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

M. Smith
N. Kottapalli
R. Bonica
Juniper Networks
F. Gont
SI6 Networks
T. Herbert
Quantonium

May 30, 2020

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

Abstract

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

Status of This Memo

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

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

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

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

Table of Contents

1. Introduction

1.1. Requirements Language

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

2. Terminology

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

3. In-Flight Extension Header Insertion Defined

3.1. In-Flight Insertion

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

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

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

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

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

3.2. In-Flight Removal

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

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

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

3.3. In-Flight Insertion Without Removal

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

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

4. EH Removal Failure Causes

4.1. Implementation Bugs

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

4.2. Partial Node Failure

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

4.3. Operator Configuration Error

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

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

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

5. Single Point of Failure

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

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

6. MUST Remove is Aspirational

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

7. Harm

7.1. Violates RFC8200 and All Of Its Ancestors.

(RFC8200 EH processing text quote)

RFC 2460 and ancestors back to RFC 1883 text quote.

7.2. Ignores Source Address Field Semantics

7.3. Breaks ICMPv6

7.3.1. Breaks PMTUD

7.4. Breaks IPsec

7.5. May Cause Faults in Subsequent Transit Networks

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

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

7.6. Incorrect Destination Host Processing

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

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

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

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

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

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

7.7. Implementation Complexity

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

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

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

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

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

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

7.8. Costly Troubleshooting

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

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

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

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

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

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

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

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

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

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

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

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

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

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

i.e. Postel's law

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

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

9. Solution: Encapsulation

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

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

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

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

9.1. IPv6 Tunnelling

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

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

9.2. MPLS

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

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

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

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

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

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

10. Reducing Tunneling Overhead

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

10.1. ROHC

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

10.2. Skinny IPv6-in-IPv6 Tunneling

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

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

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

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

11. In-Flight Insertion Considered Harmful

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

12. Security Considerations

13. Acknowledgements

Review and comments were provided by YOUR NAME HERE!

This memo was prepared using the xml2rfc tool.

14. Change Log [RFC Editor please remove]

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

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

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

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

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

15. References

15.1. Normative References

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

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

15.2. Informative References

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

[RFC5795] Sandlund, K., Pelletier, G., and L-E. Jonsson, "The RObust Header Compression (ROHC) Framework", RFC 5795, DOI 10.17487/RFC5795, March 2010, <https://www.rfc-editor.org/info/rfc5795>.

[SKINNYV6V6] "Skinny IPv6 in IPv6 Tunnelling", <https://datatracker.ietf.org/doc/draft-smith-skinny-ipv6-in-ipv6-tunnelling/>.

Authors' Addresses

Mark Smith
PO Box 521
Heidelberg, VIC 3084
AU

Email: markzzzsmith@gmail.com

Naveen Kottapalli

Email: naveen.sarma@gmail.com

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

Email: rbonica@juniper.net

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

Email: fgont@si6networks.com

Tom Herbert
Quantonium
Internet Road
Santa Clara, CA
USA

Email: tom@quantonium.net