

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 27, 2022

J. Dong
Z. Li
Huawei Technologies
C. Xie
C. Ma
China Telecom
G. Mishra
Verizon Inc.
October 24, 2021

Carrying Virtual Transport Network (VTN) Identifier in IPv6 Extension
Header
draft-dong-6man-enhanced-vpn-vtn-id-06

Abstract

Virtual Private Networks (VPNs) provide different customers with logically separated connectivity over a common network infrastructure. With the introduction and evolvement of 5G and other network scenarios, some existing or new customers may require connectivity services with advanced characteristics comparing to traditional VPNs. Such kind of network service is called enhanced VPNs (VPN+).

A Virtual Transport Network (VTN) is a virtual underlay network which consists of a set of dedicated or shared network resources allocated from the physical underlay network, and is associated with a customized logical network topology. VPN+ services can be delivered by mapping one or a group of overlay VPNs to the appropriate VTNs as the virtual underlay. In packet forwarding, some fields in the data packet needs to be used to identify the VTN the packet belongs to, so that the VTN-specific processing can be performed on each node the packet traverses.

This document proposes a new Hop-by-Hop option of IPv6 extension header to carry the VTN Resource ID, which is used to identify the set of network resources allocated to a VTN for packet processing. The procedure for processing the VTN option is also specified.

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 27, 2022.

Copyright Notice

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

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include 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	4
2. New IPv6 Extension Header Option for VTN	4
3. Procedures	5
3.1. VTN Option Insertion	5
3.2. VTN based Packet Forwarding	5
4. Operational Considerations	6
5. IANA Considerations	6
6. Security Considerations	6
7. Contributors	6
8. Acknowledgements	7
9. References	7
9.1. Normative References	7
9.2. Informative References	7
Authors' Addresses	8

1. Introduction

Virtual Private Networks (VPNs) provide different customers with logically isolated connectivity over a common network infrastructure. With the introduction and evolvement of 5G and other network

scenarios, some existing or new customers may require connectivity services with advanced characteristics comparing to traditional VPNs, such as resource isolation from other services or guaranteed performance. Such kind of network service is called enhanced VPN (VPN+). VPN+ service requires the coordination and integration between the overlay VPNs and the network characteristics of the underlay.

[I-D.ietf-teas-enhanced-vpn] describes a framework and the candidate component technologies for providing VPN+ services. It also introduces the concept of Virtual Transport Network (VTN). A Virtual Transport Network (VTN) is a virtual underlay network which consists of a set of dedicated or shared network resources allocated from the physical underlay network, and is associated with a customized logical network topology. VPN+ services can be delivered by mapping one or a group of overlay VPNs to the appropriate VTNs as the underlay, so as to provide the network characteristics required by the customers. In packet forwarding, traffic of different VPN+ services need to be processed separately based on the network resources and the logical topology associated with the corresponding VTN.

[I-D.dong-teas-enhanced-vpn-vtn-scalability] describes the scalability considerations and the possible optimizations for providing a relatively large number of VTNs for VPN+ services. One approach to improve the data plane scalability of VTN is to introduce a dedicated VTN Resource Identifier (VTN Resource ID) in the data packet to identify the set of network resources allocated to a VTN, so that VTN-specific packet processing can be performed using that set of resources, which avoids the possible resource competition with services in other VTNs. This is called Resource Independent (RI) VTN. A VTN Resource ID represents a subset of the resources (e.g. bandwidth, buffer and queuing resources) allocated on a given set of links and nodes which constitute a logical network topology. The logical topology associated with a VTN could be defined using mechanisms such as Multi-Topology [RFC4915], [RFC5120] or Flex-Algo [I-D.ietf-lsr-flex-algo], etc.

This document proposes a mechanism to carry the VTN resource ID in a new Hop-by-Hop option of IPv6 extension header [RFC8200] of IPv6 packet, so that on each network node along the packet forwarding path, the VTN option in the packet is parsed, and the obtained VTN Resource ID is used to instruct the network node to use the set of network resources allocated to the corresponding VTN to process and forward the packet. The procedure for processing the VTN Resource ID is also specified. This provides a scalable solution to support a relatively large number of VTNs in an IPv6 network.

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 BCP14 RFC 2119 [RFC2119] RFC 8174 [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. New IPv6 Extension Header Option for VTN

A new Hop-by-Hop option type "VTN" is defined to carry the VTN related Identifier in an IPv6 packet. Its format is shown as below:

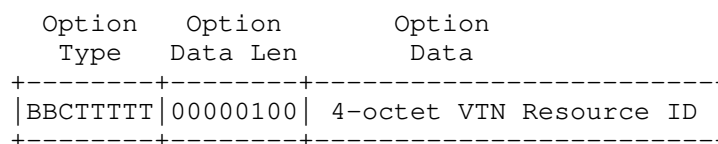


Figure 1. The format of VTN Option

Option Type: 8-bit identifier of the type of option. The type of VTN option is to be assigned by IANA. The highest-order bits of the type field are defined as below:

- o BB 00 The highest-order 2 bits are set to 00 to indicate that a node which does not recognize this type will skip over it and continue processing the header.
- o C 0 The third highest-order bit are set to 0 to indicate this option does not change en route.

Opt Data Len: 8-bit unsigned integer indicates the length of the option Data field of this option, in octets. The value of Opt Data Len of VTN option SHOULD be set to 4.

VTN Resource ID: 4-octet identifier which uniquely identifies the set of network resources allocated to a VTN.

Editor's note: The length of the VTN Resource ID is defined as 4-octet in correspondence to the 4-octet Single Network Slice Selection Assistance Information (S-NSSAI) defined in 3GPP [TS23501].

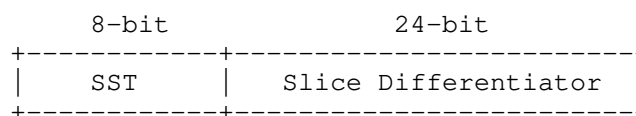


Figure 2. The format of S-NSSAI

3. Procedures

As the VTN option needs to be processed by each node along the path for VTN-specific forwarding, it SHOULD be carried in IPv6 Hop-by-Hop options header when the Hop-by-Hop options header can be either processed or ignored in forwarding plane by all the nodes along the path.

3.1. VTN Option Insertion

When an ingress node of an IPv6 domain receives a packet, according to the traffic classification or mapping policy, the packet is steered into one of the VTNs in the network, then the packet SHOULD be encapsulated in an outer IPv6 header, and the Resource ID of the VTN which the packet is mapped to SHOULD be carried in the VTN option of the Hop-by-Hop options header associated with the outer IPv6 header.

3.2. VTN based Packet Forwarding

On receipt of a packet with the VTN option, each network node which can process the VTN option in fast path SHOULD use the VTN Resource ID to determine the set of local network resources allocated to the VTN for packet processing. The packet forwarding behavior is based on both the destination IP address and the VTN Resource ID. More specifically, the destination IP address is used to determine the next-hop and the outgoing interface, and VTN Resource ID is used to determine the set of network resources on the outgoing interface which are reserved to the VTN for processing and sending the packet. The Traffic Class field of the outer IPv6 header MAY be used to provide Diffserv treatment for packets which belong to the same VTN. The egress node of the IPv6 domain SHOULD decapsulate the outer IPv6 header which includes the VTN option.

In the forwarding plane, there can be different approaches of partitioning the local network resources and allocating them to different VTNs. For example, on one physical interface, a subset of the forwarding plane resources (e.g. the bandwidth and the associated buffer and queuing resources) can be allocated to a particular VTN and represented as a virtual sub-interface with reserved bandwidth resource. In packet forwarding, the IPv6 destination address of the received packet is used to identify the next-hop and the outgoing layer-3 interface, and the VTN Resource ID is used to further identify the virtual sub-interface which is associated with the VTN on the outgoing interface.

Network nodes which do not support the processing of Hop-by-Hop options header SHOULD ignore the Hop-by-Hop options header and

forward the packet only based on the destination IP address. Network nodes which support Hop-by-Hop Options header, but do not support the VTN option SHOULD ignore the VTN option and continue to forward the packet based on the destination IP address and MAY also based on the rest of the Hop-by-Hop Options.

4. Operational Considerations

As described in [RFC8200], network nodes may be configured to ignore the Hop-by-Hop Options header, and in some implementations a packet containing a Hop-by-Hop Options header may be dropped or assigned to a slow processing path. The proposed modification to the processing of IPv6 Hop-by-Hop options header is specified in [I-D.hinden-6man-hbh-processing]. Operator needs to make sure that all the network nodes involved in a VTN can either process Hop-by-Hop Options header in the fast path, or ignore the Hop-by-Hop Option header. Since a VTN is associated with a logical network topology, it is practical to ensure that all the network nodes involved in that logical topology support the processing of the HBH options header and the VTN option. In other word, packets steered into a VTN MUST NOT be dropped due to the existence of the Hop-by-Hop Options header. It is RECOMMENDED to configure all the network nodes involved in a VTN to process the Hop-by-Hop Options header and the VTN option if there is a nob for this.

5. IANA Considerations

This document requests IANA to assign a new option type from "Destination Options and Hop-by-Hop Options" registry.

Value	Description	Reference
TBD	VTN Option	this document

6. Security Considerations

The security considerations with IPv6 Hop-by-Hop options header are described in [RFC8200], [RFC7045] and [I-D.hinden-6man-hbh-processing]. This document introduces a new IPv6 Hop-by-Hop option which is either processed in the fast path or ignored by network nodes, thus it does not introduce additional security issues.

7. Contributors

Zhibo Hu
Email: huzhibo@huawei.com

Lei Bao
Email: baolei7@huawei.com

8. Acknowledgements

The authors would like to thank Juhua Xu, James Guichard, Joel Halpern and Tom Petch for their review and valuable comments.

9. References

9.1. Normative References

- [I-D.ietf-teas-enhanced-vpn]
Dong, J., Bryant, S., Li, Z., Miyasaka, T., and Y. Lee, "A Framework for Enhanced Virtual Private Network (VPN+) Services", draft-ietf-teas-enhanced-vpn-08 (work in progress), July 2021.
- [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>>.

9.2. Informative References

- [I-D.dong-teas-enhanced-vpn-vtn-scalability]
Dong, J., Li, Z., Gong, L., Yang, G., Guichard, J. N., Mishra, G., and F. Qin, "Scalability Considerations for Enhanced VPN (VPN+)", draft-dong-teas-enhanced-vpn-vtn-scalability-03 (work in progress), July 2021.
- [I-D.hinden-6man-hbh-processing]
Hinden, R. M. and G. Fairhurst, "IPv6 Hop-by-Hop Options Processing Procedures", draft-hinden-6man-hbh-processing-01 (work in progress), June 2021.

- [I-D.ietf-lsr-flex-algo]
Psenak, P., Hegde, S., Filsfils, C., Talaulikar, K., and
A. Gulko, "IGP Flexible Algorithm", draft-ietf-lsr-flex-
algo-17 (work in progress), July 2021.
- [RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P.
Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF",
RFC 4915, DOI 10.17487/RFC4915, June 2007,
<<https://www.rfc-editor.org/info/rfc4915>>.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi
Topology (MT) Routing in Intermediate System to
Intermediate Systems (IS-IS)", RFC 5120,
DOI 10.17487/RFC5120, February 2008,
<<https://www.rfc-editor.org/info/rfc5120>>.
- [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>>.
- [TS23501] "3GPP TS23.501", 2016,
<[https://portal.3gpp.org/desktopmodules/Specifications/
SpecificationDetails.aspx?specificationId=3144](https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3144)>.

Authors' Addresses

Jie Dong
Huawei Technologies
Huawei Campus, No. 156 Beiqing Road
Beijing 100095
China

Email: jie.dong@huawei.com

Zhenbin Li
Huawei Technologies
Huawei Campus, No. 156 Beiqing Road
Beijing 100095
China

Email: lizhenbin@huawei.com

Chongfeng Xie
China Telecom
China Telecom Beijing Information Science & Technology, Beiqijia
Beijing 102209
China

Email: xiechf@chinatelecom.cn

Chenhao Ma
China Telecom
China Telecom Beijing Information Science & Technology, Beiqijia
Beijing 102209
China

Email: machh@chinatelecom.cn

Gyan Mishra
Verizon Inc.

Email: gyan.s.mishra@verizon.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 24, 2021

T. Herbert
SiPanda
June 22, 2021

Limits on Sending and Processing IPv6 Extension Headers
draft-herbert-6man-eh-limits-00

Abstract

This specification defines various limits that may be applied to receiving, sending, and otherwise processing packets that contain IPv6 extension headers. The need for such limits is pragmatic to facilitate interoperability amongst hosts and routers in the presence of extension headers and thereby increasing the feasibility of deployment of extension headers.

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 24, 2021.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	2
1.1. Related work	3
1.2. Adherence to the Robustness Principle	4
1.2.1. Be conservative in what you send	4
1.2.2. Be liberal in what you receive	4
2. Overview of extension header limits	5
2.1. Types of nodes	5
2.2. Types of limits	6
2.2.1. Limits on extension header length	6
2.2.2. Limits on option length	6
2.2.3. Limits on number of extension headers	6
2.2.4. Limits on number of options	6
2.2.5. Limits on padding options	7
2.2.6. Limit on IPv6 header chain length	8
2.3. Requirements for extension header limits	11
3. Requirements	12
3.1. Host requirements	12
3.1.1. Sending extension headers	12
3.1.2. Receiving extension headers	13
3.2. Intermediate node and intermediate destination requirements	15
3.3. Intermediate destination requirements	16
4. References	17
4.1. Normative References	17
4.2. Informative References	18
Author's Address	18

1. Introduction

Extension headers are a core component of the IPv6 protocol as specified in [RFC8200]. IPv6 extension headers were originally defined with few restrictions. For instance, there is no specified limit on the number of extension headers a packet may have, nor is there a limit on the length in bytes of extension headers in a packet (other than being limited by the MTU). Similarly, variable length extension headers typically do not have prescribed limits such as limits on the number of Hop-by-Hop or Destination options in a packet. The lack of limits essentially requires implementations to handle every conceivable usage of the protocol, including a myriad of use cases those are obviously outside the realm of ever being realistic or useful in real world deployment.

The lack of limits and the requirements for supporting virtually open-ended protocol have led to a significant lack of support and deployment of extension headers [RFC7872]. Instead of attempting to satisfy the protocol requirements concerning extension headers, some router and middlebox vendors have opted to either invent and apply their own ad hoc limits, relegate packets with extension headers to slow path processing, or have gone so far as to summarily discard all packets with extension headers. The net result of this situation is that deployment and use of extension headers is underwhelming to the extent that they are often considered unusable, and hence IPv6 extension headers have not lived up to their potential as the extensibility mechanism of IPv6.

As an example, consider that Hop-by-Hop Options and Destination Options have no limit on how many options may be placed in a packet nor any limits as to how many options a receiver must process. A single 1500 byte MTU sized packet could legally contain a Hop-by-Hop Options extension header with over seven hundred two byte options. There is no use case for this other than being a Denial of Service attack where an attacker simply creates packets with hundreds of small unknown Hop-by-Hop Options with the two high order bits in the option type set to 00 meaning to skip the unknown option. Any node in that path that attempts to dutifully process all these options per the requirements of [RFC8200] would be easily overwhelmed by the processing needed to parse these options (this is true for both hardware or software implementations).

This specification describes various limits that hosts and intermediate nodes may apply to the processing of extension headers. The goal of establishing limits is to narrow the requirements to better match reasonable use cases thereby facilitating practical implementation. Subsequently, this increases the viability of extension headers as the extensibility mechanism of IPv6.

1.1. Related work

Some of the problems of unlimited extension headers have been addressed in certain aspects.

[RFC8200] relaxed the requirement that all nodes in the path must process Hop-by-Hop Options to be:

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.

Section 5.3 of [RFC8504] defines a number of limits that hosts may apply to processing extensions. For instance:

A host MAY set a limit on the maximum number of non-padding options allowed in the destination options and Hop-by-Hop extension headers. If this feature is supported, the maximum number SHOULD be configurable, and the default value SHOULD be set to 8.

[RFC8883] defines a set of ICMP errors that may be sent if a limit concerning extension headers is exceeded and a node discards a packet as a result. This RFC allows both hosts and routers to send such messages (effectively acknowledging that some routers drop packets with extension headers even though such behavior is non-conformant).

[RFC7872] presents real-world data regarding the extent to which packets with IPv6 Extension Headers (EHs) are dropped in the Internet, and [I-D.gont-v6ops-ipv6-ehs-packet-drops] summarizes the operational implications of IPv6 extension headers, and attempts to analyze reasons why packets with IPv6 extension headers are often dropped in the public Internet.

1.2. Adherence to the Robustness Principle

The robustness principle, or Postel's Law, can be stated as "Be conservative in what you send, liberal in what you receive". This section considers the limits defined in this specification with respect to the robustness principle.

1.2.1. Be conservative in what you send

The limits on sending extension headers are well aligned with the send clause of the robustness principle. A sender of extension headers is generally constrained in its use of extension headers. Most of these limits are assumed to be the default to apply in an arbitrary environment such as the public Internet, that is they can be considered "baseline limits". These limits may be relaxed if a sender has a priori information that all possible nodes in path will properly handle packets that exceed the baseline limits. In particular, if a sender is sending in a limited domain, it might be known that all nodes in the limited domain have sufficient capabilities to handle packets exceeding the baseline limits.

1.2.2. Be liberal in what you receive

Considering the receive clause of the robustness principle, this specification recommends that receivers accept all packets with extension headers, however they may ignore extension headers or

options within extension headers. In particular, the philosophy of this specification is that intermediate nodes should not drop packets on the basis that they don't have sufficient capabilities to process all the headers in a packet. As such, intermediate nodes may define arbitrarily restrictive limits on what they process with regards to extension headers as long as the action taken when those limits are exceeded is to ignore items beyond the limit. Hosts are more constrained in this regard since they generally can't correctly process a packet without processing all the headers, so when limits are exceeded on a host, packets are dropped. It should be noted that hosts stacks inherently have more processing capabilities than intermediate nodes, so it is expected that they should be able to support higher limits.

This specification does specify one hard requirement for receiving nodes, namely nodes must be able to properly handle packets having an IPv6 header chain length up to 104 bytes. This requirement acknowledges that some intermediate nodes perform deep packet inspection at least to extract information from the transport layer headers. In this case, the data exceeding the limit may contain information that the node considers critical for correct processing, so that data cannot be ignored.

2. Overview of extension header limits

This specification considers extension header limits in three dimensions: 1) The types of nodes that may process extension headers and the requirements specific to each type, 2) The types of limits that may be applied, 3) The action taken when a limit is exceeded.

2.1. Types of nodes

For the purposes of describing handling of extension headers this specification considers three types of node in an IPv6 network:

- * Hosts: The source of an IPv6 packet, as addressed by the source address, or the final destination node of a packet as addressed by the destination address in a packet with no Routing header or as addressed by final segment in a Routing header.
- * Intermediate destination: An intermediate destination node in a Routing header as addressed by the destination address of a packet with a Routing header where the address is not the final destination in the Routing header
- * Intermediate nodes: A router on the path that is not addressed by the packet's destination address.

2.2. Types of limits

The limits and requirements for handling extension headers defined in this specification fall in the following categories:

- * Limits on extension header length
- * Limits on option length
- * Limits on number of extension headers
- * Limits on number of options
- * Limits on padding for extension headers with options
- * Limits on the length of the IPv6 header chain

2.2.1. Limits on extension header length

[RFC8504] defines limits that may be defined for the length of an extension header. Those limits are extended to be applicable to intermediate nodes. [RFC8883] defines ICMP Parameter Problem codes that may be sent when an extension header is exceeded.

2.2.2. Limits on option length

A node may establish a limit on the size Hop-by-Hop or Destination options. Conceivably, such a limit could apply to all option types, or length limits may be specific to individual options. [RFC8883] defines ICMP Parameter Problem codes that may be sent when an option length limit is exceeded.

2.2.3. Limits on number of extension headers

A node may define a limit on the number of extension headers it will process. Although [RFC8200] only defines four types of extension headers, it does not preclude the same type of extension header being present multiple times. A limit on the number of extension headers could be useful to disallow packets that contain multiple instances of the same extension header.

2.2.4. Limits on number of options

Limits may be established for the number of options sent or received (specifically applicable to Hop-by-Hop options and Destination options). The need for this limit arises from the fact that [RFC8200] does not specify a limit. Requiring nodes to process packets with tens or hundreds of options has no foreseeable use cases

in deployment except as a denial of service attack. [RFC8504] has proposed such a limit for host processing of Hop-by-Hop and Destination options with a default of eight options. This specification extends that limit to be applicable to intermediate nodes. Specific limits may be established for number of non-padding options or the number of all options including padding.

To derive a limit on all options, one can assume that at most one padding option is used between two non-padding options (an explicit limit on consecutive padding options is described below). With this assumption, we can extrapolate a reasonable limit on the number of all options that should be twice the limit of the number of non-padding options. Per [RFC8504], the recommended default limit for number of non-padding options is eight, so this specification establishes a maximum default limit of sixteen options including padding options. The choice of sixteen options as a default limit attempts to strike a balance between allowing extensibility and maintaining reasonable expectations for node processing requirements.

With regards to extensibility, it is observed that in the almost thirty year history of IPv6 there are only thirteen defined non-deprecated Destination options and Hop-by-Hop options and three temporary assigned options. Current evidence suggests that having more than one Destination option or Hop-by-Hop option in a packet is rare, and extrapolating that point with the rate of new options being defined suggests a limit of eight non-padding options allows for sufficient extensibility in the foreseeable future.

With regards to processing requirements, TLVs, e.g. Hop-by-Hop options and Destination options, have historically been considered difficult to process efficiently due to their serial processing requirements and combinatorial nature. TLV processing has been a particularly acute problem for ASIC devices. Recently, there is a strong trend in programmable implementation even in high performance routers using emerging programming frameworks such as PANDA and P4. Programmable implementations are better equipped to handle TLVs, at least for a reasonably small number. It might also be pointed out that the need to efficiently process TLVs exists in other protocols, for instance processing TCP requires processing of TLVs which are an intrinsic part of the protocol.

2.2.5. Limits on padding options

[RFC8200] defines PAD1 and PADN options that respectively provide one byte or N bytes of padding in an extension header. The purpose of padding is to properly align the following non-padding option to its expected alignment, or to add padding after the last Destination or Hop-by-Hop option so that the length of the extension header is a

multiple of eight bytes as required by [RFC8200]. [RFC8504] defines limits on number of bytes used for consecutive padding where the amount of padding between options or at the end of the extension header is no more than eight bytes; this limit is sufficient to align any following data after the padding to eight bytes. These limits are extended to be applicable to intermediate nodes.

This specification allows a receiving node to set a requirement that consecutive padding options are not present in a packet; which in turn requires a sender to not place consecutive padding options in a packet. The rationale for this limit is that a PAD1 or PADN option is able to provide one to 257 bytes of padding, so a single padding option is sufficient for any expected use case of padding. When the sender creates options, it can compute the amount of padding necessary to satisfy the alignment requirements of the following data. If one byte of padding is needed a PAD1 option is used, if more than one byte of padding is needed then an appropriate PADN options.

2.2.6. Limit on IPv6 header chain length

Intermediate nodes often perform deep packet inspection (DPI) in order to implement various functions in the network. Routers perform DPI when they inspect packets beyond the IPv6 header or beyond Hop-by-Hop options if they are present. Some router implementations must inspect the transport layer headers in order to process and forward the packet, and if the transport layer headers are not readable a packet may be dropped. Even if a transport layer header is in plain text within a packet, some devices may not be capable of reading it if the header is too deep in the packet.

Hardware devices often have constraints on how much of the headers in a packet can be parsed for DPI. A typical design is that some portion of the beginning of a received packet is loaded into a memory buffer for header parsing (i.e. the parsing buffer). The size of this parsing buffer is often fixed per device.

To derive a size limit on the IPv6 header chain, we need to take into account headers in a packet that might be subject to DPI which include the link layer header through at least the pertinent fields of the transport layer header. The most common required information is the transport layer port numbers which typically occupy the first four bytes of the transport headers (e.g. TCP, UDP, SCTP, DCCP, etc.). Inspection of port numbers may be needed for stateless load balancing as well as port filtering. There are middleboxes that may need to inspect more of transport layer headers or the transport payload, however those can be considered specialized devices that

perform work beyond simple packet forwarding and filtering and hence should have more capabilities for DPI.

In addition to limits on the length of the IP header chain, it is conceivable that there could be a limit on the length of the whole header chain. The whole header chain would comprise the IPv6 header chain as well as any headers that are part of network encapsulation that precede the innermost transport layer. The definition of such a limit is out of scope for this document, however [RFC8883] defines an ICMP error to send when a limit on size of an aggregate header chain is exceeded.

This document specifies that the minimum supported limit for IPv6 header chains is 104 bytes. The value is derived by assuming that nodes have the ability to process at least the first 128 bytes of a packet (that is they have a parsing buffer that can contain at least 128 bytes). The 128 byte parsing buffer would be expected to at least contain:

- * 16 bytes for a Layer 2 header (e.g. Ethernet header)
- * 40 bytes for the IPv6 header
- * 64 bytes for the extension headers
- * 8 bytes for the transport layer (i.e the first eight bytes of the transport layer header)

This scheme thus establishes a requirement that all Internet devices are capable of correctly processing packets with up to sixty-four bytes of extension headers, and subsequently it establishes a requirement that a host shouldn't send packets with more than sixty-four bytes of extension headers. Note that this establishes a global baseline requirement across the Internet, within a limited domain higher limits could be applied.

128 bytes is likely the minimal useful parsing buffer size in deployment today. Devices performing a very narrow DPI could conceptually use a smaller parsing buffer, for instance that could be as small as sixty-four bytes which accommodates an L2 header, IPv6 header, and eight bytes of transport header; however, such a device would be extremely limited in capabilities and if they do exist they are likely legacy devices that will eventually be decommissioned. Many routers now have the capability to perform DPI into encapsulation headers which implies they already have a larger parsing buffer than this baseline minimum.

Similar to limiting the number of options allowing in a packet, setting a limit for IP header length chain is a tradeoff between extensibility and feasible implementation.

For extensibility, the pertinent extension headers contributing to the sixty-four byte limit are mostly Hop-by-Hop and Destination options. The Routing Header extension header is really intended for limited domains and not the Internet (e.g. SRv6 Routing Header is confined to a Segment Routing Domain) and therefore would be subject to a domain specific limit for IP header chain length. Encryption Header may be used on the Internet, however encryption obfuscates the encapsulated transport headers such that intermediate nodes can't inspect them regardless of their position in a packet. Fragmentation may be used in the Internet, however only the first fragment of a fragmented packet might contain transport layer headers that could be read by an Intermediate node. In any case, the Fragment Header is only four bytes so that would not be a particularly large portion of a sixty-four byte limit.

The Authentication Header is usable on the Internet and does allow the transport layer headers to be in readable in plain text. However, Authentication Header is relatively large, typically thirty-two bytes or more, so it would contribute significantly to a limit on IP header chain length. On the other hand, the use of Authentication Header, without encryption, is currently rare on the Internet.

Individual Hop-by-Hop Destination Options may also be categorized as being intended for use over the Internet or just in limited domains. For instance, the IOAM Hop-by-Hop option is intended for use in limited domains.

Paring this down, the types extension headers and Destination and Hop-by-Hop options that might be used outside of limited domains are fairly limited. Options that are intended for use over the public Internet could be defined to be small and compact to promote not exceeding a sixty-four byte limit on extension headers, whereas options constrained to a limited domain could be larger since larger limits can be assumed.

2.2.6.1. Action when limit is exceeded

For each limit that is defined, an action is specified for when the limit is exceeded. The appropriate action depends on whether the processing node is the destination host, an intermediate destination, or an intermediate node. For a destination host, the typical action to take when a limit is exceeded is to discard the packet. This is appropriate since the destination host is required to process all of

the headers in a packet, and if a limit is exceeded then it cannot process the packet so there is no other alternative but to discard.

For intermediate nodes, the typical action to take when a limit is exceeded is to stop processing headers at the point the limit is reached and to forward the packet on. [RFC8200] allows that an intermediate may not process the Hop-by-Hop Options extension headers therefore an intermediate node may ignore all of the Hop-by-Hop options in a packet. This specification expands on that requirement to allow an intermediate node to process some arbitrary subset of consecutive Hop-by-Hop options in the TLV list and to ignore the following ones. In the case of an egregious violation of a limit, for instance an attacker sends three hundred options in a packet, the destination host can decide if the appropriate response is to drop (the destination host must process all options). Note that this provision motivates the sender to place Hop-by-Hop Options in the packet so that those considered more important are placed first. It should also be noted that [RFC8200] sets a default limit of eight; this specification adds a counterpart for sending hosts that they shouldn't send more than eight Hop-by-Hop options.

Intermediate destinations have characteristics of both hosts and intermediate nodes. If a limit is exceeded related to Hop-by-Hop options then the suggested action in this specification is to assume the same processing of limits as intermediate nodes. If limits are exceeded that affect the processing specific to an intermediate destination, such as limits on Destination options before the Routing header, then the action should be to discard packet.

2.3. Requirements for extension header limits

The set of limits that a node may apply when processing extension headers include:

- * Too many non-padding or padding options
- * Extension header too big
- * Option too big
- * Too many consecutive padding options
- * Too many consecutive bytes of padding
- * Extension header chain too long
- * Aggregate header chain too long

- * Too many extension headers

3. Requirements

This section lists the normative requirements related to sending and processing extension headers.

3.1. Host requirements

3.1.1. Sending extension headers

The requirements are:

- * A host MUST NOT send more than 8 non-padding options in Destination Options in a packet unless it has explicit knowledge that the destination, or all intermediate destinations in the case of Destination Options before the routing header, are able to process a greater number of options.
- * A host MUST NOT send more than 8 non-padding options in Hop-by-Hop Options in a packet unless it has explicit knowledge that the final destination host is able to process a greater number of options.
- * A host SHOULD NOT send more than 8 non-padding options in Hop-by-Hop Options in a packet unless it has explicit knowledge that all possible intermediate nodes are able to process a greater number of options or will ignore options that exceeds their limit.
- * A host MUST NOT send a packet with an extension header larger than 64 bytes unless it has explicit knowledge that all nodes that might process the extension header are capable of processing a larger header.
- * A host MUST NOT send a packet with a Destination option or Hop-by-Hop option with Data Length greater than 60 bytes unless it has explicit knowledge that all nodes that might process the option are capable of processing ones with a larger Data Length.
- * A host node MUST NOT send a packet with an IPv6 header chain larger than 104 bytes unless it has explicit knowledge that all nodes in the path are capable of properly handling packets with larger header chains. This requirements is equivalently stated as a host MUST NOT send a packet with more than 64 bytes of aggregate extension headers.
- * A host MUST NOT set more than one consecutive pad option, either PAD1 or PADN, in Destination options or Hop-by-Hop options.

- * A host MUST NOT send a PadN option in Hop-by-Hop Options or Destination Options with total length of more than seven bytes.
- * A host node MUST NOT send more than 16 options (padding or non-padding) Destination options in a packet unless it has explicit knowledge that the destination, or all intermediate destinations in the case of Destination Options before the routing header, are able to process a greater number of options. Note that if the above requirements on a host sending non-padding Destination options and requirements on option padding are met, then this requirement is implicitly satisfied.
- * A host node MUST NOT send more than 16 options (padding or non-padding) in Hop-by-Hop Options in a packet unless it has explicit knowledge that the final destination host is able to process a greater number of options. Note that if the above requirements on a host sending non-padding Hop-by-Hop options and requirements on padding are met, then this requirement is implicitly satisfied.

3.1.2. Receiving extension headers

Per [RFC8200], a host node that receives a packet with extension headers must process all the extension headers in the packet before accepting the payload and processing the payload.

As described in [RFC8504] a host may establish limits on the processing of extension headers. This specification reiterates and updates those requirements to allow for a host to send an RFC8883 error if a limit has been exceeded.

- * A host MAY set a limit on the maximum number of non-padding options allowed in the Destination Options or Hop-by-Hop Options extension headers. If this limit is supported then the maximum number SHOULD be configurable, the limit MUST be greater than or equal to 8, and the default value SHOULD be set to 8. The limits for Destination options and Hop-by-Hop options MAY be separately configurable. If a packet is received and the number of Destination or Hop-by-Hop options exceeds the limit, then the packet SHOULD be discarded and an ICMP Parameter Problem with code 9 MAY be sent to the packet's source address.
- * A host MAY set a limit on the maximum number of options (padding or non-padding) allowed in Destination Options or Hop-by-Hop Options extension headers. If this limit is supported then the maximum number SHOULD be configurable and the limit MUST be greater than or equal to 16. The limits for Destination options and Hop-by-Hop options MAY be separately configurable. If a packet is received and the number of destination or Hop-by-Hop

options exceeds the limit, then the packet SHOULD be discarded and an ICMP Parameter Problem with code 9 MAY be sent to the packet's source address

- * A host node MAY set a limit on the length of an extension header. If this limit is supported then the limit SHOULD be configurable and the limit MUST be greater than or equal to 64 bytes. The length limits for different extension headers MAY be separately configurable.
- * A host node MAY set a limit on the Data Length of a Hop-by-Hop or Destination option. If this limit is supported then the limit SHOULD be configurable, and the limit MUST be greater than or equal to 60 bytes. The limits for Destination options and Hop-by-Hop options MAY be separately configurable. If a packet is received and a Hop-by-Hop or destination option has a length that exceeds the limit, then the packet SHOULD be discarded and an ICMP Parameter Problem with code 10 MAY be sent to the packet's source address.
- * A host MAY limit the number of consecutive PAD1 options in destination options or Hop-by-Hop options to 7. In this case, if there are more than 7 consecutive PAD1 options present, the packet SHOULD be discarded and an ICMP Parameter Problem with code 10 MAY be sent to the packet's source address
- * A host MAY limit the number of bytes in a PADN option to be less than 8. In such a case, if a PADN option is present that has a length greater than 7, the packet SHOULD be discarded and an ICMP Parameter Problem with code 10 MAY be sent to the packet's source address.
- * A host MAY set a limit on the maximum length of Destination Options or Hop-by-Hop Options extension headers. This value SHOULD be configurable, and if the limit is used then the limit MUST be greater than or equal to 64 bytes. If a packet is received and the length of the Destination or Hop-by-Hop Options extension header exceeds the length limit, then the packet SHOULD be discarded and an ICMP Parameter Problem with code 6 MAY be sent to the packet's source address.
- * A host node MAY set a limit on the maximum length of the IPv6 header chain, or equivalently a host MAY set a limit on the aggregate length of extension headers in a packet. If the limit is used then it MUST be greater than or equal to 104 bytes, or, equivalently, the limit on aggregate header extension length MUST be greater than or equal to 64 bytes. If a packet is received and the aggregate length of the IPv6 header chain exceeds the limit

then the packet SHOULD be discarded and an ICMP Parameter Problem with code 7 MAY be sent to the packet's source address.

Additional host requirements for receive.

- * A host MAY disallow consecutive padding options, either PAD1 or PADN, to be present in a packet. If consecutive padding options are received and disallowed by the host, the then packet SHOULD be discarded and an ICMP Parameter Problem with code 9 MAY be sent to the packet's source address.

3.2. Intermediate node and intermediate destination requirements

The following requirements are established for intermediate nodes and intermediate destination nodes that receive and process packets with extension header.

- * An intermediate node MUST be able to correctly forward packets that contain an IPv6 header chain of 104 or fewer bytes, or equivalently an intermediate node MUST be able to process a packet with an aggregate length of extension headers less than or equal to 64 bytes.
- * Per [RFC8200] an intermediate node MAY be configured to not process Hop-by-Hop Options. If a node is configured as such and a packet with Hop-by-Hop options is received, the extension header MUST be skipped and the packet MUST otherwise be properly processed and forwarded.
- * An intermediate node MAY limit the number of non-padding Hop-by-Hop options that it processes. If a limit is exceeded, that is a packet contains more non-padding options than are configured to process, the intermediate SHOULD stop processing the Hop-by-Hop Option and ignore any options in the chain beyond the limit. It is NOT RECOMMENDED that an intermediate node discards the packet because the limit is exceeded, however if it does so then the intermediate node MAY send an ICMP Parameter Problem with code 10 MAY be sent to the packet's source address.
- * An intermediate node MAY limit the number of Hop-by-Hop options (padding or non-padding) that it processes. If a limit is exceeded, that is a packet contains more non-padding options than are configured to process, the intermediate SHOULD stop processing the Hop-by-Hop options and ignore any options in the chain beyond the limit. It is NOT RECOMMENDED that the intermediate node discards the packet because the limit is exceeded, however if it does so then the intermediate node MAY send an ICMP Parameter Problem with code 10 MAY be sent to the packet's source address.

- * If an intermediate node encounters an unknown Hop-by-Hop option and the two high order bits are not 00 then the node SHOULD immediately stop processing the option chain and ignore any options in the chain beyond the unknown option. An intermediate node MAY either elect to discard the packet and MAY send an ICMP Parameter Problem per the requirements of [RFC8200]; or the intermediate node MAY forward the packet.
- * An intermediate node MAY set a limit on the maximum length of Hop-by-Hop Options extension headers. This value SHOULD be configurable. If this limit is exceeded, that is a packet has an extension header larger than the limit, then the intermediate SHOULD stop processing the Hop-by-Hop Option and ignore any options in the chain beyond the limit. It is NOT RECOMMENDED that the intermediate node discards the packet because the limit is exceeded, however if it does so then the intermediate node MAY send an ICMP Parameter Problem with code 10 MAY be sent to the packet's source address.

3.3. Intermediate destination requirements

The following are requirements specific to intermediate destinations pertaining to the processing of Destination Options before the Routing header.

- * An intermediate destination MAY set a limit on the maximum length of Destination Options extension header before the Routing header. This value SHOULD be configurable, and the default is to accept options of any length. If a limit is defined is MUST be at least 64 bytes. If the limit is exceeded then the intermediate destination SHOULD discard the packet and MAY send an ICMP Parameter Problem with code 6 to the packet's source address.
- * An intermediate destination node MAY limit the number of non-padding options in Destination Options before the Routing header. If a limit is exceeded, that is a packet contains more non-padding options than are configured to process, the intermediate destination node SHOULD discard the packet and MAY send an ICMP Parameter Problem with code 10 to the packet's source address.
- * An intermediate destination node MAY limit the number of options (padding or non-padding) in Destination Options before the Routing header. If a limit is exceeded, that is a packet contains more non-padding options than are configured to process, the intermediate destination node SHOULD discard the packet and MAY send an ICMP Parameter Problem with code 10 to the packet's source address.

- * An intermediate destination MAY limit the total number bytes in consecutive PAD1 options in destination options before the Routing Header 7. If the limit is exceeded, that is there are more than seven bytes in consecutive PAD1 or PADN options present, the intermediate destination node SHOULD discard the packet and MAY send an ICMP Parameter Problem with code 10 to the packet's source address.
- * A intermediate destination MAY limit the number of bytes in a PADN option in Destination Options before the Routing header to be less than 8. In such a case, if a PADN option is present that has a length greater than 7, the packet SHOULD be discarded and the intermediate destination node SHOULD discard the packet and MAY send an ICMP Parameter Problem with code 10 to the packet's source address.
- * A intermediate destination MAY set a limit on the maximum number of non-padding options allowed in Destination options before the Routing header. If this feature is supported, the maximum number SHOULD be configurable, and the default value SHOULD be set to 8. If a packet is received and the number of Destination options before the Routing header exceeds the limit, the intermediate destination node SHOULD discard the packet and MAY send an ICMP Parameter Problem with code 10 to the packet's source address.
- * A intermediate MAY set a limit on the maximum length of Destination Options extension header before the Routing header. This value SHOULD be configurable, and the default is to accept options of any length. If a packet is received and the length of the Destination or Hop-by-Hop Options extension header exceeds the length limit, the intermediate destination node SHOULD discard the packet and MAY send an ICMP Parameter Problem with code 10 to the packet's source address.

4. References

4.1. 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>>.

- [RFC8504] Chown, T., Loughney, J., and T. Winters, "IPv6 Node Requirements", BCP 220, RFC 8504, DOI 10.17487/RFC8504, January 2019, <<https://www.rfc-editor.org/info/rfc8504>>.
- [RFC8883] Herbert, T., "ICMPv6 Errors for Discarding Packets Due to Processing Limits", RFC 8883, DOI 10.17487/RFC8883, September 2020, <<https://www.rfc-editor.org/info/rfc8883>>.

4.2. Informative References

- [I-D.gont-v6ops-ipv6-ehs-packet-drops]
Gont, F., Hilliard, N., Doering, G., Kumari, W., and G. Huston, "Operational Implications of IPv6 Packets with Extension Headers", draft-gont-v6ops-ipv6-ehs-packet-drops-04 (work in progress), July 2020.
- [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>>.

Author's Address

Tom Herbert
SiPanda
Santa Clara, CA
USA

Email: tom@sipanda.io

Network Working Group
Internet-Draft
Updates: 8200 (if approved)
Intended status: Standards Track
Expires: 4 December 2021

R. Hinden
Check Point Software
G. Fairhurst
University of Aberdeen
2 June 2021

IPv6 Hop-by-Hop Options Processing Procedures
draft-hinden-6man-hbh-processing-01

Abstract

This document specifies procedures for how IPv6 Hop-by-Hop options are processed. It modifies the procedures specified in the IPv6 Protocol Specification (RFC8200) to make processing of IPv6 Hop-by-Hop options practical with the goal of making IPv6 Hop-by-Hop options useful to deploy and use in the Internet. When published, this document updates RFC8200.

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 4 December 2021.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	2
2. Requirements Language	2
3. Terminology	2
4. Background	3
5. Hop-by-Hop Header Processing Procedures	5
5.1. Hop-by-Hop Options Per Packet	6
5.2. Hop-by-Hop Headers Processing	6
5.3. Router Alert Option	7
5.4. Configuration	8
6. New Hop-by-Hop Options	9
7. IANA Considerations	9
8. Security Considerations	10
9. Acknowledgments	11
10. Change log [RFC Editor: Please remove]	11
11. Normative References	11
12. Informative References	12
Authors' Addresses	12

1. Introduction

This document specifies procedures for how IPv6 Hop-by-Hop options are processed. It modifies the procedures specified in the IPv6 Protocol Specification (RFC8200) to make processing of IPv6 Hop-by-Hop options practical with the goal of making IPv6 Hop-by-Hop options useful to deploy and use in the Internet.

When published this document updates [RFC8200].

The current list of defined Hop-by-Hop options can be found at [IANA-HBH].

2. Requirements Language

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

3. Terminology

This document uses the following loosely defined terms:

- * Forwarding Plane: IPv6 hosts exchange user data through the forwarding plane. User data is processed by its recipient (i.e., an IPv6 host). User data can traverse intermediate nodes (i.e., routers) between its source and its destination. These intermediate nodes process metadata contained in packet headers. However, they do not process information contained in packet payloads.
- * Control Plane: IPv6 routers exchange management and routing information with controllers. They also exchange routing information with one another. Management and routing information is processed by its recipient (i.e., an IPv6 router or controller). Management and control information can traverse intermediate nodes (i.e., routers) between its source and its destination. These intermediate nodes process metadata contained in packet headers. However, they do not process information contained in packet payloads. So, from their perspective, this information is user data.
- * Fast Path: A path through a router that is optimized for forwarding packets without processing their payloads. The Fast Path may be supported by Application Specific Integrated Circuits (ASICs), Network Processor (NP), or other special purpose hardware. This is the usual processing path within a router taken by the forwarding plane.
- * Slow Path: A path through a router that is capable of general purpose processing and is not optimized for any particular function. This processing path is used for packets that require special processing or differ from assumptions made in Fast Path heuristics, or to process router control protocols used by the control plane.

NOTE: This distinct separation between hardware and software processing from [RFC6398] does not apply to all router architectures. However, a router that performs all or most processing in software might still incur more processing cost when providing special processing (aka Slow Path).

[RFC6192] is an example of how designs can separate control plane (Slow Path) and forwarding plane (Fast Path) functions.

4. Background

In the first version of the IPv6 specification, Hop-by-Hop options were required to be processed by all nodes: routers and hosts. This proved to not be practical in high speed routers due to several factors, including:

- * Inability to process the hop-by-hop options at wire speed on the Fast Path.
- * Hop-by-Hop options would be sent to the Slow Path. This could degrade the a router's performance and it's ability to process important control traffic.
- * A mechanism that forces packets from any source to the routers "Slow Path" could be exploited as a Denial of Service attack against the router.
- * Packets could contain multiple Hop-by-Hop options making the previous issues worse by increasing the complexity required to process them.

When the IPv6 Specification was updated and published in July 2017 as [RFC8200], the procedures relating to hop-by-hop options were as follows:

Extension headers (except for the Hop-by-Hop Options header) are not processed, inserted, or deleted by any node along a packet's delivery path, until the packet reaches the node (or each of the set of nodes, in the case of multicast) identified in the Destination Address field of the IPv6 header.

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 changes meant that an implementation complied with the IPv6 specification even if it did not process hop-by-hop options, and that it was expected that routers would add configuration information to control which hop-by-hop options they would process.

Unfortunately, this did not improve the processing of Hop-by-Hop options and did not significantly improve deployment and use in the Internet. Essentially, it only documented how they were being used in the Internet at the time RFC8200 was published.

The main issues remain:

- * Routers are commonly configured to drop transit packets containing hop-by-hop options that would have be processed in the Slow Path. This behavior is seen as protecting against a denial of service attack on the router. This is discussed in [I-D.ietf-v6ops-ipv6-ehs-packet-drops].
- * Allowing multiple hop-by-hop options in a single packet makes it even more expensive in router resources to process these packets. It adds complexity to the number of permutations that might need to be processed.
- * Any mechanism that can be used to force packets into the router's Slow Path can be exploited as a denial of service attack on a transit router by saturating the resources needed for router management protocols (e.g., routing protocols, network management protocols, etc.) that may cause the router to fail. This issue for the Router Alert option, which intentionally places packets on the Slow Path, is discussed in [RFC6398]. Section 3 of that RFC includes a good summary:

"In a nutshell, the IP Router Alert Option does not provide a convenient universal mechanism to accurately and reliably distinguish between IP Router Alert packets of interest and unwanted IP Router Alert packets. This, in turn, creates a security concern when the IP Router Alert Option is used, because, short of appropriate router-implementation-specific mechanisms, the router Slow Path is at risk of being flooded by unwanted traffic."

There has been research that discussed the general problem with dropping packets containing IPv6 extension headers, including the Hop-by-Hop Options header. For example [Hendriks] states that "dropping all packets with Extension Headers, is a bad practice", and that "The share of traffic containing more than one EH however, is very small. For the design of hardware able to handle the dynamic nature of EHs, we therefore recommend to support at least one EH".

This document defines a set of procedures for the hop-by-hop option header that make the processing of hop-by-hop options practical in modern transit routers.

5. Hop-by-Hop Header Processing Procedures

This section describes several changes to [RFC8200].

5.1. Hop-by-Hop Options Per Packet

The Hop-by-Hop Option Header as defined in Section 4.3 of [RFC8200] is identified by a Next Header value of 0 in the IPv6 header. Section 4.1 of [RFC8200] requires a Hop-by-Hop Options header to appear immediately after the IPv6 header. [RFC8200] also requires that a Hop-by-Hop Options header can only appear once in a packet.

The Hop-by-Hop Options Header as defined in [RFC8200] can contain one or more Hop-by-Hop options. This document updates [RFC8200] that a node MUST process the first Option in the Hop-by-Hop Header in the Fast Path and MAY process additional Hop-by-Hop Options if configured to do so. The motivation for this change is to simplify the processing of Hop-by-Hop options in the Fast Path.

Nodes creating packets with a Hop-by-Hop option headers SHOULD include a single Hop-by-Hop Option in the packet and MAY include more based on local configuration.

If there are more than one Hop-by-Hop options in the Hop-by-Hop Options header, the node MAY skip the rest of the options without having to examine these options using the "Hdr Ext Len" field in the Hop-by-Hop Options header. This field specifies the length of the Option Header in 8-octet units. The additional options do not need to be processed or verified.

5.2. Hop-by-Hop Headers Processing

Nodes that implement a differentiation between a Fast Path and a Slow Path MUST process all (with one exception noted below) Hop-by-Hop options in the Fast Path. The one exception to this is the Router Alert Option [RFC2711]. See Section 5.3 for discussion of the Router Alert.

If the node can not process an option in the Fast Path, it MUST behave as if it does not recognize the Option Type (as described in the next paragraph).

Section 4.2 of [RFC8200] defines the Option Type identifiers as internally encoded such that their highest-order 2 bits specify the action that must be taken if the processing IPv6 node does not recognize the Option Type. The text is:

- 00 - skip over this option and continue processing the header.
- 01 - discard the packet.
- 10 - discard the packet and, regardless of whether or not the packet's Destination Address was a multicast address, send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type.
- 11 - discard the packet and, only if the packet's Destination Address was not a multicast address, send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type.

This document modifies this behaviour for the "10" and "11" values that the node MAY send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type. The modified text for "10" and "11" values is:

- 10 - discard the packet and, regardless of whether or not the packet's Destination Address was a multicast address, MAY send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type.
- 11 - discard the packet and, only if the packet's Destination Address was not a multicast address, MAY send an ICMP Parameter Problem, Code 2, message to the packet's Source Address, pointing to the unrecognized Option Type.

The motivation for this change is to loosen the requirement to send ICMPv6 Parameter Problem messages to simplify what the router needs to do in the Fast Path when it does not recognize the Option Type.

When an ICMP Parameter Problem, Code 2, message is delivered to the source, the source can become aware that at least one node on the path has failed to recognize the option.

5.3. Router Alert Option

The Router Alert option [RFC2711] purpose is to tell the node that the packet needs additional processing on the Slow Path.

The Router Alert option includes a two octet Value field that describes the protocol that is carried in the packet. The current values can be found in the IANA Router Alert Value registry [IANA-RA].

DISCUSSION

The Router Alert Option is a problem since it's function is to do what this specification is proposing to eliminate, that is, process the packet in the Slow Path. One approach would be to deprecate it as it's usage appears to be limited and packets containing Hop-by-Hop options are frequently dropped. Deprecation would allow current implementations to continue and it's use could be phased out over time.

The authors current thinking is that the Router Alert function may have reasonable potential use for new functions that have to be processed in the Slow Path. We think that keeping it as the single exception for Slow Path processing with the following restrictions is a reasonable compromise to allow future flexibility. These are compatible with Section 5 of [RFC6398].

A Fast Path implementation SHOULD verify that a Router Alert contains a protocol, as indicated by the Value field in the Router Alert option, that is configured as a protocol of interest to that router. A verified packet SHOULD be sent on the Slow Path for processing [RFC6398]. Otherwise, the router implementation SHOULD forward within the Fast Path (subject to all normal policies and forwarding rules). As specified in [RFC2711] the top two bits of Option Type for the Router Alert option are always set to "00" indicating the node should skip over this option and continue processing the header in this case.

Implementations of the IP Router Alert Option SHOULD offer the configuration option to simply ignore the presence of "IP Router Alert" in IPv4 and IPv6 packets" [RFC6398].

A node that is configured to process a Router Alert option using the Slow Path MUST protect itself from infrastructure attack that could result from processing on the Slow Path. This might include some combination of access control list to only permit from trusted nodes, rate limiting of processing, or other methods [RFC6398].

5.4. Configuration

Section 4 of [RFC8200] allows for a router to control it's processing of IPv6 Hop-by-Hop options by local configuration. The text is:

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.

A possible approach to implementing this is to maintain a lookup table based on Option Type of the IPv6 options that are supported in the Fast Path. This would allow for a node to quickly determine if an option is supported and can be processed. If the option is not supported, then the node processes it as described in Section 5.2 of this document.

A node configured not to process HBH options, MUST drop the packet if the top two bits of the Option Type field of the first HBH option is non-zero.

The actions of the lookup table SHOULD be configurable by the operator of the router.

6. New Hop-by-Hop Options

Any new IPv6 Hop-by-Hop option designed in the future should be designed to be processed in the Fast Path. New options MUST NOT be defined that require Slow Path processing. New Hop-by-Hop options SHOULD have the following characteristics:

- * Straight forward to process. That is, they should be designed to keep the time to process low.
- * Fixed size in 8-octet units. Specifically any new Hop-by-Hop options should not be variable size that could extend beyond what can be executed in the Fast Path.

Any new Hop-by-Hop option that is standardized that does not meet these criteria needs to explain in detail in its specification why this can not be accomplished and that there is a reasonable expectation that it can be proceed in most Fast Path implementations.

7. IANA Considerations

There are no actions required for IANA defined in this document.

8. Security Considerations

Security issues with IPv6 Hop-by-Hop options are well known and have been documented in several places, including [RFC6398], [RFC6192], and [I-D.ietf-v6ops-ipv6-ehs-packet-drops]. The main issue, as noted in Section 4, is that any mechanism that can be used to force packets into the router's Slow Path can be exploited as a denial of service attack on a transit router by saturating the resources need for router management protocols (e.g., routing protocols, network management protocols, etc.) that may cause the router to fail. Due to this it's common for transit routers to drop packets with Hop-by-Hop options headers.

While Hop-by-Hop options are not required to be processed in the Slow Path, the Router Alert options is designed to do just that.

This document changes the way Hop-by-Hop options are processed in several ways that significantly reduces the attack surface. These changes include:

- * All Hop-by-Hop options (with one exception) must be processed in the Fast Path. Only one HBH Option MUST be processed and additional HBH Options MAY be processed based on local configuration.
- * Only the Router Alert option can be processed in the Slow Path, and the router must be configured to do so.
- * Added criteria to allow control over how Router Alert options are processed and that a node configured to support these options must protect itself from attacks using the Router Alert.
- * Limited the default number of Hop-by-Hop options that that can be in a packet to a single Hop-by-Hop option.
- * Additional Hop-by-Hop options MAY be included, based on local configuration. Although nodes only process these additional Hop-by-Hop Options if configured to do so.
- * Added restrictions to any future new Hop-by-Hop options that limit their size and computational requirements.

The authors believe that these changes significantly reduces the security issues relating to IPv6 Hop-by-Hop options and will enable them to be used safely in the Internet.

9. Acknowledgments

Helpful comments were received from Brian Carpenter, Ron Bonica, Ole Troan, Mark Heard, Tom Herbert, [your name here], and other members of the 6MAN working group.

10. Change log [RFC Editor: Please remove]

draft-hinden-6man-hbh-processing-01, 2021-June-2:

- * Expanded terminology section to include Forwarding Plane and Control Plane.
- * Changed draft that only one HBH Option MUST be processed and additional HBH Options MAY be processed based on local configuration.
- * Clarified that all HBH options (with one exception) must be processed on the Fast Path.
- * Kept the Router Alert options as the single exception for Slow Path processing.
- * Rewrote and expanded section on New Hop-by-Hop Options.
- * Removed requirement for HBH Option size and alignment.
- * Removed sections evaluating currently defined HBH Options.
- * Added content to the Security Considerations section.
- * Added people to the acknowledgements section.
- * Numerous editorial changes

draft-hinden-6man-hbh-processing-00, 2020-Nov-29:

- * Initial draft.

11. 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>>.

12. Informative References

- [Hendriks] Hendriks, L., Velan, P., Schmidt, R.O., Boer, P., and A. Aiko, "Threats and Surprises behind IPv6 Extension Headers", August 2017,
<http://dl.ifip.org/db/conf/tma/tma2017/tma2017_paper22.pdf>.
- [I-D.ietf-v6ops-ipv6-ehs-packet-drops] Gont, F., Hilliard, N., Doering, G., Kumari, W., Huston, G., and W. (. Liu, "Operational Implications of IPv6 Packets with Extension Headers", Work in Progress, Internet-Draft, draft-ietf-v6ops-ipv6-ehs-packet-drops-06, 8 April 2021, <<https://tools.ietf.org/html/draft-ietf-v6ops-ipv6-ehs-packet-drops-06>>.
- [IANA-RA] "IPv6 Router Alert Option Values",
<<https://www.iana.org/assignments/ipv6-routeralert-values/ipv6-routeralert-values>>.
- [RFC2711] Partridge, C. and A. Jackson, "IPv6 Router Alert Option", RFC 2711, DOI 10.17487/RFC2711, October 1999,
<<https://www.rfc-editor.org/info/rfc2711>>.
- [RFC6192] Dugal, D., Pignataro, C., and R. Dunn, "Protecting the Router Control Plane", RFC 6192, DOI 10.17487/RFC6192, March 2011, <<https://www.rfc-editor.org/info/rfc6192>>.
- [RFC6398] Le Faucheur, F., Ed., "IP Router Alert Considerations and Usage", BCP 168, RFC 6398, DOI 10.17487/RFC6398, October 2011, <<https://www.rfc-editor.org/info/rfc6398>>.

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: gorry@erg.abdn.ac.uk

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

R. Hinden
Check Point Software
G. Fairhurst
University of Aberdeen
July 5, 2019

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

Abstract

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

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

Status of This Memo

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

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

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

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

Copyright Notice

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

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

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

Table of Contents

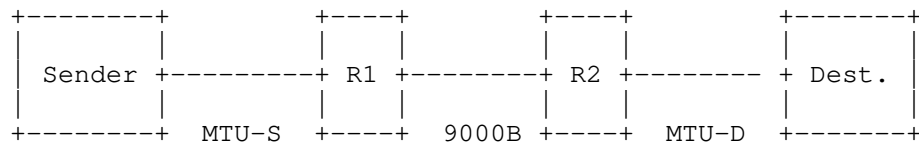
1. Introduction	2
2. Motivation and Problem Solved	4
3. Requirements Language	5
4. Applicability Statements	5
5. IPv6 Minimum Path MTU Hop-by-Hop Option	5
6. Router, Host, and Transport Behaviors	6
6.1. Router Behaviour	6
6.2. Host Behavior	7
6.3. Transport Behavior	9
7. IANA Considerations	10
8. Security Considerations	11
9. Acknowledgments	11
10. Change log [RFC Editor: Please remove]	11
11. References	12
11.1. Normative References	12
11.2. Informative References	13
Appendix A. Planned Experiments	13
Authors' Addresses	14

1. Introduction

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

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

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



The scenarios are described:

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

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

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

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

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

- o The Hop-by-Hop Options header is not inserted or deleted, but may be examined or processed by any node along a packet's delivery path, until the packet reaches the node (or each of the set of

nodes, in the case of multicast) identified in the Destination Address field of the IPv6 header. The Hop-by-Hop Options header, when present, must immediately follow the IPv6 header. Its presence is indicated by the value zero in the Next Header field of the IPv6 header.

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

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

2. Motivation and Problem Solved

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

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

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

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

The purpose of the this draft is to improve the situation by defining a mechanism that does not rely on nodes in the middle of the network to send ICMPv6 Packet Too Big messages, instead it provides the destination host information on the minimum Path MTU and it can send

this information back to the source host. This is expected to work better than the current RFC8201 based mechanisms.

3. Requirements Language

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

4. Applicability Statements

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

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

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

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

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

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

Option Type:

- BB 00 Skip over this option and continue processing.
- C 1 Option data can change en route to the packet's final destination.
- TTTT 11110 Experimental Option Type from [IANA-HBH].
- Length: 4 The size of the each value field in Option Data field supports Path MTU values from 0 to 65,535 octets.
- Min-PMTU: n 16-bits. The minimum PMTU in octets, reflecting the smallest link MTU that the packet experienced across the path. This is called the Reported PMTU. A value less than the IPv6 minimum link MTU [RFC8200] should be ignored.
- Rtn-PMTU: n 15-bits. The returned minimum PMTU, carrying the 15 most significant bits of the latest received Min-PMTU field. The value zero means that no Reported MTU is being returned.
- R n 1-bit. R-Flag. Set by the source to signal that the destination should include the received Reported PMTU in Rtn-PMTU field.

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

6. Router, Host, and Transport Behaviors

6.1. Router Behaviour

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

Routers that support Hop-by-Hop Options, but do not recognize this option SHOULD ignore the option and SHOULD forward the packet.

Routers that recognize this option SHOULD compare the Reported PMTU in the Min-PMTU field and the MTU configured for the outgoing link. If the MTU of the outgoing link is less than the Reported PMTU, the router rewrites the Reported PMTU in the Option to use the smaller value.

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

Discussion:

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

6.2. Host Behavior

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

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

Discussion:

- o This option does not need to be sent in all packets belonging to a flow. A transport protocol (or packetization layer) can set this option only on specific packets used to test the path.
- o In the case of TCP, the option could be included in packets carrying a SYN segment as part of the connection set up, or can periodically be sent in packets carrying other segments. Including this packet in a SYN could increase the probability that SYN segment is lost, when routers on the path drop packets with this option. Including this option in a large packet is not likely to be useful, since the large packet might itself also be dropped by a link along the path with a smaller MTU, preventing the Reported PMTU information from reaching the Destination node.
- o The use with datagram transport protocols (e.g. UDP) is harder to characterize because applications using datagram transports range from very short-lived (low data-volume applications) exchanges, to longer (bulk) exchanges of packets between the Source and Destination nodes [RFC8085].

- o For applications that use Anycast, this option should be included in all packets as the actual destination will vary due to the nature of Anycast.
- o Simple-exchange protocols (i.e low data-volume applications [RFC8085] that only send one or a few packets per transaction, could be optimized by assuming that the Path MTU is symmetrical, that is where the Path MTU is the same in both directions, or at least not smaller in the return path. This optimisation does not hold when the paths are not symmetric.
- o The use of this option with DNS and DNSSEC over UDP ought to work as long as the paths are symmetric. The DNS server will learn the Path MTU from the DNS query messages. If the return Path MTU is smaller, then the large DNSSEC response may be dropped and the known problems with PMTUD will occur. DNS and DNSSEC over transport protocols that can carry the Path MTU should work.

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

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

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

- o It is the first Reported PMTU value it has received from the Source.
- o The Reported PMTU value is lower than previously received.

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

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

For a connection-oriented upper layer protocol, this could be implemented by saving the value of the last received option within the connection context. This last received value is then used to set the return Path MTU field for all packets belonging to this flow that carry the IPv6 Minimum Path MTU Hop-by-Hop Option.

A connection-less protocol, e.g., based on UDP, requires the application to be updated to cache the Received PMTU value, and to ensure that this corresponding value is used to set the last Received PMTU in the Rtn-PMTU field of any PMTU Option that it sends.

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

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

Discussion:

- o A simple mechanism could only send an MTU Option with the Rtn-PMTU field filled in the first time this option is received or when the Received PMTU is reduced. This is good because it limits the number sent, but there is no provision for retransmission of the PMTU Option fails to reach the sender, or the sender loses state.
- o The Reported PMTU value could increase or decrease over time. For instance, it would increase when the path changes and the packets become then forwarded over a link with a MTU larger than the link previously used.

6.3. Transport Behavior

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

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

- o If the Received PMTU value returned by the Destination is the same as the initial Reported PMTU value, there could still be a router or layer 2 device on the path that does not support this PMTU. The usable PMTU therefore needs to be confirmed.

- o If the Received PMTU value returned by the Destination is smaller than the initial Reported PMTU value, this is an indication that there is at least one router in the path with a smaller MTU. There could still be another router or layer 2 device on the path that does not support this MTU.
- o If the Received PMTU value returned by the Destination is larger than the initial Reported PMTU value, this may be a corrupted, delayed or mis-ordered response, and SHOULD be ignored.

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

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

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

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

NOTE: Further details to be included in next version.

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

7. IANA Considerations

No IANA assignments are requested. Document uses experimental option from [IANA-HBH].

8. Security Considerations

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

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

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

TBD

9. Acknowledgments

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

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

10. Change log [RFC Editor: Please remove]

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

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

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

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

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

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

- o Initial draft.

11. References

11.1. Normative References

[IANA-HBH]

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

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

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

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

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

11.2. Informative References

- [RFC1063] Mogul, J., Kent, C., Partridge, C., and K. McCloghrie, "IP MTU discovery options", RFC 1063, DOI 10.17487/RFC1063, July 1988, <<https://www.rfc-editor.org/info/rfc1063>>.
- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <<https://www.rfc-editor.org/info/rfc1191>>.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <<https://www.rfc-editor.org/info/rfc2460>>.
- [RFC7637] Garg, P., Ed. and Y. Wang, Ed., "NVGRE: Network Virtualization Using Generic Routing Encapsulation", RFC 7637, DOI 10.17487/RFC7637, September 2015, <<https://www.rfc-editor.org/info/rfc7637>>.
- [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <<https://www.rfc-editor.org/info/rfc8085>>.

Appendix A. Planned Experiments

TBD

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

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

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

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

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

Authors' Addresses

Robert M. Hinden
Check Point Software
959 Skyway Road
San Carlos, CA 94070
USA

Email: bob.hinden@gmail.com

Godred Fairhurst
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen AB24 3UE
UK

Email: gorry@erg.abdn.ac.uk

6MAN Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 30, 2022

G. Fioccola
T. Zhou
Huawei
M. Cociglio
Telecom Italia
F. Qin
China Mobile
R. Pang
China Unicom
April 28, 2022

IPv6 Application of the Alternate Marking Method
draft-ietf-6man-ipv6-alt-mark-14

Abstract

This document describes how the Alternate Marking Method can be used as a passive performance measurement tool in an IPv6 domain. It defines a new Extension Header Option to encode Alternate Marking information in both the Hop-by-Hop Options Header and Destination Options 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 October 30, 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. Terminology	3
1.2. Requirements Language	3
2. Alternate Marking application to IPv6	3
2.1. Controlled Domain	5
2.1.1. Alternate Marking Measurement Domain	6
3. Definition of the AltMark Option	7
3.1. Data Fields Format	7
4. Use of the AltMark Option	8
5. Alternate Marking Method Operation	10
5.1. Packet Loss Measurement	10
5.2. Packet Delay Measurement	12
5.3. Flow Monitoring Identification	13
5.4. Multipoint and Clustered Alternate Marking	15
5.5. Data Collection and Calculation	16
6. Security Considerations	16
7. IANA Considerations	20
8. Acknowledgements	20
9. References	20
9.1. Normative References	20
9.2. Informative References	21
Authors' Addresses	22

1. Introduction

[I-D.ietf-ippm-rfc8321bis] and [I-D.ietf-ippm-rfc8889bis] describe a passive performance measurement method, which can be used to measure packet loss, latency and jitter on live traffic. Since this method is based on marking consecutive batches of packets, the method is often referred to as the Alternate Marking Method.

This document defines how the Alternate Marking Method can be used to measure performance metrics in IPv6. The rationale is to apply the Alternate Marking methodology to IPv6 and therefore allow detailed packet loss, delay and delay variation measurements both hop-by-hop and end-to-end to exactly locate the issues in an IPv6 network.

The Alternate Marking is an on-path telemetry technique and consists of synchronizing the measurements in different points of a network by

switching the value of a marking bit and therefore dividing the packet flow into batches. Each batch represents a measurable entity recognizable by all network nodes along the path. By counting the number of packets in each batch and comparing the values measured by different nodes, it is possible to precisely measure the packet loss. Similarly, the alternation of the values of the marking bits can be used as a time reference to calculate the delay and delay variation. The Alternate Marking operation is further described in Section 5.

The format of IPv6 addresses is defined in [RFC4291] while [RFC8200] defines the IPv6 Header, including a 20-bit Flow Label and the IPv6 Extension Headers.

This document introduces a new TLV (type-length-value) that can be encoded in the Options Headers (Hop-by-Hop or Destination) for the purpose of the Alternate Marking Method application in an IPv6 domain.

The threat model for the application of the Alternate Marking Method in an IPv6 domain is reported in Section 6. As with all on-path telemetry techniques, the only definitive solution is that this methodology MUST be applied in a controlled domain.

1.1. Terminology

This document uses the terms related to the Alternate Marking Method as defined in [I-D.ietf-ippm-rfc8321bis] and [I-D.ietf-ippm-rfc8889bis].

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. Alternate Marking application to IPv6

The Alternate Marking Method requires a marking field. Several alternatives could be considered such as IPv6 Extension Headers, IPv6 Address and Flow Label. But, it is necessary to analyze the drawbacks for all the available possibilities, more specifically:

Reusing existing Extension Header for Alternate Marking leads to a non-optimized implementation;

Using the IPv6 destination address to encode the Alternate Marking processing is very expensive;

Using the IPv6 Flow Label for Alternate Marking conflicts with the utilization of the Flow Label for load distribution purpose ([RFC6438]).

In the end, a new Hop-by-Hop or a new Destination Option is the best choice.

The approach for the Alternate Marking application to IPv6 specified in this memo is compliant with [RFC8200]. It involves the following operations:

- o The source node is the only one that writes the Option Header to mark alternately the flow (for both Hop-by-Hop and Destination Option). The intermediate nodes and destination node MUST only read the marking values of the option without modifying the Option Header.
- o In case of Hop-by-Hop Option Header carrying Alternate Marking bits, it is not inserted or deleted, but can be read by any node along the path. The intermediate nodes may be configured to support this Option or not and the measurement can be done only for the nodes configured to read the Option. As further discussed in Section 4, the presence of the hop-by-hop option should not affect the traffic throughput both on nodes that do not recognize this option and on the nodes that support it. However, it is worth mentioning that there is a difference between theory and practice. Indeed, in a real implementation it can happen that packets with hop-by-hop option could also be skipped or processed in the slow path. While some proposals are trying to address this problem and make Hop-by-Hop Options more practical ([I-D.peng-v6ops-hbh], [I-D.hinden-6man-hbh-processing]), these aspects are out of the scope for this document.
- o In case of Destination Option Header carrying Alternate Marking bits, it is not processed, inserted, or deleted by any node along the path until the packet reaches the destination node. Note that, if there is also a Routing Header (RH), any visited destination in the route list can process the Option Header.

Hop-by-Hop Option Header is also useful to signal to routers on the path to process the Alternate Marking. However, as said, routers will only examine this option if properly configured.

The optimization of both implementation and scaling of the Alternate Marking Method is also considered and a way to identify flows is

required. The Flow Monitoring Identification field (FlowMonID), as introduced in Section 5.3, goes in this direction and it is used to identify a monitored flow.

The FlowMonID is different from the Flow Label field of the IPv6 Header ([RFC6437]). The Flow Label field in the IPv6 header is used by a source to label sequences of packets to be treated in the network as a single flow and, as reported in [RFC6438], it can be used for load-balancing/equal cost multi-path (LB/ECMP). The reuse of Flow Label field for identifying monitored flows is not considered because it may change the application intent and forwarding behavior. Also, the Flow Label may be changed en route and this may also invalidate the integrity of the measurement. Furthermore, since the Flow Label is pseudo-random, there is always a finite probability of collision. Those reasons make the definition of the FlowMonID necessary for IPv6. Indeed, the FlowMonID is designed and only used to identify the monitored flow. Flow Label and FlowMonID within the same packet are totally disjoint, have different scope, are used to identify flows based on different criteria, and are intended for different use cases.

The rationale for the FlowMonID is further discussed in Section 5.3. This 20 bit field allows easy and flexible identification of the monitored flow and enables improved measurement correlation and finer granularity since it can be used in combination with the traditional TCP/IP 5-tuple to identify a flow. An important point that will be discussed in Section 5.3 is the uniqueness of the FlowMonID and how to allow disambiguation of the FlowMonID in case of collision.

The following section highlights an important requirement for the application of the Alternate Marking to IPv6. The concept of the controlled domain is explained and it is considered an essential precondition, as also highlighted in Section 6.

2.1. Controlled Domain

[RFC8799] introduces the concept of specific limited domain solutions and, in this regard, it is reported the IPv6 Application of the Alternate Marking Method as an example.

IPv6 has much more flexibility than IPv4 and innovative applications have been proposed, but for a number of reasons, such as the policies, options supported, the style of network management and security requirements, it is suggested to limit some of these applications to a controlled domain. This is also the case of the Alternate Marking application to IPv6 as assumed hereinafter.

Therefore, the IPv6 application of the Alternate Marking Method MUST be deployed in a controlled domain. It is RECOMMENDED that an implementation filters packets that carry Alternate Marking data and are entering or leaving the controlled domains.

A controlled domain is a managed network where it is required to select, monitor and control the access to the network by enforcing policies at the domain boundaries in order to discard undesired external packets entering the domain and check the internal packets leaving the domain. It does not necessarily mean that a controlled domain is a single administrative domain or a single organization. A controlled domain can correspond to a single administrative domain or can be composed by multiple administrative domains under a defined network management. Indeed, some scenarios may imply that the Alternate Marking Method involves more than one domain, but in these cases, it is RECOMMENDED that the multiple domains create a whole controlled domain while traversing the external domain by employing IPsec [RFC4301] authentication and encryption or other VPN technology that provides full packet confidentiality and integrity protection. In a few words, it must be possible to control the domain boundaries and eventually use specific precautions if the traffic traverse the Internet.

The security considerations reported in Section 6 also highlight this requirement.

2.1.1. Alternate Marking Measurement Domain

The Alternate Marking measurement domain can overlap with the controlled domain or may be a subset of the controlled domain. The typical scenarios for the application of the Alternate Marking Method depend on the controlled domain boundaries, in particular:

the user equipment can be the starting or ending node, only in case it is fully managed and if it belongs to the controlled domain. In this case the user generated IPv6 packets contain the Alternate Marking data. But, in practice, this is not common due to the fact that the user equipment cannot be totally secured in the majority of cases.

the CPE (Customer Premises Equipment) is most likely to be the starting or ending node since it connects the user's premises with the service provider's network and therefore belongs to the operator's controlled domain. Typically the CPE encapsulates a received packet in an outer IPv6 header which contains the Alternate Marking data. The CPE can also be able to filter and drop packets from outside of the domain with inconsistent fields to make effective the relevant security rules at the domain

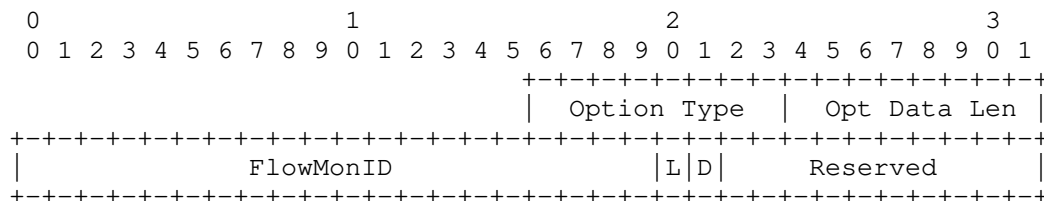
boundaries, for example a simple security check can be to insert the Alternate Marking data if and only if the destination is within the controlled domain.

3. Definition of the AltMark Option

The definition of a new TLV for the Options Extension Headers, carrying the data fields dedicated to the Alternate Marking method, is reported below.

3.1. Data Fields Format

The following figure shows the data fields format for enhanced Alternate Marking TLV (AltMark). This AltMark data can be encapsulated in the IPv6 Options Headers (Hop-by-Hop or Destination Option).



where:

- o Option Type: 8-bit identifier of the type of Option that needs to be allocated. Unrecognized Types MUST be ignored on processing. For Hop-by-Hop Options Header or Destination Options Header, [RFC8200] defines how to encode the three high-order bits of the Option Type field. The two high-order bits specify the action that must be taken if the processing IPv6 node does not recognize the Option Type; for AltMark these two bits MUST be set to 00 (skip over this Option and continue processing the header). The third-highest-order bit specifies whether the Option Data can change en route to the packet's final destination; for AltMark the value of this bit MUST be set to 0 (Option Data does not change en route). In this way, since the three high-order bits of the AltMark Option are set to 000, it means that nodes can simply skip this Option if they do not recognize and that the data of this Option do not change en route, indeed the source is the only one that can write it.
- o Opt Data Len: 4. It is the length of the Option Data Fields of this Option in bytes.

- o FlowMonID: 20-bit unsigned integer. The FlowMon identifier is described in Section 5.3. As further discussed below, it has been picked as 20 bits since it is a reasonable value and a good compromise in relation to the chance of collision. It MUST be set pseudo randomly by the source node or by a centralized controller.
- o L: Loss flag for Packet Loss Measurement as described in Section 5.1;
- o D: Delay flag for Single Packet Delay Measurement as described in Section 5.2;
- o Reserved: is reserved for future use. These bits MUST be set to zero on transmission and ignored on receipt.

4. Use of the AltMark Option

The AltMark Option is the best way to implement the Alternate Marking method and it is carried by the Hop-by-Hop Options header and the Destination Options header. In case of Destination Option, it is processed only by the source and destination nodes: the source node inserts and the destination node processes it. While, in case of Hop-by-Hop Option, it may be examined by any node along the path, if explicitly configured to do so.

It is important to highlight that the Option Layout can be used both as Destination Option and as Hop-by-Hop Option depending on the Use Cases and it is based on the chosen type of performance measurement. In general, it is needed to perform both end to end and hop by hop measurements, and the Alternate Marking methodology allows, by definition, both performance measurements. In many cases the end-to-end measurement is not enough and it is required the hop-by-hop measurement, so the most complete choice can be the Hop-by-Hop Options Header.

IPv6, as specified in [RFC8200], allows nodes to optionally process Hop-by-Hop headers. Specifically 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. Also, it is 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.

Another scenario that can be mentioned is the presence of a Routing Header, in particular it is possible to consider SRv6. A new type of Routing Header, referred as Segment Routing Header (SRH), has been defined in [RFC8754] for SRv6. Like any other use case of IPv6, Hop-

by-Hop and Destination Options are usable when SRv6 header is present. Because SRv6 is implemented through a Segment Routing Header (SRH), Destination Options before the Routing Header are processed by each destination in the route list, that means, in case of SRH, by every SR node that is identified by the SR path. More details about the SRv6 application are described in [I-D.fz-spring-srv6-alt-mark].

In summary, it is possible to list the alternative possibilities:

- o Destination Option not preceding a Routing Header => measurement only by node in Destination Address.
- o Hop-by-Hop Option => every router on the path with feature enabled.
- o Destination Option preceding a Routing Header => every destination node in the route list.

In general, Hop-by-Hop and Destination Options are the most suitable ways to implement Alternate Marking.

It is worth mentioning that new Hop-by-Hop Options are not strongly recommended in [RFC7045] and [RFC8200], unless there is a clear justification to standardize it, because nodes may be configured to ignore the Options Header, drop or assign packets containing an Options Header to a slow processing path. In case of the AltMark data fields described in this document, the motivation to standardize a new Hop-by-Hop Option is that it is needed for OAM (Operations, Administration, and Maintenance). An intermediate node can read it or not, but this does not affect the packet behavior. The source node is the only one that writes the Hop-by-Hop Option to mark alternately the flow, so, the performance measurement can be done for those nodes configured to read this Option, while the others are simply not considered for the metrics.

The Hop-by-Hop Option defined in this document is designed to take advantage of the property of how Hop-by-Hop options are processed. Nodes that do not support this Option SHOULD ignore them. This can mean that, in this case, the performance measurement does not account for all links and nodes along a path. The definition of the Hop-by-Hop Options in this document is also designed to minimize throughput impact both on nodes that do not recognize the Option and on node that support it. Indeed, the three high-order bits of the Options Header defined in this draft are 000 and, in theory, as per [RFC8200] and [I-D.hinden-6man-hbh-processing], this means "skip if do not recognize and data do not change en route". [RFC8200] also mentions that the nodes only examine and process the Hop-by-Hop Options header

if explicitly configured to do so. For these reasons, this Hop-by-Hop Option should not affect the throughput. However, in practice, it is important to be aware that the things may be different in the implementation and it can happen that packets with Hop-by-Hop are forced onto the slow path, but this is a general issue, as also explained in [I-D.hinden-6man-hbh-processing]. It is also worth mentioning that the application to a controlled domain should avoid the risk of arbitrary nodes dropping packets with Hop-by-Hop Options.

5. Alternate Marking Method Operation

This section describes how the method operates.

[I-D.ietf-ippm-rfc8321bis] introduces several applicable methods which are reported below, and a new field is introduced to facilitate the deployment and improve the scalability.

5.1. Packet Loss Measurement

The measurement of the packet loss is really straightforward in comparison to the existing mechanisms, as detailed in [I-D.ietf-ippm-rfc8321bis]. The packets of the flow are grouped into batches, and all the packets within a batch are marked by setting the L bit (Loss flag) to a same value. The source node can switch the value of the L bit between 0 and 1 after a fixed number of packets or according to a fixed timer, and this depends on the implementation. The source node is the only one that marks the packets to create the batches, while the intermediate nodes only read the marking values and identify the packet batches. By counting the number of packets in each batch and comparing the values measured by different network nodes along the path, it is possible to measure the packet loss occurred in any single batch between any two nodes. Each batch represents a measurable entity recognizable by all network nodes along the path.

Both fixed number of packets and fixed timer can be used by the source node to create packet batches. But, as also explained in [I-D.ietf-ippm-rfc8321bis], the timer-based batches are preferable because they are more deterministic than the counter-based batches. There is no definitive rule for counter-based batches, differently from timer-based batches. Using a fixed timer for the switching offers better control over the method, indeed the length of the batches can be chosen large enough to simplify the collection and the comparison of the measures taken by different network nodes. In the implementation the counters can be sent out by each node to the controller that is responsible for the calculation. It is also possible to exchange this information by using other on-path techniques. But this is out of scope for this document.

Packets with different L values may get swapped at batch boundaries, and in this case, it is required that each marked packet can be assigned to the right batch by each router. It is important to mention that for the application of this method there are two elements to consider: the clock error between network nodes and the network delay. These can create offsets between the batches and out-of-order of the packets. The mathematical formula on timing aspects, explained in section 5 of [I-D.ietf-ippm-rfc8321bis], must be satisfied and it takes into considerations the different causes of reordering such as clock error and network delay. The assumption is to define the available counting interval where to get stable counters and to avoid these issues. Specifically, if the effects of network delay are ignored, the condition to implement the methodology is that the clocks in different nodes MUST be synchronized to the same clock reference with an accuracy of $\pm B/2$ time units, where B is the fixed time duration of the batch, which refers to the original marking interval at the source node considering that this interval could fluctuate along the path. In this way each marked packet can be assigned to the right batch by each node. Usually the counters can be taken in the middle of the batch period to be sure to take still counters. In a few words this implies that the length of the batches MUST be chosen large enough so that the method is not affected by those factors. The length of the batches can be determined based on the specific deployment scenario.

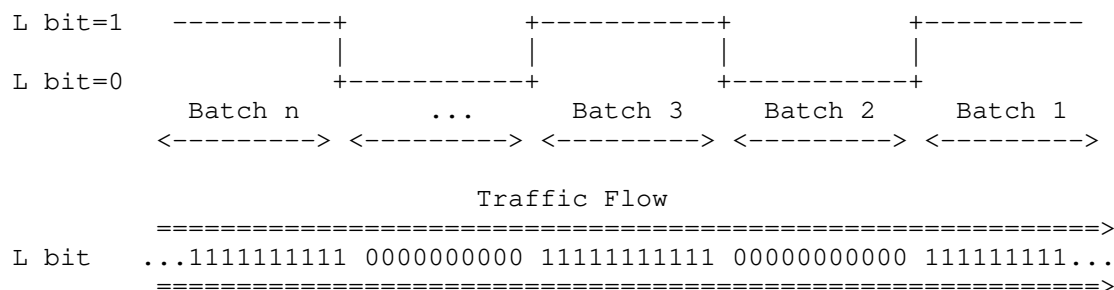


Figure 1: Packet Loss Measurement and Single-Marking Methodology using L bit

It is worth mentioning that the duration of the batches is considered stable over time in the previous figure. In theory, it is possible to change the length of batches over time and among different flows for more flexibility. But, in practice, it could complicate the correlation of the information.

5.2. Packet Delay Measurement

The same principle used to measure packet loss can be applied also to one-way delay measurement. Delay metrics MAY be calculated using the two possibilities:

1. **Single-Marking Methodology:** This approach uses only the L bit to calculate both packet loss and delay. In this case, the D flag MUST be set to zero on transmit and ignored by the monitoring points. The alternation of the values of the L bit can be used as a time reference to calculate the delay. Whenever the L bit changes and a new batch starts, a network node can store the timestamp of the first packet of the new batch, that timestamp can be compared with the timestamp of the first packet of the same batch on a second node to compute packet delay. But this measurement is accurate only if no packet loss occurs and if there is no packet reordering at the edges of the batches. A different approach can also be considered and it is based on the concept of the mean delay. The mean delay for each batch is calculated by considering the average arrival time of the packets for the relative batch. There are limitations also in this case indeed, each node needs to collect all the timestamps and calculate the average timestamp for each batch. In addition, the information is limited to a mean value.
2. **Double-Marking Methodology:** This approach is more complete and uses the L bit only to calculate packet loss and the D bit (Delay flag) is fully dedicated to delay measurements. The idea is to use the first marking with the L bit to create the alternate flow and, within the batches identified by the L bit, a second marking is used to select the packets for measuring delay. The D bit creates a new set of marked packets that are fully identified over the network, so that a network node can store the timestamps of these packets; these timestamps can be compared with the timestamps of the same packets on a second node to compute packet delay values for each packet. The most efficient and robust mode is to select a single double-marked packet for each batch, in this way there is no time gap to consider between the double-marked packets to avoid their reorder. Regarding the rule for the selection of the packet to be double-marked, the same considerations in Section 5.1 apply also here and the double-marked packet can be chosen within the available counting interval that is not affected by factors such as clock errors. If a double-marked packet is lost, the delay measurement for the considered batch is simply discarded, but this is not a big problem because it is easy to recognize the problematic batch and skip the measurement just for that one. So in order to have more

information about the delay and to overcome out-of-order issues this method is preferred.

In summary the approach with double marking is better than the approach with single marking. Moreover, the two approaches provide slightly different pieces of information and the data consumer can combine them to have a more robust data set.

Similar to what said in Section 5.1 for the packet counters, in the implementation the timestamps can be sent out to the controller that is responsible for the calculation or could also be exchanged using other on-path techniques. But this is out of scope for this document.

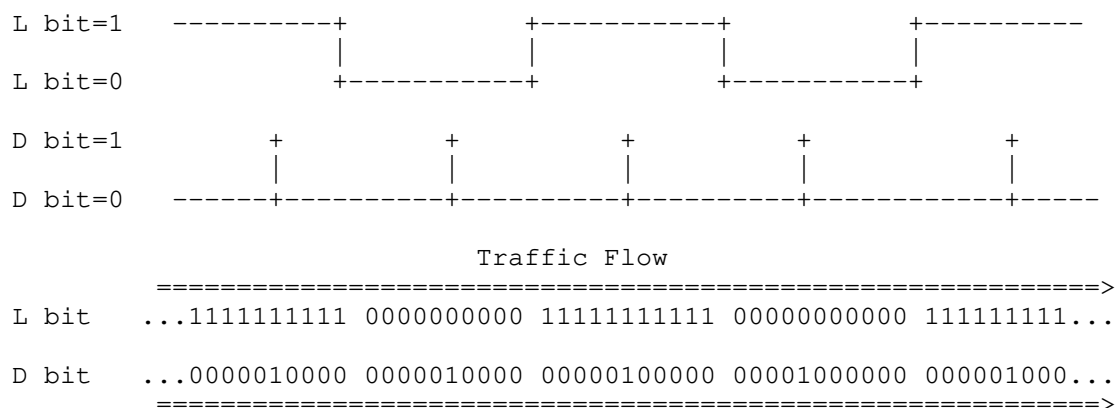


Figure 2: Double-Marking Methodology using L bit and D bit

Likewise to packet delay measurement (both for Single Marking and Double Marking), the method can also be used to measure the inter-arrival jitter.

5.3. Flow Monitoring Identification

The Flow Monitoring Identification (FlowMonID) identifies the flow to be measured and is required for some general reasons:

First, it helps to reduce the per node configuration. Otherwise, each node needs to configure an access-control list (ACL) for each of the monitored flows. Moreover, using a flow identifier allows a flexible granularity for the flow definition, indeed, it can be used together with other identifiers (e.g. 5-tuple).

Second, it simplifies the counters handling. Hardware processing of flow tuples (and ACL matching) is challenging and often incurs into performance issues, especially in tunnel interfaces.

Third, it eases the data export encapsulation and correlation for the collectors.

The FlowMonID MUST only be used as a monitored flow identifier in order to determine a monitored flow within the measurement domain. This entails not only an easy identification but improved correlation as well.

The value of 20 bits has been selected for the FlowMonID since it is a good compromise and implies a low rate of ambiguous FlowMonIDs that can be considered acceptable in most of the applications. The disambiguation issue can be solved by tagging the pseudo randomly generated FlowMonID with additional flow information. In particular, it is RECOMMENDED to consider the 3-tuple FlowMonID, source and destination addresses:

- o If the 20 bit FlowMonID is set independently and pseudo randomly in a distributed way there is a chance of collision. Indeed, by using the well-known birthday problem in probability theory, if the 20 bit FlowMonID is set independently and pseudo randomly without any additional input entropy, there is a 50% chance of collision for 1206 flows. So, for more entropy, FlowMonID is combined with source and destination addresses. Since there is a 1% chance of collision for 145 flows, it is possible to monitor 145 concurrent flows per host pairs with a 1% chance of collision.
- o If the 20 bits FlowMonID is set pseudo randomly but in a centralized way, the controller can instruct the nodes properly in order to guarantee the uniqueness of the FlowMonID. With 20 bits, the number of combinations is 1048576, and the controller should ensure that all the FlowMonID values are used without any collision. Therefore, by considering source and destination addresses together with the FlowMonID, it can be possible to monitor 1048576 concurrent flows per host pairs.

A consistent approach MUST be used in the Alternate Marking deployment to avoid the mixture of different ways of identifying. All the nodes along the path and involved into the measurement SHOULD use the same mode for identification. As mentioned, it is RECOMMENDED to use the FlowMonID for identification purpose in combination with source and destination addresses to identify a flow. By considering source and destination addresses together with the FlowMonID it can be possible to monitor 145 concurrent flows per host pairs with a 1% chance of collision in case of pseudo randomly

generated FlowMonID, or 1048576 concurrent flows per host pairs in case of centralized controller. It is worth mentioning that the solution with the centralized control allows finer granularity and therefore adds even more flexibility to the flow identification.

The FlowMonID field is set at the source node, which is the ingress point of the measurement domain, and can be set in two ways:

- a. It can be algorithmically generated by the source node, that can set it pseudo-randomly with some chance of collision. This approach cannot guarantee the uniqueness of FlowMonID since conflicts and collisions are possible. But, considering the recommendation to use FlowMonID with source and destination addresses the conflict probability is reduced due to the FlowMonID space available for each endpoint pair (i.e. 145 flows with 1% chance of collision).
- b. It can be assigned by the central controller. Since the controller knows the network topology, it can allocate the value properly to avoid or minimize ambiguity and guarantee the uniqueness. In this regard, the controller can verify that there is no ambiguity between different pseudo-randomly generated FlowMonIDs on the same path. The conflict probability is really small given that the FlowMonID is coupled with source and destination addresses and up to 1048576 flows can be monitored for each endpoint pair. When all values in the FlowMonID space are consumed, the centralized controller can keep track and reassign the values that are not used any more by old flows.

If the FlowMonID is set by the source node, the intermediate nodes can read the FlowMonIDs from the packets in flight and act accordingly. While, if the FlowMonID is set by the controller, both possibilities are feasible for the intermediate nodes which can learn by reading the packets or can be instructed by the controller.

5.4. Multipoint and Clustered Alternate Marking

The Alternate Marking method can also be extended to any kind of multipoint to multipoint paths, and the network clustering approach allows a flexible and optimized performance measurement, as described in [I-D.ietf-ippm-rfc8889bis].

The Cluster is the smallest identifiable subnetwork of the entire Network graph that still satisfies the condition that the number of packets that goes in is the same that goes out. With network clustering, it is possible to use the partition of the network into clusters at different levels in order to perform the needed degree of detail. So, for Multipoint Alternate Marking, FlowMonID can identify

in general a multipoint-to-multipoint flow and not only a point-to-point flow.

5.5. Data Collection and Calculation

The nodes enabled to perform performance monitoring collect the value of the packet counters and timestamps. There are several alternatives to implement Data Collection and Calculation, but this is not specified in this document.

There are documents on the control plane mechanisms of Alternate Marking, e.g. [I-D.ietf-idr-sr-policy-ifit], [I-D.chen-pce-pcep-ifit].

6. Security Considerations

This document aims to apply a method to perform measurements that does not directly affect Internet security nor applications that run on the Internet. However, implementation of this method must be mindful of security and privacy concerns.

There are two types of security concerns: potential harm caused by the measurements and potential harm to the measurements.

Harm caused by the measurement: Alternate Marking implies modifications on the fly to an Option Header of IPv6 packets by the source node, but this must be performed in a way that does not alter the quality of service experienced by the packets and that preserves stability and performance of routers doing the measurements. As already discussed in Section 4, it is RECOMMENDED that the AltMark Option does not affect the throughput and therefore the user experience.

Harm to the measurement: Alternate Marking measurements could be harmed by routers altering the fields of the AltMark Option (e.g. marking of the packets, FlowMonID) or by a malicious attacker adding AltMark Option to the packets in order to consume the resources of network devices and entities involved. As described above, the source node is the only one that writes the Option Header while the intermediate nodes and destination node only read it without modifying the Option Header. But, for example, an on-path attacker can modify the flags, whether intentionally or accidentally, or deliberately insert a new option to the packet flow or delete the option from the packet flow. The consequent effect could be to give the appearance of loss or delay or invalidate the measurement by modifying option identifiers, such as FlowMonID. The malicious implication can be to cause actions from the network administrator where an intervention is not necessary or to hide real issues in the

network. Since the measurement itself may be affected by network nodes intentionally altering the bits of the AltMark Option or injecting Options headers as a means for Denial of Service (DoS), the Alternate Marking MUST be applied in the context of a controlled domain, where the network nodes are locally administered and this type of attack can be avoided. For this reason, the implementation of the method is not done on the end node if it is not fully managed and does not belong to the controlled domain. Packets generated outside the controlled domain may consume router resources by maliciously using the HbH Option, but this can be mitigated by filtering these packets at the controlled domain boundary. This can be done because, if the end node does not belong to the controlled domain, it is not supposed to add the AltMark HbH Option, and it can be easily recognized.

An attacker that does not belong to the controlled domain can maliciously send packets with AltMark Option. But if Alternate Marking is not supported in the controlled domain, no problem happens because the AltMark Option is treated as any other unrecognized option and will not be considered by the nodes since they are not configured to deal with it, so the only effect is the increased MTU (by 48 bits). While if Alternate Marking is supported in the controlled domain, it is also necessary to avoid that the measurements are affected and external packets with AltMark Option MUST be filtered. As any other Hop-by-Hop Options or Destination Options, it is possible to filter AltMark Options entering or leaving the domain e.g. by using ACL extensions for filtering.

The flow identifier (FlowMonID) composes the AltMark Option together with the two marking bits (L and D). As explained in Section 5.3, there is a chance of collision if the FlowMonID is set pseudo randomly and a solution exists. In general this may not be a problem and a low rate of ambiguous FlowMonIDs can be acceptable, since this does not cause significant harm to the operators or their clients and this harm may not justify the complications of avoiding it. But, for large scale measurements, a big number of flows could be monitored and the probability of a collision is higher, thus the disambiguation of the FlowMonID field can be considered.

The privacy concerns also need to be analyzed even if the method only relies on information contained in the Option Header without any release of user data. Indeed, from a confidentiality perspective, although AltMark Option does not contain user data, the metadata can be used for network reconnaissance to compromise the privacy of users by allowing attackers to collect information about network performance and network paths. AltMark Option contains two kinds of metadata: the marking bits (L and D bits) and the flow identifier (FlowMonID).

The marking bits are the small information that is exchanged between the network nodes. Therefore, due to this intrinsic characteristic, network reconnaissance through passive eavesdropping on data-plane traffic is difficult. Indeed, an attacker cannot gain information about network performance from a single monitoring point. The only way for an attacker can be to eavesdrop on multiple monitoring points at the same time, because they have to do the same kind of calculation and aggregation as Alternate Marking requires.

The FlowMonID field is used in the AltMark Option as the identifier of the monitored flow. It represents a more sensitive information for network reconnaissance and may allow a flow tracking type of attack because an attacker could collect information about network paths.

Furthermore, in a pervasive surveillance attack, the information that can be derived over time is more. But, as further described hereinafter, the application of the Alternate Marking to a controlled domain helps to mitigate all the above aspects of privacy concerns.

At the management plane, attacks can be set up by misconfiguring or by maliciously configuring AltMark Option. Thus, AltMark Option configuration MUST be secured in a way that authenticates authorized users and verifies the integrity of configuration procedures. Solutions to ensure the integrity of AltMark Option are outside the scope of this document. Also, attacks on the reporting of the statistics between the monitoring points and the network management system (e.g. centralized controller) can interfere with the proper functioning of the system. Hence, the channels used to report back flow statistics MUST be secured.

As stated above, the precondition for the application of the Alternate Marking is that it MUST be applied in specific controlled domains, thus confining the potential attack vectors within the network domain. [RFC8799] analyzes and discusses the trend towards network behaviors that can be applied only within a limited domain. This is due to the specific set of requirements especially related to security, network management, policies and options supported which may vary between such limited domains. A limited administrative domain provides the network administrator with the means to select, monitor and control the access to the network, making it a trusted domain. In this regard it is expected to enforce policies at the domain boundaries to filter both external packets with AltMark Option entering the domain and internal packets with AltMark Option leaving the domain. Therefore, the trusted domain is unlikely subject to hijacking of packets since packets with AltMark Option are processed and used only within the controlled domain.

As stated, the application to a controlled domain ensures the control over the packets entering and leaving the domain, but despite that, leakages may happen for different reasons, such as a failure or a fault. In this case, nodes outside the domain MUST simply ignore packets with AltMark Option since they are not configured to handle it and should not process it.

Additionally, it is to be noted that the AltMark Option is carried by the Options Header and it may have some impact on the packet sizes for the monitored flow and on the path MTU, since some packets might exceed the MTU. However, the relative small size (48 bit in total) of these Option Headers and its application to a controlled domain help to mitigate the problem.

It is worth mentioning that the security concerns may change based on the specific deployment scenario and related threat analysis, which can lead to specific security solutions that are beyond the scope of this document. As an example, the AltMark Option can be used as Hop-by-Hop or Destination Option and, in case of Destination Option, multiple administrative domains may be traversed by the AltMark Option that is not confined to a single administrative domain. In this case, the user, aware of the kind of risks, may still want to use Alternate Marking for telemetry and test purposes but the controlled domain must be composed by more than one administrative domains. To this end, the inter-domain links need to be secured (e.g., by IPsec, VPNs) in order to avoid external threats and realize the whole controlled domain.

It might be theoretically possible to modulate the marking or the other fields of the AltMark Option to serve as a covert channel to be used by an on-path observer. This may affect both the data and management plane, but, here too, the application to a controlled domain helps to reduce the effects.

The Alternate Marking application described in this document relies on a time synchronization protocol. Thus, by attacking the time protocol, an attacker can potentially compromise the integrity of the measurement. A detailed discussion about the threats against time protocols and how to mitigate them is presented in [RFC7384]. Network Time Security (NTS), described in [RFC8915], is a mechanism that can be employed. Also, the time, which is distributed to the network nodes through the time protocol, is centrally taken from an external accurate time source, such as an atomic clock or a GPS clock. By attacking the time source it can be possible to compromise the integrity of the measurement as well. There are security measures that can be taken to mitigate the GPS spoofing attacks and a network administrator should certainly employ solutions to secure the network domain.

7. IANA Considerations

The Option Type should be assigned in IANA's "Destination Options and Hop-by-Hop Options" registry.

This draft requests the following IPv6 Option Type assignment from the Destination Options and Hop-by-Hop Options sub-registry of Internet Protocol Version 6 (IPv6) Parameters (<https://www.iana.org/assignments/ipv6-parameters/>).

Hex Value	Binary Value act chg rest			Description	Reference
TBD	00	0	tbd	AltMark	[This draft]

8. Acknowledgements

The authors would like to thank Bob Hinden, Ole Troan, Martin Duke, Lars Eggert, Roman Danyliw, Alvaro Retana, Eric Vyncke, Warren Kumari, Benjamin Kaduk, Stewart Bryant, Christopher Wood, Yoshifumi Nishida, Tom Herbert, Stefano Previdi, Brian Carpenter, Greg Mirsky, Ron Bonica for the precious comments and suggestions.

9. References

9.1. Normative References

- [I-D.ietf-ippm-rfc8321bis]
Fioccola, G., Cociglio, M., Mirsky, G., Mizrahi, T., and T. Zhou, "Alternate-Marking Method", draft-ietf-ippm-rfc8321bis-01 (work in progress), April 2022.
- [I-D.ietf-ippm-rfc8889bis]
Fioccola, G., Cociglio, M., Sapio, A., Sisto, R., and T. Zhou, "Multipoint Alternate-Marking Clustered Method", draft-ietf-ippm-rfc8889bis-01 (work in progress), April 2022.
- [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>>.

9.2. Informative References

- [I-D.chen-pce-pcep-ifit]
Yuan, H., Zhou, T., Li, W., Fioccola, G., and Y. Wang, "Path Computation Element Communication Protocol (PCEP) Extensions to Enable IFIT", draft-chen-pce-pcep-ifit-06 (work in progress), February 2022.
- [I-D.fz-spring-srv6-alt-mark]
Fioccola, G., Zhou, T., and M. Cociglio, "Segment Routing Header encapsulation for Alternate Marking Method", draft-fz-spring-srv6-alt-mark-02 (work in progress), February 2022.
- [I-D.hinden-6man-hbh-processing]
Hinden, R. M. and G. Fairhurst, "IPv6 Hop-by-Hop Options Processing Procedures", draft-hinden-6man-hbh-processing-01 (work in progress), June 2021.
- [I-D.ietf-idr-sr-policy-ifit]
Qin, F., Yuan, H., Zhou, T., Fioccola, G., and Y. Wang, "BGP SR Policy Extensions to Enable IFIT", draft-ietf-idr-sr-policy-ifit-03 (work in progress), January 2022.
- [I-D.peng-v6ops-hbh]
Peng, S., Li, Z., Xie, C., Qin, Z., and G. Mishra, "Processing of the Hop-by-Hop Options Header", draft-peng-v6ops-hbh-06 (work in progress), August 2021.
- [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>>.
- [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>>.
- [RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <<https://www.rfc-editor.org/info/rfc6437>>.

- [RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <<https://www.rfc-editor.org/info/rfc6438>>.
- [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>>.
- [RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <<https://www.rfc-editor.org/info/rfc7384>>.
- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.
- [RFC8799] Carpenter, B. and B. Liu, "Limited Domains and Internet Protocols", RFC 8799, DOI 10.17487/RFC8799, July 2020, <<https://www.rfc-editor.org/info/rfc8799>>.
- [RFC8915] Franke, D., Sibold, D., Teichel, K., Dansarie, M., and R. Sundblad, "Network Time Security for the Network Time Protocol", RFC 8915, DOI 10.17487/RFC8915, September 2020, <<https://www.rfc-editor.org/info/rfc8915>>.

Authors' Addresses

Giuseppe Fioccola
Huawei
Riesstrasse, 25
Munich 80992
Germany

Email: giuseppe.fioccola@huawei.com

Tianran Zhou
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: zhoutianran@huawei.com

Mauro Cociglio
Telecom Italia
Via Reiss Romoli, 274
Torino 10148
Italy

Email: mauro.cociglio@telecomitalia.it

Fengwei Qin
China Mobile
32 Xuanwumenxi Ave.
Beijing 100032
China

Email: qinfengwei@chinamobile.com

Ran Pang
China Unicom
9 Shouti South Rd.
Beijing 100089
China

Email: pangran@chinaunicom.cn

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.

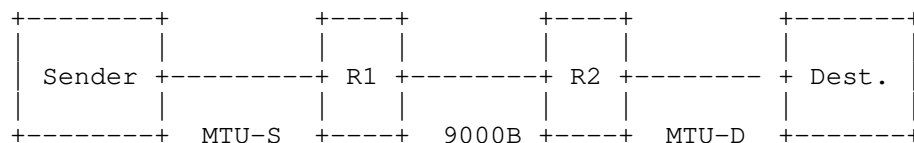


Figure 1

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://gerrit.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

Network Working Group
Internet-Draft
Intended status: Informational
Expires: 20 October 2022

S. Peng
Z. Li
Huawei Technologies
C. Xie
China Telecom
Z. Qin
China Unicom
G. Mishra
Verizon Inc.
18 April 2022

Operational Issues with Processing of the Hop-by-Hop Options Header
draft-ietf-v6ops-hbh-01

Abstract

This document describes the processing of the Hop-by-Hop Options Header (HBH) in today's routers in the aspects of standards specification, common implementations, and default operations. This document outlines the reasons why the Hop-by-Hop Options Header is rarely utilized in current networks. In addition, this document describes how the HBH could be used as a powerful mechanism allowing deployment and operations of new services requiring a more optimized way to leverage network resources of an infrastructure. The Hop-by-Hop Options Header is taken into consideration by several network operators as a valuable container for carrying the information facilitating the introduction of new services. The purpose of this draft is to document the reasons why the HBH is rarely used within networks and to define a proper list of requirements aiming to allow a better leverage of the HBH capability.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] [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 20 October 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	3
2. Terminology	4
3. Modern Router Architecture	4
4. Specification of RFC 8200	7
5. Common Implementations	8
5.1. Historical Reasons	9
5.2. Consequences	9
6. Typical Processing	9
7. New Services	10
8. Requirements	10
9. Migration Strategies	11
10. Security Considerations	11
11. IANA Considerations	11
12. Acknowledgements	11
13. References	11
13.1. Normative References	11
13.2. Informative References	12
Authors' Addresses	13

1. Introduction

Due to historical reasons, such as incapable ASICs, limited IPv6 deployments, and few service requirements, the most common Hop-by-Hop Options header (HBH) processing implementation is that the node sends the IPv6 packets with the Hop-by-Hop Options header to the control plane of the node. The option type of each option carried within the Hop-by-Hop Options header will not even be examined before the packet is sent to the control plane. Very often, such processing behavior is the default configuration or, even worse, is the only behavior of the ipv6 implementation of the node.

Such default processing behavior of the Hop-by-Hop Options header could result in various unpleasant effects such as a risk of Denial of Service (DoS) attack on the router control plane and inconsistent packet drops due to rate limiting on the interface between the router control plane and forwarding plane, which will impact the normal end-to-end IP forwarding of the network services.

This actually introduced a circular problem:

- > An implementation problem caused HBH to become a DoS vector.
- > Because HBH is a DoS vector, network operators deployed ACLs that discard packets containing HBH.
- > Because network operators deployed ACLs that discard packets containing HBH, network designers stopped defining new HBH Options.
- > Because network designers stopped defining new HBH Options, the community was not motivated to fix the implementation problem that cause HBH to become a DoS vector.

Driven by the wide deployments of IPv6 and ever-emerging new services, the Hop-by-Hop Options Header is taken as a valuable container for carrying the information to facilitate these new services.

The purpose of this work is to

- * Break the endless cycle that resulted in HBH to become a DOS vector.
- * Enable the HBH options header to be utilized in a safe and secure way without impacting the management plane.

- * Ease the deployments of the new HBH based network services in a multi-vendor scenario that can now be deployed without operational impact.

In this draft, the reasons why the HBH is rarely used within networks will be documented and a proper list of requirements aiming to allow a better leverage of the HBH capability will be defined.

2. Terminology

The Forwarding Plane and Control Plane used in this draft can refer to the same terminologies as defined in [I-D.ietf-6man-hbh-processing], respectively.

3. Modern Router Architecture

Modern router architecture design maintains a strict separation of the router control plane and its forwarding plane [RFC6192], as shown in Figure 1. Either the control plane or the forwarding plane is composed of both software and hardware, but each plane is responsible for different functions. In this draft, we focus on only the routers following the architecture as shown in Figure 1 and those being deployed in the network rather than those at home.

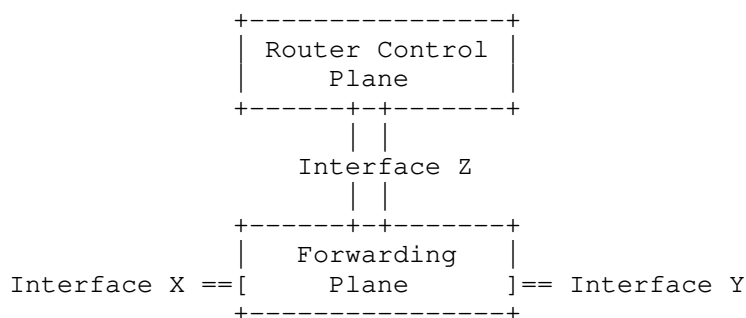


Figure 1. Modern Router Architecture

The router control plane supports routing and management functions, handling packets destined to the device as well as building and sending packets originated locally on the device, and also drives the programming of the forwarding plane. The router control plane is generally realized in software on general-purpose processors, and its hardware is usually not optimized for high-speed packet handling. Because of the wide range of functionality, it is more susceptible to security vulnerabilities and a more likely a target for a DoS attack.

The forwarding plane is typically responsible for receiving a packet on an incoming interface, performing a lookup to identify the packet's next hop and determine the outgoing interface towards the destination, and forwarding the packet out through the appropriate outgoing interface. Typically, forwarding plane functionality is realized in high-performance Application Specific Integrated Circuits (ASICs) or Network Processors (NPs) that are capable of handling very high packet rates.

The router control plane interfaces with its forwarding plane through the Interface Z, as shown in the Figure 1, and the forwarding plane connects to other network devices via Interfaces such as X and Y. Since the router control plane is vulnerable to the DoS attack, usually a traffic filtering mechanism is implemented on Interface Z in order to block unwanted traffic. In order to protect the router control plane, a rate-limiting mechanism is always implemented on this interface. However, such rate limiting mechanism will always cause inconsistent packet drops, which will impact the normal IP forwarding.

Semiconductor chip technology has advanced significantly in the last decade, and as such the widely used network processing and forwarding process can now not only forward packets at line speed, but also easily support other feature processing such as QoS for DiffServ/MPLS, Access Control List (ACL), Firewall, and Deep Packet Inspection (DPI).

A Network Processing Unit (NPU) is a non-ASIC based Integrated Circuit (IC) that is programmable through software. It performs all packet header operations between the physical layer interface and the switching fabric such as packet parsing and forwarding, modification, and forwarding. Many equipment vendors implement these functions in fixed function ASICs rather than using "off-the-shelf" NPUs, because of proprietary algorithms.

Classification Co-processor is a specialized processor that can be used to lighten the processing load on an NPU by handling the parsing and classification of incoming packets such as IPv6 extended header HBH options processing. This advancement enables network processors to do the general process to handle simple control messages for traffic management, such as signaling for hardware programming, congestion state report, OAM, etc. Industry trend is for intelligent multi-core CPU hardware using modern NPUs for forwarding packets at line rate while still being able to perform other complex tasks such as HBH forwarding options processing without having to punt to the control plane.

Many of the packet-processing devices employed in modern switch and router designs are fixed-function ASICs to handle proprietary functions. While these devices can be very efficient for the set of functions they are designed for, they can be very inflexible. There is a tradeoff of price, performance and flexibility when vendors make a choice to use a fixed function ASIC as opposed to NPU. Due to the inflexibility of the fixed function ASIC, tasks that require additional processing such as IPv6 HBH header processing must be punted to the control plane. This problem is still a challenge today and is the reason why operators to protect against control plane DOS attack vector must drop or ignore HBH options. As industry shifts to Merchant Silicon based NPU evolution from fixed function ASIC, the gap will continue to close increasing the viability ubiquitous HBH use cases due to now processing in the forwarding plane.

Most modern routers maintain a strict separation between forwarding plane and control plane hardware. Forwarding plane bandwidth and resources are plentiful, while control plane bandwidth and resources are constrained. In order to protect scarce control plane resources, routers enforce policies that restrict access from the forwarding plane to the control plane. Effective policies address packets containing the HBH Options Extension header, because HBH control options require access from the forwarding plane to the control plane. Many network operators perceive HBH Options to be a breach of the separation between the forwarding and control planes. In this case HBH control options would be required to be punted to control plane by fixed function ASICs as well as NPUs.

The maximum length of an HBH Options header is 2,048 bytes. A source node can encode hundreds of options in 2,048 bytes [I-D.herbert-6man-eh-limits]. With today's technology it would be cost prohibitive to be able to process hundreds of options with either NPU or proprietary fixed function ASIC.

As per [RFC8200], 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. This can be beneficial in cases where transit nodes are legacy hardware and the destination endpoint PE is newer NPU based hardware that can process HBH in the forwarding plane.

IPv6 Extended Header limitations that need to be addressed to make HBH processing more efficient and viable in the forwarding plane:

[RFC8504] defines the IPv6 node requirements and how to protect a node from excessive header chain and excessive header options with various limitations that can be defined on a node. [RFC8883] defines ICMPv6 Errors for discarding packets due to processing limits. Per

[RFC8200] HBH options must be processed serially. However, an implementation of options processing can be made to be done with more parallelism in serial processing grouping of similar options to be processed in parallel.

The IPv6 standard does not currently limit the header chain length or number of options that can be encoded.

Each Option is encoded in a TLV and so processing of a long list of TLVs is expensive. Zero data length encoded options TLVs are a valid option. A DOS vector could be easily generated by encoding 1000 HBH options (Zero data length) in a standard 1500 MTU packet. So now imagine if you have a Christmas tree long header chain to parse each with many options.

4. Specification of RFC 8200

[RFC8200] defines several IPv6 extension header types, including the Hop-by-Hop (HBH) Options header. As specified in [RFC8200], the Hop-by-Hop (HBH) Options header is used to carry optional information that will be examined and processed by every node along a packet's delivery path, and it is identified by a Next Header value of zero in the IPv6 header.

The Hop-by-Hop (HBH) Options header contains the following fields:

-- Next Header: 8-bit selector, identifies the type of header immediately following the Hop-by-Hop Options header.

-- Hdr Ext Len: 8-bit unsigned integer, the length of the Hop-by-Hop Options header in 8-octet units, not including the first 8 octets.

-- Options: Variable-length field, of length such that the complete Hop-by-Hop Options header is an integer multiple of 8 octets long.

The Hop-by-Hop (HBH) Options header carries a variable number of "options" that are encoded in the format of type-length-value (TLV).

The highest-order two bits (i.e., the ACT bits) of the Option Type specify the action that must be taken if the processing IPv6 node does not recognize the Option Type. The third-highest-order bit (i.e., the CHG bit) of the Option Type specifies whether or not the Option Data of that option can change en route to the packet's final destination.

As per [RFC8200], 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. It means that the HBH processing behavior in a node depends on its configuration.

However, in the current [RFC8200], there is no explicit specification of the possible configurations. Therefore, the nodes may be configured to ignore the Hop-by-Hop Options header, drop packets containing a Hop-by-Hop Options header, or assign packets containing a Hop-by-Hop Options header to the control plane [RFC8200]. Because of these likely uncertain processing behaviors, new hop-by-hop options are not recommended.

5. Common Implementations

In the current common implementations, once an IPv6 packet, with its Next Header field set to 0, arrives at a node, it will be directly sent to the control plane of the node. With such implementations, the value of the Next Header field in the IPv6 header is the only trigger for the default processing behavior. The option type of each option carried within the Hop-by-Hop Options header will not even be examined before the packet is sent to the control plane.

Very often, such processing behavior is the default configuration on the node, which is embedded in the implementation and cannot be changed or reconfigured.

Another critical component of IPv6 HBH processing, in some cases overlooked, is the operator core network which can be designed to use the global Internet routing table for internet traffic and in other cases use an overlay MPLS VPN to carry Internet traffic.

In the global Internet routing table scenario where only an underlay global routing table exists, and no VPN overlay carrying customer Internet traffic, the IPv6 HBH options can be used as a DOS attack vector for both the operator nodes, adjacent inter-as peer nodes as well as customer nodes along a path.

In a case where the Internet routing table is carried in a MPLS VPN overlay payload, the HBH options header does not impact the operator underlay framework and only impacts the VPN overlay payload and thus the operator underlay topmost label global table routing FEC LSP instantiation is not impacted as the operator underlay is within the operators closed domain.

However, HBH options DOS attack vector in the VPN overlay can still impact the customer CE destination end nodes as well as other adjacent inter-as operators that only use underlay global Internet

routing table. In an operator closed domain where MPLS VPN overlay is utilized to carry internet traffic, the operator has full control of the underlay and IPv6 Extended header chain length as well as the number of HBH options encoded.

In the global routing table scenario for Internet traffic there is no way to control the IPv6 Extended header chain length as well as the number of HBH options encoded.

5.1. Historical Reasons

When IPv6 was first implemented on high-speed routers, HBH options were not yet well-understood and ASICs were not as capable as they are today. So, early IPv6 implementations dispatched all packets that contain HBH options to their control plane.

5.2. Consequences

Such implementation introduces a risk of a DoS attack on the control plane of the node, and a large flow of IPv6 packets could congest the control plane, causing other critical functions (including routing and network management) that are executed on the control plane to fail. Rate limiting mechanisms will cause inconsistent packet drops and impact the normal end-to-end IP forwarding of the network services.

6. Typical Processing

To mitigate this DoS vulnerability, many operators deployed Access Control Lists (ACLs) that discard all packets containing HBH Options.

[RFC6564] shows the Reports from the field indicating that some IP routers deployed within the global Internet are configured either to ignore or to drop packets having a hop-by-hop header. As stated in [RFC7872], many network operators perceive HBH Options to be a breach of the separation between the forwarding and control planes. Therefore, several network operators configured their nodes so as to discard all packets containing the HBH Options Extension Header, while others configured nodes to forward the packet but to ignore the HBH Options. [RFC7045] also states that hop-by-hop options are not handled by many high-speed routers or are processed only on a control plane. [I-D.vyncke-v6ops-james] shows that the HBH options header cannot reliably traverse the global Internet; only small headers with 'skipable' options have some chances.

Due to such behaviors observed and described in these specifications, new hop-by-hop options are not recommended in [RFC8200] hence the usability of HBH options is severely limited.

Besides service providers' networks, other sectors such as industrial IoT networks are slowly replacing a dozen of semi-proprietary protocols in industrial automation into IP. The proper processing of the HBH options header is also required.

7. New Services

As IPv6 is being rapidly and widely deployed worldwide, more and more applications and network services are migrating to or directly adopting IPv6. More and more new services that require HBH are emerging and the HBH Options header is going to be utilized by the new services in various scenarios.

In-situ OAM (IOAM) with IPv6 encapsulation [I-D.ietf-ippm-ioam-ipv6-options] is one of the examples. IOAM in IPv6 is used to enhance diagnostics of IPv6 networks and complements other mechanisms, such as the IPv6 Performance and Diagnostic Metrics Destination Option described in [RFC8250]. The IOAM data fields are encapsulated in "option data" fields of the Hop-by-Hop Options header.

Alternate Marking Method can be used as the passive performance measurement tool in an IPv6 domain. The AltMark Option is defined as a new IPv6 extension header option to encode alternate marking technique and Hop-by-Hop Options Header is considered [I-D.ietf-6man-ipv6-alt-mark].

The Minimum Path MTU Hop-by-Hop Option is defined in [I-D.ietf-6man-mtu-option] to record the minimum Path MTU along the forward path between a source host to a destination host. This Hop-by-Hop option is intended to be used in environments like Data Centers and on paths between Data Centers as well as other environments including the general Internet. It provides a useful tool for allowing to better take advantage of paths able to support a large Path MTU.

As more services start utilizing the HBH Options header, more packets containing HBH Options are going to be injected into the networks. According to the current common configuration in most network deployments, all the packets of the new services are going to be sent to the control plane of the nodes, with the possible consequence of causing a DoS on the control plane. The packets will be dropped and the normal IP forwarding may be severely impacted. The deployment of new network services involving multi-vendor interoperability will become impossible.

8. Requirements

- * The HBH options header SHOULD NOT become a possible DDoS Vector. Therefore, the control plane MUST be preserved from unwanted incoming traffic due to HBH header present in the packet.
- * HBH options SHOULD be designed in a manner so that they don't reduce the probability of packet delivery.
- * HBH processing MUST be efficient. That is, it MUST be possible to produce implementations that perform well at a reasonable cost without endanger the security of the router.
- * The Router Alert Option MUST NOT impact the processing of other HBH options that should be processed more quickly.
- * HBH Options MAY influence how a packet is forwarded. However, with the exception of the Router Alert Option, an HBH Option MUST NOT cause control plane state to be created, modified or destroyed on the processing node. As per [RFC6398], protocol developers SHOULD avoid future use of the Router Alert Option.
- * More requirements are to be added.

9. Migration Strategies

In order to make the HBH options header usable and facilitate the ever-emerging new services to be deployed across multiple vendors' devices, the new HBH header scheme, SHOULD allow a smooth migration from old to new behavior without disruption time. Also, co-existence between old and news scheme MUST be possible.

10. Security Considerations

The same as the Security Considerations apply as in [RFC8200] for the part related with the HBH Options header.

11. IANA Considerations

This document does not include an IANA request.

12. Acknowledgements

The authors would like to acknowledge Ron Bonica, Fred Baker, Bob Hinden, Stefano Previdi, and Donald Eastlake for their valuable review and comments.

13. References

13.1. Normative References

- [I-D.ietf-6man-hbh-processing]
Hinden, R. M. and G. Fairhurst, "IPv6 Hop-by-Hop Options Processing Procedures", Work in Progress, Internet-Draft, draft-ietf-6man-hbh-processing-00, 29 January 2022, <<https://www.ietf.org/archive/id/draft-ietf-6man-hbh-processing-00.txt>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC6192] Dugal, D., Pignataro, C., and R. Dunn, "Protecting the Router Control Plane", RFC 6192, DOI 10.17487/RFC6192, March 2011, <<https://www.rfc-editor.org/info/rfc6192>>.
- [RFC6398] Le Faucheur, F., Ed., "IP Router Alert Considerations and Usage", BCP 168, RFC 6398, DOI 10.17487/RFC6398, October 2011, <<https://www.rfc-editor.org/info/rfc6398>>.
- [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>>.
- [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>>.
- [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>>.

13.2. Informative References

- [I-D.herbert-6man-eh-limits]
Herbert, T., "Limits on Sending and Processing IPv6 Extension Headers", Work in Progress, Internet-Draft, draft-herbert-6man-eh-limits-00, 22 June 2021, <<https://www.ietf.org/archive/id/draft-herbert-6man-eh-limits-00.txt>>.

- [I-D.ietf-6man-ipv6-alt-mark]
Fioccola, G., Zhou, T., Cociglio, M., Qin, F., and R. Pang, "IPv6 Application of the Alternate Marking Method", Work in Progress, Internet-Draft, draft-ietf-6man-ipv6-alt-mark-13, 31 March 2022, <<https://www.ietf.org/archive/id/draft-ietf-6man-ipv6-alt-mark-13.txt>>.
- [I-D.ietf-6man-mtu-option]
Hinden, R. M. and G. Fairhurst, "IPv6 Minimum Path MTU Hop-by-Hop Option", Work in Progress, Internet-Draft, draft-ietf-6man-mtu-option-14, 15 April 2022, <<https://www.ietf.org/archive/id/draft-ietf-6man-mtu-option-14.txt>>.
- [I-D.ietf-ippm-ioam-ipv6-options]
Bhandari, S. and F. Brockners, "In-situ OAM IPv6 Options", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-ipv6-options-07, 6 February 2022, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-ipv6-options-07.txt>>.
- [I-D.vyncke-v6ops-james]
Vyncke, É., Léas, R., and J. Iurman, "Just Another Measurement of Extension Header Survivability (JAMES)", Work in Progress, Internet-Draft, draft-vyncke-v6ops-james-01, 19 March 2022, <<https://www.ietf.org/archive/id/draft-vyncke-v6ops-james-01.txt>>.
- [RFC2711] Partridge, C. and A. Jackson, "IPv6 Router Alert Option", RFC 2711, DOI 10.17487/RFC2711, October 1999, <<https://www.rfc-editor.org/info/rfc2711>>.
- [RFC8250] Elkins, N., Hamilton, R., and M. Ackermann, "IPv6 Performance and Diagnostic Metrics (PDM) Destination Option", RFC 8250, DOI 10.17487/RFC8250, September 2017, <<https://www.rfc-editor.org/info/rfc8250>>.
- [RFC8504] Chown, T., Loughney, J., and T. Winters, "IPv6 Node Requirements", BCP 220, RFC 8504, DOI 10.17487/RFC8504, January 2019, <<https://www.rfc-editor.org/info/rfc8504>>.
- [RFC8883] Herbert, T., "ICMPv6 Errors for Discarding Packets Due to Processing Limits", RFC 8883, DOI 10.17487/RFC8883, September 2020, <<https://www.rfc-editor.org/info/rfc8883>>.

Authors' Addresses

Shuping Peng
Huawei Technologies
Beijing
China
Email: pengshuping@huawei.com

Zhenbin Li
Huawei Technologies
Beijing
China
Email: lizhenbin@huawei.com

Chongfeng Xie
China Telecom
China
Email: xiechf@chinatelecom.cn

Zhuangzhuang Qin
China Unicom
Beijing
China
Email: qinzhuangzhuang@chinaunicom.cn

Gyan Mishra
Verizon Inc.
United States of America
Email: gyan.s.mishra@verizon.com

v6ops
Internet-Draft
Intended status: Informational
Expires: 7 October 2022

G. Lencse
BUTE
J. Palet Martinez
The IPv6 Company
L. Howard
Retevia
R. Patterson
Sky UK
I. Farrer
Deutsche Telekom AG
5 April 2022

Pros and Cons of IPv6 Transition Technologies for IPv4aaS
draft-ietf-v6ops-transition-comparison-03

Abstract

Several IPv6 transition technologies have been developed to provide customers with IPv4-as-a-Service (IPv4aaS) for ISPs with an IPv6-only access and/or core network. All these technologies have their advantages and disadvantages, and depending on existing topology, skills, strategy and other preferences, one of these technologies may be the most appropriate solution for a network operator.

This document examines the five most prominent IPv4aaS technologies considering a number of different aspects to provide network operators with an easy to use reference to assist in selecting the technology that best suits their needs.

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 7 October 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	3
2. Overview of the Technologies	4
2.1. 464XLAT	4
2.2. Dual-Stack Lite	5
2.3. Lightweight 4over6	6
2.4. MAP-E	6
2.5. MAP-T	7
3. High-level Architectures and their Consequences	8
3.1. Service Provider Network Traversal	8
3.2. Network Address Translation	9
3.3. IPv4 Address Sharing	10
3.4. IPv4 Pool Size Considerations	11
3.5. CE Provisioning Considerations	13
3.6. Support for Multicast	13
4. Detailed Analysis	14
4.1. Architectural Differences	14
4.1.1. Basic Comparison	14
4.2. Tradeoff between Port Number Efficiency and Stateless Operation	14
4.3. Support for Public Server Operation	17
4.4. Support and Implementations	18
4.4.1. OS Support	18
4.4.2. Support in Cellular and Broadband Networks	18
4.4.3. Implementation Code Sizes	19
4.5. Typical Deployment and Traffic Volume Considerations	19
4.5.1. Deployment Possibilities	19
4.5.2. Cellular Networks with 464XLAT	19
4.5.3. Wireline Networks with 464XLAT	20
4.6. Load Sharing	20
4.7. Logging	21
4.8. Optimization for IPv4-only devices/applications	22
5. Performance Comparison	22

6. Acknowledgements	23
7. IANA Considerations	23
8. Security Considerations	23
9. References	24
9.1. Normative References	24
9.2. Informative References	28
Appendix A. Change Log	30
A.1. 01 - 02	30
A.2. 02 - 03	31
A.3. 03 - 04	31
A.4. 04 - 05	31
A.5. 05 - 06	31
A.6. 06 - 00-WG Item	31
A.7. 00 - 01	31
A.8. 01 - 02	31
A.9. 02 - 03	32
Authors' Addresses	32

1. Introduction

As the deployment of IPv6 becomes more prevalent, it follows that network operators will move to building single-stack IPv6 core and access networks to simplify network planning and operations. However, providing customers with IPv4 services continues to be a requirement for the foreseeable future. To meet this need, the IETF has standardized a number of different IPv4aaS technologies for this [LEN2019] based on differing requirements and deployment scenarios.

The number of technologies that have been developed makes it time consuming for a network operator to identify the most appropriate mechanism for their specific deployment. This document provides a comparative analysis of the most commonly used mechanisms to assist operators with this problem.

Five different IPv4aaS solutions are considered. The following IPv6 transition technologies are covered:

1. 464XLAT [RFC6877]
2. Dual Stack Lite [RFC6333]
3. lw4o6 (Lightweight 4over6) [RFC7596]
4. MAP-E [RFC7597]
5. MAP-T [RFC7599]

We note that [RFC6180] gives guidelines for using IPv6 transition mechanisms during IPv6 deployment addressing a much broader topic, whereas this document focuses on a small part of it.

2. Overview of the Technologies

The following sections introduce the different technologies analyzed in this document, describing some of their most important characteristics.

2.1. 464XLAT

464XLAT may use double translation (stateless NAT46 + stateful NAT64) or single translation (stateful NAT64), depending on different factors, such as the use of DNS by the applications and the availability of a DNS64 function (in the host or in the service provider network).

The customer-side translator (CLAT) is located in the customer's device, and it performs stateless NAT64 translation [RFC7915] (more precisely, stateless NAT46, a stateless IP/ICMP translation from IPv4 to IPv6). IPv4-embedded IPv6 addresses [RFC6052] are used for both source and destination addresses. Commonly, a /96 prefix (either the 64:ff9b::/96 Well-Known Prefix, or a Network-Specific Prefix) is used as the IPv6 destination for the IPv4-embedded client traffic.

In the operator's network, the provider-side translator (PLAT) performs stateful NAT64 [RFC6146] to translate the traffic. The destination IPv4 address is extracted from the IPv4-embedded IPv6 packet destination address and the source address is from a pool of public IPv4 addresses.

Alternatively, when a dedicated /64 is not available for translation, the CLAT device uses a stateful NAT44 translation before the stateless NAT46 translation.

In general, state close to the end-user network (i.e. at the CE - Customer Edge router) is not perceived as problematic as state in the operators network.

In typical deployments, 464XLAT is used together with DNS64 [RFC6147], see Section 3.1.2 of [RFC8683]. When an IPv6-only client or application communicates with an IPv4-only server, the DNS64 server returns the IPv4-embedded IPv6 address of the IPv4-only server. In this case, the IPv6-only client sends out IPv6 packets, and the CLAT functions as an IPv6 router and the PLAT performs a stateful NAT64 for these packets. In this case, there is a single translation.

Alternatively, one can say that DNS64 + stateful NAT64 is used to carry the traffic of the IPv6-only client and the IPv4-only server, and the CLAT is used only for the IPv4 traffic from applications or devices that use literal IPv4 addresses or non-IPv6 compliant APIs.

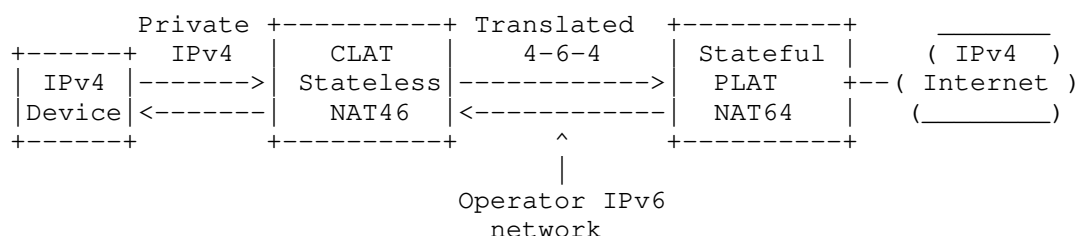


Figure 1: Overview of the 464XLAT architecture

Note: in mobile networks, CLAT is commonly implemented in the user's equipment (UE or smartphone).

2.2. Dual-Stack Lite

Dual-Stack Lite (DS-Lite) [RFC6333] was the first of the considered transition mechanisms to be developed. DS-Lite uses a 'Basic Broadband Bridging' (B4) function in the customer's CE router that encapsulates IPv4 in IPv6 traffic and sends it over the IPv6 native service-provider network to a centralized 'Address Family Transition Router' (AFTR). The AFTR performs encapsulation/decapsulation of the 4in6 [RFC2473] traffic and translates the IPv4 payload to public IPv4 source address using a stateful NAT44 [RFC2663] function.

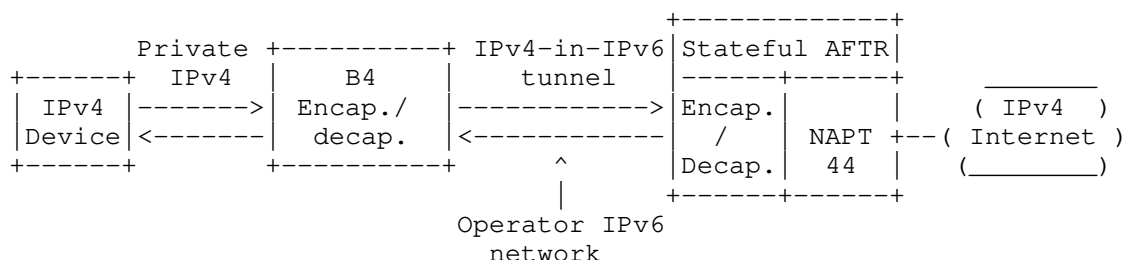


Figure 2: Overview of the DS-Lite architecture

2.3. Lightweight 4over6

Lightweight 4over6 (lw4o6) is a variant of DS-Lite. The main difference is that the stateful NAPT44 function is relocated from the centralized AFTR to the customer's B4 element (called a lwB4). The AFTR (called a lwAFTR) function therefore only performs A+P routing and 4in6 encapsulation/decapsulation.

Routing to the correct client and IPv4 address sharing is achieved using the Address + Port (A+P) model [RFC6346] of provisioning each lwB4 with a unique tuple of IPv4 address and a unique range of layer-4 ports. The client uses these for NAPT44.

The lwAFTR implements a binding table, which has a per-client entry linking the customer's source IPv4 address and allocated range of layer-4 ports to their IPv6 tunnel endpoint address. The binding table allows egress traffic from customers to be validated (to prevent spoofing) and ingress traffic to be correctly encapsulated and forwarded. As there needs to be a per-client entry, an lwAFTR implementation needs to be optimized for performing a per-packet lookup on the binding table.

Direct communication (that is, without translation) between two lwB4s is performed by hair-pinning traffic through the lwAFTR.

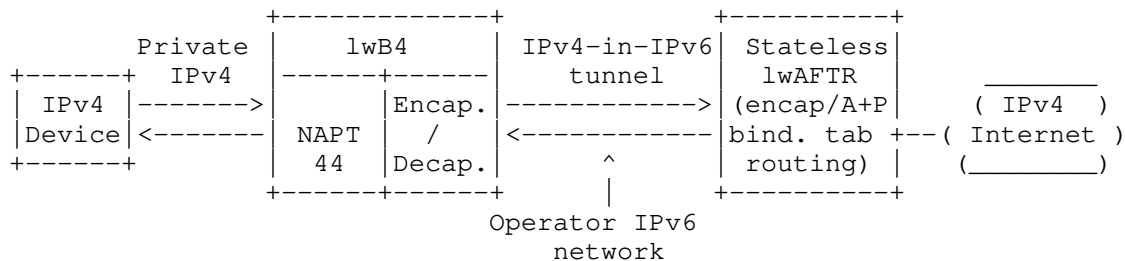


Figure 3: Overview of the lw4o6 architecture

2.4. MAP-E

Like 464XLAT (Section 2.1), MAP-E and MAP-T use [RFC6052] IPv4-embedded IPv6 addresses to represent IPv4 hosts outside the MAP domain.

MAP-E and MAP-T use a stateless algorithm to embed portions of the customer's allocated IPv4 address (or part of an address with A+P routing) into the IPv6 prefix delegated to the client. This allows

for large numbers of clients to be provisioned using a single MAP rule (called a MAP domain). The algorithm also allows for direct IPv4 peer-to-peer communication between hosts provisioned with common MAP rules.

The CE (Customer-Edge) router typically performs stateful NAPT44 [RFC2663] to translate the private IPv4 source addresses and source ports into an address and port range defined by applying the MAP rule to the delegated IPv6 prefix. The client address/port allocation size is a design parameter. The CE router then encapsulates the IPv4 packet in an IPv6 packet [RFC2473] and sends it directly to another host in the MAP domain (for peer-to-peer) or to a Border Router (BR) if the IPv4 destination is not covered in one of the CE's MAP rules.

The MAP BR is provisioned with the set of MAP rules for the MAP domains it serves. These rules determine how the MAP BR is to decapsulate traffic that it receives from client, validating the source IPv4 address and layer 4 ports assigned, as well as how to calculate the destination IPv6 address for ingress IPv4 traffic.

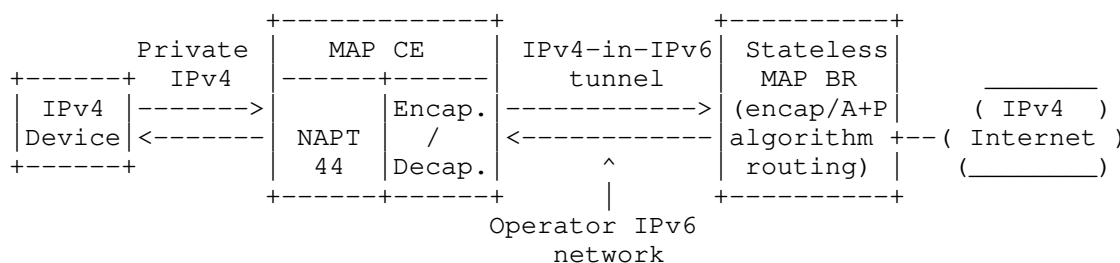


Figure 4: Overview of the MAP-E architecture

2.5. MAP-T

MAP-T uses the same mapping algorithm as MAP-E. The major difference is that double stateless translation (NAT46 in the CE and NAT64 in the BR) is used to traverse the ISP's IPv6 single-stack network. MAP-T can also be compared to 464XLAT when there is a double translation.

A MAP CE router typically performs stateful NAPT44 to translate traffic to a public IPv4 address and port-range calculated by applying the provisioned Basic MAP Rule (BMR - a set of inputs to the algorithm) to the delegated IPv6 prefix. The CE then performs stateless translation from IPv4 to IPv6 [RFC7915]. The MAP BR is provisioned with the same BMR as the client, enabling the received IPv6 traffic to be statelessly NAT64 translated back to the public IPv4 source address used by the client.

Using translation instead of encapsulation also allows IPv4-only nodes to correspond directly with IPv6 nodes in the MAP-T domain that have IPv4-embedded IPv6 addresses.

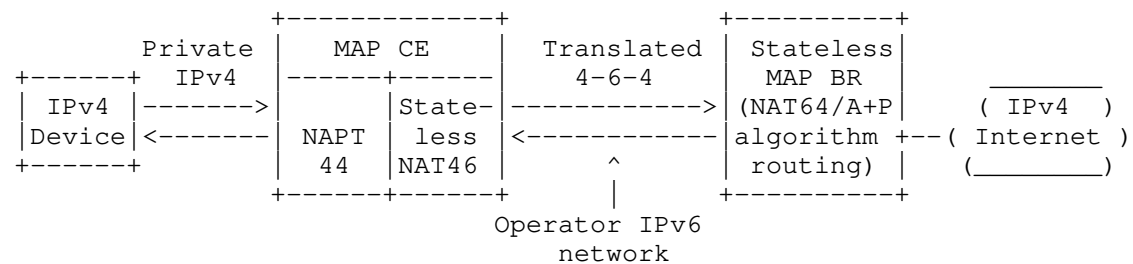


Figure 5: Overview of the MAP-T architecture

3. High-level Architectures and their Consequences

3.1. Service Provider Network Traversal

For the data-plane, there are two approaches for traversing the IPv6 provider network:

- * 4-6-4 translation
- * 4-in-6 encapsulation

	464XLAT	DS-Lite	lw4o6	MAP-E	MAP-T
4-6-4 trans.	X				X
4-6-4 encap.		X	X	X	

Table 1: Available Traversal Mechanisms

In the scope of this document, all of the encapsulation based mechanisms use IP-in-IP tunnelling [RFC2473]. This is a stateless tunneling mechanism which does not require any additional tunnel headers.

It should be noted that both of these approaches result in an increase in the size of the packet that needs to be transported across the operator's network when compared to native IPv4. 4-6-4 translation adds a 20-bytes overhead (the 20-byte IPv4 header is replaced with a 40-byte IPv6 header). Encapsulation has a 40-byte overhead (an IPv6 header is prepended to the IPv4 header).

The increase in packet size can become a significant problem if there is a link with a smaller MTU in the traffic path. This may result in traffic needing to be fragmented at the ingress point to the IPv6 only domain (i.e., the NAT46 or 4in6 encapsulation endpoint). It may also result in the need to implement buffering and fragment re-assembly in the BR node.

The advice given in [RFC7597] Section 8.3.1 is applicable to all of these mechanisms: It is strongly recommended that the MTU in the IPv6-only domain be well managed and that the IPv6 MTU on the CE WAN-side interface be set so that no fragmentation occurs within the boundary of the IPv6-only domain.

3.2. Network Address Translation

For the high-level solution of IPv6 service provider network traversal, MAP-T uses double stateless translation. First at the CE from IPv4 to IPv6 (NAT46), and then from IPv6 to IPv4 (NAT64), at the service provider network.

464XLAT may use double translation (stateless NAT46 + stateful NAT64) or single translation (stateful NAT64), depending on different factors, such as the use of DNS by the applications and the availability of a DNS64 function (in the host or in the service provider network). For deployment guidelines, please refer to [RFC8683].

The first step for the double translation mechanisms is a stateless NAT from IPv4 to IPv6 implemented as SIIT (Stateless IP/ICMP Translation Algorithm) [RFC7915], which does not translate IPv4 header options and/or multicast IP/ICMP packets. With encapsulation-based technologies the header is transported intact and multicast can also be carried.

Single and double translation results in native IPv6 traffic with a layer-4 next-header. The fields in these headers can be used for functions such as hashing across equal-cost multipaths or ACLs. For encapsulation, there is an IPv6 header followed by an IPv4 header. This results in less entropy for hashing algorithms, and may mean that devices in the traffic path that perform header inspection (e.g. router ACLs or firewalls) require the functionality to look into the payload header.

Solutions using double translation can only carry port-aware IP protocols (e.g. TCP, UDP) and ICMP when they are used with IPv4 address sharing (please refer to Section 4.3 for more details). Encapsulation based solutions can carry any other protocols over IP, too.

An in-depth analysis of stateful NAT64 can be found in [RFC6889].

3.3. IPv4 Address Sharing

As public IPv4 address exhaustion is a common motivation for deploying IPv6, transition technologies need to provide a solution for allowing public IPv4 address sharing.

In order to fulfill this requirement, a stateful NAPT function is a necessary function in all of the mechanisms. The major differentiator is where in the architecture this function is located.

The solutions compared by this document fall into two categories:

- * CGN-based approaches (DS-Lite, 464XLAT)
- * A+P-based approaches (lw4o6, MAP-E, MAP-T)

In the CGN-based model, a device such as a CGN/AFTR or NAT64 performs the NAPT44 function and maintains per-session state for all of the active client's traffic. The customer's device does not require per-session state for NAPT.

In the A+P-based model, a device (usually a CE) performs stateful NAPT44 and maintains per-session state only co-located devices, e.g. in the customer's home network. Here, the centralized network function (lwAFTR or BR) only needs to perform stateless encapsulation/decapsulation or NAT64.

Issues related to IPv4 address sharing mechanisms are described in [RFC6269] and should also be considered.

The address sharing efficiency of the five technologies is significantly different, it is discussed in Section 4.2.

lw4o6, MAP-E and MAP-T can also be configured without IPv4 address sharing, see the details in Section 4.3. However, in that case, there is no advantage in terms of public IPv4 address saving. In the case of 464XLAT, this can be achieved as well through EAMT [RFC7757].

Conversely, both MAP-E and MAP-T may be configured to provide more than one public IPv4 address (i.e., an IPv4 prefix shorter than a /32) to customers.

Dynamic DNS issues in address-sharing contexts and their possible solutions using PCP (Port Control Protocol) are discussed in detail in [RFC7393].

3.4. IPv4 Pool Size Considerations

In most networks, it is possible to, using existing data about flows to CDNs/caches or other well-known IPv6-enabled destinations, calculate the percentage of traffic that would turn into IPv6 if it is enabled on that network or part of it.

Knowing that, it is possible to calculate the IPv4 pool size required for a given number of subscribers, depending on the IPv4aaS technology being used.

Often it is assumed that each user-device (computer, tablet, smartphone) behind a NAT, could simultaneously use about 300 ports. Typically, in the case of a residential subscriber, there will be a maximum of 4 of those devices in use simultaneously, which means a total of 1,200 ports.

If for example, 80% of the traffic is expected towards IPv6 destinations, only 20% will actually be using IPv4 ports, so in our example, that will mean 240 ports required per subscriber.

From the 65,535 ports available per IPv4 address, we could even consider reserving 1,024 ports, in order to allow customers that need EAMT entries for incoming connections to System Ports (0-1023, also called well-known ports) [RFC7605], which means 64,511 ports actually available per each IPv4 address.

According to this, a /22 (1.024 public IPv4 addresses) will be sufficient for over 275,000 subscribers
($1,024 \times 64,511 / 240 = 275,246.93$).

Similarly, a /18 (16,384 public IPv4 addresses) will be sufficient for over 4,403,940 subscribers, and so on.

This is a conservative approach, which is valid in the case of 464XLAT, because ports are assigned dynamically by the NAT64, so it is not necessary to consider if one user is actually using more or less ports: Average values work well.

As the deployment of IPv6 progresses, the use of NAT64, and therefore of public IPv4 addresses, decreases (more IPv6/ports, less IPv4/ports), so either more subscribers can be accommodated with the same number of IPv4 addresses, or some of those addressed can be retired from the NAT64.

For comparison, if dual-stack is being used, any given number of users will require the same number of public IPv4 addresses. For instance, a /14 will provide 262,144 IPv4 public addresses for 262,144 subscribers, versus 275,000 subscribers being served with a only a /22.

In the other IPv4aaS technologies, this calculation will only match if the assignment of ports per subscriber can be done dynamically, which is not always the case (depending on the vendor implementation).

An alternative approximation for the other IPv4aaS technologies, when dynamically assignment of addresses is not possible, must ensure sufficient number of ports per subscriber. That means 1,200 ports, and typically, it comes to 2,000 ports in many deployments. In that case, assuming 80% of IPv6 traffic, as above, which will allow only 30 subscribers per each IPv4 address, so the closer approximation to 275,000 subscribers per our example with 464XLAT (with a /22), will be using a /19, which serves 245,760 subscribers (a /19 has 8,192 addresses, 30 subscribers with 2,000 ports each, per address).

If the CGN (in case of DS-Lite) or the CE (in case of lw4o6, MAP-E and MAP-T) make use of a 5-tuple for tracking the NAT connections, the number of ports required per subscriber can be limited as low as 4 ports per subscriber. However, the practical limit depends on the desired limit for parallel connections that any single host behind the NAT can have to the same address and port in Internet. Note that it is becoming more common that applications use AJAX and similar mechanisms, so taking that extreme limit is probably not a very a safe choice.

This extremely reduced number of ports "feature" could also be used in case the CLAT-enabled CE with 464XLAT makes use of the 5-tuple NAT connections tracking, and could also be further extended if the NAT64 also use the 5-tuple.

3.5. CE Provisioning Considerations

All of the technologies require some provisioning of customer devices. The table below shows which methods currently have extensions for provisioning the different mechanisms.

Provisioning Method	464XLAT	DS-Lite	lw4o6	MAP-E	MAP-T
DHCPv6 [RFC8415]		X	X	X	X
RADIUS [RFC8658]		[RFC6519]	X	X	X
TR-069	*	X	*	X	X
DNS64 [RFC7050]	X				
YANG [RFC7950]	[RFC8512]	X	X	X	X
DHCP4o6 [RFC7341]			X	X	

Table 2: Available Provisioning Mechanisms

*: Work started at BroadBand Forum (2021).

X: Supported by the provisioning method.

3.6. Support for Multicast

The solutions covered in this document are all intended for unicast traffic. [RFC8114] describes a method for carrying encapsulated IPv4 multicast traffic over an IPv6 multicast network. This could be deployed in parallel to any of the operator's chosen IPv4aaS mechanism.

4. Detailed Analysis

4.1. Architectural Differences

4.1.1. Basic Comparison

The five IPv4aaS technologies can be classified into 2x2=4 categories on the basis of two aspects:

- * Technology used for service provider network traversal. It can be single/double translation or encapsulation.
- * Presence or absence of NAT44 per-flow state in the operator network.

	464XLAT	DS-Lite	lw4o6	MAP-E	MAP-T
4-6-4 trans.	X				X
4-in-4 encap.		X	X	X	
Per-flow state in op. network	X	X			

Table 3: Available Provisioning Mechanisms

4.2. Tradeoff between Port Number Efficiency and Stateless Operation

464XLAT and DS-Lite use stateful NAT at the PLAT/AFTR devices, respectively. This may cause scalability issues for the number of clients or volume of traffic, but does not impose a limitation on the number of ports per user, as they can be allocated dynamically on-demand and the allocation policy can be centrally managed/adjusted.

A+P based mechanisms (lw4o6, MAP-E, and MAP-T) avoid using NAT in the service provider network. However, this means that the number of ports provided to each user (and hence the effective IPv4 address sharing ratio) must be pre-provisioned to the client.

Changing the allocated port ranges with A+P based technologies, requires more planning and is likely to involve re-provisioning both hosts and operator side equipment. It should be noted that due to the per-customer binding table entry used by lw4o6, a single customer can be re-provisioned (e.g., if they request a full IPv4 address) without needing to change parameters for a number of customers as in a MAP domain.

It is also worth noting that there is a direct relationship between the efficiency of customer public port-allocations and the corresponding logging overhead that may be necessary to meet data-retention requirements. This is considered in Section 4.7 below.

Determining the optimal number of ports for a fixed port set is not an easy task, and may also be impacted by local regulatory law, which may define a maximum number of users per IP address, and consequently a minimum number of ports per user.

On the one hand, the "lack of ports" situation may cause serious problems in the operation of certain applications. For example, Miyakawa has demonstrated the consequences of the session number limitation due to port number shortage on the example of Google Maps [MIY2010]. When the limit was 15, several blocks of the map were missing, and the map was unusable. This study also provided several examples for the session numbers of different applications (the highest one was Apple's iTunes: 230-270 ports).

The port number consumption of different applications is highly varying and e.g. in the case of web browsing it depends on several factors, including the choice of the web page, the web browser, and sometimes even the operating system [REP2014]. For example, under certain conditions, 120-160 ports were used (URL: sohu.com, browser: Firefox under Ubuntu Linux), and in some other cases it was only 3-12 ports (URL: twitter.com, browser: Iceweasel under Debian Linux).

There may be several users behind a CE router, especially in the broadband case (e.g. Internet is used by different members of a family simultaneously), so sufficient ports must be allocated to avoid impacting user experience.

Furthermore, assigning too many ports per CE router will result in waste of public IPv4 addresses, which is a scarce and expensive resource. Clearly this is a big advantage in the case of 464XLAT where they are dynamically managed, so that the number of IPv4 addresses for the sharing-pool is smaller while the availability of ports per user don't need to be pre-defined and is not a limitation for them.

There is a direct tradeoff between the optimization of client port allocations and the associated logging overhead. Section 4.7 discusses this in more depth.

We note that common CE router NAT44 implementations utilizing Netfilter, multiplexes active sessions using a 3-tuple (source address, destination address, and destination port). This means that external source ports can be reused for unique internal source and destination address and port sessions. It is also noted, that Netfilter cannot currently make use of multiple source port ranges (i.e. several blocks of ports distributed across the total port space as is common in MAP deployments), this may influence the design when using stateless technologies.

Stateful technologies, 464XLAT and DS-Lite (and also NAT444) can therefore be much more efficient in terms of port allocation and thus public IP address saving. The price is the stateful operation in the service provider network, which allegedly does not scale up well. It should be noticed that in many cases, all those factors may depend on how it is actually implemented.

Measurements have been started to examine the scalability of a few stateful solutions in two areas:

- * How their performance scales up with the number of CPU cores?
- * To what extent their performance degrades with the number of concurrent connections?

The details of the measurements and their results are available from [I-D.lencse-v6ops-transition-scalability].

We note that some CGN-type solutions can allocate ports dynamically "on the fly". Depending on configuration, this can result in the same customer being allocated ports from different source addresses. This can cause operational issues for protocols and applications that expect multiple flows to be sourced from the same address. E.g., ECMP hashing, STUN, gaming, content delivery networks. However, it should be noticed that this is the same problem when a network has a NAT44 with multiple public IPv4 addresses, or even when applications in a dual-stack case, behave wrongly if happy eyeballs is flapping the flow address between IPv4 and IPv6.

The consequences of IPv4 address sharing [RFC6269] may impact all five technologies. However, when ports are allocated statically, more customers may get ports from the same public IPv4 address, which may result in negative consequences with higher probability, e.g. many applications and service providers (Sony PlayStation Network, OpenDNS, etc.) permanently blocking IPv4 ranges if they detect that they are used for address sharing.

Both cases are, again, implementation dependent.

We note that although it is not of typical use, one can do deterministic, stateful NAT and reserve a fixed set of ports for each customer, as well.

4.3. Support for Public Server Operation

Mechanisms that rely on operator side per-flow state do not, by themselves, offer a way for customers to present services on publicly accessible layer-4 ports.

Port Control Protocol (PCP) [RFC6887] provides a mechanism for a client to request an external public port from a CGN device. For server operation, it is required with NAT64/464XLAT, and it is supported in some DS-Lite AFTR implementations.

A+P based mechanisms distribute a public IPv4 address and restricted range of layer-4 ports to the client. In this case, it is possible for the user to configure their device to offer a publicly accessible server on one of their allocated ports. It should be noted that commonly operators do not assign the Well-Known-Ports to users (unless they are allocating a full IPv4 address), so the user will need to run the service on an allocated port, or configure port translation.

Lw4o6, MAP-E and MAP-T may be configured to allocated clients with a full IPv4 address, allowing exclusive use of all ports, and non-port-based layer 4 protocols. Thus, they may also be used to support server/services operation on their default ports. However, when public IPv4 addresses are assigned to the CE router without address sharing, obviously there is no advantage in terms of IPv4 public addresses saving.

It is also possible to configure specific ports mapping in 464XLAT/NAT64 using EAMT [RFC7757], which means that only those ports are "lost" from the pool of addresses, so there is a higher maximization of the total usage of IPv4/port resources.

4.4. Support and Implementations

4.4.1. OS Support

A 464XLAT client (CLAT) is implemented in Windows 10, Linux (including Android), Windows Mobile, Chrome OS and iOS, but at the time of writing is not available in MacOS.

The remaining four solutions are commonly deployed as functions in the CE device only, however in general, except DS-Lite, the vendors support is poor.

The OpenWRT Linux based open-source OS designed for CE devices offers a number of different 'opkg' packages as part of the distribution:

- * '464xlat' enables support for 464XLAT CLAT functionality
- * 'ds-lite' enables support for DSLite B4 functionality
- * 'map' enables support for MAP-E and lw4o6 CE functionality
- * 'map-t' enables support for MAP-T CE functionality

At the time of publication some free open-source implementations exist for the operator side functionality:

- * Jool [jool] (CLAT, NAT64, EAMT, MAP-T CE, MAP-T BR).
- * VPP/fd.io [vpp] (MAP-BR, lwAFTR, CGN, CLAT, NAT64).
- * Snabb [snabb] (lwAFTR).
- * AFTR [aftr] (DSLite AFTR).

4.4.2. Support in Cellular and Broadband Networks

Several cellular networks use 464XLAT, whereas there are no deployments of the four other technologies in cellular networks, as they are neither standardised nor implemented in UE devices.

In broadband networks, there are some deployments of 464XLAT, MAP-E and MAP-T. Lw4o6 and DS-Lite have more deployments, with DS-Lite being the most common, but lw4o6 taking over in the last years.

Please refer to Table 2 and Table 3 of [LEN2019] for a limited set of deployment information.

4.4.3. Implementation Code Sizes

As hint to the relative complexity of the mechanisms, the following code sizes are reported from the OpenWRT implementations of each technology are 17kB, 35kB, 15kB, 35kB, and 48kB for 464XLAT, lw4o6, DS-Lite, MAP-E, MAP-T, and lw4o6, respectively (<https://openwrt.org/packages/start>).

We note that the support for all five technologies requires much less code size than the total sum of the above quantities, because they contain a lot of common functions (data plane is shared among several of them).

4.5. Typical Deployment and Traffic Volume Considerations

4.5.1. Deployment Possibilities

Theoretically, all five IPv4aaS technologies could be used together with DNS64 + stateful NAT64, as it is done in 464XLAT. In this case the CE router would treat the traffic between an IPv6-only client and IPv4-only server as normal IPv6 traffic, and the stateful NAT64 gateway would do a single translation, thus offloading this kind of traffic from the IPv4aaS technology. The cost of this solution would be the need for deploying also DNS64 + stateful NAT64.

However, this has not been implemented in clients or actual deployments, so only 464XLAT always uses this optimization and the other four solutions do not use it at all.

4.5.2. Cellular Networks with 464XLAT

Figures from existing deployments (end of 2018), show that the typical traffic volumes in an IPv6-only cellular network, when 464XLAT technology is used together with DNS64, are:

- * 75% of traffic is IPv6 end-to-end (no translation)
- * 24% of traffic uses DNS64 + NAT64 (1 translation)
- * Less than 1% of traffic uses the CLAT in addition to NAT64 (2 translations), due to an IPv4 socket and/or IPv4 literal.

Without using DNS64, 25% of the traffic would undergo double translation.

4.5.3. Wireline Networks with 464XLAT

Figures from several existing deployments (end of 2020), mainly with residential customers, show that the typical traffic volumes in an IPv6-only network, when 464XLAT is used with DNS64, are in the following ranges:

- * 65%-85% of traffic is IPv6 end-to-end (no translation)
- * 14%-34% of traffic uses DNS64 + NAT64 (1 translation)
- * Less than 1-2% of traffic uses the CLAT in addition to NAT64 (2 translations), due to an IPv4 socket and/or IPv4 literal.

Without using DNS64, 16%-35% of the traffic would undergo double translation.

4.6. Load Sharing

If multiple network-side devices are needed as PLAT/AFTR/BR for capacity, then there is a need for a load sharing mechanism. ECMP (Equal-Cost Multi-Path) load sharing can be used for all technologies, however stateful technologies will be impacted by changes in network topology or device failure.

Technologies utilizing DNS64 can also distribute load across PLAT/AFTR devices, evenly or unevenly, by using different prefixes. Different network specific prefixes can be distributed for subscribers in appropriately sized segments (like split-horizon DNS, also called DNS views).

Stateless technologies, due to the lack of per-flow state, can make use of anycast routing for load sharing and resiliency across network-devices, both ingress and egress; flows can take asymmetric paths through the network, i.e., in through one lwAFTR/BR and out via another.

Mechanisms with centralized NAPT44 state have a number of challenges specifically related to scaling and resilience. As the total amount of client traffic exceeds the capacity of a single CGN instance, additional nodes are required to handle the load. As each CGN maintains a stateful table of active client sessions, this table may need to be synchronized between CGN instances. This is necessary for two reasons:

- * To prevent all active customer sessions being dropped in event of a CGN node failure.

- * To ensure a matching state table entry for an active session in the event of asymmetric routing through different egress and ingress CGN nodes.

4.7. Logging

In the case of 464XLAT and DS-Lite, the user of any given public IPv4 address and port combination will vary over time, therefore, logging is necessary to meet data retention laws. Each entry in the PLAT/AFTR's generates a logging entry. As discussed in Section 4.2, a client may open hundreds of sessions during common tasks such as web-browsing, each of which needs to be logged so the overall logging burden on the network operator is significant. In some countries, this level of logging is required to comply with data retention legislation.

One common optimization available to reduce the logging overhead is the allocation of a block of ports to a client for the duration of their session. This means that logging entry only needs to be made when the client's port block is released, which dramatically reducing the logging overhead. This comes at the cost of less efficient public address sharing as clients need to be allocated a port block of a fixed size regardless of the actual number of ports that they are using.

Stateless technologies that pre-allocate the IPv4 addresses and ports only require that copies of the active MAP rules (for MAP-E and MAP-T), or binding-table (for lw4o6) are retained along with timestamp information of when they have been active. Support tools (e.g., those used to serve data retention requests) may need to be updated to be aware of the mechanism in use (e.g., implementing the MAP algorithm so that IPv4 information can be linked to the IPv6 prefix delegated to a client). As stateless technologies do not have a centralized stateful element which customer traffic needs to pass through, so if data retention laws mandate per-session logging, there is no simple way of meeting this requirement with a stateless technology alone. Thus a centralized NAT44 model may be the only way to meet this requirement.

Deterministic CGN [RFC7422] was proposed as a solution to reduce the resource consumption of logging.

4.8. Optimization for IPv4-only devices/applications

When IPv4-only devices or applications are behind a CE connected with IPv6-only and IPv4aaS, the IPv4-only traffic flows will necessarily, be encapsulated/decapsulated (in the case of DS-Lite, lw4o6 and MAP-E) and will reach the IPv4 address of the destination, even if that service supports dual-stack. This means that the traffic flow will cross thru the AFTR, lwAFTR or BR, depending on the specific transition mechanism being used.

Even if those services are directly connected to the operator network (for example, CDNs, caches), or located internally (such as VoIP, etc.), it is not possible to avoid that overhead.

However, in the case of those mechanism that use a NAT46 function, in the CE (464XLAT and MAP-T), it is possible to take advantage of optimization functionalities, such as the ones described in [I-D.ietf-v6ops-464xlat-optimization].

Using those optimizations, because the NAT46 has already translated the IPv4-only flow to IPv6, and the services are dual-stack, they can be reached without the need to translate them back to IPv4.

5. Performance Comparison

We plan to compare the performances of the most prominent free software implementations of the five IPv6 transition technologies using the methodology described in "Benchmarking Methodology for IPv6 Transition Technologies" [RFC8219].

The Dual DUT Setup of [RFC8219] makes it possible to use the existing "Benchmarking Methodology for Network Interconnect Devices" [RFC2544] compliant measurement devices, however, this solution has two kinds of limitations:

- * Dual DUT setup has the drawback that the performances of the CE and of the ISP side device (e.g. the CLAT and the PLAT of 464XLAT) are measured together. In order to measure the performance of only one of them, we need to ensure that the desired one is the bottleneck.
- * Measurements procedures for PDV and IPDV measurements are missing from the legacy devices, and the old measurement procedure for Latency has been redefined in [RFC8219].

The Single DUT Setup of [RFC8219] makes it possible to benchmark the selected device separately, but it either requires a special Tester or some trick is need, if we want to use legacy Testers. An example

for the latter is our stateless NAT64 measurements testing Throughput and Frame Loss Rate using a legacy [RFC5180] compliant commercial tester [LEN2020a]

Siitperf, an [RFC8219] compliant DPDK-based software Tester for benchmarking stateless NAT64 gateways has been developed recently and it is available from GitHub [SIITperf] as free software and documented in [LEN2021]. Originally, it literally followed the test frame format of [RFC2544] including "hard wired" source and destination port numbers, and then it has been complemented with the random port feature required by [RFC4814]. The new version is documented in [LEN2020b]

Further DPDK-based, [RFC8219] compliant software testers are being developed at the Budapest University of Technology and Economics as student projects. They are planned to be released as free software, too.

Information about the benchmarking tools, measurements and results will be made available in [I-D.lencse-v6ops-transition-benchmarking].

6. Acknowledgements

The authors would like to thank Ole Troan and Warren Kumari for their thorough review of this draft and acknowledge the inputs of Mark Andrews, Edwin Cordeiro, Fred Baker, Alexandre Petrescu, Cameron Byrne, Tore Anderson, Mikael Abrahamsson, Gert Doering, Satoru Matsushima, Mohamed Boucadair, Tom Petch, Yannis Nikolopoulos, and Havard Eidnes.

7. IANA Considerations

This document does not make any request to IANA.

8. Security Considerations

According to the simplest model, the number of bugs is proportional to the number of code lines. Please refer to Section 4.4.3 for code sizes of CE implementations.

For all five technologies, the CE device typically contains a DNS proxy. However, the user may change DNS settings. If it happens and lw4o6, MAP-E and MAP-T are used with significantly restricted port set, which is required for an efficient public IPv4 address sharing, the entropy of the source ports is significantly lowered (e.g. from 16 bits to 10 bits, when 1024 port numbers are assigned to each subscriber) and thus these technologies are theoretically less resilient against cache poisoning, see [RFC5452]. However, an

efficient cache poisoning attack requires that the subscriber operates an own caching DNS server and the attack is performed in the service provider network. Thus, we consider the chance of the successful exploitation of this vulnerability as low.

An in-depth security analysis of all five IPv6 transition technologies and their most prominent free software implementations according to the methodology defined in [LEN2018] is planned.

As the first step, an initial security analysis of 464XLAT was done in [Azz2021].

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [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>>.
- [RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <<https://www.rfc-editor.org/info/rfc2544>>.
- [RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <<https://www.rfc-editor.org/info/rfc2663>>.
- [RFC4814] Newman, D. and T. Player, "Hash and Stuffing: Overlooked Factors in Network Device Benchmarking", RFC 4814, DOI 10.17487/RFC4814, March 2007, <<https://www.rfc-editor.org/info/rfc4814>>.
- [RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, DOI 10.17487/RFC5180, May 2008, <<https://www.rfc-editor.org/info/rfc5180>>.

- [RFC5452] Hubert, A. and R. van Mook, "Measures for Making DNS More Resilient against Forged Answers", RFC 5452, DOI 10.17487/RFC5452, January 2009, <<https://www.rfc-editor.org/info/rfc5452>>.
- [RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, DOI 10.17487/RFC6052, October 2010, <<https://www.rfc-editor.org/info/rfc6052>>.
- [RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <<https://www.rfc-editor.org/info/rfc6146>>.
- [RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, DOI 10.17487/RFC6147, April 2011, <<https://www.rfc-editor.org/info/rfc6147>>.
- [RFC6180] Arkko, J. and F. Baker, "Guidelines for Using IPv6 Transition Mechanisms during IPv6 Deployment", RFC 6180, DOI 10.17487/RFC6180, May 2011, <<https://www.rfc-editor.org/info/rfc6180>>.
- [RFC6269] Ford, M., Ed., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, DOI 10.17487/RFC6269, June 2011, <<https://www.rfc-editor.org/info/rfc6269>>.
- [RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, DOI 10.17487/RFC6333, August 2011, <<https://www.rfc-editor.org/info/rfc6333>>.
- [RFC6346] Bush, R., Ed., "The Address plus Port (A+P) Approach to the IPv4 Address Shortage", RFC 6346, DOI 10.17487/RFC6346, August 2011, <<https://www.rfc-editor.org/info/rfc6346>>.
- [RFC6519] Maglione, R. and A. Durand, "RADIUS Extensions for Dual-Stack Lite", RFC 6519, DOI 10.17487/RFC6519, February 2012, <<https://www.rfc-editor.org/info/rfc6519>>.

- [RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <<https://www.rfc-editor.org/info/rfc6877>>.
- [RFC6887] Wing, D., Ed., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, DOI 10.17487/RFC6887, April 2013, <<https://www.rfc-editor.org/info/rfc6887>>.
- [RFC6889] Penno, R., Saxena, T., Boucadair, M., and S. Sivakumar, "Analysis of Stateful 64 Translation", RFC 6889, DOI 10.17487/RFC6889, April 2013, <<https://www.rfc-editor.org/info/rfc6889>>.
- [RFC7050] Savolainen, T., Korhonen, J., and D. Wing, "Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis", RFC 7050, DOI 10.17487/RFC7050, November 2013, <<https://www.rfc-editor.org/info/rfc7050>>.
- [RFC7341] Sun, Q., Cui, Y., Siodelski, M., Krishnan, S., and I. Farrer, "DHCPv4-over-DHCPv6 (DHCP 4o6) Transport", RFC 7341, DOI 10.17487/RFC7341, August 2014, <<https://www.rfc-editor.org/info/rfc7341>>.
- [RFC7393] Deng, X., Boucadair, M., Zhao, Q., Huang, J., and C. Zhou, "Using the Port Control Protocol (PCP) to Update Dynamic DNS", RFC 7393, DOI 10.17487/RFC7393, November 2014, <<https://www.rfc-editor.org/info/rfc7393>>.
- [RFC7422] Donley, C., Grundemann, C., Sarawat, V., Sundaresan, K., and O. Vautrin, "Deterministic Address Mapping to Reduce Logging in Carrier-Grade NAT Deployments", RFC 7422, DOI 10.17487/RFC7422, December 2014, <<https://www.rfc-editor.org/info/rfc7422>>.
- [RFC7596] Cui, Y., Sun, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the Dual-Stack Lite Architecture", RFC 7596, DOI 10.17487/RFC7596, July 2015, <<https://www.rfc-editor.org/info/rfc7596>>.
- [RFC7597] Troan, O., Ed., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, Ed., "Mapping of Address and Port with Encapsulation (MAP-E)", RFC 7597, DOI 10.17487/RFC7597, July 2015, <<https://www.rfc-editor.org/info/rfc7597>>.

- [RFC7599] Li, X., Bao, C., Dec, W., Ed., Troan, O., Matsushima, S., and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)", RFC 7599, DOI 10.17487/RFC7599, July 2015, <<https://www.rfc-editor.org/info/rfc7599>>.
- [RFC7605] Touch, J., "Recommendations on Using Assigned Transport Port Numbers", BCP 165, RFC 7605, DOI 10.17487/RFC7605, August 2015, <<https://www.rfc-editor.org/info/rfc7605>>.
- [RFC7757] Anderson, T. and A. Leiva Popper, "Explicit Address Mappings for Stateless IP/ICMP Translation", RFC 7757, DOI 10.17487/RFC7757, February 2016, <<https://www.rfc-editor.org/info/rfc7757>>.
- [RFC7915] Bao, C., Li, X., Baker, F., Anderson, T., and F. Gont, "IP/ICMP Translation Algorithm", RFC 7915, DOI 10.17487/RFC7915, June 2016, <<https://www.rfc-editor.org/info/rfc7915>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8114] Boucadair, M., Qin, C., Jacquenet, C., Lee, Y., and Q. Wang, "Delivery of IPv4 Multicast Services to IPv4 Clients over an IPv6 Multicast Network", RFC 8114, DOI 10.17487/RFC8114, March 2017, <<https://www.rfc-editor.org/info/rfc8114>>.
- [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>>.
- [RFC8219] Georgescu, M., Pislaru, L., and G. Lencse, "Benchmarking Methodology for IPv6 Transition Technologies", RFC 8219, DOI 10.17487/RFC8219, August 2017, <<https://www.rfc-editor.org/info/rfc8219>>.
- [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>>.

- [RFC8512] Boucadair, M., Ed., Sivakumar, S., Jacquenet, C., Vinapamula, S., and Q. Wu, "A YANG Module for Network Address Translation (NAT) and Network Prefix Translation (NPT)", RFC 8512, DOI 10.17487/RFC8512, January 2019, <<https://www.rfc-editor.org/info/rfc8512>>.
- [RFC8658] Jiang, S., Ed., Fu, Y., Ed., Xie, C., Li, T., and M. Boucadair, Ed., "RADIUS Attributes for Software Mechanisms Based on Address plus Port (A+P)", RFC 8658, DOI 10.17487/RFC8658, November 2019, <<https://www.rfc-editor.org/info/rfc8658>>.
- [RFC8683] Palet Martinez, J., "Additional Deployment Guidelines for NAT64/464XLAT in Operator and Enterprise Networks", RFC 8683, DOI 10.17487/RFC8683, November 2019, <<https://www.rfc-editor.org/info/rfc8683>>.

9.2. Informative References

- [aftr] ISC, "ISC implementation of AFTR", 2022, <<https://www.isc.org/downloads/>>.
- [Azz2021] Al-Azzawi, A. and G. Lencse, "Identification of the Possible Security Issues of the 464XLAT IPv6 Transition Technology", Infocommunications Journal, vol. 13, no. 4, pp. 10-18, DOI: 10.36244/ICJ.2021.4.2, December 2021, <https://www.infocommunications.hu/2021_4_2>.
- [I-D.ietf-v6ops-464xlat-optimization] Martinez, J. P. and A. D'Egidio, "464XLAT/MAT-T Optimization", Work in Progress, Internet-Draft, draft-ietf-v6ops-464xlat-optimization-03, 28 July 2020, <<https://www.ietf.org/archive/id/draft-ietf-v6ops-464xlat-optimization-03.txt>>.
- [I-D.lencse-v6ops-transition-benchmarking] Lencse, G., "Performance Analysis of IPv6 Transition Technologies for IPv4aaS", Work in Progress, Internet-Draft, draft-lencse-v6ops-transition-benchmarking-00, 16 October 2021, <<https://www.ietf.org/archive/id/draft-lencse-v6ops-transition-benchmarking-00.txt>>.
- [I-D.lencse-v6ops-transition-scalability] Lencse, G., "Scalability of IPv6 Transition Technologies for IPv4aaS", Work in Progress, Internet-Draft, draft-lencse-v6ops-transition-scalability-02, 7 March 2022, <<https://www.ietf.org/archive/id/draft-lencse-v6ops-transition-scalability-02.txt>>.

- [jool] NIC.MX, "Open Source SIIT and NAT64 for Linux", 2022, <<http://www.jool.mx>>.
- [LEN2018] Lencse, G. and Y. Kadobayashi, "Methodology for the identification of potential security issues of different IPv6 transition technologies: Threat analysis of DNS64 and stateful NAT64", Computers & Security (Elsevier), vol. 77, no. 1, pp. 397-411, DOI: 10.1016/j.cose.2018.04.012, 1 August 2018, <<http://www.hit.bme.hu/~lencse/publications/ECS-2018-Methodology-revised.pdf>>.
- [LEN2019] Lencse, G. and Y. Kadobayashi, "Comprehensive Survey of IPv6 Transition Technologies: A Subjective Classification for Security Analysis", IEICE Transactions on Communications, vol. E102-B, no.10, pp. 2021-2035., DOI: 10.1587/transcom.2018EBR0002, 1 October 2019, <http://www.hit.bme.hu/~lencse/publications/e102-b_10_2021.pdf>.
- [LEN2020a] Lencse, G., "Benchmarking Stateless NAT64 Implementations with a Standard Tester", Telecommunication Systems, vol. 75, pp. 245-257, DOI: 10.1007/s11235-020-00681-x, 15 June 2020, <<https://link.springer.com/article/10.1007/s11235-020-00681-x>>.
- [LEN2020b] Lencse, G., "Adding RFC 4814 Random Port Feature to Siitperf: Design, Implementation and Performance Estimation", International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems, vol 9, no 3, pp. 18-26, DOI: 10.11601/ijates.v9i3.291, 2020, <<http://ijates.org/index.php/ijates/article/view/291>>.
- [LEN2021] Lencse, G., "Design and Implementation of a Software Tester for Benchmarking Stateless NAT64 Gateways", IEICE Transactions on Communications, DOI: 10.1587/transcom.2019EBN0010, 2021, <https://www.jstage.jst.go.jp/article/transcom/E104.B/2/E104.B_2019EBN0010/_article>.
- [MIY2010] Miyakawa, S., "IPv4 to IPv6 transformation schemes", IEICE Trans. Commun., vol.E93-B, no.5, pp. 1078-1084, DOI:10.1587/transcom.E93.B.10, May 2010, <https://www.jstage.jst.go.jp/article/transcom/E93.B/5/E93.B_5_1078/_article>.

- [REP2014] Repas, S., Hajas, T., and G. Lencse, "Port number consumption of the NAT64 IPv6 transition technology", Proc. 37th Internat. Conf. on Telecommunications and Signal Processing (TSP 2014), Berlin, Germany, DOI: 10.1109/TSP.2015.7296411, July 2014, <<http://www.hit.bme.hu/~lencse/publications/TSP-2014-PC.pdf>>.
- [SIITperf] Lencse, G., "Siitperf: an RFC 8219 compliant SIIT (stateless NAT64) tester", November 2019, <<https://github.com/lencsegabor/siitperf>>.
- [snabb] Igalia, "Snabb implementation of lwAFTR", 2022, <<https://github.com/Igalia/snabb>>.
- [vpp] "VPP Implementations of IPv6-only with IPv4aaS", 2022, <<https://gerrit.fd.io/r/#/admin/projects/>>.

Appendix A. Change Log

A.1. 01 - 02

- * Ian Farrer has joined us as an author.
- * Restructuring: the description of the five IPv4aaS technologies was moved to a separate section.
- * More details and figures were added to the description of the five IPv4aaS technologies.
- * Section titled "High-level Architectures and their Consequences" has been completely rewritten.
- * Several additions/clarification throughout Section titled "Detailed Analysis".
- * Section titled "Performance Analysis" was dropped due to lack of results yet.
- * Word based text ported to XML.
- * Further text cleanups, added text on state sync and load balancing. Additional comments inline that should be considered for future updates.

A.2. 02 - 03

- * The suggestions of Mohamed Boucadair are incorporated.
- * New considerations regarding possible optimizations.

A.3. 03 - 04

- * Section titled "Performance Analysis" was added. It mentions our new benchmarking tool, siitperf, and highlights our plans.
- * Some references were updated or added.

A.4. 04 - 05

- * Some references were updated or added.

A.5. 05 - 06

- * Some references were updated or added.

A.6. 06 - 00-WG Item

- * Stats dated and added for Broadband deployments.
- * Other clarifications and references.
- * New section: IPv4 Pool Size.
- * Typos.

A.7. 00 - 01

To facilitate WGLC, the unfinished parts were moved to two new drafts:

- * New I-D for scale up measurements. (Including the results of iptables.)
- * New I-D for benchmarking measurements. (Only a stub.)

A.8. 01 - 02

Update on the basis of the AD review.

Update of the references.

A.9. 02 - 03

Nits and changes from IESG review.

Updated wrong reference to PCP.

Authors' Addresses

Gabor Lencse
Budapest University of Technology and Economics
Budapest
Magyar tudosok korutja 2.
H-1117
Hungary
Email: lencse@hit.bme.hu

Jordi Palet Martinez
The IPv6 Company
Molino de la Navata, 75
28420 La Navata - Galapagar Madrid
Spain
Email: jordi.palet@theipv6company.com
URI: <http://www.theipv6company.com/>

Lee Howard
Retevia
9940 Main St., Suite 200
Fairfax, Virginia 22031
United States of America
Email: lee@asgard.org

Richard Patterson
Sky UK
1 Brick Lane
London
EQ 6PU
United Kingdom
Email: richard.patterson@sky.uk

Ian Farrer
Deutsche Telekom AG
Landgrabenweg 151
53227 Bonn
Germany

Email: ian.farrer@telekom.de

v6ops
Internet-Draft
Intended status: Informational
Expires: 8 September 2022

G. Lencse
Szechenyi Istvan University
7 March 2022

Scalability of IPv6 Transition Technologies for IPv4aaS
draft-lencse-v6ops-transition-scalability-02

Abstract

Several IPv6 transition technologies have been developed to provide customers with IPv4-as-a-Service (IPv4aaS) for ISPs with an IPv6-only access and/or core network. All these technologies have their advantages and disadvantages, and depending on existing topology, skills, strategy and other preferences, one of these technologies may be the most appropriate solution for a network operator.

This document examines the scalability of the five most prominent IPv4aaS technologies (464XLAT, Dual Stack Lite, Lightweight 4over6, MAP-E, MAP-T) considering two aspects: (1) how their performance scales up with the number of CPU cores, (2) how their performance degrades, when the number of concurrent sessions is increased until hardware limit is reached.

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 8 September 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. Requirements Language	3
2. Scalability of iptables	3
2.1. Measurement Method	3
2.2. Performance scale up against the number of CPU cores . .	4
2.3. Performance degradation caused by the number of sessions	6
2.4. Connection tear down rate	8
3. Scalability of Jool	9
3.1. Measurement Method	10
3.2. Performance scale up against the number of CPU cores . .	10
3.3. Performance degradation caused by the number of sessions	11
3.4. Connection tear down rate	12
4. Acknowledgements	13
5. IANA Considerations	13
6. Security Considerations	13
7. References	13
7.1. Normative References	13
7.2. Informative References	14
Appendix A. Change Log	15
A.1. 00	15
A.2. 01	15
Author's Address	15

1. Introduction

IETF has standardized several IPv6 transition technologies [LEN2019] and occupied a neutral position trusting the selection of the most appropriate ones to the market.

[I-D.ietf-v6ops-transition-comparison] provides a comprehensive comparative analysis of the five most prominent IPv4aaS technologies to assist operators with this problem. This document adds one more detail: measurement data regarding the scalability of the examined IPv4aaS technologies.

Currently, this document contains only the scalability measurements of the iptables stateful NAT44 implementation. It serves as a sample to test if the disclosed results are (1) useful and (2) sufficient for the network operators.

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 BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Scalability of iptables

2.1. Measurement Method

[RFC8219] has defined a benchmarking methodology for IPv6 transition technologies. [I-D.lencse-bmwg-benchmarking-stateful] has amended it by addressing how to benchmark stateful NATxy gateways using pseudorandom port numbers recommended by [RFC4814]. It has defined a measurement procedure for maximum connection establishment rate and reused the classic measurement procedures like throughput, latency, frame loss rate, etc. from [RFC8219]. We used two of them: maximum connection establishment rate and throughput to characterize the performance of the examined system.

The scalability of iptables is examined in two aspects:

- * How its performance scales up with the number of CPU cores?
- * How its performance degrades, when the number of concurrent sessions is increased?

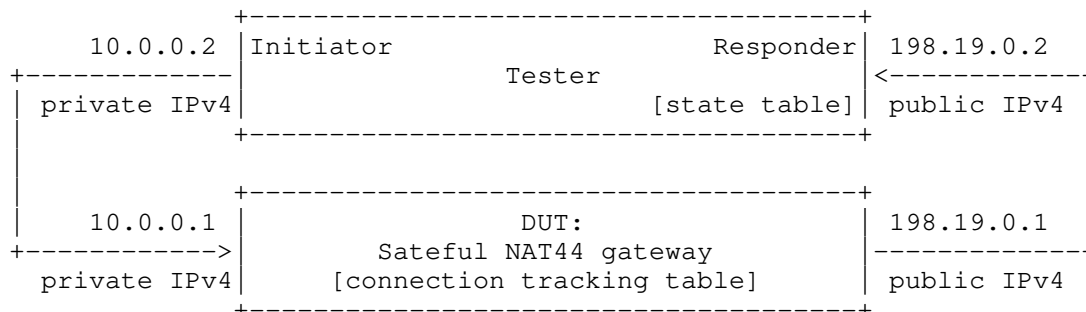


Figure 1: Test setup for benchmarking stateful NAT44 gateways

The test setup in Figure 1 was followed. The two devices, the Tester and the DUT (Device Under Test), were both Dell PowerEdge R430 servers having two 2.1GHz Intel Xeon E5-2683 v4 CPUs, 384GB 2400MHz DDR4 RAM and Intel 10G dual port X540 network adapters. The NICs of the servers were interconnected by direct cables, and the CPU clock frequency was set to fixed 2.1 GHz on both servers. They had Debian 9.13 Linux operating system with 4.9.0-16-amd64 kernel. The measurements were performed by siitperf [LEN2021] using the "stateful" branch (latest commit Aug. 16, 2021). The DPDK version was 16.11.11-1+deb9u2. The version of iptables was 1.6.0.

The ratio of number of connections in the connection tracking table and the value of the hashsize parameter of iptables significantly influences its performance. Although the default setting is `hashsize=nf_conntrack_max/8`, we have usually set `hashsize=nf_conntrack_max` to increase the performance of iptables, which was crucial, when high number of connections were used, because then the execution time of the tests was dominated by the preliminary phase, when several hundreds of millions connections had to be established. (In some cases, we had to use different settings due to memory limitations. The tables presenting the results always contain these parameters.)

The size of the port number pool is an important parameter of the benchmarking method for stateful NATxy gateways, thus it is also given for all tests.

2.2. Performance scale up against the number of CPU cores

To examine how the performance of iptables scales up with the number of CPU cores, the number of active CPU cores was set to 1, 2, 4, 8, 16 using the `"maxcpus="` kernel parameter.

The number of connections was always 4,000,000 using 4,000 different source port numbers and 1,000 different destination port numbers. Both the connection tracking table size and the hash table size was set to 2^{23} .

The error of the binary search was chosen to be lower than 0.1% of the expected results. The experiments were executed 10 times.

Besides the connection establishment rate and the throughput of iptables, also the throughput of the IPv4 packet forwarding of the Linux kernel was measured to provide a basis for comparison.

The results are presented in Figure 2. The unit for the maximum connection establishment rate is 1,000 connections per second. The unit for throughput is 1,000 packets per second (measured with bidirectional traffic, and the number of all packets per second is displayed).

num. CPU cores	1	2	4	8	16
src ports	4,000	4,000	4,000	4,000	4,000
dst ports	1,000	1,000	1,000	1,000	1,000
num. conn.	4,000,000	4,000,000	4,000,000	4,000,000	4,000,000
conntrack t. s.	2 ²³	2 ²³	2 ²³	2 ²³	2 ²³
hash table size	2 ²³	2 ²³	2 ²³	2 ²³	2 ²³
c.t.s/num.conn.	2.097	2.097	2.097	2.097	2.097
num. experiments	10	10	10	10	10
error	100	100	100	1,000	1,000
cps median	223.5	371.1	708.7	1,341	2,383
cps min	221.6	367.7	701.7	1,325	2,304
cps max	226.7	375.9	723.6	1,376	2,417
cps rel. scale up	1	0.830	0.793	0.750	0.666
throughput median	414.9	742.3	1,379	2,336	4,557
throughput min	413.9	740.6	1,373	2,311	4,436
throughput max	416.1	746.9	1,395	2,361	4,627
tp. rel. scale up	1	0.895	0.831	0.704	0.686
IPv4 packet forwarding (using the same port number ranges)					
error	200	500	1,000	1,000	1,000
throughput median	910.9	1,523	3,016	5,920	11,561
throughput min	874.8	1,485	2,951	5,811	10,998
throughput max	914.3	1,534	3,037	5,940	11,627
tp. rel. scale up	1	0.836	0.828	0.812	0.793
throughput ratio (%)	45.5	48.8	45.7	39.5	39.4

Figure 2: Scale up of iptables against the number of CPU cores
(Please refer to the next figure for the explanation of the abbreviations.)

abbreviation	explanation
num. CPU cores	number of CPU cores
src ports	size of the source port number range
dst ports	size of the destination port number range
num. conn.	number of connections = src ports * dst ports
conntrack t. s.	size of the connection tracking table of the DUT
hash table size	size of the hash table of the DUT
c.t.s/num.conn.	conntrack table size / number of connections
num. experiments	number of experiments
error	the difference between the upper and the lower bound of the binary search when it stops
cps (median/min/max)	maximum connection establishment rate (median, minimum, maximum)
cps rel. scale up	the relative scale up of the maximum connection establishment rate against the number of CPU cores
tp. rel. scale up	the relative scale up of the throughput
throughput ratio (%)	the ratio of the throughput of iptables and the throughput of IPv4 packet forwarding

Figure 3: Explanation of the abbreviations for the scale up of iptables against the number of CPU cores

Whereas the throughput of IPv4 packet forwarding scaled up from 0.91Mpps to 11.56Mpps showing a relative scale up of 0.793, the throughput of iptables scaled up from 414.9kpps to 4,557kpps showing a relative scale up of 0.686 (and the relative scale up of the maximum connection establishment rate is only 0.666). On the one hand, this is the price of the stateful operation. On the other hand, this result is quite good compared to the scale-up results of NSD (a high performance authoritative DNS server) presented in Table 9 of [LEN2020], which is only 0.52. (1,454,661/177,432=8.2-fold performance using 16 cores.) And DNS is not a stateful technology.

2.3. Performance degradation caused by the number of sessions

To examine how the performance of iptables degrades with the number connections in the connection tracking table, the number of connections was increased fourfold by doubling the size of both the source port number range and the destination port number range. Both the connection tracking table size and the hash table size was also increased four fold. However, we reached the limits of the hardware at 400,000,000 connections: we could not set the size of the hash table to 2^{29} but only to 2^{28} . The same value was used at 800,000,000 connections too, when the number of connections was only

doubled, because 1.6 billion connections would not fit into the memory.

The error of the binary search was chosen to be lower than 0.1% of the expected results. The experiments were executed 10 times (except for the very long lasting measurements with 800,000,000 connections).

The results are presented in Figure 4. The unit for the maximum connection establishment rate is 1,000,000 connections per second. The unit for throughput is 1,000,000 packets per second (measured with bidirectional traffic, and the number of all packets per second is displayed).

num. conn.	1.56M	6.25M	25M	100M	400M	800M
src ports	2,500	5,000	10,000	20,000	40,000	40,000
dst ports	625	1,250	2,500	5,000	10,000	20,000
conntrack t. s.	2 ²¹	2 ²³	2 ²⁵	2 ²⁷	2 ²⁹	2 ³⁰
hash table size	2 ²¹	2 ²³	2 ²⁵	2 ²⁷	2 ²⁸	2 ²⁸
num. exp.	10	10	10	10	10	5
error	1,000	1,000	1,000	1,000	1,000	1,000
n.c./h.t.s.	0.745	0.745	0.745	0.745	1.490	2.980
cps median	2.406	2.279	2.278	2.237	2.013	1.405
cps min	2.358	2.226	2.226	2.124	1.983	1.390
cps max	2.505	2.315	2.317	2.290	2.050	1.440
throughput med.	5.326	4.369	4.510	4.516	4.244	3.689
throughput min	5.217	4.240	3.994	4.373	4.217	3.670
throughput max	5.533	4.408	4.572	4.537	4.342	3.709

Figure 4: Performance of iptables against the number of sessions

The performance of iptables shows degradation at 6.25M connections compared to 1.56M connections very likely due to the exhaustion of the L3 cache of the CPU of the DUT. Then the performance of iptables is fairly constant up to 100M connections. A small performance decrease can be observed at 400M connections due to the lower hash table size. A more significant performance decrease can be observed at 800M connections. It is caused by two factors:

- * on average, about 3 connections were hashed to the same place
- * non NUMA local memory was also used.

We note that the CPU has 2 NUMA nodes, cores 0, 2, ... 14 belong to NUMA node 0, and cores 1, 3, ... 15 belong to NUMA node 1. The maximum memory consumption with 400,000,000 connections was below 150GB, thus it could be stored in NUMA local memory.

Therefore, we have pointed out important limitations of the stateful NAT44 technology:

- * there is a performance decrease, when approaching hardware limits
- * there is a hardware limit, beyond which the system cannot handle the connections at all (e.g. 1600M connections would not fit into the memory).

Therefore, we can conclude that, on the one hand, a well tailored hashing may guarantee an excellent scale-up of stateful NAT44 regarding the number of connections in a wide range, however, on the other hand, stateful operation has its limits resulting both in performance decrease, when approaching hardware limits and also in inability to handle more sessions, when reaching the memory limits.

2.4. Connection tear down rate

[I-D.lencse-bmwg-benchmarking-stateful] has defined connection tear down rate measurement as an aggregate measurement, that is, N number of connections are loaded into the connection tracking table of the DUT and then the entire content of the connection tracking table is deleted, and its deletion time is measured (T). Finally, the connection tear down rate is computed as: N/T .)

We have observed that the deletion of an empty connection tracking table of iptables may take a significant amount of time depending on its size. Therefore, we made our measurements more accurate by subtracting the deletion time of the empty connection tracking table from that of the filled one, thus we got the time spent with the deleting of the connections.

The same setup and parameters were used as in Section 2.3 and the experiments were executed 10 times (except for the long lasting measurements with 800,000,000 connections).

The results are presented in Figure 5.

num. conn.	1.56M	6.35M	25M	100M	400M	800M
src ports	2,500	5,000	10,000	20,000	40,000	40,000
dst ports	625	1,250	2,500	5,000	10,000	20,000
conntrack t. s.	2 ²¹	2 ²³	2 ²⁵	2 ²⁷	2 ²⁹	2 ³⁰
hash table size	2 ²¹	2 ²³	2 ²⁵	2 ²⁷	2 ²⁸	2 ²⁸
num. exp.	10	10	10	10	10	5
n.c./h.t.s.	0.745	0.745	0.745	0.745	1.490	2.980
full contr. del med	4.33	18.05	74.47	305.33	1,178.3	2,263.1
full contr. del min	4.25	17.93	72.04	299.06	1,164.0	2,259.6
full contr. del max	4.38	18.20	75.13	310.05	1,188.3	2,275.2
empty contr. del med	0.55	1.28	4.17	15.74	31.2	31.2
empty contr. del min	0.55	1.26	4.16	15.73	31.1	31.1
empty contr. del max	0.57	1.29	4.22	15.79	31.2	31.2
conn. deletion time	3.78	16.77	70.30	289.59	1,147.2	2,232.0
conn. tear d. rate	413,360	372,689	355,619	345,316	348,690	358,429

Figure 5: Connection tear down rate of iptables against the number of connections

The connection tear down performance of iptables shows significant degradation at 6.25M connections compared to 1.56M connections very likely due to the exhaustion of the L3 cache of the CPU of the DUT. Then it shows only a minor degradation up to 100M connections. A small performance increase can be observed at 400M connections due to the relatively lower hash table size. A more visible performance decrease can be observed at 800M connections. It is likely caused by keeping the hash table size constant and doubling the number of connections. The same thing that caused performance degradation of the maximum connection establishment rate and throughput, made now the deletion of the connections faster and thus caused an increase of the connection tear down rate.

We note that according to the recommended settings of iptables, 8 connections are hashed to each place of the hash table on average, but we wilfully used much smaller number (0.745 whenever it was possible) to increase the maximum connection establishment rate and thus to speed up experimenting. However, finally this choice significantly slowed down our experiments due to the very low connection tear down rate.

3. Scalability of Jool

3.1. Measurement Method

The same methodology was used as in Section 2, but now the test setup in Figure 6 was followed. The same Tester and DUT devices were used as before, but the operating system of the DUT was updated to Debian 10.11 with 4.19.0-18-amd64 kernel to meet the requirement of the jool-tools package. The version of Jool was 4.1.6. (The most mature version of Jool at the date of starting the measurements, Release Date: 2021-12-10.)

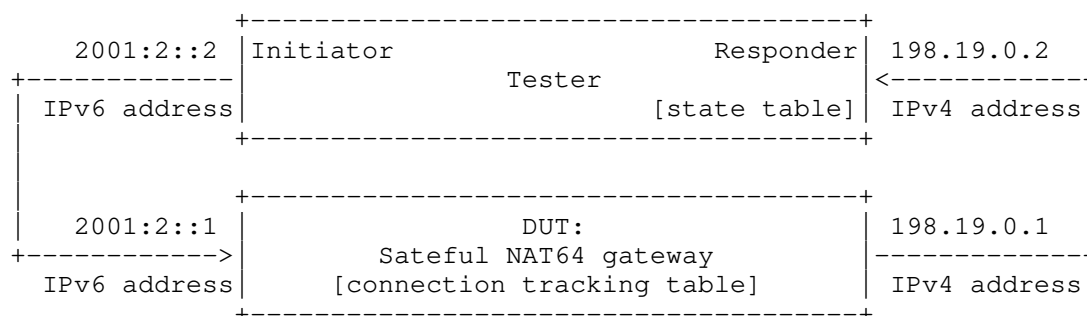


Figure 6: Test setup for benchmarking stateful NAT64 gateways

Unlike with iptables, we did not find any way to tune the hashsize or any other parameters of Jool.

3.2. Performance scale up against the number of CPU cores

The number of connections was always 1,000,000 using 2,000 different source port numbers and 500 different destination port numbers.

The error of the binary search was chosen to be lower than 0.1% of the expected results. The experiments were executed 10 times.

The results are presented in Figure 7. The unit for the maximum connection establishment rate is 1,000 connections per second. The unit for throughput is 1,000 packets per second (measured with bidirectional traffic, and the number of all packets per second is displayed).

num. CPU cores	1	2	4	8	16
src ports	2,000	2,000	2,000	2,000	2,000
dst ports	500	500	500	500	500
num. conn.	1,000,000	1,000,000	1,000,000	1,000,000	1,000,000
num. experiments	10	10	10	10	10
error	100	100	100	100	100
cps median	228.6	358.5	537.4	569.9	602.6
cps min	226.5	352.5	530.7	562.0	593.7
cps max	230.5	362.4	543	578.3	609.7
cps rel. scale up	1	0.784	0.588	0.312	0.165
throughput median	251.8	405.7	582.4	604.1	612.3
throughput min	249.8	402.9	573.2	587.3	599.8
throughput max	253.3	409.6	585.7	607.2	616.6
tp. rel. scale up	1	0.806	0.578	0.300	0.152

Figure 7: Scale up of Jool against the number of CPU cores

Both the maximum connection establishment rate and the throughput scaled up poorly with the number of active CPU cores. The increase of the performance was very low above 4 CPU cores.

3.3. Performance degradation caused by the number of sessions

To examine how the performance of Jool degrades with the number connections, the number of connections was increased fourfold by doubling the size of both the source port number range and the destination port number range. We did not reach the limits of the hardware regarding the number of connections, because unlike iptables, Jool worked also with 1.6 billion connections.

The error of the binary search was chosen to be lower than 0.1% of the expected results and the experiments were executed 10 times (except for the very long lasting measurements with 800 million and 1.6 billion connections to save execution time).

The results are presented in Figure 8. The unit for the maximum connection establishment rate is 1,000 connections per second. The unit for throughput is 1,000 packets per second (measured with bidirectional traffic, and the number of all packets per second is displayed).

num. conn.	1.56M	6.35M	25M	100M	400M	1600M
src ports	2,500	5,000	10,000	20,000	40,000	40,000
dst ports	625	1,250	2,500	5,000	10,000	40,000
num. exp.	10	10	10	10	5	5
error	100	100	100	100	1,000	1,000
cps median	480.2	394.8	328.6	273.0	243.0	232.0
cps min	468.6	392.7	324.9	269.4	243.0	230.5
cps max	484.9	397.4	331.3	280.6	244.5	233.6
throughput med.	511.5	423.9	350.0	286.5	257.8	198.4
throughput min	509.2	420.3	348.2	284.2	257.8	195.3
throughput max	513.1	428.3	352.5	290.8	260.9	201.6

Figure 8: Performance of Jool against the number of sessions

The performance of Jool shows degradation at the entire range of the number of connections. We did not analyze the root cause of the degradation yet. And we are not aware of the implementation of its connection tracking table. We also plan to check the memory consumption of Jool, what is definitely lower than that of iptables.

3.4. Connection tear down rate

Basically, the same measurement method was used as in Section 2.4, however having no parameter of Jool to tune, only a single measurement series was performed to determine the deletion time of the empty connection tracking table. The median, minimum and maximum values of the 10 measurements were 0.46s, 0.42s and 0.50s respectively.

The same setup and parameters were used as in Section 2.3 and the experiments were executed 10 times (except for the long lasting measurements with 800,000,000 connections).

The results are presented in Figure 9. The unit for the connection tear down rate is 1,000,000 connections per second.

Num. conn.	1.56M	6.35M	25M	100M	400M	1600M
src ports	2,500	5,000	10,000	20,000	40,000	40,000
dst ports	625	1,250	2,500	5,000	10,000	40,000
num. exp.	10	10	10	10	10	5
full contr. del med	0.87	2.05	7.84	36.38	126.09	474.68
full contr. del min	0.80	2.02	7.80	36.27	125.84	473.20
full contr. del max	0.91	2.09	7.94	36.80	127.54	481.38
empty contr. del med	0.46	0.46	0.46	0.46	0.46	0.46
conn. deletion time	0.41	1.59	7.38	35.92	125.63	474.22
conn. t. d. r. (M)	3.811	3.931	3.388	2.784	3.184	3.374

Figure 9: Connection tear down rate of Jool against the number of connections

The connection tear down performance of Jool is excellent at any number of connections. It is about an order of magnitude higher than its connection establishment rate and than the connection tear down rate of iptables. (A slight degradation can be observed at 100M connections.)

4. Acknowledgements

The measurements were carried out by remotely using the resources of NICT StarBED, 2-12 Asahidai, Nomi-City, Ishikawa 923-1211, Japan. The author would like to thank Shuuhei Takimoto for the possibility to use StarBED, as well as to Satoru Gonno and Makoto Yoshida for their help and advice in StarBED usage related issues.

The author would like to thank Ole Troan for his comments on the v6ops mailing list, while the scalability measurements of iptables were intended to be a part of [I-D.ietf-v6ops-transition-comparison].

5. IANA Considerations

This document does not make any request to IANA.

6. Security Considerations

TBD.

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>>.
- [RFC4814] Newman, D. and T. Player, "Hash and Stuffing: Overlooked Factors in Network Device Benchmarking", RFC 4814, DOI 10.17487/RFC4814, March 2007, <<https://www.rfc-editor.org/info/rfc4814>>.
- [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>>.

- [RFC8219] Georgescu, M., Pislaru, L., and G. Lencse, "Benchmarking Methodology for IPv6 Transition Technologies", RFC 8219, DOI 10.17487/RFC8219, August 2017, <<https://www.rfc-editor.org/info/rfc8219>>.

7.2. Informative References

- [I-D.ietf-v6ops-transition-comparison] Lencse, G., Martinez, J. P., Howard, L., Patterson, R., and I. Farrer, "Pros and Cons of IPv6 Transition Technologies for IPv4aaS", Work in Progress, Internet-Draft, draft-ietf-v6ops-transition-comparison-02, 3 March 2022, <<https://www.ietf.org/archive/id/draft-ietf-v6ops-transition-comparison-02.txt>>.
- [I-D.lencse-bmwg-benchmarking-stateful] Lencse, G. and K. Shima, "Benchmarking Methodology for Stateful NATxy Gateways using RFC 4814 Pseudorandom Port Numbers", Work in Progress, Internet-Draft, draft-lencse-bmwg-benchmarking-stateful-03, 4 March 2022, <<https://www.ietf.org/archive/id/draft-lencse-bmwg-benchmarking-stateful-03.txt>>.
- [LEN2019] Lencse, G. and Y. Kadobayashi, "Comprehensive Survey of IPv6 Transition Technologies: A Subjective Classification for Security Analysis", IEICE Transactions on Communications, vol. E102-B, no.10, pp. 2021-2035., DOI: 10.1587/transcom.2018EBR0002, 1 October 2019, <http://www.hit.bme.hu/~lencse/publications/e102-b_10_2021.pdf>.
- [LEN2020] Lencse, G., "Benchmarking Authoritative DNS Servers", IEEE Access, vol. 8, pp. 130224-130238, DOI: 10.1109/ACCESS.2020.3009141, July 2020, <<https://ieeexplore.ieee.org/document/9139929>>.
- [LEN2021] Lencse, G., "Design and Implementation of a Software Tester for Benchmarking Stateless NAT64 Gateways", IEICE Transactions on Communications, DOI: 10.1587/transcom.2019EBN0010, 1 February 2021, <<http://www.hit.bme.hu/~lencse/publications/IEICE-2020-siitperf-revised.pdf>>.

Appendix A. Change Log

A.1. 00

Initial version: scale up of iptables.

A.2. 01

Added the scale up of Jool.

Author's Address

Gabor Lencse
Szechenyi Istvan University
Gyor
Egyetem ter 1.
H-9026
Hungary
Email: lencse@size.hu

IPPM
Internet-Draft
Intended status: Standards Track
Expires: July 8, 2021

H. Song
Futurewei
Z. Li
S. Peng
Huawei Technologies
J. Guichard
Futurewei
January 4, 2021

Approaches on Supporting IOAM in IPv6
draft-song-ippm-ioam-ipv6-support-02

Abstract

It has been proposed to encapsulate IOAM tracing option data fields in IPv6 HbH options header. However, due to size of the trace data and the extension header location in the IPv6 packets, the proposal creates practical challenges for implementation, especially when other extension headers, such as a routing header, also exist and require in-network processing. We propose several alternative approaches to address this challenge, including separating the IOAM trace data to a different extension header, using the postcard-based telemetry (e.g., IOAM DEX) instead, and applying the segment IOAM trace export scheme, based on the network scenario and application requirements. We discuss the pros and cons of each approach and hope to foster standard convergence and industry adoption.

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 July 8, 2021.

Copyright Notice

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

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

Table of Contents

1. Introduction	2
2. IOAM Data Separate and Postpose	4
2.1. IOAM Trace Data Encapsulation	5
3. Segment IOAM Data Export	5
3.1. Independent of SRv6	5
3.2. Export at SRv6 node	6
4. Direct Export Option	7
5. Comparison	7
6. Security Considerations	8
7. Normative References	8
Authors' Addresses	10

1. Introduction

In-situ OAM (IOAM) [I-D.ietf-ippm-ioam-data] defines two tracing options, pre-allocated tracing option and incremental tracing option, which record hop-by-hop data along a packet's forwarding path. The draft [I-D.ietf-ippm-ioam-ipv6-options] proposes the method to encapsulate IOAM trace option data fields in IPv6. Because the tracing options requires per hop processing, such options can only be encapsulated in IPv6 Hop-by-Hop (HbH) options header. The draft [I-D.ioametal-ippm-6man-ioam-ipv6-deployment] further describes some deployment approaches.

[RFC8200] mandates that the HbH options header, if exists, must be the first extension header following the IPv6 header. However, the IOAM trace data can be large, which easily amount to tens to hundreds of bytes, making accessing other headers after it difficult or even impossible. There are practical limitations on how far the hardware

can reach into a packet in forwarding hardware. The IOAM tracing option cannot be applied if it makes other extension headers inaccessible. Even if the other headers can be reached, the deeper they are, the higher the cost to access and process them, and the lower the forwarding performance. A potentially more detrimental issue is that the incremental tracing option will expand the HbH header at each hop and push back all other headers, which keeps shifting the locations of the other extension headers, further complicating the hardware implementation and impeding the forwarding.

The issue becomes more severe when SRv6 and IOAM coexist. The Segment Routing Extension Header (SRH) [I-D.ietf-6man-segment-routing-header] is encapsulated in a routing header which is after the HbH options header. SRH itself can be large. It requires read and write operations at each SRv6 node. If it is deeply embedded in a packet and its location keeps shifting, either it is beyond the reach of hardware or the forwarding performance degrades.

We can avoid the problem by simply not using both at the same time, but apparently this is not ideal, because IOAM is an important OAM tool and it is even more wanted when SRv6 brings more operational complexity into IPv6 networks.

Our second recourse is to limit the IOAM to SRv6 nodes only. That is, consider SRv6 as an overlay tunnel over IPv6 and apply the IOAM pipe mode as discussed in [I-D.song-ippm-ioam-tunnel-mode], which only collects data at each SRv6 nodes. To realize this, [I-D.ali-spring-ioam-srv6] describes an approach that encapsulates the IOAM option data fields in an SRH TLV. [I-D.song-6man-srv6-pbt] and [I-D.ietf-6man-spring-srv6-oam] describe another approach to enable postcard-based telemetry for SRv6 without needing IOAM option encapsulation. In either case, the SRH is close to the packet front and its location is fixed. [I-D.song-spring-siam] proposes to support IOAM in the payload of the dedicated SRv6 probing packets only. While these approaches are useful for use cases that only need to monitor the segment end points, it fails to cover all the IPv6 nodes in a network.

So the proposition of this draft is, suppose we need to apply IOAM on all nodes in an SRv6 network, how we can amend the approach in [I-D.ietf-ippm-ioam-ipv6-options] or use alternative approaches to circumvent the aforementioned issues. In this draft, we propose three such approaches: (1) separating the IOAM trace data to a different extension header, (2) using the postcard-based telemetry (e.g., IOAM DEX) instead, and (3) applying the segment IOAM trace export scheme. We discuss the pros and cons of each approach and hope to foster standard convergence and industry adoption.

2. IOAM Data Separate and Postpose

An IOAM trace type data fields contain two parts: instruction and trace data. Although by convention the trace data part immediately follow the instruction part, there is not fundamental reason why these two parts must stick together. This observation provides us an optimization opportunity to amend the original proposal in [I-D.ietf-ippm-ioam-ipv6-options].

We separate the IOAM trace type data fields into the instruction part and the trace data part. We encapsulate only the instruction part in the HbH options header, and encapsulate the trace data part in another metadata extension header after all the IPv6 extension headers and before upper layer protocol headers. This arrangement allows high performance hardware implementation. When using the incremental data trace, even if the data trace size increases at each node, all IPv6 extension headers remain intact and new data is inserted at a fixed location.

Figure 1 shows the HbH option format for IOAM trace type instruction. The field specification is identical to that in [RFC8200] and [I-D.ietf-ippm-ioam-data].

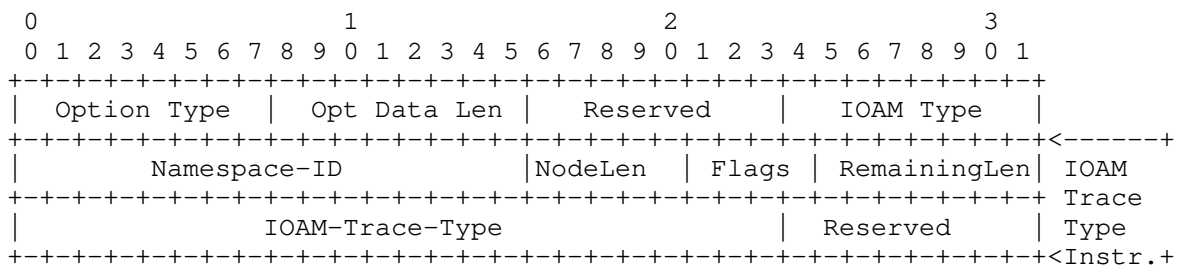


Figure 1: HbH Option Format for IOAM Trace Type Instruction

Figure 2 shows the TLV option format for IOAM trace type data. The IOAM trace type data format is compliant with [I-D.ietf-ippm-ioam-data].

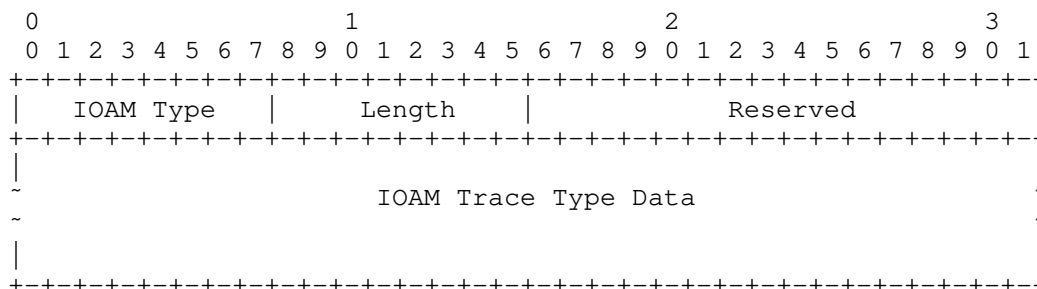


Figure 2: Option Format for IOAM Trace Type Data

2.1. IOAM Trace Data Encapsulation

We have basically two methods to encapsulate the IOAM trace data. First, we can define a new IPv6 extension header which is dedicated to metadata. Once standardized, this extension header can also be used to host potential metadata from other applications such as NSH for SFC [RFC8300]. Second, this option can be carried as a TLV option in another existing extension header such as the destination header. The only requirement is that this extension header should be the last one in the extension header chain. The first method is cleaner but it requires extra standard effort; the second method is simpler but it needs to overcome the access constraints exerted by [RFC8300].

3. Segment IOAM Data Export

If the overhead of the IOAM trace type data fields is under control, we may still manage to encapsulate both instruction and data in HbH options header as in [I-D.ietf-ippm-ioam-ipv6-options]. To this end, we introduce two sub-approaches.

3.1. Independent of SRv6

[I-D.song-ippm-segment-ioam] proposes an enhancement to IOAM trace type which can configure the allowable overhead of the IOAM trace type data fields. Once the trace data size is up to the limit at a network node (i.e., a segment or a fixed number of network nodes are traversed), the trace data will be stripped and exported so room is made to accommodate new trace data from nodes in the next segment of the forwarding path.

This approach requires some moderate updates to the IOAM trace type data fields, as described in [I-D.song-ippm-segment-ioam]. Figure 3 shows the format of the HbH Option Header containing Segment IOAM trace type data fields. A flag bit (#23) in the Flags field is used

to indicate the current header is a segment IOAM header. In this context, the last octet in the IOAM header is partitioned into two 4-bit nibbles. The first nibble (SSize) is used to save the segment size and the second nibble (RHop) is used to save the remaining hops. This limits the maximum segment size to 15.

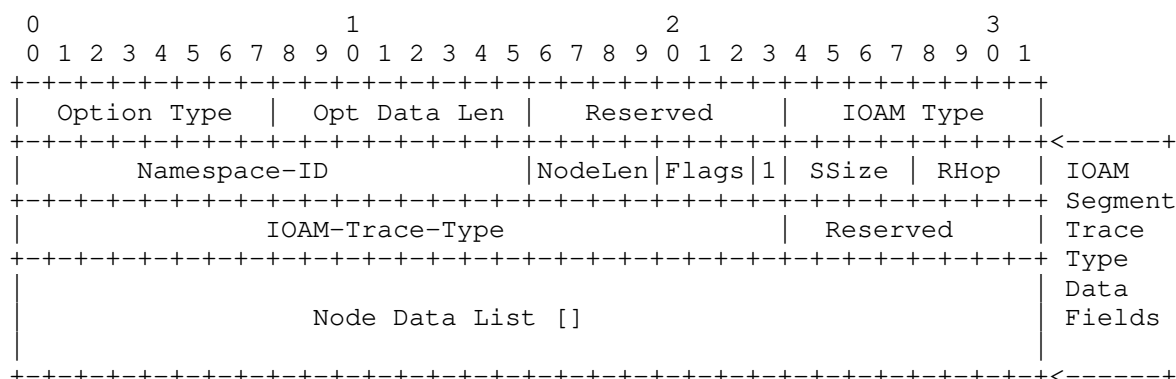


Figure 3: HbH Option Format for Segment IOAM Trace Type Data Fields

At the beginning of each segment, the segment size (SSize) and the remaining hops (RHop) are initialized: RHop is set to equal to SSize. At each hop, if RHop is not zero, the node data is added to the node data list and then RHop is decremented by 1. If RHop is equal to 0 when receiving the packet, the node needs to remove (in incremental trace option) or clear (in pre-allocated trace option) the IOAM node data list and reset RHop to SSize.

In this case, if we use the IOAM pre-allocated trace type, the size and location of each IPv6 extension header is fixed and predictable, and the hardware capability and performance can be guaranteed.

3.2. Export at SRv6 node

Whenever a packet with the IOAM option reaches a SRv6 node which needs to access the SRH, we can configure the node to export immediately the IOAM trace data accumulated so far. In this case, basically at each SRv6 node, the HbH header size is fixed and the header contains an IOAM option with only the instruction part. After the SRH processing, this node can add local IOAM trace data in the HbH option header before forwarding the packet.

The incremental trace type can be used in this approach. In an extreme case when every node is also an SRv6 node, this approach regresses to a per-hop postcard-based telemetry approach as described in [I-D.song-ippm-postcard-based-telemetry]. In this case, the HbH

option for IOAM can even be avoided altogether if we can find a way to simply mark the packet for postcard export.

4. Direct Export Option

As an embodiment of the PBT-I approach introduced in [I-D.song-ippm-postcard-based-telemetry], IOAM Direct Export (DEX) Option Type discussed in [I-D.ioamteam-ippm-ioam-direct-export] can be used to replace the IOAM trace type. IOAM DEX only needs to encapsulate a fix-size instruction header in the HbH option header.

Figure 4 shows the HbH option format for IOAM DEX type fields. The field specification is identical to that in [RFC8200] and [I-D.ioamteam-ippm-ioam-direct-export].

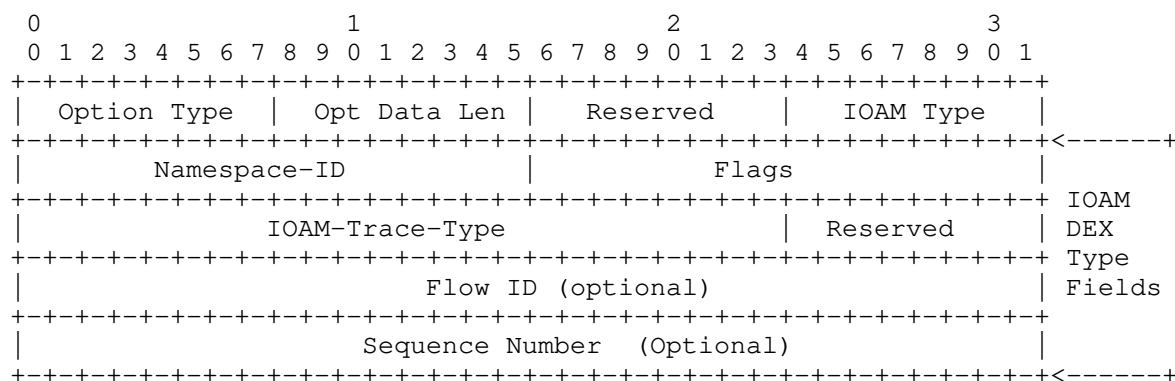


Figure 4: HbH Option Format for IOAM DEX Type Fields

5. Comparison

The following table compares the existing approach and the four other alternative approaches proposed in this draft.

Approach	Pros	Cons
IOAM Trace in HbH	Comply w/ IOAM Data Spec	Variable and long HbH header impeding access of later extension headers
IOAM Trace Data Separate and Postpose (Sec. 2)	Fix-size and short HbH header, good for later extension header access	Need extra extension header to hold trace data
Segment IOAM Data Export (Sec. 3.1)	Fix-size and controllable HbH header size	Need to enhance IOAM trace type data field spec.
Trace Export at SRv6 nodes (Sec. 3.2)	Can be done through configuration	Specific to SRv6; No better than PB & IOAM DEX in the worst case
IOAM Direct Export in HbH (Sec. 4)	Comply w/ IOAM DEX Spec; Fix-size and short HbH	Need export data correlation

Figure 5: Comparison of Different Approaches

6. Security Considerations

TBD.

7. Normative References

[I-D.ali-spring-ioam-srv6]

Ali, Z., Gandhi, R., Filsfils, C., Brockners, F., Nainar, N., Pignataro, C., Li, C., Chen, M., and G. Dawra, "Segment Routing Header encapsulation for In-situ OAM Data", draft-ali-spring-ioam-srv6-03 (work in progress), November 2020.

[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-6man-spring-srv6-oam]

Ali, Z., Filsfils, C., Matsushima, S., Voyer, D., and M. Chen, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", draft-ietf-6man-spring-srv6-oam-08 (work in progress), October 2020.

[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.ietf-ippm-ioam-ipv6-options]

Bhandari, S., Brockners, F., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lapukhov, P., Spiegel, M., Krishnan, S., Asati, R., and M. Smith, "In-situ OAM IPv6 Options", draft-ietf-ippm-ioam-ipv6-options-04 (work in progress), November 2020.

[I-D.ioametal-ippm-6man-ioam-ipv6-deployment]

Bhandari, S., Brockners, F., Mizrahi, T., Kfir, A., Gafni, B., Spiegel, M., Krishnan, S., and M. Smith, "Deployment Considerations for In-situ OAM with IPv6 Options", draft-ioametal-ippm-6man-ioam-ipv6-deployment-03 (work in progress), March 2020.

[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.song-6man-srv6-pbt]

Song, H., "Support Postcard-Based Telemetry for SRv6 OAM", draft-song-6man-srv6-pbt-01 (work in progress), October 2019.

[I-D.song-ippm-ioam-tunnel-mode]

Song, H., Li, Z., Zhou, T., and Z. Wang, "In-situ OAM Processing in Tunnels", draft-song-ippm-ioam-tunnel-mode-00 (work in progress), June 2018.

[I-D.song-ippm-postcard-based-telemetry]

Song, H., Zhou, T., Li, Z., Mirsky, G., Shin, J., and K. Lee, "Postcard-based On-Path Flow Data Telemetry using Packet Marking", draft-song-ippm-postcard-based-telemetry-08 (work in progress), October 2020.

- [I-D.song-ippm-segment-ioam]
Song, H. and T. Zhou, "Control In-situ OAM Overhead with Segment IOAM", draft-song-ippm-segment-ioam-01 (work in progress), April 2018.
- [I-D.song-spring-siam]
Song, H. and T. Pan, "SRv6 In-situ Active Measurement", draft-song-spring-siam-00 (work in progress), December 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>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <<https://www.rfc-editor.org/info/rfc8300>>.

Authors' Addresses

Haoyu Song
Futurewei
USA

Email: haoyu.song@futurewei.com

Zhenbin Li
Huawei Technologies
China

Email: lizhenbin@huawei.com

Shuping Peng
Huawei Technologies
China

Email: pengshuping@huawei.com

James Guichard
Futurewei
USA

Email: james.n.guichard@futurewei.com

IPv6 Operations (v6ops) Working Group
Internet Draft
Intended status: Informational
Expires: Aug. 2022

X. Xiao
Huawei Technologies
E. Metz
KPN
G. Mishra
Verizon Inc.
February 16, 2022

Neighbor Discovery Protocol Deployment Guidelines
draft-xiao-v6ops-nd-deployment-guidelines-01

Abstract

Neighbor Discovery (ND) is an integral part of IPv6 first-hop. Due to limitation of certain L2 media's support to ND, a number of issues can happen in certain scenarios. Solutions for these issues have been designed. These issues and solutions are summarized in RFC3756, RFC6583, RFC9099. However, there is no guideline on how to prevent the issues or how to select the proper solutions.

This document analyzes the existing solutions and summarizes their wisdoms into an insight: isolating hosts into different L2 links or different L3 subnets can be effective in preventing ND issues. In deployment scenarios where the ND issues can occur, this prevention approach can be more effective than deploying the corresponding solutions to solve the issues. Based on this insight, a set of guidelines is proposed for future ND deployments. These guidelines describe where to isolate hosts in L2 or in subnet to prevent ND issues, and how to select suitable solutions for the remaining issues. This will likely simplify ND deployment. The impact of such isolation to other components of IPv6 first-hop is also analyzed. The impact appears small. Therefore, the guidelines will likely simplify the overall IPv6 first-hop deployment.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents

at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

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

Table of Contents

1. Introduction.....	2
1.1. Terminology.....	3
2. Summary of ND Issues and Existing Solutions.....	4
2.1. Multicast Issues and Solutions.....	4
2.2. DAD-unreliable Issues and Solutions.....	6
2.3. On-demand NCE Installation Issues and Solutions.....	6
2.4. Security Issues and Solutions.....	7
2.5. Observations on the Solutions and an Insight Learned.....	9
3. Isolating Hosts in L2 and in Subnet to Simplify ND Deployment.....	11
3.1. Applicability of L2 Isolation and Subnet Isolation.....	11
3.2. Deployment Guidelines.....	14
4. Impact of L2 Isolation and Subnet Isolation to Other Components of IPv6 First-hop.....	16
5. Applying the Guidelines to Real World Scenarios.....	18
6. Security Considerations.....	19
7. IANA Considerations.....	19
8. References.....	19
8.1. Informative References.....	19
9. Acknowledgments.....	22

1. Introduction

ND is an integral part of IPv6 first-hop. Due to limitation of certain L2 media's support to the protocol, a number of issues have

been discovered, and the corresponding solutions have been designed. These issues and solutions are dispersed in more than 20 RFCs.

[RFC6583] summarized the issues and solutions up to 2012. [RFC9099] summarized known IPv6 security issues, including ND issues, up to 2021. Wireless ND (WiND) [RFC6775][RFC8505][RFC8928][RFC8929] discussed ND issues and solutions in Low-Power and Lossy Networks (LLNs) [RFC7102]. However, two ND deployment problems still exist:

1. None of [RFC6583], [RFC9099] and WiND provides guidelines on how to prevent the issues;
2. Some issues have multiple solutions. There is no guideline on how to select the suitable solutions depending on the deployment scenarios.

[RFC8273] recommends using "Unique IPv6 Prefix per Host" as a best practice for ND. By doing so, certain ND issues can be prevented. So, it partially solved Problem 1 above. But [RFC8273] cannot be used everywhere. In fact, some concerns about [RFC8273] have been expressed in [8273D].

This document aims to solve both problems by providing deployment guidelines for ND. Depending on the usage scenarios, the guidelines firstly try to prevent the issues as much as possible, then recommend suitable solutions for the remaining issues. This can result in the simplest ND deployment.

1.1. Terminology

Familiarity with [ND] and [SLAAC] are prerequisite to understand this document. Familiarity with the ND issues and solutions discussed in [RFC6583] and [RFC9099] Section 2.3 are also essential to understand this document.

Some frequently used terms are defined in this section.

L2 isolation for hosts - One host per link. Consequently, A host cannot send packets via the L2 medium to any other hosts. Router will be the only node reachable in L2. This can be realized on P2P media, or on multi-access media with Private VLAN [PVLAN] or Split Horizon [TR177] or wireless isolation [W-Iso] to prevent hosts from communicating with each other in L2.

Subnet isolation for hosts - One host per subnet. This can be done by assigning a unique subnet prefix to each host. Therefore, it is also known as "unique prefix per host" [RFC8273].

NCE - Neighbor Cache Entry, a binding of a neighbor's IPv6 address and link layer address.

2. Summary of ND Issues and Existing Solutions

This section summarizes the main ND issues and solutions. More information can be found in [RFC6583][RFC6775][RFC9099].

2.1. Multicast Issues and Solutions

ND uses multicast extensively for Node Solicitations (NSs), Router Solicitations (RSs) and Router Advertisements (RAs). When multicast messages are sent over an L2 medium, they may be mapped into L2 broadcast messages. For many wireless media, L2 broadcast may consume more network resources than multiple P2P unicast combined [RFC6775][RFC7668], and have lower probability of delivery. In addition, multicast has no acknowledgement. Consequently, multicast may cause a number of issues:

- . Multicast-resource-consumption issue: multicast in L3 and the resulted broadcast in L2 may consume many times more resources than unicast;
- . Multicast-power-consumption issue: multicast consumes more power of the sender; In addition, an L2 broadcast message may reach more nodes than the L3 multicast intended. This may consume some receiving nodes' power unnecessarily. For power constrained nodes, this is an issue;
- . Multicast-reliability issue: some multicast messages may not reach the destinations. Sleeping nodes may not respond to a multicast message. As a result, protocol procedures that rely on multicast can be unreliable.

Existing ND solutions that can prevent or address multicast issues include:

- . Mobile Broadband IPv6 (MBBv6) and Fixed Broadband IPv6 (FBBv6): MBBv6 is defined in "IPv6 in 3GPP EPS" [RFC6459] and "IPv6 for 3GPP Cellular Hosts" [RFC7066]. MBBv6 isolates each host in L2 by putting it in a P2P link with the router. FBBv6 is defined in "IPv6 in the context of TR-101" [TR177]. FBBv6 also isolates each host in L2, either by putting it in a P2P link with the router, or by implementing split horizon on Ethernet to prevent direct host communication. Note that a host here is either a mobile User Equipment (UE), or a routed Residential Gateway (RG), or a real host (e.g. a computer) behind a bridged RG. The router here is either the mobile gateway or the Broadband Network Gateway (BNG). MBBv6 and FBBv6 also give each host a unique prefix and thus put it in its own subnet. Consequently, every host can only communicate with the router in both L2 and L3, and there is effectively no multicast.

Note 1: because in FBBv6, bridged RG situation is very similar to routed RG situation, except that in the former, the hosts are real hosts (e.g. computers), while in the latter, the hosts are the routed RGs. For description simplicity, bridged RGs will not be further discussed in the document.

- . Wireless ND (WiND): the solution is defined in a series of RFCs [RFC6775][RFC8505][RFC8928][RFC8929]. WiND changes host and router behaviors to use multicast only for router discovery and not for other protocol procedures. The key points are (1) hosts use unicast to proactively register their addresses at the routers; (2) routers use unicast to communicate with hosts, and become the central address register and arbitrator for the hosts. Router also proactively installs Neighbor Cache Entries (NCEs) for the hosts; (3) each host communicates only with the router. Consequently, multicast is largely eliminated;
- . Unique IPv6 Prefix per Host (UPPH) [RFC8273]: this solution gives each host a unique prefix, thus puts it in its own subnet. Multicast is greatly reduced, because (1) a host will not use multicast for address resolution for other hosts, because there is no other host in the same subnet; (2) downstream multicast from router to hosts are eliminated and turned into unicast ([RFC8273] Section 4 Paragraph 3).

Note 2: the pros and cons of RFC8273 will be briefly reviewed in Section 3.1. An in-depth discussion can be found in [8273D].

- . IP Point to Point Ethernet Subnet Model [P2Peth]: if progressed, this draft may provide a solution similar to

RFC8273, making use of unique prefix per host to reduce multicast.

2.2. DAD-unreliable Issue and Solutions

Duplicate Address Detection (DAD) uses absence of response as indication of no duplicate. This can be unreliable for two reasons. Firstly, the Neighbor Solicitation or the Neighbor Advertisement messages may be lost in transmission, especially in wireless environment. Secondly, sleeping nodes may not respond to the DAD messages.

Existing ND solutions that can prevent or address DAD-unreliable issue include:

- . MBBv6 and FBBv6: for MBBv6, the UE's Interface ID (IID) is assigned by the mobile gateway, and is guaranteed to be unique [RFC6459]. There is no need for DAD. For FBBv6, the RG's Global Unicast Address (GUA) prefix is unique. There will be no duplicate for GUA address, and no DAD will be performed. RG will perform DAD for its Link Local Address (LLA), but only to the BNG [TR177]. In such an environment, there is no sleeping node or lossy media. DAD has no reliability issue.
- . WiND: every host must register its address at the router before using it. The registration will succeed only if the router explicitly approves it, and the router will not approve if it is a duplicate. Therefore, the DAD procedure becomes reliable.
- . RFC8273: For GUA addresses, since each host is given a different prefix, duplicate address will not exist. For link local address, since each subnet has only one host, there is no possibility of duplication in the subnet. A duplicate link local address may exist in another subnet but that does not matter. Therefore, DAD-unreliable issue is prevented.

2.3. On-demand NCE Installation Issue and Solutions

ND address resolution is on-demand. It is done only when a packet needs to be sent but the link layer address of the on-link destination is unknown. Consequently, the packet has to be queued until link layer address is determined. This increases delay and may affect application performance. For high performance environment, this can be an issue.

Existing ND solutions that can prevent or address on-demand NCE installation issue include:

- . MBBv6 and FBBv6: MBBv6 and some FBBv6 use IPv6 over PPP [RFC5072]. In this case, there is no need to find out the link layer address before sending a packet on the PPP interface. In other words, there is no NCE installation issue. Some FBBv6 implementations use IPoE. There is a need for NCE. But because every host is given a unique prefix by the BNG in FBBv6, the BNG needs to know the host's link layer address which uniquely identify the host in order to do so. Therefore, the BNG can install NCE when assigning a prefix to the host, not waiting until a data packet is to be sent to that host. On-demand NCE installation issue is therefore avoided in this case too.
- . Wireless ND (WiND): when hosts register their IPv6 addresses, they will also present their link layer address. Therefore, NCE entries can be installed proactively before data packets arrive.
- . Gratuitous Neighbor Discovery [GRAND]: this solution changes router and host behaviors to allow routers to proactively create an NCE when a new IPv6 address is assigned to a host, and to recommend that hosts send unsolicited Neighbor Advertisements upon having a new IPv6 address. It can be considered as the IPv6 equivalent of Gratuitous ARP in IPv4.

2.4. Security Issues and Solutions

ND assumes all nodes can be trusted. Otherwise, ND has various security issues. These issues are described in [RFC3756][RFC6583] and [RFC9099]. They are briefly summarized here.

The security issues can be classified into two categories:

- . Redirect attacks, in which a malicious node redirects packets destined for the first-hop router or a legitimate receiver to itself. This is usually done by faking the source IPv6 address to pretend to be another node, or by sending Router Advertisement to pretend to be a router. For example:
 - o an attacker can send out Router Advertisement claiming itself as the first-hop router, and redirect hosts' traffic to itself.

- o an attacker can send Neighbor Advertisements to the first-hop router, spoofing the IPv6 address of another node while using its own link layer address. This will redirect traffic from the router to the victim to the attacker instead.
- . Denial-of-Service (DoS) attacks, in which a malicious node prevents communication between the victim node and other nodes. For example:
 - o Whenever a host performs DAD, an attacker can reply to claim that the selected address is in use. The host will not be able to get an address.
 - o /64 scan attack: an attacker sends packets to up to 2^{64} mostly non-existing hosts, forcing the first-hop router to try to install NCEs for this huge number of non-existing hosts. If unprotected, the router will run out of resource and stop functioning. Note that for other attacks to happen, the attacker must be on-link. But for this /64 scan attack, the attacker can be off-link.

It is worth noting that redirect attacks are generally considered as more harmful than DoS attacks. Therefore, higher priority is given to protect against redirect attacks.

Existing ND solutions that can prevent or address the security issues include:

- . MBBv6 and FBBv6: because every host is isolated in L2, all on-link security issues are prevented. Because every host has its own prefix, and the mobile gateway or BNG maintains state information for the prefix instead of for individual IPv6 address, off-link /64 scan attack will not cause harm - the mobile gateway or BNG will not create an NCE for every IP address in the /64. Such attack messages can simply be dropped.
- . WiND: normally, every host is isolated in L2 and can only communicate with the first-hop router. Therefore, all on-link security issues are prevented. But if hosts are not isolated in L2, e.g. when there is a bridge between the hosts and the router, then on-link security issue can happen. Off-link /64 scan attack will not cause harm because in WiND, NCE installation is proactive. In other words, the router will not create any NCE when receiving such messages. Such attack messages can simply be dropped.

- . RFC8273: by giving each host a different prefix and keep track of host-prefix binding, an attacker host cannot change the NCE at the router for another host by sending Neighbor Advertisement with a spoofed source IPv6 address of that host. The attacker thus cannot redirect traffic from router to that host to itself. The router will maintain only one NCE entry for each prefix/host. Therefore, off-link /64 scan attack will not cause harm. RFC8273's protection effect against other security issues depends on whether the hosts are also isolated in L2. If yes, all the on-link security issues will be prevented. If not, certain on-link security issues remain.
- . Source Address Validation Improvement [SAVI], Router Advertisement Guard [RA-Guard][RA-Guard+]: these solutions protect against redirect attacks and some DoS attacks. SAVI binds an address to a port, and rejects claims from other ports for that address. Therefore, a node cannot spoof the IP address of another node. RA-Guard and RA-Guard+ only allow RAs from a port that a router is connected. Therefore, nodes on other ports cannot pretend to be a router. In order to protect against some other DoS attacks, e.g. off-link /64 scan attack, other security measures are needed, e.g. rate limiting on NSs, and filtering on NAs/RAs.
- . Secure Neighbor Discovery [SeND] and Cryptographically Generated Addresses [CGA]: these solutions have tried to make ND secure, but have not been widely deployed. They will not be further discussed in this document.

2.5. Observations on the Solutions and an Insight Learned

MBBv6 and FBBv6 prevent or solve all the ND issues. These solutions have two common characteristics that are effective for preventing or solving ND issues:

- (1) Hosts (i.e. UEs or RGs or real hosts) are isolated in L2 and in different subnets.
- (2) The first-hop router (i.e. mobile gateway or BNG) maintains some state information across reboots, e.g. prefix to host binding. The router also maintains some controls over the hosts, e.g. which prefix each host gets.

WiND prevents or solves all the ND issues in its deployment scenarios, e.g. Low power and Lossy Networks (LLNs) [RFC7102]. WiND also has two characteristics:

- (1) Hosts are normally isolated in L2 but not in different subnets. In the rare cases where hosts are not isolated in L2, e.g. because there is a bridge between the hosts and the router, then some ND issues like on-link security issues may happen, and additional solutions will be needed to address those issues.
- (2) The first-hop router maintains some state information across reboots, e.g. host's address registration result including IPv6 address to link layer address binding. The router also uses such state information to maintain some controls over the hosts, e.g. which host wins when there is an address contention.

RFC8273 solves certain ND issues but not all. It has two characteristics:

- (1) Hosts are isolated in different subnets, but RFC8273 does not specify whether hosts should also be isolated in L2. If hosts are also isolated in L2, all ND issues will be prevented.
- (2) The first-hop router maintains some state information across reboots, e.g. prefix-host binding. The router also maintains some controls over the hosts, e.g. which prefix a host gets.

SAVI, RA-Guard, RA-Guard+ and GRAND solve specific ND issues. They have two characteristics:

- (1) These solutions make no assumptions on L2 isolation or subnet isolation. They just solve the ND issues that they are designed to solve.
- (2) SAVI maintains some state information at the Ethernet switch between the hosts and the first-hop router(s). Such states are learned dynamically from packet snooping, or configured manually. The Ethernet switch uses such state information to control the hosts, e.g., RAs are not allowed from the hosts.

An insight can be learned from these observations. That is, L2 isolation and subnet isolation can be effective in preventing ND issues. But L2 isolation and subnet isolation also have their cost. For example, because hosts are isolated and cannot directly coordinate with each other, the router must have new functionality to coordinate the hosts, e.g. to be the arbitrator when there is an address contention. In order to do so, the router must maintain some state information, e.g. the (IP address, link layer address) binding for each host.

The next section discusses how to apply this insight to future ND deployments.

3. Isolating Hosts in L2 and in Subnet to Simplify ND Deployment

3.1. Applicability of L2 Isolation and Subnet Isolation

This section describes how to isolate hosts in L2 and in subnet, and the advantages and disadvantages of doing so.

Isolating hosts in L2 can be done by: (1) putting a host in a P2P link with the router, or (2) using private VLAN [PVLAN], or split horizon [TR177], or wireless isolation [W-Iso] on multi-access medium to prevent hosts from communicating with each other. These are a matter of device configuration so it can usually be done as long as the devices support these isolating features.

Isolating hosts in L2 is different from setting ND Prefix Information Option (PIO) L-bit=0. For example, in the latter, there can be multiple hosts in a link, and a malicious host can launch on-link security attacks at other hosts.

When hosts are isolated in L2, DAD messages can only reach the router but no other hosts. Therefore, the router must act to prevent hosts from using duplicate GUA or link local address. This functionality is called DAD Proxy [TR177][RFC6957].

The advantages of isolating hosts in L2 include:

- o Prevention of on-link security and upstream multicast issues (from hosts), because hosts cannot reach each other directly.

The disadvantage of isolating hosts in L2 include:

- o The router must support additional functionality: DAD Proxy.
- o Downstream multicast issues, DAD-unreliable issue, On-demand NCE Installation issue and off-link /64 scan issue still remain.
- o All host communication must go through the router. In a high performance environment, the router may become the bottleneck.
- o Services relying on multicast, e.g. Multicast DNS [mDNS], will not work, unless the router can provide multicast proxy.

- o There is additional work (e.g. PVLAN, split horizon or wireless isolation) to isolate hosts in L2 in multi-access media, but this is small amount of work.
- o Many more interfaces or sub-interfaces are likely needed on the router, for example if point-to-point VLANs are used for L2 isolation.

From the analysis above, it is clear that L2 isolation alone is not advantageous. The benefits are small while the costs are high. Therefore, L2 isolation is better used with something else, e.g. subnet isolation or some specially designed solutions like WiND.

The known solutions supporting host isolation in L2 include WiND, and in more special cases (i.e. with both L2 and subnet isolation), MBBv6 and FBBv6.

Host isolation in subnet (i.e. Unique Prefix Per Host, or UPPH) can be done either with [DHCPv6], where the prefix can be of any length supported by DHCPv6, or with SLAAC as specified in RFC8273, where the prefix must be /64 or shorter. If [P2Peth] is progressed, it can provide another solution.

In order to isolate hosts in subnet, the router must be able to assign a unique prefix to each host and keep track of the prefix-host link layer address binding. This will be referred to as "UPPH support" later in this document. This may be achieved by some mechanism that RFC8273 alluded to but did not specify, or by some other mechanisms if DHCPv6 is used [TR177]. Note that with SLAAC/RFC8273, such router implementation exists [8273D], while with DHCPv6, FBBv6-capable BNG is one of such implementations.

The advantages of isolating hosts in subnet are:

- o Downstream Multicast, DAD-unreliable, on-demand NCE installation and off-link /64 scan issues are prevented.
- o It is the only way for the router to do subscriber management for the hosts in a SLAAC environment. Imagine a public Wi-Fi scenario where the mobile UEs only support SLAAC. If the router does not give each UE a unique prefix and keep track of UE-prefix binding, network administrators do not know which IPv6 address corresponds to which UE, because each UE picks its own IID and uses the same prefix. Therefore, network administrators cannot keep track of which UE did what. This would be unacceptable from an operation perspective.

The disadvantages of isolating hosts in subnet are:

- o The routers must support new functionality: "UPPH support".
- o Upstream multicast and on-link security issues can happen, unless the hosts are also isolated in L2
- o Many prefixes will be needed instead of just one. But this disadvantage may not be as severe as it appears. After all, 3GPP has given every mobile UE a /64 [RFC6459], and BBF has given every routed RG a /56 with DHCPv6 PD [TR177]. Outside of MBB, FBB and IoT, not many scenarios have a large number of hosts. Giving each host a /64 prefix may not be a problem.
- o Many more interfaces or sub-interfaces are needed on the router.

The known solutions supporting subnet isolation include RFC8273, and in more special cases (i.e. supporting both L2 isolation and subnet isolation), MBBv6 and FBBv6.

The above analysis reveals that L2 isolation and subnet isolation are synergetic. When they are done together, the advantages are:

- o All ND issues are prevented.
- o It provides a way for the router to do subscriber management for the hosts in a SLAAC environment.
- o DAD Proxy needed for L2 isolation is no longer needed, because with unique prefix per host, GUA cannot be duplicate. For link local address, since each subnet has only one host, there is no possibility of duplication. A duplicate link local address may exist in another subnet but that does not matter. Therefore, there is no need for DAD Proxy to prevent duplicate GUA/LLA addresses.

The disadvantages are:

- o The routers must support new functionality: "UPPH support".
- o Many prefixes will be needed, one per host.
- o All host communication must go through the router. In a high performance environment, the router may become the bottleneck.

- o Services relying on multicast, e.g. mDNS, will not work, unless the router can provide multicast proxy.
- o There is additional work (e.g. PVLAN, split horizon or wireless isolation) to isolate hosts in L2 in multi-access media, but this is small amount of work.
- o Many more interfaces or sub-interfaces are needed on the router.

The known solutions supporting host isolation in L2 and in subnet include RFC8273, MBBv6 and FBBv6.

3.2. Deployment Guidelines

Given the applicability analysis in Section 3.1, network administrators can decide where to use which isolation option. Note that in all the following steps, if DHCPv6 (IA_NA or PD) is supported and desired, use DHCPv6 to assign address prefix, as it may provide prefix length flexibility. Otherwise, use SLAAC as SLAAC is more widely supported.

1. If isolating hosts in both L2 and in subnet is desirable and feasible:

Based on the applicability discussion in Section 3.1, the scenarios here likely have some or all of the following characteristics: (1) It is a public access environment where subscriber management is needed; (2) Many ND issues can happen and will require solutions. This is why it is desirable to prevent as many issues as possible, to simplify the deployment; (3) Neither high performance communication nor multicast service discovery is applicable here.

With suitable "UPPH support", all the ND issues will be prevented. Some possible "UPPH support" solutions are:

- a) If the deployment scenario is MBB or FBB, then MBBv6 or FBBv6 can be used.
 - b) Otherwise, use a RFC8273 implementation (using SLAAC) to realize "UPPH support". Note that this is a special use case of RFC8273 where each host is not only given a unique prefix, but also isolated from other hosts in L2.
2. Otherwise, if isolating hosts in L2 but not in subnet is considered appropriate:

In this scenario, hosts are in different links but in the same subnet. This architecture is called Multi-Link SubNet (MLSN). MLSN has been a controversial scheme in the IETF. The original IPv6 Addressing Architecture [RFC1884] stated that "IPv6 continues the IPv4 model that a subnet is associated with one link. Multiple subnets may be assigned to the same link". This ruled out MLSN in the IPv6 WG (now 6man). However, some other WGs like MANET and 6lo use MLSN in their solutions [RFC7668][RFC8929]. [RFC4903] documented the concerns about MLSN.

For solutions related to ND, only WiND supports MLSN. Therefore, the deployment scenario here is most likely one that WiND is suitable for, e.g. Low-power and Lossy Networks (LLNs). WiND has all functionalities needed for this scenario, including proactive address registration. WiND prevents all the ND issues but is a relatively sophisticated protocol that requires changes to both router and host behaviors from normal ND.

3. Otherwise, if isolating hosts in subnet but not in L2 is considered appropriate:

Based on the applicability discussion in Section 3.1, the scenarios here likely have the following characteristics: (1) Subscriber management is needed. This is likely a public access scenario. (2) Upstream multicast is needed or cannot be avoided (otherwise L2 host isolation would have been selected to prevent more issues), e.g. for service discovery. RFC8273 can be used as the solution here. On-link security is not prevented in this case. Because this is a public access scenario, SAVI/RA-Guard/RA-Guard+ may be needed to address the on-link security issue.

4. Otherwise, no isolation in L2 or subnet is desired or feasible.

The scenarios here likely have the following characteristics: (1) It is a private environment, because subscriber management is not needed. As such, on-link security is not a big concern. (2) Either multicast service is needed, or this is a high performance scenario, because L2 host isolation is not desired. Either way, multicast and DAD-unreliable are not a concern. Normal ND can be used as the solution. Off-link /64 scan and on-demand NCE installation are the two issues left. Off-link /64 scan issue can be handled by rate limiting or unused-address filtering. If this is indeed a high performance environment, e.g. a DC network, GRAND can be used to enable proactive NCE installation.

In short, the guidelines can be summarized as: if many of the ND issues discussed in Section 3 are valid concerns and need to be addressed, then isolate hosts in L2 and in subnet to prevent the issues, and use RFC8273. If the scenario is LLN, use WiND. But if the scenario is a private environment where many ND issues are not real concerns, then just use normal ND, and adds other solutions like GRAND only when needed. Overall, these guidelines will likely result in the simplest ND solution.

4. Impact of L2 Isolation and Subnet Isolation to Other Components of IPv6 First-hop

The guidelines in section 3 will likely simplify ND deployment. But will they complicate other components of IPv6 first-hop? A preliminary impact analysis is done in this section.

IPv6 first-hop consists of:

- o The routers and the hosts, whose requirements & behaviors are defined in [RFC8504];
- o Addressing scheme;
- o The basic protocol suite: ND, SLAAC, DHCPv6, DNS, ICMPv6, MLD v1/v2;
- o Other extended protocols: mDNS.

The impact of host isolation in L2 and in subnet to other components of IPv6 first-hop comes from five parts: (1) the special topology introduced by L2 isolation; (2) the special addressing scheme introduced by subnet isolation (i.e. unique prefix per host); (3) the new functionalities introduced to the router, i.e. "DAD Proxy" for L2 isolation. (4) The increased number of interfaces on the router (5) When WiND is used, hosts must be changed to support proactive address registration etc. This is a high requirement that can be considered as complicating the IPv6 first-hops. But WiND appears to be the only solution in its applicable scenarios like LLNs. When there is no alternative, there is no need to discuss whether the solution unduly complicates other components of IPv6 first-hop. Therefore, WiND will not be further discussed. Because DAD Proxy is only needed in L2 isolation but not in subnet isolation, and this scenario uses WiND which already provides DAD Proxy, DAD Proxy will not be further discussed either.

First, regarding the impact from the special topology:

Isolating hosts should only affect the protocols that rely on multicast, i.e. ND, SLAAC and mDNS, but not the multicast protocols themselves, i.e. MLD v1/v2. As discussed in Sections 2 and 3, the impact to ND and SLAAC are positive. The impact to mDNS is negative. But the guidelines give network administrators the option not to isolate hosts at all. So if network administrators choose to isolate, the benefit must outweigh the cost, e.g. mDNS may not be needed in the deployment scenario.

Second, regarding the impact from the special addressing scheme:

IPv6 first-hop should allow any addressing scheme. Other than using more prefixes, it is not envisioned that unique prefix per host complicates addressing scheme in any way. After all, unique prefix per host is already widely in use in MBBv6 and FBBv6 deployments.

Third, regarding the impact of new router functionality "UPPH support":

The impact of "UPPH support" with RFC8273 using SLAAC has been discussed in [8273D] before RFC8273 became an RFC. The following concerns had been raised:

- . RFC8273 makes the router stateful;
- . How to reclaim unused prefix is not specified;
- . If there are multiple first-hop routers, how the solution works is unspecified: (1) do they assign prefixes to hosts independently? (2) do they need to sync up with each other?
- . Resiliency of SLAAC may be reduced as a result of the increased state at the router, i.e. the prefix-host binding.

Given that RFC8273 became an RFC, the rough consensus has been its benefit outweighs the cost. Because MBBv6 and FBBv6 also support UPPH and are widely deployed, the impact of "UPPH support" appears manageable.

Fourth, regarding the impact of increased number of interfaces on the router: for the consumer market where the number of hosts (i.e. subscribers) is large, this may increase the number of routers needed in the first-hop. However, for the consumer market, MBBv6, FBBv6 and some public Wi-Fi deployments are already using L2 isolation & subnet isolation before the guidelines are proposed. So the guidelines bring no additional burden. Outside of consumer market, a router can generally provide more interfaces than needed. So this increase should not be a problem either.

All things considered, it appears that isolating hosts in L2 and in subnet can simplify ND without unduly complicating other components of IPv6 first-hop.

5. Applying the Guidelines to Real World Scenarios

The guidelines are intended for future IPv6 first-hop deployments. But if we test the guidelines on well-known IPv6 first-hop scenarios to find the suitable ND solution, the result will be as follows:

- o MBB and FBB will end at Step 1.a: isolating hosts in L2 and in subnet, with MBBv6 as the solution with SLAAC and FBBv6 as the solution with DHCPv6, respectively;
- o Public Wi-Fi network will end at Step 1.b: isolating hosts in L2 and in subnet, with RFC8273 as the solution with SLAAC, since the hosts here may be mobile UEs that do not support DHCPv6. Note that in Wi-Fi with the Infrastructure Mode [WiFi-inf], each host (i.e. STA) communicates only with the Access Point (AP). With wireless isolation, every host can only communicate with the AP (and the router if the AP is not a router), not directly with other hosts. This is how L2 isolation can be achieved in this scenario. If L2 isolation is not done, then public Wi-Fi will end at Step 3. SAVI/RA-Guard/RA-Guard+ may be needed to address the on-link security issue.
- o Low-power and Lossy Networks (LLNs) will end at Step 2: isolating hosts in L2 but not in subnet, with WiND as the solution with SLAAC. Note that although WiND did not mandate L2 isolation, WiND works better when hosts are isolated in L2. Otherwise, additional mechanisms may be needed to address on-link security issues.
- o High speed DC Networks will end at Step 4: using normal ND with GRAND without host isolation in L2 or in subnet. SAVI, RA-Guard and RA-Guard+ may not be needed as high physical access security is likely maintained.
- o Common enterprise LANs with mixed Ethernet and Wi-Fi (not DC networks) will end at:
 - o If security is a concern to justify host isolations:
 - . Step 1.b: isolating hosts in L2 and in subnet, with RFC8273 as the solution with SLAAC;.
 - o If security is not a concern to justify host isolations:

- . Step 4: using normal ND with no special host isolation. GRAND and SAVI, RA-Guard and RA-Guard+ may not be needed.
- o [HomeNet] will end at Step 4: using normal ND with no special host isolation. GRAND and SAVI, RA-Guard and RA-Guard+ are not needed.

These results match current practices in reality. This validates the guidelines to some extent.

6. Security Considerations

This document provides guidelines on where to isolate hosts in L2 and in subnet to prevent ND issues, and how to select existing solutions for the remaining issues. When a solution is selected, the security considerations of that solution apply. This document does not introduce any new mechanisms. Therefore, it does not introduce new security issues.

7. IANA Considerations

This document has no request to IANA.

8. References

8.1. Informative References

- [8273D] Discussion on the pros and cons of RFC8273, <https://mailarchive.ietf.org/arch/msg/v6ops/M47lN8lyXaWVcx8nitvkxMfbGNA/>
- [CGA] T. Aura, "Cryptographically Generated Addresses (CGA)" , RFC3972
- [DHCPv6] T. Mrugalski M. Siodelski B. Volz A. Yourtchenko M. Richardson S. Jiang T. Lemon T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415.
- [GRAND] J. Linkova, "Gratuitous Neighbor Discovery: Creating Neighbor Cache Entries on First-Hop Routers", <https://datatracker.ietf.org/doc/html/draft-ietf-6man-grand-07>

- [HomeNet] T. Chown, J. Arkko, A. Brandt, O. Troan, J. Weil, "IPv6 Home Networking Architecture Principles", RFC 7368, DOI 10.17487/RFC7368, October 2014, <<https://www.rfc-editor.org/info/rfc7368>>.
- [mDNS] S. Cheshire, M. Krochmal, "Multicast DNS", RFC 6762.
- [ND] 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>>.
- [PVLAN] https://en.wikipedia.org/wiki/Private_VLAN
- [P2Peth] O. Troan, "IP Point to Point Ethernet Subnet Model", <https://datatracker.ietf.org/doc/draft-troan-6man-p2p-ethernet/>
- [RA-Guard] E. Levy-Abegnoli, G. Van de Velde, C. Popoviciu, J. Mohacsi, "IPv6 Router Advertisement Guard", RFC 6105, DOI 10.17487/RFC6105, February 2011, <<https://www.rfc-editor.org/info/rfc6105>>.
- [RA-Guard+] F. Gont, "Implementation Advice for IPv6 Router Advertisement Guard (RA-Guard)", RFC 7113, DOI 10.17487/RFC7113, February 2014, <<https://www.rfc-editor.org/info/rfc7113>>.
- [RFC1884] R. Hinden, S. Deering, "IP Version 6 Addressing Architecture", RFC 1884.
- [RFC3756] P. Nikander, J. Kempf, E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756.
- [RFC4903] D. Thaler, "Multi-Link Subnet Issues", RFC 4903
- [RFC5072] S. Varada, D. Haskins, E. Allen, "IP Version 6 over PPP", RFC 5072
- [RFC6459] J. Korhonen, J. Soininen, B. Patil, T. Savolainen, G. Bajko, K. Iisakkila, "IPv6 in 3rd Generation Partnership Project (3GPP) Evolved Packet System (EPS)", RFC 6459.
- [RFC6583] I. Gashinsky, J. Jaeggli, W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583.

- [RFC6775] Z. Shelby, S. Chakrabarti, E. Nordmark, C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775.
- [RFC6957] F. Costa, J-M. Combes, X. Pournard, H. Li, "Duplicate Address Detection Proxy", RFC 6957
- [RFC7066] J. Korhonen, J. Arkko, T. Savolainen, S. Krishnan, "IPv6 for Third Generation Partnership Project (3GPP) Cellular Hosts", RFC 7066.
- [RFC7102] JP. Vasseur, "Terms Used in Routing for Low-Power and Lossy Networks", RFC 7102.
- [RFC7668] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, C. Gomez, "IPv6 over BLUETOOTH(R) Low Energy", RFC7668.
- [RFC8273] J. Brzozowski, G. Van de Velde, "Unique IPv6 Prefix per Host", RFC 8273.
- [RFC8504] T. Chown, J. Loughney, T. Winters, "IPv6 Node Requirements", RFC 8504, January 2019, <<https://www.rfc-editor.org/info/rfc8504>>.
- [RFC8505] P. Thubert, E. Nordmark, S. Chakrabarti, C. Perkins, "Registration Extensions for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Neighbor Discovery", RFC 8505.
- [RFC8928] P. Thubert, B. Sarikaya, M. Sethi, R. Struik, "Address-Protected Neighbor Discovery for Low-Power and Lossy Networks", RFC 8928.
- [RFC8929] P. Thubert, C.E. Perkins, E. Levy-Abegnoli, "IPv6 Backbone Router", RFC 8929.
- [RFC9099] E. Vyncke, K. Chittimaneni, M. Kaeo, E. Rey, "Operational Security Considerations for IPv6 Networks", RFC 9099.
- [RIPE IPv6 security] RIPE NCC, "IPv6 Security Training Course", <https://www.ripe.net/support/training/material/ipv6-security/ipv6security-slides.pdf>
- [SAVI] J. Wu, J. Bi, M. Bagnulo, F. Baker, C. Vogt, "Source Address Validation Improvement (SAVI) Framework", RFC 7039

- [SeND] J. Arkko, J. Kempf, B. Zill, P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC3971
- [SLAAC] 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>>.
- [TR177] S. Ooghe, B. Varga, W. Dec, D. Allan, "IPv6 in the context of TR-101", Broadband Forum, TR-177.
- [WiFi-inf] Wi-Fi Infrastructure Mode, <https://www.howtogeek.com/180649/htg-explains-whats-the-difference-between-ad-hoc-and-infrastructure-mode/>
- [W-Iso] Wireless Isolation, <https://www.quora.com/What-is-wireless-isolation>

9. Acknowledgments

The authors would like to thank Eduard Vasilenko, Pascal Thubert, and Ole Troan for the discussion and input.

Authors' Addresses

XiPeng Xiao
Huawei Technologies
Hansaallee 205, 40549 Dusseldorf, Germany

Email: xipengxiao@huawei.com

Eduard Metz
KPN N.V.
Maanplein 55, 2516CK The Hague, The Netherlands

Email: eduard.metz@kpn.com

Gyan Mishra
Verizon Inc.

Email: gyan.s.mishra@verizon.com

