Network Working Group
Internet-Draft
Updates: RFC8200 (if approved)
Intended status: Standards Track
Expires: 5 July 2026

T. Herbert
XDPnet

1 January 2026

Infight Removal of IPv6 Hop-by-Hop and Routing Headers
draft-herbert-eh-inflight-removal-06

Abstract

This document specifies a method to allow intermediate nodes to remove IPv6 Hop-by-Hop Options headers and Routing headers from packets inflight. The goal is to reduce the probability of packets being dropped because they contain these extension headers without impacting functionality. An additional goal is to limit visibility of information in extension headers to those nodes that need to process the 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 5 July 2026.

Copyright Notice

Table of Contents

1. Introduction

This document specifies a protocol to allow intermediate nodes to remove IPv6 Hop-by-Hop Options headers or Routing headers from packets inflight.

Current data suggests that there are very high drop rates for packets with Hop-by-Hop Options sent over the Internet. The goal of this protocol is to reduce the probability of the packet being dropped by a downstream node without reducing functionality, thereby improving the viability and usability of Hop-by-Hop Options.

A second goal is to allow removal of Hop-by-Hop Options headers and Routing headers when packets egress a limited domain in order to limit exposure of data to only those nodes that legitimately need to process it. This an alternative to discarding packets at an egress router of a limited domain and facilitates the use Hop-by-Hop Options or Routing headers for the portion of a packet’s delivery path within a source’s limited domain.

This specification is limited only to removal of the whole Hop-by-Hop Options header or Routing header. It does not set requirements for removing individual Hop-by-Hop options in a Hop-by-Hop Options header, nor does it specify any method for routers to insert a Hop-by-Hop Options header, options in a Hop-by-Hop header, or a Routing header in packets.

If approved, this document updates [RFC8200].

1.1. Terminology

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. Motivation

This section provides the motivations for allowing intermediate nodes to remove Hop-by-Hop Options or Routing headers from packets.

2.1. Hop-by-Hop Options drop rate

The latest measurements indicate that packets with Hop-by-Hop Options have high drop rates when sent on the Internet. From [APNIC-EH]:

> The HBH option was experiencing an average packet drop rate of 99.5% across all HBH option sizes

The reported drops rates for Hop-by-Hop Options are greater than that of packets with Destination Options headers or Fragment headers. An explanation for this difference is that Hop-by-Hop Options are intended to be processed by intermediate nodes in a network, and hence a network operator may be motivated to drop packets with Hop-by-Hop options entering their network from untrusted sources to protect their infrastructure. This is mentioned in [RFC9098] as a reason that packets containing IPv6 Hop-by-Hop Options are dropped:

> The Hop-by-Hop Options header has been particularly challenging since, in most circumstances, the corresponding packet is punted to the control plane for processing. As a result, many operators drop IPv6 packets containing this extension header [RFC7872]. [RFC6192] provides advice regarding protection of a router's control plane.

Given that there doesn't seem to be an easy fix to make Hop-by-Hop Options work over the Internet, the commonly proposed alternative is to limit use of Hop-by-Hop Options to limited domains [RFC8799]. It should be noted that Hop-by-Hop Options are only useful when at least some of nodes in the path process them, so a network operator would likely only deploy routers that process Hop-by-Hop Options if they perceived there is some benefit. If a network supports value add services that use Hop-by-Hop Options, it stands to reason that packets with Hop-by-Hop Options wouldn't be dropped while their within the limited domain of the network operator.

If a destination is not within the limited domain, a source host might still desire to use Hop-by-Hop Options to affect packet processing in the part of the path that is within the source's limited domain. To facilitate this, a packet might be created with Hop-by-Hop Options, the packet traverses the local network to an egress router, and at the egress router the Hop-by-Hop Options header is removed from the packet and the packet is forwarded outside of the limited domain without Hop-by-Hop Options.

2.2. Routing domain firewalls

When a host sends a packet with a Routing header, for example a Segment Routing header [RFC8754], the intermediate destinations are considered to be in the same limited domain. For example, in Segment Routing all of the intermediate destinations in a Segment Routing header must be in the same segment routing domain.

The final destination of a Routing header might not be in the routing domain. It may, in fact, be outside of the limited domain. An example use case of this would be if a Routing header was used to route a packet to an egress router of the domain. The egress router

would be the penultimate destination in the segment list such that the Segments Left field is set to zero and all downstream nodes would ignore the Routing header. In this case, a packet can forwarded beyond the limited domain without a routing header and no impact on behavior.

2.3. Removing extension headers

2.3.1. Removal by egress routers

To contain the Hop-by-Hop Options and Routing header to their limited domain, this specification proposes that egress routers may remove the extension headers from packets before forwarding them beyond the limited domain.

Hop-by-Hop Options would be removed by an egress router in order to increase the likelihood that packets sent with Hop-by-Hop Options are successfully delivered. The assumption is that the Hop-by-Hop Options are typically not useful beyond the limited domain.

A Routing header would be removed at an egress router when its being used to route a packet from a host beyond the limited domain. When the penultimate destination processes the routing header, it sets the final Destination Address and Segments Left to zero, so at that point the Routing header can be removed without impacting downstream processing of the packet.

2.3.2. Removal by ingress routers

Hop-by-Hop Options could be removed from packets by ingress routers as an alternative to the current common practice of dropping the packets with Hop-by-Hop Options. In this case, the network operator doesn’t process Hop-by-Hop Options, or it only processes Hop-by-Hop Options from source hosts in the local domain that it trusts. Removing Hop-by-Hop Options instead of dropping them allows packets to be delivered without loss of functionality or risk to the network infrastructure. Note that removing Hop-by-Hop Options has the same operational effect in routers as ignoring them which is permitted by [RFC8754] and [RFC9673].

2.4. Alternatives to Extension Header removal

This section discusses some of the alternatives to extension header removal that have been proposed.

2.4.1. Host routing

It is conceivable that a host network stack could maintain routes to destinations or networks with an indication that the destination is within the limited domain. So when a packet is being created, the routing table could be consulted to determine if it's safe to send packets with Hop-by-Hop Options to the destination.

The main drawback of this approach is that it requires significant changes to the host networking stack: in the routing infrastructure, the APIs presented to the application to set Hop-by-Hop Options, and probably applications themselves need changes. Additionally, in all but trivial network topologies it won't be obvious just given an address whether the destination is in the same limited domain as the host. In some simpler topologies, it might be possible to configure hosts with all the network prefixes that belong to the limited domain, however for a more complex topology hosts may need to participate in a routing protocol or a discovery protocol with the network.

2.4.2. Probing

Capabilities probing has been successfully employed in other contexts such as "Happy Eyeballs" for IPv6. Probing could similarly be used to determine the viability of Hop-by-Hop Options to a destination. In this case, a host could probe each destination to determine if Hop-by-Hop Options are viable. The advantage of this method is that it requires no special assistance from the network.

The main drawback of this approach is the complexity in the host stack and applications. Probing assumes bidirectional communications, state needs to be maintained for each destination or flow, procedures need to be specified for probing, and considerations need to be made for route changes that might affect the disposition of packets with Hop-by-Hop Options in the network. Additionally, the implementation for probing would be different for UDP and TCP: probing in the UDP case would most likely need support in the application and userspace libraries, probing for TCP would likely need to be supported in the Operating System kernel.

2.4.3. IPinIP Encapsulation from source

In order to use Hop-by-Hop Options in the part of the path in a limited domain, a source host may encapsulate the packet in an IPinIP encapsulation [RFC2473]. The outer IPv6 header would contain the Hop-by-Hop Options header and the destination would be the address of an egress router for the limited domain. At the egress router, the packet would be decapsulated and the packet can be forwarded without Hop-by-Hop Options.

The main problem to this approach is that the sending host would need to know the correct Destination Address to set in the encapsulating header; that is, the host would need to know the address of the correct egress router for the packet. That information is not normally available to hosts and might not even be available to intermediate nodes including the first hop router. In a complex, multi-homed, network topology that might support mobile hosts, the only way to determine the current egress router for a packet may be to actually route through the network to the external destination address.

If the network did maintain the association between destinations and the egress router then conceptually it could share that information with hosts using a routing protocol or discovery protocol. This information could be saved in an augmented routing table on the host similar to that described in Section 2.4.1.

If the network provides the addresses of egress routers that is potentially divulging network topology information to the hosts and could be considered a security risk.

Conceivably, a host could be configured with a single anycast address to be used as Destination Address of the egress router when encapsulating. If the host routing table includes limited domain information, as described in Section 2.4.1, then this would be sufficient to route packets to an egress router. In this case though, the anycast address represents a default router which might not be the same one had the packet been routed based on its final destination-- this could be suboptimal routing or cause out-of-order packets if not all packets of a flow are encapsulated.

This solution is complex from a host implementation point of view. An IPinIP encapsulation adds at least forty bytes of overhead to the packet, which reduces the effective MTU for the application and requires special end host processing that may be prohibitive on low end devices. Even if an anycast address is configured, a host stack will need to maintain routing information to determine which packets need to be encapsulated. Furthermore, setting the Hop-by-Hop Options

is done by the application without regard to whether the packet is being encapsulated. When a packet is sent and it needs to encapsulated, the host stack will need to remove the Hop-by-Hop Options from the original packet and set them in the encapsulating IPv6 headers.

2.4.4. IPinIP Encapsulation from egress router

Another solution using IPinIP encapsulation would be for an egress router to encapsulate a packet containing Hop-by-Hop Options in IPinIP. The outer IPv6 header contains no Hop-by-Hop Options and the inner IPv6 header contains the options. The Destination Address of the outer and inner IP headers are the same.

This solution is not robust since the encapsulation increases packet size and reduces the Path MTU seen by the sender which can cause systematic packet drops. For example, suppose a host sends a packet with minimum MTU size of 1,280, and an egress router encapsulates the packet so that its length increases to 1,320 bytes. If a downstream router has link MTU of 1,280 then the packet will be dropped since its length exceeds the link MTU. Since the host sent a minimum MTU sized packet, it cannot fallback to a smaller MTU using PLMTUD hence there is no recovery. Note the encapsulation is being done when packet egress a domain and there is no expectation that all the potential paths outside of the domain have a large enough MTU to accommodate encapsulation.

Sending encapsulated packets into the Internet requires that they can successfully transit the Internet. IPinIP encapsulation number could be filtered by some networks (similar to how networks can block packets with Hop-by-Hop Options header). Using a UDP encapsulation, such as VXLAN [RFC7348], might have better success than IPinIP.

All potential receivers would need to do decapsulation. This could be modeled as an anonymous encapsulation. Currently, this is not enabled on commodity host stacks, and would be a major change in deployment.

Packets to a destination may undergo Network Address Translation such that the outer addresses might not match the inner addresses of an encapsulation. If a flow contains a mix of encapsulated and non-encapsulated packets then the destination may view packets in the same flow as being in different flows. In order to prevent this, a router could encapsulate all packets, but that would be very costly for what is currently a narrow use case.

3. Arguments against in-flight extension header removal

Section 4 of [I-D.smith-6man-in-flight-eh-insertion-harmful] presents the problems of in-flight extension header removal in the context of extension headers being inserted in-flight. If extension headers are inserted in-flight then it is expected that those headers are removed before exiting the domain in which they were inserted. Failure to remove inserted extension headers could have detrimental behaviors include systematic packet drop and and leaking sensitive information outside of a limited domain.

This specification only allows removal of extension headers that were created by the source host, so the problems related to failing to remove inserted extension headers are not directly relevant. However, the effects of failing to remove non-inserted extension headers that we're intended to be removed by the operator can still be considered.

[I-D.smith-6man-in-flight-eh-insertion-harmful] describes the possible causes of extension header removal to fail:

* Implementation bugs

* Partial Node Failure

* Operator Configuration Error

With respect to removing non-inserted extension headers, the effects of these different failure modes are the same.

Given the current data, the most probable effect when extension headers are not removed as intended is that those packets will be dropped in the Internet. Since the primary purpose of dropping Hop-by-Hop or Routing headers is to avoid packet loss, failure to remove an extension header does not introduce any new detrimental or incorrect behavior. If extension headers aren't removed as intended then they may be processed by the network instead of dropped; this behavior is also correct and protocol conformant.

The secondary purpose for removing extension headers in-flight is to avoiding leaking information outside of a limited domain. If an egress router fails to remove an extension header then sensitive information may be exposed and this is a security risk. However, even without extension header removal, a firewall would still be needed to block packets with Hop-by-Hop Options or Routing headers from leaving the limited domain in order to enforce security policy. There is no reason to believe that a firewall that blocks packets would be no less susceptible to bugs, partial node failures, or configuration errors than one that removes extension headers and forwards packets.

4. Considerations

4.1. Reflection of Hop-by-Hop Options

Some Hop-by-Hop options are designed to be reflected by a remote host back to the sender. For example, IOAM Loopback [RFC9332] is used to report measurements on the forward path of a sender, and the Minimum Path MTU Hop-by-Hop Option [RFC9268] returns the path MTU of the forward path to a sender. Note that Hop-by-Hop Options reflection is not guaranteed and hence is an opportunistic mechanism, hence it cannot be assumed that options will always be properly reflected.

In the case that an intermediate node removes Hop-by-Hop Options, reflection won’t happen since the destination host does not see the Hop-by-Hop option to be reflected. A sender should be cognizant of this and may want to limit the use of options that require reflection to destinations that it knows are in the same limited domain as itself.

4.2. End host processing of Routing Headers

Per [RFC8200], "If Segments Left is zero, the node must ignore the Routing header and proceed to process the next header in the packet". Effectively, this means once the last segment has been processed and the final destination is set then the routing header carries no useful information to any downstream nodes, so removal of the extension header doesn’t affect how the packet is processed.

A possible exception is that the destination host may elect to validate the Routing header. For instance, the end host may validate the HMAC TLV in a Segment Routing header. Since routing headers are most likely used only in limited domains, which is an explicit requirement in Segment Routing, the network nodes processing the routing header should know if the final destination participates is required to validate the routing header-- if it’s not then the header can be safely removed.

4.3. ICMP errors

When an ICMP error message is sent for a packet with removed extension headers, the packet headers in the ICMP data will be different then what the host sent. Operationally, this should not be an issue since a sender doesn't normally need to correlate ICMP errors to packets that were originally sent with Hop-by-Hop options or a Routing header, host stacks don't typically maintain sufficient state to make a precise correlation.

4.4. Use with Authentication Header

In-flight removal of Hop-by-Hop Options or the Routing header is incompatible with the Authentication Header. A node may attempt to detect the presence of an Authentication header and one is present it can take some other action than removing the Hop-by-Hop Options or the Routing header. Note that the Authentication Header is essentially deprecated.

5. Requirements

An intermediate node MAY remove a Hop-by-Hop Options extension header from a packet if the following conditions are met:

* The Payload Length of the packet is non-zero and the Hop-by-Hop options does not include a Jumbo Payload Option (if the packet contains a Jumbo Payload option then the Payload Length should be zero)

* The packet does not contain an Authentication Header. This is an optional condition as it is not required that a node scans the IPv6 header chain to determine if an Authentication Header is present. If it is unknown whether the packet contains an Authentication Header then it is the discrection of the node to proceed with removing the Hop-by-Hop Options header or take some other action such as discarding the packet.

An intermediate node MAY remove a Routing header extension header from a packet if the following conditions are met:

* The Destination Address has been set to the address of the final destination and the Segments Left field is zero

* The final destination is not required to process or validate the Routing header

* The routing header does not contain options (segment routing TLVs

for instance), or the destination host doesn’t need to process or validate the options.

* The packet does not contain an Authentication Header. This is an optional condition as it is not required that a node scans the IPv6 header chain to determine if an Authentication Header is present. If it is unknown whether the packet contains an Authentication Header then it is the disrection of the node to proceed with removing the Routing header or take some other action such as discarding the packet.

6. Procedures

This section describes the procedures for removing a Hop-by-Hop Options header, removing a Routing header, and removing a Hop-by-Hop Options header and Routing header at the same time.

6.1. Removing a Hop-by-Hop Options Header

The procedures for removing a Hop-by-Hop Options header are:

1. Save the value in the Next Header field of the Hop-by-Hop Options header in a temporary variable

2. Determine the length of the Hop-by-Hop Options header and save in a temporary variable. This is equal to the value of the Hdr Ext Len field times eight plus eight

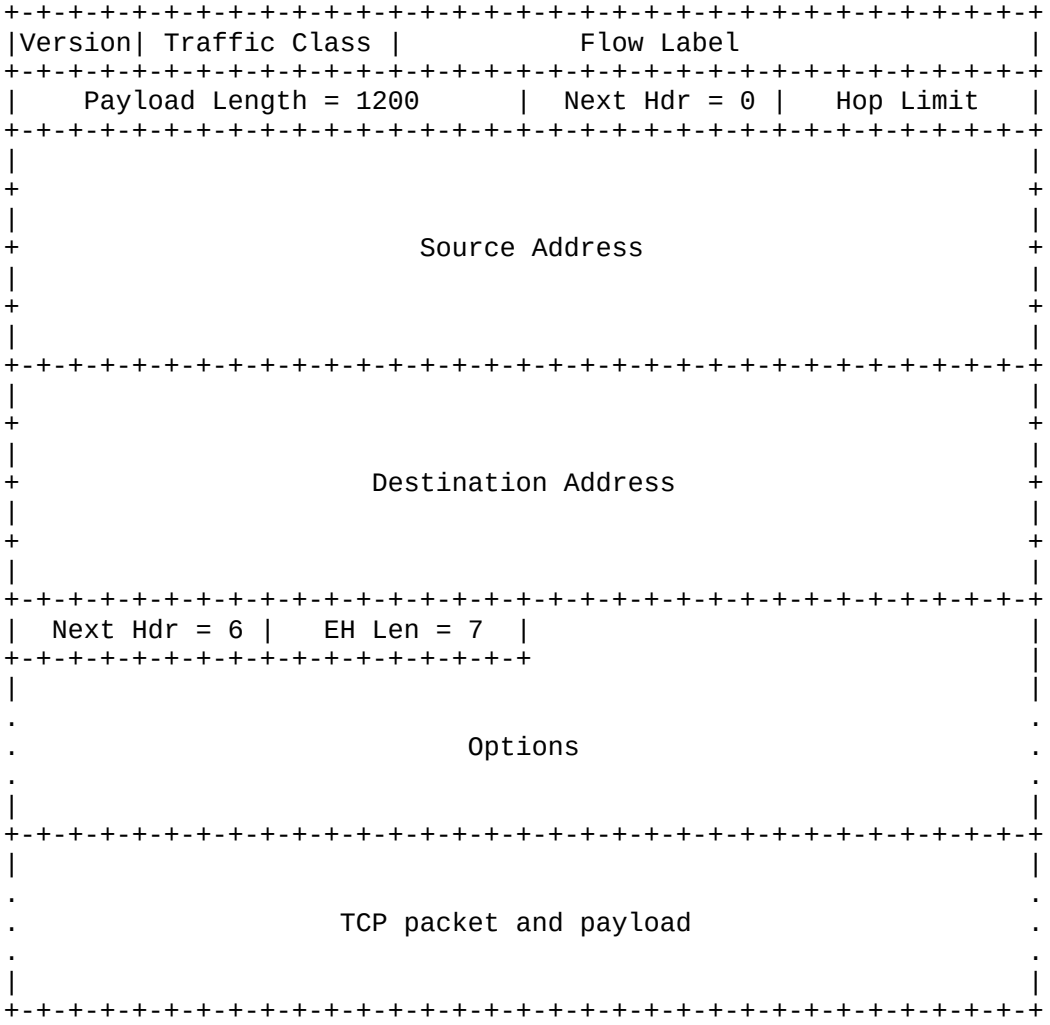
3. Copy the IPv6 header with length forty bytes to the offset in the packet equal to the length of the Hop-by-Hop options header that was determined in step 2

4. Set the Next Header field in the copied IPv6 header to the value saved in step 1

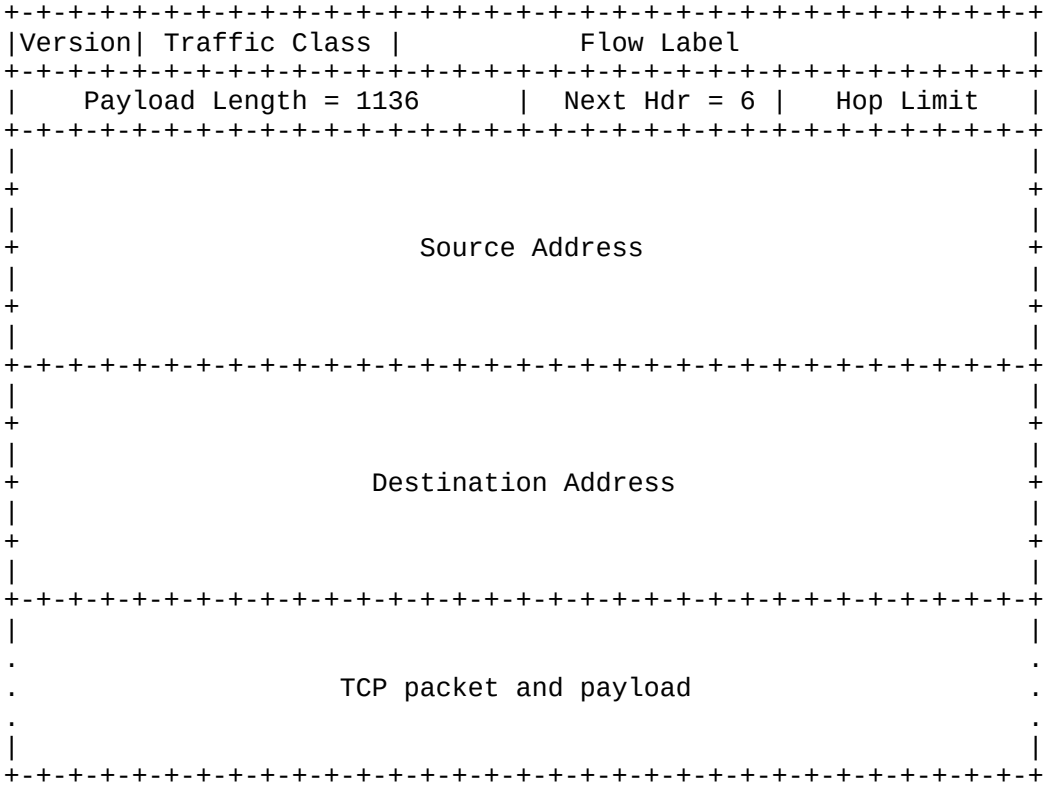
5. Subtract the length of the Hop-by-Hop Options header (determined in step 2) from the Payload Length in the copied IPv6 header. Set the result as the Payload Length in the copied IPv6 header

An example of removing Hop-by-Hop Options header is shown in the diagrams below.

The diagram below illustrates shows an example TCP/IPv6 packet with a Hop-by-Hop Options header. The Payload Length is 1200 bytes and the length of the Hop-by-Hop Options header is sixty-four bytes.



The diagram below illustrates the packet after the Hop-by-Hop Options header has been removed. Note that the Payload Length is now 1,136 bytes which is the original payload length minus the length of the Hop-by-Hop Options header that was removed.



6.2. Removing a Routing Header

The procedures for removing a Routing header are:

1. Save the value in the Next Header field of the Routing header in a temporary variable

2. Determine the offset of the first byte of the Routing header. This is equal to the forty plus the sum of lengths of any extension headers that precede the Routing header

3. Determine the length of the Routing header and save in a temporary variable. This is equal to the value of the Hdr Ext Len field times eight plus eight

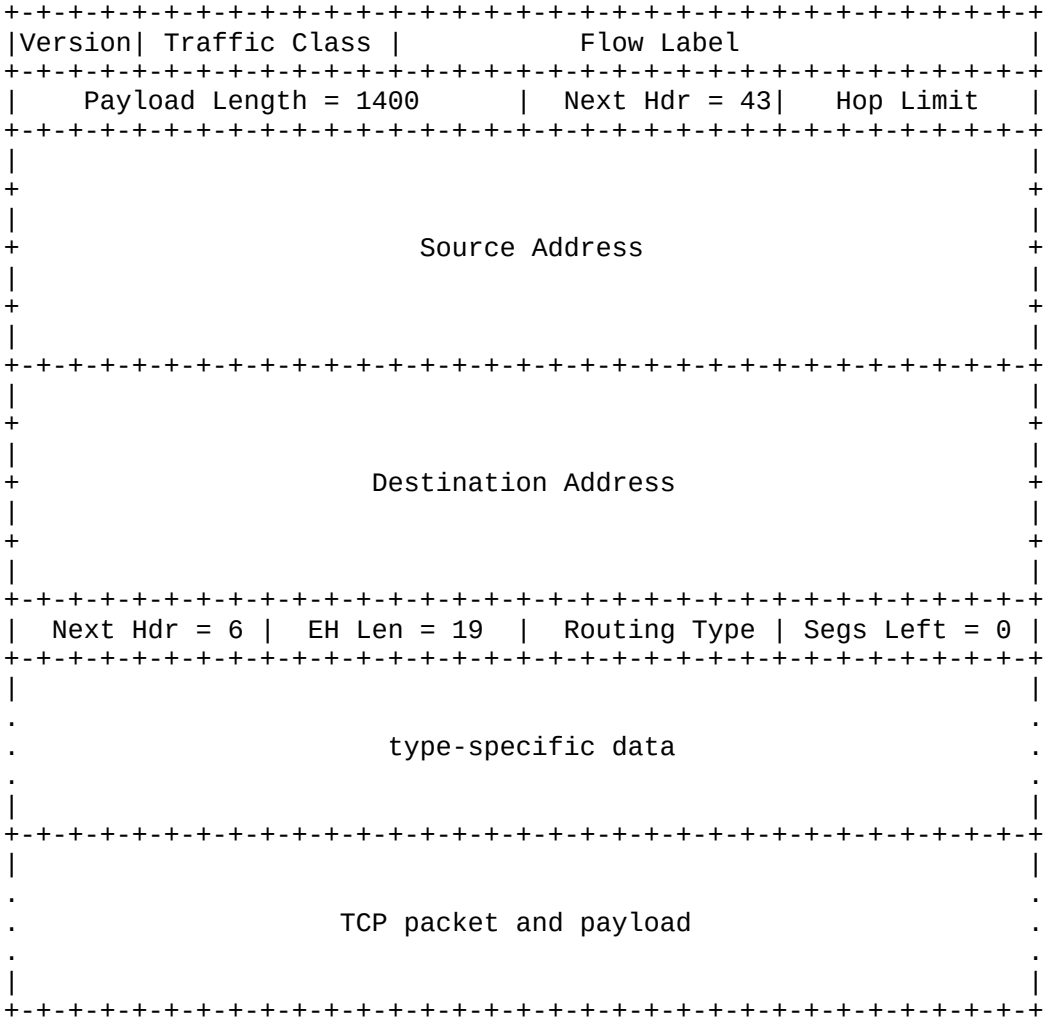
4. Copy the IPv6 and any extension headers preceding the Routing header to the offset in the packet equal to the length of the Routing header (determined in Step 3), where the number of bytes copied is equal to the offset determined in step 2

5. If there are no preceding extension headers then set the Next Header field in the copied IPv6 header to the value saved in step 1, else if there are preceding extension headers then set the Next Header field in the extension header that immediately preceded the Routing header to the value save in step 1.

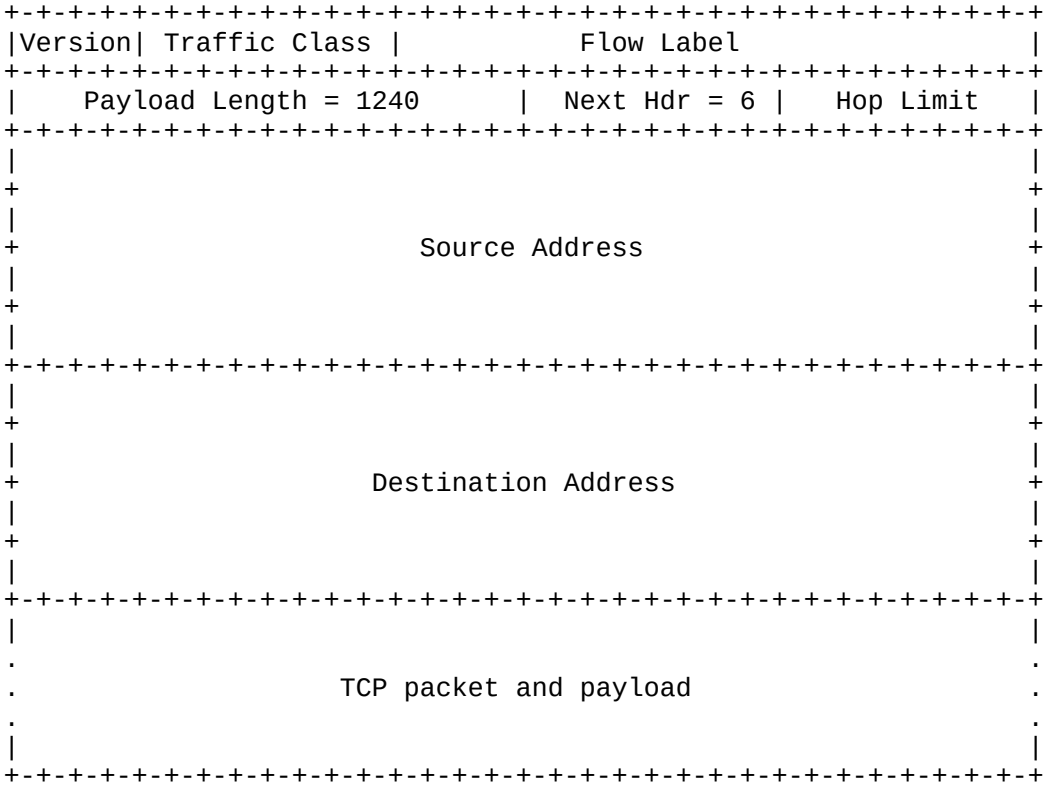
6. Subtract the length of the Routing header (determined in step 2) from the Payload Length in the copied IPv6 header. Set the result as the Payload Length in the copied IPv6 header

An example of removing a Routing header is shown in the diagrams below.

The diagram below illustrates shows an example TCP/IPv6 packet with a Routing header. The Payload Length is 1400 bytes and the length of the Routing header is 160 bytes. The Segments Left field is set to zero so that the Routing header may be removed.



The diagram below illustrates the packet after the Routing header has been removed. Note that the Payload Length is now 1,240 bytes which is the original payload length minus the length of the Routing header that was removed.



6.3. Removing both Hop-by-Hop Options and a Routing Headers

The procedures for removing both a Hop-by-Hop Options header and a Routing header where the Routing header immediately follows the Hop-by-Hop Options header:

1. Save the value in the Next Header field of the Routing header extension header in a temporary variable

2. Determine the length of the Hop-by-Hop Options header and save in a temporary variable. This is equal to the value of the Hdr Ext Len field time eight plus eight

3. Determine the length of the Routing header and save in a temporary variable. This is equal to the value of the Hdr Ext Len field time eight plus eight

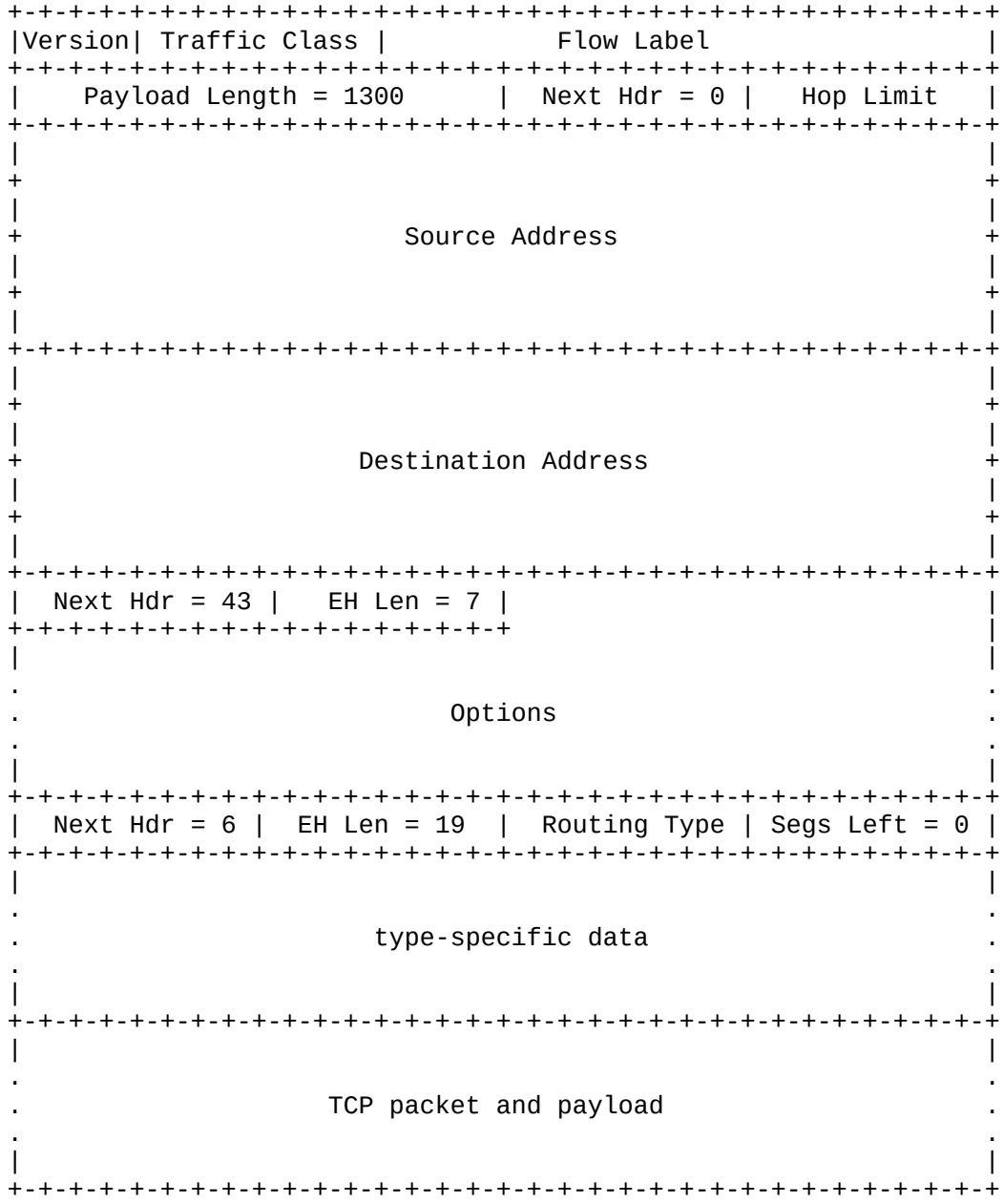
4. Copy the IPv6 header with length forty bytes to the offset in the packet equal to the length of the Hop-by-Hop Options header (determined in step 2) plus the length of the Routing header (determined in step 3)

5. Set the Next Header field in the copied IPv6 header to the value saved in step 1

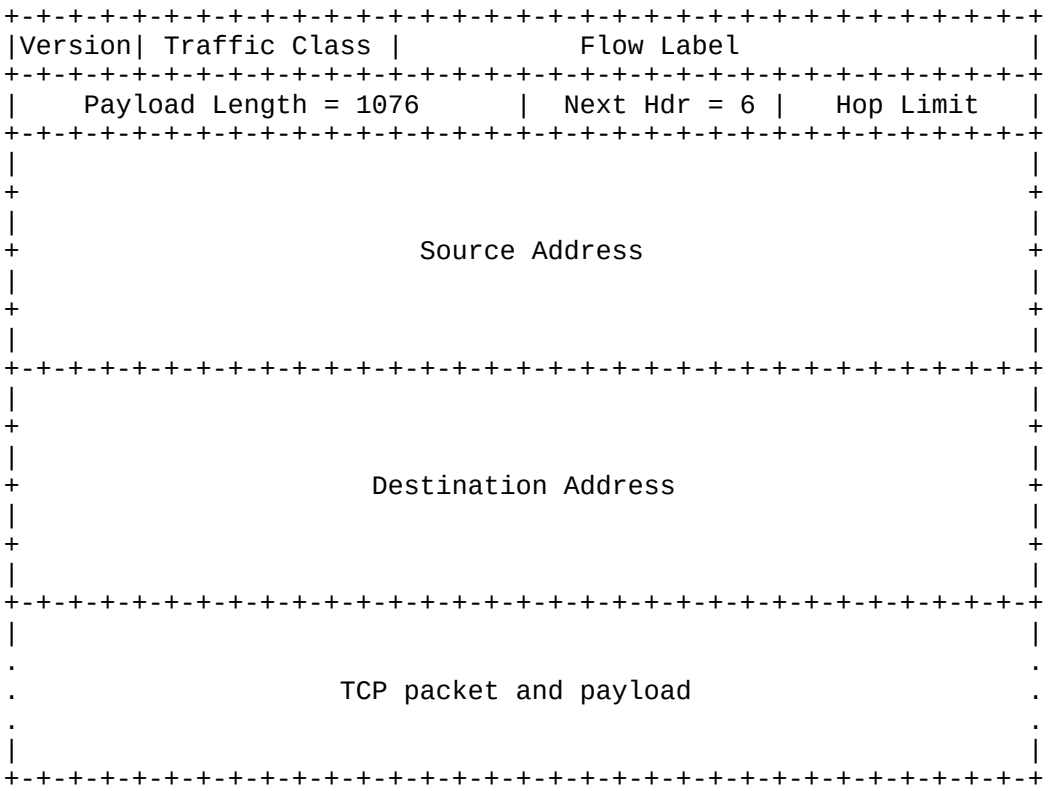
6. Subtract the length of the Hop-by-Hop Options header plus the length of the Routing header (values determined in step 2 and step 3) from the Payload Length in the copied IPv6 header. Set the result as the Payload Length in the copied IPv6 header

An example of removing a Hop-by-Hop Options header a Routing header is shown in the diagrams below.

The diagram below illustrates an example TCP/IPv6 packet with both a Hop-by-Hop Options header and a Routing header. The Payload Length is 1,300 bytes, the length of the Hop-by-Hop Options header is sixty-four bytes, the length of the Routing header is 160 bytes. The Segments Left field is set to zero so that the Routing header may be removed.


Version
Traffic Class
Flow Label
Payload Length = 1300
Next Hdr = 0
Hop Limit
Source Address
Destination Address
Next Hdr = 43
EH Len = 7
Options
Next Hdr = 6
EH Len = 19
Routing Type
Segs Left = 0
type-specific data
TCP packet and payload

The diagram below illustrates the packet after the Hop-by-Hop Options header and the Routing header have been removed. Note that the Payload Length is now 1,076 bytes which is the original payload length minus the length of the Hop-by-Hop Options header and the Routing header that were removed.



7. Implementation Considerations

Removal of extension headers must be efficient and considered a "fast path" operation in a router [RFC9673]. The most computationally complex part of removing extension headers is moving the IPv6 header. There are two methods to move the bits of the IPv6 header: memory copy and scatter/gather.

7.1. Copying the IPv6 Header

Extension header removal can be accomplished by performing a data copy of the IPv6 header (forty bytes) to the offset after the extension header being removed minus forty bytes. Since the number of bytes being moved is relatively small and fits within a typical cacheline, the data copy is amenable to efficient implementation in hardware or software. Once the copy completes, the pointer to the packet is advanced by the length of data removed. Note that an implementation may choose to move the link layer header as well.

7.2. Scatter/gather

Scatter/gather allows a packet to be constructed from a list of memory buffers where each buffer has a data pointer and length. To use scatter/gather for extension header removal, a receiver might employ header/data split to store the packet as two buffers in memory: the first buffer contains the link layer and IPv6 headers, and the second buffer contains the data following the IPv6 header. Removing an extension headers entails advancing the pointer to the second buffer by the length of the extension header being removed.

8. Updates to RFC8200

[RFC8200] is updated to allow inflight removal of Hop-by-Hop Options and the Routing header.

The following text replaces the third paragraph of Section 4 of [RFC8200] (where [THIS-DRAFT] would be replaced by a reference to the RFC for this draft).

OLD (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.

NEW

> Extension headers (except for the Hop-by-Hop Options header and the Routing 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. A Hop-by-Hop Options header and a Routing header may be removed from a packet by a node along a packet's delivery path per the requirements and procedures of [THIS-DRAFT].

The following should be inserted before the last third paragraph of Section 4.4 of [RFC8200] (where [THIS-DRAFT] would be replaced by a reference to the RFC for this draft).

NEW (RFC8200)

> A Routing header may be removed from a packet by a node along a packet's delivery path per the requirements and procedures of [THIS-DRAFT].

9. Security Considerations

Removing Hop-by-Hop Options and Routing headers inflight is a potential security advantage in that it reduces visibility of sensitive data to untrusted parties. Otherwise, this specification does not introduce any new security concerns,

10. IANA Considerations

There are no IANA considerations in this specification.

11. References

11.1. 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>.

11.2. Informative References

[APNIC-EH] Huston, G., "IPv6 extension headers revisited", October 2022, <https://blog.apnic.net/2022/10/13/ipv6-extension-headers-revisited>.

[I-D.smith-6man-in-flight-eh-insertion-harmful] Smith, M., Kottapalli, N., Bonica, R., Gont, F., and T. Herbert, "In-Flight IPv6 Extension Header Insertion Considered Harmful", Work in Progress, Internet-Draft, draft-smith-6man-in-flight-eh-insertion-harmful-02, 30 May 2020, <https://datatracker.ietf.org/doc/html/draft-smith-6man-in-flight-eh-insertion-harmful-02>.

[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>.

[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>.

[RFC7348] Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348, DOI 10.17487/RFC7348, August 2014, <https://www.rfc-editor.org/info/rfc7348>.

[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>.

[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>.

[RFC9098] Gont, F., Hilliard, N., Doering, G., Kumari, W., Huston, G., and W. Liu, "Operational Implications of IPv6 Packets with Extension Headers", RFC 9098, DOI 10.17487/RFC9098, September 2021, <https://www.rfc-editor.org/info/rfc9098>.

[RFC9268] Hinden, R. and G. Fairhurst, "IPv6 Minimum Path MTU Hop-by-Hop Option", RFC 9268, DOI 10.17487/RFC9268, August 2022, <https://www.rfc-editor.org/info/rfc9268>.

[RFC9332] De Schepper, K., Briscoe, B., Ed., and G. White, "Dual-Queue Coupled Active Queue Management (AQM) for Low Latency, Low Loss, and Scalable Throughput (L4S)", RFC 9332, DOI 10.17487/RFC9332, January 2023, <https://www.rfc-editor.org/info/rfc9332>.

[RFC9673] Hinden, R. and G. Fairhurst, "IPv6 Hop-by-Hop Options Processing Procedures", RFC 9673, DOI 10.17487/RFC9673, October 2024, <https://www.rfc-editor.org/info/rfc9673>.

Author’s Address

Tom Herbert
XDPnet
Los Gatos, CA,
United States of America
Email: tom@herbertland.com