Attribution Option for Extension Header Insertion
draft-herbert-6man-eh-attrib-00

Abstract

This document defines an IPv6 Hop-by-Hop Option that provides attribution for IPv6 extension headers inserted by intermediate nodes in the delivery path of a packet. The purpose of this option is twofold: first it identifies the extension headers that have been inserted, secondly it attributes the inserted extension headers to the node responsible for inserting them.

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Copyright and License 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. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.

Table of Contents

1 Introduction ............................................. 3
  1.1 Motivation for extension header insertion .......... 3
  1.2 Problems with extension header insertion .......... 4
  1.3 Inserting Hop-by-Hop Options ...................... 5
  1.4 Scope ............................................ 5
  1.5 Requirements Language ............................. 5
2 Attribution Option for extension header insertion ........ 6
  2.1 Format ........................................... 6
     2.1.1 Attribution Option with short identifier ....... 7
     2.1.2 Attribution Option with IPv6 address identifier .... 7
  2.2 Model ............................................ 8
3 Operation .............................................. 9
  3.1 Insertion .......................................... 9
     3.1.1 Insertion of Hop-by-Hop options ............... 9
     3.1.2 Insertion of extension headers ................ 10
     3.1.3 Errors during insertion ....................... 11
  3.2 Removal ........................................... 11
     3.2.1 Removal of Hop-by-Hop options ................. 11
     3.2.2 Removal of extension headers .................. 12
     3.2.3 Errors during removal ......................... 12
  3.3 Domain edge filtering ............................. 13
  3.4 ICMP processing ................................... 13
4 Security Considerations ................................ 14
5 IANA Considerations ................................... 15
6 References ............................................ 15
  6.1 Normative References ............................... 15
  6.2 Informative References ............................. 15
Author’s Address ...................................... 16
1 Introduction

Extension header insertion has been proposed as a mechanism to annotate packets for transit across controlled, or limited, domains ([SRHINS], [IOAM]). Presumably, before a packet egresses the controlled domain, any inserted extension headers need to be removed.

Extension header insertion and removal at intermediate nodes is currently prohibited by [RFC8200], and [INSHARM] provides the rationale for why extension header insertion is harmful and thus is prohibited.

This document addresses the main problem of extension header insertion which is the loss of attribution to the source of packet contents. A Hop-by-Hop option is defined to provide proper attribution. There are two salient aspects to this:

* The Hop-by-Hop Option unambiguously identifies what extension headers were inserted by intermediate nodes.
* The Hop-by-Hop Option includes an identification of the intermediate node that inserted extension headers in a packet.

1.1 Motivation for extension header insertion

IP in IP encapsulation has been proposed as an alternative to extension header insertion. While encapsulation may be functionally equivalent to header insertion, there are merits to header insertion:

* Extension header insertion can result in less bytes of overhead than encapsulation.
* The proper destination address to set in the encapsulating IP header may be unknown. For instance, a node might insert an extension header into an existing packet with the intent that the packet is routed based on the original destination to an egress node of the domain that will remove the inserted headers.
* Packets for a flow may require consistent routing whether or not extension headers are inserted. In particular, to route flows consistently in Equal Cost MultiPath (ECMP), the hash computed for ECMP should be the same for all packets of the flow. Unlike IP encapsulation, extension header insertion doesn’t affect the fields used in ECMP hash calculation (the source address, destination address, flow label, and transport layer ports), so the ECMP hash calculation consistently derives the same value for all packets of a flow with or without inserted extension headers.
1.2 Problems with extension header insertion

Extension header insertion and removal at intermediate nodes is prohibited by [RFC8200]:

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 rationale for this prohibition is articulated in [INSHARM]. A summary of cited problems with extension header insertion are:

* It breaks the attribution model of IP in that the contents of a packet are no longer attributable to the source node identified by the source address of a packet (exceptions include data that a source node sets in a packet that is explicitly specified to be modifiable).

* It breaks PMTU discovery since extension header insertion increases the packet size in flight.

* It breaks ICMP since inserted extension headers may themselves cause ICMP errors that are sent to the source address. If the source node receives such an ICMP error it cannot take any action to resolve the error since it’s not the source of the data that caused the error.

* Use of extension header insertion is generally assumed to be confined to a controlled domain where the domain is a walled garden such that inserted extension headers are always removed before packets would exit a domain. It is conceivable that configuration or implementation errors may allow packets with inserted extension headers to leak out of the controlled domain.

* It potentially violates the recommendation in [RFC8200] that extension headers appear only once in a packet as well as the recommended ordering for extension headers.

This proposal primarily addresses the attribution of packet contents problem. A solution to the attribution problem addresses or at least can mitigate some of the other problems with extension header insertion.
1.3 Inserting Hop-by-Hop Options

Inserting Hop-by-Hop Options is considered a special case of extension header insertion since per [RFC8200] there can only be one Hop-by-Hop Options extension header in a packet, and if present it must be the first extension header after the IPv6 header. If Hop-by-Hop Options are to be inserted into a packet with an existing Hop-by-Hop Options extension header, the options must be inserted into the options list for the existing extension header.

1.4 Scope

This document describes a mechanism for providing attribution in extension header insertion and procedures to use the mechanism. With the exception of inserting Hop-by-Hop Options, requirements and semantics for inserting specific types of extension headers are out of scope. Similar, security aspects, including potential leakage of inserted headers outside of a controlled domain, is not in scope.

1.5 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].
2 Attribution Option for extension header insertion

2.1 Format

The format of the Hop-by-Hop Attribution Option is:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  | Opt Data Len  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
+-------+-------+-----------------------------------------------+
|   Num_opts    |    Num_EHs    |                               |
+-------+-------+-----------------------------------------------+
| +-------------------------------+ |
| ~                        Identification | ~ |
| +-------------------------------+ |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields are:

* Option Type: value is TBA. The first three bits of the option type should be 000 to indicate that the option is to be skipped over when processed as an unknown option and that the option data is unmodifiable.

* Opt Data Len: data length for the option. The minimal data length is two. If the data length equals twenty-two then the Identification is an IPv6 address (see section 2.1.2).

* Num_opts: indicates the number of non-padding Hop-by-Hop options following the Attribution Option that are attributed as being inserted. If the value of this field is 255 then this indicates that the Hop-by-Hop Options extension header was itself inserted and all the following options are attributed to the insertion.

* Num_EHs: indicates the number of extension headers following the Hop-by-Hop extension header that have been inserted.

* Identification: indicates the source node responsible for the inserted extension headers. This can either be the IPv6 address of the responsible node or a local identifier value that is interpreted by the local network domain (see examples below).

Alignment of the Attribution Option is 4n+2. When the Attribution Option is used, it MUST be set as the first option in the Hop-by-Hop options; hence, there is no need for padding to align the Attribution Option. There may be multiple instances of the Attribution Option in a packet as described below.
2.1.1 Attribution Option with short identifier

Below is the short format of the Attribution Option.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+------------------------------------------------+
|      Type     |        4      |
+------------------------------------------------+
|   Num opts   |    Num EHs   |            Short_ID           |
+------------------------------------------------------------------+
```

Short_ID is interpreted locally. For instance, it may be used as an index to a table to map a value to an IPv6 address.

2.1.2 Attribution Option with IPv6 address identifier

Below is the format of the Attribution Option that contains an IPv6 address for attribution of the inserted extension headers.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+------------------------------------------------+
|      Type     |       22      |
+------------------------------------------------+
|   Num opts   |    Num EH    |            Local_ID           |
+------------------------------------------------------------------+
|                                                               |
|                                                               |
|                         IPv6 address                           |
|                                                               |
|                                                               |
```

Local_ID contains supplemental identification that is interpreted by the local network. This MAY be the AS of network corresponding to the node identified by the IPv6 address.
2.2 Model

The Attribution Option indicates both inserted Hop-by-Hop Options and inserted extension headers. Note that at most one Hop-by-Hop Extension header MUST be in a packet, and if the Hop-by-Hop Extension header is present it MUST be the first extension header following the IPv6 header. If Hop-by-Hop Options are to be inserted into a packet with an existing Hop-by-Hop extension header then the options MUST be inserted into the existing header.

Multiple extension headers insertions may occur during the lifetime of a packet. Insertions are treated as a stack. Extension headers MUST be inserted before any existing extension headers, and Hop-by-Hop options MUST be inserted before any existing Hop-by-Hop options. Inserted extension headers and inserted Hop-by-Hop options MUST be removed in the reverse order of insertion (i.e. inserted headers are "popped" to remove them).

The logical structure of an IPv6 packet with inserted extension headers and IP options, and the relationship between Attribution Options and inserted extension headers and Hop-by-Hop options, is shown below:

```
+------------------+
|  IPv6 header    |
+------------------+
|  Hop-by-Hop EH  |
+------------------+
     | Attribution Opt |
+------------------+
     | Inserted options |
+------------------+
     | ...             |
+------------------+
     | Attribution Opt |
+------------------+
     | Inserted options |
+------------------+
     | Original options |
+------------------+
     | Inserted EHs    |
+------------------+
     | Original EHs    |
+------------------+
```

...
3  Operation

This section describes operations for extension header insertion and removal at intermediate nodes.

3.1 Insertion

An extension header chain MAY be inserted into a packet. The packet’s size will increase, and if Hop-by-Hop Options are already present then the size of the Hop-by-Hop extension header will increase.

3.1.1 Insertion of Hop-by-Hop options

Hop-by-Hop options, including the attribution option, may be inserted into a packet.

Procedure is:

* If a Hop-by-Hop extension header does not exist in the packet:

1) Create a Hop-by-Hop extension header. This contains the Attribution Option, followed by any Hop-by-Hop options being inserted. Num_opts is set to 255 to indicate that the Hop-by-Hop extension header was inserted. Num_EHs is set to the number of extension headers being inserted not including the Hop-by-Hop extension header.

2) The new Hop-by-Hop extension header is prepended to the chain of extension headers being inserted. The nexthdr field of the Hop-by-Hop option is set to the type of the first extension header being inserted (after the Hop-by-Hop extension header).

3) The resulting extension header chain is inserted into the packet following procedures in section 3.1.2.

* Else, if Hop-by-Hop Options are already present then insert new Hop-by-Hop options into the existing header:

1) Make first Hop-by-Hop option to be the Attribution Option. Num_opts is set to the number of non-padding Hop-by-Hop options being inserted not including the Attribution Option. Num_EHs is set to the number of extension headers being inserted.

2) Following the Attribution Option, set any other options being inserted. Include padding before the options as necessary to enforce any alignment requirements.
3) Following the last inserted option, add padding such that
the alignment of the first byte after the last padding byte
is 8n+2 from the start of the Hop-by-Hop extension header.
This is necessary to preserve alignment requirements of
existing options. The amount of padding needed is:

    7 - (offset_last_inserted_byte % 8)

4) Following the last inserted option and inserted padding,
copy the original options from the packet.

5) Set length of the Hop-by-Hop extension header to reflect the
length with the inserted options and any inserted padding.

3.1.2 Insertion of extension headers

This section describes procedures for inserting extension headers
into a packet.

* If Hop-by-Hop options already exists in a packet, then extension
headers (as a chain) are inserted after the Hop-by-Hop extension
header. Note that the Attribution Option must be inserted in the
existing Hop-by-Hop extension header following the procedure in
section 3.1.1.

Procedure is:

1) The nexthdr field of the last inserted header is set to the
original nexthdr value in the Hop-by-Hop Options extension
header.

2) The nexthdr field of the Hop-by-Hop Options extension header
is set to the type of the first header being inserted.

3) Extension headers are inserted into packet following the
Hop-by-Hop extension header.

* Else, if Hop-by-Hop options don’t exist in a packet, then
extension headers (as a chain) are inserted after the IPv6
header. Note that the inserted extension headers MUST include a
Hop-by-Hop extension header containing the Attribution Option
and the Hop-by-Hop extension is the first extension header being
inserted (see section 3.1.1).

Procedure is:

1) The nexthdr field of the last inserted header is set to the
original nexthdr value in the IPv6 header.
2) The nexthdr field of the IPv6 header is set to 0 (since the first inserted header must be Hop-by-Hop Options).

3) Extension headers are inserted into packet following the IPv6 header.

If more than one extension header is inserted then the relative ordered amongst the inserted extension headers SHOULD follow the recommended extension headers ordering in [RFC8200]).

3.1.3 Errors during insertion

Errors may occur in the process of inserting extension headers in a packet. Error conditions would include the resultant packet size exceeding MTU, and the size of Hop-by-Hop Options extension header exceeding 1024 bytes (the maximum size of the Hop-by-Hop Options extension header).

If an error occurs during insertion then the node performing insertion MUST take an appropriate behavior per some configuration. The packet MAY be discarded or the unmodified packet MAY be forwarded. An error SHOULD be logged.

3.2 Removal

The top level inserted extension headers and Hop-by-Hop options, referred to by the Attribution Option, which is precisely the first option in the Hop-by-Hop Options for a packet, may be removed by an intermediate node.

3.2.1 Removal of Hop-by-Hop options

The procedure is:

* If Num_opts equals 255 then the Hop-by-Hop extension header is removed following procedures in section 3.2.2.

* If Num_opts is less than 255, then the inserted Hop-by-Hop options must be removed from the existing header:

1) Locate the last inserted option. This done by the scanning non-padding options after the Attribution Option for the count in Num_opts.

2) Compute the amount of padding that was inserted. The amount of padding that should have been inserted is:

\[ 7 - (\text{offset\_last\_option\_byte} \mod 8) \]
where offset_last_byte is the offset of the last byte of the last inserted option located in step #1.

3) Remove the bytes in the packet from first byte of the Hop-by-Hop Options data (first byte of the Attribution option) through the last byte of inserted padding as computed in step #2.

4) set the length of the Hop-by-Hop Options extension header to account for the removed bytes.

3.2.2 Removal of extension headers

The procedure is:

* If Num_opts equal 255 then the Hop-by-Hop extension header is removed along with any other inserted headers:

1) Locate the last inserted extension header. This done by the scanning extension headers after the Hop-by-Hop Options extension header for the count in Num_EHs.

2) Set nexthdr of the IPv6 header the nexthdr field value of the last extension header being removed.

3) Remove the Hop-by-Hop Options extension header and any additional inserted extension headers from packet.

* Else, if Num_opts less than 255 then extension headers after the Hop-by-Hop extension header are removed.

1) Locate the last inserted extension header. This done by the scanning extension headers after the Hop-by-Hop Options extension header for the count in Num_EHs. If Num_EHs is zero, then no extension headers are removed.

2) Set nexthdr of the Hop-by-Hop Option extension header to the nexthdr field value of the last extension header being removed.

3) Remove any inserted extension headers after the Hop-by-Hop Options extension header.

3.2.3 Errors during removal

A node performing extension header removal MUST validate packet contents.
The following attributes MUST be validated before removal:

* If Num_opts is not equal to 255 then number of non-padding options following Attribution Option MUST be greater than or equal to Num_opts.

* Necessary padding after the last inserted Hop-by-Hop option MUST be present. The amount of padding MUST be equal to the expected amount.

* The Num_opts options following the Attribution Option MUST NOT contain another Attribution Option.

* The number of extension headers following the Hop-by-Hop Options extension header MUST be greater than or equal to Num_EHs.

If any of the above validations fail, or an error is otherwise encountered in the removal process, then the processing node MUST take action. The packet SHOULD be discarded and error message SHOULD be logged.

3.3 Domain edge filtering

Filtering packets with inserted extension headers or Hop-by-Hop options is straightforward: a packet contains inserted extension headers if it contains Hop-by-Hop options where the first option is the Attribution Option. Note that the Attribution Option MUST be at fixed location in the IPv6 packet if it is present, hence it should be amenable to match such packets with inserted extension headers in hardware implementations.

3.4 ICMP processing

At described in [INSHARM], it is possible for a source node to receive ICMP errors caused by inserted headers, thus the source node has no recourse to address the error.

This section proposes some ways to apply the Attribution Option to mitigate the ICMP breakage for extension header insertion:

* ICMP errors can be filtered by nodes in the network before reaching a source node outside of the domain (at the domain edge for instance). The packet headers in the ICMP data will include the Hop-by-Hop Options extension header containing the Attribution Option at a fixed offset in the data. The filtering node MAY analyze the error to determine if it was caused by the inserted headers:
- If the error was caused by inserted extension headers, then
  the node SHOULD take appropriate actions (minimally it
  SHOULD log the error). The filtering node SHOULD not forward
  the ICMP error to the source.

- If the error was not caused by inserted headers, the
  filtering node MAY create an new ICMP error with the data
  packet that would be reflect the packet contents prior to
  extension header insertion (i.e. attempt set the packet in
  ICMP to be that which the source would have sent). This is
  done by removing the inserted extension headers of the
  packet in the ICMP data, and adjusting the Pointer field in
  an ICMP error if necessary. The revised ICMP error can then
  be forwarded to the source.

* If ICMP errors are not filtered and the source node receives an
  ICMP error for a packet containing inserted extension headers:

- If the source node is a legacy implementation that does not
  understand the Attribution Option then it will attempt to
  process the error under the assumption that it was the
  source of the packet and the data that caused the error. If
  the node logs the contents of the ICMP error, which should
  be common, then external out-of-band analysis can be done by
  network administrators to troubleshoot the ICMP errors and
  identify culprit if the error was caused by inserted
  extension headers.

- If the source node understands the Attribution Option then
  it can perform more analysis. The node MAY attempt to
  ascertain if the error was caused by inserted headers or
  not, and if not it can then attempt to fix the problem with
  the assumption the it was responsible for the data in error.

4 Security Considerations

The Attribution Option does not in itself introduce any new security
considerations. The security of containing inserted extension headers
within a controlled domain is out of scope for this document.
5 IANA Considerations

IANA is requested to assigned the following Hop-By-Hop option:

<table>
<thead>
<tr>
<th>Hex Value</th>
<th>Binary value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD</td>
<td>00 0 TBD</td>
<td>Attribution Option</td>
<td>This document</td>
</tr>
</tbody>
</table>

6 References

6.1 Normative References


6.2 Informative References


[INSHARM] Smith, M., Kottapalli, N., Bonica, R., Gont, F., and Herbert, T., "In-Flight IPv6 Extension Header Insertion Considered Harmful", draft-smith-6man-in-flight-eh-insertion-harmful-01, October 2018

Author’s Address

Tom Herbert
Intel
San Jose, CA
USA

Email: tom@herbertland.com