described in [RFC8029]. When a Reverse Path Label Stack TLV is received, and the responder supports processing it, it MUST use the labels in Reverse Path Label Stack TLV to build the echo-reply. The responder MUST follow the normal FEC validation procedures as described in [RFC8029] and [RFC8287] and this document does not suggest any change to those procedures. When the Echo-reply has to be sent out the Reverse Path label Stack TLV is used to construct the MPLS packet to send out.

3.3. Sending an Echo-Reply

The Echo-Reply message is sent as MPLS packet with a MPLS label stack. The Echo-Reply message MUST be constructed as described in the [RFC8029]. An MPLS packet is constructed with Echo-reply in the payload. The top label MUST be the first label from the Reverse Path Label Stack TLV. The remaining labels MUST follow the order from the Reverse Path Label Stack. The responder MAY check the reachability of the top label in its own LFIB before sending the Echo-Reply.

4. Detailed Example

An example topology is given in Figure 1 . This will be used in below sections to explain LSP Ping and Traceroute procedures. The PMS/Head-end has complete view of topology. PE1, P1, P2, ASBR1 and ASBR2 are in AS1. Similarly ASBR3, ASBR4, P3, P4 and PE4 are in AS2.

AS1 and AS2 have Segment Routing enabled. IGPs like OSPF/ISIS are used to flood SIDs in each Autonomous System. The ASBR1, ASBR2, ASBR3, ASBR4 advertise BGP EPE SIDs for the inter-AS links. Topology of AS1 and AS2 are advertised via BGP-LS to the controller/PMS or Head-end node. The EPE-SIDs are also advertised via BGP-LS as described in [I-D.ietf-idr-bgpls-segment-routing-epe]

The description in the document uses below notations for Segment Identifiers(SIDs).

Node SIDs : N-PE1, N-P1, N-ASBR1 etc.

Adjacency SIDs : Adj-PE1-P1, Adj-P1-P2 etc.

EPE SIDS : EPE-ASBR2-ASBR3, EPE-ASBR1-ASBR4, EPE-ASBR3-ASBR2 etc.

Let us consider a traffic engineered path built from PE1 to PE4 with label stack as below. N-P1, N-ASBR1, EPE-ASBR1-ASBR4, N-PE4 for

following procedures. This stack may be programmed by controller/PMS or Head-end router PE1 may have imported the whole topology information from BGP-LS and computed the inter-AS path.

4.1. Procedures for Segment Routing LSP ping

To perform LSP ping procedure on an SR-Path from PE1 to PE4 consisting of label stack [N-P1,N-ASBR1,EPE-ASBR1-ASBR4, N-PE4], The remote end(PE4) needs IP connectivity to head end(PE1) for the Segment Routing ping to succeed, because Echo-reply needs to travel back to PE1 from PE4. But in typical deployment scenario there will be no ip route from PE4 to PE1 as they belong to different ASes.

PE1 adds Reverse Path from PE4 to PE1 in the MPLS Echo-request using multiple labels in "Reverse Path Label Stack TLV" as defined above. An example return path label stack for PE1 to PE4 for LSP ping i [N-ASBR4, EPE-ASBR4-ASBR1, N-PE1]. An implementation may also build a Reverse Path Label stack consisting of labels to reach its own AS. Once the label stack is popped-off the Echo-reply message will be exposed. The further packet forwarding will be based on ip lookup. An example Reverse Path Label Stack for this case could be [N-ASBR4, EPE-ASBR4-ASBR1].

On receiving MPLS Echo-request PE4 first validates FEC in the Echo-request. PE4 then builds label stack to send the response from PE4 to PE1 by copying the labels from "Return Path Label Stack TLV". PE4 builds the Echo-reply packet with the MPLS label stack constructed and imposes MPLS headers on top of Echo-reply packet and sends out the packet towards PE1. This label stack can successfully steer reply back to Head-end node(PE1).

4.2. Procedures for Segment Routing LSP Traceroute

As described in the procedures for LSP ping, the return label stack may be sent from head-end in which case the LSP Traceroute procedures are similar to LSP ping. The head-end constructs the Reverse Path Label Stack TLV and the egress node uses the Reverse Path Label Stack to construct the Echo-reply packet header. Head-end/PMS is aware of the return path from every node visited in the network and builds the Reverse Path Label Stack for every visited node accordingly.

For Example:

For the same traffic engineered path PE1 to PE4 mentioned in above sections, let us assume there is no return path available from the nodes ASBR2 to PE1. During the Traceroute procedure, when PE1 has to visit ASBR2, it builds Return Path Label Stack TLV and includes label to the border-node which has the route to, PE1. In this example the

Return Path Label Stack TLV will contain [EPE-ASBR2-ASBR1]. Further down the traceroute procedure when P3 or P4 node is being visited, PE1 build the Return Path Label Stack TLV containing [N-ASBR2, EPE-ASBR2-ASBR1]. The Echo-reply will be an MPLS packet with this label stack and will be forwarded to PE1.

5. Security Considerations

TBD

6. IANA Considerations

Multiprotocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters TLVs Registry

Reverse Path label stack TLV : TBD

7. Acknowledgments

8. References

8.1. Normative References

- [I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.
- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.

8.2. Informative References

- [I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-19 (work in progress), May 2019.

- [I-D.ietf-mpls-interas-lspping]
Nadeau, T. and G. Swallow, "Detecting MPLS Data Plane Failures in Inter-AS and inter-provider Scenarios", draft-ietf-mpls-interas-lspping-00 (work in progress), March 2007.
- [I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-22 (work in progress), May 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>>.
- [RFC7743] Luo, J., Ed., Jin, L., Ed., Nadeau, T., Ed., and G. Swallow, Ed., "Relayed Echo Reply Mechanism for Label Switched Path (LSP) Ping", RFC 7743, DOI 10.17487/RFC7743, January 2016, <<https://www.rfc-editor.org/info/rfc7743>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [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>>.
- [RFC8403] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC 8403, DOI 10.17487/RFC8403, July 2018, <<https://www.rfc-editor.org/info/rfc8403>>.
- [RFC8604] Filsfils, C., Ed., Previdi, S., Dawra, G., Ed., Henderickx, W., and D. Cooper, "Interconnecting Millions of Endpoints with Segment Routing", RFC 8604, DOI 10.17487/RFC8604, June 2019, <<https://www.rfc-editor.org/info/rfc8604>>.

Authors' Addresses

Shraddha Hegde
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Kapil Arora
Juniper Networks Inc.

Email: kapilaro@juniper.net

Samson Ninan
Juniper Networks Inc.

Email: samsonn@juniper.net

Mukul Srivastava
Juniper Networks Inc.

Email: msri@juniper.net

Routing area
Internet-Draft
Intended status: Standards Track
Expires: May 6, 2020

S. Hegde
K. Arora
S. Ninan
M. Srivastava
Juniper Networks Inc.
N. Kumar
Cisco Systems, Inc.
November 3, 2019

PMS/Head-end based MPLS Ping and Traceroute in Inter-AS SR Networks
draft-ninan-spring-mpls-inter-as-oam-02

Abstract

Segment Routing (SR) architecture leverages source routing and tunneling paradigms and can be directly applied to the use of a Multiprotocol Label Switching (MPLS) data plane. Segment Routing also provides an easy and efficient way to provide inter connectivity in a large scale network as described in [RFC8604]. [RFC8287] illustrates the problem and defines extensions to perform LSP Ping and Traceroute for Segment Routing IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with an MPLS data plane. It is useful to have the LSP Ping and traceroute procedures when an SR end-to-end path spans across multiple ASes. This document describes mechanisms to facilitate LSP ping and traceroute in inter-AS SR networks in an efficient manner with simple OAM protocol extension which uses dataplane forwarding alone for sending Echo-Reply.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 May 6, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Reverse Path Segment List TLV	4
2.1. Reverse Path Segment List TLV definition	5
2.1.1. Segment sub-TLV	5
2.2. SRv6 Dataplane	9
3. Detailed Procedures	10
3.1. Sending an Echo-Request	10
3.2. Receiving an Echo-Request	10
3.3. Sending an Echo-Reply	10
4. Detailed Example	10
4.1. Procedures for Segment Routing LSP ping	11
4.2. Procedures for Segment Routing LSP Traceroute	12
5. Building Reverse Path Segment List TLV dynamically	12
5.1. The procedures to build the reverse path	12
5.2. Details with example	13
6. Security Considerations	13
7. IANA Considerations	13
8. Contributors	13
9. Acknowledgments	14
10. References	14
10.1. Normative References	14
10.2. Informative References	14
Authors' Addresses	16

1. Introduction

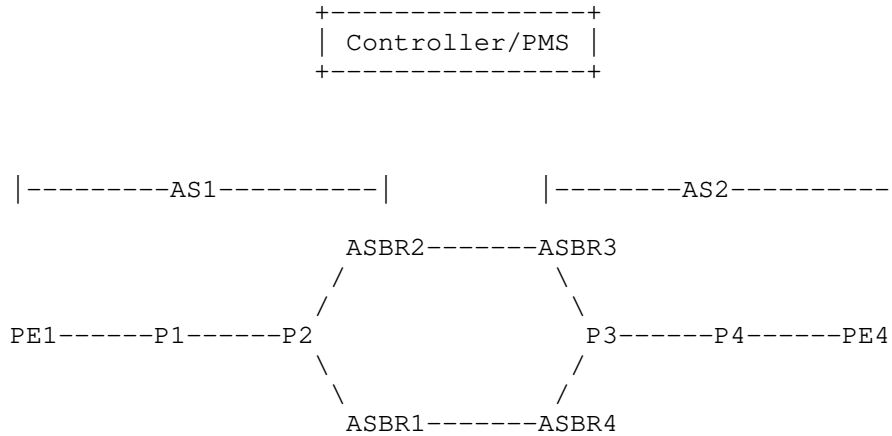


Figure 1: Inter-AS Segment Routing topology

Many network deployments have built their networks consisting of multiple Autonomous Systems either for ease of operations or as a result of network mergers and acquisitions. Segment Routing can be deployed in such scenarios to provide end to end paths, traversing multiple Autonomous systems(AS). These paths consist of Segment Identifiers(SID) of different type as per [RFC8402].

[I-D.ietf-spring-segment-routing-mpls] specifies the forwarding plane behaviour to allow Segment Routing to operate on top of MPLS data plane. [I-D.ietf-spring-segment-routing-central-epe] describes BGP peering SIDs, which will help in steering packet from one Autonomous system to another. Using above SR capabilities, paths which span across multiple Autonomous systems can be created.

For example Figure 1 describes an inter-AS network scenario consisting of ASes AS1 and AS2. Both AS1 and AS2 are Segment Routing enabled and the EPE links have EPE labels configured and advertised via [I-D.ietf-idr-bgpls-segment-routing-epe]. Controller or head-end can build end-to-end Traffic-Engineered path Node-SIDs, Adjacency-SIDs and EPE-SIDs. It is advantageous for operations to be able to perform LSP ping and traceroute procedures on these inter-AS SR paths. LSP ping/traceroute procedures use ip connectivity for Echo-reply to reach the head-end. In inter-AS networks, ip connectivity may not be there from each router in the path. For example in Figure 1 P3 and P4 may not have ip connectivity for PE1.

[RFC8403] describes mechanisms to carry out the MPLS ping/traceroute from a PMS. It is possible to build GRE tunnels or static routes to each router in the network to get IP connectivity for the reverse path. This mechanism is operationally very heavy and requires PMS to be capable of building huge number of GRE tunnels, which may not be feasible.

It is not possible to carry out LSP ping and Traceroute functionality on these paths to verify basic connectivity and fault isolation using existing LSP ping and Traceroute mechanism([RFC8287] and [RFC8029]). This is because, there exists no IP connectivity to source address of ping packet, which is in a different AS, from the destination of Ping/Traceroute.

[RFC7743] describes a Echo-relay based solution based on advertising a new Relay Node Address Stack TLV containing stack of Echo-relay ip addresses. This mechanism requires the return ping packet to reach the control plane on every relay node.

This document describes a mechanism which is efficient and simple and can be easily deployed in SR networks. This mechanism uses a new Reverse Path Segment List TLV to convey the reverse path. The TLV can either be derived by a smart application/controller which has a full topology view or by the help of intermediate nodes.

2. Reverse Path Segment List TLV

Segment Routing networks statically assign the labels to nodes and PMS/Head-end may know the entire database. The reverse path can be built from PMS/Head-end by stacking segments for the reverse path. A new TLV "Reverse Path Segment List TLV" is defined. Each TLV contains a list of segment sub-TLVs which may be a prefix/adjacency/binding SID/EPE SID. MPLS Echo -request should contain this TLV, which defines reverse path to reach source from the destination.

The new Reverse Path Segment List TLV is an optional TLV. This TLV is carried in the Echo-Request message. This optional TLV MAY appear in the Echo-request message in any order before or after Target FEC Stack TLV. The Reverse Path Segment List TLV is defined as below. Each MPLS Echo-request SHOULD contain this TLV in inter-AS cases, which will enable remote end(egress/transit routers) to send the reply to source.

In some cases, the head-end may not have complete visibility. In such cases, it can rely on downstream routers to build the reverse path. For this purpose, the TLV is carried in the Echo-Reply message. Section 5 describes one basic idea in this direction.

2.1. Reverse Path Segment List TLV definition

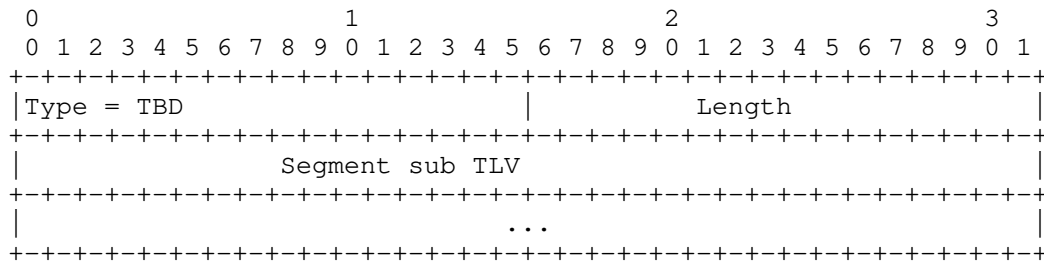


Figure 2: Reverse Path Segment List TLV

Type: TBD

Length: Length of TLV including TLV header and length of sub TLV.

There can be one or more segment sub-TLVs in a Reverse Path Segment List TLV. The applicable segment types are described in Section 2.1.1. The Segment type in a Reverse Path Segment List TLV MAY be same or different.

2.1.1. Segment sub-TLV

[I-D.ietf-spring-segment-routing-policy] defines various types of segments. These segment types are applicable here. One or more segment sub-TLV can be included. The segment sub-TLVs included MAY be of different types.

Below types of segment sub-TLVs are applicable for the Reverse Path Segment List Tlv.

Type 1: SID only, in the form of MPLS Label

Type 3: IPv4 Node Address with optional SID

Type 4: IPv6 Node Address with optional SID for SR MPLS

2.1.1.1. Type 1: SID only, in the form of MPLS Label

The Type-1 Segment Sub-TLV encodes a single SID in the form of an MPLS label. The format is as follows:

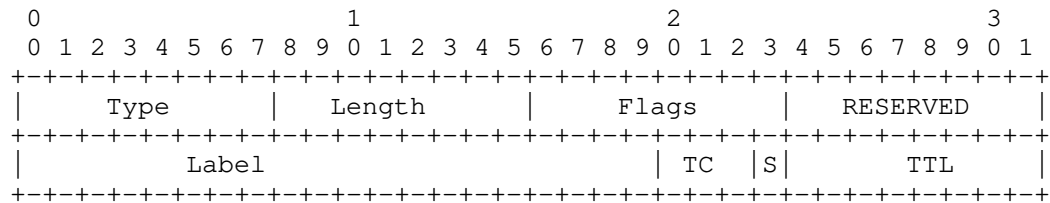


Figure 3: Type 1 Segment sub-TLV

where:

Type: 1 (to be assigned by IANA from the registry "SR Policy List Sub-TLVs" defined in [I-D.ietf-idr-segment-routing-te-policy]).

Length is 6.

Flags: 1 octet of flags as defined in Section Section 2.1.1.4.

RESERVED: 1 octet of reserved bits. SHOULD be unset on transmission and MUST be ignored on receipt.

Label: 20 bits of label value.

TC: 3 bits of traffic class

S: 1 bit of bottom-of-stack.

TTL: 1 octet of TTL.

The following applies to the Type-1 Segment sub-TLV:

The S bit SHOULD be zero upon transmission, and MUST be ignored upon reception.

If the originator wants the receiver to choose the TC value, it sets the TC field to zero.

If the originator wants the receiver to choose the TTL value, it sets the TTL field to 255.

If the originator wants to recommend a value for these fields, it puts those values in the TC and/or TTL fields.

The receiver MAY override the originator's values for these fields. This would be determined by local policy at the receiver. One

possible policy would be to override the fields only if the fields have the default values specified above.

2.1.1.2. Type 3: IPv4 Node Address with optional SID for SR-MPLS

The Type-3 Segment Sub-TLV encodes an IPv4 node address, SR Algorithm and an optional SID in the form of an MPLS label. The format is as follows:

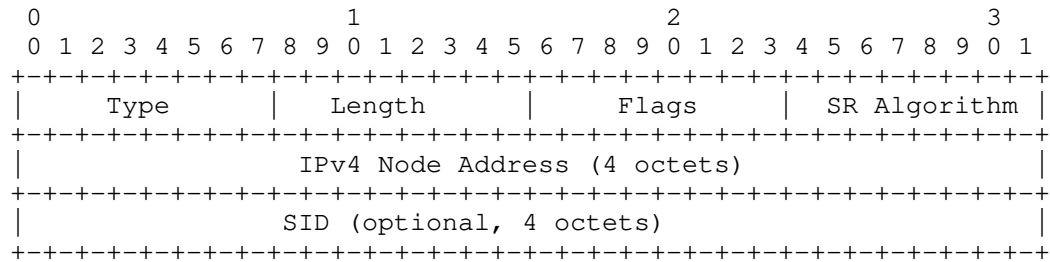


Figure 4: Type 3 Segment sub-TLV

where:

Type: 3 (to be assigned by IANA from the registry "SR Policy List Sub-TLVs" defined in [I-D.ietf-idr-segment-routing-te-policy]).

Length is 6 or 10.

Flags: 1 octet of flags as defined in Section Section 2.1.1.4.

SR Algorithm: 1 octet specifying SR Algorithm as described in section 3.1.1 in [RFC8402], when A-Flag as defined in Section Section 2.1.1.4 is present. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy]. When A-Flag is not encoded, this field SHOULD be unset on transmission and MUST be ignored on receipt.

IPv4 Node Address: a 4 octet IPv4 address representing a node.

SID: 4 octet MPLS label.

The following applies to the Type-3 Segment sub-TLV:

The IPv4 Node Address MUST be present.

The SID is optional and specifies a 4 octet MPLS SID containing label, TC, S and TTL as defined in Section Section 2.1.1.1.

If length is 6, then only the IPv4 Node Address is present.

If length is 10, then the IPv4 Node Address and the MPLS SID are present.

2.1.1.3. Type 4: IPv6 Node Address with optional SID for SR MPLS

The Type-4 Segment Sub-TLV encodes an IPv6 node address, SR Algorithm and an optional SID in the form of an MPLS label. The format is as follows:

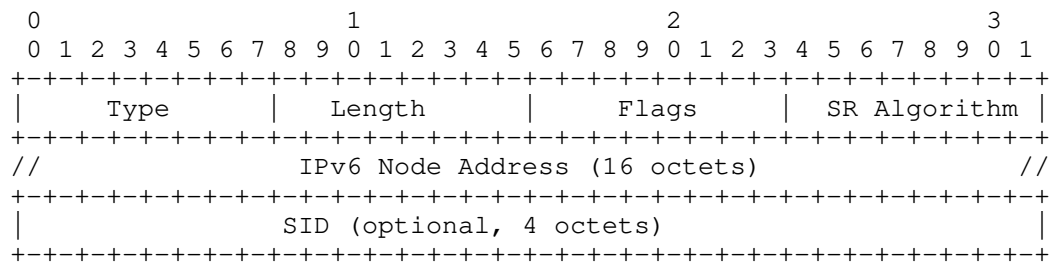


Figure 5: Type 4 Segment sub-TLV

where:

Type: 4 (to be assigned by IANA from the registry "SR Policy List Sub-TLVs" defined in [I-D.ietf-idr-segment-routing-te-policy]).

Length is 18 or 22.

Flags: 1 octet of flags as defined in Section Section 2.1.1.4.

SR Algorithm: 1 octet specifying SR Algorithm as described in section 3.1.1 in [RFC8402], when A-Flag as defined in Section Section 2.1.1.4 is present. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy]. When A-Flag is not encoded, this field SHOULD be unset on transmission and MUST be ignored on receipt.

IPv6 Node Address: a 16 octet IPv6 address representing a node.

SID: 4 octet MPLS label.

The following applies to the Type-4 Segment sub-TLV:

The IPv6 Node Address MUST be present.

The SID is optional and specifies a 4 octet MPLS SID containing label, TC, S and TTL as defined in Section 2.1.1.1 .

If length is 18, then only the IPv6 Node Address is present.

If length is 22, then the IPv6 Node Address and the MPLS SID are present.

2.1.1.4. Segment Flags

The Segment Types described above MAY contain following flags in the "Flags" field (codes to be assigned by IANA from the registry "SR Policy Segment Flags" defined in [I-D.ietf-idr-segment-routing-te-policy])

```

  0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|V|A|         |
+---+---+---+---+---+---+

```

Figure 6: Flags

where:

V-Flag: This flag is used by SRPM for the purpose of "SID verification" as described in Section 5.1 in [I-D.ietf-spring-segment-routing-policy].

A-Flag: This flag indicates the presence of SR Algorithm id in the "SR Algorithm" field applicable to various Segment Types. SR Algorithm is used by SRPM as described in section 4 in [I-D.ietf-spring-segment-routing-policy].

Unused bits in the Flag octet SHOULD be set to zero upon transmission and MUST be ignored upon receipt.

The following applies to the Segment Flags:

V-Flag is applicable to all Segment Types.

A-Flag is applicable to Segment Types 3, 4 and 9. If A-Flag appears with any other Segment Type, it MUST be ignored.

2.2. SRv6 Dataplane

SRv6 dataplane is not in the scope of this document and will be addressed in a separate document.

3. Detailed Procedures

3.1. Sending an Echo-Request

In the inter-AS scenario when there is no reverse path connectivity, LSP ping initiator MUST add a Reverse Path Segment List TLV in the Echo-request message. The reverse Segment List MUST correspond to the return path from the egress. The Reverse Path Segment List TLV is an ordered list of Segments. The first Segment corresponds to the top Segment in MPLS header that the responder MUST use while sending the Echo-reply.

3.2. Receiving an Echo-Request

When a receiver does not understand the Reverse Path Segment List TLV, it SHOULD silently ignore the TLV and proceed with normal processing as described in [RFC8029]. When a Reverse Path Segment List TLV is received, and the responder supports processing it, it MUST use the Segments in Reverse Path Segment List TLV to build the echo-reply. The responder MUST follow the normal FEC validation procedures as described in [RFC8029] and [RFC8287] and this document does not suggest any change to those procedures. When the Echo-reply has to be sent out the Reverse Path Segment List TLV is used to construct the MPLS packet to send out.

3.3. Sending an Echo-Reply

The Echo-Reply message is sent as MPLS packet with a MPLS label stack. The Echo-Reply message MUST be constructed as described in the [RFC8029]. An MPLS packet is constructed with Echo-reply in the payload. The top label MUST be constructed from the first Segment from the Reverse Path Segment List TLV. The remaining labels MUST follow the order from the Reverse Path Segment List TLV. The responder MAY check the reachability of the top label in its own LFIB before sending the Echo-Reply.

4. Detailed Example

An example topology is given in Figure 1 . This will be used in below sections to explain LSP Ping and Traceroute procedures. The PMS/Head-end has complete view of topology. PE1, P1, P2, ASBR1 and ASBR2 are in AS1. Similarly ASBR3, ASBR4, P3, P4 and PE4 are in AS2.

AS1 and AS2 have Segment Routing enabled. IGPs like OSPF/ISIS are used to flood SIDs in each Autonomous System. The ASBR1, ASBR2, ASBR3, ASBR4 advertise BGP EPE SIDs for the inter-AS links. Topology of AS1 and AS2 are advertised via BGP-LS to the controller/PMS or

Head-end node. The EPE-SIDs are also advertised via BGP-LS as described in [I-D.ietf-idr-bgpls-segment-routing-epe]

The description in the document uses below notations for Segment Identifiers(SIDs).

Node SIDs : N-PE1, N-P1, N-ASBR1 etc.

Adjacency SIDs : Adj-PE1-P1, Adj-P1-P2 etc.

EPE SIDS : EPE-ASBR2-ASBR3, EPE-ASBR1-ASBR4, EPE-ASBR3-ASBR2 etc.

Let us consider a traffic engineered path built from PE1 to PE4 with Segment List stack as below. N-P1, N-ASBR1, EPE-ASBR1-ASBR4, N-PE4 for following procedures. This stack may be programmed by controller/PMS or Head-end router PE1 may have imported the whole topology information from BGP-LS and computed the inter-AS path.

4.1. Procedures for Segment Routing LSP ping

To perform LSP ping procedure on an SR-Path from PE1 to PE4 consisting of label stacks [N-P1,N-ASBR1,EPE-ASBR1-ASBR4, N-PE4], The remote end(PE4) needs IP connectivity to head end(PE1) for the Segment Routing ping to succeed, because Echo-reply needs to travel back to PE1 from PE4. But in typical deployment scenario there will be no ip route from PE4 to PE1 as they belong to different ASes.

PE1 adds Reverse Path from PE4 to PE1 in the MPLS Echo-request using multiple Segments in "Reverse Path Segment List TLV" as defined above. An example reverse path Segment List for PE1 to PE4 for LSP ping is [N-ASBR4, EPE-ASBR4-ASBR1, N-PE1]. An implementation may also build a Reverse Path Segment List consisting of labels to reach its own AS. Once the label stack is popped-off the Echo-reply message will be exposed. The further packet forwarding will be based on ip lookup. An example Reverse Path Segment List for this case could be [N-ASBR4, EPE-ASBR4-ASBR1].

On receiving MPLS Echo-request PE4 first validates FEC in the Echo-request. PE4 then builds label stack to send the response from PE4 to PE1 by copying the labels from "Reverse Path Segment List TLV". PE4 builds the Echo-reply packet with the MPLS label stack constructed and imposes MPLS headers on top of Echo-reply packet and sends out the packet towards PE1. This Segment List stack can successfully steer reply back to Head-end node(PE1).

4.2. Procedures for Segment Routing LSP Traceroute

As described in the procedures for LSP ping, the reverse Segment List may be sent from head-end in which case the LSP Traceroute procedures are similar to LSP ping. The head-end constructs the Reverse Path Segment List TLV and the egress node uses the Reverse Path Segment List to construct the Echo-reply packet header. Head-end/PMS is aware of the reverse path from every node visited in the network and builds the Reverse Path Segment List for every visited node accordingly.

For Example:

For the same traffic engineered path PE1 to PE4 mentioned in above sections, let us assume there is no reverse path available from the nodes ASBR4 to PE1. During the Traceroute procedure, when PE1 has to visit ASBR4, it builds reverse Path Label Stack TLV and includes label to the border-node which has the route to, PE1. In this example the Reverse Path Segment List TLV will contain [EPE-ASBR4-ASBR1]. Further down the traceroute procedure when P3 or P4 node is being visited, PE1 build the Reverse Path Segment List TLV containing [N-ASBR4, EPE-ASBR4-ASBR1]. The Echo-reply will be an MPLS packet with this label stack and will be forwarded to PE1.

5. Building Reverse Path Segment List TLV dynamically

In some cases, the head-end may not have complete visibility of Inter-AS topology. In such cases, it can rely on downstream routers to build the reverse path for mpls traceroute procedures. For this purpose, the Reverse Path Segment List TLV is carried in the Echo-Reply.

5.1. The procedures to build the reverse path

When an ASBR receives an echo-request from another AS, and ASBR is configured to build the Reverse Path dynamically, ASBR MUST build a Reverse Path Segmnet List TLV and add it in echo-reply. ASBR MUST locally decide the outgoing interface for the echo-reply packet. Generally, remote ASBR will choose interface on which the incoming OAM packet was receieved to send the echo-reply out. Reverse Path Segment List TLV is built by adding two segment sub TLVs. The top segment sub TLV consists of the ASBR's Node SID and second segment consists of the EPE SID in the reverse direction to reach the AS from which the OAM packet was received. The type of segment chosen to build Reverse Path Segment List TLV is implementation dependent. In cases where the AS is configured with different SRGBs, the Node SID of the ASBR should be represented using type 3 segment so that all the nodes inside the AS can correctly translate the Node-SID to a label.

Irrespective of which type of segment is included in the Reverse Path Segment List TLV, the responder of echo-request always translates the Reverse Path Segment List TLV to a label stack and builds MPLS header for the the echo-reply packet.

5.2. Details with example

Let us consider a traffic engineered path built from PE1 to PE4 with a label stack as below. N-P1, N-ASBR1, EPE-ASBR1-ASBR4, N-PE4 for the following procedures. This traceroute doesn't need any Reverse Path Segment List TLV till it leaves AS1, because IP connectivity will be there to send echo-reply. But this traceroute requires Reverse Path Segment List TLV once it starts probing AS2 routers. According to this procedure, ASBR4 should add Reverse Path Segment List TLV in its echo-reply. ASBR4 should form this Reverse Path Segment List TLV using its own Node SID(N-ASBR4) and EPE SID (EPE-ASRB4-ASBR1) labels. Then PE1 should use this Reverse Path Segment List TLV in subsequent echo-requests. In this example, when the subsequent echo-request reaches P3, it should use this Reverse Path Segment List TLV for sending the echo-reply. The same Reverse Path Segment List TLV is enough for any router in AS2 to send the reply. Because the first label(N-ASBR4) can direct echo-reply to ASBR4 and second one (EPE-ASBR4-ASBR1) to direct echo-reply to AS1. Once echo reply reaches AS1, normal IP forwarding helps it to reach PE1 or the head-end.

6. Security Considerations

TBD

7. IANA Considerations

Multiprotocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters TLVs Registry

Reverse Path Segment List TLV : TBD

8. Contributors

1.Carlos Pignataro

Cisco Systems, Inc.

cpignata@cisco.com

2. Zafar Ali

Cisco Systems, Inc.

zali@cisco.com

9. Acknowledgments

Thanks to Bruno Decreane for suggesting use of generic Segment sub-TLV.

10. References

10.1. Normative References

- [I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Mattes, P., Rosen, E., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-07 (work in progress), July 2019.
- [I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.
- [RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <<https://www.rfc-editor.org/info/rfc8287>>.

10.2. Informative References

- [I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Talaulikar, K., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-19 (work in progress), May 2019.
- [I-D.ietf-mpls-interas-lspping]
Nadeau, T. and G. Swallow, "Detecting MPLS Data Plane Failures in Inter-AS and inter-provider Scenarios", draft-ietf-mpls-interas-lspping-00 (work in progress), March 2007.

- [I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-22 (work in progress), May 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.
- [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>>.
- [RFC7743] Luo, J., Ed., Jin, L., Ed., Nadeau, T., Ed., and G. Swallow, Ed., "Relayed Echo Reply Mechanism for Label Switched Path (LSP) Ping", RFC 7743, DOI 10.17487/RFC7743, January 2016, <<https://www.rfc-editor.org/info/rfc7743>>.
- [RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <<https://www.rfc-editor.org/info/rfc8029>>.
- [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>>.
- [RFC8403] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC 8403, DOI 10.17487/RFC8403, July 2018, <<https://www.rfc-editor.org/info/rfc8403>>.
- [RFC8604] Filsfils, C., Ed., Previdi, S., Dawra, G., Ed., Henderickx, W., and D. Cooper, "Interconnecting Millions of Endpoints with Segment Routing", RFC 8604, DOI 10.17487/RFC8604, June 2019, <<https://www.rfc-editor.org/info/rfc8604>>.

Authors' Addresses

Shraddha Hegde
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Kapil Arora
Juniper Networks Inc.

Email: kapilaro@juniper.net

Samson Ninan
Juniper Networks Inc.

Email: samsonn@juniper.net

Mukul Srivastava
Juniper Networks Inc.

Email: msri@juniper.net

Nagendra Kumar
Cisco Systems, Inc.

Email: naikumar@cisco.com