

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

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

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

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on July 12, 2022.

Copyright Notice

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

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

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

Table of Contents

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

1. Introduction

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

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

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

2. Conventions

2.1. Requirement Language

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

2.2. Abbreviations

Abbreviations used in this document:

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

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

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

4. Procedure

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

4.1. Ingress Node

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

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

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

4.2. Intermediate SR Segment Endpoint Node

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

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

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

4.3. Egress Node

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

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

5. IANA Considerations

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

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

6. Security Considerations

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

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

7. Acknowledgements

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

8. References

8.1. Normative References

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

8.2. Informative References

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

Internet-Draft

In-situ OAM SRv6 encapsulation

Authors' Addresses

Zafar Ali
Cisco Systems, Inc.

Email: zali@cisco.com

Rakesh Gandhi
Cisco Systems, Inc.
Canada

Email: rgandhi@cisco.com

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cf@cisco.com

Frank Brockners
Cisco Systems, Inc.
Germany

Email: fbrockne@cisco.com

Nagendra Kumar Nainar
Cisco Systems, Inc.

Email: naikumar@cisco.com

Carlos Pignataro
Cisco Systems, Inc.

Email: cpignata@cisco.com

Cheng Li
Huawei

Email: chenglil13@huawei.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

Gaurav Dawra
LinkedIn

Email: gdawra.ietf@gmail.com

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: 11 November 2022

R. Hinden
Check Point Software
G. Fairhurst
University of Aberdeen
10 May 2022

IPv6 Minimum Path MTU Hop-by-Hop Option
draft-ietf-6man-mtu-option-15

Abstract

This document specifies a new IPv6 Hop-by-Hop option that is used to record the minimum Path MTU along the forward path between a source host to a destination host. The recorded value can then be communicated back to the source using the return Path MTU field in the 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 11 November 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
1.1. Example Operation	3
1.2. Use of the IPv6 Hop-by-Hop Options Header	4
2. Motivation and Problem Solved	5
3. Requirements Language	6
4. Applicability Statements	6
5. IPv6 Minimum Path MTU Hop-by-Hop Option	6
6. Router, Host, and Transport Layer Behaviors	8
6.1. Router Behavior	8
6.2. Host Operating System Behavior	8
6.3. Transport Layer Behavior	9
6.3.1. Including the Option in an Outgoing Packet	10
6.3.2. Validation of the Packet that includes the Option	12
6.3.3. Receiving the Option	12
6.3.4. Using the Rtn-PMTU Field	13
6.3.5. Detecting Path Changes	14
6.3.6. Detection of Dropping Packets that include the Option	14
7. IANA Considerations	14
8. Security Considerations	14
8.1. Router Option Processing	15
8.2. Network Layer Host Processing	15
8.3. Validating use of the Option Data	16
8.4. Direct use of the Rtn-PMTU Value	16
8.5. Using the Rtn-PMTU Value as a Hint for Probing	17
8.6. Impact of Middleboxes	17
9. Experiment Goals	17
10. Implementation Status	18
11. Acknowledgments	18
12. Change log [RFC Editor: Please remove]	18
13. References	21
13.1. Normative References	21
13.2. Informative References	22
Appendix A. Examples of Usage	24
Authors' Addresses	26

1. Introduction

This document specifies a new IPv6 Hop-by-Hop (HBH) Option to record the minimum Maximum Transmission Unit (MTU) along the forward path between a source and a destination host. The source host creates a packet with this option and initializes the Min-PMTU field with the value of the MTU for the outbound link that will be used to forward the packet towards the destination host.

At each subsequent hop where the option is processed, the router compares the value of the Min-PMTU Field in the option and the MTU of its outgoing link. If the MTU of the link is less than the Min-PMTU, it rewrites the value in the option data with the smaller value. When the packet arrives at the destination host, the host can send the value of the minimum reported MTU for the path back to the source host using the Rtn-PMTU field in the option. The source host can then use this value as input to the method that sets the Path MTU (PMTU) used by upper layer protocols.

The IPv6 Minimum Path MTU Hop-by-Hop (MinPMTU HBH) Option is designed to work with packet sizes that can be specified in the IPv6 header. The maximum packet size that can be specified in an IPv6 header is 65,535 octets (2^{16}).

This method has the potential to complete Path MTU discovery in a single round trip time, even over paths that have successive links each with a lower MTU.

The mechanism defined in this document is focused on Unicast, it does not describe Multicast. That is left for future work.

1.1. Example Operation

The figure below illustrates the operation of the method. In this case, the path between the source host and the destination host comprises three links, the source has a link MTU of size MTU-S, the link between routers R1 and R2 has an MTU of size 9000 bytes, and the final link to the destination has an MTU of size MTU-D.

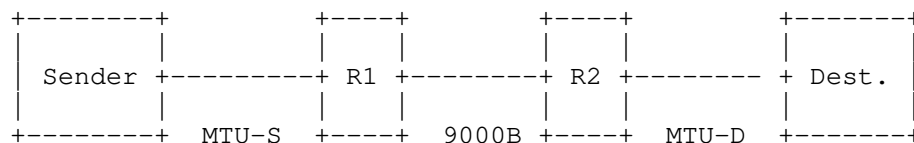


Figure 1

Three scenarios are described:

- * Scenario 1, considers all links to have an 9000 byte MTU and the method is supported by both routers. The initial Min-PMTU is not modified along the path, and therefore the PMTU is 9000 bytes.
- * Scenario 2, considers the link between R2 and destination host (MTU-D) to have an MTU of 1500 bytes. This is the smallest MTU, router R2 updates the Min-PMTU to 1500 bytes and the method

correctly updates the PMTU to 1500 bytes. Had there been another smaller MTU at a link further along the path that also supports the method, the lower MTU would also have been detected.

- * Scenario 3, considers the case where the router preceding the smallest link (R2) does not support the method, and the link to the destination host (MTU-D) has an MTU of 1500 bytes. Therefore, router R2 does not update the Min-PMTU to 1500 bytes. The method then fails to detect the actual PMTU.

In Scenarios 2 and 3, a lower PMTU would also fail to be detected in the case where PMTUD had been used and an ICMPv6 Packet Too Big (PTB) message had not been delivered to the sender [RFC8201].

These scenarios are summarized in the table below. "H" in R1 and/or R2 columns means the router understands the MinPMTU HBH option.

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

Figure 2

1.2. Use of the IPv6 Hop-by-Hop Options Header

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

- * The Hop-by-Hop Options header is not inserted or deleted, but may be examined or processed by any node along a packet's delivery path, until the packet reaches the node (or each of the set of nodes, in the case of multicast) identified in the Destination Address field of the IPv6 header. The Hop-by-Hop Options header, when present, must immediately follow the IPv6 header. Its presence is indicated by the value zero in the Next Header field of the IPv6 header.
- * NOTE: While [RFC2460] required that all nodes must examine and process the Hop-by-Hop Options header, it is now expected that nodes along a packet's delivery path only examine and process the Hop-by-Hop Options header if explicitly configured to do so.

The Hop-by-Hop Option defined in this document is designed to take advantage of this property of how Hop-by-Hop options are processed. Nodes that do not support this Option SHOULD ignore them. This can mean that the Min-PMTU value 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 mechanisms defined in [RFC8201] are known to not work well in all environments. It fails to work in various cases, including when nodes in the middle of the network do not send ICMPv6 PTB messages, or rate-limited ICMPv6 messages, or do not have a return path to the source host.

This results in many transport layer connections being configured to use smaller packets (e.g., 1280 bytes) by default and makes it difficult to take advantage of paths with a larger PMTU where they do exist. Applications that send large packets are forced to use IPv6 Fragmentation [RFC8200], which can reduce the reliability of Internet communication [RFC8900].

Encapsulations and network-layer tunnels further reduce the payload size available for a transport protocol to use. Also, some use-cases increase packet overhead, for example, Network Virtualization Using Generic Routing Encapsulation (NVGRE) [RFC7637] encapsulates L2 packets in an outer IP header and does not allow IP Fragmentation.

Sending larger packets can improve host performance, e.g., avoiding limits to packet processing by the packet rate. For example, the packet per second rate required to reach wire speed on a 10G link with 1280 byte packets is about 977K packets per second (pps), vs. 139K pps for 9000 byte packets.

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

A similar mechanism was proposed in 1988 for IPv4 in [RFC1063] by Jeff Mogul, C. Kent, Craig Partridge, and Keith McCloghrie. It was later obsoleted in 1990 by [RFC1191], the current deployed approach to Path MTU Discovery. In contrast, the method described in this document uses the Hop-by-Hop option of IPv6. It does not replace PMTUD [RFC8201], PLPPMTUD [RFC4821] or Datagram PLPMTUD [RFC8899], but rather is designed to compliment these methods.

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

The Path MTU option is designed for environments where there is control over the hosts and nodes that connect them, and where there is more than one MTU size in use. For example, in Data Centers and on paths between Data Centers, to allow hosts to better take advantage of a path that is able to support a large PMTU.

The design of the option is sufficiently simple that it can be executed on a router's fast path. A successful experiment depends on both implementation by host and router vendors and deployment by operators. The contained use-case of connections within and between Data Centers could be a driver for deployment.

The method could also be useful in other environments, including the general Internet, and offers advantage when this Hop-by-Hop Option is supported on all paths. The method is more robust when used to probe the path using packets that do not carry application data and when also paired with a method such as Packetization Layer PMTUD [RFC4821] or Datagram PLPMTUD [RFC8899].

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

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

Option Type	Option Data Len	Option Data		
BBCTTTTT	00000100	Min-PMTU	Rtn-PMTU	R

Option Type (see Section 4.2 of [RFC8200]):

BB 00 Skip over this option and continue processing.

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

TTTTT 10000 Option Type assigned from IANA [IANA-HBH].

Length: 4 The size of the value field in Option Data field supports PMTU values from 0 to 65,534 octets, the maximum size represented by the Path MTU option.

Min-PMTU: n 16-bits. The minimum MTU recorded along the path in octets, reflecting the smallest link MTU that the packet experienced along the path. A value less than the IPv6 minimum link MTU [RFC8200] MUST be ignored.

Rtn-PMTU: n 15-bits. The returned Path MTU field, carrying the 15 most significant bits of the latest received Min-PMTU field for the forward path. 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 host should include the received Rtn-PMTU field updated by the reported Min-PMTU value when the destination host is to send a PMTU Option back to the source host.

Figure 3

NOTE: The encoding of the final two octets (Rtn-PMTU and R-Flag) could be implemented by a mask of the latest received Min-PMTU value with 0xFFFE, discarding the right-most bit and then performing a logical 'OR' with the R-Flag value of the sender. This encoding fits in the minimum-sized Hop-by-Hop Option header.

6. Router, Host, and Transport Layer Behaviors

6.1. Router Behavior

Routers that are not configured to support Hop-by-Hop Options are not expected to examine or process the contents of this option [RFC8200].

Routers that support Hop-by-Hop Options, but are not configured to support this option SHOULD skip over this option and continue to processing the header [RFC8200].

Routers that support this option MUST compare the value of the Min-PMTU field with the MTU configured for the outgoing link. If the MTU of the outgoing link is less than the Min-PMTU, the router rewrites the Min-PMTU in the Option to use the smaller value. (The router processing is performed without checking the valid range of the Min-PMTU or the Rtn-PMTU fields.)

A router MUST ignore and MUST NOT change the Rtn-PMTU field or the R-Flag in the option.

6.2. Host Operating System Behavior

The PMTU entry associated with the destination in the host's destination cache [RFC4861] SHOULD be updated after detecting a change using the IPv6 Minimum Path MTU Hop-by-Hop Option. This cached value can be used by other flows that share the host's destination cache.

The value in the host destination cache SHOULD be used by PLPMTUD to select an initial PMTU for a flow. The cached PMTU is only increased by PLPMTUD when the Packetization Layer determines the path actually supports a larger PMTU [RFC4821] [RFC8899].

When requested to send an IPv6 packet with the MinPMTU HBH option, the source host includes the option in an outgoing packet. The source host MUST fill the Min-PMTU field with the MTU configured for the link over which it will send the packet on the next hop towards the destination host.

When a host includes the option in a packet it sends, the host SHOULD set the Rtn-PMTU field to the previously cached value of the received Minimum Path MTU for the flow in the Rtn-PMTU field (see Section 6.3.3). If this value is not set (for example, because there is no cached reported Min-PMTU value), the Rtn-PMTU field value MUST be set to zero.

The source host MAY request the destination host to return the reported Min-PMTU value by setting the R-Flag in the option of an outgoing packet. The R-Flag SHOULD NOT be set when the MinPMTU HBH Option was sent solely to provide requested feedback on the return Path MTU to avoid each response generating another response.

The destination host controls when to send a packet with this option in response to an R-flag, as well as which packets to include it in. The destination host MAY limit the rate at which it sends these packets.

A destination host only sets the R Flag if it wishes the source host to also return the discovered PMTU value for the path from the destination to the source.

The normal sequence of operation of the R-Flag using the terminology from the diagram in Figure 1 is:

1. The source sends a probe to the destination. The sender sets the R-Flag.
2. The destination responds by sending a probe including the received Min-PMTU as the Rtn-PMTU. A destination that does not wish to probe the return path sets the R-Flag to 0.

6.3. Transport Layer Behavior

This Hop-by-Hop option is intended to be used with a path MTU discovery method.

PLPMTUD [RFC9000] uses probe packets for two distinct functions:

- * Probe packets are used to confirm connectivity. Such probes can be of any size up to the PLPMTU. These probe packets are sent to solicit a response use the path to the remote node. These probe packets can carry the Hop-by-Hop PMTU option, providing the final size of the packet does not exceed the current PLPMTU. After validating that the packet originates from the path (section 4.6.1), the PLPMTUD method can use the reported size from the Hop-by-Hop option as the next search point when it resumes the search algorithm. (This use resembles the use of the PTB_SIZE information in section 4.6.2 of [RFC8899])
- * A second use of probe packets is to explore if a path supports a packet size greater than the current PLPMTU. If this probe packet is successfully delivered (as determined by the source host), then the PLPMTU is raised to the size of the successful probe. These probe packets do not usually set the Path MTU Hop-by-Hop option.

See section 1.2 of [RFC8899]. Section 4.1 of [RFC8899] also describes ways that a Probe Packet can be constructed, depending on whether the probe packets carry application data.

- * The PMTU Hop-by-Hop Option Probe can be sent on packets that include application data, but needs to be robust to potential loss of the packet (i.e., with the possibility that retransmission might be needed if the packet is lost).
- * Using a PMTU Probe on packets that do not carry application data will avoid the need for loss recovery if a router on the path drops packets that set this option. (This avoids the transport needing to retransmit a lost packet that includes this option.) This is the normal default format for both uses of probes.

6.3.1. Including the Option in an Outgoing Packet

The upper layer protocol can request the MinPMTU HBH option to be included in an outgoing IPv6 packet. A transport protocol (or upper layer protocol) can include this option only on specific packets used to test the path. This option does not need to be included in all packets belonging to a flow.

NOTE: Including this option in a large packet (e.g., one larger than the present PMTU) is not likely to be useful, since the large packet would itself be dropped by any link along the path with a smaller MTU, preventing the Min-PMTU information from reaching the destination host.

Discussion:

- * In the case of TCP, the option could be included in a packet that carries a TCP segment sent after the connection is established. A segment without data could be used, to avoid the need to retransmit this data if the probe packet is lost. The discovered value can be used to inform PLPMTUD [RFC4821].

NOTE: A TCP SYN can also negotiate the Maximum Segment Size (MSS), which acts as an upper limit to the packet size that can be sent by a TCP sender. If this option were to be included in a TCP SYN, it could increase the probability that the SYN segment is lost when routers on the path drop packets with this option (see Section 6.3.6), which could have an unwanted impact on the result of racing options [I-D.ietf-taps-arch] or feature negotiation.

- * 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 hosts [RFC8085].
- * Simple-exchange protocols (i.e., low data-volume applications [RFC8085] that only send one or a few packets per transaction), might assume that the PMTU is symmetrical. That is, the PMTU is the same in both directions, or at least not smaller for the return path. This optimization does not hold when the paths are not symmetric.
- * The MinPMTU HBH option can be used with ICMPv6 [RFC4443]. This requires a response from the remote node and therefore is restricted to use with ICMPv6 echo messages. The MinPMTU HBH option could provide additional information about the PMTU that might be supported by a path. This could be use as a diagnostic tool to measure the PMTU of a path. As with other uses, the actual supported PMTU is only confirmed after receiving a response to a subsequent probe of the PMTU size.
- * A datagram transport can utilise DPLPMTUD [RFC8899]. For example, QUIC (see section 14.3 of [RFC9000]), can use DPLPMTUD to determine whether the path to a destination will support a desired maximum datagram size. When using the IPv6 MinPMTU HBH option, the option could be added to an additional QUIC PMTU Probe that is of minimal size (or one no larger than the currently supported PMTU size). Once the return Path MTU value in the MinPMTU HBH option has been learned, DPLPMTUD can be triggered to test for a larger PLPMTU using an appropriately sized PLPMTU Probe Packet (see section 5.3.1 of [RFC8899]).
- * The use of this option with DNS and DNSSEC over UDP is expected to work for paths where the PMTU is symmetric. The DNS server will learn the PMTU from the DNS query messages. If the Rtn-PMTU value is smaller, then a large DNSSEC response might be dropped and the known problems with PMTUD will then occur. DNS and DNSSEC over transport protocols that can carry the PMTU ought to work.
- * This method also can be used with Anycast to discover the PMTU of the path, but the use needs to be aware that the Anycast binding might change.

6.3.2. Validation of the Packet that includes the Option

An upper layer protocol (e.g., transport endpoint) using this option needs to provide protection from data injection attacks by off-path devices [RFC8085]. This requires a method to assure that the information in the Option Data is provided by a node on the path. This validates that the packet forms a part of an existing flow, using context available at the upper layer. For example, a TCP connection or UDP application that maintains the related state and uses a randomized ephemeral port would provide this basic validation to protect from off-path data injection, see Section 5.1 of [RFC8085]. IPsec [RFC4301] and TLS [RFC8446] provide greater assurance.

The upper layer discards any received packet when the packet validation fails. When packet validation fails, the upper layer **MUST** also discard the associated Option Data from the MinPMTU HBH option without further processing.

6.3.3. Receiving the Option

For a connection-oriented upper layer protocol, caching of the received Min-PMTU could be implemented by saving the value in the connection context at the transport layer. A connection-less upper layer (e.g., one using UDP), requires the upper layer protocol to cache the value for each flow it uses.

A destination host that receives a MinPMTU HBH Option with the R-Flag **SHOULD** include the MinPMTU HBH option in the next outgoing IPv6 packet for the corresponding flow.

A simple mechanism could only include this option (with the Rtn-PMTU field set) the first time this option is received or when it notifies a change in the Minimum Path MTU. This limits the number of packets including the option packets that are sent. However, this does not provide robustness to packet loss or recovery after a sender loses state.

Discussion:

- * Some upper layer protocols send packets less frequently than the rate at which the host receives packets. This provides less frequent feedback of the received Rtn-PMTU value. However, a host always sends the most recent Rtn-PMTU value.

6.3.4. Using the Rtn-PMTU Field

The Rtn-PMTU field provides an indication of the PMTU from on-path routers. It does not necessarily reflect the actual PMTU between the source and destination hosts. Care therefore needs to be exercised in using the Rtn-PMTU value. Specifically:

- * The actual PMTU can be lower than the Rtn-PMTU value because the Min-PMTU field was not updated by a router on the path that did not process the option.
- * The actual PMTU may be lower than the Rtn-PMTU value because there is a layer-2 device with a lower MTU.
- * The actual PMTU may be larger than the Rtn-PMTU value because of a corrupted, delayed or mis-ordered response. A source host **MUST** ignore a Rtn-PMTU value larger than the MTU configured for the outgoing link.
- * The path might have changed between the time when the probe was sent and when the Rtn-PMTU value received.

IPv6 requires that every link in the Internet have an MTU of 1280 octets or greater. A node **MUST** ignore a Rtn-PMTU value less than 1280 octets [RFC8200].

To avoid unintentional dropping of packets that exceed the actual PMTU (e.g., Scenario 3 in Section 1.1), the source host can delay increasing the PMTU until a probe packet with the size of the Rtn-PMTU value has been successfully acknowledged by the upper layer, confirming that the path supports the larger PMTU. This probing increases robustness, but adds one additional path round trip time before the PMTU is updated. This use resembles that of PTB messages in section 4.6 of DPLPMTUD [RFC8899] (with the important difference that a PTB message can only seek to lower the PMTU, whereas this option could trigger a probe packet to seek to increase the PMTU.)

Section 5.2 of [RFC8201] provides guidance on the caching of PMTU information and also the relation to IPv6 flow labels. Implementations should consider the impact of Equal Cost Multipath (ECMP) [RFC6438]. Specifically, whether a PMTU ought to be maintained for each transport endpoint, or for each network address.

6.3.5. Detecting Path Changes

Path characteristics can change and the actual PMTU could increase or decrease over time. For instance, following a path change when packets are forwarded over a link with a different MTU than that previously used. To bound the delay in discovering an increase in the actual PMTU, a host with a link MTU larger than the current PMTU SHOULD periodically send the MinPMTU HBH Option with the R-bit set. DPLPMTUD provides recommendations concerning how this could be implemented (see Section 5.3 of [RFC8899]). Since the option consumes less capacity than a full-sized probe packet, there can be advantage in using this to detect a change in the path characteristics.

6.3.6. Detection of Dropping Packets that include the Option

There is evidence that some middleboxes drop packets that include Hop-by-Hop options. For example, a firewall might drop a packet that carries an unknown extension header or option. This practice is expected to decrease as an option becomes more widely used. It could result in generation of an ICMPv6 message indicating the problem. This could be used to (temporarily) suspend use of this option.

A middlebox that silently discards a packet with this option results in dropping of any packet using the option. This dropping can be avoided by appropriate configuration in a controlled environment, such as within a data centre, but needs to be considered for Internet usage. Section 6.2 recommends that this option is not used on packets where loss might adversely impact performance.

7. IANA Considerations

IANA has assigned and registered an IPv6 Hop-by-Hop Option type with Temporary status from the "Destination Options and Hop-by-Hop Options" registry [IANA-HBH]. This assignment is shown in Section 5.

IANA is requested to update this registry to point to this document and remove the Temporary status.

8. Security Considerations

This section discusses the security considerations. It first reviews router option processing. It then reviews host processing when receiving this option at the network layer. It then considers two ways in which the Option Data can be processed, followed by two approaches for using the Option Data. Finally, it discusses middlebox implications related to use in the general Internet.

8.1. Router Option Processing

This option shares the characteristics of all other IPv6 Hop-by-Hop Options, in that if not supported at line rate it could be used to degrade the performance of a router. This option, while simple, is no different to other uses of IPv6 Hop-by-Hop options.

It is common for routers to ignore the Hop-by-Hop Option header or drop packets containing a Hop-by-Hop Option header. Routers implementing IPv6 according to [RFC8200] only examine and process the Hop-by-Hop Options header if explicitly configured to do so.

8.2. Network Layer Host Processing

A malicious attacker can forge a packet directed at a host that carries the MinPMTU HBH option. By design, the fields of this IP option can be modified by the network.

For comparison, the ICMPv6 Packet Too Big message used in [RFC8201] Path MTU Discovery, the source host has an inherent trust relationship with the destination host including this option. This trust relationship can be used to help verify the option. ICMPv6 Packet Too Big messages are sent from any router on the path to the destination host, the source host has no prior knowledge of these routers (except for the first hop router).

Reception of this packet will require processing as the network stack parses the packet before the packet is delivered to the upper layer protocol. This network layer option processing is normally completed before any upper layer protocol delivery checks are performed.

The network layer does not normally have sufficient information to validate that the packet carrying an option originated from the destination (or an on-path node). It also does not typically have sufficient context to demultiplex the packet to identify the related transport flow. This can mean that any changes resulting from reception of the option applies to all flows between a pair of endpoints.

These considerations are no different to other uses of Hop-by-Hop options, and this is the use case for PMTUD. The following section describes a mitigation for this attack.

8.3. Validating use of the Option Data

Transport protocols should be designed to provide protection from data injection attacks by off-path devices and mechanisms should be described in the Security Considerations for each transport specification (see Section 5.1 of the UDP Guidelines [RFC8085]). For example, a TCP or UDP application that maintains the related state and uses a randomized ephemeral port would provide basic protection. TLS [RFC8446] or IPsec [RFC4301] provide cryptographic authentication. An upper layer protocol that validates each received packet discards any packet when this validation fails. In this case, the host **MUST** also discard the associated Option Data from the MinPMTU HBH option without further processing (Section 6.3).

A network node on the path has visibility of all packets it forwards. By observing the network packet payload, the node might be able to construct a packet that might be validated by the destination host. Such a node would also be able to drop or limit the flow in other ways that could be potentially more disruptive. Authenticating the packet, for example, using IPsec [RFC4301] or TLS [RFC8446] mitigates this attack. Note that AH style authentication [RFC4302] while authenticating the payload and outer IPv6 header, does not check Hop-by-Hop options that change on route.

8.4. Direct use of the Rtn-PMTU Value

The simplest way to utilize the Rtn-PMTU value is to directly use this to update the PMTU. This approach results in a set of security issues when the option carries malicious data:

- * A direct update of the PMTU using the Rtn-PMTU value could result in an attacker inflating or reducing the size of the host PMTU for the destination. Forcing a reduction in the PMTU can decrease the efficiency of network use, might increase the number of packets/fragments required to send the same volume of payload data, and prevents sending an unfragmented datagram larger than the PMTU. Increasing the PMTU can result in black-holing (see Section 1.1 of [RFC8899]) when the source host sends packets larger than the actual PMTU. This persists until the PMTU is next updated.
- * The method can be used to solicit a response from the destination host. A malicious attacker could forge a packet that causes the destination to add the option to a packet sent to the source host. A forged value of Rtn-PMTU in the Option Data might also impact the remote endpoint, as described in the previous bullet. This persists until a valid MinPMTU HBH option is received. This attack could be mitigated by limiting the sending of the MinPMTU HBH option in reply to incoming packets that carry the option.

8.5. Using the Rtn-PMTU Value as a Hint for Probing

Another way to utilize the Rtn-PMTU value is to indirectly trigger a probe to determine if the path supports a PMTU of size Rtn-PMTU. This approach needs context for the flow, and hence assumes an upper layer protocol that validates the packet that carries the option (see Section 8.3). This is the case when used in combination with DPLPMTUD [RFC8899]. A set of security considerations result when an option carries malicious data:

- * If the forged packet carries a validated option with a non-zero Rtn-PMTU field, the upper layer protocol could utilize the information in the Rtn-PMTU field. A Rtn-PMTU larger than the current PMTU can trigger a probe for a new size.
- * If the forged packet carries a non-zero Min-PMTU field, the upper layer protocol would change the cached information about the path from the source. The cached information at the destination host will be overwritten when the host receives another packet that includes a MinPMTU HBH option corresponding to the flow.
- * Processing of the option could cause a destination host to add the MinPMTU HBH option to a packet sent to the source host. This option will carry a Rtn-PMTU value that could have been updated by the forged packet. The impact of the source host receiving this resembles that discussed previously.

8.6. Impact of Middleboxes

There is evidence that some middleboxes drop packets that include Hop-by-Hop options. For example, a firewall might drop a packet that carries an unknown extension header or option. This practice is expected to decrease as the option becomes more widely used. Methods to address this are discussed in Section 6.3.6.

When a forged packet causes a packet to be sent including the MinPMTU HBH option, and the return path does not forward packets with this option, the packet will be dropped Section 6.3.6. This attack is mitigated by validating the option data before use and by limiting the rate of responses generated. An upper layer could further mitigate the impact by responding to an R-Flag by including the option in a packet that does not carry application data.

9. Experiment Goals

This section describes the experimental goals of this specification.

A successful deployment of the method depends upon several components being implemented and deployed:

- * Support in the sending node (see Section 6.2). This also requires corresponding support in upper layer protocols (see Section 6.3).
- * Router support in nodes (see Section 6.1). The IETF continues to provide recommendations on the use of IPv6 Hop-by-Hop options, for example Section 2.2.2 of [RFC9099]. This document does not update the way router implementations configure support for Hop-by-Hop options.
- * Support in the receiving node (see Section 6.3.3).

Experience from deployment is an expected input to any decision to progress this specification from Experimental to IETF Standards Track. Appropriate inputs might include:

- * Reports of implementation experience;
- * Measurements of the number paths where the method can be used;
- * Measurements showing the benefit realized or the implications of using specific methods over specific paths.

10. Implementation Status

At the time this document was published there are two known implementations of the Path MTU Hop-by-Hop option. These are:

- * Wireshark dissector. This is shipping in production in Wireshark version 3.2 [WIRESHARK].
- * A prototype in the open source version of the FD.io Vector Packet Processing (VPP) technology [VPP]. At the time this document was published, the source code can be found [VPP_SRC].

11. Acknowledgments

Helpful comments were received from Tom Herbert, Tom Jones, Fred Templin, Ole Troan, Tianran Zhou, Jen Linkova, Brian Carpenter, Peng Shuping, Mark Smith, Fernando Gont, Michael Dougherty, Erik Kline, and other members of the 6MAN working group.

12. Change log [RFC Editor: Please remove]

draft-ietf-6man-mtu-option-15, 2022-May-10

- * Correcting an editing mistake in Appendix A.
- * Editorial Change.

draft-ietf-6man-mtu-option-14, 2022-April-15

- * Area Director Reviews:
 - Lars Eggert's Review: Fixed "nits".
 - Eric Vyncke's Review: Added that this work is focused on Unicast, removed Discussion from Section 6.1, revised text on PLPMTUD probing, changed SHOULD to MUST in Section 6.3.4, and fixed several NITs.
 - Alvaro Retana's Review: Changed SHOULD language to more general text in Section 6.1
 - ARTART Review: Added new Appendix "Examples of Usage" with diagrams showing examples of use.
 - Zaheduzzaman Sarker's Review: Fixed some editorial issues, and updated SHOULD language.
- * Editorial Changes.

draft-ietf-6man-mtu-option-13, 2022-February-28

- * Area Directorate Reviews:
 - SECDIR Review: Fixed "nit".
 - TSVART Review: Restructured Section 6 including making Transport Behavior more prominent, added text about ICMPv6 to Section 6.3.1, moved the text about prior work in RFC1063 to Section 2.
 - GENART Review: Added text to Section 1 that this option was designed to work with packet sizes that can be specified in the IPv6 Header.
- * Editorial Changes.

draft-ietf-6man-mtu-option-12, 2022-January-26

- * Clarified a few issues raised by AD review by Erik Kline AD review.

draft-ietf-6man-mtu-option-11, 2021-September-30

- * Clarifications and editorial changes to the Security Considerations section based on early AD review by Erik Kline.

draft-ietf-6man-mtu-option-10, 2021-September-27

- * Clarifications and editorial changes based on second chair review by Ole Troan.
- * Editorial changes.

draft-ietf-6man-mtu-option-09, 2021-September-23

- * Clarifications and editorial changes based on review by Michael Dougherty.

draft-ietf-6man-mtu-option-08, 2021-September-7

- * Clarifications and editorial changes based on chair review by Ole Troan.
- * Correction and clarifications based on review by Fernando Gont.

draft-ietf-6man-mtu-option-07, 2021-August-31

- * Added Experiment Goals section.
- * Added Implementation Status section.
- * Updated the IANA Considerations section to point to this document and remove Temporary status.
- * Clarifications and editorial changes based on review by Mark Smith.

draft-ietf-6man-mtu-option-06, 2021-August-7

- * Transport usage of the mechanism clarified in response to feedback and suggestions from Jen Linkova.
- * Restructured Section 6 to improve readability.
- * Editorial changes.

draft-ietf-6man-mtu-option-05, 2021-April-28

- * Editorial changes.

draft-ietf-6man-mtu-option-04, 2020-Oct-23

- * Fixes for typos.

draft-ietf-6man-mtu-option-03, 2020-Sept-14

- * Rewrite to make text and terminology more consistent.
- * Added the notion of validating the packet before use of the HBH option data.
- * Method aligned with the way common APIs send/receive HBH option data.
- * Added reference to DPLPMTUD and clarified upper layer usage.
- * Completed security considerations section.

draft-ietf-6man-mtu-option-02, 2020-March-9

- * Editorial changes to make text and terminology more consistent.

- * Added reference to DPLPMTUD.

draft-ietf-6man-mtu-option-01, 2019-September-13

- * Changes to show IANA assigned code point.
- * Editorial changes to make text and terminology more consistent.
- * Added a reference to RFC8200 in Section 2 and a reference to RFC6438 in Section 6.3.

draft-ietf-6man-mtu-option-00, 2019-August-9

- * First 6man w.g. draft version.
- * Changes to request IANA allocation of code point.
- * Editorial changes.

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

- * Changed option format to also include the Returned PMTU value and Return flag and made related text changes in Section 6.2 to describe this behavior.
- * ICMPv6 Packet Too Big messages are no longer used for feedback to the source host.
- * Added to Acknowledgements Section that a similar mechanism was proposed for IPv4 in 1988 in [RFC1063].
- * Editorial changes.

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

- * 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.
- * Added Section 2 "Motivation and Problem Solved" section to better describe what the purpose of this document is.
- * Added appendix describing planned experiments and how the results will be measured.
- * Editorial changes.

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

- * Initial draft.

13. References

13.1. 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>>.

13.2. Informative References

- [I-D.ietf-taps-arch]
Pauly, T., Trammell, B., Brunstrom, A., Fairhurst, G., and C. Perkins, "An Architecture for Transport Services", Work in Progress, Internet-Draft, draft-ietf-taps-arch-12, 3 January 2022, <<https://datatracker.ietf.org/doc/html/draft-ietf-taps-arch-12>>.
- [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>>.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <<https://www.rfc-editor.org/info/rfc4301>>.
- [RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <<https://www.rfc-editor.org/info/rfc4302>>.

- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [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>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [RFC8899] Fairhurst, G., Jones, T., Tüxen, M., Rüngeler, I., and T. Völker, "Packetization Layer Path MTU Discovery for Datagram Transports", RFC 8899, DOI 10.17487/RFC8899, September 2020, <<https://www.rfc-editor.org/info/rfc8899>>.
- [RFC8900] Bonica, R., Baker, F., Huston, G., Hinden, R., Troan, O., and F. Gont, "IP Fragmentation Considered Fragile", BCP 230, RFC 8900, DOI 10.17487/RFC8900, September 2020, <<https://www.rfc-editor.org/info/rfc8900>>.
- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/info/rfc9000>>.

- [RFC9099] Vyncke, É., Chittimaneni, K., Kaeo, M., and E. Rey,
"Operational Security Considerations for IPv6 Networks",
RFC 9099, DOI 10.17487/RFC9099, August 2021,
<<https://www.rfc-editor.org/info/rfc9099>>.
- [VPP] "VPP/What is VPP?",
<https://wiki.fd.io/view/VPP/What_is_VPP%3F>.
- [VPP_SRC] "VPP Source", <<https://gerriet.fd.io/r/c/vpp/+/21948>>.
- [WIRESHARK] "Wireshark Network Protocol Analyzer",
<<https://www.wireshark.org>>.

Appendix A. Examples of Usage

This section provides examples that illustrate a use of the MinPMTU HBH option by a source using DPLPMTUD to discover the PLPMTU supported by a path. They consider a path where the on-path router has been configured with an outgoing MTU of d' . The source starts by transmission of packets of size a , and then uses DPLPMTUD to seek to increase the size in steps resulting in sizes of b, c, d, e , etc., (chosen by the search algorithm used by DPLPMTUD). The search algorithm terminates with a PLPMTU that is at least d and is less than or equal to d' .

The first example considers DPLPMTUD without using the MinPMTU HBH option. In this case, DPLPMTUD searches using an increasing size of probe packet. Probe packets of size (e) are sent, which are larger than the actual PMTU. In this example, PTB messages are not received from the routers and repeated unsuccessful probes result in the search phase completing. Packets of data are never sent with a size larger than the size of the last confirmed probe packet. ACKs of data packets are not shown.

```

----Packets of data size (a) ----->
----Probe size (b) ----->
<----- ACK of probe -----
----Packets of data size (b) ----->
----Probe size (c) ----->
<----- ACK of probe -----
----Packets of data size (c) ----->
----Probe size (d) ----->
<----- ACK of probe -----
----Packets of data size (d) ----->
<----- ACK of probe -----
...
----Probe size (e) -----X
      X----ICMPv6 PTB (d') --|
----Packets of data size (d) ----->
----Probe size (e) -----X (again)
      X----ICMPv6 PTB (d') --|
----Packets of data size (d) -----
...
etc, until MaxProbes are unsuccessful and search phase completes.
----Packets of data size (d) ----->

```

Figure 4

The second example considers DPLPMTUD with the MinPMTU HBH option set on a connectivity probe packet.

The IPv6 option is sent end-to-end, and the Min-PMTU is updated by a router on the path to d', which is returned in a response that also sets the MinPMTU HBH option. Upon receiving Rtn-PMTU value is received, DPLPMTUD immediately sends a probe packet of the target size (d'). If the probe packet is confirmed for the path, the PLPMTU is updated, allowing the source to use data packets up to size d'. (The search algorithm is allowed to continue to probe to see if the path supports a larger size.) Packets of data are never sent with a size larger than the last confirmed probe size, d'.

```

----Packets of data size (a) ----->
----Connectivity probe with MinPMTU-
      +--updated to minPMTU=d'----->
<-----ACK with Rtn-PMTU=d'-----
----Packets of data size (a) ----->
----Probe size (d') ----->
<----- ACK of probe -----
----Packets of data size (d') ----->
Search phase completes.
----Packets of data size (d') ----->

```

Figure 5

The final example considers DPLPMTUD with the MinPMTU HBH option set on a connectivity probe packet, but shows the effect when this connectivity probe packet is dropped.

In this case, the packet with the MinPMTU HBH option is not received. DPLPMTUD searches using probe packets of increasing size, increasing the PLPMTU when the probes are confirmed. An ICMPv6 PTB message is received when the probed size exceeds the actual PMTU, indicating a PTB_SIZE of d'. DPLPMTUD immediately sends a probe packet of the target size (d'). If the probe packet is confirmed for the path, the PLPMTU is updated, allowing the source to use data packets up to size d'. If the ICMPv6 PTB message is not received, the DPLPMTU will be the last confirmed probe size, d.

```

----Packets of data size (a) ----->
----Connectivity probe with MinPMTU -----X
----Packets of data size (a) ----->
----Probe size (b) ----->
<----- ACK of probe -----
----Packets of data size (b) ----->
----Probe size (c) ----->
<----- ACK of probe -----
----Packets of data size (c) ----->
----Probe size (d) ----->
<----- ACK of probe -----
----Packets of data size (d) ----->
----Probe size (e) -----X
<--ICMPv6 PTB PTB_SIZE(d') -|
----Packets of data size (d) ----->
----Probe size (d') using target set by PTB_SIZE ----->
<----- ACK of probe -----
Search phase completes.
----Packets of data size (d') ----->

```

Figure 6

The number of probe rounds depends on the number of steps needed by the search algorithm, and is typically larger for a larger PMTU.

Authors' Addresses

Robert M. Hinden
 Check Point Software
 959 Skyway Road
 San Carlos, CA 94070
 United States of America

Email: bob.hinden@gmail.com

Godred Fairhurst
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen
AB24 3UE
United Kingdom
Email: gorrry@erg.abdn.ac.uk

6man
Internet-Draft
Intended status: Standards Track
Expires: July 27, 2022

Z. Ali
C. Filsfils
Cisco Systems
S. Matsushima
Softbank
D. Voyer
Bell Canada
M. Chen
Huawei
January 23, 2022

Operations, Administration, and Maintenance (OAM) in Segment Routing
Networks with IPv6 Data plane (SRv6)
draft-ietf-6man-spring-srv6-oam-13

Abstract

This document describes how the existing IPv6 mechanisms for ping and traceroute can be used in an SRv6 network. The document also specifies the OAM flag in the Segment Routing Header (SRH) for performing controllable and predictable flow sampling from segment endpoints. In addition, the document describes how a centralized monitoring system performs a path continuity check between any nodes within an SRv6 domain.

Status of This Memo

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

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

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

This Internet-Draft will expire on July 27, 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
1.2. Abbreviations	3
1.3. Terminology and Reference Topology	4
2. OAM Mechanisms	5
2.1. O-flag in Segment Routing Header	5
2.1.1. O-flag Processing	6
2.2. OAM Operations	8
3. Implementation Status	8
4. Security Considerations	9
5. Privacy Considerations	9
6. IANA Considerations	9
7. References	10
7.1. Normative References	10
7.2. Informative References	10
Appendix A. Illustrations	12
A.1. Ping in SRv6 Networks	12
A.1.1. Pinging an IPv6 Address via a Segment-list	13
A.1.2. Pinging a SID	14
A.2. Traceroute	15
A.2.1. Traceroute to an IPv6 Address via a Segment-list	15
A.2.2. Traceroute to a SID	17
A.3. A Hybrid OAM Using O-flag	18
A.4. Monitoring of SRv6 Paths	21
Appendix B. Acknowledgements	22
Appendix C. Contributors	22
Authors' Addresses	23

1. Introduction

As Segment Routing with IPv6 data plane (SRv6) [RFC8402] simply adds a new type of Routing Extension Header, existing IPv6 OAM mechanisms can be used in an SRv6 network. This document describes how the existing IPv6 mechanisms for ping and traceroute can be used in an SRv6 network. This includes illustrations of pinging an SRv6 SID to verify that the SID is reachable and is locally programmed at the

target node. This also includes illustrations for tracerouting to an SRv6 SID for hop-by-hop fault localization as well as path tracing to a SID.

The document also introduces enhancements for the OAM mechanism for SRv6 networks for performing controllable and predictable flow sampling from segment endpoints using, e.g., IP Flow Information Export (IPFIX) protocol [RFC7011]. Specifically, the document specifies the O-flag in SRH as a marking-bit in the user packets to trigger the telemetry data collection and export at the segment endpoints.

The document also outlines how the centralized OAM technique in [RFC8403] can be extended for SRv6 to perform a path continuity check between any nodes within an SRv6 domain. Specifically, the document illustrates how a centralized monitoring system can monitor arbitrary SRv6 paths by creating the loopback probes that originate and terminate at the centralized monitoring system.

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. Abbreviations

The following abbreviations are used in this document:

SID: Segment ID.

SL: Segments Left.

SR: Segment Routing.

SRH: Segment Routing Header [RFC8754].

SRv6: Segment Routing with IPv6 Data plane.

PSP: Penultimate Segment Pop of the SRH [RFC8986].

USP: Ultimate Segment Pop of the SRH [RFC8986].

ICMPv6: ICMPv6 Specification [RFC4443].

IS-IS: Intermediate System to Intermediate System

OSPF: Open Shortest Path First protocol [RFC2328]

IGP: Interior Gateway Protocols (e.g., OSPF, IS-IS).

BGP-LS: Border Gateway Protocol - Link State Extensions [RFC8571]

1.3. Terminology and Reference Topology

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

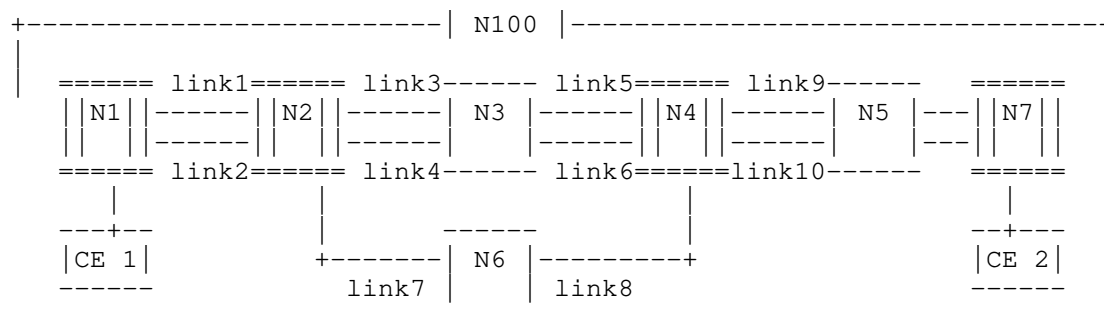


Figure 1 Reference Topology

In the reference topology:

Node j has a IPv6 loopback address 2001:db8:L:j::/128.

Nodes N1, N2, N4 and N7 are SRv6-capable nodes.

Nodes N3, N5 and N6 are IPv6 nodes that are not SRv6-capable. Such nodes are referred as non-SRv6 capable nodes.

CE1 and CE2 are Customer Edge devices of any data plane capability (e.g., IPv4, IPv6, L2, etc.).

A SID at node j with locator block 2001:db8:K::/48 and function U is represented by 2001:db8:K:j:U::.

Node N100 is a controller.

The IPv6 address of the nth Link between node i and j at the i side is represented as 2001:db8:i:j:in::, 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 N3 is 2001:db8:3:4:31::.

2001:db8:K:j:Xin:: is explicitly allocated as the End.X SID at node j towards neighbor node i via nth Link between node i and node j. e.g., 2001:db8:K:2:X31:: represents End.X at N2 towards N3 via link3 (the 1st link between N2 and N3). Similarly, 2001:db8:K:4:X52:: represents the End.X at N4 towards N5 via link10 (the 2nd link between N4 and N5). Please refer to [RFC8986] for description of End.X SID.

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

(SA,DA) (S3, S2, S1; SL) (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.

2. OAM Mechanisms

This section defines OAM enhancement for the SRv6 networks.

2.1. O-flag in Segment Routing Header

[RFC8754] describes the Segment Routing Header (SRH) and how SR capable nodes use it. The SRH contains an 8-bit "Flags" field.

This document defines the following bit in the SRH Flags field to carry the O-flag:

```

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

```

Where:

O-flag: OAM flag in the SRH Flags field defined in [RFC8754].

2.1.1. O-flag Processing

The O-flag in SRH is used as a marking-bit in the user packets to trigger the telemetry data collection and export at the segment endpoints.

An SR domain ingress edge node encapsulates packets traversing the SR domain as defined in [RFC8754]. The SR domain ingress edge node MAY use the O-flag in SRH for marking the packet to trigger the telemetry data collection and export at the segment endpoints. Based on a local configuration, the SR domain ingress edge node may implement a classification and sampling mechanism to mark a packet with the O-flag in SRH. Specification of the classification and sampling method is outside the scope of this document.

This document does not specify the data elements that need to be exported and the associated configurations. Similarly, this document does not define any formats for exporting the data elements. Nonetheless, without the loss of generality, this document assumes IP Flow Information Export (IPFIX) protocol [RFC7011] is used for exporting the traffic flow information from the network devices to a controller for monitoring and analytics. Similarly, without the loss of generality, this document assumes requested information elements are configured by the management plane through data set templates (e.g., as in IPFIX [RFC7012]).

Implementation of the O-flag is OPTIONAL. 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 control plane protocol(s).

When N receives a packet destined to S and S is a local SID, the line S01 of the pseudo-code associated with the SID S, as defined in section 4.3.1.1 of [RFC8754], is appended to as follows for the O-flag processing.

```
S01.1. IF O-flag is set and local configuration permits
      O-flag processing {
        a. Make a copy of the packet.
        b. Send the copied packet, along with a timestamp
           to the OAM process for telemetry data collection
           and export.      ;; Refl
      }
```

Refl: To provide an accurate timestamp, an implementation should copy and record the timestamp as soon as possible during packet processing. Timestamp and any other metadata is not carried in the packet forwarded to the next hop.

Please note that the O-flag processing happens before execution of regular processing of the local SID S. Specifically, the line S01.1 of the pseudo-code specified in this document is inserted between line S01 and S02 of the pseudo-code defined in section 4.3.1.1 of [RFC8754].

Based on the requested information elements configured by the management plane through data set templates [RFC7012], the OAM process exports the requested information elements. The information elements include parts of the packet header and/or parts of the packet payload for flow identification. The OAM process uses information elements defined in IPFIX [RFC7011] and PSAMP [RFC5476] for exporting the requested sections of the mirrored packets.

If the penultimate segment of a segment-list is a Penultimate Segment Pop (PSP) SID, telemetry data from the ultimate segment cannot be requested. This is because, when the penultimate segment is a PSP SID, the SRH is removed at the penultimate segment and the O-flag is not processed at the ultimate segment.

The processing node MUST rate-limit the number of packets punted to the OAM process to a configurable rate. This is to avoid hitting any performance impact on the OAM and the telemetry collection processes. Failure in implementing the rate limit can lead to a denial-of-service attack, as detailed in section 4.

The OAM process MUST NOT process the copy of the packet or respond to any upper-layer header (like ICMP, UDP, etc.) payload to prevent multiple evaluations of the datagram.

The OAM process is expected to be located on the routing node processing the packet. Although the specification of the OAM process or the external controller operations are beyond the scope of this document, the OAM process SHOULD NOT be topologically distant from the routing node, as this is likely to create significant security and congestion issues. How to correlate the data collected from

different nodes at an external controller is also outside the scope of the document. Appendix A illustrates use of the O-flag for implementing a hybrid OAM mechanism, where the "hybrid" classification is based on RFC7799 [RFC7799].

2.2. OAM Operations

IPv6 OAM operations can be performed for any SRv6 SID whose behavior allows Upper Layer Header processing for an applicable OAM payload (e.g., ICMP, UDP).

Ping to an SRv6 SID is used to verify that the SID is reachable and is locally programmed at the target node. Traceroute to a SID is used for hop-by-hop fault localization as well as path tracing to a SID. Appendix A illustrates the ICMPv6 based ping and the UDP based traceroute mechanisms for ping and traceroute to an SRv6 SID. Although this document only illustrates ICMPv6 ping and UDP based traceroute to an SRv6 SID, the procedures are equally applicable to other IPv6 OAM probing to an SRv6 SID (e.g., Bidirectional Forwarding Detection (BFD) [RFC5880], Seamless BFD (SBFD) [RFC7880], STAMP probe message processing [I-D.gandhi-spring-stamp-srpm], etc.). Specifically, as long as local configuration allows the Upper-layer Header processing of the applicable OAM payload for SRv6 SIDs, the existing IPv6 OAM techniques can be used to target a probe to a (remote) SID.

IPv6 OAM operations can be performed with the target SID in the IPv6 destination address without SRH or with SRH where the target SID is the last segment. In general, OAM operations to a target SID may not exercise all of its processing depending on its behavior definition. For example, ping to an End.X SID [RFC8986] only validates the SID is locally programmed at the target node and does not validate switching to the correct outgoing interface. To exercise the behavior of a target SID, the OAM operation should construct the probe in a manner similar to a data packet that exercises the SID behavior, i.e. to include that SID as a transit SID in either an SRH or IPv6 DA of an outer IPv6 header or as appropriate based on the definition of the SID behavior.

3. 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.

4. Security Considerations

[RFC8754] defines the notion of an SR domain and use of SRH within the SR domain. The use of OAM procedures described in this document is restricted to an SR domain. For example, similar to the SID manipulation, O-flag manipulation is not considered as a threat within the SR domain. Procedures for securing an SR domain are defined the section 5.1 and section 7 of [RFC8754].

As noted in section 7.1 of [RFC8754], compromised nodes within the SR domain may mount attacks. The O-flag may be set by an attacking node attempting a denial-of-service attack on the OAM process at the segment endpoint node. An implementation correctly implementing the rate limiting in section 2.1.1 is not susceptible to that denial-of-service attack. Additionally, SRH Flags are protected by the HMAC TLV, as described in section 2.1.2.1 of [RFC8754]. Once an HMAC is generated for a segment list with the O-flag set, it can be used for an arbitrary amount of traffic using that segment list with O-flag set.

The security properties of the channel used to send exported packets marked by the O-flag will depend on the specific OAM processes used. An on-path attacker able to observe this OAM channel could conduct traffic analysis, or potentially eavesdropping (depending on the OAM configuration), of this telemetry for the entire SR domain from such a vantage point.

This document does not impose any additional security challenges to be considered beyond security threats described in [RFC4884], [RFC4443], [RFC0792], [RFC8754] and [RFC8986].

5. Privacy Considerations

The per-packet marking capabilities of the O-flag provides a granular mechanism to collect telemetry. When this collection is deployed by an operator with knowledge and consent of the users, it will enable a variety of diagnostics and monitoring to support the OAM and security operations use cases needed for resilient network operations. However, this collection mechanism will also provide an explicit protocol mechanism to operators for surveillance and pervasive monitoring use cases done contrary to the user's consent.

6. IANA Considerations

This document requests that IANA allocate the following registration in the "Segment Routing Header Flags" sub-registry for the "Internet Protocol Version 6 (IPv6) Parameters" registry maintained by IANA:

Bit	Description	Reference
2	O-flag	This document

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>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8754] Filssils, 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] Filssils, 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-stamp-srpm] Gandhi, R., Filssils, C., Voyer, D., Chen, M., Janssens, B., and R. Foote, "Performance Measurement Using Simple TWAMP (STAMP) for Segment Routing Networks", draft-gandhi-spring-stamp-srpm-07 (work in progress), July 2021.
- [I-D.ietf-ippm-ioam-data] Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-11 (work in progress), November 2020.

- [I-D.matsushima-spring-srv6-deployment-status]
Matsushima, S., Filsfils, C., Ali, Z., Li, Z., and K. Rajaraman, "SRv6 Implementation and Deployment Status", draft-matsushima-spring-srv6-deployment-status-11 (work in progress), February 2021.
- [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <<https://www.rfc-editor.org/info/rfc792>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [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>>.
- [RFC5476] Claise, B., Ed., Johnson, A., and J. Quittek, "Packet Sampling (PSAMP) Protocol Specifications", RFC 5476, DOI 10.17487/RFC5476, March 2009, <<https://www.rfc-editor.org/info/rfc5476>>.
- [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>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.
- [RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <<https://www.rfc-editor.org/info/rfc7011>>.

- [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>>.
- [RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <<https://www.rfc-editor.org/info/rfc7799>>.
- [RFC7880] Pignataro, C., Ward, D., Akiya, N., Bhatia, M., and S. Pallagatti, "Seamless Bidirectional Forwarding Detection (S-BFD)", RFC 7880, DOI 10.17487/RFC7880, July 2016, <<https://www.rfc-editor.org/info/rfc7880>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.
- [RFC8403] Geib, R., Ed., Filsfils, C., Pignataro, C., Ed., and N. Kumar, "A Scalable and Topology-Aware MPLS Data-Plane Monitoring System", RFC 8403, DOI 10.17487/RFC8403, July 2018, <<https://www.rfc-editor.org/info/rfc8403>>.
- [RFC8571] Ginsberg, L., Ed., Previdi, S., Wu, Q., Tantsura, J., and C. Filsfils, "BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions", RFC 8571, DOI 10.17487/RFC8571, March 2019, <<https://www.rfc-editor.org/info/rfc8571>>.

Appendix A. Illustrations

This appendix shows how some of the existing IPv6 OAM mechanisms can be used in an SRv6 network. It also illustrates an OAM mechanism for performing controllable and predictable flow sampling from segment endpoints. How centralized OAM technique in [RFC8403] can be extended for SRv6 is also described in this appendix.

A.1. Ping in SRv6 Networks

The existing mechanism to perform the reachability checks, along the shortest path, continues to work without any modification. Any IPv6 node (SRv6 capable or a non-SRv6 capable) can initiate, transit, and egress a ping packet.

The following subsections outline some additional use cases of the ICMPv6 ping in the SRv6 networks.

A.1.1.1. Pinging an IPv6 Address via a Segment-list

If an SRv6-capable ingress node wants to ping an IPv6 address via an arbitrary segment list <S1, S2, S3>, it needs to initiate an ICMPv6 ping with an SR header containing the SID list <S1, S2, S3>. This is illustrated using the topology in Figure 1. User issues a ping from node N1 to a loopback of node N5, via segment list <2001:db8:K:2:X31::, 2001:db8:K:4:X52::>. The SID behavior used in the example is End.X SID, as described in [RFC8986], but the procedure is equally applicable to any other (transit) SID type.

Figure 2 contains sample output for a ping request initiated at node N1 to a loopback address of node N5 via a segment list <2001:db8:K:2:X31::, 2001:db8:K:4:X52::>.

```
> ping 2001:db8:L:5:: via segment-list 2001:db8:K:2:X31::,
    2001:db8:K:4:X52::

Sending 5, 100-byte ICMPv6 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 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 (2001:db8:L:1::, 2001:db8:K:2:X31::) (2001:db8:L:5::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::, 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 behavior indicated by the 2001:db8:K:2:X31:: SID and forwards the packet on link3 to N3.
- o Node N3, which is a non-SRv6 capable node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on the DA 2001:db8:K:4:X52:: in the IPv6 header.
- o Node N4, which is an SRv6-capable node, performs the standard SRH processing. Specifically, it observes the End.X behavior

(2001:db8:K:4:X52::) and forwards the packet on link10 towards N5. If 2001:db8:K:4:X52:: is a PSP SID, the penultimate node (Node N4) does not, should not and cannot differentiate between the data packets and OAM probes. Specifically, if 2001:db8:K:4:X52:: is a PSP SID, node N4 executes the SID like any other data packet with DA = 2001:db8:K:4:X52:: and removes the SRH.

- o The echo request packet at N5 arrives as an IPv6 packet with or without an SRH. If N5 receives the packet with SRH, it skips SRH processing (SL=0). In either case, Node N5 performs the standard ICMPv6 processing on the echo request and responds with the echo reply message to N1. The echo reply message is IP routed.

A.1.2. Pinging a SID

The ping mechanism described above applies equally to perform SID reachability check and to validate the SID is locally programmed at the target node. This is explained using an example in the following. The example uses ping to an END SID, as described in [RFC8986], but the procedure is equally applicable to ping any other SID behaviors.

Consider the example where the user wants to ping a remote SID 2001:db8:K:4::, via 2001:db8:K:2:X31::, from node N1. 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 (2001:db8:L:1::, 2001:db8:K:2:X31::) (2001:db8:K:4::, 2001:db8:K:2:X31::; SL=1; 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 behavior indicated by the 2001:db8:K:2:X31:: SID on the echo request packet. If 2001:db8:K:2:X31:: is a PSP SID, node N4 executes the SID like any other data packet with DA = 2001:db8:K:2:X31:: and removes the SRH.
- o Node N3, which is a non-SRv6 capable node, performs the standard IPv6 processing. Specifically, it forwards the echo request based on DA = 2001:db8:K:4:: in the IPv6 header.
- o When node N4 receives the packet, it processes the target SID (2001:db8:K:4::).
- o If the target SID (2001:db8:K:4::) is not locally instantiated and does not represent a local interface, the packet is discarded

- o If the target SID (2001:db8:K:4::) is locally instantiated or represents a local interface, the node processes the upper layer header. As part of the upper layer header processing node N4 respond to the ICMPv6 echo request message and responds with the echo reply message. The echo reply message is IP routed.

A.2. Traceroute

The existing traceroute mechanisms, along the shortest path, continues to work without any modification. Any IPv6 node (SRv6 capable or a non-SRv6 capable) can initiate, transit, and egress a traceroute probe.

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

A.2.1. Traceroute to an IPv6 Address via a Segment-list

If an SRv6-capable ingress node wants to traceroute to IPv6 address via an arbitrary segment list <S1, S2, S3>, it needs to initiate a traceroute probe with an SR header containing the SID list <S1, S2, S3>. User issues a traceroute from node N1 to a loopback of node N5, via segment list <2001:db8:K:2:X31::, 2001:db8:K:4:X52::>. The SID behavior used in the example is End.X SID, as described in [RFC8986], but the procedure is equally applicable to any other (transit) SID type. Figure 3 contains sample output for the traceroute request.

```
> traceroute 2001:db8:L:5:: via segment-list 2001:db8:K:2:X31::,
      2001:db8:K:4:X52::
```

Tracing the route to 2001:db8:L:5::

```
1  2001:db8:2:1:21:: 0.512 msec 0.425 msec 0.374 msec
   DA: 2001:db8:K:2:X31::,
   SRH: (2001:db8:L:5::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::, SL=2)
2  2001:db8:3:2:31:: 0.721 msec 0.810 msec 0.795 msec
   DA: 2001:db8:K:4:X52::,
   SRH: (2001:db8:L:5::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::, SL=1)
3  2001:db8:4:3::41:: 0.921 msec 0.816 msec 0.759 msec
   DA: 2001:db8:K:4:X52::,
   SRH: (2001:db8:L:5::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::, SL=1)
4  2001:db8:5:4::52:: 0.879 msec 0.916 msec 1.024 msec
   DA: 2001:db8:L:5::
```

Figure 3 A sample traceroute output at an SRv6-capable node

In the sample traceroute output, the information displayed at each hop is obtained using the contents of the "Time Exceeded" or "Destination Unreachable" ICMPv6 responses. These ICMPv6 responses are IP routed.

In the sample traceroute output, the information for link3 is returned by N3, which is a non-SRv6 capable node. Nonetheless, the ingress node is able to display SR header contents as the packet travels through the non-SRv6 capable 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 included in these ICMPv6 messages initiated by the non-SRv6 capable 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 the first hop is returned by N2, which is an SRv6-capable node. Just like for the second hop, the ingress node is able to display SR header contents for the first hop.

There is no difference in processing of the traceroute probe at an SRv6-capable and a non-SRv6 capable node. Similarly, both SRv6-capable and non-SRv6 capable nodes may use the address of the interface on which probe was received as the source address in the ICMPv6 response. ICMPv6 extensions defined in [RFC5837] can be used to 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 IP address of the interface on which the traceroute probe was received is useful. This information can also be used to verify if SIDs 2001:db8:K:2:X31:: and 2001:db8:K:4:X52:: are executed correctly by N2 and N4, respectively. Specifically, the information displayed for the second hop contains the incoming interface address 2001:db8:2:3:31:: at N3. This matches with the expected interface bound to End.X behavior 2001:db8:K:2:X31:: (link3). Similarly, the information displayed for the fourth hop contains the incoming interface address 2001:db8:4:5::52:: at N5. This matches with the expected interface bound to the End.X behavior 2001:db8:K:4:X52:: (link10).

A.2.2. Traceroute to a SID

The mechanism to traceroute an IPv6 Address via a Segment-list described in the previous section applies equally to traceroute a remote SID behavior, as explained using an example in the following. The example uses traceroute to an END SID, as described in [RFC8986], but the procedure is equally applicable to tracerouting any other SID behaviors.

Please note that traceroute to a SID is exemplified using UDP probes. However, the procedure is equally applicable to other implementations of traceroute mechanism. The UDP encoded message to traceroute a SID would use the UDP ports assigned by IANA for "traceroute use".

Consider the example where the user wants to traceroute a remote SID 2001:db8:K:4::, via 2001:db8:K:2:X31::, from node N1. The traceroute probe is processed at the individual nodes along the path as follows:

- o Node N1 initiates a traceroute probe packet as follows (2001:db8:L:1::, 2001:db8:K:2:X31::) (2001:db8:K:4::, 2001:db8:K:2:X31::; SL=1; NH=UDP) (Traceroute probe). The first traceroute probe is sent with hop-count value set to 1. The hop-count value is incremented by 1 for each following traceroute probes.
- o When node N2 receives the packet with hop-count = 1, it processes the hop-count expiry. Specifically, the node N2 responds with the ICMPv6 message (Type: "Time Exceeded", Code: "Hop limit exceeded in transit"). The ICMPv6 response is IP routed.
- o When Node N2 receives the packet with hop-count > 1, it performs the standard SRH processing. Specifically, it executes the End.X behavior indicated by the 2001:db8:K:2:X31:: SID on the traceroute probe. If 2001:db8:K:2:X31:: is a PSP SID, node N2 executes the SID like any other data packet with DA = 2001:db8:K:2:X31:: and removes the SRH.
- o When node N3, which is a non-SRv6 capable node, receives the packet with hop-count = 1, it processes the hop-count expiry. Specifically, the node N3 responds with the ICMPv6 message (Type: "Time Exceeded", Code: "Hop limit exceeded in Transit"). The ICMPv6 response is IP routed.
- o When node N3, which is a non-SRv6 capable node, receives the packet with hop-count > 1, it performs the standard IPv6 processing. Specifically, it forwards the traceroute probe based on DA 2001:db8:K:4:: in the IPv6 header.

- o When node N4 receives the packet with DA set to the local SID 2001:db8:K:4::, it processes the END SID.
- o If the target SID (2001:db8:K:4::) is not locally instantiated and does not represent a local interface, the packet is discarded.
- o If the target SID (2001:db8:K:4::) is locally instantiated or represents a local interface, the node processes the upper layer header. As part of the upper layer header processing node N4 responds with the ICMPv6 message (Type: Destination unreachable, Code: Port Unreachable). The ICMPv6 response is IP routed.

Figure 4 displays a sample traceroute output for this example.

```
> traceroute 2001:db8:K:4:X52:: via segment-list 2001:db8:K:2:X31::

Tracing the route to SID 2001:db8:K:4:X52::
 1  2001:db8:2:1:21:: 0.512 msec 0.425 msec 0.374 msec
    DA: 2001:db8:K:2:X31::,
    SRH:(2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=1)
 2  2001:db8:3:2:21:: 0.721 msec 0.810 msec 0.795 msec
    DA: 2001:db8:K:4:X52::,
    SRH:(2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=0)
 3  2001:db8:4:3:41:: 0.921 msec 0.816 msec 0.759 msec
    DA: 2001:db8:K:4:X52::,
    SRH:(2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=0)
```

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

A.3. A Hybrid OAM Using O-flag

This section illustrates a hybrid OAM mechanism using the the O-flag. Without loss of the generality, the illustration assumes N100 is a centralized controller.

The illustration is different than the In-situ OAM defined in [I.D-draft-ietf-ippm-ioam-data]. This is because In-situ OAM records operational and telemetry information in the packet as the packet traverses a path between two points in the network [I.D-draft-ietf-ippm-ioam-data]. The illustration in this subsection does not require the recording of OAM data in the packet.

The illustration does not assume any formats for exporting the data elements or the data elements that need to be exported. The

illustration assumes system clocks among all nodes in the SR domain are synchronized.

Consider the example where the user wants to monitor sampled IPv4 VPN 999 traffic going from CE1 to CE2 via a low latency SR policy P installed at Node N1. To exercise a low latency path, the SR Policy P forces the packet via segments 2001:db8:K:2:X31:: and 2001:db8:K:4:X52::. The VPN SID at N7 associated with VPN 999 is 2001:db8:K:7:DT999::. 2001:db8:K:7:DT999:: is a USP SID. N1, N4, and N7 are capable of processing O-flag but N2 is not capable of processing O-flag. N100 is the centralized controller capable of processing and correlating the copy of the packets sent from nodes N1, N4, and N7. N100 is aware of O-flag processing capabilities. Controller N100 with the help from nodes N1, N4, N7 and implements a hybrid OAM mechanism using the O-flag as follows:

- o A packet P1:(IPv4 header)(payload) is sent from CE1 to Node N1.
- o Node N1 steers the packet P1 through the Policy P. Based on a local configuration, Node N1 also implements logic to sample traffic steered through policy P for hybrid OAM purposes. Specification for the sampling logic is beyond the scope of this document. Consider the case where packet P1 is classified as a packet to be monitored via the hybrid OAM. Node N1 sets O-flag during the encapsulation required by policy P. As part of setting the O-flag, node N1 also sends a timestamped copy of the packet P1: (2001:db8:L:1::, 2001:db8:K:2:X31::) (2001:db8:K:7:DT999::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=2; O-flag=1; NH=IPv4) (IPv4 header)(payload) to a local OAM process. The local OAM process sends a full or partial copy of the packet P1 to the controller N100. The OAM process includes the recorded timestamp, additional OAM information like incoming and outgoing interface, etc. along with any applicable metadata. Node N1 forwards the original packet towards the next segment 2001:db8:K:2:X31::.
- o When node N2 receives the packet with O-flag set, it ignores the O-flag. This is because node N2 is not capable of processing the O-flag. Node N2 performs the standard SRv6 SID and SRH processing. Specifically, it executes the End.X behavior indicated by the 2001:db8:K:2:X31:: SID as described in [RFC8986] and forwards the packet P1 (2001:db8:L:1::, 2001:db8:K:4:X52::) (2001:db8:K:7:DT999::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=1; O-flag=1; NH=IPv4) (IPv4 header)(payload) over link 3 towards Node N3.
- o When node N3, which is a non-SRv6 capable node, receives the packet P1 , it performs the standard IPv6 processing.

Specifically, it forwards the packet P1 based on DA 2001:db8:K:4:X52:: in the IPv6 header.

- o When node N4 receives the packet P1 (2001:db8:L:1::, 2001:db8:K:4:X52::) (2001:db8:K:7:DT999::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=1; O-flag=1; NH=IPv4) (IPv4 header)(payload), it processes the O-flag. As part of processing the O-flag, it sends a timestamped copy of the packet to a local OAM process. Based on a local configuration, the local OAM process sends a full or partial copy of the packet P1 to the controller N100. The OAM process includes the recorded timestamp, additional OAM information like incoming and outgoing interface, etc. along with any applicable metadata. Node N4 performs the standard SRv6 SID and SRH processing on the original packet P1. Specifically, it executes the End.X behavior indicated by the 2001:db8:K:4:X52:: SID and forwards the packet P1 (2001:db8:L:1::, 2001:db8:K:7:DT999::) (2001:db8:K:7:DT999::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=0; O-flag=1; NH=IPv4) (IPv4 header) (payload) over link 10 towards Node N5.
- o When node N5, which is a non-SRv6 capable node, receives the packet P1, it performs the standard IPv6 processing. Specifically, it forwards the packet based on DA 2001:db8:K:7:DT999:: in the IPv6 header.
- o When node N7 receives the packet P1 (2001:db8:L:1::, 2001:db8:K:7:DT999::) (2001:db8:K:7:DT999::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::; SL=0; O-flag=1; NH=IPv4) (IPv4 header)(payload), it processes the O-flag. As part of processing the O-flag, it sends a timestamped copy of the packet to a local OAM process. The local OAM process sends a full or partial copy of the packet P1 to the controller N100. The OAM process includes the recorded timestamp, additional OAM information like incoming and outgoing interface, etc. along with any applicable metadata. Node N7 performs the standard SRv6 SID and SRH processing on the original packet P1. Specifically, it executes the VPN SID indicated by the 2001:db8:K:7:DT999:: SID and based on lookup in table 100 forwards the packet P1 (IPv4 header)(payload) towards CE 2.
- o The controller N100 processes and correlates the copy of the packets sent from nodes N1, N4 and N7 to find segment-by-segment delays and provide other hybrid OAM information related to packet P1. For segment-by-segment delay computation, it is assumed that clock are synchronized time across the SR domain.
- o The process continues for any other sampled packets.

A.4. Monitoring of SRv6 Paths

In the recent past, network operators demonstrated interest in performing network OAM functions in a centralized manner. [RFC8403] describes such a centralized OAM mechanism. Specifically, the document describes a procedure that can be used to perform path continuity check between any nodes within an SR domain from a centralized monitoring system. However, the document focuses on SR networks with MPLS data plane. This document describes how the concept can be used to perform path monitoring in an SRv6 network from a centralized controller.

In the reference topology in Figure 1, N100 uses an IGP protocol like OSPF or IS-IS 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. The controller leverages the visibility of the topology to monitor the paths between the various endpoints.

The controller N100 advertises an END SID [RFC8986] 2001:db8:K:100:1::.. To monitor any arbitrary SRv6 paths, the controller can create a loopback probe that originates and terminates on Node N100. To distinguish between a failure in the monitored path and loss of connectivity between the controller and the network, Node N100 runs a suitable mechanism to monitor its connectivity to the monitored network.

The loopback probes are exemplified using an example where controller N100 needs to verify a segment list <2001:db8:K:2:X31::, 2001:db8:K:4:X52::>:

- o N100 generates an OAM packet (2001:db8:L:100::, 2001:db8:K:2:X31::) (2001:db8:K:100:1::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::, SL=2) (OAM Payload). The controller routes the probe packet towards the first segment, which is 2001:db8:K:2:X31::.
- o Node N2 executes the End.X behavior indicated by the 2001:db8:K:2:X31:: SID and forwards the packet (2001:db8:L:100::, 2001:db8:K:4:X52::) (2001:db8:K:100:1::, 2001:db8:K:4:X52::, 2001:db8:K:2:X31::, SL=1) (OAM Payload) on link3 to N3.
- o Node N3, which is a non-SRv6 capable node, performs the standard IPv6 processing. Specifically, it forwards the packet based on the DA 2001:db8:K:4:X52:: in the IPv6 header.
- o Node N4 executes the End.X behavior indicated by the 2001:db8:K:4:X52:: SID and forwards the packet (2001:db8:L:100::,

2001:db8:K:100:1::) (2001:db8:K:100:1::, 2001:db8:K:4:X52::,
2001:db8:K:2:X31::, SL=0) (OAM Payload) on link10 to N5.

- o Node N5, which is a non-SRv6 capable node, performs the standard IPv6 processing. Specifically, it forwards the packet based on the DA 2001:db8:K:100:1:: in the IPv6 header.
- o Node N100 executes the standard SRv6 END behavior. It decapsulates the header and consume the probe for OAM processing. The information in the OAM payload is used to detect any missing probes, round trip delay, etc.

The OAM payload type or the information carried in the OAM probe is a local implementation decision at the controller and is outside the scope of this document.

Appendix B. Acknowledgements

The authors would like to thank Joel M. Halpern, Greg Mirsky, Bob Hinden, Loa Andersson, Gaurav Naik, Ketan Talaulikar and Haoyu Song for their review comments.

Appendix C. 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

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

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 19 May 2021

S. Jiang
Huawei Technologies Co., Ltd
G. Li
Huawei Technologies
B. E. Carpenter
Univ. of Auckland
15 November 2020

Asymmetric IPv6 for Resource-constrained IoT Networks
draft-jiang-asymmetric-ipv6-04

Abstract

This document describes a new approach to IPv6 header compression for use in scenarios where minimizing packet size is crucial but routing performance must be maximised.

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 19 May 2021.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. Proposed Solution	3
3. Address Transformation at the Gateway	6
4. Routing without Decompression	6
5. Address Configuration	7
6. Compatibility with Existing Protocols	7
7. Relationship to Static Context Header Compression	7
8. Security Considerations	8
9. IANA Considerations	8
10. Acknowledgements	8
11. References	8
Appendix A. Change log [RFC Editor: Please remove]	10
Authors' Addresses	10

1. Introduction

The large address space of IPv6 is essential for the massive expansion of the network edge that will be caused by "Internet of Things" (IoT) technology over low-power or 5G links. However, the size of a raw IPv6 packet header causes difficulty due to the small maximum transmission units (MTU) allowed by typical low-power, low-cost link layers. For 5G, the importance of header overhead in small packets is discussed in [NGMN-5G]. Thus header compression, including address compression, is an important issue. This decreases the size of raw packets, but compressed IP addresses are not routeable except by decompressing them completely in every forwarding node. There are two issues here. The first is the extra computation resource needed for compressing or decompressing in constrained IoT nodes. The second is that full-length IPv6 routing will consume more memory to store routing tables and packet queues (assuming that routing is not bypassed by tunnelling). Such resource consumption is very undesirable in constrained nodes with limited storage, CPU power, and battery capacity.

To mitigate these issues, here we propose a solution enabling the shortening of IPv6 addresses inside packets, and the routing of packets according to short addresses, without needing the overhead of a decompression step prior to route lookup. Considering that the scale and size of edge networks may vary widely, different lengths of short address can be used in different domains.

As an illustrative example, consider an edge network which is known to never require more than a few hundred nodes, which in most cases will communicate either with each other, or with application layer gateways to the rest of the Internet. Rather than needing 128-bit addresses, such a network could very well operate with 16-bit

addresses. Also, it could very likely operate without needing enhancements such as differentiated services, ECN or flow labels. If only IPv6 is supported, the version number field is pointless. There is no reason for IPv6 packets within such a network to contain 40-byte headers as specified in [RFC8200]. Therefore, the useful information could be carried in 8 bytes (see Figure 1). Furthermore, routers within the edge network can route packets directly on 16-bit addresses, reducing RIB and FIB sizes and the lookup time.

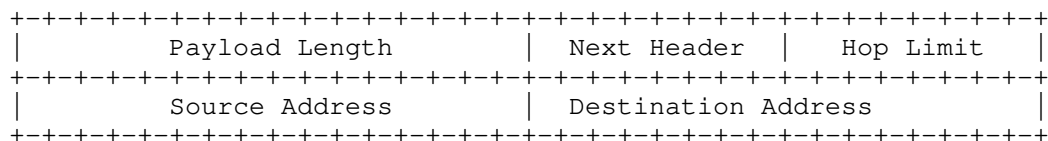


Figure 1

This work is distinct from previous work on address compression [RFC6282] [RFC7400]. Although those solutions tackle the problem of small MTU size, they do not address the problem of decompression overhead.

This work is also distinct from the work on static context header compression [RFC8724], as discussed in more detail below.

Finally, this work is distinct from the 6LoWPAN Routing Header [RFC8138], which can support truncated addresses in a different way.

2. Proposed Solution

The use of IPv6 naturally implies 128-bit addresses for both source and destination. However, this address size is huge by the standards of IoT edge networks. We propose the use of a context parameter to indicate the effective length of the IP address for every node in a local domain. If the effective length is N bits, then all addresses in the domain are assumed to be preceded by a common prefix of 128-N bits, when a full size IPv6 address is needed. Any node in the domain that needs the full address, such as a gateway node to the Internet, can therefore easily synthesize it. If a client communicates with a server that is in the local domain, short addresses will be used end-to-end.

The address length parameter may be needed by every node in the domain. It can be spread by various techniques:

- * Configure the address length in every node.

- * Obtain the address length from a gateway (next hop router) node.
- * Negotiate the address length between neighbors.

The solution operates by shortening IP address fields to save overhead. To enhance this, we propose a new field named Flexible Header Encoding (FHE). It consists of 8 bits, each indicating whether the corresponding IPv6 header field [RFC8200] exists.

- * Bit 0 indicates the Modified Version field
- * Bit 1 indicates the Traffic Class field
- * Bit 2 indicates the Flow Label field.
- * Bit 3 indicates the Payload Length field.
- * Bit 4 indicates the Next Header field. (Zero implies "No Next Header", value 59)
- * Bit 5 indicates the Hop Limit field.
- * Bit 6 indicates the Source Address field.
- * Bit 7 indicates the Destination Address field.

The "Version" field is a special case. In the context of FHE, all packets are presumed to be IPv6 so the normal version field has no purpose. The Modified Version field, if present, has the following encoded meanings:

- * 0b0000: The source address (if exist) has pre-determined length inside the domain and the destination address (if exist) uses standard 128-bit IPv6 address. (Outward traffic)
- * 0b0001: The source address (if exist) uses standard 128-bit IPv6 address and the destination address (if exist) has pre-determined length inside the domain. (Inward traffic)
- * 0b0010: The source address and destination address have the same length inside the domain. The address length will be pre-determined.
- * 0b0110: Reserved for IPv6 compatible case.
- * 0b0100: Reserved for IPv4 compatible case.
- * 0b0011~0b1111(except 0b0110, 0b0100): Reserved.

All fields, including the Modified Version field, follow the FHE in the same order as in [RFC8200], with no padding. There are no alignment requirements, but when a packet is decompressed to a normal IPv6 format, padding options as defined in RFC8200 must be inserted.

Compared to the illustrative example in Figure 1, the actual packet size would therefore be 10 bytes, a considerable improvement on the standard 40 bytes.

One implication of the above is that the source and destination addresses may be elided completely if they are implicit. Sourceless packets were originally suggested in [Crowcroft].

Figure 2 illustrates an example of the FHE format. In this example the traffic class, flow label and source address are elided, and the destination address is truncated to 16 bits. The modified version field could be 0b0001 or 0b0010.

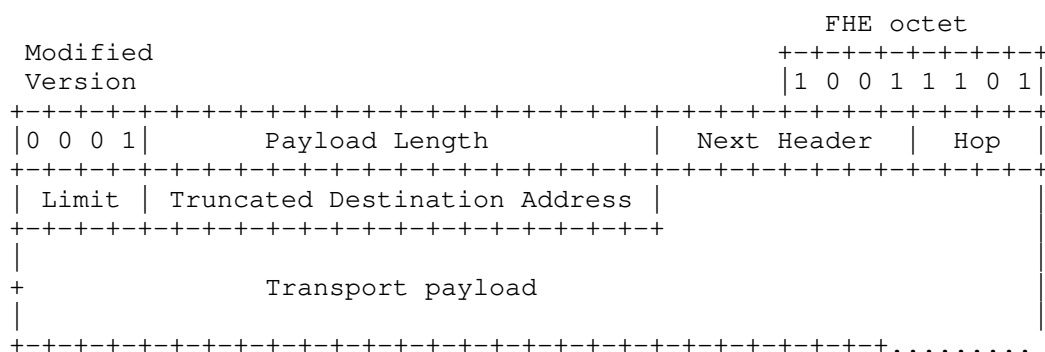


Figure 2

Note that Asymmetric IPv6 does not contain any special handling for IPv6 fragmentation, which will operate exactly as described in [RFC8200], with Asymmetric IPv6 applied to each fragment packet. However, we assume that in IoT deployment scenarios, packets whose length exceeds the IPv6 minimum link MTU before applying Asymmetric IPv6 will be rare. If the underlying link layer cannot carry complete packets even after applying Asymmetric IPv6 compression, an adaptation layer will be necessary exactly as for normal IPv6.

3. Address Transformation at the Gateway

Truncated intra-domain addresses will be used to identify nodes inside the domain. When a packet is sent from an IoT node to an external IPv6 host, the node's intra-domain address, which is unique in the domain, will be carried in the source address field. When the packet is forwarded outside the domain by a gateway, the intra-domain address will be transformed to a complete IPv6 address. To achieve this, the gateway should will maintain a globally routeable prefix for all the nodes in the domain. When a packet with an intra-domain source address is received, the gateway extracts this address and concatenates it to the prefix to form a standard, globally unique IPv6 address. Vice versa, when IPv6 packets are received from the Internet, the prefix will be removed to recover the intra-domain short address.

There are two options for handling the addresses of external hosts within the domain. One is to use their full IPv6 addresses via Modified Version codes 0b0000 and 0b0001. The other is effectively a specialized form of Network Address Translation. Here, the gateway will maintain a dynamic mapping table between synthetic intra-domain addresses and IPv6 addresses. As packets are received, the gateway performs the appropriate mapping. The transformation must be checksum-neutral for the transport layer, so the methods designed for NAT46 should be adapted [RFC6145].

It is an engineering choice whether this method is preferable to carrying full 128-bit addresses on the IOT side. Which type of resource is more expensive should be seriously considered to choose the appropriate ways, e.g. computing, memory, or transmitting in various resource-constrained IoT networks.

4. Routing without Decompression

Routing mechanisms may readily be adapted to truncated address sizes. If there is routing with an HFE domain, we assume that the truncated address size will be split into a prefix and an interface identifier, but this will not be at the traditional /64 boundary. If routing between different length addresses is required, a suitably modified Forwarding Information Base (FIB) structure is needed, as for any variable length addressing scheme. A truncated address needs to be virtually expanded to 128 bits at the router's inbound interface, although this may not be the physical implementation.

A possible routing choice for IOT edge networks is RPL [RFC6550], although a more complete survey can be found in [Talwar].

5. Address Configuration

The simplest approach to address configuration is simply to run normal IPv6 procedures (SLAAC or DHCPv6), on the argument that this is a rare process and the overhead does not matter. If the truncated address size is less than 64 bits, it will be necessary to use shorter interface identifiers than normal, but this is not a major change. Once a node has acquired an IPv6 address and has learned the local address length parameter as outlined in Section 2, it can continue in FHE mode.

6. Compatibility with Existing Protocols

Although HFE nodes can only talk directly to each other, they are essentially a special form of IPv6 node and they can communicate with the whole IPv6 Internet via gateways. The complexity is not greater than 6LoWPAN. If appropriate, the 6LoWPAN adaptation layer [RFC4944] could be used, with a specific dispatch type.

7. Relationship to Static Context Header Compression

Static Context Header Compression (SCHC) [RFC8724] is a powerful mechanism for reducing IPv6 packet size in an IoT application environment. In particular it includes a profile for UDP over IPv6, and a somewhat modified version of this profile could achieve much of what Asymmetric IPv6 proposes. In addition, SCHC provides support for fragmentation in the case of very small link MTUs. However, SCHC is by design static, and once a context is established the fields to be compressed do not change. Asymmetric IPv6 transmits the FHE and Modified Version bytes with every packet, so it provides dynamic choice as to which header elements are compressed or elided.

In a context where the desirable compression is fixed, e.g. every address is the same length, the flow label is never used, etc., SCHC can be used to the same effect as Asymmetric IPv6. However, if the behavior needs to be dynamic, the signaling power of the FHE and Modified Version bytes in Asymmetric IPv6 is needed.

Further study is needed whether the advantages of the two mechanisms can be combined.

8. Security Considerations

HFE is essentially only a non-cryptographic compression technique so it neither adds to nor reduces the intrinsic security of an IPv6 packet. The address length parameter is not a secret, since all nodes in the domain must know it. The mechanism for distributing this parameter must be no less secure than any other configuration mechanism in us.

Address-based privacy issues must be considered in deciding on the address length. If the number of bits available for the interface identifier is significantly less than the 64 currently in use, address traceability and guessability will be affected. However, if the traffic with short addresses is confined to within the edge network, the privacy issue will be minimized. [RFC7721] and [RFC7217] should be consulted prior to deciding the address length.

9. IANA Considerations

This document makes no request of the IANA.

NOTE IN DRAFT: If the solution of a 6LoWPAN dispatch type is adopted, a suitable assignment request will be added.

10. Acknowledgements

Useful comments were received from Uma Chunduri, Cheng Li, Pascal Thubert, Laurent Toutain and others.

11. References

[Crowcroft]

Crowcroft, J. and M. Bagnulo, "SNA: Sourceless Network Architecture", University of Cambridge Computer Laboratory Technical Report UCAM-CL-TR-849, 2014.

[NGMN-5G] Thibault, I., "5G Extreme Requirements: Operators' views on fundamental trade-offs", NGMN Alliance , 2017.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <<https://www.rfc-editor.org/info/rfc4944>>.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, DOI 10.17487/RFC6145, April 2011, <<https://www.rfc-editor.org/info/rfc6145>>.

- [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>>.
- [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>>.
- [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>>.
- [RFC7721] Cooper, A., Gont, F., and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms", RFC 7721, DOI 10.17487/RFC7721, March 2016, <<https://www.rfc-editor.org/info/rfc7721>>.
- [RFC8138] Thubert, P., Ed., Bormann, C., Toutain, L., and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing Header", RFC 8138, DOI 10.17487/RFC8138, April 2017, <<https://www.rfc-editor.org/info/rfc8138>>.
- [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>>.
- [RFC8724] Minaburo, A., Toutain, L., Gomez, C., Barthel, D., and JC. Zúñiga, "SCHC: Generic Framework for Static Context Header Compression and Fragmentation", RFC 8724, DOI 10.17487/RFC8724, April 2020, <<https://www.rfc-editor.org/info/rfc8724>>.
- [Talwar] Talwar, M., "Routing Techniques and Protocols for Internet of Things: a Survey", Indian J.Sci.Res. 12(1):417-423, 2015.

Appendix A. Change log [RFC Editor: Please remove]

- * draft-jiang-asymmetric-ipv6-00, 2019-06-03:
 - Initial version
- * draft-jiang-asymmetric-ipv6-01, 2019-06-21:
 - Fixed reference error
- * draft-jiang-asymmetric-ipv6-02, 2019-10-29:
 - Added illustrative example
 - Discussed fragmentation
 - Discussed relationship to SCHC
 - Fixed bit pattern errors
- * draft-jiang-asymmetric-ipv6-03, 2020-05-15:
 - Minor technical and editorial fixes
 - Converted to xml2rfc v3
- * draft-jiang-asymmetric-ipv6-04, 2020-11-15:
 - Explicitly limit the scope to resource-constrained domain
 - Add engineering choice considerations accordingly

Authors' Addresses

Sheng Jiang
Huawei Technologies Co., Ltd
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: jiangsheng@huawei.com

Guangpeng Li
Huawei Technologies
Q14, Huawei Campus
No.156 Beiqing Road
Hai-Dian District, Beijing

100095
P.R. China

Email: liguangpeng@huawei.com

Brian Carpenter
The University of Auckland
School of Computer Science
University of Auckland
PB 92019
Auckland 1142
New Zealand

Email: brian.e.carpenter@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 7, 2020

Z. Li
S. Peng
Huawei Technologies
D. Voyer
Bell Canada
C. Xie
China Telecom
P. Liu
China Mobile
C. Liu
China Unicom
K. Ebisawa
Toyota Motor Corporation
S. Previdi
Individual
J. Guichard
Futurewei Technologies Ltd.
November 04, 2019

Problem Statement and Use Cases of Application-aware IPv6 Networking
(APN6)
draft-li-apn6-problem-statement-usecases-01

Abstract

Network operators are facing the challenge of providing better network services for users. As the ever developing 5G and industrial verticals evolve, more and more services that have diverse network requirements such as ultra-low latency and high reliability are emerging, and therefore differentiated service treatment is desired by users. However, network operators are typically unaware of which applications are traversing their network infrastructure, which means that only coarse-grained services can be provided to users. As a result, network operators are only evolving their infrastructure to be large but dumb pipes without corresponding revenue increases that might be enabled by differentiated service treatment. As network technologies evolve including deployments of IPv6 and SRv6, the programmability provided by IPv6 and SRv6 encapsulations can be augmented by conveying application related information into the network. Adding application knowledge to the network layer allows applications to specify finer granularity requirements to the network operator.

This document analyzes the existing problems caused by lack of application awareness, and outlines various use cases that could benefit from an Application-aware IPv6 Networking (APN6) architecture.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on May 7, 2020.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
2. Terminology	3
3. Problem Statement	4
3.1. Large but Dumb Pipe	4
3.2. Network on Its Own	4
3.3. Decoupling of Network and Applications	4
3.4. Challenges of Traditional Differentiated Service Provisioning	5

3.5. Challenges of Supporting New 5G and Edge Computing Technologies	6
4. Key Elements of Application-aware IPv6 Networking (APN6)	6
5. Use cases for Application-aware IPv6 Networking (APN6)	8
5.1. Application-aware SLA Guarantee	8
5.2. Application-aware network slicing	9
5.3. Application-aware Deterministic Networking	9
5.4. Application-aware Service Function Chaining	10
5.5. Application-aware Network Measurement	10
6. IANA Considerations	11
7. Security Considerations	11
8. Acknowledgements	11
9. Contributors	11
10. References	12
10.1. Normative References	12
10.2. Informative References	12
Authors' Addresses	12

1. Introduction

Due to the requirement for differentiated traffic treatment driven by diverse new services, the ability to convey the characteristics of an application's traffic flow and program the network infrastructure accordingly to provide fine-grained service assurance is becoming increasingly necessary for network operators. The Application-aware IPv6 Networking (APN6) architecture is being defined to address the requirements and use cases described in this document. APN6 takes advantage of network programmability by conveying application related information in the data plane allowing applications to specify finer grained requirements to the network infrastructure.

2. Terminology

ACL: Access Control List

APN6: Application-aware IPv6 Networking

DPI: Deep Packet Inspection

PBR: Policy Based Routing

QoE: Quality of Experience

SDN: Software Defined Networking

3. Problem Statement

This section summarizes the challenges currently faced by network operators when attempting to provide fine-grained traffic operations to satisfy the various application-awareness requirements demanded by new services that require differentiated service treatment.

3.1. Large but Dumb Pipe

In today's networks, the infrastructure through which user traffic is forwarded is not able to determine information about the packet, including which application the traffic belongs to, without the introduction of middleware such as DPI, that is, the network and applications are decoupled. It is therefore difficult for network operators to provide fine-grained traffic operations for performance-demanding applications. In order to satisfy the SLA requirements network operators continue to increase the network bandwidth but only carrying very light traffic load (around 30%-40% of its capacity). This situation greatly increases the CAPEX and OPEX but only brings very little revenue from the carried services.

3.2. Network on Its Own

As the network evolves, technologies such as VPN/TE/FRR play important roles in satisfying service isolation, SLA guarantee, and high reliability, etc. These network technologies have themselves been evolving, introducing new features that forces the network operator to be continuously upgrading their network infrastructure. However, none of these network technologies make the network aware of which application traffic belongs to and the fine granularity requirements of the application. Therefore, such continuous network infrastructure upgrade doesn't always enable true fine-grained traffic operation, therefore reducing the ability to bring corresponding revenue increase.

3.3. Decoupling of Network and Applications

MPLS played a very important role in helping the network enter the generation of All-IP successfully. However, MPLS doesn't allow a close interworking with the application layer since MPLS encapsulation is, typically, not used by the packet source.

As new services continuously evolve, more encapsulations are required, and this isolation and decoupling has further become the blockage towards the seamless convergence of the network and applications.

3.4. Challenges of Traditional Differentiated Service Provisioning

Several IETF activities have been reviewed 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 an SLA is that the packets are not able to carry enough information for indicating applications and expressing their service/SLA requirements. The network devices mainly rely on the 5-tuple of the packets or DPI. However, there are some challenges for these traditional methods in differentiated service provisioning:

1. Five Tuples used for ACL/PBR

Five tuples are widely used for ACL/PBR matching of traffic. However, these features cannot provide enough information for the fine-grained service process, and can only provide indirect application information which needs to be translated in order to indicate a specific application.

2. Deep Packet Inspection (DPI)

If more information is needed, it must be extracted using DPI which can inspect deep into the packets for application specific information. However, this will introduce more CAPEX and OPEX for the network operator and imposes security challenges.

3. Orchestration and SDN-based Solution

In the era of SDN, typically, an SDN controller is used to manage and operate the network infrastructure and orchestrator elements introduce application requirements so that the network is programmed accordingly. The SDN controller can be aware of the service requirements of the applications on the network through the interface with the orchestrator, and the service requirement is used by the controller for traffic management over the network. However, this method raises the following problems:

1) The whole loop is long and time-consuming which is not suitable for fast service provisioning for critical applications;

2) Too many interfaces are involved in the loop, as shown in Figure 1, which introduce challenges of standardization and interoperability.

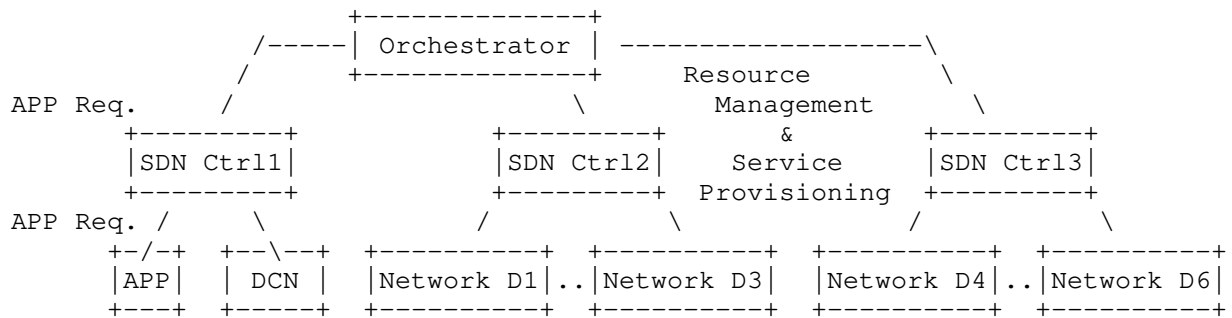


Figure 1. Many interfaces involved in the long service-provisioning loop

3.5. Challenges of Supporting New 5G and Edge Computing Technologies

New technologies such as 5G, IoT, and edge computing, are continuously developing leading to more and more new types of services accessing the network. Large volumes of network traffic with diverse requirements such as low latency and high reliability are therefore rapidly increasing. If traditional methods for differentiation of traffic continue to be utilized, it will cause much higher CAPEX and OPEX to satisfy the ever-developing applications' diverse requirements.

4. Key Elements of Application-aware IPv6 Networking (APN6)

Application-aware IPv6 Networking (APN6) aims to address the aforementioned problems associated with fine-grained traffic operations that are required in order to satisfy the various application-awareness requirements demanded by new services that need differentiated service treatment. APN6 conveys information into the network infrastructure about the characteristics of the application associated with a traffic flow (including application identification and network performance requirements), allowing the network to quickly adapt and perform the necessary network resource adjustments to maintain SLA performance guarantees, and hence better serve application fine-grained service requirements.

The advantages of using IPv6 to support APN6 include,

1. **Simplicity:** Conveying application information with IPv6 encapsulation can just be based on IP reachability.
2. **Seamless convergence:** Much easier to achieve seamless convergence between applications and network since both are based on IPv6.

3. Great extensibility: IPv6 encapsulation including its extension headers can be used to carry very rich information relevant to applications.
4. Good compatibility: On-demand network upgrade and service provisioning. If the application information is not recognized by the node, the packet will be forwarded based on pure IPv6, which ensure backward compatibility.
5. Little dependency: Information conveying and service provisioning are only based on the forwarding plane of devices, which is different from the Orchestration and SDN-based solution which involves multiple elements and diverse interfaces.
6. Quick response: Flow-driven and direct response from devices since it is based on the forwarding plane.

APN6 has the following key elements:

1. Application information should be conveyed in the data plane through augmentation of existing encapsulations such as IPv6 and/or SRv6. The conveyed application characteristic information (application-aware information) includes application identification and/or its network performance requirements. This element should not be enforced but provide an open option for applications to decide whether to input this application-aware information into their data stream.
2. Application information and network service provisioning matching providing fine-granularity network service provisioning (traffic operations) and SLA guarantee based on the application-aware information carried in APN6 packets. This element provides the network capabilities to applications. According to the application-aware information, appropriate network services are selected, provisioned, and provided to the demanding applications to satisfy their performance requirements.
3. Network measurement of network performance and update the match between the applications and corresponding network services for better fine-granularity SLA compliance. The network measurement methods include in-band and out-of-band, passive, active, per-packet, per-flow, per node, end-to-end, etc. These methods can also be integrated.

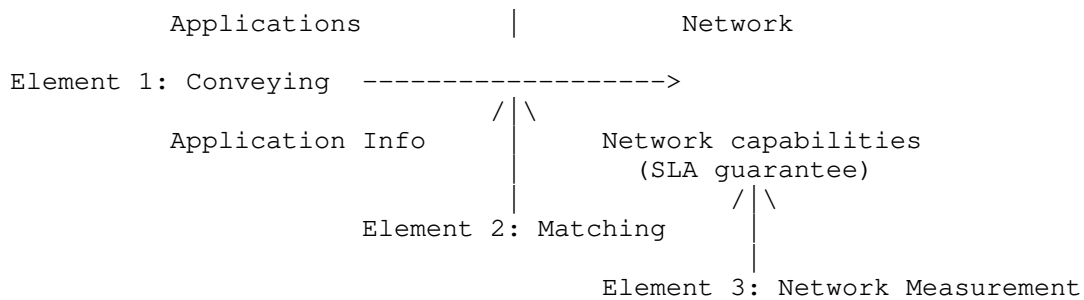


Figure 2. Illustration of the key elements of APN6

5. Use cases for Application-aware IPv6 Networking (APN6)

This section provides the use cases that can benefit from the application awareness introduced by APN6. The corresponding requirements for APN6 are also outlined.

5.1. Application-aware SLA Guarantee

One of the key objectives of APN6 is for network operators to provide fine-granularity SLA guarantees instead of coarse-grain traffic operations. Among various applications being carried and running in the network, some revenue-producing applications such as online gaming, video streaming, and enterprise video conferencing have much more demanding performance requirements such as low network latency and high bandwidth. In order to achieve better Quality of Experience (QoE) for end users and engage customers, the network needs to be able to provide fine-granularity and even application-level SLA guarantee. Differentiated service provisioning is also desired.

One of the key objective of APN6 is for network operators to provide fine-granularity SLA guarantees instead of coarse-grain traffic operations. This will enable them to provide differentiated services for different applications and increase revenue accordingly.

The APN6 architecture design MUST address the following requirements:

- o APN6 needs to perform the three key elements as described in Section 4.
- o Support application-level fine-granularity traffic operation that may include finer QoS scheduling.

5.2. Application-aware network slicing

More and more applications/services with diverse requirements are being carried over and sharing the network operators' network infrastructure. However, it is still desirable to have customized network transport that can support some application's specific requirements, taking into consideration service and resource isolation, which drives the concept of network slicing.

Network slicing provides ways to partition the network infrastructure in either the control plane or data plane into multiple network slices that are running in parallel. These network slices can serve diverse services and fulfill their various requirements at the same time. For example, the mission critical application that requires ultra-low latency and high reliability can be provisioned over a separate network slice.

The APN6 architecture design MUST address the following requirements:

- o APN6 needs to perform the three key elements as described in Section 4 in the context of network slicing. To be more specific, for element 2, it needs to match to a specific network slice according to the application information carried in the APN6 packets. The network measurement in element 3 also needs to happen within each network slice.

5.3. Application-aware Deterministic Networking

[RFC8578] documents use cases for diverse industry applications that require deterministic flows over multi-hop paths. Deterministic flows provide guaranteed bandwidth, bounded latency, and other properties relevant to the transport of time-sensitive data, and can coexist on an IP network with best-effort traffic. It also provides for highly reliable flows through provision for redundant paths.

The APN6 architecture design MUST address the following requirements:

- o APN6 needs to perform the three key elements as described in Section 4 in the context of deterministic networking. To be more specific, for the element 2, it needs to match to a specific deterministic path according to the application information carried in the APN6 packets. The network measurement in element 3 also needs to be performed on each application-aware deterministic path.

5.4. Application-aware Service Function Chaining

End-to-end service delivery often needs to go through various service functions, including traditional network service functions such as firewalls, DPIs as well as new application-specific functions, both physical and virtual. The definition and instantiation of an ordered set of service functions and subsequent steering of the traffic through them is called Service Function Chaining (SFC) [RFC7665]. SFC is applicable to both fixed and mobile networks as well as data center networks.

Generally, in order to manipulate a specific application traffic along the SFC, a DPI needs to be deployed as the first service function of the chain to detect the application, which will impose high CAPEX and consume long processing times. For encrypted traffic, it even becomes impossible to inspect the application.

The APN6 architecture design MUST address the following requirements:

- o APN6 needs to perform the three key elements as described in Section 4 in the context of service function chaining. To be more specific, for element 1 class information can be conveyed. For element 2, it needs to match to a specific service function chain and subsequent steering according to the application information carried in the APN6 packets. The network measurement in element 3 also needs to happen within each app-aware service function chain.

5.5. Application-aware Network Measurement

Network measurement can be used for locating silent failure and predicting QoE satisfaction, which enables real-time SLA awareness/proactive OAM. Operations, Administration, and Maintenance (OAM) refers to a toolset for fault detection and isolation, and network performance measurement. In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network.

The APN6 architecture MUST address the following requirements:

- o APN6 needs to perform the two key elements as described in Section 4 in the context of network measurement. The network measurement in element 3 does not need to be considered here.

6. IANA Considerations

This document does not include an IANA request.

7. Security Considerations

Since the application information is conveyed into the network, it does involve some security and privacy issues.

First, APN6 only provides the capability to the applications to provide their profiles and requirements to the network, but it leaves the applications to decide whether to input this information. If the applications decide not to provide any information, they will be treated in the same way as today's network and cannot get the benefits from APN6.

Once the application information has been carried in the IPv6 packets and conveyed into the network, the IPv6 extension headers, AH and ESP, can be used to guarantee the authenticity of the added application information.

Any scheme involving an information exchange between layers (application and network layers in this case) will obviously require an accurate valuation of security mechanism in order to prevent any leak of critical information. Some additional considerations may be required for multi-domain use cases. For example, how to agree upon which application information/ID to use and guarantee authenticity for packets traveling through multiple domains (network operators).

8. Acknowledgements

The authors would like to acknowledge Robert Raszuk (Bloomberg LP) and Yukito Ueno (NTT Communications Corporation) for their valuable review and comments.

9. Contributors

Liang Geng
China Mobile
China

Email: gengliang@chinamobile.com

Chang Cao
China Unicom
China

Email: caoc15@chinaunicom.cn

Cong Li
China Telecom
China

Email: licong.bri@chinatelecom.cn

10. References

10.1. 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>>.
- [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>>.
- [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>>.
- [RFC8578] Grossman, E., Ed., "Deterministic Networking Use Cases", RFC 8578, DOI 10.17487/RFC8578, May 2019, <<https://www.rfc-editor.org/info/rfc8578>>.

10.2. Informative References

- [I-D.ietf-6man-segment-routing-header] Filks, 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-26 (work in progress), October 2019.
- [I-D.ietf-spring-srv6-network-programming] Filks, C., Camarillo, P., Leddy, J., daniel.voyer@bell.ca, d., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-05 (work in progress), October 2019.

Authors' Addresses

Zhenbin Li
Huawei Technologies
China

Email: lizhenbin@huawei.com

Shuping Peng
Huawei Technologies
China

Email: pengshuping@huawei.com

Daniel Voyer
Bell Canada
Canada

Email: daniel.voyer@bell.ca

Chongfeng Xie
China Telecom
China

Email: xiechf.bri@chinatelecom.cn

Peng Liu
China Mobile
China

Email: liupengyjy@chinamobile.com

Chang Liu
China Unicom
China

Email: liuc131@chinaunicom.cn

Kentaro Ebisawa
Toyota Motor Corporation
Japan

Email: ebisawa@toyota-tokyo.tech

Stefano Previdi
Individual
Italy

Email: stefano@previdi.net

James N Guichard
Futurewei Technologies Ltd.
USA

Email: jguichar@futurewei.com

IPv6 Maintenance
Internet-Draft
Updates: 4861 (if approved)
Intended status: Standards Track
Expires: May 29, 2020

J. Linkova
Google
November 26, 2019

Gratuitous Neighbor Discovery: Creating Neighbor Cache Entries on First-
Hop Routers
draft-linkova-6man-grand-01

Abstract

Neighbor Discovery (RFC4861) is used by IPv6 nodes to determine the link-layer addresses of neighboring nodes as well as to discover and maintain reachability information. This document updates [RFC4861] to allow routers to proactively create a Neighbor Cache entry when a new IPv6 address is assigned to a host. It also updates [RFC4862] and recommends hosts to send unsolicited Neighbor Advertisements upon assigning a new IPv6 address. The proposed change will minimize the delay and packet loss when a host initiate connections to off-link destination from a new IPv6 address.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 29, 2020.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	2
1.1. Requirements Language	3
1.2. Terminology	3
2. Proposed Changes to Neighbor Discovery	4
2.1. Hosts Sending Gratuitous Neighbor Advertisements	4
2.2. Routers Creating Cache Entries Upon Receiving Unsolicited Neighbor Advertisements	4
3. Avoiding Disruption	5
3.1. Neighbor Cache Entry Exists in Any State Other Than INCOMPLETE	5
3.2. Neighbor Cache Entry Does Not Exist	5
3.3. Neighbor Cache Entry is in INCOMPLETE state	6
4. Modifications to RFC-Mandated Behavior	6
4.1. Modification to RFC4861 Neighbor Discovery for IP version 6 (IPv6)	6
4.1.1. Modification to the section 7.2.5	6
4.1.2. Modification to the section 7.2.6	7
5. IANA Considerations	8
6. Security Considerations	8
7. Acknowledgements	8
8. References	9
8.1. Normative References	9
8.2. Informative References	9
Author's Address	10

1. Introduction

The Neighbor Discovery state machine defined in [RFC4861] implies that communications between IPv6 nodes are in most cases bi-directional and if a host A is trying to communicate to its neighbor, host B, the return traffic flows could be expected. So when the host A starts the address resolution process, the target host would also create an entry for the host A address in its neighbor cache. That entry will be used for sending the return traffic to the host A.

However when a host sends traffic to off-link destinations the different scenario is observed. After receiving a Router Advertisement the host populates its neighbor cache with the default router IPv6 and link-layer addresses and is able to send traffic to

off-link destinations. At the same time the router does not have any cache entries for the host global addresses yet and only starts address resolution upon receiving the first packet of the return traffic flow. While waiting for the resolution to complete routers only keep a very small number of packets in the queue (as recommended in [RFC4861] Section 7.2.2. All subsequent packets arriving before the resolution process finishes are likely to be dropped. It might cause user-visible packet loss and performance degradation

The detailed problem statement and various solution approaches could be found in [I-D.ietf-v6ops-nd-cache-init]. This document summarized the proposed neighbor discovery updates to address the issue.

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

ND: Neighbor Discovery, [RFC4861].

SLAAC: IPv6 Stateless Address Autoconfiguration, [RFC4862].

NS: Neighbor Solicitation, [RFC4861].

NA: Neighbor Advertisement, [RFC4861].

RS: Router Solicitation, [RFC4861].

RA: Router Advertisement, [RFC4861].

LLA: Link-Layer Address.

SLLA: Source link-layer Address, an option in the ND packets containing the link-layer address of the sender of the packet ([RFC4861]).

TLLA: Target link-layer Address, an option in the ND packets containing the link-layer address of the target ([RFC4861]).

GUA: Global Unicast Address ([RFC4291]).

DAD: Duplicate Address Detection, [RFC4862].

Optimistic DAD: a modification of DAD, [RFC4429].

2. Proposed Changes to Neighbor Discovery

The following changes are proposed to minimize the delay in creating new entries in a router neighbor cache

- o A host SHOULD send unsolicited NAs upon assigning a new IPv6 address to its interface.
- o A router SHOULD create a new cache entry upon receiving an unsolicited NA from a host.

The following sections discuss these changes in more detail.

2.1. Hosts Sending Gratuitous Neighbor Advertisements

The section 7.2.6 of [RFC4861] discusses using unsolicited Neighbor Advertisement to inform node neighbors of the new link-layer address quickly. The same mechanism could be used to notify the host neighbors about the new network-layer address as well: the host can send gratuitous unsolicited Neighbor Advertisements upon assigning a new global IPv6 address to its interface.

To minimize the potential disruption in case of duplicate addresses the host SHOULD NOT set the Override flag for a preferred address and MUST NOT set the Override flag if the address is in Optimistic [RFC4429] state.

As the main purpose of sending unsolicited NAs upon configuring a new address is to proactively create a Neighbor Cache entry on the first-hop routers, the gratuitous NAs SHOULD be sent to all-routers multicast address (ff02::2). Limiting the recipients to routers only would help reduce the multicast noise level.

2.2. Routers Creating Cache Entries Upon Receiving Unsolicited Neighbor Advertisements

The section 7.2.5 of [RFC4861] states: "When a valid Neighbor Advertisement is received (either solicited or unsolicited), the Neighbor Cache is searched for the target's entry. If no entry exists, the advertisement SHOULD be silently discarded. There is no need to create an entry if none exists, since the recipient has apparently not initiated any communication with the target".

The reasoning behind dropping unsolicited Neighbor Advertisements ("the recipient has apparently not initiated any communication with the target") is valid for onlink host-to-host communication but, as

discussed in [I-D.ietf-v6ops-nd-cache-init] does not really apply for the scenario when the host is announcing its address to routers. Therefore it would be beneficial to allow routers creating new entries upon receiving an unsolicited Neighbor Advertisement.

This document suggests that routers SHOULD create a new Neighbor Cache entry when receive an unsolicited Neighbor Advertisement.

3. Avoiding Disruption

If hosts following the recommendations in this document are using the DAD mechanism defined in [RFC4862], they would send unsolicited NA as soon as the address changes the state from tentative to preferred (after its uniqueness has been verified). However hosts willing to minimize network stack configuration delays might be using optimistic addresses, which means there is a possibility of the address not being unique on the link. The section 2.2 of [RFC4429] discusses measures to ensure that ND packets from the optimistic address do not override any existing neighbor cache entries as it would cause traffic interruption of the rightful address owner in case of address conflict. As hosts willing to speed up their network stack configuration are most likely to be affected by the problem outlined in this document it seems reasonable for such hosts to advertise their optimistic GUAs by sending unsolicited NAs. The main question to consider is the potential risk of overriding the cache entry for the rightful address owner if the optimistic address happens to be duplicated.

3.1. Neighbor Cache Entry Exists in Any State Other Than INCOMPLETE

If the router Neighbor Cache entry for the target address already exists in any state other than INCOMPLETE, then as per section 7.2.5 of [RFC4861] an unsolicited NA with the Override flag cleared would change the entry state from REACHABLE to STALE but would not update the entry in any other way. Therefore even if the host sends an unsolicited NA from the its Optimistic address the router cache entry would not be updated with the new Link-Layer address and no impact to the traffic for the rightful address owner is expected.

3.2. Neighbor Cache Entry Does Not Exist

If there is no entry then it would be created/updated with the supplied LLA and its state set to STALE. In that case as soon as the entry is used for sending traffic to the host, the entry state will be changed to DELAY and the Neighbor Unreachability Detection would be started and the rightful owner LLA will be entered in the cache. So in the scenario when the rightful owner does not use the address for communication then it might be a short (a few seconds) period of

time when the data packets sent from the outside could reach the host with the optimistic address. However it seems likely that hosts using Optimistic DAD would start sending/receiving traffic right away, so the first return packet would trigger the NUD process and rewrite the cache.

3.3. Neighbor Cache Entry is in INCOMPLETE state

Another corner case is the INCOMPLETE cache entry for the address. If the host sends an unsolicited NA from the Optimistic address it would update the entry with the host LLA and set the entry to the STALE state. As the INCOMPLETE entry means that the router has started the ND process for the address and the multicast NS has been sent, the rightful owner is expected to reply with solicited NA with the Override flag set. Upon receiving a solicited NA with the Override flag the cache entry will be updated with the TLLA supplied and (as the NA has the Solicited flag set), the entry state will be set to REACHABLE. IT would would recover the cache entry and set the LLA to the one of the rightful owner. The only potential impact would be for packets arriving to the router after the unsolicited NA from the host but before the rightful owner responded with the solicited NA. Those packets would be sent to the host with the optimistic address instead of its rightful owner. However those packets would have been dropped anyway as until the solicited NA is received the router can not send the traffic.

4. Modifications to RFC-Mandated Behavior

All normative text in this memo is contained in this section.

4.1. Modification to RFC4861 Neighbor Discovery for IP version 6 (IPv6)

4.1.1. Modification to the section 7.2.5

This document proposes the following changes to the section 7.2.5 of [RFC4861]:

OLD TEXT:

When a valid Neighbor Advertisement is received (either solicited or unsolicited), the Neighbor Cache is searched for the target's entry. If no entry exists, the advertisement SHOULD be silently discarded. There is no need to create an entry if none exists, since the recipient has apparently not initiated any communication with the target.

NEW TEXT:

When a valid Neighbor Advertisement is received (either solicited or unsolicited), the Neighbor Cache is searched for the target's entry. If no entry exists, hosts SHOULD silently discard the advertisement. There is no need to create an entry if none exists, since the recipient has apparently not initiated any communication with the target. Routers SHOULD create a new entry for the target address with the link-layer address set to the Target link-layer address option (if supplied). The entry its reachability state MUST also be set to STALE. If the received Neighbor Advertisement does not contain the Target link-layer address option the advertisement SHOULD be silently discarded.

4.1.1.2. Modification to the section 7.2.6

This document proposes the following changes to the section 7.2.6 of [RFC4861]:

OLD TEXT:

In such cases, a node MAY send up to MAX_NEIGHBOR_ADVERTISEMENT unsolicited Neighbor Advertisement messages to the all-nodes multicast address. These advertisements MUST be separated by at least RetransTimer seconds.

NEW TEXT:

In such cases, a node MAY send up to MAX_NEIGHBOR_ADVERTISEMENT unsolicited Neighbor Advertisement messages to the all-nodes multicast address. These advertisements MUST be separated by at least RetransTimer seconds.

A host may also wish to notify its first-hop routers when it configures a new global IPv6 address so the routers can proactively populate their neighbor caches with the corresponding entries. In such cases a host SHOULD send up to MAX_NEIGHBOR_ADVERTISEMENT Neighbor Advertisement messages. If the address is preferred then the Override flag SHOULD NOT be set. If the address is in the Optimistic state then the Override flag MUST NOT be set. The destination address SHOULD be set to the all-routers multicast address. These advertisements MUST be separated by at least RetransTimer seconds. The first advertisement SHOULD be sent as soon as one of the following events happens:

- o if Optimistic DAD [RFC4429] is used: a new Optimistic GUA is assigned to the host interface.
- o if Optimistic DAD is not used: a GUA changes the state from tentative to preferred.

5. IANA Considerations

This memo asks the IANA for no new parameters.

6. Security Considerations

One of the potential attack vectors to consider is a cache spoofing when the attacker might try to install a cache entry for the victim's IPv6 address and the attacker's Link-Layer address. However it should be noted that this document does not propose any changes for the scenario when the ND cache for the given IPv6 address already exists. Therefore it is not possible for the attacker to override any existing cache entry.

A malicious host could attempt to exhaust the neighbor cache on the router by creating a large number of STALE entries. However this attack vector is not new and this document does not increase the risk of such an attack: the attacker could do it, for example, by sending a NS or RS packet with SLLAO included. All recommendations from [RFC6583] still apply.

Announcing a new address to all-routers multicast address may inform an on-link attacker about IPv6 addresses assigned to the host. However hiding information about the specific IPv6 address should not be considered a security measure as it falls into 'Security through obscurity' category. If peer-to-peer onlink communications are not desirable they should be prevented by proper layer2 security mechanisms. Therefore the risk of allowing hosts to send unsolicited Neighbor Advertisements to all-routers multicast address is low.

7. Acknowledgements

Thanks to the following people (in alphabetical order) for their comments, review and feedback: Lorenzo Colitti, Tatuya Jinmei, Erik Kline, Warren Kumari, Erik Nordmark, Michael Richardson, Dave Thaler, Pascal Thubert, Loganaden Velvindron, Eric Vyncke.

8. References

8.1. Normative References

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

8.2. Informative References

- [I-D.ietf-v6ops-nd-cache-init] Linkova, J., "Neighbor Cache Entries on First-Hop Routers: Operational Considerations", draft-ietf-v6ops-nd-cache-init-00 (work in progress), October 2019.
- [RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, DOI 10.17487/RFC4941, September 2007, <<https://www.rfc-editor.org/info/rfc4941>>.
- [RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583, DOI 10.17487/RFC6583, March 2012, <<https://www.rfc-editor.org/info/rfc6583>>.

Author's Address

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

Email: furry@google.com

Internet Engineering Task Force
Internet-Draft
Updates: RFC4291, RFC4443, RFC6724 (if
approved)
Intended status: Standards Track
Expires: May 6, 2020

M. Smith
November 3, 2019

IPv6 Formal Anycast Addresses and Functional Anycast Addresses
draft-smith-6man-form-func-anycast-addresses-01

Abstract

Currently, IPv6 anycast addresses are chosen from within the existing IPv6 unicast address space, with the addresses nominated as anycast addresses through configuration. An alternative scheme would be to have a special class of addresses for use as anycast addresses. This memo proposes a distinct general anycast addressing class for IPv6, and a more specific scheme for functional anycast addresses.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 6, 2020.

Copyright Notice

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

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

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	4
3. Drawbacks of Informal Anycast Addresses	4
4. Formal Anycast Addresses	5
4.1. Address Format	5
4.2. Address Fields	5
4.2.1. Formal Anycast Prefix	6
4.2.2. Visible Scope	6
4.2.3. Anycast Identifier Format	6
4.2.4. Anycast Identifier	6
4.3. Anycast Address Registration Protocol	6
4.4. Network Service Provider Visible Scope	7
4.5. Link-Local Visible Scope	7
4.6. ICMPv6 Destination Unreachable Message	8
4.7. Default Address Selection	9
4.7.1. Formal Anycast Scope Comparison	9
4.7.2. Source Address Selection	9
4.7.3. Destination Address Selection	9
4.8. Non-Local Anycast Forwarding	10
4.9. Advice on Structuring the Anycast Identifier Field Values	11
5. Functional Anycast Addresses	13
5.1. Features	13
5.2. Address Format	14
5.3. Assignment of Anycast Function Identifiers	16
5.4. Assigned Anycast Function Identifiers	17
5.5. Sources of Inspiration for Anycast Function Identifiers	18
5.6. Global Scope Functional Anycast Addresses on the Internet	19
5.7. Example Use Cases	20
5.7.1. Devices Factory Configured with NTP Functional Anycast Addresses	20
5.7.2. Branch Office DNS Resolvers	22
5.7.3. Automatic eBGP Session Establishment	23
5.7.4. An ISP's Anycast DNS Resolvers	25
5.7.5. Microservices Architecture Applications	27
5.7.6. Global Time Distribution Network	27
5.7.7. Multipath Transport Layer Protocols	27
6. Security Considerations	29
7. IANA Considerations	29
8. Acknowledgements	30
9. Change Log [RFC Editor please remove]	30
10. References	30
10.1. Normative References	30

10.2. Informative References	30
Author's Address	35

1. Introduction

[RFC1546] was the first description of host anycast services, and proposed two ways of supporting them in terms of addressing:

- o using parts of the existing address space
- o create a special class of addresses for anycast use

The first method of supporting anycast addresses, by using parts of the existing (unicast) address space, could be described as informal. From the address itself, it is not possible to determine that the apparent unicast address is actually being used as an anycast address.

As the second method would create a special class of addresses for anycast use, it could be described as formal. Encoded within the addresses would be a well known value that indicates they are anycast addresses, regardless of context.

In terms of a spectrum of packet delivery, ranging from delivery to a single destination (unicast), through to delivery to multiple destinations (multicast), anycast addresses are a distinct class of addresses when compared to unicast and multicast addresses.

Packets sent to a unicast destination are intended to be delivered to one and only one unique destination host. Packets sent to a multicast destination are intended to be delivered to a group of interested destination hosts, with the interested group consisting of one or more members, and the packets being duplicated by the network when and where necessary.

Packets sent to an anycast destination are intended to be delivered to only one host, however that host is a member a set of hosts sharing the same anycast address. As a type of address, anycast addresses can be imagined to fall between unicast and multicast address types on a packet delivery spectrum. Packet delivery to an anycast address shares characteristics of both unicast and multicast address packet delivery.

IPv6 anycast addresses [RFC4291] are currently from within the existing unicast address address space. Therefore, this memo gives these IPv6 anycast addresses the name "Informal Anycast" addresses.

This memo proposes a distinct and formal class of IPv6 addresses for anycast use, calling them "Formal Anycast" addresses.

The described IPv6 Formal Anycast address class can support a total of 16 sub-classes of anycast address formats and structures, allowing other semantics to be encoded in the anycast address. Following the definition of the Formal Anycast address class, this memo then proposes the first sub-class, called "Functional Anycast" addresses.

There are some existing reserved and well known anycast addresses within the existing Informal Anycast address space, that have been assigned by IANA [IANA-IPV6ANYC]. While well known, they do not have any of the formal attributes that the proposed formal Functional Anycast addresses have, other than having specified and well known values; they could be described as semi-formal. Well known Functional Anycast addresses are proposed that correspond to these existing semi-formal anycast addresses.

"MRS:" comments - points to consider further, to eventually be removed.

2. Terminology

Anycast Domain

Formal Anycast Address

Functional Anycast Address

Informal Anycast Address

Semi-Formal Anycast Address

3. Drawbacks of Informal Anycast Addresses

There are drawbacks and limitations of the existing IPv6 Informal Anycast addresses:

- o As mentioned in the Introduction, there is nothing specifically encoded in an Informal Anycast address to distinguish it from a unicast address, such as being from within a well known address prefix. In some situations, this unintentional obfuscation may be of use, however in others, such as while troubleshooting, it can be detrimental. For example, duplicate routes for an address or prefix appearing in a route table with different announcing routers may be a fault if unintentional, meaning it is a duplicate unicast address assignment. Alternatively, it may be the intended configuration if the address or prefix routes are Informal Anycast

routes i.e., the address or prefix from within the unicast address space is being used as an anycast address or anycast prefix. The duplicate routes and the addresses or prefixes themselves provide no indication of whether the configuration is intentional or not.

- o Constraining the visibility and reachability of an anycast provided service or function may be useful for security reasons, which can be fundamentally enforced by encoding and limiting the scope of or domain where packets are intended and able to be forwarded. Informal Anycast addresses can only have one of three fundamental forwarding scopes encoded in the address, matching those of the three types of IPv6 unicast addresses - limited to a link scope (Link-Local Address), limited to a local network scope (Unique Local Address) or having global Internet scope (Global Unicast Address) [RFC4291][RFC4193]. These scopes are coarse. More fine grained scopes, such as those used in IPv6 multicast [RFC7346], would provide much more control over anycast service or function visibility.
- o Some transport layer protocols, such as SCTP [RFC3286] and Multipath TCP [RFC6824], and some applications, deal directly with IPv6 addresses, and supply them to their communications peer or peers. Currently, if these transport layer protocols or applications are dealing with both unicast and anycast addresses, and wish to provide only unicast or anycast types of address or set of addresses to their peer or peers during their communications transactions, it would be necessary to manually configure the transport protocol or application so that it can distinguish between unicast and anycast addresses. A well known address class that identifies anycast addresses would allow these transport layer protocols or applications to identify the different address types without any manual configuration.

4. Formal Anycast Addresses

4.1. Address Format

The following diagram shows the structure of an IPv6 Formal Anycast address.

DIAGRAM TO COME

4.2. Address Fields

4.2.1. Formal Anycast Prefix

A 8 bit prefix value of TBD (aa00::/8 preferred, indicating a Anycast Address; fa00::/8 an alternative, indicating a Formal Anycast address), identifying this address as a Formal Anycast address.

4.2.2. Visible Scope

A 4 bit Visible Scope field used to express and enforce visibility and assumed reachability of the Formal Anycast address. The values and meanings of the values of this field are the same as those for the IPv6 multicast address scope field [RFC4291][RFC7346].

When packets with a Formal Anycast destination address are being forwarded, this field's value takes precedence over a non-zero Hop Limit field value, meaning the packet MUST be discarded at the edge of the indicated visibility domain even though it may have a non-zero Hop Limit value. A specific ICMPv6 Destination Unreachable [RFC4443] message, described below, SHOULD be generated and returned to the packet's sender indicating the packet was discarded as it reached the edge of its Visible Scope.

4.2.3. Anycast Identifier Format

A 4 bit field identifying the format of the following Anycast Identifier field, holding digits in the range of 0x0 through 0xf in hexadecimal.

The first assigned value is 0, specifying that the following Anycast Identifier Format is that of a Functional Anycast addresses, which is described later in this memo.

Other values will be assigned by IANA as future Anycast Identifier Formats are specified.

4.2.4. Anycast Identifier

A 112 bit field holding the Anycast Identifier value. The format and structure of this field is encoded in the previous Anycast Identifier Format field.

4.3. Anycast Address Registration Protocol

Rather than having to manually configure a network's routing protocol to distribute a host's anycast address, or have a host participate in the network's routing protocol, a protocol for hosts to automatically register an anycast address for routing protocol distribution would be beneficial.

Development of this protocol is left for a later memo, however as the requirements of such an anycast address registration protocol are very similar to that of hosts' multicast address registration with a network, it is likely that an anycast address registration protocol could be modelled on and derived from the IPv6 Multicast Listener Discovery protocol [RFC2710][RFC3810].

4.4. Network Service Provider Visible Scope

A network service provider, such as an Internet Service Provider, may wish to use an anycast address to provide a service with a visibility limited to all of its direct customers.

When using a Formal Anycast address for this service, that reuses IPv6 multicast scopes, this means the address needs to have a scope that is greater than the Organization-Local scope, yet smaller than the unlimited Global scope.

This memo defines a new scope called the Network Service Provider (NSP) scope, that falls between the Organization-Local and Global IPv6 multicast scopes. This NSP scope's hexadecimal value is B. This scope can be used with both Formal Anycast and IPv6 multicast addresses.

(MRS: perhaps this scope shouldn't be at B, but instead hard up against the Organization-Local scope i.e. at value 9? Could there be a need for any other future scopes between Organization-Local and Network Service Provider - which would be scopes within the Network Service Provider's network.)

4.5. Link-Local Visible Scope

One of the possible Visible Scope values is the Link-Local scope, specifying that the Formal Anycast address's visibility is limited to a link that the host is directly attached to.

Nodes on the link will need to consider Formal Anycast addresses with a Link-Local Visible Scope on-link, so that they perform Neighbor Discovery [RFC4861] for these addresses.

Similar to the unicast Link-Local prefix [RFC5942], IPv6 implementations SHOULD BE updated to consider the Formal Anycast prefix with a Link-Local Visible Scope on-link. Using the (preferred, IANA TBA) aa00::/8 Formal Anycast prefix, this means IPv6 implementations will consider aa20::/12 to be on-link.

Unlike the unicast Link-Local prefix, updated IPv6 implementations MUST NOT use SLAAC [RFC4862] to generate an automatic address from within this Formal Anycast Link-Local Visible Scope prefix.

(MRS: Need to further consider this constraint, and more generally the idea of host automatically generated dynamic anycast addresses, rather than either having well known or system administrator chosen and configured anycast addresses. If there is an anycast address registration protocol that hosts can use (suggested above), then hosts could possibly dynamically generate anycast addresses and then register them.)

Note that unlike the unicast Link-Local prefix, IPv6 nodes may not and typically would not have an address from within the Formal Anycast Link-Local Visible Scope prefix. One of the node's Link-Local addresses on the same link should be used as a source address when sending to a Formal Anycast Link-Local Visible Scope destination. This does not preclude using other greater scope unicast source addresses.

In the interrim IPv6 implementation update period, IPv6 nodes can be informed that the Formal Anycast Link-Local Visible Scope prefix is on-link in one of two ways:

- o A manually configured entry in the host's Prefix List [RFC4861].
- o A dynamic update to nodes' Prefix Lists using a Router Advertisement Prefix Information Option (PIO) [RFC4861] for the Formal Anycast Link-Local Visible Scope prefix of aa20::/12, with the 'L' or on-link flag set to on, and the 'A' or autonomous address-configuration flag set to off (as this prefix MUST NOT be used by the node to automatically generate an address for its use within this prefix). The Valid and Preferred Lifetimes for the Formal Anycast Link-Local Visible Scope prefix in the PIO are set to infinity.

4.6. ICMPv6 Destination Unreachable Message

As mentioned previously, if a packet with a Formal Anycast destination address reaches the edge of the Visible Scope for the address, a ICMPv6 Destination Unreachable [RFC4443] message SHOULD be generated and sent back to trigger packet's sender.

Note that if the router at the edge of the visibility domain is also assigned the Formal Anycast address, the packet is host processed locally rather than being discarded, and an ICMPv6 Destination Unreachable message IS NOT generated.

When a router implementation formally supports Formal Anycast addresses, the ICMPv6 Code for the Destination Unreachable message is IANA-TBD, indicating that the Edge of the Visible Scope [was] Reached.

If a router implementation does not formally support Formal Anycast addresses an operator should use packet filters to enforce the Visible Scope boundary. A packet failing to pass the packet filter should cause the router to generate a Destination Unreachable Communication with destination administratively prohibited message [RFC4443] (Code 1) message, which is semantically similar to the formal Edge of Visible Scope Reached message.

Note that ICMPv6 messages are not sent reliably, so Formal Anycast packet senders will need to be able to handle not receiving a ICMPv6 Destination Unreachable message in response to a packet reaching the edge of the visibility domain.

There may be situations where silently discarding Formal Anycast packets at the Visible Scope boundary may be preferred. In this case, a packet discard route, covering the Visible Scope prefix can be installed in a router's forwarding table, saving router control plane resources.

4.7. Default Address Selection

4.7.1. Formal Anycast Scope Comparison

As the Formal Anycast address scopes are defined to be the same as Multicast address scopes, the same Multicast scope comparison methods, described in [RFC6724], are also used with Formal Anycast address scopes.

4.7.2. Source Address Selection

As mentioned in Appendix B. of [RFC6724], anycast addresses are candidates during source address selection.

4.7.3. Destination Address Selection

An IPv6 implementation may be presented with a candidate set of destination addresses that consists of both Formal Anycast and unicast addresses. The implementation needs to make a choice or choices as to which of these candidate addresses to attempt to use.

The decision to use a Formal Anycast address instead of a unicast address by the destination is an active and conscious one. Therefore, when a choice needs to be made between a Formal Anycast

address and a unicast address, the Formal Anycast address should be preferred.

In terms of the Destination Address Selection algorithm described in [RFC6724], this preference of Formal Anycast over unicast addresses introduces a new rule between Rule 1 ("Avoid unusable destinations") and Rule 2 ("Prefer matching scope"), specifically (using "1.5" here to indicate the position):

Rule 1.5: Prefer Formal Anycast addresses.

If DA is a unicast address, and DB is a Formal Anycast address, then prefer DB.

Note that there may be instances where an application would prefer to use a unicast address over a Formal Anycast address. In this case, Formal Anycast addresses can be easily identified and ignored using the well known 8 bit Formal Anycast prefix.

4.8. Non-Local Anycast Forwarding

(MRS: This section is being left here for the moment, however this idea should probably be moved to a different memo, as it is describing forwarding that isn't unicast forwarding (unicast forwarding is used by conventional anycast))

One possible use for Formal Anycast addresses is to represent a function that is performed on the packet by the network that is beyond conventional destination address based unicast IPv6 forwarding, as the packet traverses the path towards its final delivery point.

Currently, hop-by-hop processing of an IPv6 packet as it traverses the network is indicated using the Hop-by-Hop Options (Extension) Header [RFC8200].

Drawbacks of using the Hop-by-Hop Option Header are that some high speed routers ignore them [RFC7045], and that packets with the Hop-by-Hop Options Header may be dropped by transit Autonomous System (AS) networks [RFC7872]. The dropping of these types of packets by transit ASes may be due to a default deny policy for Extension Header types other than those of TCP, UDP, ICMPv6 and possibly IPsec ESP.

Encoding the intent of hop-by-hop processing of a packet as an anycast IPv6 destination address has the advantage of the packet always being processed by all router implementations, including high speed implementations, as processing a packet's IPv6 destination address is required to perform IPv6 destination address based

forwarding. As there is no explicit Hop-by-Hop Options Header in the packet, a transit AS is less likely to drop the packet, unless it explicitly implements IPv6 Destination Address packet filters that drop packets with Formal Anycast addresses.

Another advantage is that there may now be no need for the addition of an Extension Header to the IPv6 packet for hop-by-hop processing, increasing the packet's effective payload size.

Conventional unicast IPv6 forwarding based on destination address prioritises a node's local addresses over all others. This means that when a node originates a packet with one of its own addresses as the destination, the node will deliver the packet internally for local processing, rather than sending it out one of the node's network interfaces.

The functional requirements for Non-Local Anycast Forwarding are:

- o When being sent by the node, the packet is not delivered internally to the node's own instance of the Formal Anycast address.
- o After being submitted to the network for forwarding, the packet must not be sent back towards its source, as this would potentially cause the packet to follow a loop in its forwarding path.

4.9. Advice on Structuring the Anycast Identifier Field Values

The Anycast Identifier Format field within a Formal Anycast address specifies the format and structure of the 112 bit Anycast Identifier field. The following is advice and guidelines to use when developing a new Anycast Identifier field format and structure.

Forwarding towards anycast addresses is the same as forwarding towards unicast addresses, which uses the longest match rule BCP 198 [RFC7608]. Longest match forwarding facilitates summarisation of forwarding information, where a single more general forwarding route can summarise a number of more specific forwarding routes. Summarisation saves entries in forwarding tables outside of the summarised forwarding domain, provides simpler destination based filtering for security purposes, and facilitates easier destination address based traffic analysis.

The use of route summarisation with anycast addresses effectively creates an anycast domain that is being identified and summarised by the anycast summary route. Outside of the anycast domain, a single summary route exists, covering all anycast addresses within the

domain. Within the anycast domain, individual routes for individual anycast addresses exist.

When designing a new Anycast Identifier field format and structure, the following guidelines should be followed. These guidelines should allow a set of more specific anycast routes to be summarised as well as improving operator usability.

- o The order of fields within the Anycast Identifier field should be from the most general to most specific, in the direction following the high order to low order bits of the Anycast Identifier field and the broader IPv6 address.
- o The order of bits within fields should also be from the most general to most specific, matching the direction of high order to low order of bits within the Anycast Identifier field.
- o The bits in fields holding bits that are matched exactly, such as flag fields or fields holding numeric values that are matched exactly, can be ordered to suit the field's use and application. However, a hierarchical order, from most significant to least significant bit, following Anycast Identifier field bit order, is suggested. Although initially defined hierarchially, the order of flags in flag fields may later deviate from this recommendation due to later flag bit definition.
- o End-users of the functions and services being provided using Formal Anycast addresses are unlikely and ideally should never see these addresses. However, operators of these functions and services will deal with these addresses during planning, configuration and troubleshooting. Where possible, fields and their values should be ordered and structured to assist with these tasks. Field boundaries within the Anycast Identifier field should align with 16 bit word, 8 bit octet or 4 bit nibble positions within the whole IPv6 address. For example, if part of an IPv6 prefix is included in the Anycast Identifier, it should start at a 16 bit "piece" boundary, where colons appear [RFC4291], within the IPv6 address. Note that this guideline should not take precedence over any previous measures to facilitate more specific anycast route summarisation.
- o A further address usability recommendation is to set field and bit values to zero for the likely most common or likely most secure meaning for these fields or bit values. In IPv6 addresses zero field values can be compressed [RFC5952], resulting in a shorter address for an operator to type. A shorter address to enter naturally reduces the opportunities for and likelihood of errors

in the address, and reduces the possibilities of security issues caused by errors in the Formal Anycast address.

5. Functional Anycast Addresses

The first defined sub-class or sub-format of Formal Anycast addresses is the Functional Anycast address sub-class.

5.1. Features

The following are the features of Functional Anycast addresses. In many cases they're inspired by and mirror IPv6 multicast address features [RFC4291][RFC3306].

- o Provides separate globally well known and local network defined anycast function or service identifier spaces. Globally well known identifiers can be encoded in applications during their development as constants, avoiding the need for them to be specified and configured during deployment of the application.
- o Provides a minimum of 24 bits for use as identifiers for anycast functions or services, supporting more than 16 million values.
- o Provides 8 bits for the identification of up to 256 local instances, versions or revisions of the same function or service, assisting with function or service deployment or maintenance. These 8 bits can also be used to increase the size of the function or service identifier space to 32 bits where useful, increasing the range of values to more than 4 billion.
- o Identifies an anycast function or service identifier space, known as an anycast domain, using an IPv6 unicast address prefix of up to 64 bits in length.

A network can create multiple distinct anycast domains by using multiple of its IPv6 prefixes, from its Global [RFC4291] or Unique Local [RFC4193] address spaces (the Link-Local prefix could be used to create a distinct anycast domain, however it can only be used once, despite the network having many instances of the Link-Local prefix (as many as it has links)).

A "unspecified" anycast domain is supported using an all zeros 64 bit IPv6 prefix.

External to the anycast domain, the identifying 64 bit prefix can be used to create a single summary route for the anycast function or service identifier space, which will help routing scaling for anycast functions or services. The anycast domain boundary could

also correspond to routing protocol scaling boundaries, such as OSPFv3 areas [RFC5340] or IS-IS level [RFC5308] boundaries, when useful.

5.2. Address Format

The format of Functional Anycast addresses is modelled on the IPv6 multicast address format [RFC4291].

The format of an Functional Anycast address is as follows:

DIAGRAM TO COME

The address fields are as follows:

- o Anycast Domain Prefix – a 64 bit field holding a IPv6 unicast address prefix identifying the anycast domain that is either the provider and possible authority for the following Anycast Function Identifier space.

The length of the prefix is specified in the following Prefix-Length field, with the remaining bits of the field set to zero.

An all zero Anycast Domain Prefix means an unspecified Anycast Function Identifier provider. An all zeros Prefix in effect means "this" provider, with "this" meaning the current anycast domain.

Link-Local, Unique-Local [RFC4193], Global and possible future other unicast prefixes [RFC4291] identify a specific Anycast Function Identifier provider (MRS: Need to think about more about using Link-Local prefix, as it really isn't specific – perhaps either prohibit, or make it all zeros "current" equivalent). Within an anycast function domain, this allows multiple anycast function sub-domains to be created, identified by different unicast prefixes in this field.

- o Reserved – a 2 bit reserved field.

Set to zero upon transmission, ignored upon receipt.

- o Prefix-Length – An 6 bit field specifying the length of the previous Anycast Domain Prefix.

A value of zero means a 64 bit length prefix, while prefix lengths of 1 through 63 (0x01 through 0x3f) are encoded natively.

The unspecified Anycast Domain Prefix of all zeros is considered to be 64 bits in length, meaning a Prefix-Length value of 0 for this prefix.

This is an informational field to assist with operation and troubleshooting.

(MRS: This field is inspired by the equivalent field when IPv6 multicast addresses contain a unicast prefix per [RFC3306]. I'm not entirely sure it is necessary, as we don't embed prefix length in unicast addresses, and unicast routing protocols, also used for anycast, carry prefix length information separately.)

- o Flags - A 8 bit flag field.

The lower 7 flags are reserved and must be set to zero upon transmission, and ignored upon receipt.

The high order flag is the 'T' or Transient flag. T=0 indicates that the later Anycast Function Identifier is well known and assigned by the Internet Assigned Numbers Authority (IANA). T=1 indicates that the Anycast Function Identifier is transient or dynamically assigned, and has been assigned by the Functional Anycast domain's local authority.

(MRS: is there a way to avoid this flag field? Some of the flags that are used in IPv6 multicast addresses could instead be accommodated by defining an entirely new 112 bit Anycast Identifier Format. Avoiding this field and the Prefix-Length field frees up 16 bits which makes this Anycast Identifier Format simpler and would provide 16 more bits for the following fields, which may be useful.)

- o Local Instance - An 8 bit field holding a identifier of the instance, version or revision of the function identified by the following Anycast Function Identifier field, local to the current anycast domain.

The default value of this field is zero, indicating the default and first instance of the anycast function.

Non-default values are chosen by the local anycast domain operator, even when the following Anycast Function Identifier is using a well-known IANA Anycast Function Identifier value.

An anycast domain operator may choose to assign other semantics to this field, as long as they're both less significant than the

previous fields and more significant than the following Anycast Function Identifier field.

When the 'T' bit in the Flags field is set to 1, meaning transient Anycast Function Identifiers, this field could be used to effectively increase the size of the following Anycast Function Identifier field to 32 bits, increasing the value range of Anycast Function Identifiers from in the order of 16 million to in the order of 4 billion.

- o Anycast Function Identifier - A 24 bit field identifying the anycast function to be performed on the packet when it arrives at a host that has been configured with the Functional Anycast address.

When the 'T' bit in the Flags field is set to zero, the Anycast Function Identifiers values are from a well known Anycast Function Identifier registry maintained by IANA, with initial entries specified later in this memo.

When the 'T' bit in the Flags field is set to 1, the values in the Anycast Function Identifier field are local to and assigned by the authority identified in the Prefix field in any manner that suits their purposes and requirements.

5.3. Assignment of Anycast Function Identifiers

In the history of the Internet, it has been common to conflate a function or service with a protocol.

For example, historically, the TELNET protocol [RFC0854] had been the most popular "Network Virtual Terminal" protocol. In more recent times, the SSH protocol [RFC4252] has become the de facto network virtual terminal protocol. Accessing the network virtual terminal service has either been referred to as "TELNETting in" or "SSHing in" to the host providing the service, using the protocol being used to refer to the service being accessed.

In either case, TELNET and SSH protocols are being used to access a remote network virtual terminal service. Functionally, from the perspective of network virtual terminal access, the differences are relatively minor; data security and integrity via encryption and authentication is where the primary differences between TELNET and SSH are - in TELNET authentication [RFC2941] and encryption [RFC2946] of the data stream is optional, where as with SSH it is mandatory.

When both IANA and local anycast domain operators assign Anycast Function Identifiers, it is recommended that they're allocated and

identified by protocol agnostic function or service type rather than to a specific protocol that provides that function or service. As the particular protocol being used to access the function or service will be encoded in the upper transport layer protocols and ports in the IPv6 packet, service or function based Anycast Function Identifiers can support and stay constant across the use and evolution of different function or service access protocols.

For example, with a well-known Anycast Function Identifier specifically allocated to a Network Virtual Terminal (NVT) [RFC0318] service, the hosts providing the NVT service could initially support both TELNET (assuming TELNET is considered secure enough) and SSH. If both TELNET and SSH become deprecated, and a new NVT access protocol is developed, the same Anycast Function Identifier for the NVT service could be used to reach a node supporting this new access protocol.

Another example is the domain name service. Currently domain name resolution commonly takes place using the Domain Name Service protocol [RFC1035], carried directly over UDP and TCP, using port 53. More recently, work has been taking place to operate DNS over TLS [RFC7858] and HTTPS [RFC8484] to enhance the security of the domain name resolution function. The use of multiple protocols to access fundamentally the same domain name resolution function suggests a protocol agnostic domain name resolution Anycast Function Identifier.

This doesn't preclude Anycast Function Identifiers being used to support and identify specific protocols (examples of this occur later). There may be current and future cases where the allocation and use of an Anycast Function Identifier for a specific protocol is the better choice. This should be considered and evaluated on a case by case basis.

5.4. Assigned Anycast Function Identifiers

A number of past RFCs have reserved anycast addresses and identifiers. These addresses and identifiers are mapped to the following corresponding and well known Anycast Function Identifiers, and are to be listed in the IANA Anycast Function Address Identifier registry if it is created.

[RFC2526] reserves the highest 127 values of a subnet prefix Interface Identifier for anycast addresses. The equivalent values for Functional Anycast addresses are the highest 127 values of the 32 bit Anycast Function Identifier, a range of (in IPv6 address format, excluding the high order 96 bits) :ffff:fff8 through :ffff:ffff. The IANA Internet Protocol Version 6 (IPv6) Anycast Addresses registry [IANA-IPV6ANYC] records assignments for subnet prefix anycast

addresses within the Interface Identifier space. The current and future values of these anycast subnet prefix Interface Identifier values are to also be recorded in the Anycast Function Address Identifier registry.

[RFC4291] reserves an Interface Identifier of all zeros within a unicast prefix as the Subnet-Router anycast address. The equivalent 32 bit Anycast Function Identifier value for Functional Anycast addresses is also all-zeros.

[RFC7723] reserves the IPv6 address 2001:1::1/128 for the use as the Port Control Protocol Anycast address. The equivalent 32 bit Anycast Function Identifier value is (in IPv6 address format, excluding the high order 96 bits) :0001:0001.

[RFC8155] reserves the IPv6 address 2001:1::2/128 for the use as the Traversal Using Relays around NAT Anycast address. The equivalent 32 bit Anycast Function Identifier value is (in IPv6 address format, excluding the high order 96 bits) :0001:0002.

5.5. Sources of Inspiration for Anycast Function Identifiers

A future possible source of inspiration for well known assigned Anycast Function Identifiers could be DHCPv6 [RFC8415] options that encode IPv6 addresses for services.

A number of these options encode multiple IPv6 addresses as candidates for access to the service (for example, the SIP Servers IPv6 Address List option [RFC3310]). The use of anycast for service resilience would allow a single Anycast Function Identifier value to provide equivalent service, although this wouldn't preclude defining multiple different Anycast Function Identifiers to the service to provide the service client concurrent access to multiple service instances. For example, 3 Functional Anycast addresses could be allocated for DNS resolvers, providing a client with separately verifiable DNS resolver services from up to 3 different resolvers, and allowing the client to distribute requests across all 3 resolvers.

Another source of inspiration for well known assigned Anycast Function Identifiers could be the IANA IPv6 Multicast Address Space [IANA-IPV6MCAST] registry, where some of the multicast addresses represent services that could also be useful when provided via anycast.

While using these possible source of inspiration, the recommendation to choose protocol agnostic function or service identifiers still

stands. DHCPv6 or multicast groups can be used to inspire more generic function or service identifiers.

5.6. Global Scope Functional Anycast Addresses on the Internet

(MRS: Need to fully review this idea and section. An idea to help overcome the /48 prefix length constraint would be to have all Formal Anycast addresses that are going to be used on the Internet come from an IANA reserved prefix within the existing GUA address space e.g. 20e0::/12 (i.e. operators would accept a prefix announcement of length of up to /80 within 20e0::/12). aa::/8 would still be used for smaller than global scope anycast, because a prefix of "aa" is very helpful to identify anycast addresses.)

Functional Anycast addresses could be used to provide anycast services across the Internet, by using the the Global scope.

When being used on the Internet, many of the possible values of the Prefix field are ambiguous, meaning that they wouldn't unambiguously identify the party using the Functional Anycast address to provide the service or function. Examples of ambiguous prefixes are the all-zeros unspecified prefix, any ULA [RFC4193] prefixes, and the Link-Local [RFC4291] prefix. Other ambiguous prefixes are those in the IPv6 reserved address registry [IANA-IPV6RESA] that are not valid on the Internet.

To overcome this ambiguity, if Global scope Functional Addresses are used over the Internet, the Prefix field MUST be set to a GUA [RFC4291] prefix value assigned to the party providing the anycast service to Internet clients. A network either accepting or originating a Global scope Functional Address prefix for announcement from a downstream stub autonomous system for announcement onto the Internet MUST only accept or originate a route announcement for a Global scope Functional Anycast prefix that includes an explicitly identified GUA prefix. All other Global scope Functional Anycast prefix announcements with ambiguous or non-explicitly identified GUA prefixes MUST be ignored.

As forwarding towards anycast addresses is functionally the same as forwarding towards unicast addresses, Functional Anycast prefixes would be announced into the Internet's unicast forwarding route table. These Functional Anycast prefixes SHOULD BE aggregate announcements with the aggregation boundary occurring directly after the Anycast Domain Prefix.

It is common practice today to limit the prefix length of unicast IPv6 Internet routes accepted to a length of no more than 48 bits i.e. a /48. This is a blunt and simple way to attempt to somewhat

limit the number of IPv6 routes in the Internet route table. It is imposing this limit by enforcing a minimal level of aggregation at the /48 boundary. Networks using prefix lengths longer than /48 are expected to aggregate those networks into a route advertisement that is /48 or shorter.

This practice of limiting advertised unicast route prefix lengths to 48 bits will limit the size of the Prefix included in Global scope Functional Anycast announcements to 32 bits, as the high order 16 bits of the Functional Anycast prefix are used to encode the Functional Address type prefix and address scope. This would limit the use of Global scope Functional Anycast addresses to provide global Internet anycast services to those organisations who have a /32 or shorter assignment from an RIR.

As Functional Anycast addresses are a separate class of addresses, and are all identified by a unique /8 prefix, this /48 prefix length limit could be specifically relaxed for Functional Anycast routes. A /48 prefix, when included in a Functional Anycast, results in a Functional Anycast prefix length of /64. Imposing a /64 prefix length limit on Functional Anycast routes, identified by a high order prefix of aae0::/16, and a GUA anycast domain prefix, would achieve the same outcome of attempting to reducing the number of entries in the IPv6 Internet route table.

Wide acceptance of Functional Anycast prefixes of up to 64 bits in length on the Internet may take same time to occur. Use of Global scope Functional Anycast addresses by organisations who have RIR /32 assignments, which will be accepted by unicast /48 prefix filters present today, would raise awareness of Functional Anycast addresses. This increased awareness could be leveraged to motivate the changing of prefix filters to accept Global scope Functional Anycast prefixes up to 64 bits in length.

5.7. Example Use Cases

This section provides some example use cases of Functional Anycast addresses that would suit the described scenarios.

5.7.1. Devices Factory Configured with NTP Functional Anycast Addresses

Assume IANA has allocated a set of well-known Anycast Function Identifier values of 0x004440, 0x004441, 0x004442 and 0x004443 for use with the Network Time Protocol [RFC5905], to facilitate meeting the NTP best practice of having a minimum of 4 NTP time sources [RFC8633].

A device manufacturer uses this set of well-known Anycast Function Identifier to set factory default Functional Anycast addresses for access to a device customer's NTP servers.

The corresponding Functional Anycast address is constructed as follows:

- o 8 bit Formal Anycast Prefix value (0xaa proposed).
- o 4 bit Visible Scope value of 0x8, corresponding to Organization-Local [RFC7346], as the manufacturer is unlikely to have any knowledge of device customers' use of or preference for smaller scopes.
- o 4 bit Anycast Identifier Format value of 0x0, corresponding to the Functional Anycast format.
- o 112 bit Anycast Identifier in the Functional Anycast format:
 - o
 - * 64 bit Anycast Domain Prefix value of the all zeros unspecified prefix.
 - * 2 bit reserved field set to zero.
 - * 6 bit Prefix-Length field set to zero, meaning a 64 bit length Anycast Domain Prefix value.
 - * 8 bit Flags field set to all zeros. The upper 7 bits are zero as they're served, while the lowest 'T' or Transcient flag is set to zero indicating an IANA assigned well known Anycast Function Identifier.
 - * 8 bit Local Instance flag set to the default value of zero.
 - * 24 bit Anycast Function Identifier field set to either 0x004440, 0x004441, 0x00442 or 0x004443; one of the IANA assigned well known Anycast Function Identifier values for the NTP protocol.

All of the above mean that the NTP server Functional Anycast addresses the device manufacturer sets as the defaults would be (in IPv6 address compressed format):

- o aa80::4440
- o aa80::4441

- o aa80::4442
- o aa80::4443

5.7.2. Branch Office DNS Resolvers

An enterprise network operator decides to use Functional Anycast addresses to provide DNS resolver service to end-user devices, located in various corporate offices.

Specifically for a single mid-sized branch office, with in the order of 200 staff, the operator decides to provide two DNS resolvers located in the office. This will provide lower latency DNS resolution through DNS caching, reducing perceivable application response time [NORMNEIL]. Access to a third, geographically close, off-site DNS resolver is provided for redundancy. This off-site DNS resolver will be one of the other branch office's local DNS resolvers.

All three DNS resolvers will provide their services to clients via Functional Anycast addresses. Different clients will receive the on-site DNS resolver addresses in alternating order, both before the off-site DNS resolver address. This provides rudimentary on-site DNS resolver load balancing and keeps both DNS resolvers' lookup caches populated to reduce the client visible performance impact of the fail-over to the remaining on-site DNS resolver, should its sibling fail. The on-site DNS resolvers will watch each others' availability, taking over its sibling's Functional Anycast address if the sibling becomes unavailable. Should both on-site DNS resolvers become unavailable, clients will resort to using the remaining off-site DNS resolver.

Assume IANA have allocated the well known Anycast Function Identifier values of 5300, 5301 and 5302 for use with anycast DNS resolvers.

The operator allocates decimal identifiers of 703, 9556, 4739, 38809 and 2764 to the corporate offices, with the order reflecting geographic proximity. Each office will have its own unicast /48 from within a globally unique address space of 2001:db8::/32, meaning that the office prefixes are 2001:db8:2bf::/48, 2001:db8:2554::/48, 2001:db8:9779::/48 and 2001:db8:acc::/48.

The corresponding Functional Anycast address is constructed as follows:

- o

The Visibility Scope for these DNS resolvers' Functional Anycast addresses will be Organization-Local (8).

For office 9556, using office 4739 as an off-site backup, the Functional Anycast addresses for the three DNS resolvers will be:

1. aa80:2001:db8:2554::5301
2. aa80:2001:db8:2554::5302
3. aa80:2001:db8:9779::5303

5.7.3. Automatic eBGP Session Establishment

Assume IANA has allocated a well-known Anycast Function Identifier value of 0x000179 for use with automatic eBGP [RFC4271] session establishment.

Smaller, stub site routers are preconfigured with a Functional Anycast address to attempt to automatically establish an eBGP session with a one or two upstream eBGP peer aggregation routers over one or two different designated ("WAN") links upon initialisation.

The eBGP Functional Anycast address would be a Link-Local Visible scope address. The stub router would use its link's, SLAAC generated [RFC4861] and link unique Link-Local address [RFC4291] as the source address to reach the eBGP Functional Anycast address. Using Link-Local scope Formal Anycast and unicast addresses for this eBGP session would provide a basic level of eBGP access security.

The stub site router will need to also be preconfigured or somehow automatically generate an Autonomous System Number (ASN) to use for establishing the eBGP session or sessions. How the ASN is preconfigured or generated is out of scope for this memo, and is left to future work.

Once the eBGP session is established, the peer eBGP routers trade routes. These traded routes could include the upstream eBGP providing a default route or other more specific routes, and the downstream stub site router providing a route to its downstream prefix or prefixes.

The downstream prefix or prefixes could be those the stub site router has learned via DHCPv6 Prefix Delegation (DHCPv6-PD) [RFC3633]. An advantage of having the stub site router inject DHCPv6-PD prefixes into the BGP routing domain is that the route information for this or these prefixes within the BGP routing domain would more accurately reflect the state and therefore the availability of the prefixes at

the site they've been assigned and are being used it. Stub site routers announcing their own prefixes would also distribute the announcement processing load across the stub site routers rather than concentrating it at the upstream aggregation router(s). This also avoids the upstream aggregation router having to process the DHCPv6-PD response to determine the assigned delegated prefix for subsequent BGP announcement [RFC3633], meaning it can act as a much simpler and pure DHCPv6 relay [RFC8415].

The upstream link(s) that a stub site is attached to does not have to be limited to being a true link-layer point-to-point link, meaning that the link only supports a single router pair of a stub and upstream aggregation router. The link could be a multi-access link, with the single link supporting many stub site routers and a number of upstream aggregation routers.

As the eBGP Functional Anycast address is a Link-Local Visible Scope address, the address is configured as an anycast address on the upstream aggregation routers' stub site facing network interface. This results in the receiver of the Neighbor Advertisements for this address using the information received in the first received Neighbor Advertisement to update its neighbor cache, rather than the last and most recently received Neighbor Advertisement. These types of Neighbor Advertisements are known as "Anycast Neighbor Advertisements" in [RFC4861].

[RFC4861] says that Anycast Neighbor Advertisements should be delayed a random amount of time between 0 and MAX_ANYCAST_DELAY_TIME, a variable with a default value of 1 second. This random delay is to reduce the probability of loss of the Neighbor Advertisement due to network congestion.

Specific to this eBGP use case, the Anycast Neighbor Advertisements delay could include other metrics in the calculation to more intelligently distribute the eBGP sessions across the set of upstream aggregation routers. For example, the number of existing eBGP sessions could be a metric, where an upstream aggregation router delays its Anycast Neighbor Advertisement longer when it has more established eBGP sessions.

An operator set router preference metric could be considered, allowing the operator to more gracefully phase out a legacy upstream aggregation router by setting its preference lower than the newer upstream aggregation routers. The operator would then manually terminate the eBGP sessions individually on the legacy upstream aggregation router, at a rate of something like one ever 3 seconds, causing them to be reestablished on the higher preference and newer upstream aggregation routers. This would be more graceful than

terminating all eBGP sessions at once on the legacy upstream aggregation router by, for example, switching it off.

Branch office stub router, automatically attempts to establish a BGP session with a well-known functional anycast address "out of the box" over the default WAN interface.

IANA assigned well-known BGP Anycast Function Identifier

Link-local scope Functional Anycast address. Provides a minimum level of security, as only possible to establish BGP sessions between direct link peers.

Use unspecified prefix (comment that fe80::/64 could be used, although unnecessary)

Well-known AS Number used by all stub routers. This makes the BGP sessions eBGP sessions. Routers will reject routes from other stub routers using the same ASN, however this is both fine and ideal as this is a stub router - default only plus announcing its local prefix(es).

Sub router acquires a delegated prefix via DHCPv6-PD

The delegated prefix is announced via the BGP session. Stops the upstream aggregation router needing to observe a DHCPv6 server's relayed response to then announce the delegated prefix into the network.

Upstream router accepts and establishes BGP sessions with any link-local address from the well known ASN, to the Functional Anycast BGP address.

There are potential trust issues here. BGPsec? Use the first BGP session to bootstrap connectivity to then establish a more trusted connection of some sort via PKI. Requirement for being link-local peers adds a minimal level of security and trust, but not much.

5.7.4. An ISP's Anycast DNS Resolvers

ISPs' IPv4 DNS resolvers have been the target of Distributed Denial of Service attacks (DDoS) [REF], with the attacks launched from the Internet.

These attacks have been possible because ISPs' have assigned their DNS resolvers public IPv4 addresses, with the public IPv4 addresses being both globally unique and globally reachable.

The need for global uniqueness comes from the requirement for the DNS resolvers to have an addresses that are not present within the ISP's customers' networks. Inherent with global uniqueness of a public IPv4 address is global reachability. To protect the DNS resolvers from DDoS attacks, an ISP has to actively configure protection mechanisms such as packet filters that discard traffic from the Internet, while continuing to allow the ISP's customers to use the DNS resolver.

There are two drawbacks of having to use purposely configured protection mechanisms such as packet filters once the DNS resolver has a publically unique and reachable address. Firstly, as the public IPv4 address is normally reachable, an error in configuring the packet filter means a "fail unsafe" consequence.

Secondly, packet filters can only be applied within the local network. This means that the large volume of DDoS traffic will reach the local network before it can be discarded. This discarded DDoS traffic consumes network capacity on paths towards the network that should instead be available to legitimate traffic towards the network.

Ideally, the ISP could assign its DNS resolvers addresses that should be unique within both the ISP's network and all of its customers' networks. The addresses should be reachable from the customers' networks while not being reachable from the Internet. Inherently, these customer and ISP only visible addresses would protect the DNS resolvers from Internet launched DDoS attacks. There would be no need for the ISP to configure packet filters to protect the DNS resolvers from the Internet, and as the DNS resolvers addresses are unreachable from the Internet, it would not be possible to send large volumes of DDoS traffic towards them. The ISP's Internet transit capacity would be more available for legitimate traffic.

Unique Local Addresses (ULA) [RFC4193] would appear to be addresses that could be used for this purpose. They are intended to be globally unique, due to their embedded 41 bit random number, meaning that they should not collide with any of the ISP's customers network's addresses. They are also not intended to be reachable globally across the Internet.

However, the ISP's ULA addresses assigned to its DNS resolvers would be unlikely be realiably reachable from all of its customers' networks. The IPv6 source address selection algorithm tries to pick source addresses that have high order prefix address bits that match those of the destination [RFC6724]. Consequently, to reach an ISP's ULA addressed DNS resolvers, customers' hosts would pick their ULA source addresses, should they have them. These packets may reach the

ISP's DNS resolvers, due to customers' default routes, however the DNS response return packets are unlikely to reach the customers' hosts as the ISP is unlikely to know customers' ULA routing prefixes, due to trading of ULA routing prefixes being prohibited by default [RFC4193].

So the source addresses that customers' hosts use when sending to the ISP's DNS resolvers need to be of greater scope and reachability than ULA addresses, while the ISP's DNS resolvers need to have addresses that have a greater scope and reachability than ULA addresses, yet are not IPv6 Globally Unique Addresses [RFC4291].

Customers' GUA addresses would meet this customer host source address requirement, while IPv6 Functional Anycast addresses with a Network Service Provider Visibility Scope would meet the ISP's DNS resolver address requirements.

5.7.5. Microservices Architecture Applications

(MRS: Any possible application here? Perhaps service redundancy and service anycast service addresses.

5.7.6. Global Time Distribution Network

We Have The Time Company (WHTT) are an enterprise who specialise in providing accurate time to global clients across the Internet, via the Network Time Protocol.

To provide robust time across the Internet, they provide access to their NTP servers via Functional Anycast addresses.

WHTT have a GUA /32 assignment from their Regional Internet Registry. They provide time to clients via the following Global scope Functional Anycast address, with a Global scope, an anycast domain prefix of 2001:db8::/32, a Prefix Length of 32 (0x20), and the well known NTP Anycast Function Identifier of 0x101.

- o aae0:2001:db8:0:0:2000::101

5.7.7. Multipath Transport Layer Protocols

Multipath transport layer protocols, such as MPTCP [RFC6824] and SCTP [RFC3286], establish a full multipath session between hosts in three stages, using multiple connections.

Stage one involves establishing an initial connection between the hosts. During stage two, the hosts' remaining set of IP addresses are exchanged. Finally, in stage three, the full multipath session

is established, with the hosts establishing further connections between the alternative IP addresses exchanged during stage two. Note that during the multipath session, any of the connections could fail or could be purposely torn down, and as long as at least one connection persists, the multipath session continues.

When multipath transport protocols are used for client-server applications, a single common IPv6 anycast address could be used by multiple servers, and then for the initial connection between the clients and servers during stage one.

During stage two, the server limits the set of alternative IP addresses it supplies to clients to its unicast addresses, excluding its one or more anycast addresses.

Subsequent connections that establish the full multipath session during stage three would then be limited to only being established between the unicast addresses of the clients and server.

Once any of these stage three unicast address connections is established, the server could actively tear down the initial connection to its anycast address, meaning that all of the now remaining connections are established with the individual server's unicast address or addresses.

Switching to using only unicast connections for the remainder of the multipath session overcomes one of the significant limitations of the use of anycast with connection oriented transport layer protocols [RFC1546]; the limitation that where the anycast address instance's packets are delivered to depends on the network's forwarding domains topology, and if the network topology changes, packets may be delivered to a different anycast instance that is unaware of and has not previously been involved in the transport layer connection.

With this use of both anycast and unicast addresses in combination with multipath transport protocols, the effects of this anycast limitation are reduced to the time between establishment of the initial client-server connection to the server's anycast address, and when the first client-server connection is established using exclusively unicast addresses. This is in contrast to this risk possibility existing for the entire duration of a single path transport layer protocol connection to an anycast address.

The benefit of the above use of anycast in combination with multipath transport layer protocols applies to all types of anycast addresses discussed in this memo.

There is a further advantage if Formal Anycast addresses are used. This is that as a Formal Anycast address is easily identified due to its well known prefix, the multipath transport layer protocol implementation does not have to be configured with which of the server's IPv6 addresses are its anycast addresses, so they can be excluded from the IPv6 address exchange that occurs during stage two.

6. Security Considerations

Functional Anycast addresses should not introduce any new security concerns in comparison to the use of addresses from within the unicast address space as anycast addresses. [RFC7094] provides considerable anycast related security discussion and references.

The ability to identify a Functional Anycast address using its well known 8 bit prefix, and the inclusion of forwarding scopes in the addresses, provide opportunities to enhance security of anycast services.

7. IANA Considerations

IANA are requested to register the aa00::/8 prefix in the Internet Protocol Version 6 Address Space registry for use with Formal Anycast addresses. If aa00::/8 is not chosen, then fa00::/8 is a proposed alternative.

IANA are requested to update the use of the IPv6 multicast scopes registry to also record use with Formal Anycast IPv6 addresses.

IANA are requested to record a new IPv6 multicast and Formal Anycast scope named the "Network Service Provider" scope, with a scope value of B in hexadecimal.

IANA are requested to register a new ICMPv6 Destination Unreachable code for Edge of Visible Scope Reached.

IANA are requested to establish a registry for the Flags field of Functional Anycast addresses, and to reserve the T flag to indicate transient Anycast Function Identifiers.

IANA are requested to establish a registry for well known Anycast Function Identifiers, and to reserve the values described previously in the "Assigned Anycast Function Identifiers" section of this memo.

8. Acknowledgements

Gavin Owen prompted the lunch time conversation where the author joined together and immediately recognised the benefits of using of anycast addresses in combination with multipath transport layer protocols to provide more robust anycast services.

Review and comments were provided by YOUR NAME HERE!

This memo was prepared using the xml2rfc tool.

9. Change Log [RFC Editor please remove]

draft-smith-6man-form-func-anycast-addresses-00, initial version, 2018-10-22

draft-smith-6man-form-func-anycast-addresses-01, updates, 2019-11-03

- o Fixed scope location error in addresses used throughout
- o Added section on the idea of an anycast address registration protocol
- o Reference updates

10. References

10.1. Normative References

- [RFC0854] Postel, J. and J. Reynolds, "Telnet Protocol Specification", STD 8, RFC 854, DOI 10.17487/RFC0854, May 1983, <<https://www.rfc-editor.org/info/rfc854>>.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [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>>.

10.2. Informative References

- [IANA-IPV6ANYC] "Internet Protocol Version 6 (IPv6) Anycast Addresses", <<https://www.iana.org/assignments/ipv6-anycast-addresses/ipv6-anycast-addresses.xhtml>>.

- [IANA-IPV6MCAST] "IPv6 Multicast Address Space Registry",
<<https://www.iana.org/assignments/ipv6-multicast-addresses/ipv6-multicast-addresses.xhtml>>.
- [IANA-IPV6RESA] "IPv6 Multicast Address Space Registry",
<<https://www.iana.org/assignments/ipv6-multicast-addresses/ipv6-multicast-addresses.xhtml>>.
- [RFC0318] Postel, J., "Telnet Protocols", RFC 318, DOI 10.17487/RFC0318, April 1972, <<https://www.rfc-editor.org/info/rfc318>>.
- [RFC1546] Partridge, C., Mendez, T., and W. Milliken, "Host Anycasting Service", RFC 1546, DOI 10.17487/RFC1546, November 1993, <<https://www.rfc-editor.org/info/rfc1546>>.
- [RFC2526] Johnson, D. and S. Deering, "Reserved IPv6 Subnet Anycast Addresses", RFC 2526, DOI 10.17487/RFC2526, March 1999, <<https://www.rfc-editor.org/info/rfc2526>>.
- [RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, DOI 10.17487/RFC2710, October 1999, <<https://www.rfc-editor.org/info/rfc2710>>.
- [RFC2941] Ts'o, T., Ed. and J. Altman, "Telnet Authentication Option", RFC 2941, DOI 10.17487/RFC2941, September 2000, <<https://www.rfc-editor.org/info/rfc2941>>.
- [RFC2946] Ts'o, T., "Telnet Data Encryption Option", RFC 2946, DOI 10.17487/RFC2946, September 2000, <<https://www.rfc-editor.org/info/rfc2946>>.
- [RFC3286] Ong, L. and J. Yoakum, "An Introduction to the Stream Control Transmission Protocol (SCTP)", RFC 3286, DOI 10.17487/RFC3286, May 2002, <<https://www.rfc-editor.org/info/rfc3286>>.
- [RFC3306] Haberman, B. and D. Thaler, "Unicast-Prefix-based IPv6 Multicast Addresses", RFC 3306, DOI 10.17487/RFC3306, August 2002, <<https://www.rfc-editor.org/info/rfc3306>>.
- [RFC3310] Niemi, A., Arkko, J., and V. Torvinen, "Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)", RFC 3310, DOI 10.17487/RFC3310, September 2002, <<https://www.rfc-editor.org/info/rfc3310>>.

- [RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, DOI 10.17487/RFC3633, December 2003, <<https://www.rfc-editor.org/info/rfc3633>>.
- [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>>.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <<https://www.rfc-editor.org/info/rfc4193>>.
- [RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <<https://www.rfc-editor.org/info/rfc4252>>.
- [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>>.
- [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>>.
- [RFC4786] Abley, J. and K. Lindqvist, "Operation of Anycast Services", BCP 126, RFC 4786, DOI 10.17487/RFC4786, December 2006, <<https://www.rfc-editor.org/info/rfc4786>>.
- [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>>.

- [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>>.
- [RFC5505] Aboba, B., Thaler, D., Andersson, L., and S. Cheshire, "Principles of Internet Host Configuration", RFC 5505, DOI 10.17487/RFC5505, May 2009, <<https://www.rfc-editor.org/info/rfc5505>>.
- [RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <<https://www.rfc-editor.org/info/rfc5905>>.
- [RFC5942] Singh, H., Beebe, 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>>.
- [RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <<https://www.rfc-editor.org/info/rfc5952>>.
- [RFC6724] Thaler, D., Ed., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, DOI 10.17487/RFC6724, September 2012, <<https://www.rfc-editor.org/info/rfc6724>>.
- [RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <<https://www.rfc-editor.org/info/rfc6824>>.
- [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>>.
- [RFC7094] McPherson, D., Oran, D., Thaler, D., and E. Osterweil, "Architectural Considerations of IP Anycast", RFC 7094, DOI 10.17487/RFC7094, January 2014, <<https://www.rfc-editor.org/info/rfc7094>>.

- [RFC7346] Droms, R., "IPv6 Multicast Address Scopes", RFC 7346, DOI 10.17487/RFC7346, August 2014, <<https://www.rfc-editor.org/info/rfc7346>>.
- [RFC7608] Boucadair, M., Petrescu, A., and F. Baker, "IPv6 Prefix Length Recommendation for Forwarding", BCP 198, RFC 7608, DOI 10.17487/RFC7608, July 2015, <<https://www.rfc-editor.org/info/rfc7608>>.
- [RFC7723] Kiesel, S. and R. Penno, "Port Control Protocol (PCP) Anycast Addresses", RFC 7723, DOI 10.17487/RFC7723, January 2016, <<https://www.rfc-editor.org/info/rfc7723>>.
- [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>>.
- [RFC7872] Gont, F., Linkova, J., Chown, T., and W. Liu, "Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World", RFC 7872, DOI 10.17487/RFC7872, June 2016, <<https://www.rfc-editor.org/info/rfc7872>>.
- [RFC8155] Patil, P., Reddy, T., and D. Wing, "Traversal Using Relays around NAT (TURN) Server Auto Discovery", RFC 8155, DOI 10.17487/RFC8155, April 2017, <<https://www.rfc-editor.org/info/rfc8155>>.
- [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>>.
- [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>>.
- [RFC8633] Reilly, D., Stenn, H., and D. Sibold, "Network Time Protocol Best Current Practices", BCP 223, RFC 8633, DOI 10.17487/RFC8633, July 2019, <<https://www.rfc-editor.org/info/rfc8633>>.

Author's Address

Mark Smith
PO BOX 521
HEIDELBERG, VIC 3084
AU

Email: markzzzsmith@gmail.com

Internet Engineering Task Force
Internet-Draft
Intended status: Best Current Practice
Expires: December 1, 2020

M. Smith
N. Kottapalli

R. Bonica
Juniper Networks
F. Gont
SI6 Networks
T. Herbert
Quantonium
May 30, 2020

In-Flight IPv6 Extension Header Insertion Considered Harmful
draft-smith-6man-in-flight-eh-insertion-harmful-02

Abstract

In the past few years, as well as currently, there have and are a number of proposals to insert IPv6 Extension Headers into existing IPv6 packets while in-flight. This contradicts explicit prohibition of this type of IPv6 packet processing in the IPv6 standard. This memo describes the possible failures that can occur with EH insertion, the harm they can cause, and the existing model that is and should continue to be used to add new information to an existing IPv6 and other packets.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 1, 2020.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
1.1. Requirements Language	3
2. Terminology	3
3. In-Flight Extension Header Insertion Defined	3
3.1. In-Flight Insertion	3
3.2. In-Flight Removal	4
3.3. In-Flight Insertion Without Removal	4
4. EH Removal Failure Causes	4
4.1. Implementation Bugs	4
4.2. Partial Node Failure	5
4.3. Operator Configuration Error	5
5. Single Point of Failure	5
6. MUST Remove is Aspirational	6
7. Harm	6
7.1. Violates RFC8200 and All Of Its Ancestors.	6
7.2. Ignores Source Address Field Semantics	6
7.3. Breaks ICMPv6	6
7.3.1. Breaks PMTUD	6
7.4. Breaks IPsec	6
7.5. May Cause Faults in Subsequent Transit Networks	6
7.6. Incorrect Destination Host Processing	6
7.7. Implementation Complexity	7
7.8. Costly Troubleshooting	8
8. Be conservative in what you send,	10
9. Solution: Encapsulation	10
9.1. IPv6 Tunnelling	11
9.2. MPLS	11
10. Reducing Tunneling Overhead	12
10.1. ROHC	12
10.2. Skinny IPv6-in-IPv6 Tunneling	12
11. In-Flight Insertion Considered Harmful	13

12. Security Considerations	13
13. Acknowledgements	13
14. Change Log [RFC Editor please remove]	13
15. References	13
15.1. Normative References	14
15.2. Informative References	14
Authors' Addresses	14

1. Introduction

1.1. Requirements Language

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

2. Terminology

- o In-Flight - the state of a packet while it is travelling through the network between its original source IPv6 and final destination IPv6 hosts. The packet will be being forwarded along a series of hops along a set of IPv6 routers interconnecting the source and destination IPv6 hosts.

3. In-Flight Extension Header Insertion Defined

3.1. In-Flight Insertion

At a point somewhere along the path an IPv6 [RFC8200] packet travels between the packet's source IPv6 host, identified in the packet's Source Address field, and the packet's final IPv6 destination host, identified in the packet's Destination Address field, the packet is split apart after the IPv6 fixed header and before the packet payload. Then, one or more new Extension Headers (EHs) [RFC8200] are inserted between those two existing packet parts. The new EH or EHs may be the sole EH or EHs in the packet after insertion, or it, or they, may be inserted at the start, within, or after the packet's set of original EHs.

Importantly, note that the packet's original source and Destination Address field values are left unchanged when EH insertion takes place. It is likely that other immutable fields of the IPv6 header are also left unchanged, with possible exception to the immutable Next Header field [RFC8200] if the inserted EH or EHs are inserted directly after the IPv6 fixed header.

For IPv6 tunnel packets [RFC2473], where there may be two or more instances of an IPv6 fixed header throughout the packet, EH insertion

could be occurring between any of the IPv6 fixed headers and their respective following payloads, although it is most likely to occur after the first of the IPv6 fixed header, commonly known as the (outer) tunnel header.

An example of where this in-flight EH insertion may take place is when a packet enters a transit BGP autonomous system network [RFC4271] along its path across the Internet.

3.2. In-Flight Removal

At some later point along the IPv6 packet's path towards its final destination, the packet is somehow determined to need to have the previously inserted EH removed, independently of the Destination Address of the packet. The packet is again split apart, at the point where the one or more inserted EHs exists, and then the inserted EH or EHs are removed. The packet is then reassembled, and sent further towards its final destination.

Again, the packet's original source and Destination Address field values are left unchanged when EH removal takes place. As with insertion, the likely only IPv6 fixed header field modified during EH removal would be the immutable Next Header field.

An example of where this in-flight EH removal would take place is when a packet leaves a transit BGP autonomous system network that has previously inserted one or more EHs.

3.3. In-Flight Insertion Without Removal

A possibility is that in-flight insertion of the EH occurs without the intention that it is subsequently removed while the packet is in-flight.

In this instance, the device that is intended to process the inserted EH is the IPv6 host identified in the packet's (unchanged) Destination Address field.

4. EH Removal Failure Causes

4.1. Implementation Bugs

Despite being configured to remove the inserted one or more EHs, an implementation bug could cause some or all packets not to have the inserted EH or EHs removed.

4.2. Partial Node Failure

Even though the software or firmware that is to perform EH removal is bug free, it is possible that a hardware fault could cause EH removal to not occur, while packets are still sent towards their final destination. This could occur because the hardware fault that does not cause the node to entirely fail, only partially performing some of its functions..

4.3. Operator Configuration Error

Due to human error, the function to remove the inserted EH or EHs may be misconfigured. Consequently, the inserted EH or EHs may not be removed for some or all packets.

When the packets to have the EH(s) removed are transit packets, meaning these packets are likely leaving the operator's own network, and entering another operator's network, it is less likely that the packets leaving are inspected to ensure the EH removal function has been configured correctly. It is common to assume that if traffic is leaving the local network in the expected volumes, then the traffic is being processed correctly by the egress network device. This can be because the equipment, time and effort to validate this egressing traffic can be very expensive when traffic volumes are in the 10s or perhaps 100s of gigabits per second.

The receiving network will also not detect or be able to detect that the inserted EHs have not been removed, as the inserted EH or EHs will appear to have been placed in the packet by the IPv6 host identified in the packet's Source Address field.

5. Single Point of Failure

When functions that inspect or modify packets beyond standard IP packet forwarding are performed at the edge of a network, such as a network firewall or a Network Address Translation, it is typical for there to only be one device performing that will perform this function at the packets' exit from the network. It is rare to have two devices in-line or in series that are performing this same inspection or modification, providing redundancy for the function should it fail to be performed correctly at the first function instance.

In a scenario where EHs are to be removed, it is likely that the device that is to perform EH removal will be a single point of failure.

6. MUST Remove is Aspirational

RFCs/IDs say the inserted EH MUST be removed at the EH insertion boundary, and then use that to say it is a safe operation. This is ignoring the reality of all of the above possible causes of an inserted EH failing to be removed. Such a MUST statement is no more than aspirational - it is a theoretically true statement in 100% of cases, but in practice cannot ever be assured to be true in 100% of cases, due to the removal failure causes, described previously.

7. Harm

7.1. Violates RFC8200 and All Of Its Ancestors.

(RFC8200 EH processing text quote)

RFC 2460 and ancestors back to RFC 1883 text quote.

7.2. Ignores Source Address Field Semantics

7.3. Breaks ICMPv6

7.3.1. Breaks PMTUD

7.4. Breaks IPsec

7.5. May Cause Faults in Subsequent Transit Networks

If an in-flight inserted EH is not removed, and the packet travels into another subsequent transit network, that subsequent transit network may have an alternative interpretation of the inserted EH, causing a fault.

The subsequent transit network, if using EH insertion, would likely blindly insert another instance of the EH, resulting in a packet with two EHs. At network egress, the incorrect EH may be removed, which would also still leave a remaining inserted EH to travel into further subsequent networks. A directly subsequent network that is also performing EH insertion is unlikely to act as a sanitiser for EHs that were inserted by previous upstream networks.

7.6. Incorrect Destination Host Processing

Should an in-flight inserted EH fail to be removed, the receiving IPv6 host may process it incorrectly. Incorrect processing could involve discarding the packet when it should be further processed, or processing the packet when it should be discarded.

An failed to be removed, in-flight inserted EH is less likely to be understood by a typical receiving IPv6 host, as the inserted EH is being used for a network function.

If an IPv6 host receives an EH that it doesn't understand, how to process the EH is encoded in the highest order two bits of the EH type [RFC8200]. If the highest order bits are all zeros, skip this EH and continue processing the header. If the highest order bits are 01, discard the packet. If the highest order bits are 10 or 11, then discard the packet, and either universally generate and send an ICMP Parameter Problem for all Destination Address types, or for the latter value, generate and send an ICMP Parameter Problem for only non-multicast Destination Addresses.

A failed to be removed, in-flight inserted EH may not have these highest order bits set correctly to best suit the application's and its end-user's goals.

For example, if the packet was carrying a streaming video application's data, then an unknown inserted EH, yet failed to be removed network function EH may be harmless to the application and its end-user if it can be skipped over by the receiving IPv6 host. However, if the inserted yet not-removed EH has non-zero highest order bits, the packet would be discarded, causing the video data not to be displayed to the end-user, despite there being no harm in doing so.

Alternatively, there could be cases where the inserted, yet failed to be removed EH should cause a packet to be discarded by the host with the Destination Address, perhaps for security reasons. However, if the inserted EH has highest order two bits that are all zero, meaning ignore the unknown EH and continue processing the header, the packet will instead be further processed by the receiving IPv6 host. Perhaps the packet will be further processed in a way that violates a security policy that should be being enforced when the inserted, yet failed to be removed EH is being processed.

7.7. Implementation Complexity

IPv6 uses a packet's Destination Address to determine the point where forwarding across the network stops, and processing up the protocol stack at a destination host starts.

In other words, the Destination Address of a packet identifies the point in the network where processing of the packet starts going beyond the IPv6 fixed header, and where the intention of the packet processing stops being limited to forwarding towards the packet's destination.

This is the fundamental distinction between an IPv6 router and a host; an IPv6 router forwards packets with non-local addresses [RFC8200], while an IPv6 host, with that holds address that matches a packet's Destination Address, processes the packet locally, with processing occurring beyond the IPv6 packet's fixed header. Note that these definitions of IPv6 router and host are functional; a router as a device implements both IPv6 router and host functions - the device's forwarding plane implementing the IPv6 router function, and the device's control plane implementing IPv6 host functions.

This means that all IPv6 addresses that appear in an IPv6 packet's Source Address and Destination Address field are, without exception, host addresses.

The decision as to whether to process the packet beyond the fixed header or not is binary and simple - does the current node holding the packet possess the IPv6 address recorded in the Destination Address field of the packet?

Identifying packets that have had EH's inserted, to then remove and process the EH, is much more complex than the simple, Destination Address match selector. The EH chain inside each packet has to be processed to find the EH that was inserted, should it exist.

7.8. Costly Troubleshooting

The lack of attribution of which device inserted the EH could incur high costs during troubleshooting, in terms of time, effort and financially for a commercially operated network, should EH removal fail.

Imagine a scenario where there is a popular streaming video on demand (SVOD) content service on the Internet providing content to large number of customers at a residential "eyeball" ISP. Between the SVOD network and the ISP network, there are 8 different transit networks.

One or more of those transit networks decides to implement a local function using EH insertion. Unfortunately, EH removal at the egress of one or possibly more of these transit networks fails, due to one of the possible causes mentioned previously. This or these failures occurs somewhere in the path from the SVOD network to the ISP network.

As the Destination Address of this packet is as it was when the SVOD network sent the packet, prior to when the EH was inserted while in transit, this packet will continue to be forwarded and then delivered to the destination host at the ISP network.

When the packet arrives at the destination host, the host is required to process the Extension Headers in order [RFC8200]. Should an Extension Header be encountered that the host does not recognise, the host may discard the packet based on the two highest order bits of the EH type. The packet's video data will not be available to the video application and will not be displayed to the end-user.

As the transit network(s) that inserted the EH, yet failed to remove it may be carrying the SVOD traffic for 100s or 1000s of customers at the residential ISP, 100s or 1000s of customers will fail to receive their SVOD service. These customers will either contact the support helpdesk of the ISP or the support helpdesk of the SVOD service to report the fault.

In either case, the network operators trying to resolve this fault will have no indication which of the 8 transit networks is inserting the EH yet failing to remove it.

Consequently, the only way to troubleshoot this is through a brute-force process of elimination. It would be necessary to contact all of the 8 transit networks, and ask them if they're inserting EHs while packets are in-flight. If they are, then it may be necessary to convince them that their inserted EHs are failing to be removed at the egress of their network, as they may be sceptical, since there are no local effects of the fault. Providing a packet capture with the inserted EH that is causing the fault does not provide any supporting evidence to show that a specific transit network is failing to remove inserted EHs.

Once the operator or operators of the networks that are inserting EHs are convinced that their network may not be removing EHs, those operators will now have to arrange to inspect the traffic leaving their network, after it has been sent by their network's egress device.

Organising and executing this traffic inspection is likely to be time and possibly resource intensive. The egress transit link attached to the device that is failing to remove the inserted EHs may be carrying 10s or perhaps 100 or more Gbps of transit traffic. Inserting a traffic inspection device within the link will cause this traffic to shift to other links if available, either when the link is broken, or in preparation for breaking the link.

As this will be a service impacting event, it will likely need to go through change management procedures for review. Given the event's severity, service impact notification may involve a number of days, prior the event being executed.

Once the faulty device has been identified, it needs to be rectified. This may involve rectification by the device's vendor if the fault cause is a software or firmware bug.

Given the above troubleshooting process, the amount of parties involved, and the time it could take to perform the troubleshooting and rectification steps, in this scenario, troubleshooting and rectification would likely take in the order of at least a week, if not a number of weeks. This will have a very significant business impact on either or both of the SVOD provider or the residential ISP, both in terms of market preception and lost customers, frustrated with how long this fault is taking to resolve to the point where they cancel their service.

8. Be conservative in what you send, ...

i.e. Postel's law

"Be conservative in what you send, ..." is saying try to avoid sending anything that the receiver may not be expecting and that may confuse the receiver. The "be liberal in what you accept" is advising robustness to attempt to tolerate a sender that has failed to be conservative.

In-flight EH insertion violates the conservative sender part, because [RFC8200] compliant receivers are not expecting to receive EHs in a packet that were not placed there by the device identified in the packet's Source Address field. A device performing in-flight EH insertion is intentionally not being conservative with what it is sending, in comparison to the scope of what an [RFC8200] compliant receiver expects to receive.

9. Solution: Encapsulation

In the Internet Protocol Architecture [RFC1122][RFC6272], adding new information to an existing protocol data unit is achieved through encapsulation. The new information is recorded in a new header and possibly a new trailer, which are then used to surround or enclose the existing protocol data unit, similar to how an envelope is used to enclose the contents of a letter in the physical mail system.

In addition to other new information, the new encapsulation header records the source of that new information. For the link-layer that is the source node's link-layer address; for the IP layer it is either the IPv4 or IPv6 source host's address; and for the transport layer, it is the source transport layer port, or some other transport layer source entity identifier.

The new encapsulation also records the destination entity or entities that is or are intended to receive and process the new information. For the link-layer, the destination node's link-layer address, or a single group address that identifies a set of link-layer nodes; for the IP layer, the IPv4 or IPv6 destination host, or a single group address that identifies a set of hosts; and for the transport layer, the destination transport layer port or other transport layer destination entity identifier.

The source and destination entity identification in the encapsulation header provides unambiguous and explicit identification of both which entity created and sent the new information, and which entity or entities are to process the new information.

9.1. IPv6 Tunnelling

If additional IPv6 information is to be added to an existing IPv6 packet while it is in-flight, such as a new Extension Header, then a new IPv6 header is required. This new IPv6 header will unambiguously record the identity of the IPv6 host that has added the new IPv6 information in the Source Address field, and will unambiguously record the identity of the IPv6 host (or group of hosts) that is to process the added IPv6 information in the Destination Address field. A new IPv6 packet is created using the new IPv6 header, followed by the new supplementary information, followed by the existing IPv6 packet, appearing in the payload field of the new packet. IPv6-in-IPv6 encapsulation is commonly known as "tunneling", and is specified in [RFC2473], which includes showing how new information added via Extension Headers occurs. [intarea-tunnels] provides more discussion of IP tunneling in the context of the Internet Architecture.

Conceptually, IPv6-in-IPv6 tunneling is a form of link-layer encapsulation from the perspective of the existing (and eventually inner) IPv6 packet. It just happens to be a coincidence that the outer link-layer encapsulation header and other new information (i.e. Extension Headers) has the same protocol format and field semantics as the existing, inner IPv6 packet.

9.2. MPLS

Despite using terms such as "label imposition" or "label swapping", MPLS [RFC3031] also follows this encapsulation model to add new information, via labels, to an existing in-flight protocol data unit, such as an IPv6 packet. In-flight insertion of MPLS labels never occurs.

At each hop through the MPLS network where labels are processed, at devices known as Label Switching Routers (LSRs), upon egress from the

LSR, a new link-layer header is created that both unambiguously identifies the current LSR in the link-layer Source Address field, and unambiguously identifies the next LSR (or set of LSRs) that is to process the set of labels that are encoded in the link-layer protocol data unit sent by the current LSR. The labels are encoded following this new header, and then the original packet follows in the link-layer payload field.

If in-flight MPLS label insertion were to be actually occurring, then it would mean that as a packet was label switched across a set of LSRs along a Label Switched Path (LSP), the link-layer header Source Address would not change across the LSP - it would remain as the Source Address of the LSR at the head end of the LSP, regardless of how many subsequent LSRs the packet is label switched through.

In-flight MPLS label insertion would also mean that the Destination Address in the link-layer header would also not change as the packet is label switched along the LSP. It would remain unchanged regardless of how many LSRs the packet traverses, and would likely identify the final LSR at the tail end of the LSP.

If MPLS had used an in-flight insertion model, then MPLS would have likely suffered from problems similar to those described above that can occur with IPv6 EH insertion.

10. Reducing Tunneling Overhead

As a tunnel is creating a virtual link layer, link-layer compression of the inner IPv6 header and its payload can be used to effectively reduce the tunneling overhead.

10.1. ROHC

"The Robust Header Compression (ROHC) protocol provides an efficient, flexible, and future-proof header compression concept. It is designed to operate efficiently and robustly over various link technologies with different characteristics." [RFC5795]

10.2. Skinny IPv6-in-IPv6 Tunneling

Skinny-IPv6-in-IPv6 tunnelling [SKINNYV6V6] is a stateless form of tunnelling compression that leverages two characteristics of IPv6 and IPv6 networks:

- o The common semantics between the inner IPv6 and outer IPv6 tunnel packet's headers. While the inner IPv6 packet is in flight over the IPv6 tunnel, the large majority of its header field values are

carried and proxied by the outer IPv6 tunnel header's corresponding fields.

- o The availability of many /64 prefixes within an IPv6 network, using /64s rather than /128s to identify IPv6 tunnel end-points. This allows the inner packet's 64 bit IIDs to be carried in the outer IPv6 tunnel packet's IID fields while the inner packet is carried over the IPv6 tunnel.

11. In-Flight Insertion Considered Harmful

More generally, insertion within an existing, in-flight packet at any location within the packet is considered harmful. EH insertion, as described and discussed previously, is a more specific instance of a harmful practise.

12. Security Considerations

13. Acknowledgements

Review and comments were provided by YOUR NAME HERE!

This memo was prepared using the xml2rfc tool.

14. Change Log [RFC Editor please remove]

draft-smith-6man-in-flight-eh-insertion-harmful-00, initial version, 2019-09-09

draft-smith-6man-in-flight-eh-insertion-harmful-01, update, 2019-11-03

- o Added co-authors
- o Link-layer compression section
- o Costly troubleshooting scenario section

draft-smith-6man-in-flight-eh-insertion-harmful-01, update, 2019-11-03

- o Version bump. Request for WG adoption per discussion at IETF 106.

15. References

15.1. 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>>.
- [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>>.

15.2. Informative References

- [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>>.
- [RFC5795] Sandlund, K., Pelletier, G., and L-E. Jonsson, "The RObust Header Compression (ROHC) Framework", RFC 5795, DOI 10.17487/RFC5795, March 2010, <<https://www.rfc-editor.org/info/rfc5795>>.
- [SKINNYV6V6] "Skinny IPv6 in IPv6 Tunnelling", <<https://datatracker.ietf.org/doc/draft-smith-skinny-ipv6-in-ipv6-tunnelling/>>.

Authors' Addresses

Mark Smith
PO Box 521
Heidelberg, VIC 3084
AU

Email: markzzzsmith@gmail.com

Naveen Kottapalli

Email: naveen.sarma@gmail.com

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

Email: rbonica@juniper.net

Fernando Gont
SI6 Networks
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires
Argentina

Email: fgont@si6networks.com

Tom Herbert
Quantonium
Internet Road
Santa Clara, CA
USA

Email: tom@quantonium.net

6man
Internet-Draft
Intended status: Standards Track
Expires: April 16, 2020

H. Song, Ed.
Futurewei Technologies
October 14, 2019

Support Postcard-Based Telemetry for SRv6 OAM
draft-song-6man-srv6-pbt-01

Abstract

Applications such as SRv6 TE may require to collect detailed performance data on SR paths. Existing in-situ OAM techniques incur encapsulation and header overhead issues. This document describes a method based on Postcard-based Telemetry with Packet Marking for SRv6 on-path OAM, which avoids the extra overhead for encapsulating telemetry-related instruction and metadata in SRv6 packets.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on April 16, 2020.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. PBT Triggered by Marking for SRv6	3
2.1. Data Template	3
2.2. Postcard Correlation	4
2.3. Operational Considerations	4
3. Use Cases	4
4. Security Considerations	4
5. IANA Considerations	4
6. Contributors	5
7. Acknowledgments	5
8. References	5
8.1. Normative References	5
8.2. Informative References	5
Author's Address	6

1. Introduction

The ability to collect the on-path data about SRv6 packets at each segment is important for SRv6 OAM, especially for monitoring the application-aware services. Some SR-TE algorithms need to acquire realtime flow forwarding performance on each path. The In-situ OAM (IOAM) [I-D.ietf-ippm-ioam-data] trace option can be used for such purpose. However, SRv6's SRH can be large due to the long segment list. The IOAM trace option introduces significant additional overhead to the SRv6 packets with its instruction and data trace. The large header overhead complicates the packet processing and may exceed the forwarding hardware's header processing capability.

The extra IOAM trace option header also brings encapsulation challenges as documented in [I-D.li-6man-ipv6-sfc-ift]. Here we only restate a subtle issue about the IOAM scope: if IOAM header is encapsulated as another IPv6 extension header, the juxtaposition of IOAM and SRH makes it ambiguous to determine the scope and coverage of IOAM: it is unclear if the IOAM is applied to the entire forwarding path or just to the segment nodes. In reality, either case can find its application.

The Direct EXport (DEX) option of IOAM described in [I-D.ioamteam-ippm-ioam-direct-export] partially relieves the packet overhead pressure by avoiding including trace data in SRv6 packet, but the encapsulation issue remains, so does the aforementioned ambiguity. In this document, we propose to apply the PBT-M scheme from [I-D.song-ippm-postcard-based-telemetry] for on-path SRv6 telemetry, which can help to solve the encapsulation and overhead issues.

2. PBT Triggered by Marking for SRv6

PBT-M requires marking a packet as a trigger to collect on-path data about the packet. The collected data are exported by an independent "postcard" packet. Therefore, there is no new header encapsulation requirement.

Eight flag bits are currently reserved in SRH. One of those bits can be used as the marking flag, as shown in the following figure. If the "T"-bit is set to 1, the segment node which process the SRH needs to export the on-path data about this packet as pre-configured through management interface.

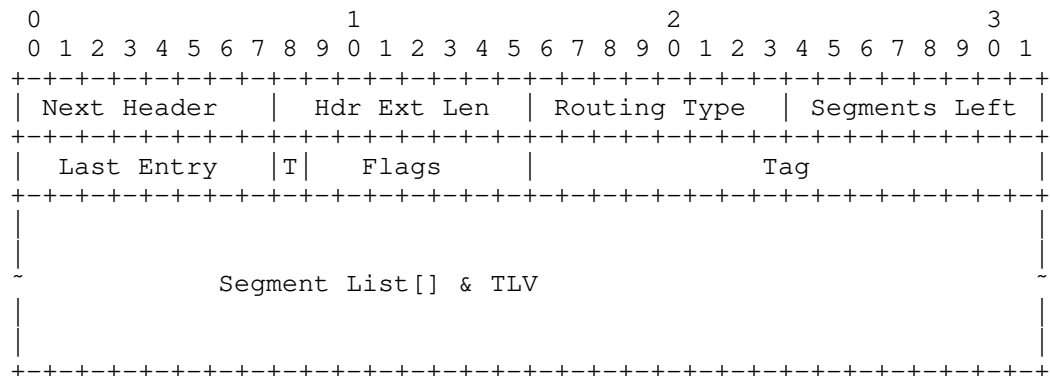


Figure 1: SRH with PBT Mark Flag

2.1. Data Template

It is possible to have the same configuration for all the segment nodes on the data set to collect. However, different flows may require different data collection profiles. It would be more flexible to have multiple different data templates supported by the segment nodes and each packet can designate one template that best

suits its interests to use. The template ID can be carried as a TLV in SRH.

2.2. Postcard Correlation

As discussed in [I-D.song-ippm-postcard-based-telemetry], PBT-M has some issues to correlate the postcards from the different segment nodes for the same user packet. While several solutions are given to mitigate the problem, it is ideal to be able to correlate the postcards without any constraint and precondition.

A flow ID and a sequence number can be included as TLVs in SRH. The format and usage of the flow ID and the sequence number are the same as those in IOAM DEX option in [I-D.ioamteam-ippm-ioam-direct-export]. Further, the exported postcard may include the SRH or the current SID which provides a trace to order the postcards.

2.3. Operational Considerations

The SR source node is responsible to determine the policy for setting or resetting the "T"-bit.

A segment node can decide independently whether or not to react on the "T"-bit.

3. Use Cases

TBD.

4. Security Considerations

Since PBT incurs some extra packet processing and transport cost, "T" flag is usually selectively set on a subset of packets by the source node. A potential DoS attack may set the "T" flag for all the packet with the intention to overwhelm the segment nodes. Therefore, the postcards should be generated on the basis of the best effort.

5. IANA Considerations

[I-D.ietf-6man-segment-routing-header] defines a new registry named "Segment Routing Header Flags". This document requests the allocation of a new flag bit "T" for the telemetry trigger mark.

6. Contributors

TBD.

7. Acknowledgments

TBD.

8. References

8.1. Normative References

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

8.2. Informative 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-24 (work in progress), October 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-07 (work in progress), September 2019.
- [I-D.ioamteam-ippm-ioam-direct-export] Song, H., Gafni, B., Zhou, T., Li, Z., Brockners, F., Bhandari, S., Sivakolundu, R., and T. Mizrahi, "In-situ OAM Direct Exporting", draft-ioamteam-ippm-ioam-direct-export-00 (work in progress), October 2019.

[I-D.li-6man-ipv6-sfc-ifit]

Li, Z., Peng, S., and K. LEE, "IPv6 Encapsulation for SFC and IFIT", draft-li-6man-ipv6-sfc-ifit-02 (work in progress), September 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-05 (work in progress), September 2019.

Author's Address

Haoyu Song (editor)
Futurewei Technologies
2330 Central Expressway
Santa Clara
USA

Email: hsong@futurewei.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: May 24, 2021

D. Voyer, Ed.
Bell Canada
C. Filsfils
D. Dukes, Ed.
Cisco Systems, Inc.
S. Matsushima
Softbank
J. Leddy
Individual Contributor
Z. Li
Huawei
J. Guichard
Futurewei
November 20, 2020

Deployments With Insertion of IPv6 Segment Routing Headers
draft-voyer-6man-extension-header-insertion-10

Abstract

SRv6 is deployed in multiple provider networks.

This document describes the usage of SRH insertion and deletion within the SR domain and how security and end-to-end integrity is guaranteed.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on May 24, 2021.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. Deployment Report	3
2.1. Deployments	3
2.2. Vendor and Open-Source Support	3
3. Deployment Experience With SRH Header Operarion	4
3.1. The SR Domain	5
4. Baseline Usecase	5
5. Choosing an SRv6 SID Block	6
6. Securing the SR Domain	7
7. MTU within the SR domain	7
8. VPN with SRv6	7
9. TILFA with SRv6	8
9.1. SRH Insertion Process	8
9.2. The Penultimate SID of the Inserted SRH is of PSP flavor	9
9.3. MTU-delta	9
10. Security Considerations	10
11. IANA Considerations	10
12. Contributors	10
13. References	10
13.1. Normative References	10
13.2. Informative References	11
Authors' Addresses	11

1. Introduction

[I-D.matsushima-spring-srv6-deployment-status] records multiple SRv6 deployments in multiple networks

In each deployment, traffic traversing an SR domain is encapsulated in an outer IPv6 header for its journey through the SR domain.

To implement transport services within the SR domain, insertion or removal of an SRH after the outer IPv6 header is performed. Any segment within the SRH is strictly contained within the SR domain.

The SR domain always preserves the end-to-end integrity of traffic traversing it. No extension header is manipulated, inserted or removed from an inner transported packet. The packet leaving the SR domain is exactly the same (except for the hop-limit update) as the packet entering the SR domain.

The SR domain is designed with link MTU sufficiently greater than the MTU at the ingress edge of the SR domain.

2. Deployment Report

The following deployments are as of November 2019.

2.1. Deployments

Six operators have publicly reported SRv6 deployments with commercial traffic supported by linerate hardware. Each deployment follows the network design and SRH add/remove behavior described in this document.

Softbank

China Telecom

Iliad

China Unicom

CERNET2

MTN Uganda Ltd.

Further information can be found in
[I-D.matsushima-spring-srv6-deployment-status]

2.2. Vendor and Open-Source Support

Eighteen unique implementations of SRv6 are available from multiple vendors and open source initiatives that support the SRH add/remove behavior described in this document:

Cisco ASR 9000

Cisco NCS 5500

Cisco NCS 560

Cisco NCS 540

Cisco ASR1000

Huawei ATN

Huawei CX600

Huawei NE40E

Huawei ME60

Huawei NE5000E

Huawei NE9000

Huawei NG-OLT MA5800

Barefoot Tofino 1 NPU

Barefoot Tofino 2 NPU

Broadcom Jericho 1, 1+

Broadcom Jericho 2

Linux kernel

FD.io VPP

Marvell's Prestera Falcon CX 8500 family

Intel PAC N3000

Further information can be found in
[I-D.matsushima-spring-srv6-deployment-status]

3. Deployment Experience With SRH Header Operation

3.1. The SR Domain

An SR Domain is defined in [RFC8402].

Section 5.2 of [I-D.ietf-6man-segment-routing-header] further describes the SR domain as a single system with delegation among components. It states:

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.

In other words, all packets within the SR domain have a source and destination address within the SR Domain.

The SR domain is secured as per Section 5.1 of [I-D.ietf-6man-segment-routing-header] and no external packet can enter the domain with a destination address equal to a segment of the domain.

In other words, no node outside the SR domain may send packets to, nor make direct use of, segments within the SR domain.

4. Baseline Usecase

The following abstract illustration shows the SR Domain, how traffic is encapsulated when traversing the SR domain, and (in subsequent sections) how an SRH is inserted and processed for a packet traversing the SR domain. It is representative of all deployments in Section 2.1.

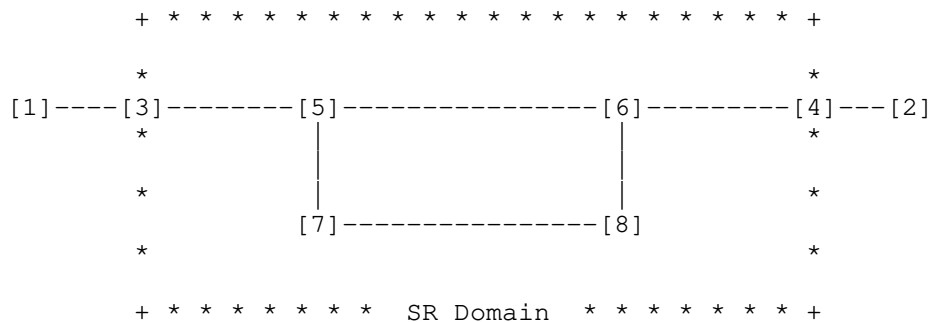


Figure 1

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

Since all inter domain packets are encapsulated for the part of the packet journey that is within the SR Domain, a packet sent from 1 and destined to 2 is encapsulated in an outer IP v6 header between nodes 3 and 4.

5. Choosing an SRv6 SID Block

Without revealing the specifics of each deployment, the following well-known technique can be used:

Obtain a GUA block from the respective registry (e.g.
 PPPP:PPP0::/28)

Sub-allocate a block for SID allocation (e.g.
 PPPP:PPPB:BB00::/40)

Allocate a /64 SID locator to each node in the domain that needs to provide network instruction (e.g. node 4 gets
 PPPP:PPPB:BB00:0004::/64 as a SID locator)

Vendors and operators have automated the process of locator selection, the details of which are outside the scope of this document.

6. Securing the SR Domain

The security measures defined in [I-D.ietf-6man-segment-routing-header] Section 5.1 are applied.

Protection level 1: filter external traffic entering the SR domain. For example, node 4 (on its interface from node 2) applies an ingress ACL that drops any packet with DA within the PPPP:PPPB:BB00::/40 block.

Protection level 2: filter internal traffic. For example, node 4 (on its interface from node 6) applies an ingress ACL that drops any packet with DA in PPPP:PPPB:BB00:0004::/64 block if SA is not within the block PPPP:PPP0::/28

Vendors and operators have automated the application of these protection levels, the details of which are outside the scope of this document.

7. MTU within the SR domain

The deployments, Section 2.1, leverage the extensively used practice of ensuring an MTU within the domain is bigger than the MTU on the external links of the domain.

More specifically, the MTU difference (MTU-Delta) is designed to be larger than the maximum encapsulation overhead deemed required by the deployment.

The exact number is operator specific and is outside the scope of this document. Some indications on how to plan this are provided in the following sections.

Any packet exceeding the MTU of a link generates an IPv6 ICMP error message "packet too big" back to the source of the packet.

8. VPN with SRv6

The deployments involve the creation of commercial SRv6-based VPN traffic as described in [I-D.ietf-bess-srv6-services].

The salient point to note is that no SRH needs to be inserted to realize an SRv6 VPN service.

The ingress PE encapsulates the inner packet in an outer header and sets the outer DA to the END.DT/DX SID signaled by the egress PE.

MTU-Delta must be ≥ 40 bytes to allow for the outer IPv6 encapsulation without fragmentation.

9. TILFA with SRv6

The deployments involve the delivery of sub-50msec TILFA protection to the commercial SRv6-based VPN traffic transported by the operator network [I-D.ietf-rtgwg-segment-routing-ti-lfa].

In these deployments, when a failure is detected, the Point of Local Repair (PLR) inserts an SRH implementing the precomputed TILFA backup path.

The following salient points are discussed:

SRH insertion process

The penultimate SID of the inserted SRH is of PSP flavor

MTU-delta planning

9.1. SRH Insertion Process

When an SRH is inserted by an intermediate node it walks the IPv6 header chain to the first header after the IPv6 header and inserts the SRH prior to that header.

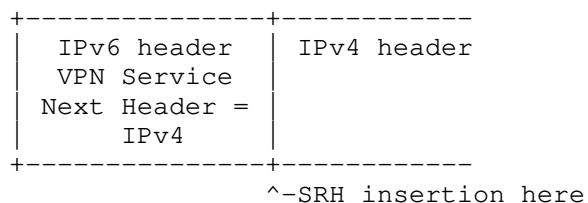


Figure 2

An SR Policy headend within the SR domain inserts an SRH as follows:

1. Determine where to insert the SRH.
2. Copy the destination address from the IPv6 header to Segment List[0] of the SRH to be inserted. This ensures the original destination address is restored upon execution of the final segment in the inserted SRH.
3. Increase the IPv6 header payload length field by the length in bytes of the inserted SRH.

If the resulting payload length exceeds 2^{16} bytes generate an ICMP "Packet To Big" error message to the source with an MTU of 2^{16} minus the length in bytes of the SRH and discard the packet. Note: this does not occur in reported deployments given the MTU design constraint.

4. Set the SRH next header field to the value in the next header field of the header that will precede the SRH.
5. Set the next header field of the header that will precede the SRH to the routing extension header (43)
6. Set the IPv6 destination address to the first segment in the segment list of the SRH to be inserted. This segment may or may not be present in the SRH depending on the use of a reduced SRH, see section 4.1.1 of [I-D.ietf-6man-segment-routing-header].
7. Insert the SRH into the packet at the location it should be inserted and resubmit the packet to the IPv6 module for transmission to the new destination.

9.2. The Penultimate SID of the Inserted SRH is of PSP flavor

The TILFA protection service is essentially a transparent service: it seeks to make the loss of a link, node or SRLG invisible to the transport service. It is also a very transient service as it only lasts a few hundreds of msec while the IGP converges.

Consistent with this transparent service definition, the deployments leverage a TILFA computation that ensures that the penultimate SID of the inserted SRH is of PSP flavor.

9.3. MTU-delta

The vendors reporting the listed deployments have collectively deployed TILFA in tens of SR-MPLS networks, in 6 SRv6 networks and have simulated their SRv6 algorithm in tens of collected real topologies. The inferred experience is that the probability that a TILFA backup path requires more than 2 SRv6 SIDs is very rare.

MTU-Delta must be ≥ 80 bytes.

40 bytes (VPN service)
+ 8 (SRH) (TILFA)
+ 2 * 16 (2 TILFA SID's)

The maximum encapsulation size of any node within the SR domain is limited to a specific value, this maximum is used to calculate the maximum link MTU of interfaces ingress to the SR domain.

10. Security Considerations

Section 6 describes the method of securing the SR domain in the deployments listed.

All security considerations discussed in [I-D.ietf-6man-segment-routing-header] are equally applicable when an SRH insertion is performed.

11. IANA Considerations

This document doesn't introduce any IANA request.

12. Contributors

The authors would like to thank the following for their contributions: Robert Raszuk, Stefano Previdi, Stefano Salsano, Antonio Cianfrani, David Lebrun, Olivier Bonaventure, Prem Jonnalagadda, Milad Sharif, Hani Elmalky, Ahmed Abdelsalam, Arthi Ayyangar, Dirk Steinberg, Wim Henderickx.

13. References

13.1. Normative References

- [I-D.ietf-6man-segment-routing-header]
Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-26 (work in progress), October 2019.
- [I-D.ietf-bess-srv6-services]
Dawra, G., Filsfils, C., Talaulikar, K., Raszuk, R., Decraene, B., Zhuang, S., and J. Rabadan, "SRv6 BGP based Overlay services", draft-ietf-bess-srv6-services-05 (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>>.

[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>>.

13.2. Informative References

[I-D.ietf-rtgwg-segment-routing-ti-lfa]
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., Francois, P., Voyer, D., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-04 (work in progress), August 2020.

[I-D.matsushima-spring-srv6-deployment-status]
Matsushima, S., Filsfils, C., Ali, Z., Li, Z., and K. Rajaraman, "SRv6 Implementation and Deployment Status", draft-matsushima-spring-srv6-deployment-status-09 (work in progress), November 2020.

Authors' Addresses

Daniel Voyer (editor)
Bell Canada
Canada

Email: daniel.voyer@bell.ca

Clarence Filsfils
Cisco Systems, Inc.
Belgium

Email: cfilsfil@cisco.com

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

Email: ddukes@cisco.com

Satoru Matsushima
Softbank
Japan

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

John Leddy
Individual Contributor
USA

Email: john@leddy.net

Zhenbin Li
Huawei
China

Email: lizhenbin@huawei.com

James Guichard
Futurewei
USA

Email: james.n.guichard@futurewei.com

Network
Internet-Draft
Intended status: Informational
Expires: May 6, 2020

C. Weiqiang
China Mobile
P. Shaofu
L. Aihua
ZTE Corporation
G. Mirsky
ZTE Corp.
W. Xiaolan
New H3C Technologies Co. Ltd
C. Wei
Centec
S. Zadok
Broadcom
November 3, 2019

Unified Identifier in IPv6 Segment Routing Networks
draft-wmsaxw-6man-usid-id-use-00

Abstract

Segment Routing architecture leverages the paradigm of source routing. It can be realized in a network data plane by prepending the packet with a list of instructions, a.k.a. segments. A segment can be encoded as a Multi-Protocol Label Switching (MPLS) label, IPv4 address, or IPv6 address. Segment Routing can be applied in the MPLS data plane by encoding segments in an MPLS label stack. It also can be applied to the IPv6 data plane by encoding a list of segment identifiers in IPv6 Segment Routing Extension Header (SRH). In this document is described the use of unified segment identifiers in use cases where interworking between SR-MPLS and SRv6 is required.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 6, 2020.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
1.1. Conventions used in this document	3
1.1.1. Terminology	3
1.1.2. Requirements Language	3
2. Requirements for Using SRv6 in Backhaul	4
3. Using SRv6 U-SID in Backhaul	4
3.1. Smoothly Upgrading to SRv6 from SR-MPLS	4
3.2. Interworking Between SRv6 and SR-MPLS	5
3.3. Compressing SRv6 Header Effectively	6
3.4. Support a Super-large-scale Networking and Flexibility in Assigning Addresses	6
4. Operations with Unified Segment Identifier	6
5. IANA Considerations	7
6. Security Considerations	7
7. Acknowledgements	7
8. Normative References	7
Authors' Addresses	8

1. Introduction

Many functions related to Operation, Administration and Maintenance (OAM) require identification of the SR tunnel ingress and the path, constructed by segments, between the ingress and the egress SR nodes. Combination of IPv6 encapsulation [RFC8200] and the Source Routing Extension Header (SRH) [I-D.ietf-6man-segment-routing-header], referred to as SRv6, comply with these requirements while it is challenging when applying SR in MPLS networks [I-D.ietf-spring-segment-routing-mpls], also referred to as SR-MPLS.

On the other hand, the size of the IPv6 segment identifier (SID) presents a scaling challenge to use topological instructions that

define a strict explicitly routed path in combination with service-based instructions. At the same time, that is where the SR-MPLS approach provides better results due to smaller SID length.

SR-MPLS currently, more often than SRv6, is used in metro networks. With the gradual deployment of SRv6 in the core networks, it becomes necessary to support interworking between SR-MPLS and SRv6. Operationally it would be more efficient and straightforward if SRv6 can use the same size SIDs as in SR-MPLS. The SRH can be extended to use the same as in SR-MPLS SID length to support the unified segment identifier (U-SID) [I-D.mirsky-6man-unified-id-sr]. As a result of using this approach, U-SIDs can be used end-to-end across a tunnel that spans over SR-MPLS and SRv6 domains.

In this document is described the use of unified segment identifiers, encoded as MPLS label and/or 32 bits-long address, in use cases when interworking between SR-MPLS and SRv6 networks is required.

1.1. Conventions used in this document

1.1.1. Terminology

SR: Segment Routing

SRH: Segment Routing Extension Header

MPLS: Multiprotocol Label Switching

SR-MPLS: Segment Routing using MPLS data plane

SID: Segment Identifier

IGP: Interior Gateway Protocol

OAM: Operation, Administration and Maintenance

SRv6: Segment Routing in IPv6

U-SID: Unified Segment Identifier

1.1.2. 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.

2. Requirements for Using SRv6 in Backhaul

2G/3G/4G backhaul networks widely deploy MPLS to connect wireless services. Many operators are already deploying 5G networks. To optimize the operation of the network, many operators intent to adopt the segment routing. Currently, given maturity of SR-MPLS, it has been deployed on a large scale. Meanwhile the requirements of 5G super-large-scale number of connections accelerate the deployment of IPv6 networks. Thus, logically, operators consider SRv6 solution to fulfill the 5G backhaul requirement. But the backhaul network could not deploy SRv6 in one day, especially if it has already been using MPLS and SR-MPLS. It might be reasonable to upgrade from MPLS to SR-MPLS and then to SRv6. There are several essential operational requirements for the deployment of SRv6 in 5G backhaul network:

1. Ensure the ability to transform the existing SR-MPLS backhaul network into an SRv6 5G backhaul network incrementally.
2. Support interworking between SRv6 and SR-MPLS domains in the network.
3. Support SRv6 header compressing.
4. Support super-large-scale networking and address planning

3. Using SRv6 U-SID in Backhaul

U-SID provides a solution that complies to the 5G backhaul requirements.

3.1. Smoothly Upgrading to SRv6 from SR-MPLS

SR-MPLS uses a segment encoded as a label in an MPLS label stack to simplify the backhaul network. It leverages the advantages of both source-routing and MPLS. Existing backhaul networks that use MPLS can be first updated to use SR-MPLS. SRv6 uses the segment encoded as an identifier in IPv6 SRH. The SR-MPLS and SRv6 protocol stacks are illustrated in Figure 1.

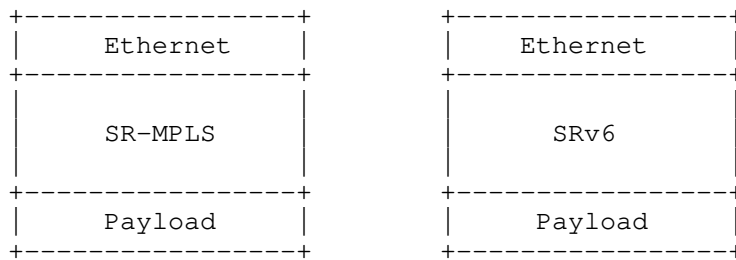


Figure 1: SR-MPLS and SRv6 Protocol Stacks

A segment identifier in SR-MPLS occupies 32 bits, and in SRv6 - 128 bits. As the backhaul infrastructure being upgraded to IPv6, operators are looking for technology that would reuse SR-MPLS by re-mapping the label table. But the namespace in SR-MPLS is limited and couldn't build the new segment identifiers to the global network. Using U-SID with SRv6 allows the reuse of the 32-bit SIDs, which are the same as in SR-MPLS. Thus, U-SID with SRv6 can be reused in backhaul to minimize the impact on existing SR-MPLS services and support smooth rollout of SRv6. The only additional task is to assign U-SIDs to the SRv6 domain. The controller could create an end-to-end SR tunnel using 32bit-long segments identifiers to stitch the SR-MPLS and SRv6 domains.

3.2. Interworking Between SRv6 and SR-MPLS

For a 5G backhaul network, the operators want to try their best to reuse the existing transport network. Consequently, they must consider the SRv6 interworking with SR-MPLS while deploying SRv6. Using U-SID offers a practical approach to native interworking between SR-MPLS and SRv6 domains because an operator in both domains can use segment identifiers of the same format, U-SID.

Using U-SID interworking between SRv6 and SR-MPLS brings some significant advantages:

1. An end-to-end LSP can be created across the access/aggregation network with SR-MPLS and core network with SRv6.
2. An end-to-end OAM and protection mechanism can be supported reusing SR-MPLS

The SR-MPLS and SRv6 interworking is illustrated in Figure 2. An end-to-end SR tunnel from A to F crosses the SR-MPLS and SRv6 domains. Using U-SID end-to-end LSP can reuse SR-MPLS forwarding, and support end-to-end OAM and protection.

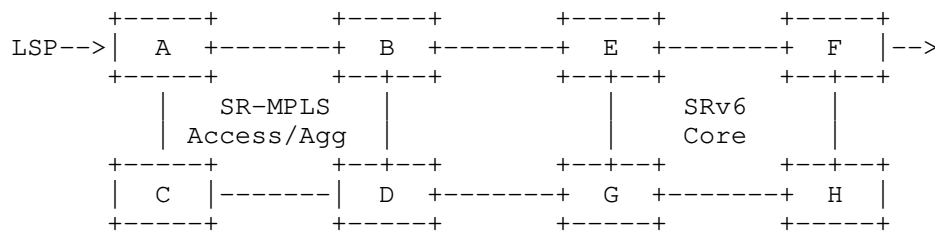


Figure 2: SR-MPLS and SRv6 Interworking

3.3. Compressing SRv6 Header Effectively

While deploying SRv6 in the backhaul network, the SRv6 header overhead must be considered. Typically there a maximum of ten hops for an end-to-end transport path. The header overhead is 1280 bits (10*128 bit SRH) using SRH with the 128-bit SID without OAM and protection. It will be reduced to 320 bits (3*128 bit SRH) using U-SID SRv6 with 32-bit SID. So the compressing rate is more than 70% (from at least 10*128 bit SRH to 3*128 bit SRH).

3.4. Support a Super-large-scale Networking and Flexibility in Assigning Addresses

The scale of the backhaul network is up to 10K nodes. A network of such size needs to support to address up to 10K nodes. U-SID SRv6 can support the 2^{20} labels as the same with MPLS, and it's enough for a super-large-scale backhaul networking. Since IPv6 solves the problem of a shortage of IPv4 addresses, it should not be using a shorter IPv6 address, i.e., a shorter prefix plus a shorter offset. That will violate the original IPv6 design. On the other hand, using SRv6 should not require the assignment of special addresses for the operator's network. U-SID can preserve the full 128-bit addresses by re-mapping the table. To use U-SID in SRv6 doesn't require the IPv6 address and SRv6 segments planning, such as the address prefix allocation. The operator would reuse the current address assignment and planning, thus minimizing the impact on the backhaul network.

4. Operations with Unified Segment Identifier

When the SRH is used to include 20-bits or 32-bits U-SIDs the ingress and transit nodes of an SR tunnel act as described in Section 5.1 and Section 5.2 of [I-D.ietf-6man-segment-routing-header] respectively.

5. IANA Considerations

This document has no requests to IANA. This section can be removed before the publication.

6. Security Considerations

This specification inherits all security considerations of [RFC8402] and [I-D.ietf-6man-segment-routing-header].

7. Acknowledgements

TBD

8. 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-26 (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.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.
- [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>>.

Authors' Addresses

Cheng Weiqiang
China Mobile
Beijing
China

Email: chengweiqiang@chinamobile.com

Peng Shaofu
ZTE Corporation
No.50 Software Avenue, Yuhuatai District
Nanjing
China

Email: peng.shaofu@zte.com.cn

Liu Aihua
ZTE Corporation
Zhongxing Industrial Park, Nanshan District
Shenzhen
China

Email: liu.aihua@zte.com.cn

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Wan Xiaolan
New H3C Technologies Co. Ltd
No.8, Yongjia Road, Haidian District
Beijing
China

Email: wxlan@h3c.com

Cheng Wei
Centec
Building B, No.5 Xing Han Street, Suzhou Industrial Park
Suzhou
China

Email: Chengw@centecnetworks.com

Shay
Broadcom
Israel

Email: shay.zadok@broadcom.com