Workgroup: SPRING Working Group
Internet-Draft:
draft-li-spring-srh-tlv-processing-
programming-06
Published: 19 February 2024
Intended Status: Standards Track
Expires: 22 August 2024
Authors: C. Li
Huawei Technologies
Y. Xia
Huawei Technologies
D. Dhody
Huawei Technologies
Z. Li
Huawei Technologies

SRH TLV Processing Programming

Abstract

This document proposes a mechanism to program the processing rules of Segment Routig Header (SRH) optional TLVs explicitly on the ingress node. In this mechanism, there is no need to configure local configuration at the node to support SRH TLV processing. A network operator can program to process specific TLVs on specific segment endpoint nodes for specific packets on the ingress node, which is more efficient for SRH TLV processing.

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 22 August 2024.

Copyright Notice

Table of Contents

1. Introduction

Segment routing (SR) [RFC8402] is a source routing paradigm that explicitly indicates the forwarding path for packets at the ingress node by inserting an ordered list of instructions, called segments.

When segment routing is deployed on the IPv6 data plane, it is called SRv6 [RFC8754]. For support of SR, a new routing header called Segment Routing Header (SRH), containing a list of segments, optional TLVs and other information, has been defined in [RFC8754].

Currently, when TLVs are carried in an SRH, they are ignored by the nodes by default, unless there are some local policies on nodes to enable the SRH TLV processing [RFC8754].

When a node is configured to process a TLV, it needs to examine all the SRH TLVs for processing a single TLV (TLVs except HMAC in SRH MAY appear in any order), which is inefficient.

Furthermore, in order to deploy a new service, network operators need to configure multiple nodes along the path to support SRH TLVs processing, which is complicated. Also, it is not easy to dynamically adjustment the local policy for meeting dynamic service requirements. However, SRv6 does not have the compability to program the rules of SRH TLVs processing on the ingress node currently.

In summary, network operator are not able to program the SRH TLV processing rules on the ingress node to process specific TLVs on specific segment endpoint nodes for some packets dynamically.

This document proposes a mechanism to program the SRH TLVs processing rules explicitly and dynamically on the ingress node. In this mechanism, there is no need to configure nodal local policy to support SRH TLV processing. It can be used for the following use cases:

*Service Function Chaining (SFC): In SFC, SRH TLVs like Firewall related TLVs [I-D.guichard-spring-srv6-simplified-firewall] may only be processed on some specific nodes instead of all the nodes along the path.

*Smart In-situ OAM (IOAM): In IOAM, the IOAM metadata will be collected by all the nodes along the path. However, in the most cases, only the metadatda on some nodes are important for OAM, while the others are redundant or irrelevant. For example, congestion may occur only on some nodes, not all nodes. In addition, congestion may occur on link A at the last moment and may occur on link B at the next moment. To implement smarter and more efficient IOAM, the scope of IOAM metadata collection needs to be dynamically adjusted (without modifying the segment list) based on the result of IOAM measurement to reduce unnecessary IOAM information collection.

2. Terminology

This document makes use of the terms defined in [RFC8754], and the reader is assumed to be familiar with that terminology. This document introduces the following terms:

TPI: TLV Processing Indicator

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. SRH TLV Processing Programming

This document defines a new flavor in SRv6 to indicate the SRv6 endpoint node to process SRH TLVs. Also, this document defines an SRH TLV processing rule TLV in SRH to describe how to process the TLV on SRv6 endpoint nodes.

3.1. TLV Processing Indicator Flavor

Currently, SRv6 endpoint nodes will ignore the SRH TLV if there is no local policy to enable processing.

When receives an SRv6 packet, in order to explicitly indicate to process SRH TLVs, a TLV Processing Indicator (TPI) Flavor is defined in this document. By default, the node should ignore the SRH TLV. With TPI flavor, SRH TLV processing can be triggered by TPI flavor SID without local configuration.

When a TPI flavor SID is processed at an SRv6 node, the node MUST process the SRH TLVs. Otherwise, the SRH TLVs SHOULD be ignored by default or processed based on the local policies as per [RFC8754].

3.2. TLV Processing Indicator TLV

When an SRv6 endpoint node receives an SRv6 packet with SRH TLVs, it will process all the TLVs within the SRH, but actually only some TLVs should be processed at this node while most of the TLVs SHOULD be skipped.

For example, SRH "S-class" and "D-class" TLVs [I-D.guichard-spring-srv6-simplified-firewall] are processed at Firewall node only and they SHOULD NOT be processed at other nodes along the path.

In order to enhance the performance of SRH TLV processing, this section defines TLV processing Indicator (TPI) TLV to describe how to process the SRH TLVs. If the TPI TLV appears in SRH, it MUST be the first TLV for better processing efficiency. Only one TPI TLV is allowed in SRH. If multiple TPI TLVs are included, only the first TLV will be processed and the rest will be ignored. If Its format is shown below.

[Editor's notes] This part may be moved to 6man draft in the future since this is an IPv6 dataplane extension.

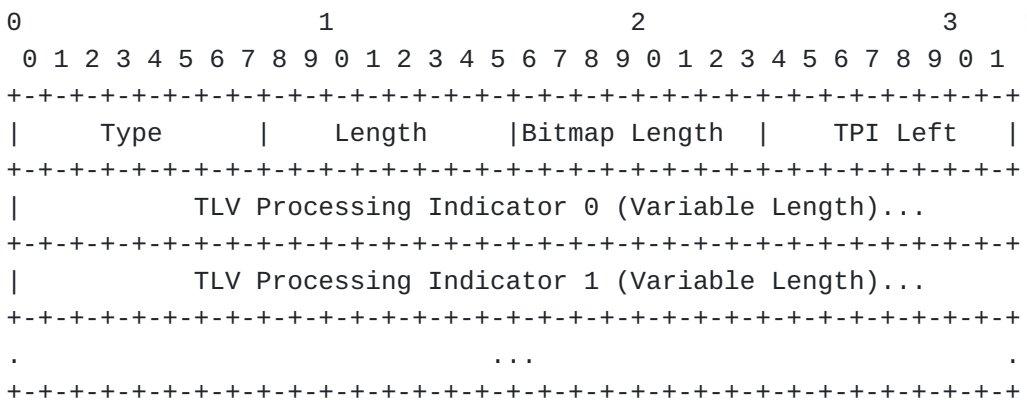

Figure 1. SRH TLV Processing Indicator TLV(Variable Length)

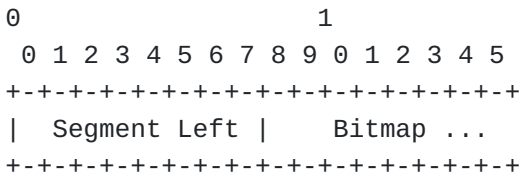

Figure 2. TLV Processing Indicator(TPI) Entry

*Type: type of TPI TLV, TBA. The TLV MUST be ignored if the node does not have the capability to process the TPI TLV.

*Length: Length of the TPI TLV.

*Bitmap Length: The length of Bitmap in a TLV Processing Indicator (TPI) entry, in byte. For instance, if there are 6 TLVs (exclude TPI TLV) within the SRH, the length of Bitmap is 1 bytes. If there are 12 TLVs within the SRH (exclude TPI TLV), the length of Bitmap is 2 Bytes.

*TPI Left: Index of the active TPI entry in the TPI TLV.

*TLV Processing Indicator: A TLV Processing Indicator indicates how to process the SRH TLVs at a specific node associated with the SID in SRH[TPI.SL]. An TPI TLV can include multiple TPI entries to specify the processing rules on multiple nodes. The length of a TPI entry is variable depends on the length of the bitmap.

-Segment Left: Segment Left (SL) is the key of a TPI entry, which indicates the node associated with SID in SRH[TPI.SL] needs to process the SRH TLVs according to the TPI entry. When a node processes the TPI TLV, it examines the TPI entry located at TPI-List[TPI Left]. If the value of SRH.SL is equivalent with TPI-List[TPI Left].SL, the node MUST process

the SRH TLVs based on the TPI entry, and decrement TPI Left by 1 if TPI Left is greater than 0. If the value of SRH.SL is not equivalent, the processing of the SRH TLVs is skipped.

-Bitmap: The bitmap indicates which SRH TLVs are needed to be processed on the node associated with SRH[TPI.SL]. Setting the nth bit means the (n+2)th SRH TLV is required to be processed, since the first TLV in SRH MUST be the TPI TLV and the index of the bitmap begins with 0. For instance, If the second TLV (the First TLV after the TPI TLV) in the SRH is needed to be processed on the node, the first bit (bit 0) in the bitmap is set. If the second TLV and the sixth TLV are needed to be processed, the bit 0 and bit 4 are set in the bitmap.

Multiple TPI Entries are encoded after the first 32 bits in TPI TLV following the descending order of SL in TPI entries.

[Editor's notes: The TPI TLV MUST be the first TLV in SRH, therefore, the HMAC TLV should be the second one, this may require to update [RFC8754]].

4. Illustration

In order to easy understanding, this section describes a simple example. The topology is shown in Figure 3.

For instance, an SRv6 packet is forwarded from node 1 to node 6. Therefore, <SID2, SID3, SID4, SID5, SID6> is encoded in the SRH. According to the sevice requirements, the SID3 and SID6 are TPI flavor SID, which indicate the nodes to process SRH TLVs. 4 TLVs are encoded in the SRH, TLV 1 and TLV2 will be processed at node 3, while the TLV3 and TLV 4 will be processed at node 6. Other nodes are not required to process any SRH TLVs of this packet.

In the SRH TLV fields, a TPI TLV and the other 4 TLVs are encoded, and the TPI TLV is the first TLV. The value of bitmap length field is 1 since there are only 4 TLVs (TPI TLV is excluded) in the SRH.

Two TPI entries are encoded after the first 32 bits in TPI TLV. The length of each TPI entry is 2 bytes, 1 byte for SL and 1 byte for the bitmap.

The first TPI entry (TPI-List[0]) describes the SRH TLV processing rules on node 6, and its SL is 0. The bit 2 and bit 3 are set in its bitmap to indicate to process the TLV3 and TLV 4.

The last TPI entry (TPI List[1]) describes the SRH TLV processing rules on node 3, and its SL is 3. The bit 0 and bit 1 are set in its bitmap to indicate to process the TLV1 and TLV 2.

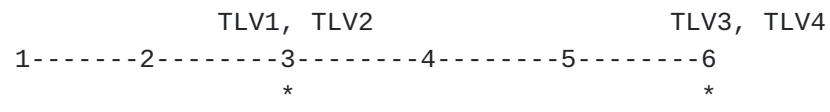

Figure 3. Illustration of ESTP

* means TPI flavor SID is processed on that node.

The TPI Left is initiated as 1 at node 1, and the encoding of TPI TLV in the case is shown below.

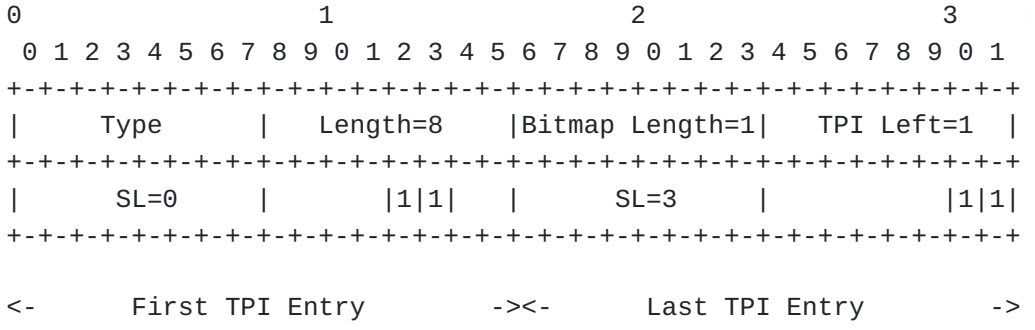

Figure 4. Instantiation of TPI TLV at node 1

When the packet is received at node 2, the SRH TLVs are skipped by default.

When the packet is received at node 3, the SRH TLVs are processed because the SID3 is a TPI floavor SID allocated by node 3. When the node 3 processes SRH TLVs, the first TLV to be processed is the TPI TLV. Node 3 compares the TPI-List[TPI Left].SL and SRH.SL, if they are equivalent, the node 3 processes the TLV 1 and TLV 2 according to the bitmap and updates the TPI Left to be 0.

When the packet is received at node 4, the SRH TLVs are skipped by default.

When the packet is received at node 5, the SRH TLVs are skipped by default.

When the packet is received at node 6, the SRH TLVs are processed because the SID6 is a TPI floavor SID allocated by node 6. When the node 6 processes SRH TLVs, the first TLV to be processed is the TPI TLV. Node 6 compares the TPI-List[TPI Left].SL and SRH.SL, if they are equivalent, the node 6 processes the TLV 3 and TLV 4 according to the bitmap. The TPI Left will not be updated because it is 0 already.

5. IANA Considerations

TBD

6. Security Considerations

TBD

7. Contributors

TBD

8. Acknowledgements

TBD

9. References

9.1. 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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[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>.

[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>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

9.2. Informative References

[I-D.guichard-spring-srv6-simplified-firewall]

Guichard, J., Filsfils, C., Bernier, D., Li, Z., Clad, F., Camarillo, P., and A. Abdelsalam, "Simplifying Firewall Rules with Network Programming and SRH Metadata", Work in Progress, Internet-Draft, draft-guichard-spring-srv6-simplified-firewall-02, 8 April 2020, <https://datatracker.ietf.org/doc/html/draft-guichard-spring-srv6-simplified-firewall-02>.

Authors' Addresses

Cheng Li
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing
100095
China

Email: c.l@huawei.com

Yang Xia
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing
100095
China

Email: yolanda.xia@huawei.com

Dhruv Dhody
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore 560066
India

Email: dhruv.ietf@gmail.com

Zhenbin Li
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing
100095
China

Email: lizhenbin@huawei.com