In-Flight IPv6 Extension Header Insertion Considered Harmful

draft-smith-6man-in-flight-eh-insertion-harmful
IETF-106

Mark Smith
markzzzsmith@gmail.com

Naveen Kottapalli
naveen.sarma@gmail.com

Ron Bonica
rbonica@juniper.net

Fernando Gont
fgont@si6networks.com

Tom Herbert
tom@quantonium.net

RFC 1883, RFC 2460
”Internet Protocol, Version 6 (IPv6) Specification”

“With one exception, **extension headers are not 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.”

RFC 1883, RFC 2460
”Internet Protocol, Version 6 (IPv6) Specification”

“The **exception** referred to in the preceding paragraph is the **Hop-by-Hop Options header**, which carries information that must be **examined** and **processed** by **every node along** a **packet's delivery path**, including the source and destination nodes. 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.”

RFC 8200
”Internet Protocol, Version 6 (IPv6) Specification”

“**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.”

RFC 8200
”Internet Protocol, Version 6 (IPv6) Specification”

“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.”

RFC 8200 Changes Motivations

“must be **examined** and **processed** by **every node along** a **packet's delivery path**”

to

“**may be examined or processed** by **any node** along a **packet's delivery path**”

High speed routers observed ignoring HbH header “must”

RFC 8200 Changes Motivations

“With one exception, **extension headers are not examined or processed** by any node **along a packet's delivery path**, **until** ...”

rephrased to be more explicit:

“**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** ...”

“Insertion of IPv6 Segment Routing Headers in a Controlled Domain”
draft-voyer-6man-extension-header-insertion

(Full) Internet Standard 86
(of 92)

0086 Internet Protocol, Version 6 (IPv6) Specification. S. Deering, R. Hinden. July 2017. (Format: TXT=93658 HTML= bytes) (Obsoletes RFC2460) (Also RFC8200)

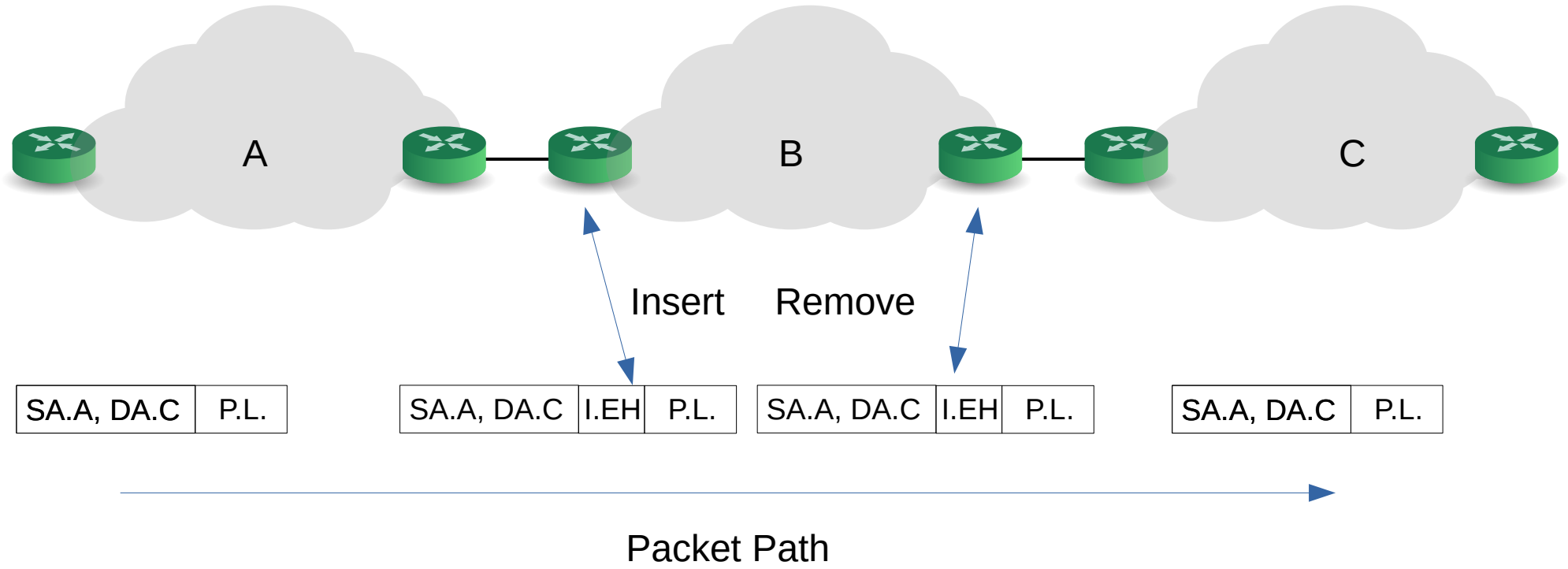
Internet Draft Purpose

“In-Flight IPv6 Extension Header Insertion Considered Harmful”

Record reasons and motivation for RFC1883/RFC2460 and RFC 8200 text.

Describe IPv6 architecturally compliant solution.

In-Flight EH Insertion Defined



Key Observations

Original SA and DA are not modified during insertion/removal.

Packets are being modified without attribution - “anonymous modification”.

Packet size got larger.

Immutable Next Header field got modified.

Motivation?

Has never been specifically stated in `draft-voyer-6man-extension-header-insertion`.

A set of 128 bit Segment Routing Segment IDs (SIDs) is definitely going to add significant overhead.

Try to save overhead somewhere else instead of having smaller SIDs?

Internet Draft Theory

“Since the SRH inserted within an intermediate node MUST be removed when all segments within the SRH have been visited, it is not possible to leak the SRH to the destination outside the source domain.”

`draft-voyer-6man-extension-header-insertion-06, July 2019`

Actual Network Practice

Implementation Bugs

Partial Device Failures

Device Misconfiguration by Network Operator

“Since the SRH inserted within an intermediate node MUST be removed when all segments within the SRH have been visited, ...”

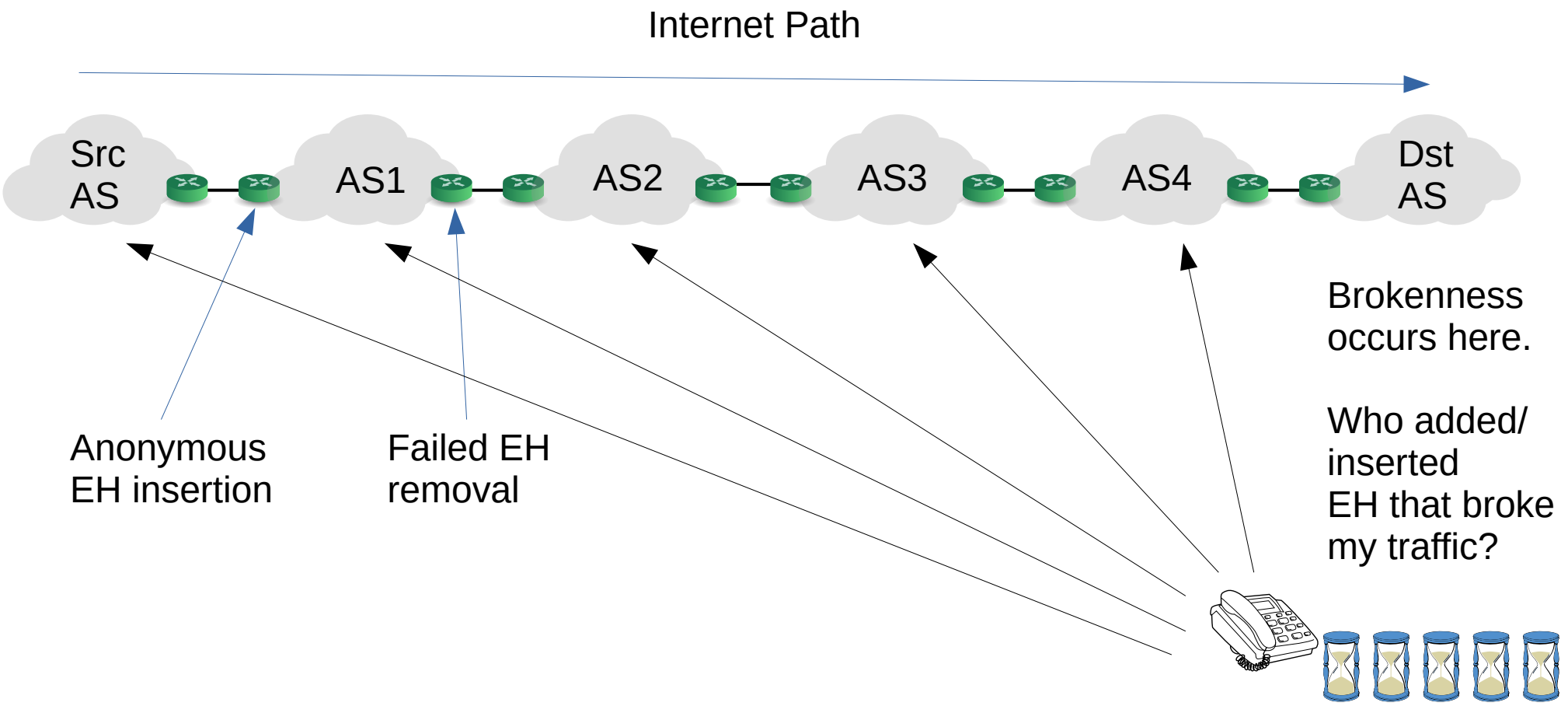
This **MUST** is an **aspiration**, not an assurance.

Single Point of Failure

The boundary of the EH insertion domain is likely to be defined by a single level of boundary devices.

That means the boundary is possible Single Point of Failure.

Failed EH Removal Scenario



Consequences and Impacts

Ignores Source Address Field Semantics

Once the EH is inserted, the packet’s unchanged Source Address field is now not correctly identifying all of the sources of the packet’s contents.

Two devices are now responsible for the contents of the packet.

One is anonymous.

Mechanisms and protocols that rely on using the Source Address, if triggered by the inserted EH, may fail.

Breaks ICMPv6

ICMPv6 sends messages back to trigger packet's Source Address.

ICMPv6 message triggered by inserted EH will not be sent to the EH insertion device, as it is not identified in the packet's Source Address field.

Breaks ICMPv6 PMTUD

Packet size increase due to inserted EH could trigger PMTUD.

ICMPv6 Packet Too Big not sent to EH insertion device, as packet’s SA is not EH insertion device.

Breaks IPsec

If the inserted EH fails to be removed, it will look like unauthorised packet modification to IPsec.

Fault in Subsequent Transit Network

If an inserted EH fails to be removed, and the packet travels through a subsequent transit network that is also inserting EHs, the non-removed EH may interfere.

Incorrect Destination Host Processing

Non-removed EH could cause packet to be discarded when it shouldn’t be e.g. unknown EH skipped over to next EH or UDP/TCP etc. header.

Non-removed EH could cause packet to be processed when it should be discarded.

Handling of unknown EHs is described in high order two bits of EH type. Non-removed EH type high order bits could be incorrect for packet’s payload and use.

Implementation Complexity

An EH insertion domain egress device will have to look into each and every packet’s EH chain to see if there is an EH to remove.

This is more complex than using simple packet Destination Address value match to select either further simple forwarding or deeper EH processing.

Postel’s Law or The Robustness Principle

"Be conservative in what you send, be liberal in what you accept"

Inserted EHs are not expected by RFC 8200 compliant receivers, as RFC 8200 prohibits them.

Purposely sending them is not being “conservative in what you send”.

Solution: Encapsulation

Encapsulation is the tried, tested and proven method used to add new information to existing PDUs in the Internet architecture.

e.g. TCP encaps application PDU, IP encaps TCP, Link-Layer encaps IP.

Adding new EH via IPv6 tunnel encapsulation

RFC 2473, “Generic Packet Tunneling in IPv6 Specification.”

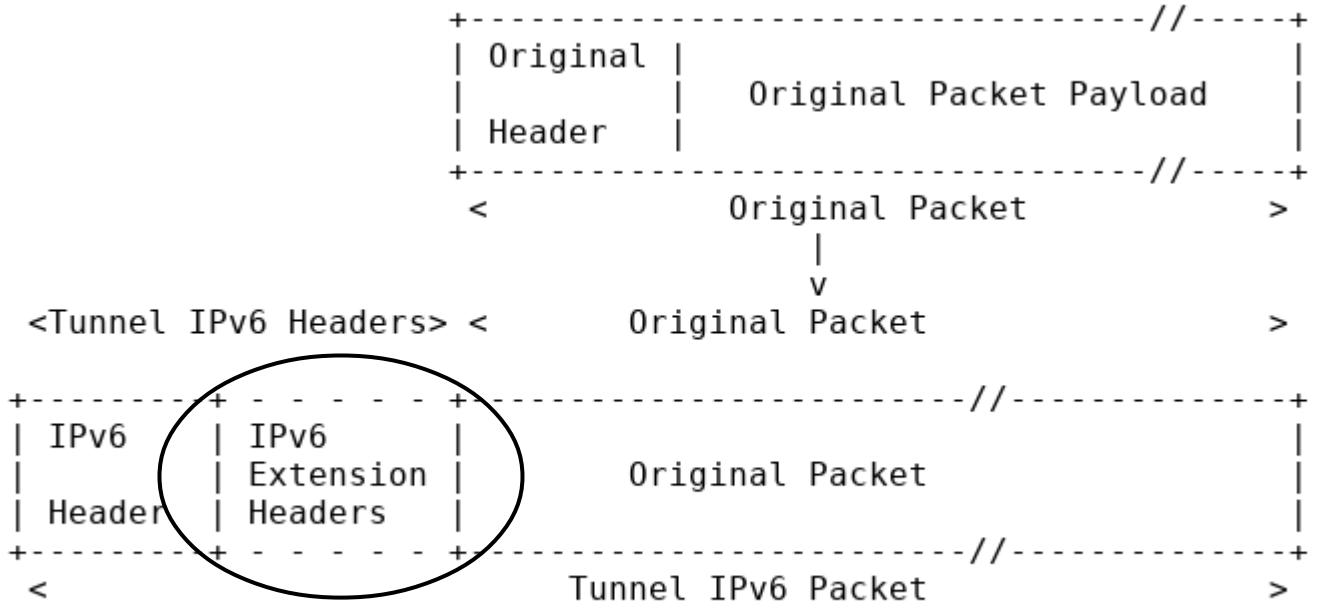


Fig.3 Encapsulating a Packet

RFC 2473 provides

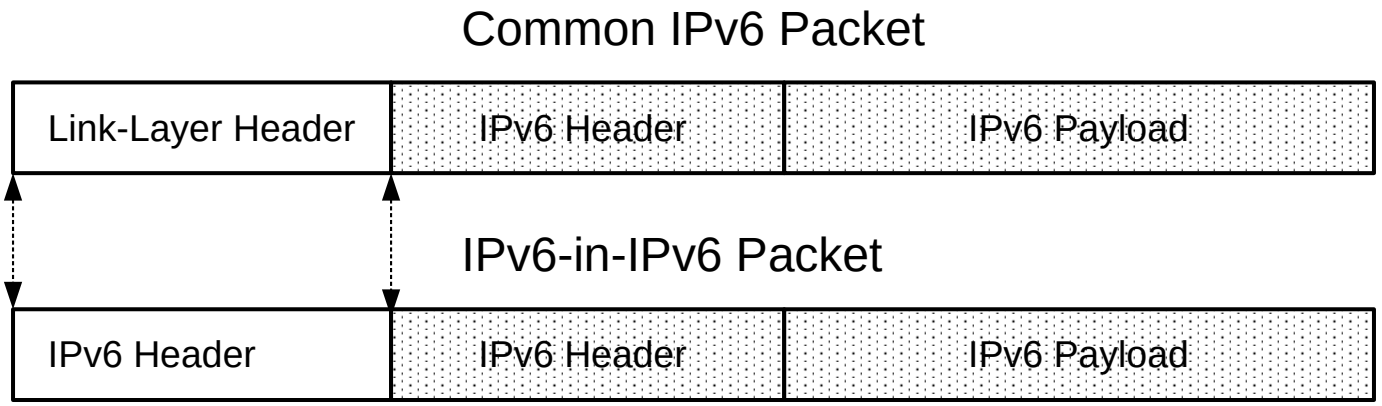
EH addition attribution, via outer IPv6 packet Source Address

Correctly working PMTUD as tunnel end-point that increased packet size is in outer SA.

“IP Tunnels in the Internet Architecture”

draft-ietf-intarea-tunnels

Reducing Tunnelling Overhead



Outer IPv6 header is a Link-Layer header in the context of the inner IPv6 packet.

Coincidence that the Link-Layer header and the IPv6 packet's header have the same structure and field semantics.

Use suitable Link-Layer compression on link-layer "payload" inner IPv6 packet while in flight over tunnel.

Link-Layer payload inner IPv6 packet compression

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]

Link-Layer payload inner IPv6 packet compression

Skinny IPv6-in-IPv6 Tunnelling (draft-smith-skinny-ipv6-in-ipv6-tunnelling):

- Leverages common inner and outer IPv6 header field semantics to carry many inner header field values in outer header.

- Uses /64s to identify tunnel endpoints, allowing outer header address IID parts to carry inner packet IID field values.

Thoughts?

Questions?