H. Song
Futurewei
Z. Li
S. Peng
Huawei Technologies
J. Guichard
Futurewei
17 February 2025

Approaches on Supporting IOAM in IPv6
draft-song-ippm-ioam-ipv6-support-05

Abstract

IOAM pre-allocated trace option data fields can be encapsulated in IPv6 HbH options header as described in RFC9486. However, due to the potential large size of the trace data and the HbH extension header location in the IPv6 packets, the scheme creates practical challenges for implementation, especially when other extension headers, such as a routing header, also exist and require on-path processing. We propose two alternative approaches to address this challenge in addition to IOAM DEX: separating the IOAM incremental trace data from the IOAM instruction header, or applying the segment IOAM trace data export scheme, based on the network scenario and application requirements. We discuss the pros and cons of each approach in this document.

Requirements Language

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

Status of This Memo

This Internet-Draft will expire on 21 August 2025.

Copyright Notice

Table of Contents

1. Introduction

In-situ OAM (IOAM) [RFC9197] defines two trace options, pre-allocated trace option and incremental trace option, which record hop-by-hop data along a packet's forwarding path. [RFC9486] describes the method to encapsulate IOAM pre-allocated trace option data fields in IPv6. Because the trace options requires per hop processing, such options can only be encapsulated in IPv6 Hop-by-Hop (HbH) options header.

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

IOAM trace option cannot be applied if it makes other extension headers inaccessible. Even if the other headers can be reached, the deeper they are, the higher the cost to access and process them, and the lower the forwarding performance. Note that [RFC9486] does not support the incremental trace option because it would expand the HbH header at each hop and push back all other headers after it. The changing location of the later extension headers could further complicate the hardware implementation and affect the forwarding performance.

The issue becomes more severe when SRv6 and IOAM coexist. The Segment Routing Extension Header (SRH) [RFC8754] is encapsulated in a routing header which is after the HbH options header. SRH itself can be large. It requires read and write operations at each SRv6 segment endpoint node. If it is deeply embedded in a packet and its location keeps shifting, either it is beyond the reach of hardware or the forwarding performance degrades.

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

The second recourse is to limit the IOAM to SRv6 nodes only. That is, consider SRv6 as an overlay tunnel over IPv6 and apply the IOAM pipe mode as discussed in [I-D.song-ippm-ioam-tunnel-mode], which only collects data at each SRv6 segment endpoint nodes. To realize this, [I-D.ali-spring-ioam-srv6] describes an approach that encapsulates the IOAM option data fields in an SRH TLV. [RFC9259] describes another approach to enable postcard-based telemetry for SRv6 without needing IOAM option encapsulation. In either case, the SRH is close to the packet front and its location is fixed. While these approaches are useful for use cases that only need to monitor the segment endpoints, it fails to cover all the IPv6 nodes on the packet forwarding path in an IOAM domain.

So the proposition of this draft is, if we need to apply IOAM on all nodes in an SRv6 network, how we can amend the approach in [RFC9486] or use alternative approaches to circumvent the aforementioned issues. In this draft, we propose two viable approaches: (1) separating the IOAM trace data from the instruction header to a different extension header option after the routing header if it exists, and (2) applying the segment IOAM trace export scheme. We discuss the pros and cons of each approach.

2. IOAM Trace Data Separate and Postpose

An IOAM trace type data fields contain two parts: instruction and trace data. Although by convention the trace data part immediately follows the instruction part, there is not fundamental reason why these two parts must stick together. This observation provides us an optimization opportunity to amend the original proposal in [RFC9486].

We separate the IOAM trace type data fields into the instruction part and the trace data part. We encapsulate only the instruction part in the HbH options header, and encapsulate the trace data part in another extension header option after all the IPv6 extension headers that need to be examined and processed on the packet forwarding path (e.g., a routing header). This arrangement allows us to use the incremental trace option efficiently. Even if the data trace increases its size at each node, all IPv6 extension headers before it remain a fixed size, and new data is guaranteed to be inserted at a fixed location.

Figure 1 shows the HbH option format for IOAM incremental trace type instruction. The field specification is identical to that in [RFC8200] and [RFC9197].

```
 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 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |   Reserved      |   IOAM Type   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<---
|        Namespace-ID           |NodeLen  | Flags | RemainingLen| IOAM
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ Trace
|               IOAM-Trace-Type                 |  Reserved     | Type
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<Inst.
```

Figure 1: HbH Option Format for IOAM Incremental Trace Type Instruction

Figure 2 shows the TLV option format for IOAM trace type data. The IOAM trace type data format is compliant with [RFC9197].

```
 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 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   IOAM Type   |     Length    |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                      IOAM Trace Type Data                     ~
~                                                               ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Option Format for IOAM Incremental Trace Type Data

2.1. IOAM Incremental Trace Data Encapsulation

We have basically two methods to encapsulate the IOAM incremental trace data. First, we can define a new IPv6 extension header which is dedicated to metadata. Once standardized, this extension header can also be used to host potential metadata from other applications such as NSH for SFC [RFC8300]. Second, this option can be carried as a TLV option in another existing extension header such as the Destination Option Header (DOH). The only requirement is that this extension header should be the last one in the extension header chain. The first method is cleaner but it requires to standardize a new extension header type; the second method is simpler but it needs to overcome the access constraints exerted by [RFC8200].

3. Segment IOAM Data Export

If the overhead of the IOAM trace type data fields is under control, we may still manage to encapsulate both instruction and data in HbH options header as described in [RFC9486]. To this end, we introduce two sub-approaches.

3.1. Independent of SRv6

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

This approach requires some moderate updates to the IOAM trace type data fields, as described in [I-D.song-ippm-segment-ioam]. Figure 3 shows the format of the HbH Option Header containing Segment IOAM trace type data fields. A flag bit (#23) in the Flags field is used to indicate the current header is a segment IOAM header. In this context, the last octet in the IOAM header is partitioned into two 4-bit nibbles. The first nibble (SSize) is used to save the segment size and the second nibble (RHop) is used to save the remaining hops. This limits the maximum segment size to 15.

```
 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 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |    Reserved     |   IOAM Type   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-------+
|       Namespace-ID            |NodeLen|Flags|1| SSize | RHop  | IOAM
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ Segment
|               IOAM-Trace-Type                 |  Reserved     | Trace
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ Type
|                                                               | Data
|                  Node Data List []                            | Fields
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+<-------+
```

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

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

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

3.2. Export at SRv6 node

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

The incremental trace type is more proper in this approach than the pre-allocated trace type, due to the uncertainty of the number of hops between two segment endpoints. In an extreme case when every node is also an SRv6 node, this approach regresses to a per-hop postcard-based telemetry approach such as IOAM DEX as described in [RFC9326].

4. Direct Export Option

It is worth noting that, instead of using the IOAM trace options, IOAM Direct Export (DEX) Option Type [RFC9326] can be used for fixed and small packet overhead since it only needs to encapsulate a fix-size instruction header in the HbH option header. The scheme is covered in [RFC9486].

5. Comparison

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

Approach	Pros	Cons
IOAM Trace Option in HbH (RFC9486)	Comply w/ IOAM Data Spec	Variable, long HbH header impedes access of other extension headers
IOAM Trace Data Separate and Postpose (Sec. 2)	Fix-size and short HbH header, good for accessing other extension headers	Need extra extension header option to hold trace data
Segment IOAM Data Export (Sec. 3.1)	Fix-size and controllable HbH header size	Need to update IOAM trace type data field spec.
Trace Export at SRv6 nodes (Sec. 3.2)	Can be done through configuration	Specific to SRv6; No better than IOAM DEX in the worst case
IOAM Direct Export in HbH (RFC9486)	Comply w/ IOAM DEX Spec; Fix-size and short HbH	Need export data correlation, and other issues of DEX (RFC9326)

Figure 4: Comparison of Different Approaches

IOAM trace option is easy to implement and extensible for data types. Complementary to [RFC9486], the scalable solutions for using it in IPv6 networks discussed in this document can fully carry out the benefits of IOAM trace option and meanwhile avoid potential issues associated with variable and high header overhead.

6. Security Considerations

No new security issue is identified other than those for IOAM trace option [RFC9197], IOAM DEX [RFC9326], and IPv6 extension headers [RFC8200].

7. IANA Considerations

This document requires no IANA actions.

8. Acknowledgments

9. Normative References

[I-D.ali-spring-ioam-srv6]
Ali, Z., Gandhi, R., Filsfils, C., Brockners, F., Nainar, N. K., Pignataro, C., Li, C., Chen, M., and G. Dawra, "Segment Routing Header encapsulation for In-situ OAM Data", Work in Progress, Internet-Draft, draft-ali-spring-ioam-srv6-06, 10 July 2022, <https://datatracker.ietf.org/doc/html/draft-ali-spring-ioam-srv6-06>.

[I-D.song-ippm-ioam-tunnel-mode]
Song, H., Li, Z., Zhou, T., and Z. Wang, "In-situ OAM Processing in Tunnels", Work in Progress, Internet-Draft, draft-song-ippm-ioam-tunnel-mode-00, 27 June 2018, <https://datatracker.ietf.org/doc/html/draft-song-ippm-ioam-tunnel-mode-00>.

[I-D.song-ippm-segment-ioam]
Song, H. and T. Zhou, "Control In-situ OAM Overhead with Segment IOAM", Work in Progress, Internet-Draft, draft-song-ippm-segment-ioam-01, 17 April 2018, <https://datatracker.ietf.org/doc/html/draft-song-ippm-segment-ioam-01>.

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

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

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

[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>.

[RFC9197] Brockners, F., Ed., Bhandari, S., Ed., and T. Mizrahi, Ed., "Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9197, DOI 10.17487/RFC9197, May 2022, <https://www.rfc-editor.org/info/rfc9197>.

[RFC9259] Ali, Z., Filsfils, C., Matsushima, S., Voyer, D., and M. Chen, "Operations, Administration, and Maintenance (OAM) in Segment Routing over IPv6 (SRv6)", RFC 9259, DOI 10.17487/RFC9259, June 2022, <https://www.rfc-editor.org/info/rfc9259>.

[RFC9326] Song, H., Gafni, B., Brockners, F., Bhandari, S., and T. Mizrahi, "In Situ Operations, Administration, and Maintenance (IOAM) Direct Exporting", RFC 9326, DOI 10.17487/RFC9326, November 2022, <https://www.rfc-editor.org/info/rfc9326>.

[RFC9486] Bhandari, S., Ed. and F. Brockners, Ed., "IPv6 Options for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9486, DOI 10.17487/RFC9486, September 2023, <https://www.rfc-editor.org/info/rfc9486>.

Authors’ Addresses

Haoyu Song
Futurewei
United States of America
Email: haoyu.song@futurewei.com

Zhenbin Li
Huawei Technologies
China

Email: lizhenbin@huawei.com

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

James Guichard
Futurewei
United States of America
Email: james.n.guichard@futurewei.com