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

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

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

Abstract

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

Requirements Language

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

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

2. Conventions Used in This Document

2.1. Abbreviations

The following abbreviations are used in this document:

SID: Segment ID.

SL: Segment Left.

SR: Segment Routing.

SRH: Segment Routing Header.

SRv6: Segment Routing with IPv6 Data plane.

TC: Traffic Class.

ICMPv6: multi-part ICMPv6 messages [RFC4884].

2.2. Terminology and Reference Topology

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

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

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

Figure 1 Reference Topology

In the reference topology:

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

Nodes N3, N5 and N6 are classic IPv6 nodes.

Node N100 is a controller.

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

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

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

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

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

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

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

* SRH with SID list <S1, S2, S3> with SegmentsLeft = SL

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

* (payload) represents the the payload of the packet.

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

3. OAM Building Blocks

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

3.1. O-flag in Segment Routing Header

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

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

Where:

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

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

3.1.1. O-flag Processing

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

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

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

```
1. IF SRH.Flags.O-flag is one and local configuration permits THEN
       a. Make a copy of the packet.
       b. Send the copied packet, along with an accurate timestamp
          to the OAM process.      ;; Ref1
Ref1: An implementation SHOULD copy and record the timestamp as soon as
possible during packet processing. Timestamp is not carried in the packet
forwarded to the next hop.
```

3.2. OAM Segments

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

3.3. End.OP: OAM Endpoint with Punt

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

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

```
1.   Send the packet to the OAM process
```

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

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

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

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

1. Timestamp the packet ;; Ref1, Ref2

2. Send the packet, along with an accurate timestamp, to the OAM process.

Ref1: Timestamping SHOULD be done in hardware, as soon as possible during the packet processing.
Ref2: An implementation should not generate further ICMP error during local SID S processing. If local SID S processing requires generation of an ICMP error, the error is generated by the local OAM process.

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

3.5. SRH TLV

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

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

4. OAM Mechanisms

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

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

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

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

4.1. Ping

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

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

4.1.1. Classic Ping

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

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

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

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

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

Figure 2 A sample ping output at an SRv6 capable node

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

- o Node N1 initiates an ICMPv6 ping packet with SRH as follows (A:1::, B:2:C31)(A:5::, B:4:C52, B:2:C31, SL=2, NH = ICMPv6)(ICMPv6 Echo Request).

- o Node N2, which is an SRv6 capable node, performs the standard SRH processing. Specifically, it executes the END.X function (B:2:C31) and forwards the packet on link3 to N3.

- o Node N3, which is a classic IPv6 node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on DA B:4:C52 in the IPv6 header.

- o Node N4, which is an SRv6 capable node, performs the standard SRH processing. Specifically, it observes the END.X function (B:4:C52) with PSP (Penultimate Segment POP) on the echo request packet and removes the SRH and forwards the packet across link10 to N5.

- o The echo request packet at N5 arrives as an IPv6 packet without an SRH. Node N5, which is a classic IPv6 node, performs the standard IPv6/ ICMPv6 processing on the echo request and responds, accordingly.

4.1.2. Pinging a SID Function

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

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

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

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

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

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

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

- o Node N1 initiates an ICMPv6 ping packet with SRH as follows (A:1::, B:2:C31)(B:4:C52, B:4:OTP, B:2:C31; SL=2; NH=ICMPv6)(ICMPv6 Echo Request).

- o Node N2, which is an SRv6 capable node, performs the standard SRH processing. Specifically, it executes the END.X function (B:2:C31) on the echo request packet.

- o Node N3 receives the packet as follows (A:1::, B:4:OTP)(B:4:C52, B:4:OTP, B:2:C31 ; SL=1; NH=ICMPv6)(ICMPv6 Echo Request). Node N3, which is a classic IPv6 node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on DA B:4:OTP in the IPv6 header.

- o When node N4 receives the packet (A:1::, B:4:OTP)(B:4:C52, B:4:OTP, B:2:C31 ; SL=1; NH=ICMPv6)(ICMPv6 Echo Request), it processes the END.OTP SID, as described in the pseudocode in Section 3. The packet gets punted to the ICMPv6 process for processing. The ICMPv6 process checks if the next SID in SRH (the target SID B:4:C52) is locally programmed.

- o If the target SID is not locally programmed, N4 responses with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "SID not locally implemented (TBA)"); otherwise a success is returned.

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

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

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

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

- o Node N1 initiates an ICMPv6 ping packet with SRH as follows (A:1::, B:2:C31)(B:4:C52, B:2:C31; SL=1, Flags.O=1; NH=ICMPv6)(ICMPv6 Echo Request).

- o When node N2 receives the packet (A:1::, B:2:C31)(B:4:C52, B:2:C31; SL=1, Flags.O=1; NH=ICMPv6)(ICMPv6 Echo Request) packet, it processes the O-flag in SRH, as described in the pseudocode in Section 3. A time-stamped copy of the packet gets punted to the ICMPv6 process for processing. Node N2 continues to apply the B:2:C31 SID function on the original packet and forwards it, accordingly. As B:4:C52 is a USP SID, N2 does not remove the SRH. The ICMPv6 process at node N2 checks if its local SID (B:2:C31) is locally programmed or not and responds to the ICMPv6 Echo Request.

- o If the target SID is not locally programmed, N4 responses with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "SID not locally implemented (TBA)"); otherwise a success is returned. Please note that, as mentioned in Section 3, if node N2 does not support the O-flag, it simply ignores it and process the local SID, B:2:C31.

- o Node N3, which is a classic IPv6 node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on DA B:4:C52 in the IPv6 header.

- o When node N4 receives the packet (A:1::, B:4:C52)(B:4:C52, B:2:C31; SL=0, Flags.O=1; NH=ICMPv6)(ICMPv6 Echo Request), it processes the O-flag in SRH, as described in the pseudocode in Section 3. A time-stamped copy of the packet gets punted to the ICMPv6 process for processing. The ICMPv6 process at node N4 checks if its local SID (B:2:C31) is locally programmed or not and responds to the ICMPv6 Echo Request. If the target SID is not locally programmed, N4 responses with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "SID not locally implemented (TBA)"); otherwise a success is returned.

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

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

4.1.3. Error Reporting

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

- 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.

```
> 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.

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

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

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

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

- o Node N1 initiates a traceroute probe packet with a monotonically increasing value of hop count and SRH as follows (A:1::, B:2:C31)(B:4:C52, B:4:OTP, B:2:C31; SL=2; NH=UDP)(Traceroute probe).

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

- o When Node N2 receives the packet with hop-count > 1, it performs the standard SRH processing. Specifically, it executes the END.X function (B:2:C31) on the traceroute probe.

- o When node N3, which is a classic IPv6 node, receives the packet (A:1::, B:4:OTP)(B:4:C52, B:4:OTP, B:2:C31 ; HC=1, SL=1; NH=UDP)(Traceroute probe) with hop-count = 1, it processes the hop count expiry. Specifically, the node N3 responses with the ICMPv6 message (Type: "Time Exceeded", Code: "Time to Live exceeded in Transit").

- o When node N3, which is a classic IPv6 node, receives the packet with hop-count > 1, it performs the standard IPv6 processing. Specifically, it forwards the traceroute probe based on DA B:4:OTP in the IPv6 header.

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

Figure 4 displays a sample traceroute output for this example.

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

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

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

4.2.2.2. Tracing SRv6 Overlay

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

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

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

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

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

- 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 responses with the ICMPv6 message (Type: "SRv6 OAM (TBA)", Code: "O-flag punt at Transit (TBA)"). Please note that, as mentioned in Section 3, if node N2 does not support the O-flag, it simply ignores it and processes the local SID, B:2:C31.

- 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 RegistrySEC

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

TBA (suggested value: 162) SRv6 OAM Message.

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

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

6.2. SRv6 OAM Endpoint Types

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

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

7. Acknowledgements

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

8. Contributors

The following people have contributed to this document:

Robert Raszuk
Bloomberg LP
Email: robert@raszuk.net

John Leddy
Individual
Email: john@leddy.net

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

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

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

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

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

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

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

Cheng Li
Huawei
Email: chengli13@huawei.com

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

9. References

9.1. Normative References

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-21 (work in progress), June 2019.

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

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

9.2. Informative References

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

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <https://www.rfc-editor.org/info/rfc4884>.

[RFC5837] Atlas, A., Ed., Bonica, R., Ed., Pignataro, C., Ed., Shen, N., and JR. Rivers, "Extending ICMP for Interface and Next-Hop Identification", RFC 5837, DOI 10.17487/RFC5837, April 2010, <https://www.rfc-editor.org/info/rfc5837>.

[RFC8403] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC 8403, DOI 10.17487/RFC8403, July 2018, <https://www.rfc-editor.org/info/rfc8403>.

Authors' Addresses

Zafar Ali
Cisco Systems

Email: zali@cisco.com

Clarence Filsfils
Cisco Systems

Email: cfilsfil@cisco.com

Satoru Matsushima
Softbank

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

Daniel Voyer
Bell Canada

Email: daniel.voyer@bell.ca

Mach Chen
Huawei

Email: mach.chen@huawei.com

SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 10, 2023

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

July 10, 2022

Segment Routing Header encapsulation for In-situ OAM Data
draft-ali-spring-ioam-srv6-06

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

Copyright Notice

Table of Contents

1. Introduction

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

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

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

2. Conventions

2.1. Requirement Language

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

2.2. Abbreviations

Abbreviations used in this document:

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

3. OAM Metadata Piggybacked in Data Packets

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

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

3.1 IOAM Data Field Encapsulation in SRH

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

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  SRH-TLV-Type |     LEN       |    RESERVED                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-+
|  IOAM-Type    | IOAM HDR LEN  |    RESERVED                   |  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  I
!                                                               |  O
!                                                               |  A
~                 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.

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

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

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

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

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

4. Procedure

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

4.1. Ingress Node

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

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

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

4.2. Intermediate SR Segment Endpoint Node

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

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

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

4.3. Egress Node

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

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

5. IANA Considerations

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

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

6. Security Considerations

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

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

7. Acknowledgements

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

8. References

8.1. Normative References

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.

[I-D.spring-srv6-network-programming] Filsfils, C. et al. "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming, work in progress.

[I-D.6man-segment-routing-header] Previdi, S., Filsfils, C. et al, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header, work in progress.

[I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and Bernier, D., "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data, work in progress.

[I-D.spring-segment-routing-policy] Filsfils, C., et al., "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy, work in progress.

8.2. Informative References

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

Authors' Addresses

Zafar Ali
Cisco Systems, Inc.

Email: zali@cisco.com

Rakesh Gandhi
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Frank Brockners
Cisco Systems, Inc.
Germany

Email: fbrockne@cisco.com

Nagendra Kumar Nainar
Cisco Systems, Inc.

Email: naikumar@cisco.com

Carlos Pignataro
Cisco Systems, Inc.

Email: cpignata@cisco.com

Cheng Li
Huawei

Email: chengli13@huawei.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Gaurav Dawra
LinkedIn

Email: gdawra.ietf@gmail.com

6man
Internet-Draft
Intended status: Experimental
Expires: 25 December 2023

R. Bonica
Juniper Networks
Y. Kamite
NTT Communications Corporation
A. Alston
D. Henriques
Liquid Telecom
L. Jalil
Verizon
23 June 2023

The IPv6 Compact Routing Header (CRH)
draft-bonica-6man-comp-rtg-hdr-31

Abstract

This document describes an experiment in which two new IPv6 Routing headers are implemented and deployed. Collectively, they are called the Compact Routing Headers (CRH). Individually, they are called CRH-16 and CRH-32.

One purpose of this experiment is to demonstrate that the CRH can be implemented and deployed in a production network. Another purpose is to demonstrate that the security considerations, described in this document, can be addressed with access control lists. Finally, this document encourages replication of the experiment.

Status of This Memo

Table of Contents

1. Introduction

IPv6 [RFC8200] source nodes use Routing headers to specify the path that a packet takes to its destination. The IETF has defined several Routing header types [IANA-RH]. This document defines two new Routing header types. Collectively, they are called the Compact Routing Headers (CRH). Individually, they are called CRH-16 and CRH-32.

The CRH allows IPv6 source nodes to specify the path that a packet takes to its destination. The CRH:

* Can be encoded in relatively few bytes.

* Is designed to operate within a limited domain. (See Section 9).

The following are reasons for encoding the CRH in as few bytes as possible:

* Many ASIC-based forwarders copy headers from buffer memory to on-chip memory. As header sizes increase, 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 is excessive.

This document describes an experiment whose purposes are:

* To demonstrate that the CRH can be implemented and deployed.

* To demonstrate that the security considerations, described in this document, can be addressed with access control lists.

* To encourage replication of the experiment.

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 Headers (CRH)

Both CRH versions (i.e., CRH-16 and CRH-32) contain the following fields:

* Next Header - Defined in [RFC8200].

* Hdr Ext Len - Defined in [RFC8200].

* Routing Type - Defined in [RFC8200]. (CRH-16 value is 5. CRH-32 value is 6).

* Segments Left - Defined in [RFC8200].

* Type-specific Data - Described in [RFC8200].

In the CRH, the Type-specific data field contains a list of Segment Identifiers (SIDs). Each SID identifies an entry in the CRH Forwarding Information Base (CRH-FIB) (Section 4). Each CRH-FIB entry identifies an interface on the path that the packet takes to its destination.

SIDs are listed in reverse order. So, the first SID in the list represents the final interface in the path. Because segments are listed in reverse order, the Segments Left field can be used as an index into the SID list. In this document, the "current SID" is the SID list entry referenced by the Segments Left field.

The first segment in the path can be omitted from the list. See Appendix A for examples.

In the CRH-16 (Figure 1), each SID is encoded in 16-bits. In the CRH-32 (Figure 2), each SID is encoded in 32-bits.

In all cases, the CRH MUST end on a 64-bit boundary. So, the Type-specific data field MUST be padded with zeros if the CRH would otherwise not end on a 64-bit boundary.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             SID[0]            |          SID[1]               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          .........
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

Figure 1: CRH-16

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
+                            SID[0]                             +
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
+                            SID[1]                             +
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                              //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
+                            SID[n]                             +
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: CRH-32

4. The CRH Forwarding Information Base (CRH-FIB)

Each SID identifies a CRH-FIB entry.

Each CRH-FIB entry contains:

* An IPv6 address.

* A topological function.

* Arguments for the topological function. (Optional).

The first ten bits of the IPv6 address MUST NOT be fe00. That prefix is reserved for link-local [RFC6890] addresses.

The topological function specifies how the processing node forwards the packet to the interface identified by the IPv6 address. The following are examples:

* Forward the packet through the least-cost path to the interface identified by the IPv6 address (i.e., loose source routing).

* Forward the packet through a specified interface to the interface identified by the IPv6 address (i.e.,strict source routing)

Some topological functions require parameters. For example, a topological function might require a parameter that identifies the interface through which the packet is forwarded.

The CRH-FIB can be populated:

* By an operator, using a Command Line Interface (CLI).

* By a controller, using the Path Computation Element (PCE) Communication Protocol (PCEP) [RFC5440] or the Network Configuration Protocol (NETCONF) [RFC6241].

* By a distributed routing protocol [ISO10589-Second-Edition], [RFC5340], [RFC4271].

5. Processing Rules

The following rules describe CRH processing:

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

* If Hdr Ext Len indicates that the CRH is larger than the implementation can process, discard the packet and send an ICMPv6 [RFC4443] Parameter Problem, Code 0, message to the Source Address, pointing to the Hdr Ext Len field.

* Compute L, the minimum CRH length (Section 5.1).

* If 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 Segments Left field.

* Decrement Segments Left.

* Search for the current SID in the CRH-FIB. In this document, the "current SID" is the SID list entry referenced by the Segments Left field.

* If the search does not return a CRH-FIB entry, discard the packet and send an ICMPv6 Parameter Problem, Code 0, message to the Source Address, pointing to the current SID.

* If Segments Left is greater than 0 and the CRH-FIB entry contains a multicast address, discard the packet and send an ICMPv6 Parameter Problem, Code 0, message to the Source Address, pointing to the current SID.

* Copy the IPv6 address from the CRH-FIB entry to the Destination Address field in the IPv6 header.

* Decrement the IPv6 Hop Limit.

* Submit the packet, its topological function and its parameters to the IPv6 module. See NOTE.

NOTE: By default, the IPv6 module determines the next-hop and forwards the packet. However, the topological function may elicit another behavior. For example, the IPv6 module may forward the packet through a specified interface.

5.1. Computing Minimum CRH Length

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

* Routing Type (i.e., CRH-16 or CRH-32).

* Segments Left.

It yields L, the minimum CRH length. The minimum CRH length is measured in 8-octet units, not including the first 8 octets.

```
<CODE BEGINS>
switch(Routing Type) {
    case CRH-16:
        if (Segments Left <= 2)
            return(0)
        sidsBeyondFirstWord = Segments Left - 2;
        sidPerWord = 4;
    case CRH-32:
        if (Segments Left <= 1)
            return(0)
        sidsBeyondFirstWord = Segments Left - 1;
        sidsPerWord = 2;
    case default:
        return(0xFF);
    }

words = sidsBeyondFirstWord div sidsPerWord;
if (sidsBeyondFirstWord mod sidsPerWord)
    words++;

return(words)
<CODE ENDS>
```

6. Mutability

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

7. Applications And SIDs

A CRH contains one or more SIDs. Each SID is processed by exactly one node.

Therefore, a SID is not required to have domain-wide significance. Applications can:

* Allocate SIDs so that they have domain-wide significance.

* Allocate SIDs so that they have node-local significance.

8. Management Considerations

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

9. Security Considerations

Networks that process the CRH MUST NOT accept packets containing the CRH from outside of their limited domain. Domain border routers SHOULD discard packets that satisfy the following criteria:

* The packet contains a CRH

* The Segments Left field in the CRH has a value greater than 0

* The Destination Address field in the IPv6 header represents an interface that resides inside of the limited domain.

Many border routers cannot filter packets based upon the Segments Left value. These border routers MAY discard packets that satisfy the following criteria:

* The packet contains a CRH

* The Destination Address field in the IPv6 header represents an interface that resides inside of the limited domain.

10. Implementation and Deployment Status

Juniper Networks has produced experimental implementations of the CRH on the MX-series (ASIC-based) router

Liquid Telecom has produced experimental implementations of the CRH on software based routers.

The CRH has carried non-production traffic in CERNET and Liquid Telecom.

11. IANA Considerations

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

Value	Description	Reference
5	CRH-16	This document
6	CRH-32	This document

12. Acknowledgements

Thanks to Dr. Vanessa Ameen, Fernando Gont, Naveen Kottapalli, Joel Halpern, Tony Li, Xing Li, Gerald Schmidt, Nancy Shaw, Ketan Talaulikar, and Chandra Venkatraman for their contributions to this document.

13. Contributors

Gang Chen

Baidu

No.10 Xibeiwang East Road Haidian District

Beijing 100193 P.R. China

Email: phdgang@gmail.com

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

Gyan Mishra

Verizon

Silver Spring, Maryland, USA

Email: hayabusagsm@gmail.com

14. References

14.1. Normative References

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

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

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

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

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

14.2. Informative References

[IANA-RH] IANA, "Routing Headers", <https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xhtml#ipv6-parameters-3>.

[ISO10589-Second-Edition]
International Organization for Standardization, ""Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473)", ISO/IEC 10589:2002, Second Edition,", November 2001.

[RFC2151] Kessler, G. and S. Shepard, "A Primer On Internet and TCP/IP Tools and Utilities", FYI 30, RFC 2151, DOI 10.17487/RFC2151, June 1997, <https://www.rfc-editor.org/info/rfc2151>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6890] Cotton, M., Vegoda, L., Bonica, R., Ed., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC 6890, DOI 10.17487/RFC6890, April 2013, <https://www.rfc-editor.org/info/rfc6890>.

Appendix A. CRH Processing Examples

This appendix demonstrates CRH processing in the following scenarios:

* The SID list contains one entry for each segment in the path (Appendix A.1).

* The SID list omits the first entry in the path (Appendix A.2).

```
 -----------                 -----------                 -----------
|Node: S    |               |Node: I1   |               |Node: I2   |
|Loopback:  |---------------|Loopback:  |---------------|Loopback:  |
|2001:db8::a|               |2001:db8::1|               |2001:db8::2|
 -----------                 -----------                 -----------
      |                                                        |
      |                      -----------                       |
      |                     |Node: D    |                      |
       ---------------------|Loopback:  |---------------------
                            |2001:db8::b|
                             -----------
```

Figure 3: Reference Topology

Figure 3 provides a reference topology that is used in all examples.

SID	IPv6 Address	Forwarding Method
2	2001:db8::2	Least-cost path
11	2001:db8::b	Least-cost path

Table 1: Node SIDs

Table 1 describes two entries that appear in each node's CRH-FIB.

A.1. The SID List Contains One Entry For Each Segment In The Path

In this example, Node S sends a packet to Node D, via I2. In this example, I2 appears in the CRH segment list.

As the packet travels from S to I2:	
Source Address = 2001:db8::a	Segments Left = 1
Destination Address = 2001:db8::2	SID[0] = 11
	SID[1] = 2

Table 2

As the packet travels from I2 to D:	
Source Address = 2001:db8::a	Segments Left = 0
Destination Address = 2001:db8::b	SID[0] = 11
	SID[1] = 2

Table 3

A.2. The SID List Omits The First Entry In The Path

In this example, Node S sends a packet to Node D, via I2. In this example, I2 does not appear in the CRH segment list.

As the packet travels from S to I2:	
Source Address = 2001:db8::a	Segments Left = 1
Destination Address = 2001:db8::2	SID[0] = 11

Table 4

As the packet travels from I2 to D:	
Source Address = 2001:db8::a	Segments Left = 0
Destination Address = 2001:db8::b	SID[0] = 11

Table 5

Authors' Addresses

Ron Bonica
Juniper Networks
2251 Corporate Park Drive
Herndon, Virginia 20171
United States of America
Email: rbonica@juniper.net

Yuji Kamite
NTT Communications Corporation
3-4-1 Shibaura, Minato-ku,
108-8118
Japan
Email: y.kamite@ntt.com

Andrew Alston
Liquid Telecom
Nairobi
Kenya
Email: Andrew.Alston@liquidtelecom.com

Daniam Henriques
Liquid Telecom
Johannesburg
South Africa
Email: daniam.henriques@liquidtelecom.com

Luay Jalil
Verizon
Richardson, Texas
United States of America
Email: luay.jalil@one.verizon.com

6man
Internet-Draft
Intended status: Standards Track
Expires: April 12, 2021

R. Bonica
Juniper Networks

October 9, 2020

The Per-Segment Service Instruction (PSSI) Option
draft-bonica-6man-seg-end-opt-09

Abstract

This draft has been withdrawn..

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This draft has been withdrawn

2. Security Considerations

None.

3. IANA Considerations

None.

Author's Address

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

Email: rbonica@juniper.net

6man
Internet-Draft
Intended status: Standards Track
Expires: 18 July 2024

R. Bonica
Juniper Networks
Y. Kamite
NTT Communications Corporation
L. Jalil
Verizon
Y. Zhou
ByteDance
G. Chen
Baidu

15 January 2024

The IPv6 Tunnel Payload Forwarding (TPF) Option
draft-bonica-6man-vpn-dest-opt-21

Abstract

This document explains how IPv6 options can be used in IPv6 tunnels. It also defines the IPv6 Tunnel Payload Forwarding (TPF) 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 18 July 2024.

Table of Contents

1. Introduction

This document explains how IPv6 options [RFC8200] can be used in IPv6 tunnels. It also defines the IPv6 Tunnel Payload Forwarding (TPF) option.

An IPv6 tunnel [RFC2473] connects two nodes, called the entry-point and the exit-point. The entry-point receives a packet and encapsulates it in a Tunnel IPv6 Header. Figure 1 depicts the encapsulation.

```
                              +-----------------------------------//-----+
                              | Original |                                |
                              |          |   Original Packet Payload      |
                              | Header   |                                |
                              +-----------------------------------//-----+
                               <            Original Packet             >
                                                  |
                                                  v
       <Tunnel IPv6 Headers> <       Original Packet                    >

      +---------+ - - - - - +-------------------------//--------------+
      | IPv6    | IPv6      |                                         |
      |         | Extension |        Original Packet                  |
      | Header  | Headers   |                                         |
      +---------+ - - - - - +-------------------------//--------------+
       <                          Tunnel IPv6 Packet                 >
```

Figure 1: IPv6 Tunnel Encapsulation

The original packet can be any layer-2 or layer-3 packet (e.g., Ethernet, IPv4, IPv6). The Tunnel Header is an IPv6 header followed by zero or more extension headers. The resulting packet is a Tunnel IPv6 Packet.

The entry-point sends the Tunnel IPv6 Packet to the exit-point which then executes the following procedure:

* Process the Tunnel IPv6 Header.

* Remove the Tunnel IPv6 Header, exposing the original packet.

* Submit the original packet to the next-protocol engine.

The exit-point node processes the Tunnel IPv6 Header in strict left-to-right order. It processes the IPv6 header first and then processes extension headers in the order that they appear in the packet. The IPv6 header, and each extension header, includes a Next Header field. The last Next Header field processed identifies the next-protocol engine.

Entry-point nodes can send optional information to the next-protocol engine on the exit-point node. For example, the entry-point can indicate:

* The interface through which the next-protocol engine should send the packet.

* The routing table that the next-protocol engine should use to process the packet.

To send this information, the entry-point node includes an IPv6 Destination Option header in the Tunnel IPv6 Header. The IPv6 Destination Options header includes an IPv6 TPF option and the IPv6 TPF option includes TPF information. The next-protocol engine on the exit-point node uses TPF information when it forwards the original packet.

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 IPv6 Tunnel Payload Forwarding (TPF) Option

The TPF Option contains the following fields:

* Option Type: 8-bit selector. TPF option. Value TBD by IANA. (Suggested value: 0x41). 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.

* Option Data - 32-bits. Tunnel Payload Forwarding (TPF) Information.

The TPF 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.

NOTE : The highest-order two bits of the Option Type (i.e., the "act" bits) are 01. These bits specify the action taken by a destination node that does not recognize the option. The required action is to discard the packet. 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.

4. TPF Information Determines Next-Protocol Engine Behavior

An exit-point node supports one or more next-protocol engines (e.g., Ethernet, IPv4, IPv6). Each next-protocol engine supports a default forwarding procedure and zero or more special forwarding procedures.

When an exit-point node submits a packet to a next-protocol engine without TPF information, the next-protocol engine executes its default forwarding procedure. For example, assume that the exit-point node receives the following Tunnel IPv6 Packet:

* The Tunnel IPv6 Packet does not contain TPF information.

* The original packet is IPv4.

In this case, the exit-point node processes and removes the Tunnel IPv6 Header. It then submits the original packet, without any TPF information, to the IPv4 protocol engine.

The IPv4 protocol engine executes its default forwarding procedure. It searches its Forwarding Information Base (FIB) for and entry that matches the original packet's destination address. If the search returns a FIB entry, the protocol engine forwards the packet through an interface that the FIB entry identifies.

When an exit-point node submits a packet to a next-protocol engine with TPF information, the next-protocol engine executes a special forwarding procedure. For example, assume that the exit-point node receives the following Tunnel IPv6 packet:

* The Tunnel IPv6 Packet contains TPF information that identifies an interface.

* The original packet is IPv4.

In this case, the exit-point node processes and removes the Tunnel IPv6 Header. It then submits the original packet, along with TPF information, to the IPv4 protocol engine.

The IPv4 protocol engine executes a special forwarding procedure. It forwards the packet through the interface identified by TPF information, without searching the FIB.

5. TPF Information Semantics

TPF information is opaque. While it must be understood by the entry-point node and the exit-point node, it does not need to be understood by any other node.

6. Virtual Private Networking (VPN) Applications

The IPv6 TPF option is useful in deployments where IPv6 tunnels carry:

* Layer 3 Virtual Private Network (L3VPN) [RFC4364] traffic.

* Ethernet Virtual Private Network (EVPN) [RFC7432] traffic.

When an IPv6 tunnel carries L3VPN traffic, VPN context information can be encoded in an IPv6 TPF option. Therefore, the MPLS service label that is normally present in an L3VPN packet can be eliminated.

When an IPv6 tunnel carries EVPN traffic, VPN context information can be encoded in an IPv6 TPF option. Therefore, the UDP and VXLAN headers that might otherwise be present can be eliminated.

7. Security Considerations

TPF information MUST NOT be accepted from untrusted sources. The following are acceptable methods of risk mitigation:

* Authenticate the IPv6 TPF option using the IPv6 Authentication Header (AH) [RFC4302] or the IPv6 Encapsulating Security Payload (ESP) Header [RFC4303].

* Maintain a secure TPF domain.

All nodes at the edge of a secure TPF domain discard packets that satisfy the following criteria:

* Contain an IPv6 TPF option.

* Contain an IPv6 Destination Address that represents an interface inside of the secure TPF domain.

8. 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 "Tunnel Payload Forwarding Option". The "act" bits are 01 and the "chg" bit is 0. The suggested value is 0x41.

9. Acknowledgements

Thanks to Dr. Vanessa Ameen, Brian Carpenter, Adrian Farrel, Ishaan Gandhi, Tom Herbert, John Leddy, Srihari Sangli and Tony Li for their comments.

10. Contributors

Chris Lenart

Verizon

22001 Loudoun County Parkway

Ashburn, Virginia 20147 USA

Email: chris.lenart@verizon.com

Greg Presbury

Hughes Network Systems

11717 Exploration Lane

Germantown, Maryland 20876 USA

Email: greg.presbury@hughes.com

11. References

11.1. Normative References

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

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

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <https://www.rfc-editor.org/info/rfc4302>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

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

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

11.2. Informative References

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

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

Authors' Addresses

Ron Bonica
Juniper Networks
2251 Corporate Park Drive
Herndon, Virginia 20171
United States of America
Email: rbonica@juniper.net

Yuji Kamite
NTT Communications Corporation
3-4-1 Shibaura, Minato-ku,
108-8118
Japan
Email: y.kamite@ntt.com

Luay Jalil
Verizon
Richardson, Texas
United States of America
Email: luay.jalil@one.verizon.com

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

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

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

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

October 15, 2019

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

Abstract

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

Status of This Memo

Copyright Notice

Table of Contents

1. Overview

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

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

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

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

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

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

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

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

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

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

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

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

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

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

2. Requirements Language

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

3. Paths, Segments And Instructions

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

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

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

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

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

```
  ----      ----      ----      ----      ----      ----
 |Node|----|Node|----|Node|----|Node|----|Node|----|Node|
 | A  |    | B  |    | C  |    | D  |    | E  |    | F  |
  ----      ----      ----      ----      ----      ----
    |                   |         |                   |
     -------------------|         |-------------------|
        Segment A-C     |---------|    Segment D-F
                       Segment C-D
    |                                                 |
     -------------------------------------------------
                         SRm6 Path
```

Figure 1: Paths, Segments And Instructions

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

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

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

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

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

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

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

4. Segment Types

SRm6 supports the following segment types:

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

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

Each segment type is described below.

4.1. Adjacency Segments

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

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

- An interface identifier.

The topological instruction behaves as follows:

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

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

- Forward the packet through the above-mentioned interface.

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

4.2. Node Segments

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

The topological instruction behaves as follows:

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

- Overwrite the packet's Destination Address with the destination address that was received 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.

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

4.3. Binding Segments

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

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

The topological instruction behaves as follows:

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

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

5. Segment Identifiers (SID)

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

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

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

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

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

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

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

5.1. Range

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

- o 65,535 (i.e., 2**16 minus 1).
- o 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 SRm6 domain to have the smallest feasible MSV. The following paragraphs explain how an operator determines the smallest feasible MSV.

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

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

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

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

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

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

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

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

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

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

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

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

5.2. Assigning SIDs to Adjacency Segments

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

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

5.3. Assigning SIDs to Node Segments

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

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

5.4. Assigning SIDs to Binding Segments

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

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

6. Service Instructions

SRm6 supports the following service instruction types:

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

Each is described below.

6.1. Per-Segment

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

The following are examples of per-segment service instructions:

- o Expose a packet to a firewall policy.

- o Expose a packet to a sampling policy.

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

6.2. Per-Path

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

The following are examples of per-path service instructions:

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

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

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

7. The IPv6 Data Plane

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

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

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

- o If an SRm6 path is augmented by a per-path service instruction, the IPv6 header chain that represents it MUST contain a

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

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

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

7.1. The Routing Header

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

Both CRH versions contain the following fields:

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

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

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

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

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

- o Decrement Segments Left.

- o Search for the current SID in a local table that maps SID's to topological instructions. If the current SID cannot be found in that table, send an ICMPv6 Parameter Problem message to the packet's Source Address and discard the packet.

- o Execute the topological instruction found in the table as described in Section 4. This causes the packet to be forwarded to the segment egress node.

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

7.2. The Destination Options Header

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

Therefore, SRm6 defines the following new IPv6 options:

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

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

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

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

8. Control Plane

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

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

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

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

9. Differences Between SRv6 and SRv6+

9.1. Routing Header Size

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

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

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

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

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

Large Routing headers are undesirable for the following reasons:

- o Many ASIC-based forwarders copy the entire IPv6 extension header chain from buffer memory to on-chip memory. As the size of the IPv6 extension header chain increases, so does the cost of this copy.

- o Because Path MTU Discovery (PMTUD) [RFC8201] is not entirely reliable, many IPv6 hosts refrain from sending packets larger than the IPv6 minimum link MTU (i.e., 1280 bytes). When packets are small, the overhead imposed by large Routing headers becomes pronounced.

9.2. Decoupling of Topological and Service Instructions

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

```
                 ----------          ----------          ----------
                | Ethernet |        | Ethernet |        | Ethernet |
                 ----------          ----------          ----------
 Service        |  VXLAN   |        |  Dest    |        |   Dest   |
 Instruction     ----------         |          |        |          |
                |   UDP    |        |  Option  |        |  Option  |
                 ----------          ----------          ----------
 Topological    |                   |          |        |          |
 Instructions   |   CRH    |        |          |        |    CRH   |
                 ----------         |          |         ----------
                |   IPv6   |        |   IPv6   |        |   IPv6   |
                 ----------          ----------          ----------

                 Option 1            Option 2            Option 3
```

Figure 2: EVPN Design Alternatives

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

9.3. Authentication

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

9.4. Traffic Engineering Capability

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

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

9.5. IP Addressing Architecture

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

- o A network interface

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

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

10. Compliance

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

- o Be able to process IPv6 options as described in Section 4.2 of [RFC8200].

- o Be able to process the Routing header as described in Section 4.4 of [RFC8200].

- o Be able to process the Destination Options header as described in Section 4.6 of [RFC8200].

- o Support the CRH-16 and the CRH-32

Additionally, an SRm6 implementation MAY:

- o Recognize the Per-Segment Service Instruction Option.

- o Recognize the Per-Path Service Instruction Option.

11. Operational Considerations

11.1. Ping and Traceroute

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

11.2. ICMPv6 Rate Limiting

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

12. IANA Considerations

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

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

13. Security Considerations

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

- o Discard all inbound SRm6 packets whose IPv6 destination address represents domain infrastructure.

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

14. Acknowledgements

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

15. References

15.1. Normative References

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

[I-D.bonica-6man-comp-rtg-hdr]
Bonica, R., Kamite, Y., Niwa, T., Alston, A., Henriques, D., So, N., Xu, F., Chen, G., Zhu, Y., Yang, G., and Y. Zhou, "The IPv6 Compressed Routing Header (CRH)", draft-bonica-6man-comp-rtg-hdr-07 (work in progress), September 2019.

[I-D.bonica-6man-seg-end-opt]
Bonica, R., Halpern, J., Kamite, Y., Niwa, T., So, N., Xu, F., Chen, G., Zhu, Y., Yang, G., and Y. Zhou, "The Per-Segment Service Instruction (PSSI) Option", draft-bonica-6man-seg-end-opt-04 (work in progress), July 2019.

[I-D.bonica-6man-vpn-dest-opt]
Bonica, R., Kamite, Y., Lenart, C., So, N., Xu, F., Presbury, G., Chen, G., Zhu, Y., Yang, G., and Y. Zhou, "The Per-Path Service Instruction (PPSI) Option", draft-bonica-6man-vpn-dest-opt-06 (work in progress), July 2019.

[I-D.bonica-lsr-crh-isis-extensions]
Kaneriya, P., Shetty, R., Hegde, S., and R. Bonica, "IS-IS Extensions To Support The IPv6 Compressed Routing Header (CRH)", draft-bonica-lsr-crh-isis-extensions-00 (work in progress), May 2019.

[I-D.ssangli-idr-bgp-vpn-srv6-plus]
Ramachandra, S. and R. Bonica, "BGP based Virtual Private Network (VPN) Services over SRv6+ enabled IPv6 networks", draft-ssangli-idr-bgp-vpn-srv6-plus-02 (work in progress), July 2019.

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

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

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

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

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

15.2. Informative References

[I-D.filsfils-spring-net-pgm-extension-srv6-usid] Filsfils, C., Camarillo, P., Cai, D., Jiang, Z., daniel.voyer@bell.ca, d., Shawky, A., Leymann, N., Steinberg, D., Zandi, S., Dawra, G., Meilik, I., Uttaro, J., Jalil, L., So, N., Fiumano, M., and M. Khaddam, "Network Programming extension: SRv6 uSID instruction", draft-filsfils-spring-net-pgm-extension-srv6-usid-02 (work in progress), August 2019.

[I-D.ietf-6man-segment-routing-header] Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and d. daniel.voyer@bell.ca, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-24 (work in progress), October 2019.

[I-D.ietf-spring-segment-routing-mpls] Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-22 (work in progress), May 2019.

[I-D.ietf-spring-srv6-network-programming] Filsfils, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-04 (work in progress), October 2019.

[I-D.li-spring-compressed-srv6-np] Li, Z., Li, C., Peng, S., Wang, Z., and B. Liu, "Compressed SRv6 Network Programming", draft-li-spring-compressed-srv6-np-00 (work in progress), July 2019.

[I-D.mirsky-6man-unified-id-sr]
Cheng, W., Mirsky, G., Peng, S., Aihua, L., Wan, X., and C. Wei, "Unified Identifier in IPv6 Segment Routing Networks", draft-mirsky-6man-unified-id-sr-03 (work in progress), July 2019.

[RFC2151] Kessler, G. and S. Shepard, "A Primer On Internet and TCP/IP Tools and Utilities", FYI 30, RFC 2151, DOI 10.17487/RFC2151, June 1997, <https://www.rfc-editor.org/info/rfc2151>.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, DOI 10.17487/RFC3031, January 2001, <https://www.rfc-editor.org/info/rfc3031>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <https://www.rfc-editor.org/info/rfc4302>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

Authors' Addresses

Ron Bonica
Juniper Networks
Herndon, Virginia 20171
USA

Email: rbonica@juniper.net

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

Email: shraddha@juniper.net

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

Email: y.kamite@ntt.com

Andrew Alston
Liquid Telecom
Nairobi
Kenya

Email: Andrew.Alston@liquidtelecom.com

Daniam Henriques
Liquid Telecom
Johannesburg
South Africa

Email: daniam.henriques@liquidtelecom.com

Luay Jalil
Verizon
Richardson, Texas
USA

Email: luay.jalil@one.verizon.com

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

Email: joel.halpern@ericsson.com

Jen Linkova
Google
Mountain View, California 94043
USA

Email: furry@google.com

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

Email: phdgang@gmail.com

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: September 13, 2020

X. Geng
M. Chen
Huawei
Y. Zhu
China Telecom

March 12, 2020

DetNet SRv6 Data Plane Encapsulation
draft-geng-detnet-dp-sol-srv6-02

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

Copyright Notice

Table of Contents

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 :

o Source routing, which steers the DetNet flows through the network according to an explicit path with allocated resources;

o 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:

  SegmentList[0]=S3

  SegmentList[1]=S2

  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

```
              |                                          |
 ----IPv6--->|<---------------SRv6 DetNet------------->|<----IPv6---
              |                                          |
              |             +------+T2+----+             |
   +---+    +---+        +-+-+          +-+-+        +---+    +---+
   | E1+----| In|--+T1+--+R1 |          |R2 |--+T4+--| Eg+----+ E2|
   +---+    +---+        +-+-+          +-+-+        +---+    +---+
                           +-----+T3+-----+
```

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 optionally.

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.

If Flow Identification and Sequence Number are not carried in the SRH in the ingress, add it them in the SRH.

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.

SRH is as follows, which defined in [I-D.ietf-6man-segment-routing-header]

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Hdr Ext Len |  Routing Type |  Segment Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Last Entry  |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                       Segment List[0]                         |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                       Segment List[n]                         |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Optional TLVs                        |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The SRH specification allows the use of optional TLVs.

Each SRv6 Segment in the segment list 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.

as defined in [I-D.filsfils-spring-srv6-network-programming].

5.1. TLV Based SRv6 Data Plane Solution

5.1.1. Encapsulation

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:

Option 1: separated TLVs for flow identification and sequence number

```
 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            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        RESERVED       |           Flow Identification         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

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

```
 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            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|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.

Option 2: unified TLV for flow identification and sequence number

```
 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     |    Flow Identification        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       |                    Sequence Number                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             RESERVED                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

- o Type: 8 bits, to be assigned by IANA.
- o Length: 12.
- o Flow Identification: 20 bits, which is used for identifying DetNet flow.
- o Sequence Number: 28 bits, which is used for indicating sequence number of a DetNet flow.
- o RESERVED: 32 bits. MUST be 0 on transmission and ignored on receipt.

5.1.2. Replication SID and Elimination for DetNet

New SIDs, replication SID and elimination SID, are defined as follows:

5.1.2.1. Replication SID

Redundancy SID is a variation of binding SID defined in [I-D.ietf-spring-segment-routing-policy] Redundancy SID indicates the following operations:

- o Steering the packet into the corresponding redundancy policy
- o Packet replication based on the redundancy policy, e.g., the number of replication copies
- o Encapsulate the packet with necessary meta data (e.g., flow identification, sequence number) if it is not included in the original packet

When N receives a packet whose IPv6 DA is S and S is a Replication SID, N could do:

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. Elimination SID

Elimination SID indicates the following operations:

- o Packet elimination: forward the first received packets and eliminate the redundant packets.

- o Packet ordering(optional): reorder the packets if the packet arrives out of order

When N receives a packet whose IPv6 DA is S and S is a Elimination SID, N could do:

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
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Hdr Ext Len |  Routing Type |  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:

- o 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]);

- o FLOW IDENTIFICATION: 20 bits, in the DetNet TLVs in the SRH, used for DetNet flow identification in the DetNet relay node;

- o 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. Encapulation

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

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Hdr Ext Len |  Routing Type |  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]
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., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-26 (work in progress), October 2019.

[I-D.ietf-detnet-architecture]
Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-13 (work in progress), May 2019.

[I-D.ietf-detnet-dp-sol-mpls]
Korhonen, J. and B. Varga, "DetNet MPLS Data Plane Encapsulation", draft-ietf-detnet-dp-sol-mpls-02 (work in progress), March 2019.

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Sivabalan, S., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-06 (work in progress), December 2019.

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

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

Authors' Addresses

Xuesong Geng
Huawei

Email: gengxuesong@huawei.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Yongqing Zhu
China Telecom

Email: zhuyq@gsta.com

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: January 6, 2020

R. Hinden
Check Point Software
G. Fairhurst
University of Aberdeen

July 5, 2019

IPv6 Minimum Path MTU Hop-by-Hop Option
draft-hinden-6man-mtu-option-02

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.

carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

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.

```
+--------+         +----+        +----+         +-------+
|        |         |    |        |    |         |       |
| Sender +---------+ R1 +--------+ R2 +-------- + Dest. |
|        |         |    |        |    |         |       |
+--------+  MTU-S  +----+  9000B +----+  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.

	MTU-S	MTU-D	R1	R2	Rec PMTU	Note
1	9000B	9000B	H	H	9000 B	Endpoints attempt to use an 9000 B PMTU.
2	9000B	1500B	H	H	1500 B	Endpoints attempt to use a 1500 B PMTU.
3	9000B	1500B	H	-	9000 B	Endpoints attempt to use an 9000 B PMTU, but need to implement a method to fall back use a 1500 B PMTU.

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

o 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.

- o 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 the 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    Option    Option
  Type    Data Len   Data
+--------+--------+--------+--------+---------+-------+-+
|BBCTTTTT|00000100|     Min-PMTU    |     Rtn-PMTU    |R|
+--------+--------+--------+--------+---------+-------+-+
```

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 the 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 mimimum 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 0xFFFE, 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:

- o 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:

- o 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.

- o 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.

- o 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:

- o 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.

- o 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:

- o 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.

- o 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.

- o 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.

- 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, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

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

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

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

11.2. Informative References

[RFC1063] Mogul, J., Kent, C., Partridge, C., and K. McCloghrie, "IP MTU discovery options", RFC 1063, DOI 10.17487/RFC1063, July 1988, <https://www.rfc-editor.org/info/rfc1063>.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <https://www.rfc-editor.org/info/rfc1191>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <https://www.rfc-editor.org/info/rfc2460>.

[RFC7637] Garg, P., Ed. and Y. Wang, Ed., "NVGRE: Network Virtualization Using Generic Routing Encapsulation", RFC 7637, DOI 10.17487/RFC7637, September 2015, <https://www.rfc-editor.org/info/rfc7637>.

[RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <https://www.rfc-editor.org/info/rfc8085>.

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

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 19, 2020

T. Herbert
Intel

March 18, 2020

ICMPv6 errors for discarding packets due to processing limits
draft-ietf-6man-icmp-limits-08

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 several new 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 use the information to diagnose packet loss and may modify what it sends in future packets to avoid subsequent packet discards.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document specifies several new 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 to supplement those defined in [RFC4443]. Six 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 maximum length of 2048 octets and must fit into a single packet. 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 the Path MTU from the source to the destination. 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 headers 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 Options 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 six Parameter Problem codes that may be sent by a node that discards a packet due to processing limits of extension headers being exceeded. The information in these ICMPv6 errors may be used for diagnostics to determine why packets are being dropped. Additionally, a source node that receives these ICMPv6 errors may be able to 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

Some 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 will 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. The information in this ICMPv6 error may be used for diagnostics to determine why packets are being dropped. Additionally, a source node that receives this ICMPv6 error may be able to modify the headers used in subsequent packets to try to avoid further occurrences of packets being discarded.

1.3. 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, the sending of ICMP errors is RECOMMENDED even in cases where a node might be discarding packets per a nonconformant behavior.

1.4. Terminology

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. ICMPv6 errors for extension header limits

Six new codes are defined for the Parameter Problem type.

2.1. Format

The format of the ICMPv6 Parameter Problem message [RFC4443] 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 [RFC8200]         |
```

IPv6 Header 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)

TBA1 - Unrecognized Next Header type encountered by intermediate node
TBA2 - Extension header too big
TBA3 - Extension header chain too long
TBA4 - Too many extension headers
TBA5 - Too many options in extension header
TBA6 - 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 Invoking Packet if the field exceeding the limit is beyond what can fit in the maximum size of an ICMPv6 error message extension. If the pointer is used as an offset to read the data in the invoking packet then a node MUST first validate that the pointer value is less than the length of the Invoking Packet data.

2.2. Unrecognized Next Header type encountered by intermediate node (code TBA1)

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 this code is sent by intermediate nodes, and SHOULD NOT be sent by a final destination. If a final destination node observes an unrecognized header then it SHOULD send an ICMP Parameter Problem message with an ICMP Code value of 1 ("unrecognized Next Header type encountered") as specified in [RFC8200].

2.3. Extension header too big (code TBA2)

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 TBA3)

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 a limit on the total size in octets of the header chain. The ICMPv6 Pointer is set to first octet beyond the limit.

2.5. Too many extension headers (code TBA4)

An ICMPv6 Parameter Problem with code for "too many extension headers" SHOULD be sent when a node discards a packet with an extension header chain that exceeds a limit on the number of extension headers in the chain. The ICMPv6 Pointer is set to the offset of first octet of the first extension header that is beyond the limit.

2.6. Too many options in extension header (code TBA5)

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.

2.7. Option too big (code TBA6)

An ICMPv6 Parameter Problem with code for "option too big" is sent in two different cases: when the length of an individual Hop-by-Hop or Destination option exceeds a limit, or when the length or number of consecutive Hop-by-Hop or Destination padding options exceeds a limit. In the case that the length of an option exceeds a processing limit, the ICMPv6 Pointer field is set to the offset of the first octet of the option that exceeds the limit. In the cases that the length or number of padding options exceeds a limit, 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.

This ICMP error may be applied to other headers in a packet than just the IPv6 header or IPv6 extension headers. Therefore, a Destination Unreachable type with a multi-part ICMPv6 message format is used in lieu of the Parameter Problem type which only indicates errors concerning IPv6 headers.

3.1. Format

The error for aggregate header limits employs a multi-part ICMPv6 message format as defined in [RFC4884]. The extension object class "Extended Information" is defined to contain objects for ancillary information pertaining to an ICMP Destination Unreachable error. Within this object class, the sub-type "Pointer" is defined which contains the Pointer field with similar semantics to the Pointer field in ICMP Parameter Problem errors.

The format of the ICMPv6 message for an aggregate header limit exceeded 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             | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ I
|    Length     |                  Unused                       | C
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ M
|                           Invoking Packet                     | P
~      As much of invoking packet as possible without the       ~ |
|    ICMPv6 packet exceeding the minimum IPv6 MTU [RFC8200]     |/
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+/
|Version|       Reserved        |           Checksum            |\
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ E
|             Length            |   Class-Num   |   C-Type      | X
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ T
|                            Pointer                            | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+/
```

IPv6 Header 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)
TBA7 - Headers too long

Length
Length of the padded Invoking Packet measured in 64-bit words. The ICMP extension structure immediately follows the padded Invoking Packet

Invoking Packet
Contains as much of invoking packet as possible without the ICMPv6 packet exceeding the minimum IPv6 MTU. The Invoking Packet MUST be zero padded to the nearest 64-bit boundary [RFC4884]. If the original invoking packet did not contain 128 octets, the Invoking Packet MUST be zero padded to 128 octets.

ICMP Extension Fields:

Version
2 - per [RFC4884]

Reserved
0

Checksum
The one's complement checksum of the ICMP extension [RFC4884]

Length
8 - length of the object header and Pointer field

Class-Num
TBA8 - Extended Information class

C-Type
TBA9 - Pointer sub-type

Pointer
Identifies the octet offset within the invoking packet where a limit was exceeded.

The pointer will point beyond the end of the Invoking Packet if the field exceeding the limit is beyond what can fit in the maximum size of an ICMPv6 error message with the ICMP extension. If the pointer is used as an offset to read the data in the invoking packet then a node MUST first validate that the pointer value is less than the length of the Invoking Packet data.

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.

This error is sent in response to a node dropping a packet because the aggregate header chain exceeds the processing limits of a node. The aggregate header chain may be composed of protocol headers other than an IPv6 header and IPv6 extension headers. For instance, in the case of a node parsing a UDP encapsulation protocol, the encapsulating UDP header would be considered to be in the aggregate header chain.

As noted in section 4.1, the ICMPv6 Destination Unreachable error with code for "headers too long" has the lowest precedence of the ICMP errors discussed in this specification. If a packet contains an error corresponding to a Parameter Problem code then a node SHOULD send the Parameter Problem error instead of sending the ICMPv6 Destination Unreachable error with code for "headers too long".

4. Operation

Nodes that send or receive ICMPv6 errors due to header processing limits MUST 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) Existing ICMP errors defined in [RFC4443]

2) "Unrecognized Next Header type encountered by 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" headers exceeding a limit

8) "Too many extension headers"

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 appropriate action as suggested below.

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 to modify its use of extension headers or headers to avoid further packet discards.

* A host system SHOULD take appropriate action if it is creating packets with extension headers 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. 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 deduce the reason that packets are being discarded. For instance, the messages may be correlated with information attained through Packetization Layer Path MTU Discovery (PLMTUD) [RFC4821] or Happy Eyeballs for IPv6 [RFC8305]. Details of the interaction with alternative mechanisms are out of scope of this specification.

5.2. Processing limits

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

5.2.1. Long headers and header chains

The trend towards longer and more complex headers and header chains needing to be processed by end nodes, as well as intermediate nodes, is driven by:

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

* Deployment of routing headers. For instance, [I-D.ietf-6man-segment-routing-header] 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 [I-D.ioametal-ippm-6man-ioam-ipv6-options].

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

5.2.2. At end hosts

End 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.

5.2.3. 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. Processing of a TLV list might be implemented as an unrolled loop so that the number of TLVs that can be processed is limited.

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 a means to covertly deduce processing capabilities of nodes. 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" [IANA-PARAMPROB]:

* Unrecognized Next Header type encountered by intermediate node (value TBA1)

* Extension header too big (value TBA2)

* Extension header chain too long (value TBA3)

* Too many extension headers (value TBA4)

* Too many options in extension header (value TBA5)

* Option too big (value TBA6)

7.2. Destination Unreachable codes

IANA is requested to assign the following code for ICMPv6 type 1 "Destination Unreachable" [IANA-DESTUNREACH]:

* Headers too long (value TBA7)

7.3. ICMP Extension Object Classes and Class Sub-types

IANA is requested to assign the following Class value in the "ICMP Extension Object Classes and Class Sub-types" registry [IANA-ICMPEXT]:

* Extended information (value TBA8)

IANA is requested to create a Sub-type registry for the "Extended information" ICMP extension object class. The registration procedure for this registry shall be "Standards Action". The Sub-type value of 0 shall be reserved, values greater than zero may be assigned.

IANA is requested to assign the following Sub-type within the "Extended information" ICMP extension object class:

* Pointer (value TBA9)

8. Acknowledgments

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

9. References

9.1. Normative References

[IANA-DESTUNREACH]
"Internet Control Message Protocol version 6 (ICMPv6) Parameters, Type 1 - Destination Unreachable", <https://www.iana.org/assignments/icmpv6-parameters/icmpv6-parameters.xhtml#icmpv6-parameters-codes-2>.

[IANA-ICMPEXT]
"ICMP Extension Object Classes and Class Sub-types", <https://www.iana.org/assignments/icmp-parameters/icmp-parameters.xhtml#icmp-parameters-ext-classes>.

[IANA-PARAMPROB]
"Internet Control Message Protocol version 6 (ICMPv6) Parameters, Type 4 - Parameter Problem", <https://www.iana.org/assignments/icmpv6-parameters/icmpv6-parameters.xhtml#icmpv6-parameters-codes-5>.

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

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <https://www.rfc-editor.org/info/rfc4884>.

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, December 2013, <https://www.rfc-editor.org/info/rfc7045>.

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

9.2. Informative References

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-26 (work in progress), October 2019.

[I-D.ioametal-ippm-6man-ioam-ipv6-options]
Bhandari, S., Brockners, F., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., Spiegel, M., Krishnan, S., and R. Asati, "In-situ OAM IPv6 Options", draft-ioametal-ippm-6man-ioam-ipv6-options-02 (work in progress), March 2019.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <https://www.rfc-editor.org/info/rfc4821>.

[RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", RFC 4890, DOI 10.17487/RFC4890, May 2007, <https://www.rfc-editor.org/info/rfc4890>.

[RFC8305] Schinazi, D. and T. Pauly, "Happy Eyeballs Version 2: Better Connectivity Using Concurrency", RFC 8305, DOI 10.17487/RFC8305, December 2017, <https://www.rfc-editor.org/info/rfc8305>.

[RFC8504] Chown, T., Loughney, J., and T. Winters, "IPv6 Node Requirements", BCP 220, RFC 8504, DOI 10.17487/RFC8504, January 2019, <https://www.rfc-editor.org/info/rfc8504>.

Author's Address

Tom Herbert
Intel
Santa Clara, CA
USA

Email: tom@quantonium.net

IPv6 Maintenance
Internet-Draft
Intended status: Standards Track
Expires: June 21, 2020

L. Colitti
J. Linkova
Google

December 19, 2019

Discovering PREF64 in Router Advertisements
draft-ietf-6man-ra-pref64-09

Abstract

This document specifies a Neighbor Discovery option to be used in Router Advertisements to communicate NAT64 prefixes to hosts.

Status of This Memo

Copyright Notice

Table of Contents

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 Neighbor Discovery [RFC4861] option to be used in Router Advertisements to communicate NAT64 prefixes to hosts.

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

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:

- o Enabling DNS64 functions on end hosts. In particular:
  - * Local DNSSEC validation (DNS64 in stub-resolver mode). 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." Therefore to perform the DNS64 function the stub resolver needs to know the NAT64 prefix. This is required in order to use DNSSEC on a NAT64 network.
  - * 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 [RFC7858] or DNS-over-HTTPS [RFC8484] server 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.
- o Enabling NAT64 address translation functions on end hosts. For example:
  - * IPv4 address literals on an IPv6-only host. As described in [RFC8305] section 7.1, IPv6-only hosts connecting to IPv4 address literals can translate the IPv4 literal to an IPv6 literal.
  - * 464XLAT [RFC6877]. 464XLAT requires the host be aware of the NAT64 prefix.

3. Why include the NAT64 prefix in Router Advertisements

Fate sharing: NAT64 requires routing to be configured. IPv6 routing configuration requires receiving an IPv6 Router Advertisement [RFC4861]. Therefore using Router Advertisements to provide hosts with NAT64 prefix ensures that NAT64 reachability information shares fate with the rest of network configuration on the host.

Atomic configuration: including the NAT64 prefix in the Router Advertisement minimizes the number of packets required to configure a

host. Only one packet (a Router Advertisement) is required to complete the network configuration. 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 Neighbor Discovery [RFC4861] so just a minor extension to the existing implementation is required. Other options such as [RFC7225] require implementing other protocols (e.g. PCP [RFC7225]) which could be considered an obstacle for deployment.

4. Option 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     |     Scaled Lifetime     | PLC |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|              Highest 96 bits of the Prefix                    |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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. The sender MUST set the length to 2. The receiver MUST ignore the PREF64 option if the length field value is not 2.
Scaled Lifetime	13-bit unsigned integer. The maximum time in units of 8 seconds over which this NAT64 prefix MAY be used. See Section 4.1 for the Scaled Lifetime field processing rules.
PLC (Prefix Length Code)	3-bit unsigned integer. This field encodes the NAT64 Prefix Length defined in [RFC6052]. The PLC field values 0, 1, 2, 3, 4 and 5 indicate the NAT64 prefix length of 96, 64, 56, 48, 40 and 32 bits respectively. The receiver MUST ignore the PREF64 option if the prefix length code field is not set to one of those values.
Highest 96 bits of the prefix	96-bit unsigned integer. Contains bits 0 - 95 of the NAT64 prefix.

4.1. Scaled Lifetime Processing

It would be highly undesirable for the NAT64 prefix to have a lifetime shorter than the Router Lifetime, which is defined in the Section 4.2 of [RFC4861] as 16-bit unsigned integer. If the NAT64 prefix lifetime is not at least equal to the default router lifetime it might lead to scenarios when the NAT64 prefix lifetime expires before the arrival of the next unsolicited RA. Therefore the Scaled Lifetime encodes the NAT64 prefix lifetime in units of 8 seconds. The receiver MUST multiply the Scaled Lifetime value by 8 (for example, by logical left shift) to calculate the maximum time in seconds the prefix MAY be used. The maximum lifetime of the NAT64 prefix is thus 65528 seconds. To ensure that the NAT64 prefix does not expire before the default router, when using this option it is NOT RECOMMENDED to configure default router lifetimes greater than 65528 seconds. Lifetime of 0 indicates that the prefix SHOULD NOT be used anymore.

The value of the Scaled Lifetime field SHOULD by default be set to the lesser of 3 x MaxRtrAdvInterval ([RFC4861]) divided by 8, or 8191.

Router vendors SHOULD allow administrators to specify non-zero lifetime values which are not divisible by 8. In such cases the

router SHOULD round the provided value up to the nearest integer that is divisible by 8 and smaller than 65536, then divide the result by 8 (or perform a logical right-shift by 3), and set the Scaled Lifetime field to the resulting value. If such a non-zero lifetime value to be divided by 8 (to be subjected to a logical right-shift by 3) is less than 8 then the Scaled Lifetime field SHOULD be set to 1. This last step ensures that lifetimes under 8 seconds are encoded as a non-zero Scaled Lifetime.

5. Usage Guidelines

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 specific sub-prefixes of the NAT64 prefix to those devices. For example, suppose an operator is using the [RFC1918] address space 10.0.0.0/8 internally. That operator might want to route 10.0.0.0/8 through NAT64 device A, and the rest of the IPv4 space through NAT64 device B. If 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::/96 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.

5.1. 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 a given interface.

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.

The host SHOULD treat the PREF64 as being specific to the network interface it was received on. Provisioning Domain (PvD, [RFC7556]) aware hosts MUST treat the PREF64 as being scoped to the implicit or explicit PvD.

5.2. PREF64 Consistency

Section 6.2.7 of [RFC4861] recommends that routers inspect RAs sent by other routers to ensure that all routers onlink advertise 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:

- o set of PREF64 with non-zero lifetime;

o set of PREF64 with zero lifetime.

Provisioning Domain (PvD, [RFC7556]) aware routers MUST only compare information scoped to the same implicit or explicit PvD.

6. 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]

7. 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 reduces the attack surface to be targeted by malicious attackers to provide hosts with invalid configuration as compared to distributing the configuration through multiple different mechanisms that need to be secured independently.

If a host is provided with an incorrect NAT64 prefix the IPv6-only host might not be able to communicate with IPv4-only destinations. Connectivity to destinations reachable over IPv6 would not be impacted just by providing a host with an incorrect prefix (however if attackers are capable of sending rogue RAs they can perform denial-of-service or man-in-the-middle attacks, as described in [RFC6104]).

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].

8. 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, Robert Hinden, Martin Hunek, Tatuya Jinmei, Benjamin Kaduk, Erik Kline, Suresh Krishnan, Warren Kumari, David Lamparter, Barry Leiba, Jordi Palet Martinez, Tommy Pauly, Alexandre Petrescu, Michael Richardson, David Schinazi, Ole Troan, Eric Vynke, Bernie Volz.

9. References

9.1. Normative References

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

[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, <https://www.rfc-editor.org/info/rfc4861>.

[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, <https://www.rfc-editor.org/info/rfc6052>.

[RFC7050] Savolainen, T., Korhonen, J., and D. Wing, "Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis", RFC 7050, DOI 10.17487/RFC7050, November 2013, <https://www.rfc-editor.org/info/rfc7050>.

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

9.2. Informative References

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <https://www.rfc-editor.org/info/rfc1918>.

[RFC6104] Chown, T. and S. Venaas, "Rogue IPv6 Router Advertisement Problem Statement", RFC 6104, DOI 10.17487/RFC6104, February 2011, <https://www.rfc-editor.org/info/rfc6104>.

[RFC6105] Levy-Abegnoli, E., Van de Velde, G., Popoviciu, C., and J. Mohacsi, "IPv6 Router Advertisement Guard", RFC 6105, DOI 10.17487/RFC6105, February 2011, <https://www.rfc-editor.org/info/rfc6105>.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <https://www.rfc-editor.org/info/rfc6146>.

[RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, DOI 10.17487/RFC6147, April 2011, <https://www.rfc-editor.org/info/rfc6147>.

[RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <https://www.rfc-editor.org/info/rfc6877>.

[RFC7225] Boucadair, M., "Discovering NAT64 IPv6 Prefixes Using the Port Control Protocol (PCP)", RFC 7225, DOI 10.17487/RFC7225, May 2014, <https://www.rfc-editor.org/info/rfc7225>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <https://www.rfc-editor.org/info/rfc7556>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC8305] Schinazi, D. and T. Pauly, "Happy Eyeballs Version 2: Better Connectivity Using Concurrency", RFC 8305, DOI 10.17487/RFC8305, December 2017, <https://www.rfc-editor.org/info/rfc8305>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

9.3. URIs

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

Authors' Addresses

Lorenzo Colitti
Google
Shibuya 3-21-3
Shibuya, Tokyo 150-0002
JP

Email: lorenzo@google.com

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

Email: furry@google.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 24, 2020

C. Filsfils, Ed.
D. Dukes, Ed.
Cisco Systems, Inc.
S. Previdi
Huawei
J. Leddy
Individual
S. Matsushima
Softbank
D. Voyer
Bell Canada

October 22, 2019

IPv6 Segment Routing Header (SRH)
draft-ietf-6man-segment-routing-header-26

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

Copyright Notice

Table of Contents

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 Segment Routing Architecture [RFC8402] describes Segment Routing and its instantiation in two data planes; MPLS and IPv6.

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:

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

where:

- o Next Header: Defined in [RFC8200] Section 4.4
- o Hdr Ext Len: Defined in [RFC8200] Section 4.4
- o Routing Type: TBD, to be assigned by IANA (suggested value: 4).
- o Segments Left: Defined in [RFC8200] Section 4.4
- o Last Entry: contains the index (zero based), in the Segment List, of the last element of the Segment List.

- o 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|
+-+-+-+-+-+-+-+-+
```

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

- o 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 that are not defined in this document. The allocation and use of tag is outside the scope of this document.

- o 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.

- o 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 4.3.1.1.1). 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:

- o 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.

- o Limit the length in PadN to 5.

- o Limit the maximum number of non-Pad TLVs to be processed.

- o 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.

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-----------------------
|     Type      |    Length     | Variable length data
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-----------------------
```

Type: An 8 bit codepoint from Segment Routing Header TLVs Registry TBD IANA Reference. Unrecognized Types MUST be ignored on receipt.

Length: The length of the Variable length data in bytes.

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

Type Length Value (TLV) entries 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 codepoint 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 (bit 0) 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 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. A Pad1 TLV MUST NOT be used if more than one consecutive byte of padding is required.

2.1.1.2. PADN

Alignment requirement: none

```
 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     |      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 semantic. 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    |D|        RESERVED             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      HMAC Key ID (4 octets)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               //
|                      HMAC (Variable)                          //
|                                                               //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

- o Type: to be assigned by IANA (suggested value 5).
- o Length: The length of the variable length data in bytes.
- o D: 1 bit. 1 indicates the Destination Address verification is disabled due to use of reduced segment list, Section 4.1.1.
- o RESERVED: 15 bits. MUST be 0 on transmission.
- o HMAC Key ID: A 4 octet opaque number which uniquely identifies the pre-shared key and algorithm used to generate the HMAC.
- o HMAC: Keyed HMAC, in multiples of 8 octets, at most 32 octets.

The HMAC TLV is used to verify that the SRH applied to a packet was selected by an authorized party, and to ensure that the segment list is not modified after generation. This also allows for verification that the current segment (by virtue of being in the authorized segment list) is authorized for use. The SR Domain ensures the source node is permitted to use the source address in the packet via ingress filtering mechanisms as defined in BCP 84 [RFC3704], or other strategies as appropriate.

2.1.2.1. HMAC Generation and Verification

Local configuration determines when to check for an HMAC. 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 supports 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. The check is successful when either

- o HMAC D bit is 1 and Segments Left is greater than Last Entry.
- o HMAC Segments Left is less than or equal to Last Entry and destination address is equal to Segment List [Segments Left].

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

- o key: the pre-shared key identified by HMAC Key ID
- o HMAC algorithm: identified by the HMAC Key ID
- o 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 16 bits following Length
  - * 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. Subsequent octets MUST be set to zero such that the HMAC length is a multiple of 8 octets.

If HMAC verification is successful, processing proceeds as normal.

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 and the packet discarded.

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.

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

At the HMAC TLV generating node, the Text for the HMAC computation is set to the IPv6 header fields and SRH fields as they would appear at the verification node(s), 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.

The HMAC TLV generating node may need to revoke an SRH for which it previously generated an HMAC. Revocation is achieved by allocating a new key and key ID, then rolling over the key ID associated with the SRH to be revoked. The HMAC TLV verifying node drops packets with the revoked SRH.

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:

- o in the configuration of the HMAC generating or verifying nodes, either by static configuration or any SDN oriented approach

- o dynamically using a trusted key distribution protocol such as [RFC6407]

While key management is outside the scope of this document, the recommendations of BCP 107 [RFC4107] should be considered when choosing the key management system.

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 the SRH flag or to add 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.

TLVs (including HMAC) may be set according to their specification.

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 distributed by means outside the scope of this document. For example, [RFC5308] or [RFC5340] may be used to advertise a prefix covering the SIDs on a node.

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,
         whose type is identified by the Next Header field in
         the Routing header.
S04.   }
S05.   Else {
S06.     If local configuration requires TLV processing {
S07.       Perform TLV processing (see TLV Processing)
S08.     }
S09.     max_last_entry  =  ( Hdr Ext Len /  2 ) - 1
S10.     If  ((Last Entry > max_last_entry) or
S11.          (Segments Left is greater than (Last Entry+1)) {
S12.       Send an ICMP Parameter Problem, Code 0, message to
           the Source Address, pointing to the Segments Left
           field, and discard the packet.
S13.     }
S14.     Else {
S15.       Decrement Segments Left by 1.
S16.       Copy Segment List[Segments Left] from the SRH to the
           destination address of the IPv6 header.
S17.       If the IPv6 Hop Limit is less than or equal to 1 {
S18.         Send an ICMP Time Exceeded -- Hop Limit Exceeded in
             Transit message to the Source Address and discard
             the packet.
S19.       }
S20.       Else {
S21.         Decrement the Hop Limit by 1
S22.         Resubmit the packet to the IPv6 module for transmission
             to the new destination.
S23.       }
S24.     }
S25.   }
S26. }
```

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:

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

Encapsulation with an outer IPv6 header and SRH share the same MTU and fragmentation considerations as IPv6 tunnels described in [RFC2473]. Further investigation on the limitation of various tunneling methods (including IPv6 tunnels) are discussed in [I-D.ietf-intarea-tunnels] and SHOULD be considered by operators when considering MTU within the SR Domain.

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 ultimate destination address of the IPv6 header may be required. The following logic is used to determine the destination address for use by protocol error handlers.

- o Walk all extension headers of the invoking IPv6 packet to the routing extension header preceding the upper layer header.
  - * If routing header is type TBD IANA (SRH)
    - + The SID at Segment List[0] may be used 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:

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

- o SRH with SID list <S1, S2, S3> with SegmentsLeft = SL.

- o 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

- o 3 and 4 are SR Domain edge routers
- o 5, 6, and 7 are all SR Domain routers
- o 8 and 9 are hosts within the SR Domain
- o 1 and 2 are hosts outside the SR Domain

- o The SR domain implements ingress filtering 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) without SRH. 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. 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 apply 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 ID and key associated with the HMAC TLV 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 including 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 [RFC2827] SHOULD be used.

7.1. Source Routing Attacks

An SR domain implements distributed and per node protection as described in section 5.1. Additionally, domains deny traffic with spoofed addresses by implementing the recommendations in BCP 84 [RFC3704].

Full implementation of the recommended protection blocks the attacks documented in [RFC5095] from outside the SR domain, including bypassing filtering devices, reaching otherwise unreachable Internet systems, network topology discovery, bandwidth exhaustion, and defeating anycast.

Failure to implement distributed and per node protection allows attackers to bypass filtering devices and exposes the SR Domain to these attacks.

Compromised nodes within the SR Domain may mount the attacks listed above along with other known attacks on IP networks (e.g. DOS/DDOS, topology discovery, man-in-the-middle, traffic interception/ siphoning).

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 as having been generated by a trusted source 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:

Suggested Value	Description	Reference
4	Segment Routing Header (SRH)	This document

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:

CODE	NAME/DESCRIPTION
TBD IANA	SR Upper-layer Header Error

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the SRH, in accordance with BCP 26, [RFC8126].

The following terms are used here with the meanings defined in BCP 26: "namespace", "assigned value", "registration".

The following policies are used here with the meanings defined in BCP 26 [RFC8126]: "IETF Review".

8.1. Segment Routing Header Flags Registry

This document requests the creation of a new IANA managed registry to identify SRH Flags Bits. The registration procedure is "IETF Review". Suggested registry name is "Segment Routing Header Flags". Flags is 8 bits.

8.2. Segment Routing Header TLVs Registry

This document requests the creation of a new IANA managed registry to identify SRH TLVs. The registration procedure is "IETF Review". 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:

Assigned Value	Description	Reference
0	Pad1 TLV	This document
1	Reserved	This document
2	Reserved	This document
3	Reserved	This document
4	PadN TLV	This document
5	HMAC TLV	This document
6	Reserved	This document
124-126	Experimentation and Test	This document
127	Reserved	This document
252-254	Experimentation and Test	This document
255	Reserved	This document

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

Kamran Raza, Zafar Ali, Brian Field, Daniel Bernier, Ida Leung, Jen Linkova, Ebben Aries, Tomoya Kosugi, Eric Vyncke, David Lebrun, Dirk Steinberg, Robert Raszuk, Dave Barach, John Brzozowski, Pierre Francois, Nagendra Kumar, Mark Townsley, Christian Martin, Roberta Maglione, James Connolly, Aloys Augustin contributed to the content of this document.

11. Acknowledgements

The authors would like to thank Ole Troan, Bob Hinden, Ron Bonica, Fred Baker, Brian Carpenter, Alexandru Petrescu, Punit Kumar Jaiswal, David Lebrun, Benjamin Kaduk, Frank Xialiang, Mirja Kuhlewind, Roman Danyliw, Joe Touch, and Magnus Westerlund for their comments to this document.

12. References

12.1. Normative References

[FIPS180-4] National Institute of Standards and Technology, "FIPS 180-4 Secure Hash Standard (SHS)", March 2012, <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

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

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

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <https://www.rfc-editor.org/info/rfc2827>.

[RFC3704] Baker, F. and P. Savola, "Ingress Filtering for Multihomed Networks", BCP 84, RFC 3704, DOI 10.17487/RFC3704, March 2004, <https://www.rfc-editor.org/info/rfc3704>.

[RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, DOI 10.17487/RFC4107, June 2005, <https://www.rfc-editor.org/info/rfc4107>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <https://www.rfc-editor.org/info/rfc4302>.

[RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <https://www.rfc-editor.org/info/rfc5095>.

[RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, DOI 10.17487/RFC6407, October 2011, <https://www.rfc-editor.org/info/rfc6407>.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <https://www.rfc-editor.org/info/rfc6437>.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <https://www.rfc-editor.org/info/rfc6438>.

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

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

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

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.

[I-D.ietf-intarea-tunnels]
Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", draft-ietf-intarea-tunnels-10 (work in progress), September 2019.

[I-D.matsushima-spring-srv6-deployment-status]
Matsushima, S., Filsfils, C., Ali, Z., and Z. Li, "SRv6 Implementation and Deployment Status", draft-matsushima-spring-srv6-deployment-status-02 (work in progress), October 2019.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <https://www.rfc-editor.org/info/rfc5308>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[SRN] Lebrun, D., Jadin, M., Clad, F., Filsfils, C., and O. Bonaventure, "Software Resolved Networks: Rethinking Enterprise Networks with IPv6 Segment Routing", 2018, <https://inl.info.ucl.ac.be/system/files/sosr18-final15-embedfonts.pdf>.

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

6man
Internet-Draft
Intended status: Informational
Expires: 9 November 2019

T. Jones
G. Fairhurst
University of Aberdeen

8 May 2019

Change Status of RFC 2675 to Historic
draft-jones-6man-historic-rfc2675-00

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.

Table of Contents

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 refered 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 Guidlines [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] (histroric), IPv6 Avian Carriers [RFC6214], IPv6 node requirements [RFC6434], 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

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

[RFC2675] Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, DOI 10.17487/RFC2675, August 1999, <https://www.rfc-editor.org/info/rfc2675>.

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

7.2. Informative References

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, DOI 10.17487/RFC3686, January 2004, <https://www.rfc-editor.org/info/rfc3686>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <https://www.rfc-editor.org/info/rfc3711>.

[RFC3790] Mickles, C., Ed., Nesser, P., and , "Survey of IPv4 Addresses in Currently Deployed IETF Internet Area Standards Track and Experimental Documents", RFC 3790, DOI 10.17487/RFC3790, June 2004, <https://www.rfc-editor.org/info/rfc3790>.

[RFC4294] Loughney, J., Ed., "IPv6 Node Requirements", RFC 4294, DOI 10.17487/RFC4294, April 2006, <https://www.rfc-editor.org/info/rfc4294>.

[RFC4309] Housley, R., "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)", RFC 4309, DOI 10.17487/RFC4309, December 2005, <https://www.rfc-editor.org/info/rfc4309>.

[RFC4614] Duke, M., Braden, R., Eddy, W., and E. Blanton, "A Roadmap for Transmission Control Protocol (TCP) Specification Documents", RFC 4614, DOI 10.17487/RFC4614, September 2006, <https://www.rfc-editor.org/info/rfc4614>.

[RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, DOI 10.17487/RFC5102, January 2008, <https://www.rfc-editor.org/info/rfc5102>.

[RFC5225] Pelletier, G. and K. Sandlund, "RObust Header Compression Version 2 (ROHCv2): Profiles for RTP, UDP, IP, ESP and UDP-Lite", RFC 5225, DOI 10.17487/RFC5225, April 2008, <https://www.rfc-editor.org/info/rfc5225>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", RFC 5405, DOI 10.17487/RFC5405, November 2008, <https://www.rfc-editor.org/info/rfc5405>.

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities

and Consequences", RFC 5506, DOI 10.17487/RFC5506, April 2009, <https://www.rfc-editor.org/info/rfc5506>.

[RFC6214] Carpenter, B. and R. Hinden, "Adaptation of RFC 1149 for IPv6", RFC 6214, DOI 10.17487/RFC6214, April 2011, <https://www.rfc-editor.org/info/rfc6214>.

[RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node Requirements", RFC 6434, DOI 10.17487/RFC6434, December 2011, <https://www.rfc-editor.org/info/rfc6434>.

[RFC6691] Borman, D., "TCP Options and Maximum Segment Size (MSS)", RFC 6691, DOI 10.17487/RFC6691, July 2012, <https://www.rfc-editor.org/info/rfc6691>.

[RFC6904] Lennox, J., "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, DOI 10.17487/RFC6904, April 2013, <https://www.rfc-editor.org/info/rfc6904>.

[RFC7012] Claise, B., Ed. and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, DOI 10.17487/RFC7012, September 2013, <https://www.rfc-editor.org/info/rfc7012>.

[RFC7122] Kruse, H., Jero, S., and S. Ostermann, "Datagram Convergence Layers for the Delay- and Disruption-Tolerant Networking (DTN) Bundle Protocol and Licklider Transmission Protocol (LTP)", RFC 7122, DOI 10.17487/RFC7122, March 2014, <https://www.rfc-editor.org/info/rfc7122>.

[RFC7323] Borman, D., Braden, B., Jacobson, V., and R. Scheffenegger, Ed., "TCP Extensions for High Performance", RFC 7323, DOI 10.17487/RFC7323, September 2014, <https://www.rfc-editor.org/info/rfc7323>.

[RFC7414] Duke, M., Braden, R., Eddy, W., Blanton, E., and A. Zimmermann, "A Roadmap for Transmission Control Protocol (TCP) Specification Documents", RFC 7414, DOI 10.17487/RFC7414, February 2015, <https://www.rfc-editor.org/info/rfc7414>.

[RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <https://www.rfc-editor.org/info/rfc8085>.

[RFC8468] Morton, A., Fabini, J., Elkins, N., Ackermann, M., and V. Hegde, "IPv4, IPv6, and IPv4-IPv6 Coexistence: Updates for the IP Performance Metrics (IPPM) Framework", RFC 8468, DOI 10.17487/RFC8468, November 2018, <https://www.rfc-editor.org/info/rfc8468>.

[RFC8504] Chown, T., Loughney, J., and T. Winters, "IPv6 Node Requirements", BCP 220, RFC 8504, DOI 10.17487/RFC8504, January 2019, <https://www.rfc-editor.org/info/rfc8504>.

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.

Document	Status	Title	Summary
RFC3686	PS	Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)	Considerations to cover large packets
RFC3711	PS (Updated by [RFC5506], [RFC6904])	The Secure Real-time Transport Protocol (SRTP)	(except for ipv6 "jumbograms" [RFC2675], which are not likely to be used for RTP-based multimedia traffic).
RFC3790	Informational	Survey of IPv4 Addresses in Currently Deployed IETF Internet Area Standards Track and Experimental Documents	"This document defines a IPv6 packet format and is therefore not discussed in this document."
RFC4294	Informational (Obsoleted by	IPv6 Node Requirements	"ipv6 jumbograms "[RFC-2675]" MAY

	[RFC6434])		be supported."
RFC4309	PS	Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)	Size parameters are set so they will cover Jumbograms
RFC4614	Informational (Obsoleted by [RFC7414])	A Roadmap for Transmission Control Protocol (TCP) Specification Documents	Mentions that jumbograms exist
RFC5102	PS (Obsoleted by [RFC7012])	Informationalormation Model for IP Flow Informationalormation Experimentalort	Adds Jumbogram size considerations to length fields
RFC5225	PS	RObust Header Compression Version 2 (ROHCv2): Profiles for RTP, UDP, IP, ESP and UDP-Lite	Does not support Jumbograms
RFC5405	BCP (Obsoleted by [RFC8085])	Unicast UDP Usage Guidelines for Application Designers	Jumbograms exist
RFC6214	Informational	Adaptation of RFC 1149 for IPv6	
RFC6434	Informational (Obsoleted by [RFC8504])	IPv6 Node Requirements	"to date, few implementations exist, and there is essentially no reported experience from usage."
RFC6691	Informational	TCP Options and Maximum Segment Size (MSS)	Treat 65,353 value in MSS and Urgent Pointer fields as infinite
RFC7122	Experimental	Datagram Convergence Layers for the Delay- and Disruption-Tolerant	Jumbograms exist though rarely used

		Networking (DTN) Bundle Protocol and Licklider Transmission Protocol (LTP)	
RFC7323	PS	TCP Extensions for High performance	Jumbograms weaken the TCP checksum
RFC7414	Informational	A Roadmap for Transmission Control Protocol (TCP) Specification Documents	Jumbograms exist
RFC8085	BCP	UDP Usage Guidelines	Jumbograms exist
RFC8468	Informational	IPv4, IPv6, and IPv4-IPv6 Coexistence: (Updated by)dates for the IP Performance Metrics (IPPM) Framework	Length considerations
RFC8504	BCP	IPv6 Node Requirements	"removed Jumbograms (RFC 2675) as they aren’t deployed."

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

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 13, 2020

Z. Li
S. Peng
Huawei Technologies
K. LEE
LG U+

September 10, 2019

IPv6 Encapsulation for SFC and IFIT
draft-li-6man-ipv6-sfc-ifit-02

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

Copyright Notice

Table of Contents

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

IOAM: In-situ OAM

PBT: Postcard-Based Telemetry

IFA: Inband Flow Analyzer

SRH: Segment Routing Header

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 subsequent 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 (normally 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 whichs 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 2 3 4 5 6 7 8 9 0 1
                                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                |  Option Type  |  Opt Data Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver|O|U|    TTL    |   Length  |U|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

```
 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
                                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                |  Option Type  |  Opt Data Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Namespace-ID           |NodeLen  | Flags | RemainingLen|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               IOAM-Trace-Type                 |  Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2. IPv6 Options with IOAM instructions

Option Type: TBD_1

Opt Data Len: 8 octets.

Other fields: refer to [I-D.ietf-ippm-ioam-data].

3. PBT 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 2 3 4 5 6 7 8 9 0 1
                                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                |  Option Type  |  Opt Data Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  TIH Length   |   Reserved    |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Flow ID                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Flow ID                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Sequence Number                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Data Set ID                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3. IPv6 Options with PBT instructions

Option Type: TBD_2

Opt Data Len: 20 octets.

Other fields: refer to [I-D.song-ippm-postcard-based-telemetry].

4. IFA 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 2 3 4 5 6 7 8 9 0 1
                                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                |  Option Type  |  Opt Data Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver=2.0|  GNS  |NextHdr = IP_xx|R|R|R|M|T|I|T|C|   Max Length  |
|       |       |               | | | |F|S| |A| |               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4. IPv6 Options with IFA instructions

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.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   SFC Type    |     Length    |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      SFC Metadata Class       |      Type     |U|    Length   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Variable-Length Metadata                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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.

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

Figure 6. IFA Service Metadata

IFA Type	8-bit identifier of the IFA Service Metadata type. The value is TBD-6.
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.
IFA Service Metadata Options	IFA option data is present as specified by the IFA Type field.

5. IANA Considerations

Value	Description	Reference
TBD_0	NSH Service Option	[This draft]
TBD_1	IOAM Service Option	[This draft]
TBD_2	PBT Service Option	[This draft]
TBD_3	IFA Service Option	[This draft]
TBD_4	SFC Service Metadata Type	[This draft]
TBD_5	IOAM Service Metadata Type	[This draft]
TBD_6	IFA Service Metadata Type	[This draft]

6. Security Considerations

TBD.

7. References

7.1. Normative References

[I-D.guichard-spring-nsh-sr]
Guichard, J., Song, H., Tantsura, J., Halpern, J., Henderickx, W., Boucadair, M., and S. Hassan, "NSH and Segment Routing Integration for Service Function Chaining (SFC)", draft-guichard-spring-nsh-sr-01 (work in progress), March 2019.

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

[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, H., Ghanwani, A., Cai, D., Ou, H., and L. Yizhou, "Inband 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-04 (work in progress), September 2019.

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

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

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

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

7.2. Informative References

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

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

Kihoon LEE
LG U+
71, Magokjungang 8-ro, Gangseo-gu
Seoul
Republic of Korea

Email: soho8416@lguplus.co.kr

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 12, 2019

Z. Li
S. Peng
Huawei Technologies
March 11, 2019

Service-aware IPv6 Network
draft-li-6man-service-aware-ipv6-network-00

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

Table of Contents

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

```
+-----+
       +-----+
|App x|----\
 /---->|App x|
+-----+    |    +-----------+    +--------+       +---------+       +---------+
  |    +-----+
           \--->|           |    |        |---A---|         |---A---|         |-
--/
                |Edge Device|----|Head-End|---B---|Mid-Point|---B---|End-Point|
           /--->|           |    |        |---C---|         |---C---|         |-
--\
+-----+    |    +-----------+    +--------+       +---------+       +---------+
  |    +-----+
|App y|----/
 \---->|App y|
+-----+
       +-----+
```

Figure 1 Service-aware IPv6 Network

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.filsfils-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:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                    IPv6 Service-aware ID                      +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2. IPv6 Service-aware ID Option

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:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       SLA Level   |      APP ID       |     User ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  SLA Level   |   APP ID     |  User ID    |     Arguments     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Locator Address      |   Function ID |     Arguments     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5. IPv6 Service-aware ID Structure III

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.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                                                               .
.                Service Para Sub-TLVs(Variable)                .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6. IPv6 Service-Para Option

Option Type	TBD
Opt Data Len	8-bit unsigned integer. Length of the Service Para Sub-TLVs.
Service Para Sub-TLVs	Variable-length field with Service Para Sub-TLVs.

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:

```
 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    |  Class Type   |    RESERVED   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Bandwidth                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7. BW Sub-TLV

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:

```
 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   |                    Delay                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8. Delay 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: 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:

```
 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     |               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:

```
 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   |                    Packet Loss Ratio          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 10. Packet Loss Ratio 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.

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:

Type Value

TBD Service-aware ID Option

TBD Service-Para Option

TBD BW Sub-TLV

TBD Delay Sub-TLV

TBD Delay Variation Sub-TLV

TBD Packet Loss Sub-TLV

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, <https://www.rfc-editor.org/info/rfc2119>.

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

8.2. Informative References

[RFC3272] Awduche, D., Chiu, A., Elwalid, A., Widjaja, I., and X. Xiao, "Overview and Principles of Internet Traffic Engineering", RFC 3272, DOI 10.17487/RFC3272, May 2002, <https://www.rfc-editor.org/info/rfc3272>.

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

6man Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 9, 2021

C. Li
Huawei Technologies
W. Cheng
China Mobile
Y. Zhu
China Telecom
Z. Li
D. Dhody
Huawei Technologies
May 8, 2021

Encapsulation of Path Segment in SRv6
draft-li-6man-srv6-path-segment-encap-06

Abstract

Segment Routing (SR) allows for a flexible definition of end-to-end paths by encoding an ordered list of instructions, called "segments". The SR architecture can be implemented over an IPv6 data plane, called SRv6. In some use-cases such as end-to-end SR Path Protection and Performance Measurement (PM), an SRv6 path needs to be identified. An SRv6 Path Segment can be used for identifying an SRv6 path. This document defines a P-flag in the Segment Routing Header to indicate the appearence of SRv6 Path Segment.

Status of This Memo

Copyright Notice

Table of Contents

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 an IPv6 data plane, it is called SRv6, and it uses a new IPv6 [RFC8200] Routing Header (EH) called the IPv6 Segment Routing Header (SRH) [RFC8754] to construct an SRv6 path. As per [RFC8986], an SRv6 segment identifier is a 128-bit value.

In several use cases, such as binding bidirectional path [I-D.ietf-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.

An SRv6 path MAY be identified by the content of a segment list in the SRH. However, the segment list may not be a good key, since the length of a segment list is flexible according to the number of required SIDs. Also, the length of a segment list may be too long to be a key when it contains many SIDs. For instance, if packet A uses

an SRH with 3 SIDs while Packet B uses an SRH with 10 SIDs, the key to identify these two paths will be a 384-bits value and a 1280-bits value, respectively. Furthermore, an SRv6 path cannot be identified by the information carried by the SRH in reduced mode [RFC8754] as the first SID is not present. Also, different SRv6 policies may use the same segment list for different candidate paths, so the traffic of different SRv6 policies are merged, resulting in the inability to measure the performance of the specific path. Therefore, [I-D.ietf-spring-srv6-path-segment] defines an SRv6 Path Segment to identify an SRv6 path.

This document defines a P-bit in SRH to indicate the appearence of SRv6 Path Segment in SRH.

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 [RFC8986].

2. Encoding of an SRv6 Path Segment

This section describes the SRH encoding of an SRv6 Path Segment [I-D.ietf-spring-srv6-path-segment].

2.1. SRH.P-flag

As per [I-D.ietf-spring-srv6-path-segment], an SRv6 Path Segment is a 128-bits value, and it MUST appear only once in a SID list, and it MUST appear as the last entry.

To indicate the existence of a Path Segment in the SRH, this document defines a P-flag in the SRH flag field. The encapsulation of SRv6 Path Segment is shown below.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Last Entry   |     Flags   |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

- o P-bit: set when SRv6 Path Segment is inserted. It MUST be ignored when a node does not support SRv6 Path Segment processing.

SRH.P-bit processing can be enabled or disabled by configuration on devices, it can be done by CLI, NETCONF YANG or other ways, and this is out of the scope of this document.

The pseudo code of SRH.P-bit processing is described as below.

```
S01.  if SRH.P-flag processing is enabled:
S02.     if SRH.P-flag is set:
S03.        SRv6 Path Segment processing       ;;ref1
```

Ref1: The SRv6 Path Segment processing is accosiated with the specific application, such as SRv6 Path Segment based Performance measurement, and this is out of the scope of this document.

In some use cases, only the egress need to process the SRv6 Path Segment, therefore, the P-bit processing can be done at the egress node only while the intermediate nodes do not need to process it. This feature can be enabled by configuration like CLI , NETCONF YANG or other ways. In this case, the pseudo code is described as below.

```
S01.  if SRH.P-flag processing is enabled:
S02.    if intermediate node processing is disabled:
S03.        if SRH.P-flag is set and SRH.SL == 0:
S03.            SRv6 Path Segment processing
S04     else:
S05.        if SRH.P-flag is set:
S06.            SRv6 Path Segment processing
```

3. Processing of SRv6 Path Segment

The processing of SRv6 path segment is out of the scope of this document and is defined in [I-D.ietf-spring-srv6-path-segment].

4. IANA Considerations

This document requests IANA to allocate bit position TBA within the "Segment Routing Header Flags" registry defined in [RFC8402].

5. Security Considerations

TBA

6. Acknowledgements

TBA

7. References

7.1. Normative References

[I-D.ietf-spring-srv6-path-segment]
Li, C., Cheng, W., Chen, M., Dhody, D., and R. Gandhi, "Path Segment for SRv6 (Segment Routing in IPv6)", draft-ietf-spring-srv6-path-segment-00 (work in progress), November 2020.

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

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

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

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

[RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <https://www.rfc-editor.org/info/rfc8754>.

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

7.2. Informative References

[I-D.gandhi-spring-twamp-srpm]
Gandhi, R., Filsfils, C., Voyer, D., Chen, M., and B. Janssens, "Performance Measurement Using TWAMP Light for Segment Routing Networks", draft-gandhi-spring-twamp-srpm-11 (work in progress), October 2020.

[I-D.ietf-idr-sr-policy-path-segment]
Li, C., Li, Z., Chen, H., Cheng, W., and K. Talaulikar, "SR Policy Extensions for Path Segment and Bidirectional Path", draft-ietf-idr-sr-policy-path-segment-03 (work in progress), March 2021.

[I-D.ietf-pce-sr-bidir-path]
Li, C., Chen, M., Cheng, W., Gandhi, R., and Q. Xiong, "Path Computation Element Communication Protocol (PCEP) Extensions for Associated Bidirectional Segment Routing (SR) Paths", draft-ietf-pce-sr-bidir-path-05 (work in progress), January 2021.

[I-D.ietf-pce-sr-path-segment]
Li, C., Chen, M., Cheng, W., Gandhi, R., and Q. Xiong, "Path Computation Element Communication Protocol (PCEP) Extension for Path Segment in Segment Routing (SR)", draft-ietf-pce-sr-path-segment-03 (work in progress), February 2021.

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-11 (work in progress), April 2021.

Authors' Addresses

Cheng Li
Huawei Technologies

Email: c.l@huawei.com

Weiqiang Cheng
China Mobile

Email: chengweiqiang@chinamobile.com

Yongqing Zhu
China Telecom
Guangzhou

Email: zhuyq8@chinatelecom.cn

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

6lo
Internet-Draft
Intended status: Standards Track
Expires: 12 May 2022

P. Thubert, Ed.
E.L.A. Levy-Abegnoli
Cisco Systems
8 November 2021

IPv6 Neighbor Discovery Unicast Lookup
draft-thubert-6lo-unicast-lookup-02

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 12 May 2022.

Table of Contents

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.

[RFC8929] 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 [RFC8928]) 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 [RFC8929], 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.

"IPv6 Neighbor Discovery Multicast Address Listener Registration" [I-D.ietf-6lo-multicast-registration] extends [RFC8505] to register anycast and multicast addresses. This entails accepting more than one registration for the same address and the use of the ROVR field as a complement to the index. In that case the 6LBR does not consider multiple registrations as duplicate and retains a state for the anycast addresses that it can use to reply a query.

It results that the 6LBR is capable of providing the LLA mapping for any unicast and anycast address that was proactively registered with an SLLAO. This draft defines the protocol elements and the operations to try a unicast Layer-2 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. In case of an anycast address, this draft does not provide a policy/ logic to select between multiple registrations.

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 [RFC8929] 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. 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.

2.2. 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" [RFC8929].

2.3. Glossary

This document uses the following acronyms:

6BBR	6LoWPAN Backbone Router
6BBR	6LoWPAN Border Router
6LN	6LoWPAN Node
6LR	6LoWPAN Router
6CIO	Capability Indication Option
AMC	Address Mapping Confirmation
AMR	Address Mapping Request
ARO	Address Registration Option
DAC	Duplicate Address Confirmation
DAD	Duplicate Address Detection
DAR	Duplicate Address Request
EARO	Extended Address Registration Option
EDAC	Extended Duplicate Address Confirmation
EDAR	Extended Duplicate Address Request
DODAG	Destination-Oriented Directed Acyclic Graph
LLN	Low-Power and Lossy Network
NA	Neighbor Advertisement
NCE	Neighbor Cache Entry
ND	Neighbor Discovery
NS	Neighbor Solicitation
ROVR	Registration Ownership Verifier
RA	Router Advertisement
RS	Router Solicitation

TID Transaction ID

2.4. 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.

This document uses terminology defined in [RFC8505], in particular:

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.

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) Wi-Fi [IEEE Std 802.11] and Bluetooth (Low Energy) [IEEE Std 802.15.1], or a Route-Over LLN such as the Wi-SUN mesh [I-D.heile-lpwan-wisun-overview] that leverages 6LoWPAN [RFC4919][RFC6282] and RPL [RFC6550] over [IEEE Std 802.15.4].

```
                  |
                  |
             +-----+              +-----+       +-----+
     (default) |     |          6LBR |     |       |     | IPv6
        Router |     |              |     |       |     | Node
             +-----+              +-----+       +-----+
                  |  Backbone side      |             |
      ----+-------+-----------------+---+-------------+----+-----
          |                         |                     |
       +------+                 +------+               +------+
       | 6BBR |                 | 6BBR |               | 6BBR |
       |      |                 |      |               |      |
       +------+                 +------+               +------+
          o     Wireless side   o   o  o                 o o
      o o   o  o            o o   o  o  o             o  o  o  o o
     o  o o  o o            o   o  o  o  o            o  o  o o o
     o   o  o  o               o    o  o               o  o   o
       o   o o                    o  o                     o o

       LLN                        LLN                      LLN
```

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 for unicast and anycast IPv6 addresses 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 [RFC8929].

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.

```
 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 = 1  |     Reserved    |A|D|L|B|P|E|G|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           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 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:

Code Prefix	Meaning	Reference
0	Duplicate Address Detection	Duplicate Address Detection
1	Address Mapping	This RFC
2...15	Duplicate Address Detection	

Table 2: New Code Prefixes for ICMP type 157 DAR message

Code Prefix	Meaning	Reference
0	Duplicate Address Detection	Duplicate Address Detection
1	Address Mapping	This RFC
2...15	Duplicate Address Detection	

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:

ARO Status	Meaning	Reference
11 (suggested)	Not Found	This RFC

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:

Capability Bit	Meaning	Reference
9 (suggested)	AM Support (A bit)	This RFC

Table 5: New 6LoWPAN Capability Bits

8. Acknowledgments

9. Normative References

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

[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, <https://www.rfc-editor.org/info/rfc4861>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <https://www.rfc-editor.org/info/rfc4862>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <https://www.rfc-editor.org/info/rfc6775>.

[RFC7400] Bormann, C., "6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 7400, DOI 10.17487/RFC7400, November 2014, <https://www.rfc-editor.org/info/rfc7400>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

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

[RFC8505] Thubert, P., Ed., Nordmark, E., Chakrabarti, S., and C. Perkins, "Registration Extensions for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Neighbor Discovery", RFC 8505, DOI 10.17487/RFC8505, November 2018, <https://www.rfc-editor.org/info/rfc8505>.

[I-D.ietf-6lo-multicast-registration] Thubert, P., "IPv6 Neighbor Discovery Multicast Address Listener Registration", Work in Progress, Internet-Draft, draft-ietf-6lo-multicast-registration-01, 22 October 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-6lo-multicast-registration-01>.

10. Informative References

[RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, DOI 10.17487/RFC4919, August 2007, <https://www.rfc-editor.org/info/rfc4919>.

[RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <https://www.rfc-editor.org/info/rfc6282>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC8928] Thubert, P., Ed., Sarikaya, B., Sethi, M., and R. Struik, "Address-Protected Neighbor Discovery for Low-Power and Lossy Networks", RFC 8928, DOI 10.17487/RFC8928, November 2020, <https://www.rfc-editor.org/info/rfc8928>.

[RFC8929] Thubert, P., Ed., Perkins, C.E., and E. Levy-Abegnoli, "IPv6 Backbone Router", RFC 8929, DOI 10.17487/RFC8929, November 2020, <https://www.rfc-editor.org/info/rfc8929>.

[I-D.ietf-mboned-ieee802-mcast-problems] Perkins, C. E., McBride, M., Stanley, D., Kumari, W., and J. C. Zuniga, "Multicast Considerations over IEEE 802 Wireless Media", Work in Progress, Internet-Draft, draft-ietf-mboned-ieee802-mcast-problems-15, 28 July 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-mboned-ieee802-mcast-problems-15>.

[I-D.bi-savi-wlan] Bi, J., Wu, J., Lin, T., and Y. Wang, "A SAVI Solution for WLAN", Work in Progress, Internet-Draft, draft-bi-savi-wlan-21, 10 May 2021, <https://datatracker.ietf.org/doc/html/draft-bi-savi-wlan-21>.

[I-D.thubert-6man-ipv6-over-wireless] Thubert, P., "IPv6 Neighbor Discovery on Wireless Networks", Work in Progress, Internet-Draft, draft-thubert-6man-ipv6-over-wireless-09, 17 May 2021, <https://datatracker.ietf.org/doc/html/draft-thubert-6man-ipv6-over-wireless-09>.

[I-D.heile-lpwan-wisun-overview] Heile, B., (Remy), B. L., Zhang, M., and C. E. Perkins, "Wi-SUN FAN Overview", Work in Progress, Internet-Draft, draft-heile-lpwan-wisun-overview-00, 3 July 2017, <https://datatracker.ietf.org/doc/html/draft-heile-lpwan-wisun-overview-00>.

[IEEE Std 802.15.4]
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 Std 802.11]
IEEE standard for Information Technology, "IEEE Standard 802.11 - 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.", <https://ieeexplore.ieee.org/document/9363693>.

[IEEE Std 802.15.1]
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
06254 Mougins - Sophia Antipolis
France

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

Eric Levy-Abegnoli
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
06254 MOUGINS - Sophia Antipolis
France

Phone: +33 497 23 26 20
Email: elevyabe@cisco.com

6MAN
Internet-Draft
Intended status: Informational
Expires: 9 September 2023

P. Thubert, Ed.
Cisco Systems
M. Richardson
Sandelman
8 March 2023

Architecture and Framework for IPv6 over Non-Broadcast Access
draft-thubert-6man-ipv6-over-wireless-15

Abstract

This document presents an architecture for IPv6 access networks that decouples the network-layer concepts of Links, Interface, and Subnets from the link-layer concepts of links, ports, and broadcast domains, and limits the reliance on link-layer broadcasts. This architecture is suitable for IPv6 over any network, including non-broadcast networks. A study of the issues with ND-Classic over wireless media is presented, and a framework to solve those issues within the new architecture is proposed.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

IEEE Std. 802.1 [IEEE Std. 802.1] 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 Wireless Local Area Networks (WLANs) generally do not benefit from the same reliable and cheap broadcast capabilities as legacy Ethernet yellow wires, and protocols that rely on broadcasts are less suited in those environments.

Similarly, the use of broadcast is discouraged in large Data Center (DC) fabrics and DC Interconnect (DCI) that extend the lower-layer links in large and physically distributed topologies, e.g., as meshes of point-to-point (P2P) tunnels. In such case, an overlay broadcast service is typically emulated as ingress (or reflector) replication and generates massive amounts of underlay unicast messages, possibly over expensive Wide Area Network (WAN) links.

All in all, as IPv6 [RFC8200] networks migrate from a physical wires to virtual or intangible media, the common requirement shows to decouple the abstractions that are manipulated at the network layer from physical properties such as broadcast capabilities and transitivity that are handled at the lower layers.

The original IPv6 Neighbor Discovery Protocol [RFC4861], [RFC4862] (ND-Classic) relies heavily on broadcast operation for Router Advertisement (RA), address Resolution (AR) and Duplicate address Detection (DAD). In modern networks, this may be inefficient (due to many replications over constrained links), unreliable (broadcast may be lost in transmission), counterproductive to network operations (broadcast storms), prone to impersonation and multiplication attacks (a unicast from the outside may cause a broadcast inside), and may be detrimental to privacy (an observer inside the network can discover other onlink addresses).

This document presents an architecture for IPv6 access networks that 1) decouples the network-layer concepts of Links, Interface, and Subnets from the link-layer concepts of links, ports, and broadcast domains, and 2) limits the reliance on (and impact thereof) link-layer broadcasts inside the Subnet. This architecture is suitable for IPv6 over any network, including modern Non-Broadcast MultiAccess

(NBMA) and non-transitive Point-to-Multipoint (P2MP) networks. A study of the issues with ND-Classic over wireless media is presented, and a framework to solve those issues within the new architecture is proposed.

2. Issues with ND-Classic-Based Access

2.1. ND-Classic and ND-Proxies

Though ND-Classic was the state of the art when designed for early Ethernet links at the end of the twentieth century, it is less appropriate for modern networks such as wireless and overlays that cannot provide the same cheap and reliable broadcast as a shared yellow wire. The reactive AR operation was designed to limit the amount of memory that is needed for the ND cache, at times where memory was scarce in the adapters. This trade-off, broadcast bandwidth vs. memory in the adapters, should be reevaluated for networks where router memory is aplenty but broadcast has become expensive.

The ND-Classic Neighbor Solicitation (NS) [RFC4861] message is used as a multicast IP packet for AR and DAD [RFC4862]. While the AR message is intended for one node that owns the Target address, the expectation for DAD is that there’s no node at all with that address. A message that is intended for at most one node is certainly a poor match for a broadcast operation.

The NS message for AR and DAD are sent at the network layer to a Solicited-Node multicast address (SNMA) [RFC4291] and should in theory only reach a very small group of nodes. But to support SNMA, the host must also support multicast Listener Discovery (MLD) [RFC3810], which may be an additional burden to a constrained stack, and the switches should support their own multicast routing and state, which is pushing the real problems to the lower layers.

Also, if implemented, the SNMA proceure would entail close to one state per address is every switch since there is often only one address with the same SNMA in the network - though the birthday paradox applies. This amount of memory may not have been available in early switches, and still makes little economical sense today when a complete ND cache requires one state per address in every router only, as opposed to one in every switch, when the Subnet prefix is advertised as not-onlink.

This is why, in practice, ND-Classic messages to a SNMA are mapped to link-layer broadcast on Ethernet and Wireless networks, and to full ingress replication in overlays. multicast NS transmissions may occur when a node joins the network, moves, or wakes up and reconnects to the network. Over a very large fabric, this can generate hundreds of broadcasts per second.

If the broadcasts were blindly copied over Wi-Fi links, the link-layer broadcast traffic associated to ND network-layer multicast could consume enough bandwidth to cause a substantial degradation to the unicast service [MCAST EFFICIENCY]. This is why ND Proxies are deployed and charged to reduce the resulting flood; sadly, ND-Proxies are not fully reliable for the lack of a deterministic state on all existing addresses, which leads to unpredictable failures in ND-Classic operations.

The ND-Classic Neighbor Advertisement (NA) [RFC4861] message can also be sent as a multicast to all nodes, as a gratuitous information that can be used to override the address mapping in nodes with an existing Neighbor Cache Entry (NCE) for the Target address. If this is done, all nodes in the broadcast domain are impected though there’s probably none or very few with an NCE. When it is not done, nodes with an NCE will be unable to reach the Target IP address until Neighbor Unreachability Detection (NUD) discovers the issue. Both alternatives are unsatisfactory, meaning that the whole approach should be revisited.

This problem can be alleviated by reducing the size of the broadcast domain that encompasses wireless access links. This has been done in the art of IP subnetting by partitioning the subnets and by routing between them, at the extreme by assigning a prefix, say a /64, to each wireless node (see [RFC8273]).

Another way to split the broadcast domain within a Subnet is to proxy the network-layer protocols that rely on link-layer broadcast operations at the boundary of the split broadcast domains. As an example, [IEEE Std. 802.11] recommends deploying an "ARP proxy" for the IPv4 address Resolution Protocol (ARP) and ND-Classic at the APs. The correct operation of the proxy requires the exhaustive list of the IP addresses for which proxying is provided. Forming and maintaining that knowledge is a hard problem in the general case of radio connectivity, which keeps changing with movements and variations in the environment that alter the range of transmissions. It is achieved in Wi-Fi networks through the proactive method of the wireless association, which is akin to the registration procedure in this architecture.

[SAVI] suggests discovering the addresses by snooping the ND-Classic protocol, but that can also be unreliable. An IPv6 address may not be discovered immediately due to a packet loss. It may never be discovered in the case of a "silent" node that is not currently using one of its addresses, e.g., a printer that waits in wake-on-lan state. A change of anchor, e.g. due to a movement, may be missed or misordered, leading to unreliable connectivity and an incomplete list of addresses. Bottom line: snooping ND-Classic is not appropriate to form and maintain a deterministic knowledge of the IPv6 addresses of all the neighbors that are reachable over a network port.

2.2. The case of Wireless

Like Transparent Bridging, the ND-Classic operation is reactive, and relies on IP multicast for the AR and DAD procedures. As discussed in Section 2, the network-layer multicast operation is typically implemented as a link-layer broadcast for the lack of an adapted and scalable link-layer multicast operation on most WLANs and Low-Power Personal Area Networks (LoWPANs). It results that on wireless, ND-Classic multicast messages are typically broadcasted.

As opposed to unicast transmissions, the broadcast transmissions over wireless links are not subject to automatic retries (ARQ) and therefore are not reliable. Reducing the speed at the physical (PHY) layer for broadcast transmissions can increase the reliability, at the expense of a higher relative cost of broadcast on the overall available bandwidth.

Excessive use of broadcast by protocols such as ND-Classic and Bonjour led network administrators to install multicast rate limiting to protect the network. Experimentally, this proved to have a dramatic effect on ND performance in large wireless networks. From some testing done at an IETF meeting a few years ago, it seemed that up to 90% of IPv6 link-local multicasts were dropped. The impact on user experience is usually limited since for the most part, the users will connect to addresses outside the Subnet and will not attempt to locate one another. The impact on the NOC might still be significant, since a failure related to ND operations might be transient and difficult to debunk after the fact.

Another experiment conducted during an IETF meeting consisted in manually forcing address duplicates to observe the DAD behavior. This experiment showed that DAD often (up to 80% of times was observed) fails to discover the duplication of IPv6 addresses, at least in a large wireless access networks, see [DAD ISSUES] for more. In practice, IPv6 addresses very rarely conflict, not because the address duplications are detected and resolved by the DAD operation, but thanks to the entropy of the typically 64-bit Interface IDs

(IIDs) that makes a collision quasi-impossible for randomized IIDs. This is why, even when DAD fails, the user experience is rarely affected.

Excessive use of broadcast also places a toll on the battery of wireless devices such as IoT sensors and smartphones. On paper, a Wi-Fi station must keep its radio turned on to listen to the periodic series of broadcast frames. Most of those broadcasts are dropped at the network layer when the receiving node finds it is not interested, as is the case of NS messages when the node is not the Target. In order to protect the battery lifetime, a typical smartphone will listen at a multiple of the broadcast period, blindly ignoring a large ratio of the broadcasts, and making ND-Classic operations even less reliable.

Net-net: broadcast transmissions are not reliable on wireless. Protocols designed for bridged networks that rely on broadcast transmissions often exhibit disappointing behaviors when employed unmodified on a local wireless medium (more in [MCAST PROBLEMS]). Even though there is at most one intended Target for a broadcast AR or DAD message, the broadcast impacts many wireless nodes over the whole Subnet (e.g., the ESS fabric), and yet the chances that intended Target receives the packet are limited. The fact that the user experience for classic-ND is not so dramatically affected only shows that those broadcasts are, for a large part, a useless waste of expensive resources.

Wi-Fi Access Points [IEEE Std. 802.11] (APs) deployed in an Extended Service Set (ESS) act as [IEEE Std. 802.1] bridges between the wireless stations (STA) and the wired backbone. As opposed to the classical Transparent (aka Learning) Bridge operation that installs the forwarding state reactively to traffic, the bridging state in the AP is established proactively, at the time of association. The association process registers the link-layer (MAC) address of the STA to the AP proactively, i.e., before it is needed. Based on that information, the AP maintains the exhaustive list of the associated MAC addresses and blocks the link-layer lookups for destination MAC addresses that are not associated to this AP. The association procedure protects the wireless medium against broadcast-intensive Transparent Bridging lookups, but the network-layer problem remains for the lack of a similar procedure at the network layer.

2.3. The case of Overlays

link-layer Overlays (VLANs) reduce the broadcast domain from the physical one, whereas network-layer overlays (e.g., VxLAN) can extend the Subnet beyond the limits of the physical network, enabling to deploy a Subnet over large physical domains. network-layer overlays are a clear indication of the need to decouple the Subnet from the limits of the physical network.

A network-layer overlay is typically a partial or full mesh of point to point tunnels between routers. In case of a full mesh, BUM (Broadcast, Unknown, and multicast) forwarding across the overlay can be implemented as a replication at the ingress router or at a dedicated reflector, but either way entails a growing congestion and latency as the overlay grows in size. BUM forwarding has become so detrimental to the network operations that some operators decide to turn it off. This means that a silent node, whose location is unknown to the fabric control plane and possibly forgotten, cannot be rediscovered by AR procedures, and will not be reachable again until it volunteers its own sign of life. To avoid this, modern protocols should be designed such that the use of overlay-wide broadcasts are limited to operations where a real distribution operation is desired, e.g., every node is interested in receiving the packet.

While a multicast ND-Classic RA message may be of interest within a site or a subsite to all local nodes, it is probably of little interest to other sites that are served by other routers, even when the overlay spans across both sites. The same goes for a local printer, the broadcast mDNS lookup should reach local printers but not faraway ones. This is another indication of the need to decouple the span of the Subnet from the lower layer broadcast domain, and dedicate the broadcast service to local operations such as discovery.

As discussed in Section 2.2, multicast ND-Classic messages are in fact broadcasted across the overlay, meaning that they contribute to the BUM traffic and are treated as full broadcast. Yet, those messages are used for AR and DAD and intend to reach at most one node, the owner of the Target address, if any. Using a BUM transmission that reaches every nodes in the overlay to communicate with at most one is a misuse of the overlay resources and should be replaced by unicast-based methods.

In data centers, overlays are typically combined with server multihoming at the edge. An advanced Network Interface Card (NIC) is equipped with more than one Ethernet ports for redundancy, which connect to different leaf switches (aka ToR) to the same cloud network. The server (say, using Kubernetes) needs a single address and would rather use that address on both ports to the same Subnet,

and use either network port for its own reasons, e.g., load balancing, independently of IP addressing considerations. This means that the IP Interface abstraction that the server needs is a logical construct, decoupled from the network ports, and capable to encompass more than one.

2.4. Power and Sustainability

In the wireless case, broadcasts are sent at the slowest speed available, which can be a hundred time slower than unicast transmissions, in order to maximize the chances that all nodes in the BSS will receive the frame. For that extra long duration, the broadcast transmission holds the spectrum locally, which adds to the unicast latency, and generates interferences remotely, which may cascade in losses and retries in adjacent networks. In the process, the broadcast transmission consumes up to a hundred time more power than unicast transmission of equivalent payload size. Power is also wasted when replicating a multicast packet to span an overlay, each time the packet is ultimately dropped by the recipient.

Constrained IoT devices conserve their power by placing themselves in deep sleep for most of the time. While a device is sleeping, it cannot answer ND-Classic messages for AR and DAD. This makes ND-Classic unsuitable to IoT devices. The 6LoWPAN WG has determined that the most appropriate model for a constrained network is a pull model where the device wakes up, negotiates with its router(s) for addresses and connectivity for some amount of time in the future, and goes back to sleep. The router can then perform a role of sleep proxy that defends the address(es) and holds traffic for the device till the device wakes up and pulls it from the router. Additionally, when the constrained network grows into a mesh, broadcast operations become rapidly inacceptable in terms of power and bandwidth, and the not-onlink model whereby for the most part hosts do not lookup one another, is mandatory.

In all cases, to make the internet greener, we must reconsider the use of broadcast over large access networks. To maintain the capability to build the Subnets we want, the IPv6 architecture must enable to decouple the Subnet from the broadcast domain, make the broadcast domain small and local, and refocus the use of broadcast to the cases where all nodes are interested. Additionally, the IPv6 architecture must enable a model where the network serves and protects low-power devices that sleep most of the time and wake up on their own schedule.

2.5. Security and Privacy

Broadcasting ND_Classic messages expose the source and the Target addresses to the whole network, including nodes that do not need to know that those addresses are present in the network. A passive listener may discover the addresses without any observable action or possibility of control by the source. Once it has discovered the presence of neighbor addresses, the onlink attacker can impersonate any host in the network, either by sourcing packets with a stolen address, or by overriding the neighbor caches with a NA that indicates the attacker's link-layer address.

It is thus desirable to avoid exposing the host IPv6 address in broadcast ND messages. A more private approach has each node see and connect to only a subset of the routers using the not-onlink model. The source may still shortcut to the destination when the destination is effectively on link, based on redirect messages from the router, when desirable. Additionally, it is desirable that only the address owner can source packets with that address, and that another party may not be able to claim and use that address.

The reactive NS lookup method can also be leveraged from the outside of the network to perform DOS attacks on constrained resources in the network. The attacker just needs to forge any random address from the Subnet prefix and send one packet to that random address. The Subnet ingress router will have to store that packet for the duration of the lookup time, and broadcast an NS to lookup the forged address. This both locks memory in the router and consumes bandwidth and energy, impacting all nodes in the Subnet.

To avoid the lookup delays and associated attacks, it makes sense to use a proactive method whereby the router knows all the address mappings in advance. When that is achieved, if the destination address of an incoming packet is not present in the router tables, then the destination does not exist in the network and the packet can safely be dropped by the forwarding engine, e.g., in hardware.

2.6. More Middleboxes

The above problems have been observed at least since the early 2000s. A number of actions were taken. For instance, IEEE std 802.11 [IEEE Std. 802.11] mandates the support of a middlebox operation called "ARP proxy" for IPv4 and IPv6 in the Access Point (AP). The "ARP Proxy" cancels broadcasts over a BSS when the IP Target of the ARP/ND message is not owned by a STA associated to the AP. With IPv4, the expectation is that the STA owns exactly one IP address, and that the address is obtained via DHCP right after the association, so it is simple and deterministic to snoop the address in the DHCP exchange

(as long as it remains in the clear) and cancel undesirable ARP lookups.

In contrast to IPv4, IPv6 enables a node to form multiple addresses, some of them temporary 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 ND-Classic 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 state persistence in the AP when a STA ceases to use an IPv6 address. It follows that snooping protocols is not a recommended technique and that it should only be used as last resort.

Because ND-Classic is so easy to attack, some vendors have deployed undocumented proprietary counter measures as middlebox operation in the switches and routers. Those middleboxes snoop the ND-Classic messages, filter them or modify them, for instance to change their link-layer scope from broadcast to unicast. Based on the snooped information, the middlebox may for instance drop an RA message that appears to be coming from a host (e.g., as inferred because it is received on a wireless adapter), or an NA message coming from a device that does not appear to be the owner. But, for the lack of an explicit contract between the host and the middlebox, the middlebox cannot determine who the real owner is, and it may deny a rightful user.

In a managed network, IP addresses are an expensive and thus a limited resource. To ensure a fair use and protect against DOS attacks, the middlebox may also block Stateless address Autoconfiguration (SLAAC) from a host above a fixed number of addresses. When that happens, the host believes that it can use the address but fails to connect with it. This might happen even if the host has ceased to use other addresses and is now within the allowed quota. Classic-ND lacks both a method for the host to know how many addresses it can own, and a method for the router to know which addresses the host uses at any point of time. The infrastructure needs a deterministic knowledge of the addresses in use, and for that a contract must be passed between the host and the network to ensure that the all the addresses are known and usable.

Maybe the most insidious side effect of those middleboxes is that as opposed to NAT, their operation is obfuscated and proprietary. From one vendor to the next, and from one product generation to the next, their behavior may evolve and affect ND-Classic in different

fashions. In the short term, this may only impact some specific deployments, which may have to work around the issues. In the longer term, this may affect the capability we have to evolve the protocol, like firewalls impact our capability to develop new transports in parallel to TCP and UDP. We must either standardize (e.g., for ND proxy) or eliminate those middlebox activities, and for that, the IPv6 ND protocol must evolve to a model where proprietary middleboxes are not needed anymore. This demands a model that minimizes the use of broadcasts, and where a contract provides mutual guarantees for the host that need IPv6 addresses and the routers that provide reachability and protection for these addresses.

2.7. Summary of Issues

ND-Classic inherited 2 majors design points from IPv4, a strong coupling of logical and physical concepts, which creates unacceptable constraints on modern deployments with virtual and intangible links, and a reactive operation for AR and DAD that requires an extensive use of broadcast spanning the Subnet. While IPv4 and IPv6 behaviors are similar for addresses obtained via DHCP, the cost of AR and DAD makes IPv6 significantly more expensive than IPv4 when SLAAC is enabled.

And while IPv4 supported NBMA and P2MP models (e.g., on Frame Relay leveraging OSPFv2), the IPv6 promise to support NBMA (for ATM) remains unmet to this day, as only P2P and Transit links are properly supported by ND-Classic. For those reasons, as well as inherent complexity and unpredictability, IPv6 with SLAAC can be significantly less attractive than IPv4 to some network administrators.

ND-Classic exposes all addresses to all nodes in the network, which is unfit for privacy. It is prone to DOS attacks from outside the network and impersonation attacks from the inside, with no method to prove the address ownership and perform Source address Validation (SAVI) later on the data traffic. To protect against such threats, the vendors had to introduce middleboxes that interfere with the protocol operation and affect the capability to evolve the protocol in the future.

ND-Classic lacks a support for mobility (which typically entails a sequence counter maintained by the host and the deprecation of state that is based on older sequences) and for anycast. This makes it very hard for the network to defend the addresses on behalf of the owner, e.g., when the owner is temporarily disconnected. It results that the operation of the middleboxes is unsatisfactory and may cause discontinuities in connectivity.

The extensive use of broadcast operations in ND-Classic is not only detrimental to bandwidth, it is also an issue for energy conservation and sustainability. Devices must be always attached and always powered on to answer NS messages, which makes ND-Classic inapplicable to power-conserving devices such as IoT sensors that sleep for the vast majority of their time.

3. An Architecture for IPv6 over Non-Broadcast Networks

3.1. Basic Concepts

This document introduces an alternate architecture for IPv6 access networks that is designed to apply to the WLANs and LoWPANs types of networks as well as other NBMA networks such as Data-Center overlays and P2MP networks such as IoT radio meshes. It may be used as a replacement to the ND-Classic reactive model in any network where the issues discussed in Section 2 are detrimental to the network operation.

The key design points in this architecture derive from the original observations made at the 6LoWPAN WG for constrained devices and networks, and focus on avoiding waste of limited resources such as spectrum and energy, by using broadcasts only when broadcast is really needed, and decoupling the IP abstraction of a Subnet from the broadcast domains to avoid Subnet-wide broadcast storms. To that effect, this architecture leverages the not-onlink model and routing inside the Subnet, which enables to form potentially large MLSNs without creating a large broadcast domain at the link layer.

To support the deployment agility that virtual (e.g., VxLAN overlays and pseudowires) and intangible (e.g., wireless, laser, and quantum) links enable, the IP abstractions of Interface, Link, and Subnet are decoupled from their classical physical counterparts of port, link, and broadcast domains. The Subnet is defined by a prefix length called Subnet Prefix Length (SPL), as the longest aggregation that can be advertised in the IGP. An SPL of 64 is typical though the architecture does not mandate it. Host routes and prefixes longer than SPL are advertised inside the Subnet only, using a separate Subnet Gateway protocol (SGP).

Any device that owns an address within the Subnet prefix belongs to the Subnet, this is now decoupled from the physical connectivity and broadcast domain. Instead, the IPv6 routers that serve a Subnet must form a connected dominating set such that every host in the Subnet is connected to at least one router and the routers are connected to one another directly (classical NBMA, aka full mesh) or indirectly via other routers (Point to MultiPoint, P2MP, aka partial mesh). The not-onlink model is used throughout, so hosts do not look each other

up, saving all the associated broadcast. Instead, they rely on the routers to forward the packets inside and outside the Subnet. This way, the Subnet can have any structure needed for the deployment, where hosts can move from router to router in the Subnet, or anywhere in the Internet provided they can lay a mobility tunnel to one of the routers for use as IP Link to the Subnet.

All IP Links are abstracted as Point-to-Point, though a lower-layer broadcast service may be used by the router to send RAs to a subset of local hosts in the Subnet, or by the host to send an RS message to a subset of the routers. An IP Interface bundles one or more subInterfaces, one per Subnet that can be reached through that Interface. A Global IPv6 address is installed on the subInterface that connects to the Subnet from which the address derives. The IP Interface connects to one or more IP Links (to different neighbors) over the same or over different physical ports (they are decoupled). A link-local address is associated to the IP Link directly. Each SubInterface connects to the subset of those IP Links that reach other nodes in the Subnet.

In a fashion similar to a IEEE std 802.11 [IEEE Std. 802.11] Association, IPv6 nodes register their addresses to one or more neighbor router(s), which may reject the registration, e.g., in case of a duplication. With the registration, the routers collectively build a complete knowledge of the hosts they serve and in return, hosts obtain guaranteed routing services for their registered addresses for a contractual lifetime.

To support distributed routers in the Subnet, an abstract registrar service maintains the state of all active registrations in the Subnet and answers queries to lookup mappings, validate ownership, and avoid duplications. The registration is abstract to the routing service and the registrar service, and it can be protected to prevent impersonation attacks. The registration enables the network to know deterministically all the IPv6 addresses and link-layer address mapping currently in use, and eliminates the need for lookups and DAD, and for the associated broadcasts.

The abstract routing service allows an ingress router to find a path to the destination address within the Subnet. It can be a simple reflection in a Hub-and-Spoke Subnet that emulates an IEEE Std. 802.11 Infrastructure BSS at the network layer. It can also be a full-fledge routing protocol, e.g., RPL (see [RFC9010]), which is designed to adapt to various LLNs such as WLAN and WPAN radio meshes, or RIFT (see [I-D.ietf-rift-rift]) or BGP/EVPN (see [RFC8365]), for application in data centers. It can be based on overlay tunnels between ingress router and egress router leveraging a resolver service such as LISP, see [RFC7834] for more. Finally, the routing

service can also be an ND proxy that emulates an IEEE Std. 802.11 Infrastructure ESS at the network layer, as specified in the IPv6 Backbone Router [RFC8929].

The abstract registrar service maintains the mapping between the registered node link-layer address and the registered IPv6 address. It contains meta data that enables to ascertain that the second registration for the same address is performed for the same registered node, so it also binds the registered node with the registered IPv6 address. The registrar service provides APIs to look up a link-layer address for an IPv6 address as well as validate IPv6 address ownership. The registrar can be implemented as a mapping server ala LISP [RFC6830], a distributed state ala ND proxy [RFC8929], or a synchronized state ala EVPN [RFC7432]. In the former case, this enables the reactive lookups to be performed as unicast requests to the map resolver. In the latter, the address mapping is synchronized by the routing protocol and known to all the routers for all nodes in the IP Subnet, so there is never a need for a reactive lookup.

On the one hand, the Architecture proposed in this document avoids the use of broadcast operation for DAD and AR, and on the other hand, it supports use cases where Subnet and link-layer domains are not congruent, which is common in wireless networks unless a specific link-layer emulation is provided.

The address registration establishes a contract between the nodes and the routers where nodes can ask for addresses which will be guaranteed to be operational for a contractual lifetime, and the network may accept or refuse granting additional addresses based on state (e.g., duplicate address) as well as policy (e.g., quota). This ways hosts and routers agree deterministically on which addresses will be served to which nodes in the Subnet. The registration is agnostic to the router to router and router to registrar interfaces. The latter interface can be implemented in various fashions that can blend in existing technologies such as legacy ND-Classic network through ND proxy, as well as EVPN-based and LISP-based overlays.

3.2. Acronyms

This document uses the following abbreviations:

6BBR: 6LoWPAN Backbone Router
6LN: 6LoWPAN Node
6LR: 6LoWPAN Router
ARO: address Registration Option
BGP: Border Gateway Protocol

DAC: Duplicate address Confirmation (message)
DAD: Duplicate address Detection
DAR: Duplicate address Request (message)
EDAC: Extended Duplicate address Confirmation
EDAR: Extended Duplicate address Request
EVPN: Ethernet VPN
IGP: Interior Gateway Protocol
LAN: Local Area Network
LISP: Locator/ID Separation Protocol
LLN: Low-Power and Lossy Network
LLA: link-local address
LoWPAN: Low-Power WPAN
MAC: Medium Access Control
MLSN: Multi-link Subnet
MLD: multicast Listener Discovery
NA: Neighbor Advertisement (message)
NBMA: Non-Broadcast Multi-Access
NCE: Neighbor Cache Entry
ND: Neighbor Discovery (protocol)
NDP: Neighbor Discovery Protocol
NS: Neighbor Solicitation (message)
P2P: Point-to-Point
P2MP: Point-to-Multipoint
RPL: IPv6 Routing Protocol for LLNs
RA: Router Advertisement (message)
RS: Router Solicitation (message)
SGP: Subnet Gateway Protocol
SPL: Subnet Prefix Length
ULP: Upper-Layer Protocol
VLAN: Virtual LAN
VxLAN: Virtual Extensible LAN
VPN: Virtual Private Network
WAN: Wide Area Network
WiND: Wireless Neighbor Discovery (protocol)
WLAN: Wireless Local Area Network
WPAN: Wireless Personal Area Network

3.3. Terminology

3.3.1. IP Links

The term "link" refers to layer 2 (comprising MAC and link layers) communication medium that can be leveraged at layer 3 (aka IP layer, aka network layer) to instantiate one IP hop (see section 2 of [RFC8200]. In this document we conserve that term (lowercase) but differentiate it from an IP Link, which is a network-layer abstraction that represents the link but is not the link, like the map is not the country.

With IPv6, IP has moved to network-layer abstractions for its operations, e.g., with the use of a link-local address (LLA), and that of IP multicast for link-scoped operations. At the same time, the concept of an IP Link emerged as an abstraction that represents how the network layer considers the link:

* An IP Link connects an IP node to one or more other IP nodes using a lower-layer subnetwork. The lower-layer subnetwork may comprise multiple links, e.g., in the case of a switched fabric or a mesh-under LLN.

* an IP Link defines the scope of an LLA, and defines the domain in which the LLA must be unique

* An IP Link is attached to a physical port, and one link-local address is associated to the IP Link.

* an IP Link provides a subset of the connectivity that is offered by the physical link at the lower layer; if the IP Link is narrower than the link-layer reachable domain, then network-layer filters must restrict the link-scoped communication to remain between peers on a same IP Link. More than one IP Link may be installed on the same network port to connect to different peers.

* an IP Link can be Point to Point (P2P), Point to Point (P2MP, forming a partial mesh and non-transitive), NBMA (non-broadcast multi-access, fully meshed), or transit (broadcast-capable and any-to-any).

It is a network design decision to use one IP Link model or another over a given lower-layer subnetwork, e.g., to map a Frame Relay network as a P2MP IP Link, or as a collection of P2P IP Links. As another example, an Ethernet fabric may be bridged, in which case the nodes that interconnect the lower-layer links are L2 switches, and the fabric can be abstracted as a single transit IP Link; or the fabric can be routed, in which case the P2P IP Links are congruent with the link-layer links, and the nodes that interconnect the links are routers.

This architecture only uses P2P Link abstractions as shown in Figure 1, where an IP Link is identified by a pair of local and remote link-layer (MAC) address. A network port may enable to reach to more than one peer at the link layer; in that case, this architecture maps each peer relationship as a different IP Link. A link-local address only needs to be unique within that peer to peer relationship.

```
+--------------------
|
|   IP Link 1 ---------------------------> Node 1
       link-layer address MAC1             link-layer address MAC3
|      link-local address LLA1             link-local address LLAN1
|
|   IP Link 2 ---------------------------> Node 2
       link-layer address MAC2             link-layer address MAC4
|      link-local address LLA2             link-local address LLAN2
|                ...
|
|             (LLA 1 may be the same as LLA 2)
+--------------------
      network port
```

Figure 1: P2P Link Abstraction

If only the network port but not the link-layer address of the peer is visible from the network layer when processing a message, then the network layer cannot discriminate the IP Link of packets arriving on the same network port, and for that reason, it will reject a second registration for the same link-local address by a second peer, meaning that a link-local address will have to be unique on a network port across IP Links. In that case, the link-local address of the peer is used to identify the IP Link, and all the addresses registered to this node with the same peer link-local address as source will be associated to the same IP Link to that peer.

3.3.2. IP Interfaces

As is the case for links, the term interface has been historically confused between the network port that provides physical connectivity, and the network-layer abstraction that connects the host with the IP Link:

* an IP Interface is an abstraction that connects the host with a collection of IP Links (for the purpose of link-local communication) and bundles the interfaces for each IP Subnet as subInterfaces. The host installs at least one link-local address on an IP Interface for each IP Link that is connected through that Interface, and one subInterface per Subnet. The same link-local address may be reused over different IP Links as long as it is not a collision for the peer on that IP Link. Similarly, the host installs one or more global scope unicast address(es) on an IP subInterface for the associated Subnet, and the address is advertised over each IP Link in the SubInterface.

* an IP Interface can be P2P, in which case it connects to a single IP Link, or P2MP, in which case it aggregates multiple IP Links. In a multihomed host, a single IP Interface can be installed to connect to the IP Links associated to different network ports, in which case the same IPv6 address may be advertised on more than one network port. Conversely, when more than one Subnet is reachable over a network port, more than one IP Interface may leverage that network port for transmission.

```
+--------------------
|
|   IP SubInterface a  ------------------> IP Link A
|      IP Subnet a::/64                    IP Link B
|         IP Addresses a::1 .. a::n           ...
|                                          IP Link N
|
|   IP SubInterface b  ------------------> IP Link A
|      IP Subnet b::/64                    IP Link D
|         IP Addresses b::1 .. b::n           ...
|                                          IP Link P
|               ...
|
|  (Link A and B may be attached to different network ports)
|  (Link A may belong to both subInterfaces a and b)
|
+--------------------
      IP Interface, using SPL=64
```

Figure 2: Interface Abstraction

3.3.3. IP Subnets

IPv6 builds another abstraction, the IP Subnet, over one shared IP Link or over a collection IP Links, forming a MLSN in the latter case. An MLSN is formed over IP Links (e.g., P2P or P2MP) that are interconnected by routers that either inject hosts routes in an SGP, in which case the topology can be anything, or perform ND proxy operations, in which case the structure of links must be strictly hierarchical to avoid loops.

```
+--------------------
|                                                 router 4
|                                                 ^    ^
| +----------------------------------+      /       |
| |      L2 broadcast domain        IP Link     IP Link
| |                              /  |              |
| |                            v    |              v
| |   router1 <--IP Link--> router 2 <--IP Link--> router 3
| |     ^  ^                 ^  ^    |              ^
| |     |    \             /    |    |              |
| | IP Link IP Link IP Link IP Link   |           IP Link
| |     |       \      /        |    |              |
| |     v         v  v          v    |              v
| |   host 1     host 2       host 3  |            host 4
| |                                   |
| +----------------------------------+
|
|  (Different IP Links may be sustained by different media)
|
+--------------------

      IP Subnet
```

Figure 3: Subnet Abstraction

It is a network design decision to use one IP Subnet model or another over a given lower-layer network. A switched fabric can host one or more IP subnets, in which case the IP Links can reach all and beyond one Subnet. On the other hand, a Subnet can encompass a collection of links; in that case, the scope of the link-local addresses, which is the IP Link, is narrower than the span of the Subnet.

A Subnet prefix is associated with the IP Subnet, and a node is a member of an IP Subnet when it has an IP address that derives from that prefix. The IP address has Global Unicast scope (in the formal sense of [RFC4291]), and, as opposed to link-local Addresses, the scope of the address is not limited to the IP Link.

The switched and routed fabric above could be the exact same network of physical links and boxes, what changes is the way the networking abstractions are mapped onto the system, and the implication of such decision include the capability to reach another node at the link layer, and the size of the broadcast domain and related broadcast storms.

3.3.4. ND Proxies

[RFC8929] defines bridging and routing ND-Classic proxies for registering nodes / registered addresses. Both forms of ND proxies interconnect IP Links and enable to isolate the link-layer broadcast domains. But in the case of a bridging proxy, the link-layer unicast communication can still exist between the link-layer domains that are covered by network-layer links, whereas in the base of a routing proxy, they are isolated, and packets must be routed back and forth. Bridging proxies are possible between compatible technologies and translational bridges (e.g., Wi-Fi to Ethernet), whereas routing proxies are required between non-bridgeable technologies and desirable to avoid exposing the link-layer addresses across, e.g., for reasons of stability and scalability.

ND proxies can also serve IPv6 nodes that still rely on ND-Classic in a coexistence scenario. The ND proxy intercepts (snoops) the multicast NS messages from the nodes and, in case or AR or DAD, polls the registrar to lookup whether an active mapping exists for the Target. When that is the case, the ND proxy may forward the NS message as a link-layer unicast to the node that owns the binding, else it may either drop the multicast or broadcast it at the link layer. Once the node formed an address, the ND proxy fills the registrar to associate the IPv6 address with the node. The method is brittle, since there is no contract with the node to guarantee the ownership, no "contract", as discussed in Section 2.6, so for those addresses, the registrar may be inaccurate.

3.3.5. Subnet Gateway Protocols

The SPL boundary creates a wall between the traditional Interior Gateway Protocols (IGP) that operate between Subnets and manipulate shorter than SPL prefixes, and Subnet Gateway Protocols (SGP) that operate inside a Subnet and manipulate longer than SPL prefixes, typically /128 host routes, and possibly more specific data like link-layer address mappings and address Proof of Ownership.

As opposed to classical IGPs, an SGP must support rapid mobility of addresses to cope with wireless devices and virtual machines mobility. In that regard, an SGP operates mores as a MANET protocol than as a classical IGP. Ideally, there should be no stale route, and no microloop. A classical method in MANETs to achieve this is to sequence the movements and advertise the sequence in the routing protocol, so only routes with the most recent sequence can be followed, and once a packet starts following a route with a certain sequence, it must be discarded rather to have to follow a path with an older sequence. To support this approach, the node that registers an address must be the owner of a mobility sequence number and update that sequence when it moves.

Multihoming being a classical requirement in DC environments, the SGP must be able to differentiate not only address duplication from movement, but also from anycast addresses, which can be advertised from multiple places in a coordinated (same mobility sequence) or uncoordinated fashion. For unicast addresses, an token that identifies the address owner can be used for address duplication avoidance, and if that token is cryptographic, it can be used as registration ownership verifier as well.

3.4. IP Models

3.4.1. Physical Broadcast Domain

At the physical (PHY) layer, a node's broadcast domain is the set of nodes that may receive a transmission that the node sends over a network port, for instance the set of nodes in range of the radio transmission. This set can comprise a single peer on a serial cable used as point-to-point link. It may also comprise multiple peer nodes on a broadcast radio or a shared physical resource such as the Ethernet wires and hubs for which ND-Classic was initially designed.

On WLAN and LoWPAN radios, the physical broadcast domain is defined relative to a particular transmitter, as the set of nodes that can receive what this transmitter is sending. Literally every frame defines its own broadcast domain since the chances of reception of a given frame 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 physical-layer communication can be established between two nodes if the physical broadcast domains of their unicast transmissions include one another. On WLAN and LoWPAN radios, that relation is usually not reflexive, since nodes disable the reception when they transmit; still they may retain a copy of the transmitted frame, so

it can be seen as reflexive at the MAC layer. It is often symmetric, meaning that if B can receive a frame from A, then A can receive a frame 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 uni-directional, 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 that specific situation can not be assumed in the general case. In other words, the relation of radio connectivity is generally not transitive, meaning that A in range of B and B in range of C does not necessarily imply that A is in range of C.

3.4.2. Link-layer Broadcast Emulations

We call Direct MAC Broadcast (DMB) the transmission mode where the broadcast domain that is usable at the MAC layer is directly the physical broadcast domain. IEEE Std. 802.15.4 [IEEE Std. 802.15.4] and IEEE Std. 802.11 [IEEE Std. 802.11] OCB (for Out of the Context of a BSS) are examples of DMB radios. DMB networks provide mostly symmetric and non-transitive transmission. This contrasts with a number of link-layer Broadcast Emulation (LLBE) schemes that are described in this section.

In the case of Ethernet, while a physical broadcast domain is constrained to a single shared wire, the IEEE Std. 802.1 [IEEE Std. 802.1] bridging function 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 transitive 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 Map Resolver. 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 to do so.

An IEEE Std. 802.11 [IEEE Std. 802.11] Infrastructure Basic Service Set (BSS) also provides a transitive closure of nodes as defined by the broadcast domain of a central AP. The AP relays both unicast and broadcast packets and provides the symmetric and transitive emulation of a shared wire between the associated nodes, with the capability to signal link-up/link-down to the upper layer. Within a 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 increase the chances 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 the longest occupancy of the medium, for a duration that can be a hundred times that of the unicast transmission of a frame of the same size.

For that reason, upper-layer protocols (ULPs) should tend to avoid the use of broadcast when operating over IEEE std 802.11 [IEEE Std. 802.11] as they already typically do over IEEE std 802.15.4 [IEEEstd802154]. To cope with these problems, APs may implement strategies such as turn a broadcast into a series of unicast transmissions, or drop the message altogether, which may impact the upper-layer protocols. For instance, some APs may not copy Router Solicitation (RS) messages under the assumption that there is no router across the wireless network. This assumption may be correct at some point of time and may become incorrect in the future. Another strategy used in Wi-Fi APS is to proxy protocols that heavily rely on broadcast, such as the address Resolution in ARP and ND-Classic, and either respond on behalf or preferably forward the broadcast frame as a unicast to the intended Target.

In an IEEE Std. 802.11 [IEEE Std. 802.11] Infrastructure Extended Service Set (ESS), infrastructure BSSes are interconnected by a bridged network, typically running Transparent Bridging and the Spanning tree Protocol or a more advanced link-layer Routing (L2R) scheme. In the original model of learning bridges, the forwarding state 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 on the reverse path. This is a reactive operation, meaning that the state is populated reactively to the need to reach a destination. It is also possible in the original model to broadcast a gratuitous frame to advertise self throughout the bridged network, and that is also a broadcast.

The process of the Wi-Fi association prepares a bridging state proactively at the AP, which avoids the need for a reactive broadcast lookup over the wireless access. In an ESS, the AP may also generate a gratuitous broadcast sourced at the MAC address of the STA to prepare or update the state in the learning bridges so they point towards the AP for the MAC address of the STA. This framework emulates that proactive method at the network layer for the operations of AR, DAD and ND proxy.

In some instances of WLANs and LoWPANs, a Mesh-Under technology (e.g., a IEEE Std. 802.11s or IEEE Std. 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 the link layer. When operating on a single frequency, this operation is known to interfere with itself, and requires inter-frame gaps to dampen the collisions, which reduces further the amount of available bandwidth.

As the cost of broadcast transmissions becomes increasingly expensive, there is a push to rethink the upper-layer protocols to reduce the dependency on broadcast operations.

3.4.3. Mapping the IP Link Abstraction

As introduced in Section 3.3.1, IPv6 defines a concept of IP Link, link scope and link-local Addresses (LLA), an LLA being unique and usable only within the scope of an IP Link. The ND-Classic [RFC4861] DAD [RFC4862] process uses a multicast transmission to detect a duplicate address, which requires that the owner of the address is connected to the link-layer broadcast domain of the sender.

On a wired medium, the IP 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 with a link-layer broadcast domain that emulates a physical broadcast domain over the mesh of wires. But the difference shows on legacy P2MP and NBMA networks 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 link-layer 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 link-layer broadcast domain. Relying on multicast for the ND operation remains feasible but becomes highly detrimental to the unicast traffic, and becomes less and less energy-efficient and reliable as the network grows.

On DMB 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 over the broadcast domain defined by one radio transmitter if that transmitter keeps meeting new peers on the go.

The scope on which the uniqueness of an LLA must be checked is each new pair of nodes for the duration of their conversation. As long as there’s no conflict, a node may use the same LLA with multiple peers

but it has to perform DAD again with each new peer. A node may need to form a new LLA to talk to a new peer, and multiple LLAs may be present in the same radio network to talk to different peers. In this framework, each pair of nodes defines a P2P IP Link, and define the domain where an LLA must be unique.

The DAD and AR procedures in ND-Classic expect 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 applicable for P2P and transit links, but requires extensions for more complex topologies.

3.4.4. Mapping the IPv6 Subnet Abstraction

As introduced in Section 3.3.3, IPv6 also defines the concept of a IP Subnet for IPv6 unicast addresses with a global scope, Global and Unique Local Addresses (GUA and ULA). All the addresses in the same Subnet share the same prefix, and by extension, a node belongs to an IP Subnet if it has an address that derives from the prefix of the Subnet. That address must be topologically correct, meaning that it must be installed on a sub-Interface that connects to the Subnet, for use with routers that expose the Subnet in their RA messages (see [RFC5942]).

Unless intently replicated in different locations for very specific purposes, a Subnet prefix is unique within a routing system; for ULAs, the routing system is typically a limited domain, whereas for GUAs, it is the whole Internet.

For that reason, it is sufficient to validate that an address that is formed from a Subnet prefix is unique within the scope of that Subnet to guarantee that it is globally unique within the whole routing system. Note that a Subnet may become partitioned due to the loss of a wired or wireless link, so even that operation is not necessarily obvious, more in [DAD APPROACHES].

The IPv6 aggregation model 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 link-layer address and deliver the packet, or, in the case of an MLSN, route the packet to the destination within the Subnet.

If the Subnet is known as on-link, then any node may also resolve the destination link-layer address and deliver the packet, but if the Subnet is not on-link, then a host in the Subnet that does not have a Neighbor Cache Entry (NCE) for the destination will also need to pass the packet to a router, more in [RFC5942].

On Ethernet, an IP Subnet is often congruent with an IP Link because both are determined by the physical attachment to a shared wire or an IEEE Std. 802.1 bridged domain. In that case, the connectivity over the IP Link is both symmetric and transitive, the Subnet can appear as on-link, and any node can resolve a destination MAC address of any other node directly using ND-Classic.

But an IP Link and an IP Subnet are not always congruent. In the case of a Shared Link, individual subnets may each encompass only a subset of the nodes connected to the link. Conversely, in Route-Over Multi-link subnets (MLSN) [RFC4903], routers federate the links between nodes that belong to the Subnet, the Subnet is not on-link and it extends beyond any of the federated links.

3.5. Stateful address Autoconfiguration and Subnet Routing

This Architecture defines a new operation for ND that is based on 2 major paradigm changes, a proactive address registration by hosts to their attachment routers and routing to host routes (/128) within the Subnet. This allows ND to avoid the expectations of transit links and Subnet-wide broadcast domains.

The proactive address registration, called Stateful address Autoconfiguration (SFAAC) by opposition to SLAAC, is agnostic to the method used for address Assignment, e.g., Manual, Semantically Opaque Autoconfiguration [RFC7217], randomized [RFC8981], or DHCPv6 [RFC8415]. It does not change the IPv6 addressing [RFC4291] or the current practices of assigning prefixes, with typically a SPL of 64, to a Subnet. But the DAD operation is performed as a unicast exchange with the abstract egistrar service.

This Architecture combines SFAAC with the not-onlink model on the IP Interfaces. Hosts do not expect the IP Subnet to be reachable over the L2 broadcast domain and rely on their routers to forward the packets inside and outside the Subnet. In turn, the router expose to each other all the IPv6 addresses that are either owned or registered to it as host routes over a Subnet Gateway Protocol, a routing protocol that is specialized in routing inside the Subnet and can be decoupled with the IGP, that is the routing protocol used between subnets.

4. A Framework for Stateful address Autoconfiguration and Subnet Routing

4.1. Implementing Stateful address Autoconfiguration

SFAAC was initially standardized for IoT and wireless links as [RFC6775], [RFC8505], and [RFC8928]. A new option in NS/NA messages, the Extended address Registration Option (EARO) signals that the Target address is being registered and provides the registration parameters [RFC8505]. This method allows to prepare and maintain the host routes in the routers and avoids the reactive address Resolution in ND-Classic and the associated link-layer broadcast transmissions.

The EARO provides information to the router that is independent to the routing protocol and routing can take multiple forms, from an SGP to a collapsed Hub-and-Spoke model where only one router owns and advertises the prefix. [RFC8505] is already referenced as the registration interface to "RIFT: Routing in Fat Trees" [I-D.ietf-rift-rift] and "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks" [RFC6550] with [RFC9010].

Wireless ND (WiND) is an example instantiation of the Architecture presented in Section 3; it combines SFAAC with a Backbone Router (6BBR) ND proxy function (more in [RFC8929]) operating as a network-layer Access Point. Multiple 6BBRs placed along the wireless edge of a Backbone link handle IPv6 Neighbor Discovery and forward packets over the backbone on behalf of the registered nodes on the wireless edge. This enables to span a Subnet over an MLSN that federates edge wireless links with a high-speed, typically Ethernet, backbone (as a network-layer ESS). The ND proxy maintains the reachability for Global Unicast and link-local Addresses within the federated MLSN, either as a routing proxy where it replies with its own MAC address or as a bridging proxy that typically forwards the multicast ND messages as unicast link-layer frames to their target. The wireless nodes can form any address they want and move freely from a wireless edge link to another, without renumbering. In that case, the registrar is distributed between the 6BBR, each 6BBR maintaining only a state for the subset of the addresses that were registered to it and for which it is authoritative. When the 6BBR is not currently authoritative for a new address being registered to it, it relies on ND-Classic that is used reactively over the backbone to obtain an existing registration state in the disaggragated registrar that the 6BBRs form collectively.

This framework allows other implementations of the abstract concept of the registrar. For instance, [EVPN-SFAAC] allows to distribute the registrar in every router, and leverages EVPN as the method to synchronize the registrar state between routers. In that case, BGP

acts both as the SGP to announce the reachability of the addresses and as the synchronization protocol between the distributed registrar. All the routers know proactively the mapping for all the addresses, and there is no need for a reactive lookup as is the case for WiND. As another example, a Locator/ID Separation Protocol (LISP) Map-Resolver [RFC6830] could support the EDAR/EDAC exchange either directly or via a proxy, and serve as registrar.

The framework allows for mixed environments with registrations and ND-Classic, using [RFC8929] to perform ND proxy operations on behalf of registered address and respond to DAD and lookups from legacy nodes, and prevent registering nodes from autoconfiguring addresses that exist in legacy nodes by performing DAD on behalf of the registering nodes, more in Section 6.

4.2. links and link-local Addresses

For link-local Addresses, DAD is typically performed between communicating pairs of nodes and an NCE can be populated with a single unicast exchange. In the case of a bridging proxies, though, the link-local traffic is bridged over the backbone and the DAD must proxied there as well.

For instance, in the case of Bluetooth Low Energy (BLE) [RFC7668][IEEEstd802151], the uniqueness of link-local Addresses needs only to be verified between the pair of communicating nodes, the central router and the peripheral host. In that example, 2 peripheral hosts connected to the same central router can not have the same link-local address because the addresses would collision at the central router which could not talk to both over the same network port, unless it can separate the IP Links, e.g., based on the remote MAC address. The DAD operation from SFAAC is appropriate for that use case, but the one from ND is not, because the peripheral hosts are not on the same broadcast domain.

On the other hand, the uniqueness of GUAs and ULAs is validated at the Subnet Level, using a logical registrar that is global to the Subnet.

4.3. Subnets and Global Addresses

SFAAC extends ND-Classic 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 a router, the Hub advertises the prefix as not-on-link to the spokes in RA messages Prefix Information Options (PIO). Acting as hosts, the Spokes autoconfigure addresses from that prefix and register them to the Hub with a corresponding lifetime.

Acting as a registrar, 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.

The Hub also maintains an NCE for the registered addresses and can deliver a packet to any of them during their respective lifetimes. It can be observed that this design builds a form of network-layer 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 registrar 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 [RFC9010]. In a degraded mode, all the Hubs are connected to a same high speed backbone such as an Ethernet bridging domain where ND-Classic is operated. In that case, it is possible to federate the Hub, Spoke and Backbone nodes as a single Subnet, operating ND proxy operations [RFC8929] at the Hubs, acting as 6BBRs. It can be observed that this latter design builds a form of network-layer Infrastructure ESS.

4.4. Anycast and Multicast Addresses

While ND-Classic is defined for unicast addresses only, [I-D.ietf-6lo-multicast-registration] extends [RFC8505] for anycast and multicast IPv6 addresses. Though RPL [RFC6550], which is extended in that document, is the SGP of choice in a Low-power Lossy Network (LLN), the registration is agnostic to the SGP and the same model applies to any SGP that is capable of advertising multicast and/or anycast addresses as well as unicast.

[I-D.ietf-6lo-multicast-registration] can be used as a replacement for MLD [RFC3810] for use cases where broadcast are not desirable, and when a device push model such as SFAAC is preferred over a network pull such as MLD and ND-Classic. With [RFC8505], the host does not need to define SNMAs for its unicast addresses and does not perform the associated MLDv2 operation. With [I-D.ietf-6lo-multicast-registration], MLDv2 and its extensive use of broadcast can be totally eliminated.

In the case of anycast, the signal enables the 6BBRs to accept more than one registration for the same address, and collectively elect the registering host receives a packet for a given anycast address.

4.5. Advertising Prefixes in the SGP

By definition, prefixes longer than SPL are inside a Subnet and do not leak outside the SGP. Still, it is valid for a node to register a prefix of any size longer than SPL, and for the router to advertise the registered prefix in the SGP. This can be useful for instance to expose a /96 Prefix that is used to transport IPv4 mapped traffic [RFC6052].

[PREFIX REGISTRATION] extends [RFC8505] to enable a node that owns or is directly connected to a Prefix to register that Prefix to neighbor routers. The registration indicates that the registered Prefix can be reached via the advertising node without a loop. The prefix registration also provides a protocol-independent interface for the node to request the router to redistribute the prefix in the SGP.

5. WiND Applicability

WiND applies equally to physical links that are P2P, transit, P2MP Hub-and-Spoke, to links that provide link-layer Broadcast Domain Emulation such as Mesh-Under and Wi-Fi BSS, and to Route-Over meshes. In either cases, the IP Link abstraction in WiND is always P2P.

There is an intersection where The IP Link and the IP 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; for more details, see the introduction of [RFC8929].

There are also a number of practical use cases in the wireless world where links and subnets are not congruent:

* The IEEE Std. 802.11 infrastructure BSS enables one Subnet per AP, and emulates a broadcast domain at the link layer. The Infrastructure ESS extends that model over a backbone and recommends the use of an ND proxy [IEEE Std. 802.11] to interoperate with Ethernet-connected nodes. WiND incorporates an ND proxy to serve that need, which was missing so far.

* Bluetooth is Hub-and-Spoke at the link 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, and each peripheral host (acting as a host) 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 [RFC9030] presents the Route-Over model over an IEEE Std. 802.15.4 Time-Slotted Channel-Hopping (TSCH) [IEEEstd802154] 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 waste of resources 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. So neither Classic-ND nor WiND is really used in that space.

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.

Snooping protocols such as ND-Classic and DHCPv6 and observing data traffic sourced at the STA provides an imperfect knowledge of the state of the STA at the AP. Missing a state or a transition may result in the loss of connectivity for some of the addresses, in particular for an address that is rarely used, belongs to a sleeping node, or one in a situation of mobility. This may also result in undesirable remanent state in the AP when the STA ceases to use an IPv6 address while remaining associated. It results that snooping protocols is not a recommended technique and that it should only be used as last resort, when the WiND registration is not available to populate the state.

The recommended alternative method is to use the WiND Registration for IPv6 Addresses. This way, 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 all of its IPv6 addresses, using the Extended address Registration Option, which provides the following elements:

* The registration state has a lifetime that limits unwanted state remanence in the network.

* The registration is optionally secured using [RFC8928] to prevent address theft and impersonation.

* The registration carries a sequence number, which enables to figure the order of events in a fast mobility scenario without loss of connectivity.

The ESS mode requires a "ARP-Proxy" operation at the AP. This includes a proxy ND operation that 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 WiND proxy ND specification that associated to the address Registration is [RFC8929]. 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 backbone router either replies to NS lookups with the MAC address of the STA, or preferably forwards the lookups to the STA as link-layer unicast frames to let the STA answer. For the data plane, the backbone router acts as a normal AP and bridges the packets to the STA as usual. As a routing proxy, the backbone router replies with its own MAC address and then routes to the STA at the network 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 symmetric 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 as an Access Point does in a BSS/ESS.

While it is still possible to operate ND-Classic, the inefficiencies of the flooding operation make the associated operations even less desirable than in a BSS, and the use of WiND is highly recommended.

5.4. Case of DMB radios

IPv6 over DMB radios uses P2P links that can be formed and maintained when a pair of DMB radios transmitters are in range from one another.

5.4.1. Using ND-Classic only

DMB radios do not provide MAC level broadcast emulation. An example of that is IEEE Std. 802.11 OCB which uses IEEE Std. 802.11 MAC/PHYs but does not provide the BSS functions.

It is possible to form P2P IP Links between each individual pairs of nodes and operate ND-Classic 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 ND-Classic.

If an external mechanism avoids duplicate addresses and if the deployment ensures the connectivity between peers, a non-transit Hub-and-Spoke deployment is also possible where the Hub is the only router in the Subnet and the Prefix is advertised as not on-link.

5.4.2. Using Wireless ND

Though this can be achieved with ND-Classic, WiND is the recommended approach since it uses unicast communications which are more reliable and less impacting for other users of the medium.

The routers send RAs with a SLLAO at a regular period. The period can be indicated in the 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 messages. The RA is sent over an unspecified IP Link where it does not conflict to anyone, so DAD is not necessary at that stage.

The host instantiates an IP Link where the router's address is not a duplicate. To achieve this, it forms a link-local address that does not conflict with that of the router and registers to the router using [RFC8505]. If the router sent an RA(PIO), the host can also autoconfigure an address from the advertised prefix and register it.

```
       (host)           (router)
          |                  |
          |   DMB link       |
          |                  |
          |  ND-Classic RS   |
          |----------------->|
          |------------>     |
          |------------------------->
          |  ND-Classic RA   |
          |<-----------------|
          |                  |
          |  NS(EARO)        |
          |----------------->|
          |                  |
          |        NA(EARO)  |
          |<-----------------|
          |                  |
                        ...
          |  NS(EARO)        |
          |----------------->|
          |                  |
          |        NA(EARO)  |
          |<-----------------|
          |                  |
                        ...
```

Figure 4: RFC 8505 Registration Flow for a link-local address

The lifetime in the registration should start with a small value (X=RMin, TBD), and exponentially grow with each re-registration 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 physical link flapping does not affect the state of the registration and when a physical link comes back up, the active registrations (i.e., registrations for which lifetime is not elapsed) are still usable. Packets should be held or destroyed when the IP Link is down.

P2P links may be federated in Hub-and-Spoke by edge routers, and the Subnet may comprise multiple edge routers, in which case each advertises its registered addresses over the SGP as illustrated in Figure 5. Note that the Extended DAR/DAC exchange can be omitted if it can be replaced with the information that is distributed in the SGP, see for instance [RFC9010] which applies to IoT environments, which needs only the first EDAR/EDAC exchange, and [EVPN-SFAAC], for EVPN-based wireless deployments in enterprise and campus, which does not use EDAR/EDAC at all.

```
         (host)             (router)       (registrar)
            |                  |               |
            |                  |               |
            |  ND-Classic RS   |               |
            |----------------->|               |
            |----------->      |               |
            |--------------------->            |
            |  ND-Classic RA   |               |
            |<-----------------|               |
            |                  |               |
            |  NS(EARO)        |               |
            |----------------->|               |
            |                  | Extended DAR  |
            |                  |-------------->|
            |                  |               |
            |                  | Extended DAC  |
            |                  |<--------------|
            |       NA(EARO)   |               |
            |<-----------------|<inject in SGP> ->
                          ...
            |                  |               |
            |  NS(EARO)        |               |
            |----------------->|               |
            |                  | Extended DAR  |
            |                  |-------------->|
            |                  |               |
            |                  | Extended DAC  |
            |                  |<--------------|
            |       NA(EARO)   |               |
            |<-----------------|<maintain in SGP> ->
            |                  |               |
                          ...
```

Figure 5: RFC 8505 Registration Flow for a Global address

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.

As opposed to unicast addresses, there is no concept of duplication with multicast and anycast addresses, and there might be multiple registrations from multiple parties for the same address. The router conserves one registration per party per multicast or anycast

address, but injects the route into the SGP only once for each address, asynchronously to the registration. On the other hand, the validation exchange with the registrar is still needed if the router checks the right for the host to listen to the anycast or multicast address.

```
       6LoWPAN Node           6LR             6LBR
         (host1)           (router)        (registrar)
            |                 |               |
            |   DMB link      |               |
            |                 |               |
            |  ND-Classic RS  |               |
            |----------------->|               |
            |------------>    |               |
            |------------------------->       |
            |  ND-Classic RA  |               |
            <-----------------|               |
            |                 |               |
            |  NS(EARO)       |               |
            |----------------->|               |
            |                 | Extended DAR  |
            |                 |-------------->|
            |                 |               |
            |                 | Extended DAC  |
            |                 <--------------|
            |        NA(EARO) |               |
            <-----------------|<inject in SGP> ->
            |                 |
                     ...
         (host2)           (router)           6LBR
            |  NS(EARO)       |               |
            |----------------->|               |
            |                 |               |
            |                 | Extended DAR  |
            |                 |-------------->|
            |                 |               |
            |                 | Extended DAC  |
            |                 <--------------|
            |        NA(EARO) |               |
            <-----------------|               |
                     ...
         (host1)           (router)
            |  NS(EARO)       |               |
            |----------------->|               |
            |                 |               |
            |        NA(EARO) |               |
            <-----------------|               |
                     ...
            |                 |<maintain in SGP> ->
                     ...
```

Figure 6: Registration Flow for an anycast or multicast address

6. Coexistence with ND-Classic

The framework allows for a mixed environment with both models, ND-Classic and SFAAC, coexist. With [RFC8929], an ethernet backbone link operating ND-Classic federates a MultiLink Subnet (MLSN) of wireless links and/or meshes, and routers called Backbone Routers (6BBR) operate as ND proxies.

In a wireless deployments, the Backbone Routers are placed along the wireless edge of a backbone (e.g., in Access Points) and federate multiple wireless links to form a single MLSN, echoing the Wi-Fi ESS structure but at the network layer, as shown in Figure 7. In that example, Optimistic Duplicate address Detection (ODAD) [RFC4429] allows the IPv6 address to be used before completion of DAD, so the whole flow below can happen in the milliseconds that follow the Wi-Fi association.

```
          6LN(STA)          6BBR(AP)          6LBR          default GW
             |                 |                |                |
             | Wi-Fi Access BSS |   IPv6 Backbone (e.g., Ethernet)  |
             |                 |                |                |
             |  RS(multicast)  |                |                |
             |----------------->|                |                |
             | RA(PIO, Unicast) |                |                |
             <-----------------|                |                |
             |   NS(EARO)      |                |                |
             |----------------->|                |                |
             |                 |  Extended DAR  |                |
             |                 |--------------->|                |
             |                 |  Extended DAC  |                |
             |                 <---------------|                |
             |                 |                |                |
             |                 |     NS-DAD(EARO, multicast)     |
             |                 |-------->                        |
             |                 |------------------->             |
             |                 |----------------------------------->|
             |                 |                                 |
             |                 |      RS(no SLLAO, for ODAD)     |
             |                 |----------------------------------->|
             |                 | if (no fresher Binding) NS(Lookup) |
             |                 |                  <----------------|
             |                 |          <-------------------------|
             |                 <----------------------------------|
             |                 |      NA(SLLAO, not(O), EARO)    |
             |                 |----------------------------------->|
             |                 |           RA(unicast)           |
             |                 <----------------------------------|
             |                 |                                 |
             |           IPv6 Packets in Optimistic Mode         |
             <------------------------------------------------------>|
             |                 |                                 |
             |                 |                                 |
             |  NA(EARO)       |  <DAD timeout>                  |
             <--------- -------|                                 |
             |                 |                                 |
```

Figure 7: Initial Registration Flow to a 6BBR Acting as a ND Proxy

In use cases such as overlays, a Map Resolver acting as 6LBR may be deployed on the Backbone Link to serve the whole Subnet, and EDAR/EDAC messages (or equivalent alternates, e.g., using LISP) can be used in combination with DAD to enable coexistence with ND-Classic over the backbone. The 6LBR proactive operations will then coexist on the Backbone with the reactive ND-Classic operation. Nodes that support [UNICAST AR] may query the mappings they look up 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. This model also enables a snooping switch acting as ND proxy to intercept Ar and DAD NS messages and perform unicast lookups to the resolver and only broadcast the original NS messgae when the unicast lookup fails.

Note that the RS sent initially by the 6LN (e.g., a Wi-Fi STA) is transmitted as a multicast, but since it is intercepted by the 6BBR, it is never effectively broadcast at link layer. The multiple arrows in Figure 7 associated to the ND messages on the backbone denote a real link-layer broadcast.

It is not necessary to isolate the registering nodes in separate physical links, but it is preferred with wireless links as it enables to isolate the broadcast domain on the ethernet link from the wireless links at the Access Points. In other words, the 6BBRs collectively form a global registrar for the Subnet that aggregates the information in each local registrar in the 6LBR. The global registrar is distributed between the 6BBRs, which leverage ND-Classic (AR and DAD) to lookup information that they do not have locally from the other 6BBRs and from nodes that are connected to the backbone.

In the case of wireless meshes, RPL may be used as local SGP in each mesh as shown in Figure 8. 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 [RFC9030].

```
       6LoWPAN Node        6LR             6LBR            6BBR
        (RPL leaf)       (router)         (root)
            |               |               |               |
            |  6LoWPAN ND   |6LoWPAN ND+RPL | 6LoWPAN ND    | ND-Classic
            |   LLN link    |Route-Over mesh|Ethernet/serial| Backbone
            |               |               |               |
            |  ND-Classic RS  |               |               |
            |-------------->|               |               |
            |----------->   |               |               |
            |------------------>            |               |
            |  ND-Classic RA  |               |               |
            |<--------------|               |               |
            |               |    <once>     |               |
            |  NS(EARO)     |               |               |
            |-------------->|               |               |
            | 6LoWPAN ND    | Extended DAR  |               |
            |               |-------------->|               |
            |               |               |  NS(EARO)     |
            |               |               |-------------->|
            |               |               |  proxy registration
            |               |               |               |
            |               |               |        NS-DAD (EARO)
            |               |               |               |------>
            |               |               |               ND proxy
            |               |               |               |
            |               |               |  NA(EARO)     |<timeout>
            |               |               |<--------------|
            |               | Extended DAC  |               |
            |               |<--------------|               |
            |  NA(EARO)     |               |               |
            |<--------------|               |               |
            |               |               |               |
```

Figure 8: Initial Registration Flow with 6BBR ND-Proxy

7. Privacy Considerations

ND-Classic exposes all addresses to all nodes in the Subnet, which is a privacy issue and makes impersonation attacks easier. In contrast, in switched and wireless networks, a host is not on-path of the unicast packets for registration and for data for other hosts, so it cannot snoop the other addresses in the network. A rogue host can only discover the existence of an addresses by trying and failing to register that address, but for that it would need to fathom which address to try and that can be very hard in, say, a SPL=64 address space that is used wisely. For that reason, this framework limits that knowledge to on-path snooping switches, to the routers and to the abstract registrar, which are typically more controlled / harder

to hack than the common host. When ND-Classic and SFAAC coexist within the same Subnet, all addresses in the Subnet, including registered addresses, can be snooped in the broadcast domain where ND-Classic is operated. It makes sense to reduce that domain to the maximum and control which device connect to it.

The exposure of addresses can be further reduced if the exchanges with the registrar (e.g., EDAR and EDAC) are encrypted, e.g., using a public key associated with the registrar. The registration and routing exchanges could also be encrypted to avoid leaking the addresses to snopping switches, but this is typically not done inside a physical site where the networking gear is tightly controlled. In a DCI environment, the inter-side (SD-WAN) links are typically encrypted, to the exchanges are obfuscated from an on-path listener.

8. Security Considerations

The registration model [RFC8505] implemented by this framework allows for a model where the ingress routers have a full knowledge of all the addresses in the Subnet. The ingress router can thus discard any packet which destination appears to be in the Subnet from its prefix, but is not known, meaning that it does not exist. This mostly defeats the traditional DoS scanning attacks against ND whereby the remote attacker sends volumes of packets to as many non-existent addresses to saturate the Neighbor Cache and clog the Subnet internal bandwidth in broadcasts.

When the ownership verifier is cryptographic, this framework enables a zerotrust model whereby only the address owner can advertise an address in ND and as source of data packets, more in [RFC8928]. This defeats the classical impersonation attacks against ND-Classic and allows to disable the proprietary middlebox software aimed at protecting the address ownership against onlink rogues.

9. IANA Considerations

This specification does not require IANA action.

10. Contributors

Brian Carpenter Provided support, hints, and text snippets throughout the lifetime of the I-Draft

11. Acknowledgments

Many thanks to the participants of the 6lo WG where a lot of the work discussed here happened, following work at ROLL, 6TiSCH, and mainly 6LoWPAN.

Special thanks to Brian Carpenter and Eric Levy-Abegnoli who provided support and useful comments throughout the development of this architecture, and to Erik Nordmark and Zach Shelby with whom this work really started during IETF 72 in Dublin.

Also many thanks to Eduard Vasilenko, XiPeng Xiao, Behcet Sarikaya for their contributions and support to this work at 6MAN, v6Ops, and ETSI IPE.

12. Normative References

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <https://www.rfc-editor.org/info/rfc4191>.

[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, <https://www.rfc-editor.org/info/rfc4861>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <https://www.rfc-editor.org/info/rfc4862>.

[RFC5942] Singh, H., Beebee, W., and E. Nordmark, "IPv6 Subnet Model: The Relationship between Links and Subnet Prefixes", RFC 5942, DOI 10.17487/RFC5942, July 2010, <https://www.rfc-editor.org/info/rfc5942>.

[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, <https://www.rfc-editor.org/info/rfc6052>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <https://www.rfc-editor.org/info/rfc6275>.

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <https://www.rfc-editor.org/info/rfc6830>.

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

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

[RFC8505] Thubert, P., Ed., Nordmark, E., Chakrabarti, S., and C. Perkins, "Registration Extensions for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Neighbor Discovery", RFC 8505, DOI 10.17487/RFC8505, November 2018, <https://www.rfc-editor.org/info/rfc8505>.

[RFC8928] Thubert, P., Ed., Sarikaya, B., Sethi, M., and R. Struik, "Address-Protected Neighbor Discovery for Low-Power and Lossy Networks", RFC 8928, DOI 10.17487/RFC8928, November 2020, <https://www.rfc-editor.org/info/rfc8928>.

[RFC8929] Thubert, P., Ed., Perkins, C.E., and E. Levy-Abegnoli, "IPv6 Backbone Router", RFC 8929, DOI 10.17487/RFC8929, November 2020, <https://www.rfc-editor.org/info/rfc8929>.

13. Informative References

[RFC3963] Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, "Network Mobility (NEMO) Basic Support Protocol", RFC 3963, DOI 10.17487/RFC3963, January 2005, <https://www.rfc-editor.org/info/rfc3963>.

[RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <https://www.rfc-editor.org/info/rfc3810>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4429] Moore, N., "Optimistic Duplicate Address Detection (DAD) for IPv6", RFC 4429, DOI 10.17487/RFC4429, April 2006, <https://www.rfc-editor.org/info/rfc4429>.

[RFC4903] Thaler, D., "Multi-Link Subnet Issues", RFC 4903, DOI 10.17487/RFC4903, June 2007, <https://www.rfc-editor.org/info/rfc4903>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <https://www.rfc-editor.org/info/rfc6775>.

[RFC7668] Nieminen, J., Savolainen, T., Isomaki, M., Patil, B., Shelby, Z., and C. Gomez, "IPv6 over BLUETOOTH(R) Low Energy", RFC 7668, DOI 10.17487/RFC7668, October 2015, <https://www.rfc-editor.org/info/rfc7668>.

[RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <https://www.rfc-editor.org/info/rfc7217>.

[RFC7834] Saucez, D., Iannone, L., Cabellos, A., and F. Coras, "Locator/ID Separation Protocol (LISP) Impact", RFC 7834, DOI 10.17487/RFC7834, April 2016, <https://www.rfc-editor.org/info/rfc7834>.

[RFC8273] Brzozowski, J. and G. Van de Velde, "Unique IPv6 Prefix per Host", RFC 8273, DOI 10.17487/RFC8273, December 2017, <https://www.rfc-editor.org/info/rfc8273>.

[RFC8365] Sajassi, A., Ed., Drake, J., Ed., Bitar, N., Shekhar, R., Uttaro, J., and W. Henderickx, "A Network Virtualization Overlay Solution Using Ethernet VPN (EVPN)", RFC 8365, DOI 10.17487/RFC8365, March 2018, <https://www.rfc-editor.org/info/rfc8365>.

[RFC8415] Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415, DOI 10.17487/RFC8415, November 2018, <https://www.rfc-editor.org/info/rfc8415>.

[RFC8981] Gont, F., Krishnan, S., Narten, T., and R. Draves, "Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6", RFC 8981, DOI 10.17487/RFC8981, February 2021, <https://www.rfc-editor.org/info/rfc8981>.

[I-D.ietf-rift-rift] Przygienda, T., Sharma, A., Thubert, P., Rijsman, B., Afanasiev, D., and J. Head, "RIFT: Routing in Fat Trees", Work in Progress, Internet-Draft, draft-ietf-rift-rift-16, 12 September 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-rift-rift-16>.

[RFC9010] Thubert, P., Ed. and M. Richardson, "Routing for RPL (Routing Protocol for Low-Power and Lossy Networks) Leaves", RFC 9010, DOI 10.17487/RFC9010, April 2021, <https://www.rfc-editor.org/info/rfc9010>.

[DAD ISSUES] Yourtchenko, A. and E. Nordmark, "A survey of issues related to IPv6 Duplicate Address Detection", Work in Progress, Internet-Draft, draft-yourtchenko-6man-dad-issues-01, 3 March 2015, <https://datatracker.ietf.org/doc/html/draft-yourtchenko-6man-dad-issues-01>.

[MCAST EFFICIENCY] Vyncke, E., Thubert, P., Levy-Abegnoli, E., and A. Yourtchenko, "Why Network-Layer Multicast is Not Always Efficient At Datalink Layer", Work in Progress, Internet-Draft, draft-vyncke-6man-mcast-not-efficient-01, 14 February 2014, <https://datatracker.ietf.org/doc/html/draft-vyncke-6man-mcast-not-efficient-01>.

[RFC9030] Thubert, P., Ed., "An Architecture for IPv6 over the Time-Slotted Channel Hopping Mode of IEEE 802.15.4 (6TiSCH)", RFC 9030, DOI 10.17487/RFC9030, May 2021, <https://www.rfc-editor.org/info/rfc9030>.

[MCAST PROBLEMS] Perkins, C. E., McBride, M., Stanley, D., Kumari, W. A., and J. C. Zúñiga, "Multicast Considerations over IEEE 802 Wireless Media", Work in Progress, Internet-Draft, draft-ietf-mboned-ieee802-mcast-problems-15, 28 July 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-mboned-ieee802-mcast-problems-15>.

[SAVI] Bi, J., Wu, J., Lin, T., Wang, Y., and L. He, "A SAVI Solution for WLAN", Work in Progress, Internet-Draft, draft-bi-savi-wlan-24, 13 November 2022, <https://datatracker.ietf.org/doc/html/draft-bi-savi-wlan-24>.

[UNICAST AR] Thubert, P. and E. Levy-Abegnoli, "IPv6 Neighbor Discovery Unicast Lookup", Work in Progress, Internet-Draft, draft-thubert-6lo-unicast-lookup-02, 8 November 2021, <https://datatracker.ietf.org/doc/html/draft-thubert-6lo-unicast-lookup-02>.

[PREFIX REGISTRATION] Thubert, P., "IPv6 Neighbor Discovery Prefix Registration", Work in Progress, Internet-Draft, draft-thubert-6lo-prefix-registration-02, 3 January 2023, <https://datatracker.ietf.org/doc/html/draft-thubert-6lo-prefix-registration-02>.

[DAD APPROACHES] Nordmark, E., "Possible approaches to make DAD more robust and/or efficient", Work in Progress, Internet-Draft, draft-nordmark-6man-dad-approaches-02, 19 October 2015, <https://datatracker.ietf.org/doc/html/draft-nordmark-6man-dad-approaches-02>.

[EVPN-SFAAC] Thubert, P., Przygienda, T., and J. Tantsura, "Secure EVPN MAC Signaling", Work in Progress, Internet-Draft, draft-thubert-bess-secure-evpn-mac-signaling-03, 31 January 2022, <https://datatracker.ietf.org/doc/html/draft-thubert-bess-secure-evpn-mac-signaling-03>.

[I-D.ietf-6lo-multicast-registration] Thubert, P., "IPv6 Neighbor Discovery Multicast and Anycast Address Listener Subscription", Work in Progress, Internet-Draft, draft-ietf-6lo-multicast-registration-14, 8 March 2023, <https://datatracker.ietf.org/api/v1/doc/document/draft-ietf-6lo-multicast-registration/>.

[IEEE Std. 802.15.4] 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 Std. 802.11]
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)".

[IEEEstd802154]
IEEE standard for Information Technology, "IEEE Standard for Local and metropolitan area networks -- Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)".

[IEEE Std. 802.1]
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".

Authors' Addresses

Pascal Thubert (editor)
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
06254 Mougins - Sophia Antipolis
France
Phone: +33 497 23 26 34
Email: pthubert@cisco.com

Michael C. Richardson
Sandelman Software Works
Email: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/