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

Copyright Notice

Table of Contents

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.

```
            +--------------------------| N100 |------------------------+
            |                                                          |
               ====== link1====== link3------ link5====== link9------
               ||N1||======||N2||======| N3 |======||N4||======| N5 |
               ||  ||------||  ||------|    |------||  ||------|    |
               ====== link2====== link4------ link6======link10------
                              |                       |
                              |       ------          |
                              +-------| N6 |---------+
                                link7 |    | link8
                                      ------
```

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.      ;; Ref1
Ref1: 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 responses 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 responses 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 responses 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.

- 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:

- 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).

- When node N2 receives the packet with hop-count = 1, it processes the hop count expiry. Specifically, the node N2 responses with the ICMPv6 message (Type: "Time Exceeded", Code: "Time to Live exceeded in Transit").

- 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.

- 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 responses 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 responses 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.O 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.O=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