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.
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.
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: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


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:

- 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.
- 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.
- 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.
- 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:

- 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).

- 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.

- 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.

- 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.

  - 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:

- 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).

- 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.

- 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.

- 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.

- 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:

- If the router receives an undeliverable IP datagram, or
- If the router receives a packet with a Hop Limit of zero, or
- If the router receives a packet such that if the router decrements the packet’s Hop Limit it becomes zero, or
- 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
- 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.

```plaintext
> 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.

- 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 \((A:1::, B:2:C31)(B:4:C52, B:4:OTP, B:2:C31; \text{SL}=2; \text{NH}=UDP)(\text{Traceroute probe})\).

- 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 \((A:1::, B:4:OTP)(B:4:C52, B:4:OTP, B:2:C31; HC=1, SL=1; NH=UDP)(\text{Traceroute probe})\) 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").
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.

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.

- 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.

- 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.

- As SL is not equal to zero (i.e., it’s not egress node), node N2 responds 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.

- 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.

- 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 Registry

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:

<table>
<thead>
<tr>
<th>Code</th>
<th>Name</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No Error</td>
<td>This document</td>
</tr>
<tr>
<td>1</td>
<td>SID is not locally implemented</td>
<td>This document</td>
</tr>
<tr>
<td>2</td>
<td>O-flag punt at Transit</td>
<td>This document</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Value (Suggested Value)</th>
<th>Endpoint Behavior</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBA (40)</td>
<td>End.OP</td>
<td>[This.ID]</td>
</tr>
<tr>
<td>TBA (41)</td>
<td>End.OTP</td>
<td>[This.ID]</td>
</tr>
</tbody>
</table>
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
9. References

9.1. Normative References

[I-D.ietf-6man-segment-routing-header]

[I-D.ietf-spring-srv6-network-programming]


9.2. Informative References


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
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.
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.iets-ippm-ioam-data] are added as sub-TLVs.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  SRH-TLV-Type |     LEN       |    RESERVED                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|  IOAM-Type    | IOAM HDR LEN  |    RESERVED                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  I
|                                                               |  O
˜                 IOAM Option and Data Space                    ˜  M
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|                                                               |
|                   Payload + Padding (L2/L3/...)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

**Figure 1: IOAM data encapsulation in SRH**

- **SRH-TLV-Type**: IOAM TLV Type for SRH is defined as TBA1.
- **IOAM-Type**: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.iets-ippm-ioam-data].
- **IOAM HDR LEN**: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.
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].

+-----------------+--------------------------+---------------+
<table>
<thead>
<tr>
<th>SRH TLV Type</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBA1 Greater</td>
<td>TLV for IOAM Data Fields</td>
<td>This document</td>
</tr>
<tr>
<td>than 128</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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


8.2. Informative References

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: fbrockner@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
Abstract

This document defines a new IPv6 Routing header type, called the Compressed Routing Header (CRH). SRv6+ nodes use the CRH to steer packets from segment to segment along SRv6+ paths.

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.
1.  Introduction

This document defines a new IPv6 [RFC8200] Routing header type, called the Compressed Routing Header (CRH). SRv6+ [I-D.bonica-spring-srv6-plus] nodes use the CRH to steer packets from segment to segment along SRv6+ paths.
For details regarding SRv6+ paths, segments, Segment Identifiers (SIDs) and instructions, see [I-D.bonica-spring-srv6-plus].

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. The Compressed Routing Header (CRH)

Figure 1: Compressed Routing Header (CRH)

Figure 1 depicts the CRH. The CRH contains the following fields:

- **Next Header** - Defined in [RFC8200].
- **Hdr Ext Len** - Defined in [RFC8200].
- **Routing Type** - Defined in [RFC8200]. Value TBD by IANA. (Suggested value: 5)
- **Segments Left** - Defined in [RFC8200].
- **Last Entry** - 8 bits. Represents the zero-based index of the last element of the Segment List.
- **Com (Compression)** - 2 bits. Represents the length of each entry in the SID List. Values are reserved (0), sixteen bits (1), thirty-two bits (2), and reserved (3). In order to maximize header compression, this value should reflect the smallest feasible Maximum SID Value (MSV). See Section 5.1 of [I-D.bonica-spring-srv6-plus] for MSV details.
- **Reserved** - SHOULD be set to zero by the sender. MUST be ignored by the receiver.

- **SID List** - Represents the SRv6+ 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.

Figure 2 and Figure 3 illustrate CRH encodings with Com equal to 1 and 2. In all cases, the CRH MUST end on a 64-bit boundary. Therefore, the CRH MAY be padded with zeros.

![Figure 2: Sixteen-bit Encoding (Com equals 1)](image1)

![Figure 3: Thirty-two bit Encoding (Com equals 2)](image2)
4. Segment Forwarding Information Base (SFIB)

A segment ingress node MUST maintain one Segment Forwarding Information Base (SFIB) entry for each segment that it originates. Each SFIB entry contains the following information:

- A SID
- A segment type
- Topological instruction parameters

The following are valid segment types:

- Strictly-routed
- Loosely-routed

The following parameters are associated with topological instructions that control strictly-routed segments:

- An IPv6 address that identifies an interface on the segment egress node.
- A primary interface identifier.
- Zero or more secondary interface identifiers.

Loosely-routed segments are associated with a single topological instruction parameter. This parameter is an IPv6 address that identifies an interface on the segment egress node.

5. Processing Rules

5.1. General

[ RFC8200 ] defines rules that apply to IPv6 extension headers, in general, and IPv6 Routing headers, in particular. All of these rules apply to the CRH.

For example:

- Extension headers (except for the Hop-by-Hop Options header) are not processed, inserted, or deleted by any node along a packet’s delivery path, until the packet reaches the node (or each of the set of nodes, in the case of multicast) identified in the Destination Address field of the IPv6 header.
o If, while processing a received packet, a node encounters a Routing header with an unrecognized Routing Type value, the required behavior of the node depends on the value of the Segments Left field. If Segments Left is zero, the node must ignore the Routing header and proceed to process the next header in the packet, whose type is identified by the Next Header field in the Routing header. If Segments Left is non-zero, the node must discard the packet and send an ICMPv6 [RFC4443] Parameter Problem, Code 0, message to the packet’s Source Address, pointing to the unrecognized Routing Type.

o If, after processing a Routing header of a received packet, an intermediate node determines that the packet is to be forwarded onto a link whose link MTU is less than the size of the packet, the node must discard the packet and send an ICMPv6 Packet Too Big message to the packet’s Source Address.

5.2. CRH Specific

When a node recognizes and processes a CRH, it executes the following procedure:

o If the IPv6 Source Address is a link-local address, discard the packet.

o If the IPv6 Source Address is a multicast address, discard the packet.

o If Segments Left equal 0, skip over the CRH and process the next header in the packet.

o If Segments Left is greater than Last Entry plus one, discard the packet and send an ICMPv6 Parameter Problem, Code 0, message to the Source Address, pointing to the Segments Left field.

o If Com is equal to (0) or (3) Reserved, discard the packet and send an ICMPv6 Parameter Problem, Code 0, message to the Source Address, pointing to the Com field.

o Compute L, the minimum CRH length (See Section 5.2.1)

o If L is equal to zero or L is greater than Hdr Ext Len, discard the packet and send an ICMPv6 Parameter Problem, Code 0, message to the Source Address, pointing to the Last Entry field.

o Decrement the packet’s Hop Count.
o If the Hop Count has expired, discard the packet and send an ICMPv6 Time Expired message to the packet’s source node.

o Decrement Segments Left

o Search for the current SID in the SFIB.

o If the above-mentioned search does not return an SFIB entry, discard the packet and send an ICMPv6 Parameter Problem, Code 0, message to the Source Address, pointing to the current SID.

o If the above-mentioned search returns an SFIB entry and the segment type is strictly-routed, execute the strictly-routed topological instruction described in Section 5.2.2.

o If the above-mentioned search returns an SFIB entry and the segment type is loosely-routed, execute the loosely-routed topological instruction described in Section 5.2.3.

The above stated rules are demonstrated in Appendix A.

5.2.1. Computing Minimum CRH Length

The algorithm described in this section accepts the following CRH fields as its input parameters:

o Compression (Com).

o Last Entry.

It yields L, the minimum CRH length. The minimum CRH length is measured in 8-octet units, not including the first 8 octets.
if (Com == 1) { /* Sixteen bit encoding */
    L = ( (Last Entry + 1) / 4 );
    if ( (Last Entry + 1) % 4 )
        L++;
}        
elsif (Com == 2) { /* Thirty-two bit encoding */
    L = ( (Last Entry + 1) / 2 );
    if ( (Last Entry + 1) % 2 )
        L++;
}        
else { /* Invalid Com */
    L = 0xFF
}

return(0)

5.2.2. Strictly-Routed Topological Instructions

A strictly-routed topological instruction accepts the following parameters:

- An IPv6 address that identifies an interface on the segment egress node.
- A primary interface identifier.
- Zero or more secondary interface identifiers.

A strictly-routed topological instruction behaves as follows:

- If none of the interfaces identified by the above-mentioned parameters are operational, discard the packet and send an ICMPv6 Destination Unreachable message (Code: 5, Source Route Failed) to the packet’s source node.
- Overwrite the packet’s Destination Address with the IPv6 address that was received as a parameter.
- If the primary interface is active, forward the packet through the primary interface.
If the primary interface is not active and any of the secondary interfaces are active, forward the packet through one of the secondary interfaces. Execute procedures so that all packets belonging to a flow are forwarded through the same secondary interface.

5.2.3. Loosely-Routed Topological Instructions

A loosely-routed topological instruction accepts a single parameter. This parameter is an IPv6 address that identifies an interface on the segment egress node.

A loosely-routed topological instruction behaves as follows:

- If the segment ingress node does not have a viable route to the IPv6 address included as a parameter, discard the packet and send an ICMPv6 Destination Unreachable message (Code:1 Net Unreachable) to the packet’s source node.
- Overwrite the packet’s Destination Address with the destination address that was included as a parameter.
- 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.

6. Mutability

In the CRH, the Segments Left field is mutable. All remaining fields are immutable.

7. Compliance

In order to be compliant with this specification, an implementation MUST support 32-bit SID encoding. It MAY also support 16-bit SID encoding.

8. Management Considerations

PING and TRACEROUTE [RFC2151] both operate correctly in the presence of the CRH.
9. Security Considerations

The CRH can be used within trusted domains only. In order to enforce this requirement, domain edge routers MUST do one of the following:

- Discard all inbound packets that are destined for infrastructure interfaces and contain a CRH
- Authenticate [RFC4302] [RFC4303] all inbound packets that are destined for infrastructure interfaces and contain a CRH

10. IANA Considerations

This document makes the following registration in the Internet Protocol Version 6 (IPv6) Parameters "Routing Type" registry maintained by IANA:

<table>
<thead>
<tr>
<th>Suggested Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Compressed Routing Header (CRH)</td>
<td>This document</td>
</tr>
</tbody>
</table>

11. Acknowledgements

Thanks to Joel Halpern, Tony Li, Gerald Schmidt, Nancy Shaw and Chandra Venkatraman for their comments.

12. References

12.1. Normative References

[I-D.bonica-spring-srv6-plus]


12.2. Informative References


Appendix A. CRH Processing Examples

This appendix provides examples of CRH processing in the following applications:

- Loose source routing (Appendix A.1)
- Loose source routing preserving the first SID (Appendix A.2)
- Strict source routing (Appendix A.3)
Figure 4: Reference Topology

Figure 4 provides a reference topology that is used in all examples.

<table>
<thead>
<tr>
<th>Instantiating Node</th>
<th>SID</th>
<th>Segment Type</th>
<th>IPv6 Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>1</td>
<td>Loosely-routed</td>
<td>2001:db8::1</td>
</tr>
<tr>
<td>All</td>
<td>2</td>
<td>Loosely-routed</td>
<td>2001:db8::2</td>
</tr>
<tr>
<td>All</td>
<td>3</td>
<td>Loosely-routed</td>
<td>2001:db8::3</td>
</tr>
<tr>
<td>All</td>
<td>10</td>
<td>Loosely-routed</td>
<td>2001:db8::a</td>
</tr>
<tr>
<td>All</td>
<td>11</td>
<td>Loosely-routed</td>
<td>2001:db8::b</td>
</tr>
</tbody>
</table>

Table 1: Loosely Routed SIDs

Table 1 describes SFIB entries that are instantiated on all nodes. All of these SFIB entries represent loosely-routed segments.
Table 2: Strictly Routed SIDs

Table 2 describes SFIB entries that are instantiated on specific nodes. All of these SFIB entries represent strictly-routed segments.

A.1. Loose Source Routing

In this example, Node S sends a packet to Node D, specifying loose source route through Node I3. In this example, the first node in the path, I3, does not appear in the CRH segment list. Therefore, the destination node may not be able to send return traffic through the same path.

As the packet travels from S to I3:

<table>
<thead>
<tr>
<th>Source Address = 2001:db8::a</th>
<th>Last Entry = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address = 2001:db8::3</td>
<td>Segments Left = 1</td>
</tr>
<tr>
<td></td>
<td>SID[0] = 11</td>
</tr>
</tbody>
</table>

As the packet travels from I3 to D:

<table>
<thead>
<tr>
<th>Source Address = 2001:db8::a</th>
<th>Last Entry = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Address = 2001:db8::b</td>
<td>Segments Left = 0</td>
</tr>
<tr>
<td></td>
<td>SID[0] = 11</td>
</tr>
</tbody>
</table>

A.2. Loose Source Routing Preserving The First SID

In this example, Node S sends a packet to Node D, specifying loose source route through Node I3. In this example, the first node in the path, I3, appears in the CRH segment list. Therefore, the destination node can send return traffic through the same path.
A.3. Strict Source Routing

In this example, Node S sends a packet to Node D, specifying the
strict source route through I1 and I3.
As the packet travels from I3 to D:

| Source Address = 2001:db8::a |
| Destination Address = 2001:db8:0:b::2 |
| Last Entry = 1 |
| Segments Left = 0 |
| SID[0] = 129 |
| SID[1] = 129 |

Authors’ Addresses

Ron Bonica  
Juniper Networks  
2251 Corporate Park Drive  
Herndon, Virginia  20171  
USA

Email: rbonica@juniper.net

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

Email: y.kamite@ntt.com

Tomonobu Niwa  
KDDI  
3-22-7, Yoyogi, Shibuya-ku  
Tokyo  151-0053  
Japan

Email: to-niwa@kddi.com

Andrew Alston  
Liquid Telecom  
Nairobi  
Kenya

Email: Andrew.Alston@liquidtelecom.com
The Per-Segment Service Instruction (PSSI) Option
draft-bonica-6man-seg-end-opt-04

Abstract

SRv6+ encodes Per-Segment Service Instructions (PSSI) in a new IPv6 option, called the PSSI Option. This document describes the PSSI Option.

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.
1. Introduction

An SRv6+ [I-D.bonica-spring-srv6-plus] path provides unidirectional connectivity from its ingress node to its egress node. While an SRv6+ path can follow the least cost path from ingress to egress, it can also follow any other path.

An SRv6+ path contains one or more segments. A segment provides unidirectional connectivity from its ingress node to its egress node.

SRv6+ paths are programmable. They support several instruction types, including Per-Segment Service Instructions (PSSI). The following are examples of PSSIs:

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

PSSIs are executed at segment egress nodes and can be used to implement limited service chains. However, they do not provide an alternative to the Network Service Header (NSH) [RFC8300].
SRv6+ encodes PSSIs in a new IPv6 option, called the PSSI Option. This document describes the PSSI Option.

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. PSSI Identifiers

PSSI Identifiers identify PSSIs. They have domain-wide significance. When a controller creates a limited service chain, also allocates a PSSI Identifier. It then distributes the following information to each node that contributes to the limited service chain:

- The PSSI Identifier.
- The PSSI that the node should execute when it receives a packet that has the PSSI Identifier encoded within it.

4. Option Format

The PSSI Option contains the following fields:

- Option Type: 8-bit selector. PSSI option. Value TBD by IANA. (Suggested value: 0x10). See Note below.
- Opt Data Len - 8-bit unsigned integer. Length of the option, in octets, excluding the Option Type and Option Length fields. This field MUST be set to 4.
- PSSI identifier - (32-bit selector). Identifies a PSSI.

The PSSI option MAY appear in any Destination Options header, regardless of whether that Destination Options header precedes a Routing header or an upper-layer header. The PSSI option MUST NOT appear in a Hop-by-hop Options header.

NOTE: The highest-order two bits of the Option Type (i.e., the "act" bits) are 00. These bits specify the action taken by a destination node that does not recognize the option. The required action is to skip over this option and continue processing the header.
The third highest-order bit of the Option Type (i.e., the "chg" bit) is 0. This indicates that Option Data cannot be modified along the path between the packet’s source and its destination.

5. Security Considerations

The PSSI option shares many security concerns with IPv6 routing headers. In particular, any boundary filtering protecting a domain from external routing headers should also protect against external PSSI options being processed inside a domain. This occurs naturally if encapsulation is used to add routing headers to a packet. If external routing headers are allowed, then protections must also include ensuring that any provided PSSI option is properly protected, e.g. with an IPSEC AH header or other suitable means.

As with Routing headers, the security assumption within a domain is that the domain is trusted to provide, and to avoid improperly modifying, the PSSI Option.

6. IANA Considerations

IANA is requested to allocate a codepoint from the Destination Options and Hop-by-hop Options registry (https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xhtml#ipv6-parameters-2). This option is called "PSSI". The "act" bits are 00 and the "chg" bit is 0. (Suggested value: 0x10).

7. Acknowledgements

Thanks to Fred Baker and Shizhang Bi for their careful review of this document.

8. Normative References

[I-D.bonica-spring-srv6-plus]


Authors’ Addresses

Ron Bonica
Juniper Networks
2251 Corporate Park Drive
Herndon, Virginia 20171
USA

Email: rbonica@juniper.net

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

Email: joel.halpern@ericsson.com

Yuji Kamite
NTT Communications Corporation
3-4-1 Shibaura, Minato-ku
Tokyo 108-8110
Japan

Email: y.kamite@ntt.com
Tomonobu Niwa  
KDDI  
3-22-7, Yoyogi, Shibuya-ku  
Tokyo 151-0053  
JP  
Email: to-niwa@kddi.com

Ning So  
Reliance Jio  
3010 Gaylord PKWY, Suite 150  
Frisco, Texas  75034  
USA  
Email: Ning.So@ril.com

Fengman Xu  
Reliance Jio  
3010 Gaylord PKWY, Suite 150  
Frisco, Texas  75034  
USA  
Email: Fengman.Xu@ril.com

Gang Chen  
Baidu  
No.10 Xibeiwang East Road Haidian District  
Beijing 100193  
P.R. China  
Email: phdgang@gmail.com

Yongqing Zhu  
China Telecom  
109 West Zhongshan Ave, Tianhe District  
Guangzhou  
P.R. China  
Email: zhuyq.gd@chinatelecom.cn
Guangming Yang
China Telecom
109 West Zhongshan Ave, Tianhe District
Guangzhou
P.R. China

Email: yanggm.gd@chinatelecom.cn

Yifeng Zhou
ByteDance
Building 1, AVIC Plaza, 43 N 3rd Ring W Rd Haidian District
Beijing 100000
P.R. China

Email: yifeng.zhou@bytedance.com
The Per-Path Service Instruction (PPSI) Option
draft-bonica-6man-vpn-dest-opt-06

Abstract

SRv6+ encodes Per-Path Service Instructions (PPSI) in a new IPv6 option, called the PPSI Option. This document describes the PPSI Option.

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 7, 2020.
1. Introduction

An SRv6+ [I-D.bonica-spring-srv6-plus] path provides unidirectional connectivity from its ingress node to its egress node. While an SRv6+ path can follow the least cost path from ingress to egress, it can also follow any other path.

SRv6+ paths are encoded as IPv6 [RFC8200] header chains. When an SRv6+ ingress node receives a packet, it encapsulates the packet in an IPv6 header chain. It then forwards the encapsulated packet to the path’s egress node. When the egress node receives the packet, it processes the SRv6+ payload (i.e., the original packet).

SRv6+ paths are programmable. They support several instruction types, including Per-Path Service Instructions (PPSI). PPSIs determine how path egress nodes process SRv6+ payloads. In the
absence of a PPSI, the egress node processes SRv6+ payloads as
described in [RFC8200].

The following are examples of PPSIs:

- Remove any SRv6+ encapsulation and forward the SRv6+ payload
  through a specified interface.
- Remove any SRv6+ encapsulation and forward the SRv6+ payload using
  a specified routing table.

SRv6+ encodes PPSIs in a new IPv6 option, called the PPSI Option.
This document describes the PPSI Option.

PPSIs can be used to support Virtual Private Networks (VPN).
Therefore, Appendix A of this document describes VPN technology and
how PPSIs can be used to support a VPN.

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. PPSI Identifiers

PPSI Identifiers identify PPSIs. When a path egress node
instantiates a PPSI, it also allocates a PPSI Identifier and
associates the PPSI with the identifier.

PPSI Identifiers have node-local significance. This means that a
path egress node must assign a unique PPSI Identifier to each PPSI
that it instantiates. However, one path egress node can assign a
PPSI Identifier to an instruction that it instantiates, while another
path egress node can assign the same PPSI Identifier to a different
PPSI that it instantiates.

4. The PPSI Option

The PPSI Option contains the following fields:

- Option Type: 8-bit selector. PPSI option. Value TBD by IANA.
  (Suggested value: 144). See Note below.
o Opt Data Len - 8-bit unsigned integer. Length of the option, in octets, excluding the Option Type and Option Length fields. This field MUST be set to 4.

o PPSI identifier - (32-bit selector). Identifies a PPSI.

The SRv6+ PPSI option MAY appear in a Destination Options header that precedes an upper-layer header. It MUST NOT appear in a Hop-by-hop Options header or in a Destination Options header that precedes a Routing header.

When the SRv6+ PPSI option appears in a Destination Options header, it MUST be the only option listed in the header. This is because the PPSI defines all path egress node behaviors.

NOTE: The highest-order two bits of the Option Type (i.e., the "act" bits) are 10. These bits specify the action taken by a destination node that does not recognize the option. The required action is to discard the packet and, regardless of whether or not the packet’s Destination Address was a multicast address, send an ICMPv6 [RFC4443] Parameter Problem, Code 2, message to the packet’s Source Address, pointing to the unrecognized Option Type.

The third highest-order bit of the Option Type (i.e., the "chg" bit) is 0. This indicates that Option Data cannot be modified along the path between the packet’s source and its destination.

5. Destination Option Header Considerations

As per [RFC8200], the Destination Options header includes a Next Header field. The Next Header field identifies the header following the Destination Options header.

SRv6+ can carry Ethernet payload after a Destination option header. Therefore, this document requests IANA to assign a protocol number for Ethernet. (The suggested value is 143.)

6. Security Considerations

SRv6+ 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 SRv6+ packets

o Authenticate [RFC4302] [RFC4303] all inbound SRv6+ packets
7. IANA Considerations

IANA is requested to allocate a code point from the Destination Options and Hop-by-hop Options registry (https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xhtml#ipv6-parameters-2). This option is called "Per-Path Service Instruction Option". The "act" bits are 10 and the "chg" bit is 0. The suggested value is 144.

IANA is also requested to allocate a code point for Ethernet from the Assigned Internet Protocol Numbers registry (https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml). The suggested value is 143.

8. Acknowledgements

Thanks to Brian Carpenter, Adrian Farrel, Tom Herbert, John Leddy and Tony Li for their comments.

9. References

9.1. Normative References

[I-D.bonica-spring-srv6-plus]


9.2. Informative References


Appendix A. Virtual Private Networks (VPN)

Virtual Private Network (VPN) technologies allow network providers to emulate private networks with shared infrastructure. For example, assume that red sites and blue sites connect to a provider network. The provider network facilitates communication among red sites and facilitates communication among blue sites. However, it prevents communication between red sites and blue sites.

The IETF has standardized many VPN technologies, including:

- Layer 2 VPN (L2VPN) [RFC6624].
- Layer 3 VPN (L3VPN) [RFC4364].
- Virtual Private LAN Service (VPLS) [RFC4761][RFC4762].
- Ethernet VPN (EVPN) [RFC7432].
- Pseudowires [RFC8077].

The above-mentioned technologies include the following components:

- Customer Edge (CE) devices.
- Provider Edge (PE) devices.
- Routing Instances.
- Service Instructions.
- Service Instruction Identifiers.
- Transport tunnels.

CE devices participate in closed communities called VPNs. CEs that participate in one VPN can communicate with one another but cannot communicate with CEs that participate in another VPN.
CE devices connect to provider networks through PE devices. Each PE maintains one Routing Instance for each VPN that it supports. A Routing Instance is a VPN specific Forwarding Information Base (FIB). In EVPN, Routing Instances are called Ethernet Virtual Instances (EVI).

Assume that one CE sends a packet through a provider network to another CE. The packet enters the provider network through an ingress PE and leaves the provider network through an egress PE. The packet may traverse one or more intermediate nodes on route from PE to PE.

When the ingress PE receives the packet, it:

- Identifies the Routing Instance that supports the originating CE’s VPN.
- Searches that Routing Instance for the packet’s destination.

If the search fails, the ingress PE discards the packet. If the search succeeds, it yields the following:

- A Service Instruction Identifier.
- The egress PE’s IP address.

The ingress PE prepends the Service Instruction Identifier and a transport header to the packet, in that order. It then forwards the packet through a transport tunnel to the egress PE.

The egress PE removes the transport header, if it has not already been removed by an upstream device. It then examines and removes the Service Instruction Identifier. Finally, it executes a service instruction that is associated with the Service Instruction Identifier. The service instruction causes the egress PE to forward the packet to its destination (i.e., a directly connected CE).

In the above-mentioned VPN technologies, the ingress PE encodes Service Instruction Identifiers in Multiprotocol Label Switching (MPLS) [RFC3031] labels. Depending upon the transport tunnel type, the transport header can be:

- A MPLS label or label stack.
o A Generic Routing Encapsulation (GRE) [RFC2784] header encapsulated in IPv4 or IPv6.

Some PE devices cannot process MPLS headers. While these devices have several alternatives to MPLS-based transport tunnels, they require an alternative to MPLS-based encoding of Service Instruction Identifiers. The PPSI Option can be used to encode Service Instruction Identifiers. It is applicable when VPN payload is transported over IPv6.

Authors’ Addresses

Ron Bonica
Juniper Networks
2251 Corporate Park Drive
Herndon, Virginia 20171
USA

Email: rbonica@juniper.net

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

Email: y.kamite@ntt.com

Chris Lenart
Verizon
22001 Loudoun County Parkway
Ashburn, Virginia 20147
USA

Email: chris.lenart@verizon.com

Ning So
Reliance Jio
3010 Gaylord PKWY, Suite 150
Frisco, Texas 75034
USA

Email: Ning.So@ril.com
Fengman Xu
Reliance Jio
3010 Gaylord PKWY, Suite 150
Frisco, Texas 75034
USA
Email: Fengman.Xu@ril.com

Greg Presbury
Hughes Network Systems
11717 Exploration Lane
Germantown, Maryland 20876
USA
Email: greg.presbury@hughes.com

Gang Chen
Baidu
No.10 Xibeiwang East Road Haidian District
Beijing 100193
P.R. China
Email: phdgang@gmail.com

Yongqing Zhu
China Telecom
109 West Zhongshan Ave, Tianhe District
Guangzhou
P.R. China
Email: zhuyq.gd@chinatelecom.cn

Guangming Yang
China Telecom
109 West Zhongshan Ave, Tianhe District
Guangzhou
P.R. China
Email: yanggm.gd@chinatelecom.cn
Yifeng Zhou
ByteDance
Building 1, AVIC Plaza, 43 N 3rd Ring W Rd Haidian District
Beijing 100000
P.R. China

Email: yifeng.zhou@bytedance.com
IPv6 Support for Segment Routing: SRv6+
draft-bonica-spring-srv6-plus-04

Abstract

This document describes SRv6+. SRv6+ 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. SRv6+ 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 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. Overview ......................................................... 3
2. Requirements Language .......................................... 4
3. Paths, Segments And Instructions ............................... 5
4. Segment Types .................................................... 6
   4.1. Strictly Routed .............................................. 6
   4.2. Loosely Routed ............................................. 7
5. Segment Identifiers (SID) ......................................... 7
   5.1. Range ....................................................... 8
   5.2. Assigning SIDs to Strictly Routed Segments .............. 10
   5.3. Assigning SIDs to Loosely Routed Segments .............. 10
6. Service Instructions ............................................... 10
   6.1. Per-Segment ............................................... 10
   6.2. Per-Path .................................................. 11
7. The IPv6 Data Plane ............................................... 11
   7.1. The Routing Header ........................................ 12
   7.2. The Destination Options Header ............................ 13
8. Control Plane ..................................................... 14
9. Differences Between SRv6 and SRv6+ ............................ 14
   9.1. Routing Header Size ...................................... 14
   9.2. Decoupling of Topological and Service Instructions ... 15
   9.3. Authentication ............................................ 16
   9.4. Traffic Engineering Capability .......................... 16
   9.5. IP Addressing Architecture ............................... 17
10. Compliance ...................................................... 17
11. Operational Considerations .................................... 18
   11.1. Ping and Traceroute ..................................... 18
   11.2. ICMPv6 Rate Limiting .................................... 18
   11.3. SID Lengths And SID Length Transitions ................ 18
12. IANA Considerations ........................................... 18
13. Security Considerations ........................................ 18
14. Acknowledgements .............................................. 19
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 also 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 mapping 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.


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

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 SRv6+ 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:

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

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

```
    A ---- B ---- C ---- D ---- E ---- F

    Segment A-C  Segment C-D  Segment D-F

SRv6+ Path
```

Figure 1: Paths, Segments And Instructions

Figure 1 depicts an SRv6+ 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

SRv6+ supports the following segment types:

- strictly routed
- loosely routed

Strictly routed segments forward packets through a specified link that connects the segment ingress node to the segment egress node. Loosely routed segments forward packets through the least cost path from the segment ingress node to the segment egress node.

Each segment type is described below.

4.1. Strictly Routed

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

- An IPv6 address that identifies an interface on the segment egress node.
- A primary interface identifier.
- Zero or more secondary interface identifiers.
The topological instruction behaves as follows:

- If none of the interfaces identified by the above-mentioned parameters are operational, discard the packet and send an ICMPv6 [RFC4443] Destination Unreachable message (Code: 5, Source Route Failed) to the packet’s source node.

- Overwrite the packet’s Destination Address with the IPv6 address that was received as a parameter.

- If the primary interface is active, forward the packet through the primary interface.

- If the primary interface is not active and any of the secondary interfaces are active, forward the packet through one of the secondary interfaces. Execute procedures so that all packets belonging to a flow are forwarded through the same secondary interface.

4.2. Loosely Routed

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

The topological instruction behaves as follows:

- If the segment ingress node does not have a viable route to the IPv6 address included as a parameter, discard the packet and send an ICMPv6 Destination Unreachable message (Code:1 Net Unreachable) to the packet’s source node.

- Overwrite the packet’s Destination Address with the destination address that was included as a parameter.

- 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.

5. Segment Identifiers (SID)

A Segment Identifier (SID) is an unsigned integer that identifies a segment. Because there is a one-to-one mapping 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.

Although SIDs have node-local significance, an SRv6+ 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.)

Although SIDs have node-local significance, they can be assigned in a
manner that facilitates debugging. 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:

- 65,535 (i.e., 2**16 minus 1)
- 4,294,967,295 (i.e., 2**32 minus 1)

In order to optimize packet encoding (Section 7.1), network operators
can configure all nodes within an SRv6+ domain to have the smallest
feasible MSV. The following paragraphs explain how an operator
determines the smallest feasible MSV.

Consider an SRv6+ 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 SRv6+ domain, we will
determine the smallest feasible MSV under the following conditions:

- The SRv6+ domain contains strictly routed segments only.
The SRv6+ domain contains loosely routed segments only.

The SRv6+ domain contains both strictly and loosely routed segments.

If an SRv6+ domain contains strictly routed segments only, and each node creates a strictly routed 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 SRv6+ 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 240 infrastructure links, the smallest feasible MSV will increase to 65,535.

If an SRv6+ domain contains loosely routed segments only, and every node creates a loosely routed 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 SRv6+ 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 SRv6+ domain contains both strictly and loosely routed 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:

- The number of loosely routed segments that each node will create, given that every node creates a loosely routed segment to every other node (i.e., 4,999).

- The number of strictly routed segments that each node will create, given that each node creates a strictly routed 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 SRv6+ 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 Strictly Routed Segments

Network operators can establish conventions by which they assign SIDs to strictly routed segments. These conventions can facilitate debugging.

For example, a network operator can reserved a range of SIDs for strictly routed 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 Loosely Routed Segments

In order to facilitate debugging, all loosely routed 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 SRv6+ 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 loosely routed 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.

6. Service Instructions

SRv6+ supports the following service instruction types:

- Per-segment
- 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:

- Expose a packet to a firewall policy.
- 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:

- De-encapsulate a packet and forward its newly exposed payload through a specified interface.
- 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

SRv6+ ingress nodes generate IPv6 header chains that represent SRv6+ 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 SRv6+ path can contain any valid combination of IPv6 extension headers. The following bullet points describe how SRv6+ leverages IPv6 extension headers:

- If an SRv6+ path contains multiple segments, the IPv6 header chain that represents it MUST contain a Routing header. The SRv6+ path MUST be encoded in the Routing header as an ordered sequence of SIDs.

- If an SRv6+ 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.

- If an SRv6+ 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 SRv6+ uses the Routing header and the Destination Options header.

7.1. The Routing Header

SRv6+ defines a new Routing header type, called the Compressed Routing Header (CRH) [I-D.bonica-6man-comp-rtg-hdr]. The CRH contains the following fields:

- Next Header - Identifies the header immediately following the CRH.
- Hdr Ext Len - Length of the CRH.
- Routing Type - Identifies the Routing header variant (i.e., CRH)
- Segments Left - The number of segments still to be traversed before reaching the path egress node.
- Last Entry - Represents the index of the last element of the Segment List.
- Com (Compression) - Represents the length of each entry in the SID List. Values are reserved (0), sixteen bits (1), thirty-two bits (2), and reserved (3). In order to maximize header compression, this value should reflect the smallest feasible MSV (Section 5.1).
- SID List - Represents the SRv6+ 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.

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:

- If Segments Left equal 0, skip over the CRH and process the next header in the packet.
- Decrement Segments Left.
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.

- 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.

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, SRv6+ defines the following new IPv6 options:

- The SRv6+ Per-Segment Service Instruction Option
  [I-D.bonica-6man-seg-end-opt]

- The SRv6+ Per-Path Service Instruction Option
  [I-D.bonica-6man-vpn-dest-opt]

The SRv6+ 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 SRv6+ 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:

- So that SRv6+ segment ingress nodes can flood information regarding strictly routed segments that they originate
- So that SRv6+ segment egress nodes can flood information regarding loosely routed segments that they terminate

BGP extensions [I-D.ssangli-idr-bgp-vpn-srv6-plus] are being defined so that SRv6+ path egress nodes can associated path-terminating service instructions with Network Layer Reachability Information (NLRI).

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, SRv6+ defines a Routing Header Type, called the Compressed Routing Header (CRH). The CRH contains a field that represents the SRv6+ path as an ordered sequence of SIDs. Within that field, SIDs can be 16 or 32 bits long.
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.

Large Routing headers are undesirable for the following reasons:

- 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.
- 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

SRv6+ 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
Therefore, network operators can use SRv6+ mechanisms to support topological instructions, service instructions, or both.

<table>
<thead>
<tr>
<th>Ethernet</th>
<th>Ethernet</th>
<th>Ethernet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Instruction</td>
<td>VXLAN</td>
<td>Dest</td>
</tr>
<tr>
<td></td>
<td>UDP</td>
<td>Option</td>
</tr>
<tr>
<td>Topological Instructions</td>
<td>CRH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IPv6</td>
<td>IPv6</td>
</tr>
</tbody>
</table>

**Figure 2: EVPN Design Alternatives**

Figure 2 illustrates this point by depicting design options available to network operators offering Ethernet Virtual Private Network (EVPN) services over Virtual eXtensible Local Area Network (VXLAN) [RFC7432]. 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 SRv6+ packets. However, AH processing is not defined in SRv6.

### 9.4. Traffic Engineering Capability

SRv6+ supports traffic engineering solutions that rely exclusively upon strictly routed segments. For example, consider an SRv6+ network whose diameter is 12 hops and whose minimum feasible MSV is 65,525. In that network, in the worst case, SRv6+ overhead is 72 bytes (i.e., a 40-byte IPv6 header and a 32-byte CRH).

SRv6 also supports traffic engineering solutions that rely exclusively upon strictly routed 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:

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

In SRv6+ an IPv6 address always represents a network interface, as per [RFC4291].

10. Compliance

In order to be compliant with this specification, an SRv6+ implementation MUST:

- Be able to process IPv6 options as described in Section 4.2 of [RFC8200].
- Be able to process the Routing header as described in Section 4.4 of [RFC8200].
- Be able to process the Destination Options header as described in Section 4.6 of [RFC8200].
- Recognize the CRH.
- Be able to encode an SRv6+ path in the CRH as an ordered sequence of 32-bit SIDs.
- Be able to process a CRH that includes 32-bit SIDs.

Additionally, an SRv6+ implementation MAY:

- Be able to encode an SRv6+ path in the CRH as an ordered sequence of 16-bit SIDs.
- Be able to process a CRH that includes 16-bit SIDs.
- Recognize the Per-Segment Service Instruction Option.
- Recognize the Per-Path Service Instruction Option.
11. Operational Considerations

11.1. Ping and Traceroute

Ping and Traceroute [RFC2151] both operate correctly in SRv6+ (i.e., in the presence of the CRH).

11.2. ICMPv6 Rate Limitting

As per [RFC4443], SRv6+ nodes rate limit the ICMPv6 messages that they emit.

11.3. SID Lengths And SID Length Transitions

An SRv6+ implementation MAY include a configuration option that determines how it encodes SIDs (i.e., in 16 or 32 bits). In order to reduce operational complexity, network operators typically configure their networks so that every node encodes SIDs identically.

As a network grows, its minimum feasible MSV may increase. In this case, the network may need to migrate from one SID encoding to another. The following bullet points describe a migration strategy for an SRv6+ network that is migrating from 16-bit SIDs to 32-bit SIDs:

- Ensure that all nodes can process a CRH that includes 32-bit SIDs.
- Configure each nodes so that encodes SIDs in 32-bits.
- Configure SIDs whose value exceeds 65,535.

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 SRv6+ Reserved SIDs". Values 0-15 are reserved for future use.

13. Security Considerations

SRv6+ domains MUST NOT span security domains. In order to enforce this requirement, security domain edge routers MUST do one of the following:

- Discard all inbound SRv6+ packets
- Authenticate [RFC4302] [RFC4303] all inbound SRv6+ packets
14. Acknowledgements

The authors wish to acknowledge Dr. Vanessa Ameen and John Scudder.

15. References

15.1. Normative References

[I-D.bonica-6man-comp-rtg-hdr]

[I-D.bonica-6man-seg-end-opt]

[I-D.bonica-6man-vpn-dest-opt]

[I-D.bonica-lsr-crh-isis-extensions]

[I-D.ssangli-idr-bgp-vpn-srv6-plus]

[RFC2119]

[RFC4291]


15.2. Informative References


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: shradhha@juniper.net
Gang Chen
Baidu
No.10 Xibeiwang East Road Haidian District
Beijing  100193
P.R. China

Email: phdgang@gmail.com
Abstract

This document specifies Deterministic Networking data plane operation for SRv6 encapsulated user data.

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 5, 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.

1. Introduction

Deterministic Networking (DetNet), as described in [I-D.ietf-detnet-architecture] provides a capability to carry specified data flows with extremely low data loss rates and bounded latency within a network domain. DetNet is enabled by a group of technologies, such as resource allocation, service protection and explicit routes.

Segment Routing (SR) leverages the source routing paradigm. An ingress node steers a packet through an ordered list of instructions, called "segments". SR can be applied over IPv6 data plane using the Segment Routing Extension Header (SRH, [I-D.ietf-6man-segment-routing-header]). A segment in segment routing terminology is not limited to a routing/forwarding function.
A segment can be associated to an arbitrary processing of the packet in the node identified by the segment. In other words, an SRv6 Segment can indicate functions that are executed locally in the node where they are defined. SRv6 network Programming ([I-D.filsfils-spring-srv6-network-programming]) describe the different segments and functions associated to them.

This document describes how to implement DetNet in an SRv6 enabled domain, including:

- Source routing, which steers the DetNet flows through the network according to an explicit path with allocated resources;
- Network programming, which applies instructions (functions) to packets in some special nodes (or even all the nodes) along the path in order to guarantee, e.g., service protection and congestion protection.

DetNet SRv6 encapsulation and new SRv6 functions ([I-D.filsfils-spring-srv6-network-programming]) for DetNet are defined in this document. Control plane and OAM are not in the scope of this document.

Control plane and OAM are not in the scope of this document.

2. Terminology and Conventions

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].

2.1. Terminology

Terminologies for DetNet go along with the definition in [I-D.ietf-detnet-architecture] and [RFC8402]. Other terminologies are defined as follows:

- NH: The IPv6 next-header field.
- SID: A Segment Identifier ([RFC8402]).
- SRH: The Segment Routing Header ([I-D.ietf-6man-segment-routing-header]).
2.2. Conventions

Conventions in the document are defined as follows:

- NH=SRH means that NH is 43 with routing type 4 which is (as defined in [I-D.ietf-6man-segment-routing-header]), the values representing the SRH.

- 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.

- 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. It has to be noted that [I-D.ietf-6man-segment-routing-header] defines the segment list encoding in the reverse order of the path. A path represented by <S1,S2,S3>, will be encoded in the SRH as follows:

  \[
  \text{SegmentList}[0]=S3 \\
  \text{SegmentList}[1]=S2 \\
  \text{SegmentList}[2]=S1 
  \]

  The reverse encoding has been defined in order to optimise the processing time of the segment list. See [draft-ietf-6man-segment-routing-header] for more details.

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

  IPv6 header with source and destination addresses SA and DA respectively, and next-header set to SRH (i.e.: 43 with type 4), with a list of segments (SIDs) <S1, S2, S3> with SegmentsLeft = SL

  The payload of the packet is not represented

  (S3, S2, S1; SL) represents the same SID list as <S1, S2, S3>, 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

3. SRv6 DetNet Data Plane Overview
3.1. SRv6 DetNet Data Plane Layers

[I-D.ietf-detnet-architecture] decomposes the DetNet data plane into two sub-layers: service sub-layer and transport sub-layer. Different from DetNet MPLS data plane solution, which uses DetNet Control Word (d-CW) and S-Label to support service sub-layer and uses T-Label to support transport sub-layer, no explicit sub-layer division exists in SRv6 data plane. A classical SRv6 DetNet data plane solution is showed in the picture below:

```
+-------------------+
| Outer IPv6 Header |
+-------------------+
| SRH               |
+-------------------+       +-------------------+
| Ipv6 Header       | ----> | Ipv6 Header       |
+-------------------+       +-------------------+
```

The outer IPv6 Header with the SRH is used for carrying DetNet flows. Traffic Engineering is instantiated in the segment list of SRH, and other functions and arguments for service protection (packet replication, elimination and ordering) and congestion control (packet queuing and forwarding) are also defined in the SRH.

3.2. SRv6 DetNet Data Plane Scenarios

The figure above shows that an IPv6 flow is sent out from the end station E1. The packet of the flow is encapsulated in an outer IPv6+SRH header as a DetNet SRv6 packet in the Ingress (In) and transported through an SRv6 DetNet domain. In the Egress (Eg), the outer IPv6 header+SRH of the packet is popped, and the packet is sent to the destination E2.

The figure above shows that an IPv6 flow is sent our from the end station E1. The packet of the flow is encapsulated as a DetNet SRv6 packet in the Ingress (In) and transported through an SRv6 DetNet domain. In the Egress (Eg), the upper IPv6 header with SRH of the packet is popped, and the packet is transmitted to the destination (E2).
The DetNet packet processing is as follows:

Ingress:

Inserts the SRv6 Policy that will steer the packet from Ingress to the destination

The methods and mechanisms used for defining, instantiating and applying the policy are outside of this document. An example of policies are described in [I-D.ietf-spring-segment-routing-policy]

Flow Identification and Sequence Number are carried in the SRH.

Relay Node 1 (Replication Node):

Replicates the payload and IPv6 Header with the SRH. This is a new function in the context of SRv6 Network Programming which will associate a given SID to a replication instruction in the node originating and advertising the SID. The replication instruction includes:

- The removal of the existing IPv6+SRH header
- The encapsulation into a new outer IPv6+SRH header. Each packet (the original and the duplicated) are encapsulated into respectively new outer IPv6+SRH headers.

Binding two different SRv6 Policies respectively to the original packet and the replicated packet, which can steer the packets from Relay Node 1 to Relay Node 2 through two tunnels.

Relay Node 2 (Elimination Node):

Eliminates the redundant packets.

Binds a new SRv6 Policy to the survival packet, which steers the packet from Relay Node 2 to Egress.

Egress:

Decapsulates the outer IPv6 header.

Sends the inter packet to the End Station 2.

The DetNet packet encapsulation is illustrated here below. It has to be noted that, in the example below, the R2 address is a SRH SID
associated to a TBD function related to the packet replication the node R1 has to perform. The same (or reverse) apply to node R2 which is in charge of the discard of the duplicated packet. Here also a new function will have a new SID allocated to it and representing the delete of the duplication in R2.

End Station1 output packet: (E1,E2)
Ingress output packet: (In, T1)(R1,T1, SL=2)(E1,E2)
Transit Node1 output packet: (In, R1)(R1,T1,SL=1)(E1,E2)
Relay Node1 output packets : (R1,T2)(R2,T2,SL=2)(E1,E2), (R1,T3)(R2,T3,SL=2)(E1,E2)
Transit Node2 output packet: (R1, R2)(R2,T2,SL=1)(E1,E2)
Transit Node3 output packet: (R1, R2)(R2,T3,SL=1)(E1,E2)
Relay Node2 output packet: (R2, T4)(Eg,T4,SL=2)(E1,E2)
Transit Node4 output packet: (R2, Eg)(Eg,T4,SL=1)(E1,E2)
Egress out : (E1,E2)

4. SRv6 DetNet Data Plane Solution Considerations

To carry DetNet over SRv6, the following elements are required:

1. A method of identifying the SRv6 payload type;

2. A suitable explicit path to deliver the DetNet flow;

3. A method of indicating packet processing, such as PREOF(Packet Replication, Elimination and Ordering as defined in [I-D.ietf-detnet-architecture]);

4. A method of identifying the DetNet flow;

5. A method of carrying DetNet sequence number;

6. A method of carrying queuing and forwarding indication to do congestion protection;

In this design, DetNet flows are encapsulated in an outer IPv6+SRH header at the Ingress Node. The SR policy identified in the SRH steers the DetNet flow along a selected path. The explicit path followed by a DetNet flow, which protect it from temporary...
interruptions caused by the convergence of routing, is encoded within the SID list of the SR policy. The network device inside the DetNet domain forwards the packet according to IPv6 Destination Address (DA), and the IPv6 DA is updated with the SID List according to SRv6 forwarding procedures defined in [I-D.ietf-6man-segment-routing-header] and [I-D.filsfils-spring-srv6-network-programming].

With SRv6 network programming, the SID list can also give instruments representing a function to be called at the node in the DetNet domain. Therefore DetNet specific functions defined in [I-D.ietf-detnet-architecture], corresponding to local packet processing in the network, can also be implemented by SRv6. New functions associated with SIDs for DetNet are defined in this document.

This document describes how DetNet flows are encapsulated/identified, and how functions of Packet Replication/Elimination/Ordering are implemented in an SRv6 domain. Congestion protection is also in the scope of this document.

Editor: This version only covers the functions of service protection and the congestion protection considerations will be added in the following versions.

5. SRv6 DetNet Data Plane Solution for Service Sub-layer

This section defines options of SRv6 data plane solution to support DetNet Service Sub-layer.

5.1. TLV Based SRv6 Data Plane Solution

5.1.1. Encapsulation

An SRv6 Segment is a 128-bit value. SID is used as a shorter reference for "SRv6 Segment Identifier" or "SRV6 Segment". SRv6 SID can also be represented as LOC:FUNCT, where:

LOC, means "LOCATION" and defines the node associated with the SID (i.e.: represented by the SID).

FUNCT, means "FUNCTION", and identifies the processing that the node specified in LOC applies to the packet. See [I-D.filsfils-spring-srv6-network-programming] for details on SRV6 Network Programming.
The SRH for DetNet in the outer IPv6 header is showed as follows, according to [I-D.ietf-6man-segment-routing-header] and [I-D.filsfils-spring-srv6-network-programming]:

```
0                   1                   2                   3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Hdr Ext Len |  Routing Type |  Segment Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Last Entry  |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Location & Function                       |
| (Segment List[0] for relay node or edge node)                 |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                     ...       |
|                                                               |
|                                                                 |
|                     Segment List[n]                           |
|                                                               |
|                                                               |
|                                                               |
|                                                                 |
|                                                               |
|                                       Optional TLVs          |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                                                      ...       |
|                                                                 |
|                                                                 |
|                                                                 |
|                                                                 |
|                                                                 |

The SRH specification allows the use of optional TLVs. Two new TLVs are defined to support DetNet service protection. DetNet Flow Identification TLV is used to uniquely identify a DetNet flow in an SRv6 DetNet node. DetNet sequence number is used to discriminate packets in the same DetNet flow. They are defined as follows:

```
0                   1                   2                   3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |           RESERVED            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        RESERVED       |           Flow Identification         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

- Type: 8bits, to be assigned by IANA.
- Length: 8 octets.
o RESERVED: 28 bits, MUST be 0 on transmission and ignored on receipt.

o Flow Identification: 20 bits, which is used for identifying DetNet flow.

```
+----------------+-----------------+-----------------+
|     Type      |    Length     |           RESERVED            |
+----------------+-----------------+-----------------+
|RESERVD|                    Sequence Number                    |
+----------------+-----------------+-----------------+
```

where:

o Type: 8 bits, to be assigned by IANA.

o Length: 8.

o RESERVED: 20 bits. MUST be 0 on transmission and ignored on receipt.

o Sequence Number: 28 bits, which is used for indicating sequence number of a DetNet flow.

5.1.2. SRv6 Network Programming new Functions

New SRv6 Network Programming functions are defined as follows:

5.1.2.1. End. B.Replication Function (Inherited argument) of segment[0] of SRH n: Packet Replication Function

1. IF NH=SRH & SL>0 THEN

2. extract the DetNet TLV values from the SRH

3. create two new outer IPv6+SRH headers: IPv6-SRH-1 and IPv6-SRH-2

   Insert the policy-instructed segment lists in each newly created SRH (SRH-1 and SRH-2). Also, add the extracted DetNet TLVs into SRH-1 and SRH-2.

4. remove the incoming outer IPv6+SRH header.

5. create a duplication of the incoming packet.

6. encapsulate the original packet into the first outer IPv6+SRH header: (IPv6-SRH-1) (original packet)
7. encapsulate the duplicate packet into the second outer IPv6+SRH header: (IPv6-SRH-2) (duplicate packet)
8. set the IPv6 SA as the local address of this node.
9. set the IPv6 DA of IPv6-SRH-1 to the first segment of the SRv6 Policy in of SRH-1 segment list.
10. set the IPv6 DA of IPv6-SRH-2 to the first segment of the SRv6 Policy in of SRH-2 segment list.
11. ELSE
12. drop the packet

5.1.2.2. End. B. Elimination: Packet Elimination Function
1. IF NH=SRH & SL>0 & "the packet is not a redundant packet" THEN
2. do not decrement SL nor update the IPv6 DA with SRH[SL]
3. extract the value of DetNet TLVs from the SRH
4. create a new outer IPv6+SRH header
5. insert the policy-instructed segment lists in the newly created SRH and add the retrieved DetNet TLVs in the newly created SRH
6. remove the incoming outer IPv6+SRH header.
7. set the IPv6 DA to the first segment of the SRv6 Policy in the newly created SRH
8. ELSE
9. drop the packet

5.2. SID Based SRv6 Data Plane Solution

5.2.1. Encapsulation

SRv6 SID can be represented as LOC:FUNCT:ARG::, where:

LOC, means "LOCATION" and defines the node associated with the SID (i.e.: represented by the SID).

FUNCT, means "FUNCTION", and identifies the processing that the node specified in LOC applies to the packet.
ARG, means "ARGUMENTS" and provides the additional arguments for the function. New SID functions for DetNet is defined in section 5.2.2. See [I-D.filsfils-spring-srv6-network-programming] for details on SRV6 Network Programming. The SRH for DetNet in the outer IPv6 header is illustrated as follows

```
0                   1                   2                   3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Hdr Ext Len |  Routing Type |  Segment Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Last Entry  |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Location & Function                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   (Segment List[0] for relay node or edge node)             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Location & Function        |   Flow Identification  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Flow ID|               Sequence Number             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   ...                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Segment List[n]               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Optional TLVS                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

- LOCATION&FUNCTION: the 80 most significant bits that are used for routing the packet towards the LOCATION (as defined in [I-D.filsfils-spring-srv6-network-programming]);
- FLOW IDENTIFICATION: 20 bits, in the DetNet TLVs in the SRH, used for DetNet flow identification in the DetNet relay node;
- SEQUENCE NUMBER: 28 bits, in the DetNet TLVs, used for dis crime packets in the same DetNet flow;

5.2.2. Functions

New SID functions are defined as follows:
5.2.2.1. End. B. Replication: Packet Replication Function

The function is similar as that has been defined in section 5.1.2.1. The only difference is that instead of retrieving the TLV values, this function retrieves the argument.

5.2.2.2. End. B. Elimination: Packet Elimination Function

The function is similar as that has been defined in section 5.1.2.2. The only difference is that instead of retrieving the TLV values, this function retrieves the argument.

5.3. DetNet SID Based SRv6 Data Plane Solution

5.3.1. Encapsulation

A non-forwarding DetNet SID is defined to carry Flow Identification and Sequence Number.

```
+---------------+---------------+---------------+---------------+---------------+
| 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1  |
| +---------------+---------------+---------------+---------------+---------------+ |
| | Next Header | Hdr Ext Len | Routing Type | Segment Left |             |
| +---------------+---------------+---------------+---------------+---------------+ |
| | Last Entry   | Flags        | Tag          |              |             |
| +---------------+---------------+---------------+---------------+---------------+ |
| | Location & Function |
| (Segment List[0] for relay node or edge node) |
| +---------------+---------------+---------------+---------------+---------------+ |
| |                     |               |               |               |               |
| +---------------+---------------+---------------+---------------+---------------+ |
| | Segment List[n] |
| +---------------+---------------+---------------+---------------+---------------+ |
| |                     |               |               |               |               |
| +---------------+---------------+---------------+---------------+---------------+ |
| |                     |               | DetNet SID   |               |               |
| +---------------+---------------+---------------+---------------+---------------+ |
| |                     |               | Optional TLVs|               |               |
| +---------------+---------------+---------------+---------------+---------------+ |
```

5.3.2. Functions

TBD

6. SRv6 DetNet Data Plane Solution for Transport Sub-layer

TBD

7. IANA Considerations

TBD

8. Security Considerations

TBD

9. Acknowledgements

Thank you for valuable comments from James Guichard and Andrew Mails.

10. Normative References

[I-D.filsfils-spring-srv6-network-programming]

[I-D.ietf-6man-segment-routing-header]

[I-D.ietf-detnet-architecture]

[I-D.ietf-detnet-dp-sol-mpls]
[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., daniel.voyer@bell.ca, d.,
bohdanov@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,

Decraene, B., Litkowski, S., and R. Shakir, "Segment
Routing Architecture", RFC 8402, DOI 10.17487/RFC8402,

Authors’ Addresses

Xuesong Geng
Huawei

Email: gengxuesong@huawei.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Yongqing Zhu
China Telecom

Email: zhuyq@gsta.com
Abstract

This document specifies a new Hop-by-Hop IPv6 option that is used to record the minimum Path MTU along the forward path between a source to a destination host. This collects a minimum recorded MTU along the path to the destination. The value can then be communicated back to the source using the return Path MTU field in the option.

This Hop-by-Hop option is intended to be used in environments like Data Centers and on paths between Data Centers, to allow them to better take advantage of paths able to support a large Path MTU.

Status of This Memo

This Internet-Draft is submitted to IETF 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."

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 (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents
1. Introduction

This draft proposes a new Hop-by-Hop Option to be used to record the minimum MTU along the forward path between the source and destination nodes. The source node creates a packet with this Hop-by-Hop Option and fills the Reported PMTU Field in the option with the value of the MTU for the outbound link that will be used to forward the packet towards the destination.

At each subsequent hop where the option is processed, the router compares the value of the Reported PMTU in the option and the MTU of its outgoing link. If the MTU of the outgoing link is less than the Reported PMTU specified in the option, it rewrites the value in the Option Data with the smaller value. When the packet arrives at the Destination node, the Destination node can send the minimum reported PMTU value back to the Source Node using the Return PMTU field in the option.

The figure below can be used to illustrate the operation of the method. In this case, the path between the Sender and Destination nodes comprises three links, the sender has a link MTU of size MTU-S, the link between routers R1 and R2 has an MTU of size 8 KBytes, and the final link to the destination has an MTU of size MTU-D.
The scenarios are described:

Scenario 1, considers all links to have an 9000 Byte MTU and the method is supported by both routers.

Scenario 2, considers the destination link to have an MTU of 1500 Byte. This is the smallest MTU, router R2 resets the reported PMTU to 1500 Byte and this is detected by the method. Had there been another smaller MTU at a link further along the path that supports the method, the lower PMTU would also have been detected.

Scenario 3, considers the case where the router preceding the smallest link does not support the method, and the method then fails to detect the actual PMTU. These scenarios are summarized in the table below. This scenario would also arise if the PTB message was not delivered to the sender.

<table>
<thead>
<tr>
<th></th>
<th>MTU-S</th>
<th>MTU-D</th>
<th>R1</th>
<th>R2</th>
<th>Rec PMTU</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9000B</td>
<td>9000B</td>
<td>H</td>
<td>H</td>
<td>9000 B</td>
<td>Endpoints attempt to use an 9000 B PMTU.</td>
</tr>
<tr>
<td>2</td>
<td>9000B</td>
<td>1500B</td>
<td>H</td>
<td>H</td>
<td>1500 B</td>
<td>Endpoints attempt to use a 1500 B PMTU.</td>
</tr>
<tr>
<td>3</td>
<td>9000B</td>
<td>1500B</td>
<td>H</td>
<td>-</td>
<td>9000 B</td>
<td>Endpoints attempt to use an 9000 B PMTU, but need to implement a method to fall back use a 1500 B PMTU.</td>
</tr>
</tbody>
</table>

IPv6 as specified in [RFC8200] allows nodes to optionally process Hop-by-Hop headers. Specifically from Section 4:

- The Hop-by-Hop Options header is not inserted or deleted, but may be examined or processed by any node along a packet’s delivery path, until the packet reaches the node (or each of the set of
nodes, in the case of multicast) identified in the Destination Address field of the IPv6 header. The Hop-by-Hop Options header, when present, must immediately follow the IPv6 header. Its presence is indicated by the value zero in the Next Header field of the IPv6 header.

- NOTE: While [RFC2460] required that all nodes must examine and process the Hop-by-Hop Options header, it is now expected that nodes along a packet’s delivery path only examine and process the Hop-by-Hop Options header if explicitly configured to do so.

The Hop-by-Hop Option defined in this document is designed to take advantage of this property of how Hop-by-Hop options are processed. Nodes that do not support this Option SHOULD ignore them. This can mean that the value returned in the response message does not account for all links along a path.

2. Motivation and Problem Solved

The current state of Path MTU Discovery on the Internet is problematic. The problems with the mechanisms defined in [RFC8201] are known to not work well in all environments. Nodes in the middle of the network may not send ICMP Packet Too Big messages or they are rate limited to the point of not making them a useful mechanism.

This results in many connection defaulting to 1280 octets and makes it very difficult to take advantage of links with larger MTU where they exist. Applications that need to send large packets over UDP are forced to use IPv6 Fragmentation.

Transport encapsulations and network-layer tunnels reduce the PMTU available for a transport to use. For example, Network Virtualization Using Generic Routing Encapsulation (NVGRE) [RFC7637] encapsulates L2 packets in an outer IP header and does not allow IP Fragmentation.

The use of 10G Ethernet will not achieve it’s potential because the packet per second rate will exceed what most nodes can send to achieve multi-gigabit rates if the packet size limited to 1280 octets. For example, the packet per second rate required to reach wire speed on a 10G Ethernet link with 1280 octet packets is about 977K packets per second (pps), vs. 139K pps for 9,000 octet packets. A significant difference.

The purpose of this draft is to improve the situation by defining a mechanism that does not rely on nodes in the middle of the network to send ICMPv6 Packet Too Big messages, instead it provides the destination host information on the minimum Path MTU and it can send
this information back to the source host. This is expected to work better than the current RFC8201 based mechanisms.

3. 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.

4. Applicability Statements

This Hop-by-Hop Option header is intended to be used in environments such as Data Centers and on paths between Data Centers, to allow them to better take advantage of a path that is able to support a large PMTU. For example, it helps inform a sender that the path includes links that have a MTU of 9,000 Bytes. This has many performance advantages compared to the current practice of limiting packets to 1280 Bytes.

The design of the option is sufficiently simple that it could be executed on a router’s fast path. To create critical mass for this to happen will have to be a strong pull from router vendors customers. This could be the case for connections within and between Data Centers.

The method could also be useful in other environments, including the general Internet.

5. IPv6 Minimum Path MTU Hop-by-Hop Option

The Minimum Path MTU Hop-by-Hop Option has the following format:
Option Type:  

BB  00  Skip over this option and continue processing.  

C    1  Option data can change en route to the packet’s final destination.  

TTTT 11110  Experimental Option Type from [IANA-HBH].  

Length:  4  The size of each value field in Option Data field supports Path MTU values from 0 to 65,535 octets.  

Min-PMTU: n 16-bits. The minimum PMTU in octets, reflecting the smallest link MTU that the packet experienced across the path. This is called the Reported PMTU. A value less than the IPv6 minimum link MTU [RFC8200] should be ignored.  

Rtn-PMTU: n 15-bits. The returned minimum PMTU, carrying the 15 most significant bits of the latest received Min-PMTU field. The value zero means that no Reported MTU is being returned.  

R    n  1-bit. R-Flag. Set by the source to signal that the destination should include the received Reported PMTU in Rtn-PMTU field.  

NOTE: The encoding of the final two octets (Rtn-PMTU and R-Flag) could be implemented by a mask of the latest received Min-MTU value with 0xFFFF, discarding the right-most bit and then performing a logical ‘OR’ with the R-Flag value of the sender.  

6. Router, Host, and Transport Behaviors  

6.1. Router Behaviour  

Routers that do not support Hop-by-Hop options SHOULD ignore this option and SHOULD forward the packet.  

Routers that support Hop-by-Hop Options, but do not recognize this option SHOULD ignore the option and SHOULD forward the packet.
Routers that recognize this option SHOULD compare the Reported PMTU in the Min-PMTU field and the MTU configured for the outgoing link. If the MTU of the outgoing link is less than the Reported PMTU, the router rewrites the Reported PMTU in the Option to use the smaller value.

The router MUST ignore and not change the Rtn-PMTU field and R-Flag in the option.

Discussion:

- The design of this Hop-by-Hop Option makes it feasible to be implemented within the fast path of a router, because the required processing is simple.

6.2. Host Behavior

The source host that supports this option SHOULD create a packet with this Hop-by-Hop Option and fill the Min-PMTU field of the option with the MTU of configured for the link over which it will send the packet on the next hop towards the destination.

The source host may request that the destination host return the received minimum MTU value by setting the R-Flag in the option. This will cause the destination host to include a PMTU option in an outgoing packet.

Discussion:

- This option does not need to be sent in all packets belonging to a flow. A transport protocol (or packetization layer) can set this option only on specific packets used to test the path.

- In the case of TCP, the option could be included in packets carrying a SYN segment as part of the connection set up, or can periodically be sent in packets carrying other segments. Including this packet in a SYN could increase the probability that SYN segment is lost, when routers on the path drop packets with this option. Including this option in a large packet is not likely to be useful, since the large packet might itself also be dropped by a link along the path with a smaller MTU, preventing the Reported PMTU information from reaching the Destination node.

- The use with datagram transport protocols (e.g. UDP) is harder to characterize because applications using datagram transports range from very short-lived (low data-volume applications) exchanges, to longer (bulk) exchanges of packets between the Source and Destination nodes [RFC8085].
o For applications that use Anycast, this option should be included in all packets as the actual destination will vary due to the nature of Anycast.

o Simple-exchange protocols (i.e. low data-volume applications [RFC8085] that only send one or a few packets per transaction, could be optimized by assuming that the Path MTU is symmetrical, that is where the Path MTU is the same in both directions, or at least not smaller in the return path. This optimisation does not hold when the paths are not symmetric.

o The use of this option with DNS and DNSSEC over UDP ought to work as long as the paths are symmetric. The DNS server will learn the Path MTU from the DNS query messages. If the return Path MTU is smaller, then the large DNSSEC response may be dropped and the known problems with PMTUD will occur. DNS and DNSSEC over transport protocols that can carry the Path MTU should work.

The Source Host can request the destination host to send a packet carrying the PMTU Option using the R-Flag.

A Destination Host SHOULD respond to each packet received with the R-Flag set, by setting the PMTU Option in the next packet that it sends to the Source Host by the same upper layer protocol instance.

The upper layer protocol MAY generate a packet when any of these conditions is met when the R Flag is set in the PMTU Option and either:

o It is the first Reported PMTU value it has received from the Source.

o The Reported PMTU value is lower than previously received.

The R-Flag SHOULD NOT be set when the PMTU Option was sent solely to carry the feedback of a Reported PMTU.

The PMTU Option sent back to the source SHOULD contain the outgoing link MTU in Min-PMTU field and SHOULD set the last Received PMTU in the Rtn-PMTU field. If these values are not present the field MUST be set to zero.

For a connection-oriented upper layer protocol, this could be implemented by saving the value of the last received option within the connection context. This last received value is then used to set the return Path MTU field for all packets belonging to this flow that carry the IPv6 Minimum Path MTU Hop-by-Hop Option.
A connection-less protocol, e.g., based on UDP, requires the application to be updated to cache the Received PMTU value, and to ensure that this corresponding value is used to set the last Received PMTU in the Rtn-PMTU field of any PMTU Option that it sends.

NOTE: The Rtn-PMTU value is specific to the instance of the upper layer protocol (i.e. matching the IPv6 flow ID, port-fields in UDP or the SPI in IPsec, etc), not the protocol itself, because network devices can make forwarding decisions that impact the PMTU based on the presence and values of these upper layer fields, and therefore these fields need to correspond to those of the packets for the flow received by the Destination Host set to ensure feedback is provided to the corresponding Source Host.

NOTE: An upper layer protocol that send packets from the Destination Host towards the Source Host less frequently than the Destination Host receives packets from the Source Host, provides less frequent feedback of the received Min-PMTU value. However, it will always needs to send the most recent value.

Discussion:

- A simple mechanism could only send an MTU Option with the Rtn-PMTU field filled in the first time this option is received or when the Received PMTU is reduced. This is good because it limits the number sent, but there is no provision for retransmission of the PMTU Option fails to reach the sender, or the sender looses state.

- The Reported PMTU value could increase or decrease over time. For instance, it would increase when the path changes and the packets become then forwarded over a link with a MTU larger than the link previously used.

6.3. Transport Behavior

A transport endpoint using this option needs to use a method to verify the information provided by this option.

The Received PMTU does not necessarily reflect the actual PMTU between the sender and destination. Care therefore needs to be exercised in using this value at the sender. Specifically:

- If the Received PMTU value returned by the Destination is the same as the initial Reported PMTU value, there could still be a router or layer 2 device on the path that does not support this PMTU. The usable PMTU therefore needs to be confirmed.
If the Received PMTU value returned by the Destination is smaller than the initial Reported PMTU value, this is an indication that there is at least one router in the path with a smaller MTU. There could still be another router or layer 2 device on the path that does not support this MTU.

If the Received PMTU value returned by the Destination is larger than the initial Reported PMTU value, this may be a corrupted, delayed or mis-ordered response, and SHOULD be ignored.

A sender needs to discriminate between the Received PMTU value in a PTB message generated in response to a Hop-by-Hop option requesting this, and a PTB message received from a router on the path.

A PMTUD or PLPMTUD method could use the Received PMTU value as an initial target size to probe the path. This can significantly decrease the number of probe attempts (and hence time taken) to arrive at a workable PMTU. It has the potential to complete discovery of the correct value in a single Round Trip Time (RTT), even over paths that may have successive links configured with lower MTUs.

Since the method can delay notification of an increase in the actual PMTU, a sender with a link MTU larger than the current PMTU SHOULD periodically probe for a PMTU value that is larger than the Received PMTU value. This specification does not define an interval for the time between probes.

Since the option consumes less capacity than an a full probe packet, there may be advantage in using this to detect a change in the path characteristics.

NOTE: Further details to be included in next version.

NOTE: A future version of the document will consider more the impact of Equal Cost Multipath (ECMP). Specifically, whether a Received PMTU value should be maintained by the method for each transport endpoint, or for each network address, and how these are best used by methods such as PLPMTUD or DPLPMTUD.

7. IANA Considerations

No IANA assignments are requested. Document uses experimental option from [IANA–HBH].
8. Security Considerations

The method has no way to protect the destination from off-path attack using this option in packets that do not originate from the source. This attack could be used to inflate or reduce the size of the reported PMTU. Mechanisms to provide this protection can be provided at a higher layer (e.g., the transport packetization layer using PLPMTUD or DPLPMTUD), where more information is available about the size of packet that has successfully traversed a path.

The method solicits a response from the destination, which should be used to generate a response to the IPv6 node originating the option packet. A malicious attacker could generate a packet to the destination for a previously inactive flow or one that advertises a change in the size of the MTU for an active flow. This would create additional work at the destination, and could induce creation of state when a new flow is created. It could potentially result in additional traffic on the return path to the sender, which could be mitigated by limiting the rate at which responses are generated.

A sender MUST check the quoted packet within the PTB message to validate that the message is in response to a packet that was originated by the sender. This is intended to provide protection against off-path insertion of ICMP PTB messages by an attacker trying to disrupt the service. Messages that fail this check MAY be logged, but the information they contain MUST be discarded.

TBD

9. Acknowledgments

A somewhat similar mechanism was proposed for IPv4 in 1988 in [RFC1063] by Jeff Mogul, C. Kent, Craig Partridge, and Keith McCloghire. It was later obsoleted in 1990 by [RFC1191] the current deployed approach to Path MTU Discovery.

Helpful comments were received from Tom Herbert, Tom Jones, Fred Templin, Ole Troan, [Your name here], and other members of the 6MAN working group.

10. Change log [RFC Editor: Please remove]

draft-hinden-6man-mtu-option-02, 2019-July-5

  o Changed option format to also include the Returned MTU value and Return flag and made related text changes in Section 6.2 to describe this behaviour.
Internet-Draft               Path MTU Option                   July 2019

o ICMP Packet Too Big messages are no longer used for feedback to
  the Source host.
o Added to Acknowledgements Section that a similar mechanism was
  proposed for IPv4 in 1988 in [RFC1063].
o Editorial changes.

draft-hinden-6man-mtu-option-01, 2019-March-05

o Changed requested status from Standards Track to Experimental to
  allow use of experimental option type (11110) to allow for
  experimentation. Removed request for IANA Option assignment.
o Added Section 2 "Motivation and Problem Solved" section to better
  describe what the purpose of this document is.
o Added Appendix A describing planned experiments and how the
  results will be measured.
o Editorial changes.

draft-hinden-6man-mtu-option-00, 2018-Oct-16

o Initial draft.

11. References

11.1. Normative References

[IANA-HBH] "Destination Options and Hop-by-Hop Options",
  <https://www.iana.org/assignments/ipv6-parameters/
  ipv6-parameters.xhtml#ipv6-parameters-2>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
  Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/
  rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
  2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,

  rfc8200>.

  "Path MTU Discovery for IP version 6", STD 87, RFC 8201,
  editor.org/info/rfc8201>.
11.2. Informative References


Appendix A. Planned Experiments

TBD

This section will describe a set of experiments planned for the use of the option defined in this document. There are many aspects of the design that require experimental data or experience to evaluate this experimental specification.

This includes experiments to understand the pathology of packets sent with the specified option to determine the likelihood that they are lost within specific types of network segment.

This includes consideration of the cost and alternatives for providing the feedback required by the mechanism and how to effectively limit the rate of transmission.

This includes consideration of the potential for integration in frameworks such as that offered by DPLPMTUD.

There are also security-related topics to be understood as described in the Security Considerations (Section 8).
Authors’ Addresses

Robert M. Hinden
Check Point Software
959 Skyway Road
San Carlos, CA  94070
USA
Email: bob.hinden@gmail.com

Godred Fairhurst
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen  AB24 3UE
UK
Email: gorry@erg.abdn.ac.uk
ICMPv6 errors for discarding packets due to processing limits
draft-ietf-6man-icmp-limits-04

Abstract

Network nodes may discard packets if they are unable to process
protocol headers of packets due to processing constraints or limits.
When such packets are dropped, the sender receives no indication so
it cannot take action to address the cause of discarded packets. This
specification defines ICMPv6 errors that can be sent by a node that
discards packets because it is unable to process the protocol
headers. A node that receives such an ICMPv6 error may be able to
modify what it sends in future packets to avoid subsequent packet
discards.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering
Task Force (IETF), its areas, and its working groups. Note that
other groups may also distribute working documents as
Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at
http://www.ietf.org/shadow.html

Copyright and License Notice

Copyright (c) 2019 IETF Trust and the persons identified as the
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 ........................................... 4
1.1 Extension header limits ............................... 4
1.2 Aggregate header limits .............................. 5
2  ICMPv6 errors for extension header limits ............ 5
2.1 Format .................................................. 5
2.2 Unrecognized Next Header type encountered (code 1) ... 6
2.3 Extension header too big (code 4) .................... 6
2.4 Extension header chain too long (code 5) ............ 7
2.5 Too many options in extension header (code 6) ...... 7
2.6 Option too big (code 7) .............................. 7
3  ICMPv6 error for aggregate header limits ............. 8
3.1 Format .................................................. 8
3.2 Usage ................................................... 9
4  Operation ............................................... 9
4.1 Priority of reporting .................................. 9
4.2 Host response .......................................... 10
5  Applicability and use cases ............................ 11
5.1 Nonconformant packet discard .......................... 11
5.2 Reliability of ICMP ................................... 11
5.3 Processing limits .................................... 11
  5.3.1 Long headers and header chains .................. 11
  5.3.2 At end nodes ...................................... 12
  5.3.3 At intermediate nodes ............................ 12
6  Security Considerations ................................ 12
7  IANA Considerations ................................... 14
7.1 Parameter Problem codes ................................ 14
7.2 Destination Unreachable codes ......................... 14
8  Acknowledgments ....................................... 14
9  References ............................................. 14
  9.1 Normative References ............................... 14
  9.2 Informative References ............................. 15
Author’s Address ........................................... 15
1 Introduction

This document specifies ICMPv6 errors that can be sent when a node discards a packet due to it being unable to process the necessary protocol headers because of processing constraints or limits. New ICMPv6 code points are defined as an update to [RFC4443]. Five of the errors are specific to processing of extension headers; another error is used when the aggregate protocol headers in a packet exceed the processing limits of a node.

1.1 Extension header limits

In IPv6, optional internet-layer information is carried in one or more IPv6 Extension Headers [RFC8200]. Extension Headers are placed between the IPv6 header and the Upper-Layer Header in a packet. The term "Header Chain" refers collectively to the IPv6 header, Extension Headers, and Upper-Layer Headers occurring in a packet. Individual extension headers may have a length of 2048 octets and must fit into one MTU. Destination Options and Hop-by-Hop Options contain a list of options in Type-length-value (TLV) format. Each option includes a length of the data field in octets; the minimum size of an option (non-pad type) is two octets and the maximum size is 257 octets. The number of options in an extension header is only limited by the length of the extension header and MTU. Options may be skipped over by a receiver if they are unknown and the Option Type indicates to skip (first two high order bits are 00).

Per [RFC8200], except for Hop by Hop options, extension headers are not examined or processed by intermediate nodes. Many intermediate nodes, however, do examine extension header for various purposes. For instance, a node may examine all extension headers to locate the transport header of a packet in order to implement transport layer filtering or to track connections to implement a stateful firewall.

Destination hosts are expected to process all extension headers and options in Hop-by-Hop and Destination Options.

Due to the variable lengths, high maximum lengths, or potential for Denial of Service attack of extension headers, many devices impose operational limits on extension headers in packets they process. [RFC7045] discusses the requirements of intermediate nodes that discard packets because of unrecognized extension headers. [RFC8504] discusses limits that may be applied to the number of options in Hop-by-Hop or Destination Options extension headers. Both intermediate nodes and end hosts may apply limits to extension header processing. When a limit is exceeded, the typical behavior is to silently discard the packet.
This specification defines four Parameter Problem codes and extends the applicability of an existing code that may be sent by a node that discards a packet due to processing limits of extension headers being exceeded. A source host that receives an ICMPv6 error may modify its use of extension headers in subsequent packets sent to the destination in order to avoid further occurrences of packets being discarded.

1.2 Aggregate header limits

Many hardware devices implement a parsing buffer of a fixed size to process packets. The parsing buffer is expected to contain all the headers (often up to a transport layer header for filtering) that a device needs to examine. If the aggregate length of headers in a packet exceeds the size of the parsing buffer, a device will either discard the packet or defer processing to a software slow path. In any case, no indication of a problem is sent back to the sender.

This document defines one code for ICMPv6 Destination Unreachable that is sent by a node that is unable to process the headers of a packet due to the aggregate size of the packet headers exceeding a processing limit. A source host that receives an ICMPv6 error can modify the headers used in subsequent packets to try to avoid further occurrences of packets being discarded or relegated to a slow path.

2 ICMPv6 errors for extension header limits

Four new codes are defined for the Parameter Problem type and applicability of one existing code is extended for ICMPv6 errors for extension header limits.

2.1 Format

The format of the ICMPv6 message for an extension header limit exceeded error is:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-------------------------------------------+------------------+--
|     Type      |     Code      |          Checksum             |
+-------------------------------------------+------------------+--
|                            Pointer                            |
+-------------------------------------------+------------------+--
|                    As much of invoking packet                 |
|               as possible without the ICMPv6 packet           |
|               exceeding the minimum IPv6 MTU [IPv6]           |
```
IPv6 Fields:

Destination Address
Copied from the Source Address field of the invoking packet.

ICMPv6 Fields:

Type
4 (Parameter Problem type)

Code (pertinent to this specification)
  1 - Unrecognized Next Header type encountered
  4 - Extension header too big
  5 - Extension header chain too long
  6 - Too many options in extension header
  7 - Option too big

Pointer
Identifies the octet offset within the invoking packet where the problem occurred.

The pointer will point beyond the end of the ICMPv6 packet if the field having a problem is beyond what can fit in the maximum size of an ICMPv6 error message.

2.2 Unrecognized Next Header type encountered (code 1)

[RFC8200] specifies that a destination host should send an "unrecognized next header type" when a Next Header value is unrecognized in a packet. This document extends this to allow intermediate nodes to send this same error for a packet that is discarded because the node does not recognize a Next Header type.

This code SHOULD be sent by an intermediate node that discards a packet because it encounters a Next Header type that is unknown in its examination. The ICMPv6 Pointer field is set to the offset of the unrecognized next header value within the original packet.

Note that when the original sender receives the ICMPv6 error it can differentiate between the message being sent by a destination host, per [RFC4443], and an error sent by an intermediate host based on matching the source address of the ICMPv6 packet and the destination address of the packet in the ICMPv6 data.

2.3 Extension header too big (code 4)

An ICMPv6 Parameter Problem with code for "extension header too big" SHOULD be sent when a node discards a packet because the size of an
extension header exceeds its processing limit. The ICMPv6 Pointer field is set to the offset of the first octet in the extension header that exceeds the limit.

2.4 Extension header chain too long (code 5)

An ICMPv6 Parameter Problem with code for "extension header chain too long" SHOULD be sent when a node discards a packet with an extension header chain that exceeds its processing limits.

There are two different limits that might be applied: a limit on the total size in octets of the header chain, and a limit on the number of extension headers in the chain. This error code is used in both cases. In the case that the size limit is exceeded, the ICMPv6 Pointer is set to first octet beyond the limit. In the case that the number of extension headers is exceeded, the ICMPv6 Pointer is set to the offset of first octet of the first extension header that is beyond the limit.

2.5 Too many options in extension header (code 6)

An ICMPv6 Parameter Problem with code for "too many options in extension header" SHOULD be sent when a node discards a packet with an extension header that has a number of options that exceed the processing limits of the node. This code is applicable for Destination options and Hop-by-Hop options. The ICMPv6 Pointer field is set to the first octet of the first option that exceeds the limit.

2.6 Option too big (code 7)

An ICMPv6 Parameter Problem with code for "option too big" is sent in two different cases: when the length of an individual option exceeds a limit, or when the length or number of consecutive padding options exceeds a limit.

If a packet is discarded because the length of a Hop-by-Hop or Destination option exceeds a processing limit, a node SHOULD send an ICMPv6 Parameter Problem with code equal to 7. The ICMPv6 Pointer field is set to the offset of the first octet of the option that exceeds the limit.

If a packet is discarded because the length or number of consecutive padding options (PAD1 and PADN) exceeds a limit, a node SHOULD send and an ICMPv6 Parameter Problem with code equal to 7. The ICMPv6 Pointer field is set to the offset of first octet of the padding option that exceeds the limit.
Possible limits related to padding include:

* The number of consecutive PAD1 options in destination options or hop-by-hop options is limited to seven octets [RFC8504].

* The length of a PADN options in destination options or hop-by-hop options is limited seven octets [RFC8504].

* The aggregate length of a set of consecutive PAD1 or PADN options in destination options or hop-by-hop options is limited to seven octets.

3 ICMPv6 error for aggregate header limits

One code is defined for Destination Unreachable type for aggregate header limits.

3.1 Format

The error for aggregate header limits employs a multi-part ICMPv6 message format as defined in [RFC4884]. The extended structure contains a pointer to the first octet beyond the limit.

The format of the ICMPv6 message for an aggregate header limit exceeded is:

```
+---------------------------------------------+--------------------------+
|                                Type      |       Code      |          Checksum             |
+---------------------------------------------+--------------------------+---------------------------------+
|                                unused    |    Length     |           unused              |
+---------------------------------------------+--------------------------+---------------------------------+
|                                      Internet Header + leading octets of original datagram   |
|                                                                                       |
|                                                                                       |
|                                                                                       |
|                                                                                       |
|                                                                                       |
|                            //                                                             |
+---------------------------------------------+--------------------------+
|                                Pointer                                             |
+---------------------------------------------+--------------------------+
```

IPv6 Fields:

Destination Address
Copied from the Source Address field of the invoking packet.
ICMPv6 Fields:

Type
  1 (Destination Unreachable type)

Code (pertinent to this specification)
  8 - Headers too long

Length
  Length of the "original datagram" measured in 64 bit words

Pointer
  Identifies the octet offset within the invoking packet where a limit was exceeded.
  The pointer will point beyond the end of the original datagram if the field exceeding the limit is beyond what can fit in the maximum size of an ICMPv6 error message.

3.2 Usage

An ICMPv6 Destination Unreachable error with code for "headers too long" SHOULD be sent when a node discards a packet because the aggregate length of headers in the packet exceeds the processing limits of the node. The Pointer in the extended ICMPv6 structure is set to the offset of the first octet that exceeds the limit.

4 Operation

Nodes that send or receive ICMPv6 errors due to header processing limits MUST generally comply with ICMPv6 processing as specified in [RFC4443].

4.1 Priority of reporting

More than one ICMPv6 error may be applicable to report for a packet. For instance, the number of extension headers in a packet might exceed a limit and the aggregate length of protocol headers might also exceed a limit. Only one ICMPv6 error SHOULD be sent for a packet, so a priority is defined to determine which error to report.

The RECOMMENDED reporting priority of ICMPv6 errors for processing limits is from highest to lowest priority:

1) Real error (existing codes)

2) "Unrecognized Next Header type" encountered by an intermediate node
3) "Extension header too big"
4) "Option too big" for length or number of consecutive padding options exceeding a limit
5) "Option too big" for the length of an option exceeding a limit
6) "Too many options in an extension header"
7) "Extension header chain too long" for number of extension headers exceeding a limit
8) "Extension header chain too long" for size of an extension header chain exceeding a limit
9) "Headers too long"

4.2 Host response

When a source host receives an ICMPv6 error for a processing limit being exceeded, it SHOULD verify the ICMPv6 error is valid and take an appropriate action.

The general validations for ICMP as described in [RFC4443] are applicable. The packet in the ICMP data SHOULD be validated to match the upper layer process or connection that generated the original packet. Other validation checks that are specific to the upper layers may be performed and are out of the scope of this specification.

The ICMPv6 error SHOULD be logged with sufficient detail for debugging packet loss. The details of the error, including the addresses and the offending extension header or data, should be retained. This, for instance, would be useful for debugging when a node is mis-configured and unexpectedly discarding packets, or when a new extension header is being deployed.

A host MAY modify its usage of protocol headers in subsequent packets to avoid repeated occurrences of the same error.

For ICMPv6 errors caused by extension header limits being exceeded:

* An error SHOULD be reported to an application if the application enabled extension headers for its traffic. In response, the application may terminate communications if extension headers are required, stop using extension headers in packets to the destination indicated by the ICMPv6 error, or attempt modify its use of extension headers or headers to avoid further packet discards.
* A host system SHOULD take appropriate action if it is automatically inserting extension headers into packets on behalf of the application. If the offending extension header is not required for communication, the host may either stop sending it or otherwise modify its use in subsequent packets sent to the destination indicated in the ICMPv6 error.

5 Applicability and use cases

5.1 Nonconformant packet discard

The ICMP errors defined in this specification may be applicable to scenarios for which a node is dropping packets outside the auspices of any standard specification. For instance, an intermediate node might send a "Headers too long" code in the case that it drops a packet because it is unable to parse deep enough to extract transport layer information needed for packet filtering. Such behavior might be considered nonconformant (with respect to [RFC8200] for instance).

This specification does not advocate behaviors that might be considered nonconformant. However, packet discard does occur in real deployments and the intent of this specification is provide visibility as to why packets are being discarded. In the spirit that providing some reason is better than silent drop, this specification RECOMMENDS the sending of ICMP errors even in cases where a node might be discarding packets per a nonconformant behavior.

5.2 Reliability of ICMP

ICMP is fundamentally an unreliable protocol and in real deployment it may consistently fail over some paths. As with any other use of ICMP, it is assumed that the errors defined in this document are only best effort to be delivered. No protocol should be implemented that relies on reliable delivery of ICMP messages. If necessary, alternative or additional mechanisms may used to augment the processes used to to deduce the reason that packets are being discarded. Such alternative mechanisms are out of scope of this specification.

5.3 Processing limits

This sections discusses the trends and motivations of processing limits that warrant ICMP errors.

5.3.1 Long headers and header chains

Historically, packet headers have been relatively simple and straightforward. For instance, the majority of packets in the
Internet are plain TCP or UDP carried in IPv4 or IPv6. The trend towards more complex headers, and hence the need to process longer headers, is driven by:

* Increasing prevalence of deep packet inspection in middleboxes. In particular, many intermediate nodes now parse into network layer encapsulation protocols.

* Deployment of routing headers. For instance, [SRH] defines an extension header format that includes a list of IPv6 addresses which may consume a considerable number of bytes.

* Development of In-situ OAM headers that allow a rich set of measurements to be gathered in the data path at the cost of additional header overhead which may be significant [IOAM].

* Other emerging use cases of Hop-by-Hop options.

5.3.2 At end nodes

End node hosts may implement limits on processing extension headers as described in [RFC8504]. Host implementations are usually software stacks that typically don’t have inherent processing limitations. Limits imposed by a software stack are more likely to be for denial of service mitigation or performance.

5.3.3 At intermediate nodes

Hardware devices that process packet headers may have limits as to how many headers or bytes of headers they can process. For instance, a middlebox hardware implementation might have a parsing buffer that contains some number of bytes of packet headers to process. Parsing buffers typically have a fixed size such as sixty-four, 128, or 256 bytes. In addition, hardware implementations (and some software implementations) often don’t have loop constructs. So for instance, processing of a TLV list might be implemented as an unrolled loop so that the number of TLVs that can be processed is limited. For instance, an implementation might unroll a TLV parsing loop to process at most eight TLVs.

6 Security Considerations

The security considerations for ICMPv6 described in [RFC4443] are applicable. The ICMP errors described in this document MAY be filtered by firewalls in accordance with [RFC4890].

In some circumstances, the sending of ICMP errors might conceptually be exploited for denial of service attack or as a means to covertly
deduce processing capabilities of nodes as a precursor to denial of service attack. As such, an implementation SHOULD allow configurable policy to withhold sending of the ICMP errors described in this specification in environments where security of ICMP errors is a concern.
7  IANA Considerations

7.1 Parameter Problem codes

IANA is requested to assign the following codes for ICMPv6 type 4 
"Parameter Problem":

4 - Extension header too big
5 - Extension header chain too long
6 - Too many options in extension header
7 - Option too big

7.2 Destination Unreachable codes

IANA is requested to assign the following codes for ICMPv6 type 1 
"Destination Unreachable":

8 - Headers too long

8  Acknowledgments

The author would like to thank Ron Bonica, Bob Hinden, Nick Hilliard, 
Michael Richardson, Mark Smith, and Suresh Krishnan for their 
comments and suggestions that improved this document.

9  References

9.1  Normative References

Control Message Protocol (ICMPv6) for the Internet Protocol 
Version 6 (IPv6) Specification", RFC 4443, DOI 

(IPv6) Specification", STD 86, RFC 8200, DOI 

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of 
IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, 
9.2 Informative References


Author’s Address

Tom Herbert
Intel
Santa Clara, CA
USA

Email: tom@quantonium.net
Discovering PREF64 in Router Advertisements

draft-ietf-6man-ra-pref64-04

Abstract

This document specifies a Router Advertisement option to communicate NAT64 prefixes to clients.

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 February 12, 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.
1. Introduction

NAT64 [RFC6146] with DNS64 [RFC6147] is a widely-deployed mechanism to provide IPv4 access on IPv6-only networks. In various scenarios, the host must be aware of the NAT64 prefix in use by the network. This document specifies a Router Advertisement [RFC4861] option to communicate the NAT64 prefix to hosts.

1.1. 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].

1.2. Terminology

Pref64 (or NAT64 prefix): an IPv6 prefix used for IPv6 address synthesis [RFC6146];

NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers ([RFC6146]);

RA: Router Advertisement, a message used by IPv6 routers to advertise their presence together with various link and Internet parameters ([RFC4861]);
DNS64: a mechanism for synthesizing AAAA records from A records ([RFC6147]);

2. Use cases for communicating the NAT64 prefix to hosts

On networks employing NAT64, it is useful for hosts to know the NAT64 prefix for several reasons, including the following:

- Local DNSSEC validation. As discussed in [RFC6147] section 2, the stub resolver in the host "will try to obtain (real) AAAA RRs, and in case they are not available, the DNS64 function will synthesize AAAA RRs for internal usage." This is required in order to use DNSSEC on a NAT64 network.

- IPv4 address literals on an IPv6-only host. As described in [RFC8305] section 7.1, IPv6-only hosts connecting to IPv4 address literals can resolve the IPv4 literal to an IPv6 address.

- 464XLAT ([RFC6877]). 464XLAT is widely deployed and requires that the host be aware of the NAT64 prefix.

- Trusted DNS server. AAAA synthesis is required for the host to be able to use a DNS server not provided by the network (e.g., a DNS-over-TLS server ([RFC7858]) with which the host has an existing trust relationship).

- Networks with no DNS64 server. Hosts that support AAAA synthesis and that are aware of the NAT64 prefix in use do not need the network to perform the DNS64 function at all.

3. Why include the NAT64 prefix in Router Advertisements

Fate sharing: NAT64 requires a routing to be configured. IPv6 routing configuration requires receiving an IPv6 Router Advertisement ([RFC4861]). Compared to currently-deployed NAT64 prefix discovery methods such as [RFC7050], including the NAT64 prefix in the Router Advertisement minimizes the number of packets required to configure a host. This speeds up the process of connecting to a network that supports NAT64/DNS64, and simplifies host implementation by removing the possibility that the host can have an incomplete layer 3 configuration (e.g., IPv6 addresses and prefixes, but no NAT64 prefix).

Updatability: it is possible to change the NAT64 prefix at any time, because when it changes, it is possible to notify hosts by sending a new Router Advertisement.
Deployability: all IPv6 hosts and networks are required to support [RFC4861]. Other options such as [RFC7225] require implementing other protocols.

4. Semantics

To support prefix lengths defined in ([RFC6052]) this option contains the prefix length field. However as /96 prefix is considered to be the most common use case, the prefix length field is optional and only presents for non-/96 prefixes. It allows to keep the option length to a minimum (16 bytes) for the most common case and increase it to 20 bytes for non-/96 prefixes only (see Section 5 below for more details).

This option specifies exactly one NAT64 prefix for all IPv4 destinations. If the network operator desires to route different parts of the IPv4 address space to different NAT64 devices, this can be accomplished by routing more specifics of the NAT64 prefix to those devices. For example, if the operator would like to route 10.0.0.0/8 through NAT64 device A and the rest of the IPv4 space through NAT64 device B, and the operator’s NAT64 prefix is 2001:db8:a:b::/96, then the operator can route 2001:db8:a:b::a00:0/104 to NAT64 A and 2001:db8:a:b::/64 to NAT64 B.

This option may appear more than once in a Router Advertisement (e.g. in case of graceful renumbering the network from one NAT64 prefix to another). Host behaviour with regards to synthesizing IPv6 addresses from IPv4 addresses SHOULD follow the recommendations given in Section 3 of [RFC7050], limited to the NAT64 prefixes that have non-zero lifetime.

In a network (or a provisioning domain) that provides both IPv4 and NAT64, it may be desirable for certain IPv4 addresses not to be translated. An example might be private address ranges that are local to the network/provisioning domain and should not be reached through the NAT64. This type of configuration cannot be conveyed to hosts using this option, or through other NAT64 prefix provisioning mechanisms such as [RFC7050] or [RFC7225]. This problem does not apply in IPv6-only networks, because in such networks, the host does not have an IPv4 address and cannot reach any IPv4 destinations without the NAT64. The multihoming and multiple provisioning domains scenarios are discussed in Section 7.

5. Option format
Figure 1: NAT64 Prefix Option Format

Fields:
Type 8-bit identifier of the Pref64 option type as assigned by IANA: TBD
Length 8-bit unsigned integer. The length of the option (including the Type and Length fields) is in units of 8 octets. If the prefix length is 96 bits the sender MUST set the Length to 2 and include the 96 bits of the prefix in the option. If the prefix length is not 96 bits then the sender MUST set the length to 3 and include all 128 bits of the prefix in the Prefix field and set the Prefix Length field to the prefix length. The receiver MUST ignore the Pref64 option if the length field value is 1. If the Length field value exceeds 3, the receiver MUST utilize the first 21 octets and ignore the rest of the option.

Lifetime 16-bit unsigned integer. The maximum time in seconds over which this NAT64 prefix MAY be used. The value of Lifetime SHOULD by default be set to lesser of 3 x MaxRtrAdvInterval or 65535 seconds. A value of zero means that the prefix MUST no longer be used.

Highest 96-bit unsigned integer. Contains bits 0 - 95 of the NAT64 prefix.

Lowest 32-bit unsigned integer. Contains bits 96 - 127 of the NAT64 prefix. This field is optional and presents only if the prefix length is not 96 bits.

Prefix Length 8-bit unsigned integer. Optional field which present only if the prefix length is not 96 bits. The sender MUST set it only to one of the following values: 32, 40, 48, 56, 64 ([RFC6052]). The receiver MUST ignore the Pref64 option if the prefix length value is not set to one of those numbers.

Reserved A 3-byte unused field. If present it MUST be initialized to zero by the sender and MUST be ignored by the receiver. This field is optional and presents only if the prefix length is not 96 bits.

6. Handling Multiple NAT64 Prefixes

In some cases a host may receive multiple NAT64 prefixes from different sources. Possible scenarios include (but are not limited to):
o the host is using multiple mechanisms to discover Pref64 prefixes (e.g. by using PCP ([RFC7225]) and/or by resolving IPv4-only fully qualified domain name ([RFC7050]) in addition to receiving the Pref64 RA option);

o The pref64 option presents in a single RA more than once;

o the host receives multiple RAs with different Pref64 prefixes on one or multiple interfaces.

When multiple Pref64 were discovered via RA Pref64 Option (the Option presents more than once in a single RA or multiple RAs were received), host behaviour with regards to synthesizing IPv6 addresses from IPv4 addresses SHOULD follow the recommendations given in Section 3 of [RFC7050], limited to the NAT64 prefixes that have non-zero lifetime..

When different Pref64 are discovered by using multiple mechanisms, hosts SHOULD select one source of information only. The RECOMMENDED order is:

o PCP-discovered prefixes ([RFC7225]), if supported;

o Pref64 discovered via RA Option;

o Pref64 resolving IPv4-only fully qualified domain name ([RFC7050])

Note that if the network provides Pref64 both via this RA option and [RFC7225], hosts that receive the Pref64 via RA option may choose to use it immediately before waiting for PCP to complete, and therefore some traffic may not reflect any more detailed configuration provided by PCP.

7. Multihoming

Like most IPv6 configuration information, the Pref64 option is specific to the network on which it is received. For example, a Pref64 option received on a particular wireless network may not be usable unless the traffic is also sourced on that network. Similarly, a host connected to a cellular network that provides NAT64 generally cannot use that NAT64 for destinations reached through a VPN tunnel that terminates outside that network.

Thus, correct use of this option on a multihomed host generally requires the host to support the concept of multiple Provisioning Domains (PvD, a set of configuration information associated with a network, [RFC7556]) and to be able to use these PvDs.
This issue is not specific to the Pref64 RA option and, for example, is quite typical for DNS resolving on multihomed hosts (e.g. a host might resolve a destination name by using the corporate DNS server via the VPN tunnel but then send the traffic via its Internet-facing interface).

8. Pref64 Consistency

Section 6.2.7 of [RFC4861] recommends that routers inspect RAs sent by other routers to ensure that all routers onlink advertise the consistent information. Routers SHOULD inspect valid Pref64 options received on a given link and verify the consistency. Detected inconsistencies indicate that one or more routers might be misconfigured. Routers SHOULD log such cases to system or network management. Routers SHOULD check and compare the following information:

- set of Pref64 with non-zero lifetime;
- set of Pref64 with zero lifetime.

PvD-aware routers MUST only compare information scoped to the same implicit or explicit PvD.

9. IANA Considerations

The IANA is requested to assign a new IPv6 Neighbor Discovery Option type for the PREF64 option defined in this document.

| +---------------+-------+ |
| Option Name    | Type   |
| +---------------+-------+ |
| PREF64 option  | (TBD)  |

Table 1

The IANA registry for these options is:

https://www.iana.org/assignments/icmpv6-parameters [1]

10. Security Considerations

Because Router Advertisements are required in all IPv6 configuration scenarios, on IPv6-only networks, Router Advertisements must already be secured, e.g., by deploying RA guard [RFC6105]. Providing all configuration in Router Advertisements increases security by ensuring
that no other protocols can be abused by malicious attackers to
provide hosts with invalid configuration.

The security measures that must already be in place to ensure that
Router Advertisements are only received from legitimate sources
eliminate the problem of NAT64 prefix validation described in section
3.1 of [RFC7050].

11. Acknowledgements

Thanks to the following people (in alphabetical order) for their
review and feedback: Mikael Abrahamsson, Mark Andrews, Brian E
Carpenter, David Farmer, Nick Heatley, Martin Hunek, Tatuya Jinmei,
Erik Kline, David Lamparter, Jordi Palet Martinez, Tommy Pauly,
Michael Richardson, David Schinazi.

12. References

12.1. 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,

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman,
"Neighbor Discovery for IP version 6 (IPv6)", RFC 4861,
DOI 10.17487/RFC4861, September 2007,

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X.
Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052,
DOI 10.17487/RFC6052, October 2010,

12.2. Informative References

[I-D.ietf-intarea-provisioning-domains]
Pfister, P., Vyncke, E., Pauly, T., Schinazi, D., and W.
Shao, "Discovering Provisioning Domain Names and Data",
draft-ietf-intarea-provisioning-domains-05 (work in
progress), June 2019.

Rose, "DNS Security Introduction and Requirements",
RFC 4033, DOI 10.17487/RFC4033, March 2005,


12.3. URIs

[1] https://www.iana.org/assignments/icmpv6-parameters

Authors’ Addresses

Lorenzo Colitti
Google
Roppongi 6-10-1
Minato, Tokyo 106-6126
JP

Email: lorenzo@google.com

Jen Linkova
Google
1 Darling Island Rd
Pyrmont, NSW 2009
AU

Email: furry@google.com
IPv6 Segment Routing Header (SRH)
draft-ietf-6man-segment-routing-header-22

Abstract

Segment Routing can be applied to the IPv6 data plane using a new type of Routing Extension Header called the Segment Routing Header. This document describes the Segment Routing Header and how it is used by Segment Routing capable nodes.

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 February 7, 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
1. Introduction

Segment Routing can be applied to the IPv6 data plane using a new type of Routing Header called the Segment Routing Header. This document describes the Segment Routing Header and how it is used by Segment Routing capable nodes.


The encoding of IPv6 segments in the Segment Routing Header is defined in this document.

This document uses the terms Segment Routing, SR Domain, SRv6, Segment ID (SID), SRv6 SID, Active Segment, and SR Policy as defined in [RFC8402].

1.1. 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.
2. Segment Routing Header

Routing Headers are defined in [RFC8200]. The Segment Routing Header has a new Routing Type (suggested value 4) to be assigned by IANA.

The Segment Routing Header (SRH) is defined as follows:

```
+-----------------------------------------------+---------+--------+----------+---------------------+
| Next Header   | Hdr Ext Len | Routing Type | Segments Left |
+-----------------------------------------------+---------+--------+----------+---------------------+
| Last Entry | Flags | Tag |
+-----------------------------------------------+---------+--------+----------+---------------------+
| Segment List[0] (128 bits IPv6 address) |
+-----------------------------------------------+---------+--------+----------+---------------------+
| |
| ... |
+-----------------------------------------------+---------+--------+----------+---------------------+
| Segment List[n] (128 bits IPv6 address) |
+-----------------------------------------------+---------+--------+----------+---------------------+

// Optional Type Length Value objects (variable)
```

where:

- **Next Header**: Defined in [RFC8200] Section 4.4
- **Hdr Ext Len**: Defined in [RFC8200] Section 4.4
- **Routing Type**: TBD, to be assigned by IANA (suggested value: 4).
- **Segments Left**: Defined in [RFC8200] Section 4.4
- **Last Entry**: contains the index (zero based), in the Segment List, of the last element of the Segment List.
- Flags: 8 bits of flags. Section 8.1 creates an IANA registry for new flags to be defined. The following flags are defined:

```
0 1 2 3 4 5 6 7
+----------+
| U U U U U U U |
+----------+

U: Unused and for future use. MUST be 0 on transmission and ignored on receipt.
```

- Tag: tag a packet as part of a class or group of packets, e.g., packets sharing the same set of properties. When tag is not used at source it MUST be set to zero on transmission. When tag is not used during SRH Processing it SHOULD be ignored. Tag is not used when processing the SID defined in Section 4.3.1. It may be used when processing other SIDs which are not defined in this document. The allocation and use of tag is outside the scope of this document.

- Segment List[n]: 128 bit IPv6 addresses representing the nth segment in the Segment List. The Segment List is encoded starting from the last segment of the SR Policy. I.e., the first element of the segment list (Segment List [0]) contains the last segment of the SR Policy, the second element contains the penultimate segment of the SR Policy and so on.

- Type Length Value (TLV) are described in Section 2.1.

In the SRH, the Next Header, Hdr Ext Len, Routing Type, and Segments Left fields are defined in Section 4.4 of [RFC8200]. Based on the constraints in that section Next Header, Header Ext Len, and Routing Type are not mutable while Segments Left is mutable.

The mutability of the TLV value is defined by the most significant bit in the type, as specified in Section 2.1.

Section 4.3 defines the mutability of the remaining fields in the SRH (Flags, Tag, Segment List) in the context of the SID defined in this document.

New SIDs defined in the future MUST specify the mutability properties of the Flags, Tag, and Segment List and indicate how the HMAC TLV (Section 2.1.2) verification works. Note, that in effect these fields are mutable.
Consistent with the source routing model, the source of the SRH always knows how to set the segment list, Flags, Tag and TLVs of the SRH for use within the SR Domain. How it achieves this is outside the scope of this document, but may be based on topology, available SIDs and their mutability properties, the SRH mutability requirements of the destination, or any other information.

2.1. SRH TLVs

This section defines TLVs of the Segment Routing Header.

A TLV provides meta-data for segment processing. The only TLVs defined in this document are the HMAC (Section 2.1.2) and PAD (Section 2.1.1) TLVs. While processing the SID defined in Section 4.3.1, all TLVs are ignored unless local configuration indicates otherwise (Section 4.3.1.1). Thus, TLV and HMAC support is optional for any implementation, however an implementation adding or parsing TLVs MUST support PAD TLVs. Other documents may define additional TLVs and processing rules for them.

TLVs are present when the Hdr Ext Len is greater than (Last Entry+1)*2.

While processing TLVs at a segment endpoint, TLVs MUST be fully contained within the SRH as determined by the Hdr Ext Len. Detection of TLVs exceeding the boundary of the SRH Hdr Ext Len results in an ICMP Parameter Problem, Code 0, message to the Source Address, pointing to the Hdr Ext Len field of the SRH, and the packet being discarded.

An implementation MAY limit the number and/or length of TLVs it processes based on local configuration. It MAY:

- Limit the number of consecutive Pad1 (Section 2.1.1.1) options to 1, if padding of more than one byte is required then PadN (Section 2.1.1.2) should be used.
- Limit the length in PadN to 5.
- Limit the maximum number of non-Pad TLVs to be processed.
- Limit the maximum length of all TLVs to be processed.

The implementation MAY stop processing additional TLVs in the SRH when these configured limits are exceeded.
Type: An 8 bit value. Unrecognized Types MUST be ignored on receipt.

Length: The length of the Variable length data.

Variable length data: Length bytes of data that is specific to the Type.

Type Length Value (TLV) contain OPTIONAL information that may be used by the node identified in the Destination Address (DA) of the packet.

Each TLV has its own length, format and semantic. The code-point allocated (by IANA) to each TLV Type defines both the format and the semantic of the information carried in the TLV. Multiple TLVs may be encoded in the same SRH.

The highest-order bit of the TLV type specifies whether or not the TLV data of that type can change en route to the packet’s final destination:

0: TLV data does not change en route
1: TLV data does change en route

All TLVs specify their alignment requirements using an xn+y format. The xn+y format is defined as per [RFC8200]. The SR Source nodes use the xn+y alignment requirements of TLVs and padding TLVs when constructing an SRH.

The "Length" field of the TLV is used to skip the TLV while inspecting the SRH in case the node doesn’t support or recognize the Type. The "Length" defines the TLV length in octets, not including the "Type" and "Length" fields.

The following TLVs are defined in this document:

Padding TLVs

HMAC TLV

Additional TLVs may be defined in the future.
2.1.1. Padding TLVs

There are two types of padding TLVs, pad1 and padN, the following applies to both:

Padding TLVs are used for meeting the alignment requirement of the subsequent TLVs.

Padding TLVs are used to pad the SRH to a multiple of 8 octets.

Padding TLVs are used for alignment.

Padding TLVs are ignored by a node processing the SRH TLV.

Multiple Padding TLVs MAY be used in one SRH

2.1.1.1. PAD1

Alignment requirement: none

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

Type: to be assigned by IANA (Suggested value 0)

A single Pad1 TLV MUST be used when a single byte of padding is required. If more than one byte of padding is required a Pad1 TLV MUST NOT be used, the PadN TLV MUST be used.

2.1.1.2. PADN

Alignment requirement: none

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type     | Length | Padding (variable) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
// Padding (variable) //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Type: to be assigned by IANA (suggested value 4).

Length: 0 to 5
Padding: Length octets of padding. Padding bits have no semantics. They MUST be set to 0 on transmission and ignored on receipt.

The PadN TLV MUST be used when more than one byte of padding is required.

2.1.2. HMAC TLV

Alignment requirement: 8n

The keyed Hashed Message Authentication Code (HMAC) TLV is OPTIONAL and has the following format:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |     Length    |          RESERVED             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      HMAC Key ID (4 octets)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                              |
|                      HMAC (32 octets)                        |
|                                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

- Type: to be assigned by IANA (suggested value 5).
- Length: 38.
- RESERVED: 2 octets. MUST be 0 on transmission and ignored on receipt.
- HMAC Key ID: A 4 octet opaque number which uniquely identifies the pre-shared key and algorithm used to generate the HMAC.
- HMAC: 32 octets of keyed HMAC.

The HMAC TLV is used to verify the source of a packet is permitted to use the current segment in the destination address of the packet, and ensure the segment list is not modified in transit.
2.1.2.1. HMAC Generation and Verification

Local configuration determines when to check for an HMAC and potentially indicates what the HMAC protects, and a requirement on where the HMAC TLV must appear (e.g. first TLV), and whether or not to verify the destination address is equal to the current segment. This local configuration is outside the scope of this document. It may be based on the active segment at an SR Segment endpoint node, the result of an ACL that considers incoming interface, HMAC Key ID, or other packet fields.

An implementation that supports the generation and verification of the HMAC SHOULD support the following default behavior as defined in the remainder of this section.

The HMAC verification begins by checking the current segment is equal to the destination address of the IPv6 header, i.e. destination address is equal to Segment List [Segments Left] and Segments Left is less than or equal to Last Segment + 1.

The HMAC field is the output of the HMAC computation as defined in [RFC2104], using:

- key: the pre-shared key identified by HMAC Key ID
- HMAC algorithm: identified by the HMAC Key ID
- Text: a concatenation of the following fields from the IPv6 header and the SRH, as it would be received at the node verifying the HMAC:
  * IPv6 header: source address (16 octets)
  * SRH: Last Entry (1 octet)
  * SRH: Flags (1 octet)
  * SRH: HMAC Key-id (4 octets)
  * SRH: all addresses in the Segment List (variable octets)

The HMAC digest is truncated to 32 octets and placed in the HMAC field of the HMAC TLV.

For HMAC algorithms producing digests less than 32 octets, the digest is placed in the lowest order octets of the HMAC field. Remaining octets MUST be set to zero.
If HMAC verification is successful, the packet is forwarded to the next segment.

If HMAC verification fails, an ICMP error message (parameter problem, error code 0, pointing to the HMAC TLV) SHOULD be generated (but rate limited) and SHOULD be logged.

2.1.2.2. HMAC Pre-Shared Key Algorithm

The HMAC Key ID field allows for the simultaneous existence of several hash algorithms (SHA-256, SHA3-256 ... or future ones) as well as pre-shared keys.

The HMAC Key ID field is opaque, i.e., it has neither syntax nor semantic except as an identifier of the right combination of pre-shared key and hash algorithm, and except that a value of 0 means that there is no HMAC field.

At the HMAC TLV verification node the Key ID uniquely identifies the pre-shared key and HMAC algorithm.

At the HMAC TLV generating node the Key ID and destination address uniquely identify the pre-shared key and HMAC algorithm. Utilizing the destination address with the Key ID allows for overlapping key IDs amongst different HMAC verification nodes. The Text for the HMAC computation is set to the IPv6 header fields and SRH fields as they would appear at the verification node, not necessarily the same as the source node sending a packet with the HMAC TLV.

Pre-shared key roll-over is supported by having two key IDs in use while the HMAC TLV generating node and verifying node converge to a new key.

An implementation supporting HMAC can support multiple hash functions. An implementation supporting HMAC MUST implement SHA-2 [FIPS180-4] in its SHA-256 variant.

The selection of pre-shared key and algorithm, and their distribution is outside the scope of this document, some options may include:

- in the configuration of the HMAC generating or verifying nodes, either by static configuration or any SDN oriented approach
- dynamically using a trusted key distribution protocol such as [RFC6407]
3. SR Nodes

There are different types of nodes that may be involved in segment routing networks: source SR nodes originate packets with a segment in the destination address of the IPv6 header, transit nodes that forward packets destined to a remote segment, and SR segment endpoint nodes that process a local segment in the destination address of an IPv6 header.

3.1. Source SR Node

A Source SR Node is any node that originates an IPv6 packet with a segment (i.e. SRv6 SID) in the destination address of the IPv6 header. The packet leaving the source SR Node may or may not contain an SRH. This includes either:

A host originating an IPv6 packet.

An SR domain ingress router encapsulating a received packet in an outer IPv6 header, followed by an optional SRH.

The mechanism through which a segment in the destination address of the IPv6 header and the Segment List in the SRH, is derived is outside the scope of this document.

3.2. Transit Node

A transit node is any node forwarding an IPv6 packet where the destination address of that packet is not locally configured as a segment nor a local interface. A transit node is not required to be capable of processing a segment nor SRH.

3.3. SR Segment Endpoint Node

A SR segment endpoint node is any node receiving an IPv6 packet where the destination address of that packet is locally configured as a segment or local interface.

4. Packet Processing

This section describes SRv6 packet processing at the SR source, Transit and SR segment endpoint nodes.

4.1. Source SR Node

A Source node steers a packet into an SR Policy. If the SR Policy results in a segment list containing a single segment, and there is
no need to add information to SRH flag or TLV, the DA is set to the
single segment list entry and the SRH MAY be omitted.

When needed, the SRH is created as follows:

Next Header and Hdr Ext Len fields are set as specified in
[RFC8200].

Routing Type field is set as TBD (to be allocated by IANA,
suggested value 4).

The DA of the packet is set with the value of the first segment.

The first element of the SRH Segment List is the ultimate segment.
The second element is the penultimate segment and so on.

The Segments Left field is set to n-1 where n is the number of
elements in the SR Policy.

The Last Entry field is set to n-1 where n is the number of
elements in the SR Policy.

HMAC TLV may be set according to Section 7.

The packet is forwarded toward the packet’s Destination Address
(the first segment).

4.1.1. Reduced SRH

When a source does not require the entire SID list to be preserved in
the SRH, a reduced SRH may be used.

A reduced SRH does not contain the first segment of the related SR
Policy (the first segment is the one already in the DA of the IPv6
header), and the Last Entry field is set to n-2 where n is the number of
elements in the SR Policy.

4.2. Transit Node

As specified in [RFC8200], the only node allowed to inspect the
Routing Extension Header (and therefore the SRH), is the node
corresponding to the DA of the packet. Any other transit node MUST
NOT inspect the underneath routing header and MUST forward the packet
toward the DA according to its IPv6 routing table.

When a SID is in the destination address of an IPv6 header of a
packet, it’s routed through an IPv6 network as an IPv6 address.
SIDs, or the prefix(es) covering SIDs, and their reachability may be
4.3. SR Segment Endpoint Node

Without constraining the details of an implementation, the SR segment endpoint node creates Forwarding Information Base (FIB) entries for its local SIDs.

When an SRv6-capable node receives an IPv6 packet, it performs a longest-prefix-match lookup on the packets destination address. This lookup can return any of the following:

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

4.3.1. FIB Entry Is Locally Instantiated SRv6 SID

This document, and section, defines a single SRv6 SID. Future documents may define additional SRv6 SIDs. In which case, the entire content of this section will be defined in that document.

If the FIB entry represents a locally instantiated SRv6 SID, process the next header chain of the IPv6 header as defined in section 4 of [RFC8200]. Section 4.3.1.1 describes how to process an SRH, Section 4.3.1.2 describes how to process an upper layer header or no next header.

Processing this SID modifies the Segments Left and, if configured to process TLVs, it may modify the "variable length data" of TLV types that change en route. Therefore Segments Left is mutable and TLVs that change en route are mutable. The remainder of the SRH (Flags, Tag, Segment List, and TLVs that do not change en route) are immutable while processing this SID.

4.3.1.1. SRH Processing
S01. When an SRH is processed {
S02.   If Segments Left is equal to zero {
S03.     Proceed to process the next header in the packet,
S04.     whose type is identified by the Next Header field in
S05.     the Routing header.
S06. } }
S07.   Else {
S08.     If local configuration requires TLV processing {
S09.       Perform TLV processing (see TLV Processing)
S10.     }
S11.     max_last_entry = (Hdr Ext Len / 2) - 1
S12.     If (Segments Left > max_last_entry) or
S13.     (Segments Left is greater than (Last Entry+1)) {
S14.       Send an ICMP Parameter Problem, Code 0, message to
S15.       the Source Address, pointing to the Segments Left
S16.       field, and discard the packet.
S17. } }
S18.   Else {
S20.     Copy Segment List[Segments Left] from the SRH to the
S21.     destination address of the IPv6 header.
S22.     If the IPv6 Hop Limit is less than or equal to 1 {
S23.       Send an ICMP Hop Limit Exceeded in
S24.       Transit message to the Source Address and discard
S25.       the packet.
S26. } }
S27.   Else {
S28.     Decrement the Hop Limit by 1
S29.     Resubmit the packet to the IPv6 module for transmission
S30.     to the new destination.
S31.   }
S32. }

4.3.1.1.1. TLV Processing

Local configuration determines how TLVs are to be processed when the
Active Segment is a local SID defined in this document. The
definition of local configuration is outside the scope of this
document.

For illustration purpose only, two example local configurations that
may be associated with a SID are provided below.
Example 1:
For any packet received from interface I2
  Skip TLV processing

Example 2:
For any packet received from interface I1
  If first TLV is HMAC {
    Process the HMAC TLV
  }
  Else {
    Discard the packet
  }

4.3.1.2. Upper-layer Header or No Next Header

When processing the Upper-layer header of a packet matching a FIB
entry locally instantiated as an SRv6 SID defined in this document.

IF (Upper-layer Header is IPv4 or IPv6) and
  local configuration permits {
  Perform IPv6 decapsulation
  Resubmit the decapsulated packet to the IPv4 or IPv6 module
}
ELSE {
  Send an ICMP parameter problem message to the Source Address and
discard the packet. Error code (TBD by IANA) "SR Upper-layer
  Header Error", pointer set to the offset of the upper-layer
  header.
}

A unique error code allows an SR Source node to recognize an error in
SID processing at an endpoint.

4.3.2. FIB Entry is a Local Interface

If the FIB entry represents a local interface, not locally
instantiated as an SRv6 SID, the SRH is processed as follows:

  If Segments Left is zero, the node must ignore the Routing header
  and proceed to process the next header in the packet, whose type
  is identified by the Next Header field in the Routing Header.

  If Segments Left is non-zero, the node must discard the packet and
  send an ICMP Parameter Problem, Code 0, message to the packet’s
  Source Address, pointing to the unrecognized Routing Type.
4.3.3. FIB Entry Is A Non-Local Route

Processing is not changed by this document.

4.3.4. FIB Entry Is A No Match

Processing is not changed by this document.

5. Intra SR Domain Deployment Model

The use of the SIDs exclusively within the SR Domain and solely for packets of the SR Domain is an important deployment model.

This enables the SR Domain to act as a single routing system.

This section covers:

- securing the SR Domain from external attempt to use its SIDs
- SR Domain as a single system with delegation between components
- handling packets of the SR Domain

5.1. Securing the SR Domain

Nodes outside the SR Domain are not trusted: they cannot directly use the SID’s of the domain. This is enforced by two levels of access control lists:

1. Any packet entering the SR Domain and destined to a SID within the SR Domain is dropped. This may be realized with the following logic, other methods with equivalent outcome are considered compliant:

   * allocate all the SID’s from a block S/s
   * configure each external interface of each edge node of the domain with an inbound infrastructure access list (IACL) which drops any incoming packet with a destination address in S/s
   * Failure to implement this method of ingress filtering exposes the SR Domain to source routing attacks as described and referenced in [RFC5095]

2. The distributed protection in #1 is complemented with per node protection, dropping packets to SIDs from source addresses outside the SR Domain. This may be realized with the following
logic, other methods with equivalent outcome are considered compliant:

* assign all interface addresses from prefix A/a

* at node k, all SIDs local to k are assigned from prefix Sk/sk

* configure each internal interface of each SR node k in the SR Domain with an inbound IACL which drops any incoming packet with a destination address in Sk/sk if the source address is not in A/a.

5.2. SR Domain as a single system with delegation among components

All intra SR Domain packets are of the SR Domain. The IPv6 header is originated by a node of the SR Domain, and is destined to a node of the SR Domain.

All inter domain packets are encapsulated for the part of the packet journey that is within the SR Domain. The outer IPv6 header is originated by a node of the SR Domain, and is destined to a node of the SR Domain.

As a consequence, any packet within the SR Domain is of the SR Domain.

The SR Domain is a system in which the operator may want to distribute or delegate different operations of the outer most header to different nodes within the system.

An operator of an SR domain may choose to delegate SRH addition to a host node within the SR domain, and validation of the contents of any SRH to a more trusted router or switch attached to the host.

Consider a top of rack switch (T) connected to host (H) via interface (I). H receives an SRH (SRH1) with a computed HMAC via some SDN method outside the scope of this document. H classifies traffic it sources and adds SRH1 to traffic requiring a specific SLA. T is configured with an IACL on I requiring verification of the SRH for any packet destined to the SID block of the SR Domain (S/s). T checks and verifies that SRH1 is valid, contains an HMAC TLV and verifies the HMAC.

An operator of the SR Domain may choose to have all segments in the SR Domain verify the HMAC. This mechanism would verify that the SRH segment list is not modified while traversing the SR Domain.
5.3. MTU Considerations

An SR Domain ingress edge node encapsulates packets traversing the SR Domain, and needs to consider the MTU of the SR Domain. Within the SR Domain, well known mitigation techniques are RECOMMENDED, such as deploying a greater MTU value within the SR Domain than at the ingress edges.

5.4. ICMP Error Processing

ICMP error packets generated within the SR Domain are sent to source nodes within the SR Domain. The invoking packet in the ICMP error message may contain an SRH. Since the destination address of a packet with an SRH changes as each segment is processed, it may not be the destination used by the socket or application that generated the invoking packet.

For the source of an invoking packet to process the ICMP error message, the correct destination address must be determined. The following logic is used to determine the destination address for use by protocol error handlers.

1. Walk all extension headers of the invoking IPv6 packet to the routing extension header preceding the upper layer header.
   * If routing header is type 4 (SRH)
     + Use the SID at Segment List[0] as the destination address of the invoking packet.

ICMP errors are then processed by upper layer transports as defined in [RFC4443].

For IP packets encapsulated in an outer IPv6 header, ICMP error handling is as defined in [RFC2473].

5.5. Load Balancing and ECMP

For any inter domain packet, the SR Source node MUST impose a flow label computed based on the inner packet. The computation of the flow label is as recommended in [RFC6438] for the sending Tunnel End Point.

For any intra domain packet, the SR Source node SHOULD impose a flow label computed as described in [RFC6437] to assist ECMP load balancing at transit nodes incapable of computing a 5-tuple beyond the SRH.
At any transit node within an SR domain, the flow label MUST be used as defined in [RFC6438] to calculate the ECMP hash toward the destination address. If flow label is not used, the transit node would likely hash all packets between a pair of SR Edge nodes to the same link.

At an SR segment endpoint node, the flow label MUST be used as defined in [RFC6438] to calculate any ECMP hash used to forward the processed packet to the next segment.

5.6. Other Deployments

Other deployment models and their implications on security, MTU, HMAC, ICMP error processing and interaction with other extension headers are outside the scope of this document.

6. Illustrations

This section provides illustrations of SRv6 packet processing at SR source, transit and SR segment endpoint nodes.

6.1. Abstract Representation of an SRH

For a node k, its IPv6 address is represented as Ak, its SRv6 SID is represented as Sk.

IPv6 headers are represented as the tuple of (source, destination). For example, a packet with source address A1 and destination address A2 is represented as (A1, A2). The payload of the packet is omitted.

An SR Policy is a list of segments. A list of segments 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.

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

- Source Address is SA, Destination Addresses is DA, and next-header is SRH.
- 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 the leftmost segment is the first segment. Whereas, (S3, S2, S1; SL) represents the same SID list but encoded in the SRH Segment List format where the leftmost segment is the last segment. 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 detailed behavior, the \((S3, S2, S1; SL)\) notation is more convenient.

At its SR Policy headend, the Segment List \(<S1,S2,S3>\) results in SRH \((S3,S2,S1; SL=2)\) represented fully as:

- Segments Left=2
- Last Entry=2
- Flags=0
- Tag=0
- Segment List[0]=S3
- Segment List[1]=S2
- Segment List[2]=S1

6.2. Example Topology

The following topology is used in examples below:

```
+ * * * * * * * * * * * * * * * * * * * * *
*                [8]                [9]            *
*                  |                  |
[1]----[3]--------[5]-----------------[6]---------[4]---[2]
*                  |                  |
|                  |
|                  |
*                  |                  |
  +--------[7]-------+                  *
  |                  |
  |                  |
+ * * * * * * SR Domain * * * * * * +
```

Figure 3

- 3 and 4 are SR Domain edge routers
- 5, 6, and 7 are all SR Domain routers
- 8 and 9 are hosts within the SR Domain
- 1 and 2 are hosts outside the SR Domain
- The SR domain is secured as per Section 5.1 and no external packet can enter the domain with a destination address equal to a segment of the domain.
6.3. Source SR Node

6.3.1. Intra SR Domain Packet

When host 8 sends a packet to host 9 via an SR Policy <S7,A9> the packet is

P1: (A8,S7)(A9,S7; SL=1)

6.3.1.1. Reduced Variant

When host 8 sends a packet to host 9 via an SR Policy <S7,A9> and it wants to use a reduced SRH, the packet is

P2: (A8,S7)(A9; SL=1)

6.3.2. Inter SR Domain Packet - Transit

When host 1 sends a packet to host 2, the packet is

P3: (A1,A2)

The SR Domain ingress router 3 receives P3 and steers it to SR Domain egress router 4 via an SR Policy <S7, S4>. Router 3 encapsulates the received packet P3 in an outer header with an SRH. The packet is

P4: (A3, S7)(S4, S7; SL=1)(A1, A2)

If the SR Policy contains only one segment (the egress router 4), the ingress Router 3 encapsulates P3 into an outer header (A3, S4). The packet is

P5: (A3, S4)(A1, A2)

6.3.2.1. Reduced Variant

The SR Domain ingress router 3 receives P3 and steers it to SR Domain egress router 4 via an SR Policy <S7, S4>. If router 3 wants to use a reduced SRH, Router 3 encapsulates the received packet P3 in an outer header with a reduced SRH. The packet is

P6: (A3, S7)(S4; SL=1)(A1, A2)

6.3.3. Inter SR Domain Packet - Internal to External

When host 8 sends a packet to host 1, the packet is encapsulated for the portion of its journey within the SR Domain. From 8 to 3 the packet is
P7: (A8,S3)(A8,A1)

In the opposite direction, the packet generated from 1 to 8 is
P8: (A1,A8)

At node 3 P8 is encapsulated for the portion of its journey within
the SR domain, with the outer header destined to segment S8.
Resulting in
P9: (A3,S8)(A1,A8)

At node 8 the outer IPv6 header is removed by S8 processing, then
processed again when received by A8.

6.4. Transit Node

Nodes 5 acts as transit nodes for packet P1, and sends packet
P1: (A8,S7)(A9,S7;SL=1)
on the interface toward node 7.

6.5. SR Segment Endpoint Node

Node 7 receives packet P1 and, using the logic in Section 4.3.1,
sends packet
P7: (A8,A9)(A9,S7;SL=0)
on the interface toward router 6.

6.6. Delegation of Function with HMAC Verification

This section describes how a function may be delegated within the SR
Domain to non SR source nodes. In the following sections consider a
host 8 connected to a top of rack 5.

6.6.1. SID List Verification

An operator may prefer to add the SRH at source 8, while 5 verifies
the SID list is valid.

For illustration purpose, an SDN controller provides 8 an SRH
terminating at node 9, with segment list <S5,S7,S6,A9>, and HMAC TLV
computed for the SRH. The HMAC key is shared with 5, node 8 does not
know the key. Node 5 is configured with an IACL applied to the
interface connected to 8, requiring HMAC verification for any packet destined to S/s.

Node 8 originates packets with the received SRH with HMAC TLV.

P15: (A8,S5)(A9,S6,S7,S5;SL=3;HMAC)

Node 5 receives and verifies the HMAC for the SRH, then forwards the packet to the next segment

P16: (A8,S7)(A9,S6,S7,S5;SL=2;HMAC)

Node 6 receives

P17: (A8,S6)(A9,S6,S7,S5;SL=1;HMAC)

Node 9 receives

P18: (A8,A9)(A9,S6,S7,S5;SL=0;HMAC)

This use of an HMAC is particularly valuable within an enterprise based SR Domain [SRN].

7. Security Considerations

This section reviews security considerations related to the SRH, given the SRH processing and deployment models discussed in this document.

As described in Section 5, it is necessary to filter packets ingress to the SR Domain, destined to SIDs within the SR Domain (i.e., bearing a SID in the destination address). This ingress filtering is via an IACL at SR Domain ingress border nodes. Additional protection is applied via an IACL at each SR Segment Endpoint node, filtering packets not from within the SR Domain, destined to SIDs in the SR Domain. ACLs are easily supported for small numbers of prefixes, making summarization important, and when the prefixes requiring filtering is kept to a seldom changing set.

Additionally, ingress filtering of IPv6 source addresses as recommended in BCP38 SHOULD be used.

7.1. Source Routing Attacks

[RFC5095] deprecates the Type 0 Routing header due to a number of significant attacks that are referenced in that document. Such attacks include bypassing filtering devices, reaching otherwise
unreachable Internet systems, network topology discovery, bandwidth exhaustion, and defeating anycast.

Because this document specifies that the SRH is for use within an SR domain protected by ingress filtering via IACLs; such attacks cannot be mounted from outside an SR Domain. As specified in this document, SR Domain ingress edge nodes drop packets entering the SR Domain destined to segments within the SR Domain.

Additionally, this document specifies the use of IACL on SR Segment Endpoint nodes within the SR Domain to limit the source addresses permitted to send packets to a SID in the SR Domain.

Such attacks may, however, be mounted from within the SR Domain, from nodes permitted to source traffic to SIDs in the domain. As such, these attacks and other known attacks on an IP network (e.g. DOS/DDOS, topology discovery, man-in-the-middle, traffic interception/siphoning), can occur from compromised nodes within an SR Domain.

7.2. Service Theft

Service theft is defined as the use of a service offered by the SR Domain by a node not authorized to use the service.

Service theft is not a concern within the SR Domain as all SR Source nodes and SR segment endpoint nodes within the domain are able to utilize the services of the Domain. If a node outside the SR Domain learns of segments or a topological service within the SR domain, IACL filtering denies access to those segments.

7.3. Topology Disclosure

The SRH is unencrypted and may contain SIDs of some intermediate SR-nodes in the path towards the destination within the SR Domain. If packets can be snooped within the SR Domain, the SRH may reveal topology, traffic flows, and service usage.

This is applicable within an SR Domain but the disclosure is less relevant as an attacker has other means of learning topology, flows, and service usage.

7.4. ICMP Generation

The generation of ICMPv6 error messages may be used to attempt denial-of-service attacks by sending an error-causing destination address or SRH in back-to-back packets. An implementation that correctly follows Section 2.4 of [RFC4443] would be protected by the ICMPv6 rate-limiting mechanism.
7.5. Applicability of AH

The SR Domain is a trusted domain, as defined in [RFC8402] Section 2 and Section 8.2. The SR Source is trusted to add an SRH (optionally verified via the HMAC TLV in this document), and segments advertised within the domain are trusted to be accurate and advertised by trusted sources via a secure control plane. As such the SR Domain does not rely on the Authentication Header (AH) as defined in [RFC4302] to secure the SRH.

The use of SRH with AH by an SR source node, and processing at a SR segment endpoint node, is not defined in this document. Future documents may define use of SRH with AH and its processing.

8. IANA Considerations

This document makes the following registrations in the Internet Protocol Version 6 (IPv6) Parameters "Routing Type" registry maintained by IANA:

<table>
<thead>
<tr>
<th>Suggested Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Segment Routing Header (SRH)</td>
<td>This document</td>
</tr>
</tbody>
</table>

This document makes the following registrations in "Type 4 - Parameter Problem" message of the "Internet Control Message Protocol version 6 (ICMPv6) Parameters" registry maintained by IANA:

<table>
<thead>
<tr>
<th>CODE</th>
<th>NAME/DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD IANA</td>
<td>SR Upper-layer Header Error</td>
</tr>
</tbody>
</table>
of values prior to the RFC being approved for publication, the
Designated Expert can approve allocations once it seems clear that an
RFC will be published. The Designated expert will post a request to
the 6man WG mailing list (or a successor designated by the Area
Director) for comment and review, including an Internet-Draft.
Before a period of 30 days has passed, the Designated Expert will
either approve or deny the registration request and publish a notice
of the decision to the 6man WG mailing list or its successor, as well
as informing IANA. A denial notice must be justified by an
explanation, and in the cases where it is possible, concrete
suggestions on how the request can be modified so as to become
acceptable should be provided.

8.1. Segment Routing Header Flags Register

This document requests the creation of a new IANA managed registry to
identify SRH Flags Bits. The registration procedure is "Expert
Review" as defined in [RFC8126]. Suggested registry name is "Segment
Routing Header Flags". Flags is 8 bits.

8.2. Segment Routing Header TLVs Register

This document requests the creation of a new IANA managed registry to
identify SRH TLVs. The registration procedure is "Expert Review" as
defined in [RFC8126]. Suggested registry name is "Segment Routing
Header TLVs". A TLV is identified through an unsigned 8 bit
codepoint value, with assigned values 0-127 for TLVs that do not
change en route, and 128-255 for TLVs that may change en route. The
following codepoints are defined in this document:

<table>
<thead>
<tr>
<th>Assigned Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Pad1 TLV</td>
<td>This document</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>This document</td>
</tr>
<tr>
<td>2</td>
<td>Reserved</td>
<td>This document</td>
</tr>
<tr>
<td>3</td>
<td>Reserved</td>
<td>This document</td>
</tr>
<tr>
<td>4</td>
<td>PadN TLV</td>
<td>This document</td>
</tr>
<tr>
<td>5</td>
<td>HMAC TLV</td>
<td>This document</td>
</tr>
<tr>
<td>6</td>
<td>Reserved</td>
<td>This document</td>
</tr>
<tr>
<td>124-126</td>
<td>Experimentation and Test</td>
<td>This document</td>
</tr>
<tr>
<td>127</td>
<td>Reserved</td>
<td>This document</td>
</tr>
<tr>
<td>252-254</td>
<td>Experimentation and Test</td>
<td>This document</td>
</tr>
<tr>
<td>255</td>
<td>Reserved</td>
<td>This document</td>
</tr>
</tbody>
</table>

Values 1,2,3,6 were defined in draft versions of this specification
and are Reserved for backwards compatibility with early
implementations and should not be reassigned. Values 127 and 255 are
Reserved to allow for expansion of the Type field in future specifications if needed.

9. Implementation Status

This section is to be removed prior to publishing as an RFC.

See [I-D.matsushima-spring-srv6-deployment-status] for updated deployment and interoperability reports.

9.1. Linux

Name: Linux Kernel v4.14

Status: Production

Implementation: adds SRH, performs END processing, supports HMAC TLV

Details: https://irtf.org/anrw/2017/anrw17-final3.pdf and [I-D.filsfils-spring-srv6-interop]

9.2. Cisco Systems

Name: IOS XR and IOS XE

Status: Production (IOS XR), Pre-production (IOS XE)

Implementation: adds SRH, performs END processing, no TLV processing

Details: [I-D.filsfils-spring-srv6-interop]

9.3. FD.io

Name: VPP/Segment Routing for IPv6

Status: Production

Implementation: adds SRH, performs END processing, no TLV processing

Details: https://wiki.fd.io/view/VPP/Segment_Routing_for_IPV6 and [I-D.filsfils-spring-srv6-interop]

9.4. Barefoot

Name: Barefoot Networks Tofino NPU

Status: Prototype
Implementation: performs END processing, no TLV processing
Details: [I-D.filsfils-spring-srv6-interop]

9.5. Juniper
Name: Juniper Networks Trio and vTrio NPU’s
Status: Prototype & Experimental
Implementation: SRH insertion mode, Process SID where SID is an interface address, no TLV processing

9.6. Huawei
Name: Huawei Systems VRP Platform
Status: Production
Implementation: adds SRH, performs END processing, no TLV processing

10. Contributors

11. Acknowledgements
The authors would like to thank Ole Troan, Bob Hinden, Ron Bonica, Fred Baker, Brian Carpenter, Alexandru Petrescu, Punit Kumar Jaiswal, and David Lebrun for their comments to this document.

12. References
12.1. Normative References
[FIPS180-4]
12.2. Informative References

[I-D.filsfils-spring-srv6-interop]
Filsfils, C., Clad, F., Camarillo, P., Abdelsalam, A., Salsano, S., Bonaventure, O., Horn, J., and J. Liste,
"SRv6 interoperability report", draft-filsfils-spring-srv6-interop-02 (work in progress), March 2019.
Internet-Draft     IPv6 Segment Routing Header (SRH)    August 2019

[I-D.matsushima-spring-srv6-deployment-status]


Authors’ Addresses

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Darren Dukes (editor)
Cisco Systems, Inc.
Ottawa
CA

Email: ddukes@cisco.com

Stefano Previdi
Huawei
Italy

Email: stefano@previdi.net

John Leddy
Individual
US

Email: john@leddy.net

Satoru Matsushima
Softbank

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

Daniel Voyer
Bell Canada

Email: daniel.voyer@bell.ca
Abstract

This document changes the status of RFC2675, IPv6 Jumbograms, from Proposed Standard to Historic.

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 9 November 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.
1. Introduction

[RFC2675] defines the IPv6 Jumbo Payload Option, which enables Jumbograms, IPv6 datagrams that carry a payload greater than 65,535 octets. Jumbograms have seen little deployment in the open Internet and there are currently no known active Internet deployments.

Note: "Jumboframe" is a commonly term that is used to describe frames that exceed 1500 bytes in length, and is different to an IPv6 Jumbo Payload Option, or Jumbogram.

When published, this document changes the status of RFC2675 to historic.

2. Conventions and Definitions

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. Rationale

Jumbograms have seen little deployment, A Roadmap for Transmission Control Protocol (TCP) Specification Documents ([RFC7414]) explains some of the protocol reasons behind this:
"This document states that jumbograms are to only be used when it can be guaranteed that all receiving nodes, including each router in the end-to-end path, will support jumbograms. If even a single node that does not support jumbograms is attached to a local network, then no host on that network may use jumbograms. This explains why jumbogram use has been rare, and why this document is considered a performance optimization and not part of TCP over IPv6’s basic functionality."

Over time, the IPv6 Node Requirements document series has reported on the deployment of Jumbograms, as follows:

* RFC4294: "IPv6 Jumbograms "[RFC-2675]" MAY be supported."

* RFC6434: "To date, few implementations exist, and there is essentially no reported experience from usage."

* RFC8504: "Removed Jumbograms (RFC 2675) as they aren’t deployed."

This document removes support for Jumbograms, and therefore paves the way for the removal of their support from operating system stacks. This also removes the need for testing Jumbogram support, which otherwise require links with a MTU greater than 65,535 bytes, making testing of implementations impractical without significant effort.

4. RFCs Referencing Jumbograms

This section summarises document in the RFC series that mention support for IPv6 Jumbograms.

The Jumbo option is mentioned in a set of documents:

* Protocols that consider the larger possible sized enabled by Jumbograms, for encryption ciphers (AES [RFC3686] and [RFC4309]).

* Protocols that are unable to support Jumbograms, due to their increased length (SRTP [RFC3711] and ROHC [RFC5225]).

* Protocols that consider the jumbogram option field as a possible length format (IPFIX [RFC5102], IPv6 transition [RFC8468]).

TCP specifications have also referred to Jumbograms. Adding support for TCP jumbograms required modification to the Maximum Segment Size and Urgent Pointer fields to interpret a value of 65,535 as infinite. These modifications resulted in references to [RFC2675] in several TCP Documents ([RFC4614], [RFC6691], [RFC7323], [RFC7414]) and the TCP Roadmap [RFC7414], which describes the fundamental changes to TCP required to support Jumbograms.
UDP Usage Guidelines [RFC8085] refers to Jumbogram support for large unfragmentable datagrams:

"IPv6 allows the option of transmitting large packets ("jumbograms") without fragmentation when all link layers along the path support this [RFC2675]."

References also appear in documents that acknowledge the existence of the jumbo option, but do not define new mechanisms. Jumbograms are mentioned in IPv6 node requirements [RFC4294], UDP Guidelines [RFC5405] (historic), IPv6 Avian Carriers [RFC6214], IPv6 node requirements [RFC6434], and DTN convergence [RFC7122].

If published, this document changes the status of RFC2675 to historic. Use of Jumbograms will no longer be specified as an IETF mechanism for use with these IETF-specified protocols.

5. Security Considerations

XXX security considerations XXX

The security considerations for in RFC2675 state: "The Jumbo Payload option and TCP/UDP jumbograms do not introduce any known new security concerns".

6. IANA Considerations

This document has no IANA actions.

7. References

7.1. Normative References


7.2. Informative References


Acknowledgments

Tom Jones and Godred Fairhurst are supported by the University of Aberdeen.

Appendix B. Appendix A

RFC editor: please remove this section before publication.

This appendix provides an annotated list of text where support is mentioned within the RFC series.

<table>
<thead>
<tr>
<th>Document</th>
<th>Status</th>
<th>Title</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC3686</td>
<td>PS</td>
<td>Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)</td>
<td>Considerations to cover large packets</td>
</tr>
<tr>
<td>RFC3711</td>
<td>PS (Updated by RFC5506, RFC6904)</td>
<td>The Secure Real-time Transport Protocol (SRTP)</td>
<td>(except for ipv6 &quot;jumbograms&quot; [RFC2675], which are not likely to be used for RTP-based multimedia traffic).</td>
</tr>
<tr>
<td>RFC3790</td>
<td>Informational</td>
<td>Survey of IPv4 Addresses in Currently Deployed IETF Internet Area Standards Track and Experimental Documents</td>
<td>&quot;This document defines a IPv6 packet format and is therefore not discussed in this document.&quot;</td>
</tr>
<tr>
<td>RFC4294</td>
<td>Informational (Obsoleted by RFC2675)</td>
<td>IPv6 Node Requirements</td>
<td>&quot;ipv6 jumbograms [RFC-2675]&quot; MAY</td>
</tr>
<tr>
<td>RFC</td>
<td>Type</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>4309</td>
<td>PS</td>
<td>RFC4309 PS Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)</td>
<td>Size parameters are set so they will cover jumbograms.</td>
</tr>
<tr>
<td>5102</td>
<td>PS (Obsoleted by [RFC7012])</td>
<td>RFC5102 PS Informational Model for IP Flow Informational Experimentalt</td>
<td>Adds Jumbogram size considerations to length fields.</td>
</tr>
<tr>
<td>5405</td>
<td>BCP (Obsoleted by [RFC8085])</td>
<td>RFC5405 BCP Unicast UDP Usage Guidelines for Application Designers</td>
<td>Jumbograms exist.</td>
</tr>
<tr>
<td>6214</td>
<td>Informational</td>
<td>RFC6214 Informational Adaptation of RFC 1149 for IPv6</td>
<td></td>
</tr>
<tr>
<td>6434</td>
<td>Informational (Obsoleted by [RFC8504])</td>
<td>RFC6434 Informational IPv6 Node Requirements</td>
<td>&quot;to date, few implementations exist, and there is essentially no reported experience from usage.&quot;</td>
</tr>
<tr>
<td>6691</td>
<td>Informational</td>
<td>RFC6691 Informational TCP Options and Maximum Segment Size (MSS)</td>
<td>Treat 65,353 value in MSS and Urgent Pointer fields as infinite.</td>
</tr>
<tr>
<td>7122</td>
<td>Experimental</td>
<td>RFC7122 Experimental Datagram Convergence Layers for the Delay- and Disruption-Tolerant</td>
<td>Jumbograms exist though rarely used.</td>
</tr>
<tr>
<td>RFC</td>
<td>Type</td>
<td>Title</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>7323</td>
<td>PS</td>
<td>TCP Extensions for High performance Jumbograms weaken the TCP checksum</td>
<td></td>
</tr>
<tr>
<td>7414</td>
<td>Informational</td>
<td>A Roadmap for Transmission Control Protocol (TCP) Specification Documents Jumbograms exist</td>
<td></td>
</tr>
<tr>
<td>8085</td>
<td>BCP</td>
<td>UDP Usage Guidelines Jumbograms exist</td>
<td></td>
</tr>
<tr>
<td>8468</td>
<td>Informational</td>
<td>IPv4, IPv6, and IPv4-IPv6 Coexistence: (Updated by )dates for the IP Performance Metrics (IPPM) Framework Length considerations</td>
<td></td>
</tr>
<tr>
<td>8504</td>
<td>BCP</td>
<td>IPv6 Node Requirements &quot;removed Jumbograms (RFC 2675) as they aren’t deployed.&quot;</td>
<td></td>
</tr>
</tbody>
</table>

Table 1

Appendix C. Appendix B

RFC editor please remove this section before publishing

Relevant quotes from PS and BCP documents that reference [RFC2675].

Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP) ([RFC3686]):

"This construction can produce enough key stream for each packet sufficient to handle any IPv6 jumbogram [JUMBO]."

"Note that ESP with 32-bit Sequence Numbers will not exceed 2^64 blocks even if all of the packets are maximum-length IPv6 jumbograms [JUMBO]."
"A 28-bit block counter value is sufficient for the generation of a key stream to encrypt the largest possible IPv6 jumbogram [JUMBO]; however, a 32-bit field is used. This size is convenient for both hardware and software implementations."

The Secure Real-time Transport Protocol (SRTP) ([RFC3711]):

"The AES has a block size of 128 bits, so $2^{16}$ output blocks are sufficient to generate the $2^{23}$ bits of keystream needed to encrypt the largest possible RTP packet (except for IPv6 "jumbograms" [RFC2675], which are not likely to be used for RTP-based multimedia traffic)."

Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP) ([RFC4309]):

"L  L indicates the size of the length field in octets. CCM defines values of L between 2 octets and 8 octets. This specification only supports L = 4. Implementations MUST support an L value of 4 octets, which accommodates a full Jumbogram [JUMBO]; however, the length includes all of the encrypted data, which also includes the ESP Padding, Pad Length, and Next Header fields."

"payload  
The payload of the ESP packet. The payload MUST NOT be longer than 4,294,967,295 octets, which is the maximum size of a Jumbogram [JUMBO]; however, the ESP Padding, Pad Length, and Next Header fields are also part of the payload."

"This construction provides more key stream for each packet than is needed to handle any IPv6 Jumbogram [JUMBO]."

Information Model for IP Flow Information Export ([RFC5102], Obsolete):

"5.4.30. payloadLengthIPv6  
Description:  
This Information Element reports the value of the Payload Length field in the IPv6 header. Note that IPv6 extension headers belong to the payload. Also note that in case of a jumbo payload option the value of the Payload Length field in the IPv6 header is zero and so will be the value reported by this Information Element."
5.7.1. ipPayloadLength

Description:
The effective length of the IP payload. For IPv4 packets, the value of this Information Element is the difference between the total length of the IPv4 packet (as reported by Information Element totalLengthIPv4) and the length of the IPv4 header (as reported by Information Element headerLengthIPv4). For IPv6, the value of the Payload Length field in the IPv6 header is reported except in the case that the value of this field is zero and that there is a valid jumbo payload option. In this case, the value of the Jumbo Payload Length field in the jumbo payload option is reported.

RObust Header Compression Version 2 (ROHCv2): Profiles for RTP, UDP, IP, ESP and UDP-Lite ([RFC5225]):

"IPv6 headers using the jumbo payload option of RFC 2675 ’[RFC2675]’ will not be compressible with this encoding method since the value of the payload length field does not match the length of the packet.

UDP Usage Guidelines ([RFC5405], Obsolete):

"IPv6 allows the option of transmitting large packets ("jumbograms") without fragmentation when all link layers along the path support this ‘[RFC2675]’.

TCP Extensions for High Performance ([RFC7323]):

"Expanding the TCP window beyond 64 KiB for IPv6 allows Jumbograms ‘[RFC2675]’ to be used when the local network supports packets larger than 64 KiB. When larger TCP segments are used, the TCP checksum becomes weaker."

"The same technique applies to IP version 6, except in the case of IPv6 Jumbograms. When IPv6 Jumbograms are supported, [RFC2675] requires additional steps for dealing with the Urgent Pointer; these steps are described in Section 5.2 of ‘[RFC2675]’.

UDP Usage Guidelines [RFC8085]:

"IPv6 allows the option of transmitting large packets ("jumbograms") without fragmentation when all link layers along the path support this ‘[RFC2675]’.

IPv6 Node Requirements ([RFC8504]):

"Removed Jumbograms (RFC 2675) as they aren’t deployed."
Authors’ Addresses

Tom Jones  
University of Aberdeen  
Email: tom@erg.abdn.ac.uk

Godred Fairhurst  
University of Aberdeen  
Email: gorry@erg.abdn.ac.uk
IPv6 Encapsulation for SFC and IFIT
draft-li-6man-ipv6-sfc-ifit-01

Abstract

Service Function Chaining (SFC) and In-situ Flow Information Telemetry (IFIT) are important path services along with the packets. In order to support these services, several encapsulations have been defined. The document analyzes the problems of these encapsulations in the IPv6 scenario and proposes the possible optimized encapsulation for IPv6.

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 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

Li & Peng Expires January 9, 2020
1. Introduction

Service Function Chaining (SFC) [RFC7665] and In-situ Flow Information Telemetry (IFIT) [I-D.song-opsawg-ifit-framework] are important path services along with the packets. In order to support these services, several encapsulations have been defined. Network Service Header (NSH) is defined in [RFC8300] as the encapsulation for SFC. For IFIT encapsulations, In-situ OAM (IOAM) Header is defined in [I-D.ietf-ippm-ioam-data] and Postcard-Based Telemetry (PBT) Header is defined in [I-D.song-ippm-postcard-based-telemetry]. Inband Flow Analyzer (IFA) is also defined in [I-D.kumar-ippm-ifa] to record flow specific information from an end station and/or switches across a network. In the application scenario of IPv6, these encapsulations propose challenges for the data plane. The document analyzes the problems and proposes the possible optimized encapsulation for IPv6.

2. Terminology

SFC: Service Function Chaining

IFIT: In-situ Flow Information Telemetry
3. Problem Statement

The problems posed by the current encapsulations for SFC and IFIT in the application scenarios of IPv6 and SRv6 include:

1. According to the encapsulation order recommended in [RFC8200], if the IOAM is encapsulated in the IPv6 Hop-by-Hop options header, in the incremental trace mode of IOAM as the number of nodes traversed by the IPv6 packets increases, the recorded IOAM information will increase accordingly. This will increase the length of the Hop-by-Hop options header and cause increasing difficulties in reading the following Segment Routing Extension Header (SRH) [I-D.ietf-6man-segment-routing-header] and thereby reduce the forwarding performance of the data plane greatly.

2. With the introduction of SRv6 network programming [I-D.ietf-spring-srv6-network-programming], the path services along with the IPv6 packets can be processed at all the IPv6 network nodes or only at the SRv6 enabled network nodes along the path. It is necessary to distinguish the encapsulations for the specific path service which should be processed by the IPv6 path or the SRv6 path.

3. Both NSH and IOAM need the Metadata field to record metadata information. However currently these metadata has to be recorded separately which may generate redundant metadata information or increase the cost of process.

4. There is unnecessary inconsistency in the current encapsulations for IOAM, IFA and PBT in the IPv6 scenario. Especially it seems unnecessary to define a new specific IPv6 header for IFA, i.e. IFA header.

4. Design Consideration

To solve the problems stated above, in the application scenarios of IPv6 and SRv6, the encapsulations of SFC and IFIT can be optimized with the following design considerations:

- To separate the SFC/IFIT path service into two parts, i.e. instruction and recording parts. The instruction part
with fixed length) can be placed in the front IPv6 extension
headers including Hop-by-Hop options header, Destination options
header, Routing header, etc. while the recording part can be
placed in the back IPv6 extension headers such as being placed
after IPv6 Routing Header. In this way the path service
instruction in the IPv6 extension headers can be fixed as much as
possible to facilitate hardware process to keep forwarding
performance while the SFC/IFIT metadata recording part is placed
afterwards which enables to stop recording when too much recording
information has to be carried to reach the limitation of hardware
process.

- To define SFC/IFIT path service instructions as IPv6 options
  uniformly which can be placed either in the Hop-by-hop options
  which indicates the path service processed by all IPv6 enabled
  nodes along the path or in the SRH option TLVs which indicates the
  path service processed only by the SRv6 nodes along the SRv6 path
  indicated by the Segment List in the SRH.

- To define a unified IPv6 metadata header which can be used as a
  container to record the service metadata of SFC, IFIT and other
  possible path services.

According to the above design optimization consideration, in the
application scenarios of IPv6 and SRv6 the encapsulations for SFC and
IFIT can be defined as below.

4.1. Service Options

1. NSH Service Option

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Ver|O|U|  TTL   |   Length   |U|U|U|MD Type| Next Protocol |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Service Path Identifier              | Service Index |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1. IPv6 Options with NSH instructions

Option Type: TBD_0
Opt Data Len: 8 octets.
Other fields: refer to [RFC8300].
2. IOAM Service Option

Option Type: TBD_1
Opt Data Len: 8 octets.
Other fields: refer to [I-D.ietf-ippm-ioam-data].

3. PBT Service Option

Option Type: TBD_2
Opt Data Len: 20 octets.
Other fields: refer to [I-D.song-ippm-postcard-based-telemetry].

4. IFA Service Option
Option Type: TBD_3
Opt Data Len: 4 octets.

Other fields: refer to [I-D.kumar-ippm-ifa].

These options can be put in the IPv6 Hop-by-Hop Options Header or SRH TLV.

4.2. IPv6 Service Metadata Options

As introduced in [I-D.li-6man-enhanced-extension-header], IPv6 Metadata Header is defined as a new type of IPv6 extension header. The metadata is the information recorded by each hop for specific path services, and carried in corresponding service metadata options. The length of the metadata is variable.

4.2.1. SFC Service Metadata Option

For the SFC service, the corresponding SFC service metadata option is defined as shown in Figure 5.

```
<table>
<thead>
<tr>
<th>SFC Type</th>
<th>Length</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFC Metadata Class</td>
<td>Type</td>
<td>U</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Variable-Length Metadata</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 5. SFC Service Metadata
SFC Type 8-bit identifier of the service type, i.e. SFC. The value is TBD-4.

Length 8-bit unsigned integer. Length of the Service Metadata field, in octets.

Metadata Class Defines the scope of the Type field to provide a hierarchical namespace. IANA has set up the "NSH MD Class" registry, which contains 16-bit values [RFC8300].

Type Indicates the explicit type of metadata being carried. The definition of the Type is the responsibility of the MD Class owner.

Unassigned bit One unassigned bit is available for future use. This bit MUST NOT be set, and it MUST be ignored on receipt.

Length Indicates the length of the variable-length metadata, in bytes. Detailed specification in [RFC8300].

4.2.2. IOAM Service Metadata Option

For the IOAM service, the corresponding IOAM service metadata option is defined as shown in Figure 6.

```
+---------------+---------------+---------------+---------------+
| IOAM Type     | Length        | Reserved      |
+---------------+---------------+---------------+
+---------------+---------------+---------------+
| IOAM Service Metadata Options (variable) |
+---------------+---------------+---------------+
```

Figure 6. IOAM Service Metadata
IOAM Type | 8-bit identifier of the IOAM Service Metadata type. The value is TBD-5.
Length | 8-bit unsigned integer. Length of the IOAM Service Metadata field, in octets.
RESERVED | 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.
IOAM Service Metadata Options | IOAM option data is present as specified by the IOAM Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

All the IOAM IPv6 options require 4n alignment. This ensures that 4 octet fields specified in [I-D.ietf-ippm-ioam-data] such as transit delay are aligned at a multiple-of-4 offset from the start of the IPv6 Metadata header.

In addition, to maintain IPv6 extension header 8-octet alignment and avoid the need to add or remove padding at every hop, the Trace-Type for Incremental Tracing Option in IPv6 MUST be selected such that the IOAM node data length is a multiple of 8-octets.

### 4.2.3. IFA Service Metadata Option

For the IOAM service, the corresponding IOAM service metadata option is defined as shown in Figure 6.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| IFA Type | Length | Reserved |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| IFA Service Metadata Options (variable) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6. IFA Service Metadata
5. IANA Considerations

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD_0</td>
<td>NSH Service Option</td>
<td>[This draft]</td>
</tr>
<tr>
<td>TBD_1</td>
<td>IOAM Service Option</td>
<td>[This draft]</td>
</tr>
<tr>
<td>TBD_2</td>
<td>PBT Service Option</td>
<td>[This draft]</td>
</tr>
<tr>
<td>TBD_3</td>
<td>IFA Service Option</td>
<td>[This draft]</td>
</tr>
<tr>
<td>TBD_4</td>
<td>SFC Service Metadata Type</td>
<td>[This draft]</td>
</tr>
<tr>
<td>TBD_5</td>
<td>IOAM Service Metadata Type</td>
<td>[This draft]</td>
</tr>
<tr>
<td>TBD_6</td>
<td>IFA Service Metadata Type</td>
<td>[This draft]</td>
</tr>
</tbody>
</table>

6. Security Considerations

TBD.

7. References

7.1. Normative References

[I-D.guichard-spring-nsh-sr]

[I-D.ietf-6man-segment-routing-header]
[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., daniel.bernier@bell.ca, d., and J. Lemon,
"Data Fields for In-situ OAM", draft-ietf-ippm-ioam-
data-06 (work in progress), July 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.

[I-D.kumar-ippm-ifa]
Kumar, J., Anubolu, S., Lemon, J., Manur, R., Holbrook,
Flow Analyzer", draft-kumar-ippm-ifa-01 (work in
progress), February 2019.

[I-D.song-ippm-postcard-based-telemetry]
Song, H., Zhou, T., Li, Z., Shin, J., and K. Lee,
"Postcard-based On-Path Flow Data Telemetry", draft-song-
ippm-postcard-based-telemetry-04 (work in progress), June
2019.

[I-D.song-opsawg-ifit-framework]
Song, H., Li, Z., Zhou, T., Qin, F., Shin, J., and J. Jin,
"In-situ Flow Information Telemetry Framework", draft-
song-opsawg-ifit-framework-02 (work in progress), June
2019.

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

Writing an IANA Considerations Section in RFCs", BCP 26,
RFC 8126, DOI 10.17487/RFC8126, June 2017,

(IPv6) Specification", STD 86, RFC 8200,
DOI 10.17487/RFC8200, July 2017,
7.2. Informative References


Authors’ Addresses

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Shuping Peng
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: pengshuping@huawei.com
Abstract

A multitude of applications are carried over the network, which have varying needs for network bandwidth, latency, jitter, and packet loss, etc. Some applications such as online gaming and live video streaming have very demanding network requirements thereof require special treatments in the network. However, since the current network is lack of enough information of service requirements of such applications it is difficult to guarantee the SLA or it may take long time to provide such guarantee. This document proposes the solution to make use of IPv6 extensions header to convey the service requirement information along with the packet to the network to facilitate the service deployment and network resource adjustment to guarantee SLA for applications. Then it defines the service-aware options which can be used in the different IPv6 extension headers for the purpose.

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 12, 2019.

Li & Peng Expires September 12, 2019 [Page 1]
1. Introduction

A multitude of applications are carried over the network, which have varying needs for network bandwidth, latency, jitter, and packet loss, etc. Some applications such as online gaming and live video streaming have very demanding network requirements thereof require special treatments in the network. However, since the current network is lack of enough information of service requirements of such applications it is difficult to guarantee the SLA or it may take long time to provide such guarantee. This document proposes the solution to take use of IPv6 extensions header to convey the service requirement information along with the packet to the network to facilitate the service deployment and network resource adjustment to guarantee SLA for applications. Then it defines the service-aware
options which can be used in the different IPv6 extension headers for the purpose.

2. Use Cases

This section shows the various demanding requirements of some applications in the following use cases. The traffic of these applications needs to be differentiated from other traffic and applied with special treatments in the network.

2.1. Online Gaming

Good network performance is normally a prerequisite for satisfactory game play, especially for the online gaming. The maximum allowable ping rate (network latency) and the required minimum download/upload speed (network bandwidth) are the key factors to make the online gaming playable. Shooting or racing online gaming is normally based on quick action and needs to update the game status in real time by continuously sending and receiving updates to/from the game server and/or other players. The network paths with low latency and low packet loss need to be explicitly selected from the game players to the game server.

2.2. Video streaming

The network latency, jitter, bandwidth, and packet loss are the key factors for the video streaming. Live video streaming has even more strict requirements. High quality video source (e.g. from Netflix) require more bandwidth in order to stream properly. Real time streaming services also requires real time content delivery from the web server to the end user ideally via carefully planned explicit TE paths. The online gaming often involves live video streaming.

3. Problem Statement

[RFC3272] reviews a number of IETF activities which are primarily intended to evolve the IP architecture to support new service definitions which allow preferential or differentiated treatment to be accorded to certain types of traffic. The challenge when using traditional ways to guarantee SLA is that the packets are not able to carry enough information of service requirements of applications. The network devices mainly relies on the 5-tuple of the packets which cannot provide fine-grained service process. If more information is needed, it has to refer to DPI which will introduce more cost in the network and impose security challenges.

In the era of SDN the orchestrator is introduced for the orchestration of applications and the network. The SDN controller
can be aware of the service requirements of the applications on the network through the interface interworking with the orchestrator. The service requirements is used by the controller for traffic management. The method raises the following problems: 1) The whole loop is long and time-consuming which is not suitable for the real-time adjustment for applications; 2) Too many interfaces are involved in the loop which proposes more challenges of standardization and inter-operability, and it is difficult to be standardized for easy interworking.

4. Framework

In the service-aware IPv6 network shown in Figure 1, there are following components:

1. Service-aware Apps: The IPv6 enabled applications runs in the host which can add the service requirements of the applications on network through the IPv6 extension header ([RFC8200]) or remove it from the IPv6 extension header. The service requirement information includes the IPv6 service-aware ID which identifies the IPv6 packets of the traffic belongs to the specific SLA level/Applications/User and the parameters for the specific service such as bandwidth, delay, delay variation, packet loss ratio, etc. The service requirements will be processed by the IPv6 enabled nodes along the path or the SRv6 ([I-D.filipsfils-spring-srv6-network-programming]) enabled node along the SRv6 path which be programmed in the host. The Apps can also need not to add any service requirement information in the IPv6 extension header.

2. Service-aware Edge Device: The Edge Device can add the service requirements of the applications on network through the IPv6 extension header on behalf of the IPv6 enabled applications or change the service requirements conveyed by the packets of the service-aware applications according to local policies which is out of the scope of this document. The service requirements will be processed by the...
IPv6 enabled nodes along the path or the SRv6 enabled node along the SRv6 path which be programmed by the Edge Device.

3. Service-process Head-End: The service requirements may be processed as a service path such as SRv6 TE path of SFC at the Service-process Head-End. The service requirements conveyed in the IPv6 packets can be mapped to a service path which satisfies the specific requirement, trigger to set up the new service path by the Head-End, or trigger the global traffic adjustment by the controller according to the information provided by the network devices. The process depends on the local policy which is out of the scope this document.

4. Service-process Mid-Point: The Mid-Point provides the path service according to the service path set up by the Head-End which satisfies the service requirement conveyed by the IPv6 packets. The Mid-Point may also adjust the resource locally to guarantee the service requirements depending on specific policies which is out of the scope of this document.

5. Service-process End-Point: The process of the specific service path will end at the End-Point. The service requirements information can be removed at the End-Point or go on to be conveyed with the IPv6 packets.

In this way the network is able to be aware of the service requirements of the applications explicitly. According to these service requirement information carried in the IPv6 packets the network is able to adjust its resource fast to satisfy the service requirement of applications. The flow-driven method also reduces the challenges of inter-operability and loop control loop.

5. Service-aware Options

Two service-aware options are defined, i.e. Service-aware ID option and Service-Para Option to support the Service-aware IPv6 network.

5.1. Service-aware ID Option

The Service-aware ID option indicates the information of the applications, users, and service requirements, which is defined in the following figure:
Option Type: TBD
Opt Data Len: 16 octets.

The IPv6 Service-aware ID is 128bits long which can have the following structures:

-- Structure I: Any combination of SLA level (e.g. Gold, Silver, Bronze), APP ID, and/or user ID. The length of each field is variable, which is shown in the following diagram:

```
+---------------------------------------------------------------+
<table>
<thead>
<tr>
<th>SLA Level</th>
<th>APP ID</th>
<th>User ID</th>
</tr>
</thead>
</table>
+---------------------------------------------------------------+
```

Figure 3. IPv6 Service-aware ID Structure I

-- Structure II: Any combination of SLA level (e.g. Gold, Silver, Bronze), APP ID, and/or user ID plus the arguments which indicates the service requirements of the identified application, which is shown in the following diagram:

```
+---------------------------------------------------------------+
<table>
<thead>
<tr>
<th>SLA Level</th>
<th>APP ID</th>
<th>User ID</th>
<th>Arguments</th>
</tr>
</thead>
</table>
+---------------------------------------------------------------+
```

Figure 4. IPv6 Service-aware ID Structure II

-- Structure III: An SRv6 SID, with its arguments as the information specified in Structure 2, which is shown in the following diagram:
This Option can be put into the IPv6 Hop-by-Hop Options, Destination Options, and SRH TLV ([I-D.ietf-6man-segment-routing-header]).

5.2. Service-Para Option

The Service-Para Option is a variable-length option carrying multiple service requirement parameters. Each service requirement parameter is put into the corresponding Service Para Sub-TLV, as shown in Figure 6. This Option can be put into the IPv6 Hop-by-Hop Options, Destination Options, and SRH TLV.

Figure 6. IPv6 Service-Para Option

<table>
<thead>
<tr>
<th>Option Type</th>
<th>Opt Data Len</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option Type</td>
<td>Opt Data Len</td>
</tr>
<tr>
<td>Service Para Sub-TLVs(Variable)</td>
<td></td>
</tr>
</tbody>
</table>

The corresponding Service Para Sub-TLVs are shown in the following figures respectively.

1. BW Sub-TLV

This BW sub-TLV indicates the bandwidth requirement of applications. The format of this sub-TLV is shown in the following diagram:
where:

Type: TBD
Length: 4

Class Type: The Bandwidth Type.

RESERVED: This field is reserved for future use. It MUST be set to 0 when sent and MUST be ignored when received.

Bandwidth: This field carries the bandwidth requirement along the path.

2. Delay Sub-TLV

This Delay Sub-TLV indicates the delay requirement of applications. The format of this sub-TLV is shown in the following diagram:

where:

Type: TBD
Length: 4

RESERVED: This field is reserved for future use. It MUST be set to 0 when sent and MUST be ignored when received.
Delay: This 24-bit field carries the delay requirements in microseconds, encoded as an integer value. When set to the maximum value 16,777,215 (16.777215 sec), then the delay is at least that value and may be larger. This value is the highest delay that can be tolerated.

3. Delay Variation Sub-TLV

This Delay Variation Sub-TLV indicates the delay variation requirement of applications. The format of this sub-TLV is shown in the following diagram:

```
+-------------------------------+
|   Type        |     Length    |
+-------------------------------+
|  RESERVED     |               |
|               | Delay Variation|
```

Figure 9. Delay Variation Sub-TLV

where:

Type: TBD
Length: 4

RESERVED: This field is reserved for future use. It MUST be set to 0 when sent and MUST be ignored when received.

Delay Variation: This 24-bit field carries the delay variation requirements in microseconds, encoded as an integer value.

4. Packet Loss Ratio Sub-TLV

This Packet Loss Ratio Sub-TLV indicates the packet loss ratio requirement of applications. The format of this sub-TLV is shown in the following diagram:
where:

Type: TBD

Length: 4

RESERVED: This field is reserved for future use. It MUST be set to 0 when sent and MUST be ignored when received.

Link Loss: This 24-bit field carries link packet loss ratio requirement. This value is the highest packet-loss ratio that can be tolerated.

6. IANA Considerations

IANA maintains the registry for the Options and Sub-TLVs.

Service-Para Option will require one new type code per sub-TLV defined in this document:

<table>
<thead>
<tr>
<th>Type Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD Service-aware ID Option</td>
</tr>
<tr>
<td>TBD Service-Para Option</td>
</tr>
<tr>
<td>TBD BW Sub-TLV</td>
</tr>
<tr>
<td>TBD Delay Sub-TLV</td>
</tr>
<tr>
<td>TBD Delay Variation Sub-TLV</td>
</tr>
<tr>
<td>TBD Packet Loss Sub-TLV</td>
</tr>
</tbody>
</table>
7. Security Considerations

TBD

8. References

8.1. Normative References

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J.,
daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6
Network Programming", draft-filsfils-spring-srv6-network-
programming-07 (work in progress), February 2019.

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Previdi, S., Leddy, J., Matsushima, S., and
d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header
(SRH)", draft-ietf-6man-segment-routing-header-16 (work in
progress), February 2019.

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

(IPv6) Specification", STD 86, RFC 8200,
DOI 10.17487/RFC8200, July 2017,

8.2. Informative References

Xiao, "Overview and Principles of Internet Traffic
Engineering", RFC 3272, DOI 10.17487/RFC3272, May 2002,

Authors’ Addresses

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing  100095
China

Email: lizhenbin@huawei.com
Shuping Peng
Huawei Technologies
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: pengshuping@huawei.com
Abstract

Segment Routing (SR) allows for a flexible definition of end-to-end paths by encoding paths as sequences of sub-paths, called "segments". Segment routing architecture can be implemented over IPv6 data plane, called SRv6. In some use-cases such as end-to-end SR Path Protection and Performance Measurement (PM), SRv6 path need to be identified. This document defines the encoding and processing of Path Segment in SRv6 networks.

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.
Internet-Draft               SRv6 PSID Encap                   July 2019

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
   1.1. Requirements Language .................................... 3
   1.2. Terminology .............................................. 3
2.  Encoding of SRv6 Path Segment ................................. 3
   2.1. Encapsulation of SRv6 Path Segment ....................... 4
   2.2. Format of SRv6 Path Segment ............................... 5
      2.2.1. SRv6 Path Segment: Locator and Local ID .............. 5
      2.2.2. SRv6 Path Segment: Global ID ......................... 5
3.  Processing of SRv6 Path Segment .............................. 6
4.  IANA Considerations ........................................... 7
5.  Security Considerations ...................................... 7
6.  Acknowledgements ............................................. 7
7.  References .................................................. 7
   7.1. Normative References .................................... 7
   7.2. Informative References ................................. 8
   Authors’ Addresses ........................................... 9

1.  Introduction

Segment routing (SR) [RFC8402] is a source routing paradigm that explicitly indicates the forwarding path for packets at the ingress node by inserting an ordered list of instructions, called segments.

When segment routing is deployed on IPv6 dataplane, it is called SRv6 [I-D.ietf-6man-segment-routing-header], and it uses the a new IPv6 [RFC8200] Extension Header (EH) called the IPv6 Segment Routing Header (SRH) [I-D.ietf-6man-segment-routing-header] to construct SRv6 path. As per [I-D.ietf-spring-srv6-network-programming], an SRv6 segment is a 128-bit value, which can be represented as LOC:FUNCT, where LOC is the L most significant bits and FUNCT is the 128-L least significant bits. Most often the LOC part of the SID is routable and leads to the node which instantiates that SID. The FUNCT part of the SID is an opaque identification of a local function bound to the SID.

In several use cases, such as binding bidirectional path [I-D.li-pce-sr-bidir-path] and end-to-end performance measurement [I-D.gandhi-spring-twamp-srpm], the ability to implement path identification is a pre-requisite. In SRv6, an SRv6 path can be identified by the content of the segment list. However, the segment
list may not be a good key to identify an SRv6 path, since the length of segment list is too long and flexible according to the number of SIDs. Therefore, [I-D.li-spring-srv6-path-segment] defines SRv6 Path Segment in order to identify an SRv6 path.

This document defines the encoding and processing of SRv6 Path Segment in SRv6 networks.

1.1. 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.

1.2. Terminology

PM: Performance Measurement.

SID: Segment ID.

SL: Segment List.

SR: Segment Routing.

SRH: Segment Routing Header.

PSID: Path Segment Identifier.

PSP: Penultimate Segment Popping.

Further, this document makes use of the terms defined in [RFC8402] and [I-D.ietf-spring-srv6-network-programming].

2. Encoding of SRv6 Path Segment

This section will describe the encoding of SRv6 Path Segment [I-D.li-spring-srv6-path-segment] in SRH. As per [I-D.li-spring-srv6-path-segment], an SRv6 Path Segment is a 128-bits value, which identifies an SRv6 path. Depending on the use case, an SRv6 Path Segment can identify:

- an SRv6 path within an SRv6 domain
- an SRv6 Policy
2.1. Encapsulation of SRv6 Path Segment

The SRv6 Path Segment MUST appear only once in a SID list, and it MUST appear at the last entry, so the SRv6 Path Segment MUST NOT be copied to the IPv6 destination address. The format of the SRv6 Path Segment follows the format described in section 2.2.

In order to indicate the existence of Path Segment in the SRH, this document defines a P-bit in SRH flag field. The encapsulation of SRv6 Path Segment is shown below.

```
+---------------+---------------+---------------+---------------+
|   Next Header |  Hdr Ext Len  |  Routing Type |   Segments Left |
|---------------+---------------+---------------+---------------|
|   Last Entry  |   Flags       |     P         |     Tag       |
+---------------+---------------+---------------+---------------|
| Segment List[0] (128 bits IPv6 address) |
| ... |
| Segment List[n-1] (128 bits IPv6 address) |
| SRv6 Path Segment (Segment List[n], 128 bits IPv6 value) |
+---------------+---------------+---------------+---------------+
// Optional Type Length Value objects (variable) //
+---------------+---------------+---------------+---------------+
```

Figure 1. SRv6 Path Segment in SID List
2.2. Format of SRv6 Path Segment

This document proposes two types of SRv6 Path Segment format.

Editor's Note: Authors would like to request comments of these encoding mechanisms of SRv6 Path Segment. The appropriate encoding will be maintained while the rest will be deleted in the future version of this document.

2.2.1. SRv6 Path Segment: Locator and Local ID

As per [I-D.ietf-spring-srv6-network-programming], an SRv6 segment is a 128-bit value, which can be represented as LOC:FUNCT, where LOC is the L most significant bits and FUNCT 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 LOC part of the SID is routable and leads to the node which instantiates that SID. The FUNCT part of the SID is an opaque identification of a local function bound to the SID. The FUNCT value zero is invalid.

SRv6 Path Segment can follow the format, where the LOC part identifies the egress node that allocates the Path Segment, and the FUNCT part is an unique local ID to identify an SRv6 Path towards to the egress on the egress.

The Function Type of SRv6 Path Segment is END.PSID (End Function with Path Segment Identifier, to be allocated by IANA).

The proposed P bit can be used to identify that the last SID is an SRv6 Path Segment.

+---------------------------------------------------------------+  +---------------------------------------------------------------+
| Locator        |   Function ID                                  |
+---------------------------------------------------------------+  +---------------------------------------------------------------+
|<----------------------128 bits----------------------------->|  |<----------------------128 bits----------------------------->|

Figure 2. PSID in Format LOC:FUNCT

2.2.2. SRv6 Path Segment: Global ID

An SRv6 Path Segment ID can be a Global ID, and its format depends on the use case.
The SRv6 Path Segment will not be copied to the IPv6 Destination Address, so the SRv6 Path Segment ID can be allocated from an independent 128-bits ID Space. In this case, a new table should be maintained at the node for SRv6 Path Segment. The proposed P bit can be used to identify that the last SID is an SRv6 Path Segment and need to be looked up in the SRv6 Path Segment table.

+--------------------------------------------------------------+
|                         Global ID/PSID                        |
+--------------------------------------------------------------+
|<------------------------128 bits------------------------->|

Figure 3. A Global ID as an PSID

3. Processing of SRv6 Path Segment

As per [I-D.li-spring-srv6-path-segment], an SRv6 Path Segment is a local segment allocated by an egress node. An SRv6 Path Segment can be allocated through several ways, such as CLI, BGP [I-D.li-idr-sr-policy-path-segment-distribution], PCEP [I-D.li-pce-sr-path-segment] or other means. The mechanisms through which an SRv6 Path Segment is allocated is out of scope of this document.

When the SRv6 Path Segment is allocated by the egress, it MUST be distributed to the ingress node. In this case, only the egress will process the SRv6 Path Segment, and other nodes specified by SIDs in the SID list do not know how to process the SRv6 Path Segment.

An SRv6 Path Segment may be distributed to the SRv6 nodes along the SRv6 path. In this case, the SRv6 nodes that learn SRv6 Path Segment may process the SRv6 Path Segment depending on the use case.

When the SRv6 Path Segment is used, the following rules apply:

- The SRv6 Path Segment MUST appear only once in a SID list, and it MUST appear at the last entry. Only the one that appears at the last entry in the SID list will be processed. SRv6 Path Segment appears at other location in the SID list will be treated as an error.

- When an SRv6 Path Segment is inserted, the SL MUST be initiated to be less than the value of Last Entry, and will not point to SRv6 Path Segment. For instance, when the Last entry is 4, the SID List[4] is the SRv6 Path Segment, so the SL MUST be set to 3 or other numbers less than Last entry.
The SRv6 Path Segment MUST NOT be copied to the IPv6 destination address.

Penultimate Segment Popping (PSP, as defined in [I-D.ietf-spring-srv6-network-programming]) MUST be disabled.

The ingress needs to set the P-bit when an SRv6 Path Segment is inserted in the SID List. Nodes that supporting SRv6 Path Segment processing will inspect the last entry to process SRv6 Path Segment when the P-bit is set. When the P-bit is unset, the nodes will not inspect the last entry.

The specific SRv6 Path Segment processing depends on use cases, and it is out of scope of this document.

4. IANA Considerations

TBA

5. Security Considerations

TBA

6. Acknowledgements

TBA

7. References

7.1. Normative References

[I-D.ietf-6man-segment-routing-header]

[I-D.ietf-spring-srv6-network-programming]

[I-D.li-spring-srv6-path-segment]
7.2. Informative References

[I-D.gandhi-spring-twamp-srpm]

[I-D.ietf-spring-segment-routing-policy]

[I-D.li-idr-sr-policy-path-segment-distribution]

[I-D.li-pce-sr-bidir-path]

Authors’ Addresses

Cheng Li
Huawei Technologies
Email: chengli13@huawei.com

Weiqiang Cheng
China Mobile
Email: chengweiqiang@chinamobile.com

Zhenbin Li
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095
China
Email: lizhenbin@huawei.com

Dhruv Dhody
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore 560066
India
Email: dhruv.ietf@gmail.com
Abstract

This document updates RFC 8505 in order to enable unicast address lookup from a 6LoWPAN Border Router acting as an Address Registrar.

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 29, 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.
1. Introduction

[RFC8505] defines the Routing Registrar and extends [RFC6775] to use a 6LoWPAN Border Router (6LBR) as a central service for Address Registration and duplicate detection amongst Routing Registrars and possibly individual Nodes that access it directly.

[I-D.ietf-6lo-backbone-router] introduces the Backbone Router (6BBR) as a Routing Registrar that performs IPv6 ND [RFC4861] [RFC4862] proxy operation between IPv6 Nodes on a federating Backbone Link and Registering Nodes attached to a LowPower Lossy Networks (LLNs) that register their addresses to the 6BBR. The federated links form a Multilink Subnet (MLSN).

The 6BBRs may exchange Extended Duplicate Address Messages (EDAR and EDAC) [RFC8505] to register the proxied addresses on behalf of the Registering Nodes to the 6LBR. The Registration Ownership Verifier (ROVR) field in the EDAR and EDAC messages is used to correlate attempts to register the same address and to detect duplications. The ROVR can also be used as a proof-of-ownership (see...
[I-D.ietf-6lo-ap-nd]) to protect the Registered address against theft and impersonation attacks (more in [I-D.bi-savi-wlan]). Conflicting registrations to different 6BBRs for the same Registered address are resolved using the TID field, which creates a temporal order and enables to recognize the freshest registration.

With [I-D.ietf-6lo-backbone-router], the Link Layer address (LLA) that the 6BBR advertises for a Registered address on behalf of the Registered Node over the Backbone can belong to the Registering Node; in that case, the 6BBR acts as a Bridging Proxy and bridges the unicast packets. Alternatively, the LLA can be that of the 6BBR on the Backbone interface, in which case the 6BBR acts as a Routing Proxy, that receives the unicast packets at Layer-3 and routes them. The 6BBR signals that LLA in a Source LLA Option (SLLAO) in the EDAR messages to the 6LBR, and the 6LBR responds with a Target LLA Option (TLLAO) that indicates the LLA associated to the current registration.

It results that the 6LBR is capable of providing the LLA mapping for any address that was proactively registered with an SLLAO. This draft defines the protocol elements and the operations to try a unicast lookup with the 6LBR. This may save a reactive IPv6 ND Neighbor Solicitation (NS) message, which is based on multicast and may be problematic in extensive wireless domains (see [I-D.ietf-mboned-ieee802-mcast-problems]) as well as in large switched fabrics.

The registration and lookup services that the 6LBR provides do not have to be limited to 6BBRs and are available to any node that supports [RFC8505] and [I-D.ietf-6lo-backbone-router] to register an address, and / or this specification to resolve a mapping. The services are available on-link using an IPv6 NDP NS and off-link using a new variation of the Extended Duplicate Address messages called Address Mapping Messages. The policy and security settings that allow the access to the 6LBR are out of scope.

2. Terminology

2.1. BCP 14

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.
2.2. References

This document uses terms and concepts that are discussed in:

- "Neighbor Discovery for IP version 6" [RFC4861] and "IPv6 Stateless address Autoconfiguration" [RFC4862],
- Neighbor Discovery Optimization for Low-Power and Lossy Networks [RFC6775], as well as
- "Registration Extensions for 6LoWPAN Neighbor Discovery" [RFC8505] and "IPv6 Backbone Router" [I-D.ietf-6lo-backbone-router].

2.3. New Terms

This document introduces the following terminology:

Address Mapping Request

An ICMP message with an ICMP type of 157 (DAR) and a Code Prefix of 1.

Address Mapping Confirm

An ICMP message with an ICMP type of 158 (DAC) and a Code Prefix of 1.

Address Registrar

The Address Registrar is an abstract database that is maintained by the 6LBR to store the state associated with its registrations.

Address Registration

An Address Registration is an abstract state associated to one registration, in other words one entry in the Address Registrar.

2.4. Acronym Definitions

This document uses the following acronyms:

6BBR: 6LoWPAN Backbone Router
6LBR: 6LoWPAN Border Router
6LR: 6LoWPAN Router
3. Overview

Figure 1 illustrates a Backbone Link that federates a collection of LLNs as a single IPv6 Subnet, with a number of 6BBRs providing proxy-ND services to their attached LLNs.

A collection of IPv6 Nodes are present on the Backbone and use IPv6 ND [RFC4861][RFC4862] procedures for DAD and Lookup.
The LLN may be a hub-and-spoke access link such as (Low-Power) IEEE STD. 802.11 (Wi-Fi) [IEEEstd80211] and IEEE STD. 802.15.1 (Bluetooth) [IEEEstd802151], or a Mesh-Under or a Route-Over network [RFC8505].

![Diagram of Backbone Link and 6LBR]

Figure 1: Backbone Link and 6LBR

A 6LBR provides registration services for the purpose of proactive IPv6 ND and maintains a registry of the active registrations as an abstract data structure called an Address Registrar. An entry in the Address Registrar is called an "Address Registration".

The Address Registration retains:

- the value for the ROVR associated to the registration, the current value of the TID, and the remaining Lifetime.
- a list of LLAs that are associated with the IPv6 address and can be used in a TLLAO as a response to a lookup.

Examples where more than one address may be available include the case of an anycast address and the case of an LLN address that is proxied by more than one 6BBR.

Unless otherwise configured, a 6LBR does the following:

- The 6LBR maintains an entry in the Address Registrar for any type of unicast and anycast addresses including those with link-local scope.
Based on that entry, it provides duplicate avoidance services within the scope of its Address Registrar.

The 6LBR also provides address lookup services for the Registered Address using unicast ICMPv6 DAR and DAC-based Address Mapping messages.

The Address Mapping messages can be exchanged using global unicast addresses as source and destination addresses, so they can be used for both on-link and off-link queries. NS and NA messages may also be used, but in that case the unicast source and destination addresses are link-local addresses and the 6LBR must be on-link.

The 6LBR proactive operations may coexist on the Backbone with reactive IPv6 ND [RFC4861][RFC4862] that rely on multicast for Duplicate Address Detection (DAD) and Address Lookup. Nodes that support this specification operate with the 6LBR before attempting the reactive operation, which may be avoided if the 6LBR is conclusive, either detecting a duplication or returning a mapping.

4. Updating RFC 8505

This specification leverages the capability to insert IPv6 ND options in the EDAR and EDAC messages that was introduced in [I-D.ietf-6lo-backbone-router].

It extends DAR and DAR ICMP messages for address lookup in Section 4.1.2 that use the same ICMP types as EDAR and EDAC but a different Code Prefix.

It also adds a new Status "Not Found" in Section 4.1.3) that indicates that the address being searched is not present in the Address Registrar.

A 6LBR signals itself by setting the "B" bit in the 6CIO of the RA messages that it generates [RFC8505]. This specification adds a new "A" bit in the 6CIO to indicate support of address mapping (see Section 4.1.1).

4.1. Extended Neighbor Discovery Options and Messages

This specification does not introduce new options; it modifies existing options and updates the associated behaviors.
4.1.1. Extending the Capability Indication Option

This specification defines a new capability bit for use in the 6CIO, as defined by [RFC7400] and extended in [RFC8505] for use in IPv6 ND messages.

The new "A" bit indicates that the 6LBR provides address mapping services per this specification.

```
+-------+-------+-------+-------+-------+-------+-------+-------+
| Type  | Length | Reserved | A | D | L | B | P | E | G |
+-------+-------+-----------+---+---+---+---+---+---+---+
| Reserved | Reserved | Reserved |   |   |   |   |   |   |   |
+-------+-------+-----------+---+---+---+---+---+---+---+
```

Figure 2: New Capability Bits in the 6CIO

Option Fields:

Type: 36
A: The 6LBR provides address mapping services.

4.1.2. New Code Prefix for Address Mapping Messages

The Extended Duplicate Address messages share a common base format defined in section 4.2 of [RFC8505], with the ICMP type respectively set to 157 and 158 that is inherited from the DAR and DAC messages defined in section 4.4 of [RFC6775]. The ICMP Code is split in two 4-bit fields, the Code Prefix and the Code Suffix, and the only Code Prefix defined in [RFC8505] is 0, signaling a DAD.

The Address Mapping messages use the same values for the ICMP Type as the corresponding Extended Duplicate Address messages. This specification adds the Code Prefix of 1 to signal Address Mapping. ICMP messages with the ICMP type set to 157 or 158, and a Code Prefix of 1 are thus respectively an Address Mapping Request (AMR) and an Address Mapping Confirm (AMC).

4.1.3. New ARO Status

The Extended Address Registration Option (EARO) is defined in section 4.1 of [RFC8505]. It contains a Status field that is common with the EDAR and EDAC messages defined in section 4.2 of [RFC8505].
This specification defines a new Status "Not Found" as indicated in Table 1

+-------+-----------------------------------------------------------+
| Value | Description                                               |
+-------+-----------------------------------------------------------+
| 0..10 | As defined in [RFC6775] and [RFC8505].                    |
|   11  | Not Found: The address is not present in the Address      |
|       | Registrar (value to be confirmed by IANA)                 |
+-------+-----------------------------------------------------------+

Table 1: EARO Status

The Status of "Not Found" can be used in an NA(EARO) and in an AMC messages as a response to an address lookup operation.

4.2. Address Mapping Messages

A 6LBR signals that support by setting the "B" bit in the 6CIO of the RA messages that it generates. A 6LBR that supports this specification MUST also set the "A" bit, indicating support of the Address Mapping messages for address lookup.

In the Address Mapping flow, the querier IPv6 Node uses an AMR message, which is characterized by an ICMPv6 Type of 157 and a Code Prefix of 1. When used on-link, the AMR message SHOULD carry a SLLAO indicating the LLA of the querier. The Code Suffix MUST be set to 0 indicating a ROVR Length of 64 bits. The ROVR, TID and Lifetime fields MUST be set to 0 and ignored by the receiver.

The 6LBR MUST respond with an AMC message, which is characterized by an ICMPv6 Type of 158 and a Code Prefix of 1.

- If the address is not present in the Address Registrar then the 6LBR MUST set the status to "Not Found". The Code Suffix MUST be set to 0 indicating a ROVR Length of 64 bits. The ROVR, TID and Lifetime fields MUST be set to 0 and ignored by the receiver.

- Else if the address is present in the Address Registrar then the AMC fields MUST be set from the ROVR, TID and remaining Lifetime values in the Address Registration and the Status MUST be set to 0.

- If at least one LLA is found in the Address Registration, then the 6LBR MUST place one in a TLLAO option in the AMC message.

The AMC is sent unicast the 6LBR to the querier.
4.3. IPv6 ND-based Address Lookup

A 6LBR that is deployed on-link SHOULD provide NS/NA-based services. It signals that support by setting the "L" bit in the 6CIO of the RA messages that it generates, indicating that it is a 6LR [RFC8505].

A 6LBR thus typically sets the "A", the "B", and the "L" bits when attached to a Backbone Link that it serves, as illustrated in Figure 1. In that case, the IPv6 Nodes and 6BBRs can use an NS/NA exchange with the 6LBR for both duplicate detection and lookup services.

The NS(Lookup) is sent unicast from link-local address of the querier to the link-local address of the 6LBR. It carries a SLLAO [RFC4861] and it MUST NOT carry an EARO option to avoid the confusion with a registration.

The 6LBR MUST respond with an NA message that contains an EARO.

- If the address is not present in the Address Registrar then the 6LBR MUST set the status to "Not Found". The ROVR, TID and Lifetime fields MUST be set to 0 and ignored by the receiver.

- Else if the address is present in the Address Registrar then the EARO fields MUST be set from the ROVR, TID and remaining Lifetime values in the Address Registration and the Status MUST be set to 0.

- If at least one LLA is found in the Address Registration, then the 6LBR MUST place one in a TLLAO option in the NA message.

The NA is sent unicast from link-local address of the 6LBR to the link-local address of the querier.

5. Backward Compatibility

6. Security Considerations

This specification extends [RFC8505], and the security section of that document also applies to this document. In particular, the link layer SHOULD be sufficiently protected to prevent rogue access.

7. IANA Considerations

Note to RFC Editor, to be removed: please replace "This RFC" throughout this document by the RFC number for this specification once it is allocated.
IANA is requested to make a number of changes under the "Internet Control Message Protocol version 6 (ICMPv6) Parameters" registry, as follows.

7.1. ICMP Codes

IANA is requested to create 2 new subregistries of the ICMPv6 "Code" Fields registry, which itself is a subregistry of the Internet Control Message Protocol version 6 (ICMPv6) Parameters for the ICMP codes.

The new subregistries relate to the ICMP type 157, Duplicate Address Request (shown in Table 2), and 158, Duplicate Address Confirmation (shown in Table 3), respectively. For those two ICMP types, the ICMP Code field is split into 2 subfields, the "Code Prefix" and the "Code Prefix". The new subregistries relate to the "Code Prefix" portion of the ICMP Code. The range of "Code Prefix" is 0..15 in all cases. The policy is "IETF Review" or "IESG Approval" [RFC8126] for both subregistries.

The new subregistries are to be initialized as follows:

<table>
<thead>
<tr>
<th>Code Prefix</th>
<th>Meaning</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Duplicate Address Detection</td>
<td>RFC 6775</td>
</tr>
<tr>
<td>1</td>
<td>Address Mapping</td>
<td>This RFC</td>
</tr>
<tr>
<td>2...15</td>
<td>Unassigned</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: New Code Prefixes for ICMP type 157 DAR message

<table>
<thead>
<tr>
<th>Code Prefix</th>
<th>Meaning</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Duplicate Address Detection</td>
<td>RFC 6775</td>
</tr>
<tr>
<td>1</td>
<td>Address Mapping</td>
<td>This RFC</td>
</tr>
<tr>
<td>2...15</td>
<td>Unassigned</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: New Code Prefixes for ICMP type 158 DAC message

7.2. New ARO Status values

IANA is requested to make additions to the Address Registration Option Status Values Registry as follows:
<table>
<thead>
<tr>
<th>ARO Status</th>
<th>Description</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Not Found</td>
<td>This RFC</td>
</tr>
</tbody>
</table>

Table 4: New ARO Status values

7.3. New 6LoWPAN Capability Bits

IANA is requested to make additions to the Subregistry for "6LoWPAN Capability Bits" as follows:

<table>
<thead>
<tr>
<th>Capability Bit</th>
<th>Description</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>AM Support (A bit)</td>
<td>This RFC</td>
</tr>
</tbody>
</table>

Table 5: New 6LoWPAN Capability Bits

8. Acknowledgments

9. References

9.1. Normative References


9.2. Informative References


[IEEEstd80211]
IEEE standard for Information Technology, "IEEE Standard for Information technology -- Telecommunications and information exchange between systems Local and metropolitan area networks -- Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications".

[IEEEstd802151]
IEEE standard for Information Technology, "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements. - Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)"

Authors’ Addresses

Pascal Thubert (editor)
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
MOUGINS - Sophia Antipolis 06254
FRANCE

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

Eric Levy-Abegnoli
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
MOUGINS - Sophia Antipolis 06254
FRANCE

Phone: +33 497 23 26 20
Email: elevyabe@cisco.com
IPv6 Neighbor Discovery on Wireless Networks
draft-thubert-6man-ipv6-over-wireless-03

Abstract

This document describes how the original IPv6 Neighbor Discovery and Wireless ND (WiND) can be applied on various abstractions of wireless media.

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 November 3, 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.
1. Introduction

IEEE STD. 802.1 [IEEEstd8021] Ethernet Bridging provides an efficient and reliable broadcast service for wired networks; applications and protocols have been built that heavily depend on that feature for their core operation. Unfortunately, Low-Power Lossy Networks (LLNs) and local wireless networks generally do not provide the broadcast capabilities of Ethernet Bridging in an economical fashion.

As a result, protocols designed for bridged networks that rely on multicast and broadcast often exhibit disappointing behaviours when employed unmodified on a local wireless medium (see [I-D.ietf-mboned-ieee802-mcast-problems]).

Wi-Fi [IEEEstd80211] Access Points (APs) deployed in an Extended Service Set (ESS) act as Ethernet Bridges [IEEEstd8021], with the property that the bridging state is established at the time of association. This ensures connectivity to the node (STA) and protects the wireless medium against broadcast-intensive Transparent Bridging reactive Lookups.

Thubert
Expires November 3, 2019
In other words, the association process is used to register the MAC Address of the STA to the AP. The AP subsequently proxies the bridging operation and does not need to forward the broadcast Lookups over the radio.

Like Transparent Bridging, IPv6 [RFC8200] Neighbor Discovery [RFC4861] [RFC4862] Protocol (IPv6 ND) is a reactive protocol, based on multicast transmissions to locate an on-link correspondent and ensure the uniqueness of an IPv6 address. The mechanism for Duplicate Address Detection (DAD) [RFC4862] was designed for the efficient broadcast operation of Ethernet Bridging. Since broadcast can be unreliable over wireless media, DAD often fails to discover duplications [I-D.yourtchenko-6man-dad-issues]. In practice, IPv6 addresses very rarely conflict because of the entropy of the 64-bit Interface IDs, not because address duplications are detected and resolved.

The IPv6 ND Neighbor Solicitation (NS) [RFC4861] message is used for DAD and address Lookup when a node moves, or wakes up and reconnects to the wireless network. The NS message is targeted to a Solicited-Node Multicast Address (SNMA) [RFC4291] and should in theory only reach a very small group of nodes. But in reality, IPv6 multicast messages are typically broadcast on the wireless medium, and so they are processed by most of the wireless nodes over the subnet (e.g., the ESS fabric) regardless of how few of the nodes are subscribed to the SNMA. As a result, IPv6 ND address Lookups and DADs over a large wireless and/or a LowPower Lossy Network (LLN) can consume enough bandwidth to cause a substantial degradation to the unicast traffic service [I-D.vyncke-6man-mcast-not-efficient].

Because IPv6 ND messages sent to the SNMA group are broadcasted at the radio MAC Layer, wireless nodes that do not belong to the SNMA group still have to keep their radio turned on to listen to multicast NS messages, which is a total waste of energy for them. In order to reduce their power consumption, certain battery-operated devices such as IoT sensors and smartphones ignore some of the broadcasts, making IPv6 ND operations even less reliable.

These problems can be alleviated by reducing the IPv6 ND broadcasts over wireless access links. This has been done by splitting the broadcast domains and by routing between subnets, at the extreme by assigning a /64 prefix to each wireless node (see [RFC8273]).

Another way is to proxy at the boundary of the wired and wireless domains the Layer-3 protocols that rely on MAC Layer broadcast operations. For instance, IEEE 802.11 [IEEEstd80211] situates proxy-ARP (IPv4) and proxy-ND (IPv6) functions at the Access Points (APs).
But proxying ND requires a perfect knowledge of the peer IPv6 addresses for which proxying is provided. In a generic fashion, radio connectivity changes with movements and variations in the environment, which makes forming and maintaining that knowledge a hard problem in the general case.

Discovering peer addresses by snooping the IPv6 ND protocol as proposed for SAVI [I-D.bi-savi-wlan] was found to be unreliable. An IPv6 address may not be discovered immediately due to a packet loss, or if a "silent" node is not currently using one of its addresses, e.g., a node that waits in wake-on-lan state. A change of state, e.g. due to a movement, may be missed or misordered, leading to unreliable connectivity and an incomplete knowledge of the set of peers.

Wireless ND (WiND) introduces a new approach to IPv6 ND that is designed to apply to the WLANs and WPANs types of networks. On the one hand, WiND avoids the use of broadcast operation for Address Resolution and Duplicate Address Detection, and on the other hand, WiND supports use cases where Subnet and MAC-level domains are not congruent, which is common in those types of networks unless a specific MAC-Level emulation is provided.

To achieve this, WiND applies routing inside the Subnets, which enables MultiLink Subnets. Hosts register their addresses to their serving routers with [RFC8505]. With the registration, routers have a complete knowledge of the hosts they serve and in return, hosts obtain routing services for their registered addresses. The registration is abstract to the routing protocol, and it can be protected to prevent impersonation attacks with [I-D.ietf-6lo-ap-nd].

The routing service can be a simple reflexion in a Hub-and-Spoke Subnet that emulates an IEEE Std 802.11 Infrastructure BSS at Layer 3. It can also be a full-fledge routing protocol, in particular RPL [RFC6550] that was designed to adapt to various LLNs such as WLAN and WPAN radio meshes with the concept of Objective Function. Finally, the routing service can also be ND proxy that emulates an IEEE Std 802.11 Infrastructure ESS at Layer 3. WiND specifies the IPv6 Backbone Router for that purpose in [I-D.ietf-6lo-backbone-router].

More details on WiND can be found in Section 4.1.

2. Acronyms

This document uses the following abbreviations:

6BBR: 6LoWPAN Backbone Router
6LBR: 6LoWPAN Border Router
6LN: 6LoWPAN Node
6LR: 6LoWPAN Router
ARO: Address Registration Option
DAC: Duplicate Address Confirmation
DAD: Duplicate Address Detection
DAR: Duplicate Address Request
EDAC: Extended Duplicate Address Confirmation
EDAR: Extended Duplicate Address Request
MLSN: Multi-Link Subnet
LLN: Low-Power and Lossy Network
NA: Neighbor Advertisement
NBMA: Non-Broadcast Multi-Access
NCE: Neighbor Cache Entry
ND: Neighbor Discovery
NDP: Neighbor Discovery Protocol
NS: Neighbor Solicitation
RPL: IPv6 Routing Protocol for LLNs
RA: Router Advertisement
RS: Router Solicitation
WiND: Wireless Neighbor Discovery
WLAN: Wireless Local Area Network
WPAN: Wireless Personal Area Network
3.  IP Models

3.1.  Physical Broadcast Domain

At the physical (PHY) Layer, a broadcast domain is the set of nodes that may receive a datagram that one sends over an interface, in other words the set of nodes in range of radio transmission. This set can comprise a single peer on a serial cable used as point-to-point (P2P) link. It may also comprise multiple peer nodes on a broadcast radio or a shared physical resource such as the legacy Ethernet shared wire.

On WLAN and WPAN radios, the physical broadcast domain is defined by a particular transmitter, as the set of nodes that can receive what this transmitter is sending. Literally every datagram defines its own broadcast domain since the chances of reception of a given datagram are statistical. In average and in stable conditions, the broadcast domain of a particular node can be still be seen as mostly constant and can be used to define a closure of nodes on which an upper-layer abstraction can be built.

A PHY-layer communication can be established between 2 nodes if their physical broadcast domains overlap.

On WLAN and WPAN radios, this property is usually reflexive, meaning that if B can receive a datagram from A, then A can receive a datagram from B. But there can be asymmetries due to power levels, interferers near one of the receivers, or differences in the quality of the hardware (e.g., crystals, PAs and antennas) that may affect the balance to the point that the connectivity becomes mostly unidirectional, e.g., A to B but practically not B to A. It takes a particular effort to place a set of devices in a fashion that all their physical broadcast domains fully overlap, and it can not be assumed in the general case. In other words, the property of radio connectivity is generally not transitive, meaning that A may be in range with B and B may be in range with C does not necessarily imply that A is in range with C.

We define MAC-Layer Direct Broadcast (DMC) a transmission mode where the broadcast domain that is usable at the MAC layer is directly the physical broadcast domain. IEEE 802.15.4 [IEEE802154] and IEEE 802.11 [IEEEstd80211] OCB (for Out of the Context of a BSS) are examples of DMC radios. This constrasts with a number of MAC-layer Broadcast Emulation schemes that are described in the next section.
3.2. MAC-Layer Broadcast Emulations

While a physical broadcast domain is constrained to a single shared wire, Ethernet Bridging emulates the broadcast properties of that wire over a whole physical mesh of Ethernet links. For the upper layer, the qualities of the shared wire are essentially conserved, with a reliable and cheap broadcast operation over a closure of nodes defined by their connectivity to the emulated wire.

In large switched fabrics, overlay techniques enable a limited connectivity between nodes that are known to a mapping server. The emulated broadcast domain is configured to the system, e.g., with a VXLAN network identifier (VNID). Broadcast operations on the overlay can be emulated but can become very expensive, and it makes sense to proactively install the relevant state in the mapping server as opposed to rely on reactive broadcast lookups.

An IEEE Std 802.11 Infrastructure Basic Service Set (BSS) also provides a closure of nodes as defined by the broadcast domain of a central Access Point (AP). The AP relays both unicast and broadcast packets and ensures a reflexive and transitive emulation of the shared wire between the associated nodes, with the capability to signal link-up/link-down to the upper layer. Within an Infrastructure BSS, the physical broadcast domain of the AP serves as emulated broadcast domain for all the nodes that are associated to the AP. Broadcast packets are relayed by the AP and are not acknowledged. To ensure that all nodes in the BSS receive the broadcast transmission, AP transmits at the slowest PHY speed. This translates into maximum co-channel interferences for others and longest occupancy of the medium, for a duration that can be 100 times that of a unicast. For that reason, upper layer protocols should tend to avoid the use of broadcast when operating over Wi-Fi.

In an IEEE Std 802.11 Infrastructure Extended Service Set (ESS), infrastructure BSSes are interconnected by a bridged network, typically running Transparent Bridging and Spanning tree Protocol. In the original model, the state in the Transparent Bridge is set by observing the source MAC address of the frames. When a state is missing for a destination MAC address, the frame is broadcasted with the expectation that the response will populate the state. This is a reactive operation, meaning that the state is populated reactively to a need for forwarding. It is also possible to send a gratuitous frame to advertise self throughout the bridged network, and that is also a broadcast. The process of the association prepares a bridging state proactively at the AP, so as to avoid the reactive broadcast lookup. It may also generates a gratuitous broadcast sourced at the MAC address of the STA to prepare or update the state in the Transparent Bridges. This model avoids the need of multicast over...
the wireless access, and it is only logical that IPv6 ND evolved
towards proposes similar methods at Layer-3 for its operation.

In some cases of WLAN and WPAN radios, a mesh-under technology (e.g.,
a IEEE 802.11s or IEEE 802.15.10) provides meshing services that are
similar to bridging, and the broadcast domain is well defined by the
membership of the mesh. Mesh-Under emulates a broadcast domain by
flooding the broadcast packets at Layer-2. When operating on a
single frequency, this operation is known to interfere with itself,
forcing deployment to introduce delays that dampen the collisions.
All in all, the mechanism is slow, inefficient and expensive.

Going down the list of cases above, the cost of a broadcast
transmissions becomes increasingly expensive, and there is a push to
rethink the upper-layer protocols so as to reduce the depency on
broadcast operations.

There again, a MAC-layer communication can be established between 2
nodes if their MAC-layer broadcast domains overlap. In the absence
of a MAC-layer emulation such as a mesh-under or an Infrastructure
BSS, the MAC-layer broadcast domain is congruent with that of the
PHY-layer and inherits its properties for reflexivity and
transitivity. IEEE 802.11p, which operates Out of the Context of a
BSS (DMC radios) is an example of a network that does not have a MAC-
Layer broadcast domain emulation, which means that it will exhibit
mostly reflexive and mostly non-transitive transmission properties.

3.3. Mapping the IPv6 Link Abstraction

IPv6 defines a concept of Link, Link Scope and Link-Local Addresses
(LLA), an LLA being unique and usable only within the Scope of a
Link. The IPv6 Neighbor Discovery (ND) [RFC4861][RFC4862] Duplicate
Adress Detection (DAD) process leverages a multicast transmission to
ensure that an IPv6 address is unique as long as the owner of the
address is connected to the broadcast domain. It must be noted that
in all the cases in this specification, the Layer-3 multicast
operation is always a MAC_Layer broadcast for the lack of a Layer-2
multicast operation that could handle a possibly very large number of
groups in order to make the unicast efficient. This means that for
every multicast packet regardless of the destination group, all nodes
will receive the packet and process it all the way to Layer-3.

On wired media, the Link is often confused with the physical
broadcast domain because both are determined by the serial cable or
the Ethernet shared wire. Ethernet Bridging reinforces that illusion
by provising a MAC-Layer broadcast domain that emulates a physical
broadcast domain over the mesh of wires. But the difference shows on
legacy Non-Broadcast Multi-Access (NBMA) such as ATM and Frame-Relay,
on shared links and on newer types of NBMA networks such as radio and composite radio-wires networks. It also shows when private VLANs or Layer-2 cryptography restrict the capability to read a frame to a subset of the connected nodes.

In mesh-under and Infrastructure BSS, the IP Link extends beyond the physical broadcast domain to the emulated MAC-Layer broadcast domain. Relying on Multicast for the ND operation remains feasible but becomes detrimental to unicast traffic, energy-inefficient and unreliable, and its use is discouraged.

On DMC radios, IP Links between peers come and go as the individual physical broadcast domains of the transmitters meet and overlap. The DAD operation cannot provide once and for all guarantees on the broadcast domain defined by one radio transmitter if that transmitter keeps meeting new peers on the go. The nodes may need to form new LLAs to talk to one another and the scope where LLA uniqueness can be dynamically checked is that pair of nodes. As long as there’s no conflict a node may use the same LLA with multiple peers but it has to revalidate DAD with every new peer node. In practice, each pair of nodes defines a temporary P2P link, which can be modeled as a sub-interface of the radio interface.

3.4. Mapping the IPv6 Subnet Abstraction

IPv6 also defines a concept of Subnet for Glocal and Unique Local Addresses. Addresses in a same Subnet share a same prefix and by extension, a node belongs to a Subnet if it has an interface with an address on that Subnet. A Subnet prefix is Globally Unique so it is sufficient to validate that an address that is formed from a Subnet prefix is unique within that Subnet to guarantee that it is globally unique. IPv6 aggregation relies on the property that a packet from the outside of a Subnet can be routed to any router that belongs to the Subnet, and that this router will be able to either resolve the destination MAC address and deliver the packet, or route the packet to the destination within the Subnet. If the Subnet is known as onlink, then any node may also resolve the destination MAC address and deliver the packet, but if the Subnet is not onlink, then a host that does not have an NCE for the destination will need to pass the packet to a router.

On IEEE Std. 802.3, a Subnet is often congruent with an IP Link because both are determined by the physical attachment to an Ethernet shared wire or an IEEE Std. 802.1 bridged broadcast domain. In that case, the connectivity over the Link is transitive, the Subnet can appear as onlink, and any node can resolve a destination MAC address of any other node directly using IPv6 Neighbor Discovery.
But an IP Link and an IP Subnet are not always congruent. In a shared Link situation, a Subnet may encompass only a subset of the nodes connected to the Link. In Route-Over Multi-Link Subnets (MLSN) [RFC4903], routers federate the Links between nodes that belong to the Subnet, the Subnet is not onlink and it extends beyond any of the federated Links.

The DAD and lookup procedures in IPv6 ND expects that a node in a Subnet is reachable within the broadcast domain of any other node in the Subnet when that other node attempts to form an address that would be a duplicate or attempts to resolve the MAC address of this node. This is why ND is only applicable for P2P and transit links, and requires extensions for other topologies.

4. Wireless ND

4.1. Introduction to WiND

Wireless Neighbor Discovery (WiND) [RFC6775][RFC8505][I-D.ietf-6lo-backbone-router][I-D.ietf-6lo-ap-nd] defines a new ND operation that is based on 2 major paradigm changes, proactive address registration by hosts to their attachment routers and routing to host routes (/128) within the subnet. This allows WiND to avoid the classical ND expectations of transit links and Subnet-wide broadcast domains.

The proactive address registration is performed with a new option in NS/NA messages, the Extended Address Registration Option (EARO) defined in [RFC8505]. This method allows to prepare and maintain the host routes in the routers and avoids the reactive NS(Lookup) found in IPv6 ND. This is a direct benefit for wireless Links since it avoids the MAC level broadcasts that are associated to NS(Lookup).

The EARO provides information to the router that is independent to the routing protocol and routing can take multiple forms, from a traditional IGP to a collapsed ub-and-Spoke model where only one router owns and advertises the prefix. [RFC8505] is already referenced for RIFT [I-D.ietf-rift-rift], RPL [RFC6550] with [I-D.thubert-roll-unaware-leaves] and IPv6 ND proxy [I-D.ietf-6lo-backbone-router].

WiND does not change IPv6 addressing [RFC4291] or the current practices of assigning prefixes to subnets. It is still typical to assign a /64 to a subnet and to use interface IDs of 64 bits. Duplicate Address detection within the Subnet is performed with a central registrar, using new ND Extended Duplicate Address messages (EDAR and EDAC) [RFC8505]. This operation modernizes ND for application in overlays with Map Resolvers and enables unicast
lookups [I-D.thubert-6lo-unicast-lookup] for addresses registered to the resolver.

WiND also enables to extend a legacy /64 on Ethernet with ND proxy over the wireless. This way nodes can form any address the want and move freely from an L3-AP (that is really a backbone router in bridging mode, more in [I-D.iietf-6lo-backbone-router]) to another, without renumbering. Backbone Routers federate multiple LLNs over a Backbone Link to form a MultiLink Subnet (MLSN). Backbone Routers placed along the LLN edge of the Backbone handle IPv6 Neighbor Discovery, and forward packets on behalf of registered nodes.

An LLN node (6LN) registers all its IPv6 Addresses using an NS(EARO) as specified in [RFC8505] to the 6BBR. The 6BBR is also a Border Router that performs IPv6 Neighbor Discovery (IPv6 ND) operations on its Backbone interface on behalf of the 6LNs that have registered addresses on its LLN interfaces without the need of a broadcast over the wireless medium.

WiND is also compatible with DHCPv6 and other forms of address assignment in which case it can still be used for DAD.

4.2. Links and Link-Local Addresses

For Link-Local Addresses, DAD is performed between communicating pairs of nodes. It is carried out as part of a registration process that is based on a NS/NA exchange that transports an EARO. During that process, the DAD is validated and a Neighbor Cache Entry (NCE) is populated with a single unicast exchange.

For instance, in the case of a Bluetooth Low Energy (BLE) [RFC7668][IEEEstd802151] Hub-and Spoke configuration, Uniqueness of Link local Addresses need only to be verified between the pairs of communicating nodes, a central router and a peripheral host. In that example, 2 peripheral hosts connected to the same central router can not have the same Link Local Address because the Binding Cache Entries (BCEs) would collide at the central router which could not talk to both over the same interface. The WiND operation is appropriate for that DAD operation, but the one from ND is not, because peripheral hosts are not on the same broadcast domain. On the other hand, Global and ULA DAD is validated at the Subnet Level, using a registrar hosted by the central router.

4.3. Subnets and Global Addresses

WiND extends IPv6 ND for Hub-and-Spoke (e.g., BLE) and Route-Over (e.g., RPL) Multi-Link Subnets (MLSNs).
In the Hub-and-Spoke case, each Hub-Spoke pair is a distinct IP Link, and a Subnet can be mapped on a collection of Links that are connected to the Hub. The Subnet prefix is associated to the Hub. Acting as 6LR, the Hub advertises the prefix as not-onlink to the spokes in RA messages Prefix Information Options (PIO). Acting as 6LNs, the Spokes autoconfigure addresses from that prefix and register them to the Hub with a corresponding lifetime. Acting as a 6LBR, the Hub maintains a binding table of all the registered IP addresses and rejects duplicate registrations, thus ensuring a DAD protection for a registered address even if the registering node is sleeping. Acting as 6LR, the Hub also maintains an NCE for the registered addresses and can deliver a packet to any of them for their respective lifetimes. It can be observed that this design builds a form of Layer-3 Infrastructure BSS.

A Route-Over MLSN is considered as a collection of Hub-and-Spoke where the Hubs form a connected dominating set of the member nodes of the Subnet, and IPv6 routing takes place between the Hubs within the Subnet. A single logical 6LBR is deployed to serve the whole mesh. The registration in [RFC8505] is abstract to the routing protocol and provides enough information to feed a routing protocol such as RPL as specified in [I-D.thubert-roll-unaware-leaves]. In a degraded mode, all the Hubs are connected to a same high speed backbone such as an Ethernet bridging domain where IPv6 ND is operated. In that case, it is possible to federate the Hub, Spoke and Backbone nodes as a single Subnet, operating IPv6 ND proxy operations [I-D.ietf-6lo-backbone-router] at the Hubs, acting as 6BBRs. It can be observed that this latter design builds a form of Layer-3 Infrastructure ESS.

5. WiND Applicability

WiND allows P2P, P2MP hub-and spoke, MAC-level broadcast domain emulation such as mesh-under and Wi-Fi BSS, and Route-Over meshes.

There is an intersection where Link and Subnet are congruent and where both ND and WiND could apply. These includes P2P, the MAC emulation of a PHY broadcast domain, and the particular case of always on, fully overlapping physical radio broadcast domain. But even in those cases where both are possible, WiND is preferable vs. ND because it reduces the need of broadcast (this is discussed in the introduction of [I-D.ietf-6lo-backbone-router]).

There are also numerous practical use cases in the wireless world where Links and Subnets are not P2P and not congruent:

- IEEE Std 802.11 infrastructure BSS enables one subnet per AP, and emulates a broadcast domain at L2. Infra ESS extends that and
recommends to use an IPv6 ND proxy [IEEEstd80211] to coexist with Ethernet connected nodes. WiND incorporates an ND proxy to serve that need and that was missing so far.

- BlueTooth is Hub-and-Spoke at the MAC layer. It would make little sense to configure a different subnet between the central and each individual peripheral node (e.g., sensor). Rather, [RFC7668] allocates a prefix to the central node acting as router (6LR), and each peripheral host (acting as a host (6LR) forms one or more address(es) from that same prefix and registers it.

- A typical Smartgrid networks puts together Route-Over MLSNs that comprise thousands of IPv6 nodes. The 6TiSCH architecture [I-D.ietf-6tisch-architecture] presents the Route-Over model over a [IEEEstd802154] Time-Slotted Channel-Hopping mesh, and generalizes it for multiple other applications. Each node in a Smartgrid network may have tens to a hundred others nodes in range. A key problem for the routing protocol is which other node(s) should this node peer with, because most of the possible peers do not provide added routing value. When both energy and bandwidth are constrained, talking to them is a bad idea and most of the possible P2P links are not even used. Peerings that are actually used come and go with the dynamics of radio signal propagation. It results that allocating prefixes to all the possible P2P Links and maintain as many addresses in all nodes is not even considered.

5.1. Case of LPWANs

LPWANs are by nature so constrained that the addresses and Subnets are fully pre-configured and operate as P2P or Hub-and-Spoke. This saves the steps of neighbor Discovery and enables a very efficient stateful compression of the IPv6 header.

5.2. Case of Infrastructure BSS and ESS

In contrast to IPv4, IPv6 enables a node to form multiple addresses, some of them temporary to elusive, and with a particular attention paid to privacy. Addresses may be formed and deprecated asynchronously to the association. Even if the knowledge of IPv6 addresses used by a STA can be obtained by snooping protocols such as IPv6 ND and DHCPv6, or by observing data traffic sourced at the STA, such methods provide only an imperfect knowledge of the state of the STA at the AP. This may result in a loss of connectivity for some IPv6 addresses, in particular for addresses rarely used and in a situation of mobility. This may also result in undesirable remanent state in the AP when a STA ceases to use an IPv6 address. It results
that snooping protocols is not a recommended technique and that it
should only be used as last resort.

The recommended alternate is to use the IPv6 Registration method
specified in p. By that method, the AP exposes its capability to
proxy ND to the STA in Router Advertisement messages. In turn, the
STA may request proxy ND services from the AP for one or more IPv6
addresses, using an Address Registration Option. The Registration
state has a lifetime that limits unwanted state remanence in the
network. The registration is optionally secured using
[I-D.ietf-6lo-ap-nd] to prevent address theft and impersonation. The
registration carries a sequence number, which enables a fast mobility
without a loss of connectivity.

The ESS mode requires a proxy ND operation at the AP. The proxy ND
operation must cover Duplicate Address Detection, Neighbor
Unreachability Detection, Address Resolution and Address Mobility to
transfer a role of ND proxy to the AP where a STA is associated
following the mobility of the STA. The proxy ND specification
associated to the address registration is
[I-D.ietf-6lo-backbone-router]. With that specification, the AP
participates to the protocol as a Backbone Router, typically
operating as a bridging proxy though the routing proxy operation is
also possible. As a bridging proxy, the proxy replies to NS lookups
with the MAC address of the STA, and then bridges packets to the STA
normally; as a routing proxy, it replies with its own MAC address and
then routes to the STA at the IP layer. The routing proxy reduces
the need to expose the MAC address of the STA on the wired side, for
a better stability and scalability of the bridged fabric.

5.3. Case of Mesh Under Technologies

The Mesh-Under provides a broadcast domain emulation with reflexive
and Transitive properties and defines a transit Link for IPv6
operations. It results that the model for IPv6 operation is similar
to that of a BSS, with the root of the mesh operating an Access Point
does in a BSS/ESS. While it is still possible to operate IPv6 ND,
the inefficiencies of the flooding operation make the IPv6 ND
operations even less desirable than in a BSS, and the use of WiND is
highly recommended.

5.4. Case of DMC radios

IPv6 over DMC radios uses P2P Links that can be formed and maintained
when a pair of DMC radios transmitters are in range from one another.
5.4.1. Using IPv6 ND only

DMC radios do not provide MAC level broadcast emulation. An example of that is OCB (outside the context of a BSS), which uses IEEE Std. 802.11 transmissions but does not provide the BSS functions.

It is possible to form P2P IP Links between each individual pairs of nodes and operate IPv6 ND over those Links with Link Local addresses. DAD must be performed for all addresses on all P2P IP Links.

If special deployment care is taken so that the physical broadcast domains of a collection of the nodes fully overlap, then it is also possible to build an IP Subnet within that collection of nodes and operate IPv6 ND.

The model can be stretched beyond the scope of IPv6 ND if an external mechanism avoids duplicate addresses and if the deployment ensures the connectivity between peers. This can be achieved for instance in a Hub-and-Spoke deployment if the Hub is the only router in the Subnet and the Prefix is advertised as not onlink.

5.4.2. Using Wireless ND

Though this can be achieved with IPv6 ND, WiND is the recommended approach since it uses more unicast communications which are more reliable and less impacting for other users of the medium.

Router and Hosts respectively send a compressed RA/NA with a SLLAO at a regular period. The period can be indicated in a RA as in an RA-Interval Option [RFC6275]. If available, the message can be transported in a compressed form in a beacon, e.g., in OCB Basic Safety Messages (BSM) that are nominally sent every 100ms. An active beaconing mode is possible whereby the Host sends broadcast RS messages to which a router can answer with a unicast RA.

A router that has Internet connectivity and is willing to serve as an Internet Access may advertise itself as a default router [RFC4191] in its RA. The NA/RA is sent over an Unspecified Link where it does not conflict to anyone, so DAD is not necessary at that stage.

The receiver instantiates a Link where the sender’s address is not a duplicate. To achieve this, it forms an LLA that does not conflict with that of the sender and registers to the sender using [RFC8505]. If the sender sent an RA(PIO) the receiver can also autoconfigure an address from the advertised prefix and register it.
The lifetime in the registration should start with a small value (X=RMin, TBD), and exponentially grow with each reregistration to a larger value (X=Rmax, TBD). The IP Link is considered down when (X=NbBeacons, TDB) expected messages are not received in a row. It must be noted that the Link flapping does not affect the state of the registration and when a Link comes back up, the active lifetime not elapsed registrations are still usable. Packets should be held or destroyed when the Link is down.

P2P Links may be federated in Hub-and-Spoke and then in Route-Over MLSNs as described above. More details on the operation of WiND and RPL over the MLSN can be found in section 3.1, 3.2, 4.1 and 4.2.2 of [I-D.ietf-6tisch-architecture].
An example Hub-and-Spoke is an OCB Road-Side Unit (RSU) that owns a prefix, provides Internet connectivity using that prefix to On-Board Units (OBUs) within its physical broadcast domain. An example of Route-Over MLSN is a collection of cars in a parking lot operating RPL to extend the connectivity provided by the RSU beyond its physical broadcast domain. Cars may then operate NEMO [RFC3963] for their own prefix using their address derived from the prefix of the RSU as CareOf Address.

6. IANA Considerations

This specification does not require IANA action.
7. Security Considerations

This specification refers to the security sections of IPv6 ND and WiND, respectively.

8. Acknowledgments

Many thanks to the participants of the 6lo WG where a lot of the work discussed here happened. Also ROLL, 6TiSCH, and 6LoWPAN.

9. References

9.1. Normative References

[I-D.ietf-6lo-ap-nd]

[I-D.ietf-6lo-backbone-router]


9.2. Informative References

[I-D.bi-savi-wlan]

[I-D.ietf-6tisch-architecture]

[I-D.ietf-mboned-ieee802-mcast-problems]

[I-D.ietf-rift-rift]

[I-D.thubert-6lo-unicast-lookup]

[I-D.thubert-roll-unaware-leaves]

[I-D.vyncke-6man-mcast-not-efficient]

IEEE standard for Information Technology, "IEEE Std. 802.15.4, Part. 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks".

IEEE standard for Information Technology, "IEEE Standard for Information technology -- Telecommunications and information exchange between systems Local and metropolitan area networks Part 1: Bridging and Architecture".

IEEE standard for Information Technology, "IEEE Standard for Information technology -- Telecommunications and information exchange between systems Local and metropolitan area networks-- Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications".

IEEE standard for Information Technology, "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements. - Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)".

IEEE standard for Information Technology, "IEEE Standard for Local and metropolitan area networks -- Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)".


Author’s Address

Pascal Thubert (editor)
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
MOUGINS - Sophia Antipolis 06254
FRANCE

Phone: +33 497 23 26 34
Email: pthubert@cisco.com