Abstract

Extension Headers (EH) carry information on in-network services and functions in an MPLS network. This document describes an architecture for EHs and what actions an EH capable Label Switching Router (LSR) takes when finding or not finding an EH in the packet.

Multiprotocol Label Switching (MPLS) is a widely deployed forwarding technology. It uses label stack entries that are pre-pended to either the EH or the ACH which in turn is pre-pended to the payload. The label stack entries are used to identify the forwarding actions by each LSR. Actions may include pushing, swapping or popping the labels, and using the labels to determine the next hop for forwarding the packet. Labels may also be used to establish the context under which the packet is forwarded.

The extension headers are carried after the MPLS Label Stack, and the presence of EHs are indicated in the label stack by an Extension Header Indicator (EHI).

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 October 7, 2022.

Copyright Notice

Copyright (c) 2022 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 (https://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. Requirement Language ................................. 4
2. Specification ............................................ 4
   2.1. Extension Header Overview ........................... 4
   2.2. Extension Header Terminology ........................ 4
3. Extension Header Basics ................................... 5
   3.1. General Principles .................................. 5
   3.2. LSPs in a EH capable Network ......................... 5
   3.3. EH capable nodes .................................... 6
   3.4. EH path and LSP ...................................... 6
   3.5. Announcement of EH Capability ........................ 7
   3.6. LSP establishment with LDP Downstream on Demand (DoD) in an EH capable network .......................... 7
   3.7. LSP establishment with LDP Downstream Unsolicited (DU) in an EH capable network .......................... 9
   3.8. Forwarding Behavior of EH Capable Nodes ............ 10
   3.9. EH for RSVP-TE tunnels ............................... 11
   3.10. Ways to indicate an EH after the Label Stack ....... 11
4. EH in VPNs ................................................. 11
5. EH and MPLS-SR ........................................... 11
6. Extension Header Applications .............................. 11
7. EH distribution and EH capability announcement ............ 12
8. Security Considerations ................................... 12
9. IANA Considerations ...................................... 12
10. Acknowledgements ........................................ 12
11. References ............................................... 12
   11.1. Normative References ............................... 12
   11.2. Informative References ............................. 13
Authors’ Addresses ......................................... 13
1. Introduction

This document specifies the architecture for the extension of MPLS to include Extension Headers (EH). EHS carry information on in-network services and functions in an MPLS network. This document describes an architecture for EHs and what actions an EH capable Label Switching Router (LSR) takes when finding or not finding an EH in the packet.

The extension headers are carried after the MPLS Label Stack, and the presence of EHs are indicated in the label stack by an Extension Header Indicator (EHI).

Below some example use cases are listed. More details will be found in [I-D.song-mpls-extension-header]

- In-situ OAM: In-situ OAM (IOAM) records flow OAM information within user packets while the packets traverse a network.
- Network Telemetry and Measurement: A network telemetry and instruction header can be carried as an extension header to instruct a node what type of network measurements should be performed on the packets.
- Network Security: Security related functions may require user packets to carry some metadata.
- Segment Routing and Network Programming: MPLS extension header could support MPLS-based segment routing. The details will be described in a separate draft.

It is possible to distinguish between two types of MPLS EHs, "hop-by-hop" (HBH) and "End to end" (E2E).

An HBH EH is processed by every node along an LSP, HBH EHs MAY be inserted by an ingress LSR or a transit LSR. A HBH EH MUST be removed by an LSR along the LSP or by the egress LSR. An LSR along the LSP may be configured to ignore HBH EHs.

An E2E EH will be inserted by the ingress LSR and, processed and MUST be removed by the egress LSR, no other LSR along the LSP will process the E2E EH.

Only EH capable LSRs will process EHs, LSR that are EH non-capable will ignore the EH and forward the packet as if the information was not there.
This document describes the interaction between EH capable neighbour LSRs, and between EH capable LSRs and a neighbour that is EH non-capable.

1.1. Requirement 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.

2. Specification

This document specifies the use of Extension Headers (EH) with MPLS. Further information on EH processing and formats will be found in [I-D.song-mpls-extension-header].

2.1. Extension Header Overview

Applications carried over an MPLS network may require that specific instructions and/or metadata are added to user packets. One such example could be In-situ OAM (IOAM) [I-D.brockners-inband-oam-requirements]. It is likely that new such applications will emerge over time.

One or more EHs may be added by an ingress node to an Extension Header Path (EHP) and be removed by one or more EH capable nodes along the EHP. Such ingress and egress nodes may be nodes at the head end and tail end of a Label Switched Path (LSP), or any other intermediate node of the LSP that is EH capable. For more details on EHPs see Figure 1.

2.2. Extension Header Terminology

This section lists the abbreviations and concepts that are used throughout this document in the context of Extension Headers.

- EH - Extension Header
- EHI - Extension Header Indicator
- LDP DoD - LDP Downstream on Demand
- LDP DU - LDP Downstream Unsolicited
- LSP - Label Switched Path
o LSR - Label Switching Router

The following concepts new for MPLS are defined:

o EH capable node - an LSR that can process Extension Headers and announce its EH capability

o EH capable LSR - this may be used interchangeably with EH capable node.

o EH non-capable node - an LSR that is unaware of and unable to process Extension Headers.

o EH path - an EH path starts at the node adding an EH and ends at the node that removes it.

3. Extension Header Basics

3.1. General Principles

Any EH capable node along an LSP may add an EH as long as it can be verified that there is another EH capable LSR downstream that can remove it. Any EH capable node downstream may remove an EH. An EH path starts when one or more EHs are added and ends where the last EH is removed. If there is no node downstream capable to remove the EH, it MUST NOT be added. It is assumed that a control plane will make this determination, the specification of which is outside the scope of this document.

In the context of the MPLS EH architecture an EH capable node assumes that all user packets on the default LSP carry EHs. As an optimization a second parallel LSP may be instantiated using a Forwarding Equivalence Class (FEC) that does not permit EHs, thus indicating to the LSR that there are no EHs in the packet.

3.2. LSPs in a EH capable Network

For an EH capable LSP between two EH capable LSRs there are two label mappings:

o first, a label mapping for the FEC that indicates that the packet carries IP

o second, a label mapping for a new FEC indicating that there are no EHs in the packet
3.3. EH capable nodes

EH capable nodes may process Extension Headers, i.e. add, augment, remove or do required processing at a transit node.

An EH capable node may not add an extension header to a packet if unless it is sure that there is a downstream node that can remove it.

If an LSP forks due to ECMP, the node that does the forking MUST be sure that all LSP branches (which may be re-merged) eventually terminate at an EH capable node which will remove the EH.

3.4. EH path and LSP

EH capable nodes may process Extension Headers, i.e. add, remove or do required processing at a transit node.

Figure 1 will be used for illustration.

A------b------c------D------E------F------G

A, D, E, F and G are EH capable nodes

b and c are non-EH capable nodes.

Figure 1: EH path vs. LSP

LSP - the LSP originates at ingress LSR A and terminates at egress LSR G, packets flow from A to G.

EHPl - EHPl originates with the EH capable node A adding an extension header to the packet and terminates when the EH capable node G removes the EH
EHP2 - EHP2 originates with the EH capable node A adding an extension header to the packet and terminates when the EH capable node E removes the EH. i.e. the EH path is shorter than the LSP.

EHP3 - EHP3 originates with the EH capable node D adding an extension header to the packet and terminates when the EH capable node G removes the EH.

EHP4 - EHP4 originates with the EH capable node D adding an extension header to the packet and terminates when the EH capable node F removes the EH, i.e. it is not necessary that an EH path originates or terminate on an MPLS LER.

EHP5 - EHP5 originates with the EH capable node F adding an extension header to the packet and terminates when the EH capable node G removes the EH.

Further discussion on the information needed in the packet to identify and process EHs are found in [I-D.song-mpls-extension-header].

3.5. Announcement of EH Capability

A node that is EH capable MUST have a way to announce this capability to other nodes in the same domain. Additions to the IGPs should be a baseline for such capabilities.

3.6. LSP establishment with LDP Downstream on Demand (DoD) in an EH capable network

LSPs for EH handling and processing in an MPLS network may be set up by LDP [RFC5036], a centralized controller and/or MPLS-SR. To enable this small extensions to the protocols are required.

In the examples in Section 3.6 and Section 3.7 we for simplicity assume that the payload of the packet is IP. It is of course possible that the payload will be a Pseudowire (PW) or a Virtual Private Network (VPN). This will be described in a later version of the document.

It is anticipated that the difference in establishment procedures for IP, PW and VPN will be minor.

It is possible to use the simplified physical topology show in Figure 2 which uses LDP Downstream on Demand (DoD) to illustrate how LSP setup work in a network with a mix of EH capable and EH non-capable nodes. In LDP DoD the action to set up an LSP is taken by the node at the head-end of the potential LSP.
A, D, E, and G are EH capable nodes

b is a non-EH capable node.

Figure 2: EH topology

The following steps would be taken assuming that node A wants to set up connectivity with node G to support EH handling and processing:

- A sends an LDP Label Request message to b, indicating that an EH capable LSP should be set up to G. A keeps track of the outstanding request.

- b is not EH capable and treat the Label Request as a normal request, however, the information indicating that an EH capable LSP is requested is transitive and sent to D.

- D receives the Label Request, forwards it to E, and keeps track of the outstanding request.

- E treats the label request the same way as D, and forward it to G.

- G receives the label request, finds out that it is the egress node for this LSP. G allocates two labels one for the IP FEC and one for the new "no EH present" FEC. G sends a label mapping to E with both labels, and asks E to PHP both LSPs.

- E receives the label mapping and installs PHP for both the IP FEC and for the new "no EH present"-FEC. E allocates two labels one for the IP FEC (label value 201) and one for the new FEC (label value 301). E sends a label mapping message to D, with the two labels.

- D receives the label mapping message and installs label 201 for the IP FEC and label value 301 for the new FEC. Since D know that b is not EH capable it will only allocate one label (202 for the IP FEC) and send a label mapping message to with that label.

- b receives the label mapping messages and installs label 202 for the IP FEC. Since b is not EH capable it will only allocate one
label (203 for the IP FEC). b sends a label mapping message to A with that label.

- A receives the label mapping and installs label value 203 for the IP FEC.

This will result in installed labels like this.

```
+---+     +---+     +---+     +---+     +---+
|   |...203...| ...202...| ...201...| ...php...|
A +-------+ b +-------+ D +-------+ E +-------+ G +
|   |     |   | ...301...|     |     |     |
| +---+     +---+     +---+     +---+     +---+
```

A, D, E and G are EH capable nodes.

b is a non-EH capable node.

Figure 3: EH topology

3.7. LSP establishment with LDP Downstream Unsolicited (DU) in an EH capable network

In LDP Downstream Unsolicited (DU) the initiative to establish a LSP is taken by the egress router. The egress will establish an LSP to every prefix it learns of from the IGP. With the exception from how the set up of the LSP(s) are triggered the label mappings are similar to how it is done with LDP DoD.

The same topology as in the LDP DoD example Figure 2 will be used for LDP DU.

- G learns that an EH capable LSP to egress LSR A is needed. G allocates two labels one for the IP FEC and one for the new "no EH present" FEC. G sends a label mapping to E with both labels, and asks E to PHP both LSPs.

- E receives the label mapping and installs PHP for both the IP FEC and for the new "no EH present"-FEC. E allocates two labels one for the IP FEC (label value 201) and one for the new FEC (label value 301). E sends a label mapping message to D, with the two labels.

- D receives the label mapping message and installs label 201 for the IP FEC and label value 301 for the new FEC. Since D know that
b is not EH capable it will only allocate one label (202 for the IP FEC) and send a label mapping message to with that label.

- b receives the label mapping messages and installs label 202 for the IP FEC. Since b is not EH capable it will only allocate one label (203 for the IP FEC). b sends a label mapping message to A with that label.

- A receives the label mapping and installs label value 203 for the IP FEC.

- This will result in the exact the same label mappings as in the DoD Example, see Figure 3.

### 3.8. Forwarding Behavior of EH Capable Nodes

A EH capable node will always search the label stack for EHs, with the exception of when a packet is received on the new FEC (no EH present).

Non-EH capable nodes will never search the label stack for EHs.

Given the configuration in Figure 3 packets will be forwarded as follows through the network.

If Node A sends a packet with an extension header following the label stack:

1. A sends a packet with label 203 with an EH after the label stack to b

2. b receives the packet and swaps the label to 202 and forward it to D.

3. D receives the packet, and since D is EH capable it will search the stack to find an EH-indicator. Since there is EH present, D will decide whether it should process the extension header or not. When that decision is taken and potential processing is done, D will swap the label to 201 and send it to E.

4. E receives the packet on LSP with a FEC that indicates that "EH may present" and will search the packet for an EH. When the EH is found by E it will, if required, process the EH, after that the top label is popped and the packet is forwarded to G.

5. G receives the packet, it will search the label stack to find the EHI. It will find the EH and since G is the egress node it will
do necessary processing and as a last step remove the EH. G will forward the packet based on the IP address.

If Node A sends a packet without an extension header:

1. A sends a packet with label 203 without an EH to b
2. b receives the packet and swaps the label to 202 and forward it to D.
3. D receives the packet, and since D is EH capable it will search the stack to find an EH. Since there is no EH present, D will swap the label to 301 and send it to E (FEC indicates no EH present).
4. E receives the packet on FEC "no EH present" and understand that it does not need to search the packet for an EH. E pops the label and forward to G
5. G receives the packet on FEC "no EH present" and understand that it does not need to search the packet for an EH. G will forward it based on the IP address.

3.9. EH for RSVP-TE tunnels

Extension Headers for RSVP-TE tunnels is for further study. Essentially it expected to be similar to the LDP case.

3.10. Ways to indicate an EH after the Label Stack

There are several ways to indicate the presence of EHs after the label stack. This will be discussed in a separate document.

4. EH in VPNs

TBA

5. EH and MPLS-SR

TBA

6. Extension Header Applications

TBA
7. EH distribution and EH capability announcement

TBA

8. Security Considerations

TBA

9. IANA Considerations

MPLS extension headers will require code point allocations from more than one IANA registry. It is not yet decided which document that will make which allocation.

However, tentatively the "No EH present" FEC will be assigned from this document.

IANA is requested to allocate lowest free value from the "IETF Review" range as new FEC from the "Forwarding Equivalence Class (FEC) Type Name Space" in the "Label Distribution Protocol (LDP) Parameters", like this:

<table>
<thead>
<tr>
<th>Value</th>
<th>Hex</th>
<th>Name</th>
<th>Label Distribution Discipline</th>
<th>Reference</th>
<th>Note/Reg. Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD</td>
<td>TBD</td>
<td>No EH present</td>
<td>DoD or DU</td>
<td>This Document</td>
<td>TBA</td>
</tr>
</tbody>
</table>

Table 1: No EH present

10. Acknowledgements

-

11. References

11.1. Normative References

[I-D.song-mpls-extension-header]
11.2. Informative References


Authors’ Addresses

Loa Andersson
Bronze Dragon Consulting

Email: loa@pi.nu

James N Guichard
Futurewei Technologies

Email: james.n.guichard@futurewei.com

Haoyu Song
Futurewei Technologies

Email: haoyu.song@futurewei.com

Stewart Bryant
University of Surrey

Email: stewart.bryant@gmail.com
MPLS Label Operations in MPLS EH capable networks
draft-andersson-mpls-eh-label-stack-operations-03

Abstract

Extension Headers (EH) carry information on in-network services and functions in an MPLS network. This document describes the operations on the MPLS label stack when an EH is found in the packet.

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 October 7, 2022.

Copyright Notice

Copyright (c) 2022 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 (https://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 ............................................. 2
   1.1.  Requirement Language ............................... 3
2.  Operations on an MPLS Label Stack in an EH capable network . 3
   2.1.  Physical Topology ................................. 3
   2.2.  A day in the life of a packet ......................... 5
       2.2.1.  Non-VPN Case ................................. 6
       2.2.1.1.  Non-VPN with the EH in the packet .......... 6
       2.2.1.2.  Non-VPN without an EH in the packet ...... 7
   2.3.  The VPN case ....................................... 8
       2.3.1.  VPN with EH in the packet .................... 8
       2.3.2.  VPN without EH in the packet ................ 9
   2.4.  RSVP-TE Tunnel case .................................. 10
       2.4.1.  RSVP Tunnel and EH present in the packet .... 11
       2.4.2.  RSVP Tunnel and no EH present in the packet . 12
       2.4.3.  EH capable RSVP-TE tunnel ..................... 13
3.  Security Considerations ...................................... 13
4.  IANA Considerations ........................................ 13
5.  Acknowledgements .......................................... 13
6.  References ................................................. 14
   6.1.  Normative References .............................. 14
   6.2.  Informative References ............................. 14
Authors’ Addresses ............................................. 14

1.  Introduction

This document provides the operating procedures for EH-capable and non-EH-capable LSRs where MPLS Extension Headers (EH) are carried below the MPLS label stack. Further we show that MPLS EHs can be gradually introduced into an existing MPLS network. The capability to handle EHs is announced throughout the MPLS network, and LSRs that don’t understand this information simply ignore it.

The extension headers are carried after the MPLS Label Stack, and the presence of EHs are indicate in the label stack by a Extended Special Purpose label called Extension Header Indicator (EHI) in the label stack.

Extension headers may for example be used when it is required that the packet carry some metadata, more details will be found in [I-D.song-mpls-extension-header]. Examples of such cases are In-situ OAM, Network Telemetry and Measurement and Network Security.
Only EH capable LSRs will process EHs, LSRs that are EH non-capable will ignore the EH and forward the packet as if the information was not there.

1.1. Requirement 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.

2. Operations on an MPLS Label Stack in an EH capable network

This document provides a set of examples to show the operations performed on MPLS encapsulated packets in a network where MPLS EHs are used. The document does also illustrated the procedures for processing of the information carried within the MPLS label stack to indicate the presence of EHs below the label stack. For the purpose of illustration, we will assume that the indicator used to point to EHs is a G-ACh Generic Associated Channel Label (GAL) [RFC5586] + G-Ach Associated Channel Header (ACH)[RFC5586] with a set of new ACH types to indicate the EH types carried below the MPLS label stack.

As discussed in [I-D.andersson-mpls-eh-architecture], [I-D.song-mpls-extension-header] and [I-D.song-mpls-eh-indicator] there are alternatives to the use of GAL as the indicator; for example an Extension Label (XL) [RFC7274] + one or more Extended Special Purpose Labels (eSPLs) [RFC7274] could also be used.

2.1. Physical Topology

Assume a physical topology that includes both EH capable LSRs and non-EH capable LSRs. The topology is intentionall kept quite simple.
Legend:
A, D, E, and F are EH capable LSRs
b and c are non-EH capable LSRs.

Figure 1: EH topology

LDP Downstream on Demand (DoD) or Downstream Unsolicited (DU), RSVP-TE, an IGP or a centralized controller could be used to create the label mappings between the LSRs in an EH capable network. Referring to Figure 1, and using LDP DU for illustration, creation of an EH path used by A to send MPLS encapsulated packets with MPLS EHs to F is as show below.

For prefix F reachable at LSR F:

- F advertises labels F:[ldp: implicit-null, EH-FEC: implicit-null] to E
- E advertises labels F:[ldp: 101, EH-FEC: 201] to D
- D advertises label F:[ldp: 102] to c
- c advertises label F:[ldp: 103] to b
- b advertises label F:[ldp: 104] to A

This will result in installed labels as shown in Figure 2.
Legend:
A, D, E and F are EH capable nodes.
b and c are non-EH capable nodes.

Figure 2: EH topology

2.2. A day in the life of a packet

This section provides examples of forwarding for some common scenarios in networks with a mix of EH-capable and non-EH-capable LSRs and packets with and without EHs following the MPLS label stack.

All the information processed in the examples below is not strictly a part of the "label stack": ACH, EHL, HEH, EH and Payload are carried after the last entry in the label stack.
For reference the following shows the full MPLS EH stack, i.e. including also the EH specific information and the payload.

```
0                             31
+---------------------------------+
| MPLS Label Stack                |
+---------------------------------+
| GAL (s-bit = 1)                 |
+---------------------------------+  * If eSPL then replace GAL with XL+EHL
| ACH Type = (EH E2E/HBH/BOTH)    |
+---------------------------------+  * If SPL then ACH not required; HEH follows XL+EHL directly
| Header of Extension Headers (HEH) |
+---------------------------------+
| Extension Header (EH) 1         |
+---------------------------------+
| Extension Header (EH) N         |
+---------------------------------+
| Upper Layer Protocols/Payload   |
+---------------------------------+
```

**Figure 3: MPLS Extension Header (EH) Stack**

2.2.1. Non-VPN Case

For non-VPN there are two variants, either the EH is present or it is not.

2.2.1.1. Non-VPN with the EH in the packet

- A sends packet to b
  
  * stack = [104, GAL, ACH, HEH, EH, IP]

- b is a legacy router so just swaps [104] to [103], and sends the packet to c
  
  * stack = [103, GAL, ACH, HEH, EH, IP]

- c is a legacy router so just swaps [103] to [102], and sends the packet to D
* stack = [102, GAL, ACH, HEH, EH, IP]

o D is an EH capable LSR and receives the packet with [102] on top of the stack; D scans the packet for an EH; D finds EH and processes and then swaps the top label to [101] and then sends the packet on to E

i Note: this goes on the standard FEC because we only announce in the packet there is NO EH. In this case EH is present.

* stack = [101, GAL, ACH, HEH, EH, IP]

o E receives [101] and scans the packet for EH; finds EH and processes and then pops the top label and send the packet to F

* stack = [GAL, ACH, HEH, EH, IP]

+ Note: E is the penultimate hop router so it pops the standard LDP label, and send on the standard FEC to F.

o F receives the packet and scans the packet for EH; finds EH and processes it. As F is the ultimate hop it pops GAL, and removes ACH, HEH and EH, processes IP and forwards the packet.

2.2.1.2. Non-VPN without an EH in the packet

In this example there is no EH present in the packet.

o A sends packet to b

* stack = [104, IP]

o b receives the packet, b is a legacy router so it just swaps [104] to [103] and sends the packet to c

* stack = [103, IP]

o c receives the packet, c is a legacy router so it just swaps [103] to [102], and sends the packet to D

* stack = [102, IP]

o D receives the packet, D is an EH capable router, D searches the packet for an EH but finds no EH, D swaps [102] to [201], and sends the packet to E

* stack = [201, IP]
+ Note: in this case D sends the packet using the EH-FEC as EH is *not* present.

+ Note: If downstream is not EH capable then D sends the packet on the standard FEC.

o E receives the packet [201] and bypasses EH processing (received on the "no EH present" FEC; E is penultimate node so it pops EH-FEC label; and sends the packet to F.

* stack = [IP]; not exactly a "label stack", but listed here for symmetry

o F receives [IP] and routes the packet

2.3. The VPN case

In these two examples there is VPN information in the label stack, in the first there also EHs in the packet.

2.3.1. VPN with EH in the packet

o A sends packet to b

* stack = [104, VPN, GAL, ACH, HEH, EH, IP]

o b receives the packet; b is a legacy router and just swaps [104] to [103] and sends the packet to c

* stack = [103, VPN, GAL, ACH, HEH, EH, IP]

o c receives the packet; c is a legacy router and just swaps [103] to [102] and sends the packet to D

* stack = [102, VPN, GAL, ACH, HEH, EH, IP]

o D receives the packet; D is EH capable LSR; D will search the packet for EH and will find and process the EH; D will then swap [102] to [101] and sends the packet to E

* stack = [101, VPN, GAL, ACH, HEH, EH, IP]

+ Note: This packet will be sent normal IP standard FEC; only packets that do not include an EH will be sent on the "no EH present" FEC.
E receives the packet; E is EH capable LSR; E will search the packet for EH and will find and process the EH; E will then pop [101] and sends the packet to F

* stack = VPN, GAL, ACH, HEH, EH, IP

+ Note: E is penultimate hop so pops the LDP label and send the packet on normal IP standard FEC; only packets that does not include an EH will be sent on the "no EH present" FEC.

F receives and scans the packet for EH; finds EH and processes it. As F is the ultimate hop it pops the GAL, and removes ACH, HEH and EH, processes the VPN label and forwards the packet.

2.3.2. VPN without EH in the packet

A sends packet to b

* stack = [104, VPN, IP]

b receives the packet; b is a legacy router and just swaps [104] to [103] and sends the packet to c

* stack = [103, VPN, IP]

c receives the packet; c is a legacy router and just swaps [103] to [102] and sends the packet to D

* stack = [102, VPN, IP]

D receives the packet; D is EH capable LSR; D will search the packet for EH; D will not find an EH; D will then swap [102] to [201] and sends the packet to E on the "no EH present" FEC. Loa

* stack = [101, VPN, IP]

+ Note: This packet will be sent on the "no EH present" FEC;

E receives the packet [201] and bypasses EH processing (received on the "no EH present" FEC; E is the penultimate node so it pops EH- FEC label; and send the packet to F on the "no EH present" FEC.

* stack = [VPN, IP]

+ Note: E is penultimate hop so E pops the "no FEC present" label and send the packet to F.
2.4. RSVP-TE Tunnel case

The RSVP-TE tunnel is not EH capable or the capability has been disabled.

Assume a physical topology that includes both EH capable LSRs and non-EH capable LSRs, as in the earlier examples. This topology also includes a low cost RSVP-TE tunnel between b and D.

![Diagram of EH topology]

Legend:
A, D, E, and F are EH capable LSRs
b and c are non-EH capable LSRs.
Nodes that transport the RSVP-TE tunnel are not EH capable, or the EH capability is disabled.

Figure 4: EH topology

For this example the following assumptions are made:

- An RSVP-TE tunnel has been established between b and D (packets will bypass c)
- F is reachable at b through RSVP-TE tunnel
- LDP is enabled on the RSVP-TE tunnel

For prefix [F]: The following label mappings are sent by the LSRs in the network.
F advertises labels F: [ldp: implicit-null, EH-FEC: implicit-null] to E.

E advertises labels F: [ldp: 101, EH-FEC: 201] to D.


c advertises label F: [ldp: 103] to b.

b advertises label F: [ldp: 104] to A.

This will result in label mappings like this.

```
+---+       +---+       +---+       +---+       +---+       +---+
|   |       |   |       |   |       |   |..101..|   |..php..|   |
| A +-------+ b +-------+ c +-------+ D +-- ----+ E +-------+ F +
+-+++ - ++++ ++++ ++++ ++++ ++++
```

Legend:
A, D, E, and F are EH capable LSRs.
b and c are non-EH capable LSRs.
Nodes that transport the RSVP-TE tunnel are not EH capable, or the EH capability is disabled. [RSVP] represents the series of tunnel top lables.

Figure 5: EH topology

To describe the label stack operations in this case the VPN label stack is used, starting with the case where an EH is present in the packet.

2.4.1. RSVP Tunnel and EH present in the packet

A sends packet to b

stack = [104, VPN, GAL, ACH, HEH, EH, IP]
o b receives the packet, since b is a legacy router it swaps [104] to [102], the next-hop reachable through the RSVP-TE tunnel; push the ingress RSVP-TE tunnel label and send it via the tunnel to the tunnel endpoint D

stack = [RSVP, 102, VPN, GAL, ACH, HEH, EH, IP]

o Intermediate tunnel LSRs will forward (swap) based on the RSVP-TE label.

o D receives the packet, D will pop the last RSVP-TE label; since D is a EH capable router it will search the stack and find the EH, after processing the EH it will swap [102] to [101], and send the packet to E over the normal FEC

stack = [101, VPN, GAL, ACH, HEH, EH, IP]

Note: this will be forwarded on the standard FEC because since the EH is present in the packet, only packet without an EH is forwarded on the "no EH present" FEC.

o E receives the packet [101]; since E is a EH capable router it will search the stack and find the EH; after processing the EH it will pop [101], and send the packet to E over the normal FEC

stack = [VPN, GAL, ACH, HEH, EH, IP]

Note: As E is the penultimate hop it will pop the standard LDP label.

o F receives the packet with the VPN label on top [VPN]; E scans the packet for EH; finds EH and processes. As F is the ultimate hop it pops GAL, and removes ACH, HEH and EH, processes VPN label and forwards the packet.

2.4.2. RSVP Tunnel and no EH present in the packet

o A sends packet to b

* stack = [104, VPN, IP]

o b receives the packet [104]; be is legacy router and will not search for an EH; b swaps [104] to [102]; pushes [RSVP] sends packet to D over the RSVP-TE tunnel.

* stack = [RSVP, 102, VPN, IP]
Intermediate tunnel LSRs will forward (swap) based on the RSVP-TE label.

- D receives pops the tunnel label [RSVP], D is EH capable and scans the packet for EH; D finds no EH is present; pops RSVP-TE label, and then swaps LDP label [102] to [201] and sends the packet to E
  * stack = [201, VPN, IP]
    + Note: in this case D sends the packet using the "no EH present" FEC, since there is no EH in the packet.
    + Note: If the downstream LSR is not EH capable then D will sends the packet on the standard FEC.

- E receives [201] and bypasses EH processing since the packet is received on the "no EH present" FEC; E is the pen-ultimate hop so it EH-FEC label and forward the packet to F
  * stack = [VPN, IP]

- F receives the packet [VPN]; and scans the packet for EH; does not find EH, processes VPN label and forwards the packet.

2.4.3. EH capable RSVP-TE tunnel

The case where an RSVP-TE tunnel is both EH capable and EH enabled is for further study.

3. Security Considerations

TBA

4. IANA Considerations

There are no requests for IANA actions in this document.

Note to the RFC Editor - this section can be removed before publication.

5. Acknowledgements

TBA

6. References

6.1. Normative References

[I-D.andersson-mpls-eh-architecture]
Andersson, L., Guichard, J. N., Song, H., and S. Bryant,
"MPLS Extension Header Architecture", draft-andersson-
mpls-eh-architecture-02 (work in progress), October 2021.

[I-D.song-mpls-eh-indicator]
Song, H., Li, Z., Zhou, T., and L. Andersson, "Options for
MPLS Extension Header Indicator", draft-song-mpls-eh-
indicator-04 (work in progress), January 2022.

[I-D.song-mpls-extension-header]
Song, H., Li, Z., Zhou, T., Andersson, L., and Z. Zhang,
"MPLS Extension Header", draft-song-mpls-extension-
header-06 (work in progress), January 2022.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,

"MPLS Generic Associated Channel", RFC 5586,
DOI 10.17487/RFC5586, June 2009,

6.2. Informative References

and Retiring Special-Purpose MPLS Labels", RFC 7274,
DOI 10.17487/RFC7274, June 2014,

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,

Authors' Addresses

Loa Andersson
Bronze Dragon Consulting

Email: loa@pi.nu
James N Guichard
Futurewei Technologies
Email: james.n.guichard@futurewei.com

Haoyu Song
Futurewei Technologies
Email: haoyu.song@futurewei.com

Stewart Bryant
University of Surrey
Email: stewart.bryant@gmail.com
MPLS Working Group                                        R. Gandhi, Ed.
Internet-Draft                                                    Z. Ali
Updates: 5586 (if approved)                                 F. Brockners
Intended status: Standards Track                     Cisco Systems, Inc.
Expires: 3 September 2022                                         B. Wen
Comcast                                                      B. Decraene
B. Decraene                                                   Orange
V. Kozak                                                      Comcast
Comcast                                                      2 March 2022

MPLS Data Plane Encapsulation for In-situ OAM Data
draft-gandhi-mpls-ioam-04

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) is used
for recording and collecting operational and telemetry information
while the packet traverses a path between two points in the network.
This document defines how IOAM data fields are transported with MPLS
data plane encapsulation using new Generic Associated Channel (G-ACh)
and updates the RFC 5586.

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 3 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the
document authors. All rights reserved.
1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) is used for recording and collecting operational and telemetry information while the packet traverses a path between two points in the network. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within the probe packets specifically dedicated to OAM. The IOAM data fields are defined in [I-D.ietf-ippm-ioam-data]. The IOAM data fields are...
This document defines how IOAM data fields are transported with MPLS data plane encapsulations using new Generic Associated Channel (G-ACh) and updates the [RFC5586].

2. Conventions

2.1. Requirement 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] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Abbreviations

Abbreviations used in this document:

- IOI  IOAM Indicator
- ECMP  Equal Cost Multi-Path
- E2E  Edge-To-Edge
- EL  Entropy Label
- ELI  Entropy Label Indicator
- ELC  Entropy Label Control
- G-ACh  Generic Associated Channel
- HBH  Hop-By-Hop
- HBI  Hop-By-Hop Indicator
- IOAM  In-situ Operations, Administration, and Maintenance
3. MPLS Extensions for IOAM Data Fields

3.1. IOAM Generic Associated Channel

The IOAM header is added containing different IOAM-Data-Fields in the MPLS header as shown in Figure 1. The IOAM-Data-Fields MUST follow the definitions corresponding to IOAM-Option-Types (e.g. see Section 5 of [I-D.ietf-ippm-ioam-data] and Section 3.2 of [I-D.ietf-ippm-ioam-direct-export]). More than one trace options can be present in the IOAM-Data-Fields.

G-ACh [RFC5586] provides a mechanism to transport OAM and other control messages over MPLS data plane. The IOAM G-ACh header [RFC5586] with new IOAM G-ACh type MUST be added immediately after the MPLS label stack in the MPLS header as shown in Figure 1, before the IOAM-Data-Fields. The G-ACh label (GAL) [RFC5586] MUST NOT be added in the MPLS label stack.

This document updates the following paragraph in Section 2.1 of [RFC5586]: "The G-ACh MUST NOT be used to transport user traffic" to "The G-ACh MAY be used with user traffic to transport OAM information".

Note that the G-ACh is not really used to transport the user traffic in this document but to transport the IOAM-Data-Fields with the user traffic.
The IOAM-Data-Fields are encapsulated using the following fields in the MPLS header:

IP Version Number 0001b: The first four octets are IP Version Field part of a G-ACh header, as defined in [RFC5586].

Version: The Version field is set to 0, as defined in [RFC4385].

Length: Length of IOAM G-ACh data in 4-octet units. Note that this field is marked as Reserved in [RFC5586] and is updated for the new IOAM G-ACh type by this document.

IOAM G-ACh: Generic Associated Channel (G-ACh) Type (value TBA1) for IOAM [RFC5586].

Reserved: Reserved Bits MUST be set to zero upon transmission and ignored upon receipt.

Block Number: The Block Number can be used to aggregate the IOAM data collected in data plane, e.g. to compute measurement metrics for each block of a data flow. It is also used to correlate the IOAM data on different nodes.

IOAM-OPT-Type: 8-bit field defining the IOAM Option type, as defined
in the "IOAM Option-Type Registry" specified in [I-D.ietf-ippm-ioam-data].

IOAM HDR Length: 8-bit unsigned integer. Length of the IOAM Header in 4-octet units.

IOAM Option and Data Space: IOAM-Data-Fields as specified by the IOAM-OPT-Type field. IOAM-Data-Fields are defined corresponding to the IOAM-Option-Type (e.g. see Section 5 of [I-D.ietf-ippm-ioam-data] and Section 3.2 of [I-D.ietf-ippm-ioam-direct-export]).

3.2. IOAM Indicators

An IOAM Indicator MUST be used to indicate the presence of the IOAM-Data-Fields in the MPLS header. If both edge and intermediate nodes need to process IOAM data then both IOAM Indicator and HBH Indicator MUST be used. The HBH Indicator allows to optimize the IOAM processing on intermediate nodes and avoids the need to check all IOAM-Data-Fields.

A flag called IOI (IOAM Indicator) in the TTL of the X-Label is defined in this document to indicate the presence of IOAM. A flag called HBI (Hop-By-Hop Indicator) in the TTL of the X-Label is defined to indicate that HBH processing is required. The bit positions of these flags in the TTL field can be user-defined, consistently in the network. Alternatively, the bit positions of these flag can be allocated by IANA.

The X-Label can be a Special Purpose Label (value TBA1) assigned by IANA or a Network Programming Label (NPL) provisioned by a user [I-D.jags-mpls-ext-hdr] or an Entropy Label [I-D.decrane-mpls-slid-encoded-entropy-label-id].

4. Edge-to-Edge IOAM

4.1. IOAM Indicator

The IOAM Indicator is used to indicate the presence of the IOAM-Data-Fields in the MPLS header as shown in Figure 2.
The E2E IOAM-Data-Fields carry the Option-Type(s) that require processing on the encapsulating and decapsulating nodes only. The IOAM Option-Type carried can be IOAM Edge-to-Edge Option-Type [I-D.ietf-ippm-ioam-data]. The E2E IOAM-Data-Fields SHOULD NOT carry any IOAM Option-Type that require IOAM processing on the intermediate nodes as it will not be processed by them.

4.2. Procedure for Edge-to-Edge IOAM

The E2E IOM procedure is summarized as following:

* The encapsulating node inserts the X-Label with the IOAM Indicator (Flag IOI) below the label whose FEC is the end (decapsulating) node and one or more IOAM-Data-Fields in the MPLS header.

* The intermediate nodes do not process IOAM-Data-Fields.

* The penultimate node MUST NOT remove the MPLS header. This is ensured by the encapsulating node by adding required MPLS header.

* The decapsulating node MAY punt a copy of the packet with the receive timestamp to the slow path for IOAM-Data-Fields processing when the node recognizes the IOAM Indicator. The receive timestamp is required by the various E2E OAM use-cases, including streaming telemetry. Note that the packet is not necessarily punted to the control-plane.

* The decapsulating node processes the IOAM-Data-Fields using the procedures defined in [I-D.ietf-ippm-ioam-data]. An example of IOAM processing is to export the IOAM-Data-Fields, send IOAM-Data-Fields via streaming telemetry, etc.
* The decapsulating node MUST remove the IOAM-Data-Fields from the received packet. The decapsulated packet is forwarded downstream or terminated locally similar to the regular IOAM-Data-Fields.

5. Hop-By-Hop IOAM

5.1. Hop-By-Hop Indicator

The IOAM Indicator (Flag IOI) along with Hop-By-Hop Indicator (Flag HBI) are used to indicate the presence of the HBH IOAM-Data-Fields in the MPLS header as shown in Figure 3.

```
+------------------------------------------+   +------------------------------------------+
|  Label                                |   | TC |S|  TTL                                          |
+------------------------------------------+   +------------------------------------------+
|                                          |   | 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 |
+------------------------------------------+   +------------------------------------------+
```

Figure 3: Example MPLS Encapsulation for HBH IOAM

The HBH IOAM-Data-Fields carry the Option-Type(s) that require processing at the intermediate and/or encapsulating and decapsulating nodes. The IOAM Option-Type carried can be IOAM Pre-allocated Trace Option-Type, IOAM Incremental Trace Option-Type and IOAM Proof of Transit (POT) Option-Type, as well as Edge-to-Edge Option-Type [I-D.ietf-ippm-ioam-data].

5.2.Procedure for Hop-By-Hop IOAM

The HBH IOAM procedure is summarized as following:

* The encapsulating node inserts the X-Label with the IOAM Indicator (Flag IOI) and HBH Indicator (Flag HBI) below the label whose FEC is the end (decapsulating) node and one or more IOAM-Data-Fields in the MPLS header.

* The intermediate node enabled with HBH IOAM function processes the data packet including the IOAM-Data-Fields as defined in [I-D.ietf-ippm-ioam-data] when the node recognizes the HBH Indicator in the MPLS header.
The intermediate node MAY punt a copy of the packet with the receive timestamp to the slow path for IOAM-Data-Fields processing when the node recognizes the HBH indicator. The receive timestamp is required by the various HBH OAM use-cases, including streaming telemetry. Note that the packet is not necessarily punted to the control-plane.

The intermediate node forwards a copy of the processed data packet downstream.

The penultimate node MUST NOT remove the MPLS header. This is ensured by the encapsulating node by adding required MPLS header.

The processing on the decapsulating node is same as E2E case.

6. Considerations for IOAM

6.1. Considerations for ECMP

The encapsulating node needs to make sure the IOAM-Data-Fields do not start with a well-known IP Version Number (e.g. 0x4 for IPv4 and 0x6 for IPv6) as that can alter the hashing function for ECMP that uses the IP header. This is achieved by using the IOAM G-ACh with IP Version Number 0001b after the MPLS label stack [RFC5586].

When entropy label [RFC6790] is used for hashing function for ECMP, the procedure defined in this document does not alter the ECMP behaviour.

6.2. Node Capability

The decapsulating node that has to remove the IOAM-Data-Fields and perform the IOAM function may not be capable of supporting it. The encapsulating node needs to know if the decapsulating node can support the IOAM function. The signaling extension for this capability exchange is outside the scope of this document.

The intermediate node that is not capable of supporting the IOAM functions defined in this document, can simply skip the IOAM processing.

6.3. Nested MPLS Encapsulation

When a packet is received with IOAM, the nested MPLS encapsulating node that supports a different IOAM, the node MUST add a new X-Label with the supported IOAM as part of the new MPLS encapsulation.
7. Security Considerations

The security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data] and apply to the procedure defined in this document.

IOAM is considered a "per domain" feature, where one or several operators decide on configuring IOAM according to their needs. IOAM is intended for deployment in limited domains [RFC8799]. As such, it assumes that a node involved in IOAM operation has previously verified the integrity of the path. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into the domain.

Routers that support G-ACh are subject to the same security considerations as defined in [RFC4385] and [RFC5586].

8. IANA Considerations

IANA maintains G-ACh Type Registry (see https://www.iana.org/assignments/g-ach-parameters/g-ach-parameters.xhtml). IANA is requested to allocate a value for IOAM G-ACh Type from "MPLS Generalized Associated Channel (G-ACh) Types (including Pseudowire Associated Channel Types)" registry.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBA1</td>
<td>IOAM G-ACh Type</td>
<td>This document</td>
</tr>
</tbody>
</table>

Table 1: IOAM G-ACh Type

9. Appendix

9.1. MPLS Encapsulation with Control Word and Another G-ACh for IOAM Data Fields

The IOAM-Data-Fields, including IOAM G-ACh header are added in the MPLS encapsulation immediately after the MPLS header. Any Control Word [RFC4385] or another G-ACh [RFC5586] MUST be added after the IOAM-Data-Fields in the packet as shown in the Figure 6 and Figure 7, respectively. This allows the intermediate nodes to easily access the HBH IOAM-Data-Fields located immediately after the MPLS header. The decapsulating node can remove the MPLS encapsulation including the IOAM-Data-Fields and then process the Control Word or another G-ACh following it. The subsequent G-ACh and Control Word are
located through the use of the "Length" field in the IOAM G-ACh.

Figure 4: Example MPLS Encapsulation with Generic PW Control Word with HBH IOAM
Figure 5: Example MPLS Encapsulation with Another G-ACh with HBH IOAM

10. References

10.1. Normative References

[I-D.ietf-ippm-ioam-data]

[I-D.ietf-ippm-ioam-direct-export]
10.2. Informative References


Acknowledgements

The authors would like to thank Patrick Khordoc, Sagar Soni, Shwetha Bhandari, Clarence Filsfils, and Vengada Prasad Govindan for the discussions on IOAM. The authors would also like to thank Tarek Saad, Loa Andersson, Greg Mirsky, Stewart Bryant, Xiao Min, and Cheng Li for providing many useful comments. The authors would also like to thank Mach Chen, Andrew Malis, Matthew Bocci, and Nick Delregno for the MPLS-RT reviews.

Authors’ Addresses

Rakesh Gandhi (editor)
Cisco Systems, Inc.
Canada
Email: rgandhi@cisco.com

Zafar Ali
Cisco Systems, Inc.
Email: zali@cisco.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany
Email: fbrockne@cisco.com

Bin Wen
Comcast
Email: Bin_Wen@cable.comcast.com

Bruno Decraene
Orange
Email: bruno.decrane@orange.com

Voitek Kozak
Comcast
Email: Voitek_Kozak@comcast.com
Deprecating the Use of Router Alert in LSP Ping

draft-kompella-mpls-lsppping-norao-00

Abstract

LSP ping messages (RFC 8029) are encapsulated in IP headers that include a Router Alert Option (RAO). The rationale for including an RAO is questionable. Furthermore, RFC6398 identifies security vulnerabilities associated with the RAO.

Therefore, this document removes the RAO from LSP ping message encapsulations. It updates RFCs 7506 and 8029.

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 12 June 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

LSP ping [RFC8029] detects data-plane failures in MPLS Label Switched Paths (LSPs). It can operate in "ping mode" or "traceroute mode". When operating in ping mode, it verifies end-to-end LSP continuity. When operating in traceroute mode, it can localize failures to a particular node along an LSP.

LSP ping defines a probe message, called the "MPLS echo request". It also defines a response message, called the "MPLS echo reply". Both messages are encapsulated in UDP and IP. The echo request message is further encapsulated in an MPLS label stack.

When operating in ping mode, LSP ping sends a single echo request message, with the MPLS TTL set to a high value (e.g., 255). This message is intended to reach the egress Label Switching Router (LSR). When operating in traceroute mode, MPLS ping sends multiple echo request messages. It manipulates the MPLS TTL so that the first message expires on the first LSR along the path and subsequent messages expire on subsequent LSRs.
The IP header that encapsulates an echo request message must include a Router Alert Option (RAO), while the IP header that encapsulates an echo reply message may include an RAO. In both cases, the rationale for including an RAO is questionable. Furthermore, [RFC6398] identifies security vulnerabilities associated with the RAO and recommends against its use outside of controlled environments.

Therefore, this document removes the RAO from both LSP ping message encapsulations. It updates RFCs 7506 [RFC7506] and 8029.

1.1. Terminology

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.

LSP: Label Switched Path

LSR: Label Switching Router

RAO: Router Alert Option

2. Router Alert for LSP Ping (RFC 8029)

2.1. Echo Request

While the MPLS echo request message must traverse every node in the LSP under test, it must not traverse any other node. Specifically, the message must not be forwarded beyond the egress Label Switching Router (LSR).

To achieve this, RFC 8029 proposes the following:

1. When the echo request message is encapsulated in IPv4, the IPv4 destination address must be chosen from the subnet 127/8. When the echo request message is encapsulated in IPv6, the IPv6 destination address must be chosen from the subnet 0:0:0:0:FFFF:7F00:0/104.

2. When the echo request message is encapsulated in IPv4, the IPv4 TTL must be equal to 1. When the echo request message is encapsulated in IPv6, the IPv6 Hop Limit must be equal to 1.
3. When the echo request message is encapsulated in IPv4, the IPv4 header must include an RAO. When the echo request message is encapsulated in IPv6, the IPv6 header chain must include a Hop-by-hop extension header and the Hop-by-hop extension header must include an RAO.

Currently, ALL of these are required. However, any one is sufficient to prevent forwarding the packet beyond the egress LSR.

Therefore, this document RECOMMENDS removing Requirement 3 from RFC 8029.

The authors are not aware of any implementation that relies on the RAO to prevent packets from being forwarded beyond the egress LSR.

2.2. Echo Reply

An LSP ping replies to the MPLS echo message with an MPLS echo reply message. It has four reply modes:

1. Do not reply
2. Reply via an IPv4/IPv6 UDP packet
4. Reply via application-level control channel

The rationale for mode 3 is questionable, if not wholly misguided. According to RFC 8029, "If the normal IP return path is deemed unreliable, one may use 3 (Reply via an IPv4/IPv6 UDP packet with Router Alert)."

However, it is not clear that the use of the RAO increases the reliability of the return path. In fact, one can argue it decreases the reliability in many instances, due to the additional burden of processing the RAO. This document RECOMMENDS removing mode 3 from RFC 8029.

The authors are not aware of any implementations of mode 3.

3. Update to RFC 7506

RFC 7506 defines the IPv6 Router Alert Option for MPLS Operations, Administration, and Management. This document RECOMMENDS that RFC 7506 be reclassified as Historic.
4. Backwards Compatibility

LSP Ping implementations SHOULD ignore RAO options when they arrive on incoming echo request and echo reply messages.

5. IANA Considerations

If this document is approved, mark the IPv6 RAO value of MPLS OAM (69) in [IANA-IPV6-RAO] as "Deprecated".

Also, mark Reply Mode 3 ("Reply via an IPv4/IPv6 UDP packet with Router Alert") in [IANA-LSP-PING] as "Deprecated".

6. Security Considerations

The recommendations this document makes do not compromise security.

7. Normative References

[IANA-IPV6-RAO]
IANA, "IPv6 Router Alert Option Values", n.d.,
<https://www.iana.org/assignments/ipv6-routeralert-values>.

[IANA-LSP-PING]
IANA, "Multiprotocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters", n.d.,

[RFC2119]  Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119,
DOI 10.17487/RFC2119, March 1997,

DOI 10.17487/RFC6398, October 2011,

[RFC7506]  Raza, K., Akiya, N., and C. Pignataro, "IPv6 Router Alert Option for MPLS Operations, Administration, and Maintenance (OAM)", RFC 7506,
DOI 10.17487/RFC7506, April 2015,


Authors’ Addresses

Kireeti Kompella
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
United States

Email: kireeti.ietf@gmail.com

Ronald Bonica
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
United States

Email: rbonica@juniper.net
Abstract

The intention of this document is to enumerate and describe the candidate schemes that can be used to indicate the presence of the MPLS extension header(s) following the MPLS label stack. After a careful evaluation of these options by comparing their pros and cons, it is expected that one should be chosen as the final standard scheme for MPLS extension header indicator.

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.

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 7 July 2022.
1. Introduction

The document [I-D.song-mpls-extension-header] presents the motivation, specification, and use cases of MPLS Extension Header (EH). An indicator is needed in the MPLS label stack to indicate the presence of the extension header(s). Multiple options are possible for this purpose. As the discussion progresses, more options could emerge.

In this document, we propound three categories of methods which can be further partitioned into five unique schemes. Four of them use explicit data plane encoding to indicate the EH and the last one
implies the EH through control plane configuration. This document details and compares these schemes, in order to foster further discussions until a final decision is made.

2. Dedicated Extension Header Label

A straightforward method is to directly encode an Extension Header Label (EHL) in the MPLS label stack. Two derived schemes are as follows.

2.1. Special Purpose Label

A new special purpose label, EHL, can be used to indicate EHs. As specified in [RFC7274], so far eight special purpose label values are still left unsigned by IANA (which are 4 to 6 and 8 to 12). This single label scheme is elegant but arguably demands a scarce resource. We cannot rule out the possibility of requiring more than one label value to differentiate EH classes (e.g., Hop-by-Hop, End-to-End, or both). If this happens, it can only aggravate the situation.

Another benefit of this scheme is that an EHL can potentially be located anywhere in an MPLS label stack. It is easier and quicker for a router to figure out the existence of extension header(s) if the EHL is close to or at the top of the label stack. However, if there are legacy devices which can reach the EHL but do not recognize it in a network, then for backward compatibility, the EHL must be located at the bottom of the stack (i.e., only the MPLS tunnel ends and EHL-aware nodes will look up and process it).

The format of an EHL is the same as an MPLS label. The first 20-bit label value will be assigned by IANA. The BoS bit is used to indicate the location of the label. The other fields, CoS and TTL, currently have no use in the context of EHL. However, these two fields can potentially be used to encode other information. If such code points are open for other purpose, it will make the single EHL idea more compelling. E.g., the EH category and/or other information, if needed, can be encoded in these fields, so that only one special label value is needed.

The following figure shows a potential scheme in which one bit from the CoS field (’H’) is used to indicate the presence of HbH EHs in the packet. If ’H’ bit is 0, it means no HbH EH follows so a P-router will not need to check the EH. The last 8 bits can be used to find the location of the extension headers (i.e., the first byte after the MPLS label stack). This information can help to avoid the scan of the label stack in case the extension headers need to be accessed.
Note that the Cos/TTL fields can be encoded to include more information. For example, in addition to indicate the EH, it can also indicate the presence of some other label-based services (e.g., EL). If we want to explore such possibilities, we have 11 bits in total at our disposal.

2.2. Extension Label plus an Extended Special Purpose Label

[RFC7274] specifies the Extension Label (XL) with the value of 15. An extended special purpose label (ESPL) following XL can be used as EHL. A large number of ESPL values are available for allocation. The XL+EHL scheme eases the concern on the reserved label space at the cost of one more label in the label stack.

Except for the fact that one more label is needed, the XL+EHL scheme shares the same property as the single special purpose EHL scheme.

3. Generic Associated Channel Extension

The similar "header extension" requirement for MPLS has led to some proposals before. A special Generic Associated Channel Label (GAL) [RFC5586] with the value of 13 has been assigned to support the identification of an Associated Channel Header (ACH). We can extend this existing mechanism to encode the MPLS EH indicator.

3.1. GAL and Associated Channel Header

The ACH is located below the bottom label. It has a 16-bit Channel Type field which provides abundant space to encode the MPLS EH indicator. This scheme has the same header overhead as the XL+EHL scheme. The format is depicted in Figure 2.
GAL has several applications already yet its heritage also has several limitations. The GAL must be located at the bottom of a label stack for its chief use cases such as MPLS-TP. So a router needs to search the entire label stack for the BoS bit and check if the corresponding label is GAL. This can impact the performance when the label stack is deep. A more serious concern is that [RFC5586] states that GAL+ACH MUST NOT be used to transport user traffic and an ACH is supposed to be followed by a non-service payload.

None of these is insurmountable but it does require an overhaul of the existing RFC in order to extend the usage of GAL.

3.2. GAL and a Different Nibble Value

To avoid changing the established semantics of ACH, a variation can be used. ACH starts with a nibble value "0001". A different nibble value may be used to redefine the remaining part of the word. The idea has been exploited by [I-D.guichard-sfc-mpls-metadata] to define a Metadata Channel Header (MCH) with the leading nibble value "0000". Similarly, we can use another nibble value (e.g., "0010") to define a new header, namely the MPLS Extension Header Indicator (EHI).

The format of the GAL and EHI is depicted in Figure 3.
The Extension Header Class field in EHI is used to differentiate the extension headers. Potentially there are three classes: Hop-by-Hop (HbH), End-to-End (E2E), or both. If finally we decide to not differentiate the extension headers, we have the opportunity to merge the HEH (see [I-D.song-mpls-extension-header] for details) into EHI, so we can reduce the header overhead by four bytes. The header format is depicted in Figure 4.

4. Extend MPLS Entropy Label

Instead of introducing a new SPL as the EH indicator, we can piggyback the indicator in some existing SPL to avoid claiming extra SPL resource and save a label overhead. The best candidate is the entropy label (EL) [RFC6790]. If we can make EL default for every MPLS packet, we can encode the EH indicator in the unused ELI/EL label fields such as CoS and TTL.
In Figure 5 we show a possible encoding method, in which the first bit of the CoS field in ELI is used to indicate the presence of EH and the TTL field in ELI can be used to indicate the location of the EH. Note that the CoS field of the EL can also be used to encode other information, if necessary.

```
          0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                ELI (7)                |I|   |S|    Offset     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 EL                    |     |S|       0       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Special EHL with EH Category Encoding

5. Configured FEC Labels

It is also possible to use FEC labels to indicate the presence of extension headers. An FEC label has the same forwarding semantics as the original label, but it also means that one or more extension headers exist below the label stack.

Although this approach avoids the need of new header encoding standards, it introduces a good deal of complexity into the control plane. Since every label needs an FEC label to indicate EH, this scheme also significantly reduces the available label space. Another issue is that this solution may not work for incremental deployment where some legacy routers cannot understand and apply the FEC labels for EH. Moreover, this configuration-based solution certainly makes the cross-domain interoperability more difficult. Hence, this is the least preferred option. We only include it here for the completeness of the discussion.

6. Summary

Evidenced by the existing and emerging use cases, MPLS networks need a standard way to support extension headers. In Figure 6, we summarize the potential schemes that allow MPLS packets to carry extension headers and list the main pros and cons for each scheme.
<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>Pros and Cons</th>
</tr>
</thead>
</table>
| 1   | Special purpose EHL                             | + Single label  
+ Location freedom  
- Need standard extension  
- Use scarce resource |
| 2   | XL(15) + EHL                                     | + Location freedom  
+ Established mechanism  
+ Abundant resource  
- One extra label than Option 1  
- Need standard extension |
| 3   | GAL + ACH with channel type extension            | + Reuse existing mechanism  
+ Abundant resource  
- Label location limitation  
- One more word than Option 1  
- Not for user traffic  
- Need standard extension/update |
| 4   | GAL + another nibble value to indicate EHs      | + No change to ACH semantics  
+ Potential overhead saving  
- Label location limitation  
- Hack scarce resource (nibble)  
- Need standard extension |
|     | (e.g., "0010")                                  |                                                                               |
| 5   | Extend ELI/EL                                    | + No need for new label  
- Need standard update  
- Need to make EL mandatory  
- One extra label than Option 1 |
| 6   | FEC label as EH indicator                       | + No need for header standard  
- Complex control plane  
- Cross-domain interoperability  
- Label space issue  
- Not for incremental deployment |

Figure 6: Potential Schemes for MPLS Extension Headers

Basically we have three groups of solutions. The scheme 1 and 2 introduce new labels, the scheme 3, 4, and 5 extend the existing solutions, and the scheme 6 relies on the control plane. Through comprehensive considerations on the pros and cons of each scheme, we expect a working group consensus can be reached to pick the final winner.
7. Considerations of EHI

The existence of Extension Headers will make the ECMP based on inner IP packet header impossible or harder. If legacy routers need to conduct this kind of ECMP, the process either fails or generates unexpected results. EH-aware routers can do this kind of ECMP but they need to skip all the EHs in order to access the inner packet header which may not be efficient (we make provision in HEH to help accelerate this process). In this case, the Entropy Label (EL) is preferred for ECMP. The Entropy Label Indicator (ELI) and EL should be put in front of the EHI to avoid confusing the legacy routers.

8. Security Considerations

TBD

9. IANA Considerations

If the EHL approach is adopted to indicate the presence of MPLS extension header(s), this document requests IANA to assign one or more new Special-Purpose MPLS Label Values from the Special-Purpose Multiprotocol Label Switching (MPLS) Label Values Registry of "Extension Header Label (EHL)".

10. Contributors

The other contributors of this document are listed as follows.

* James Guichard
* Stewart Bryant
* Bruno Decraene

11. Acknowledgments

TBD.

12. References

12.1. Normative References

12.2. Informative References

[I-D.guichard-sfc-mpls-metadata]

[I-D.song-mpls-extension-header]

Authors’ Addresses

Haoyu Song (editor)
Futurewei Technologies
2330 Central Expressway
Santa Clara,
United States of America

Email: haoyu.song@futurewei.com
MPLS Extension Header
draft-song-mpls-extension-header-06

Abstract

Motivated by the need to support multiple in-network services and functions in an MPLS network, this document describes a generic and extensible method to encapsulate extension headers into MPLS packets. The encapsulation method allows stacking multiple extension headers and quickly accessing any of them as well as the original upper layer protocol header and payload. We show how the extension header can be used to support several new network applications and optimize some existing network services.

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.

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."
1. Motivation

Some applications require adding instructions and/or metadata to user packets within a network. Such examples include In-situ OAM (IOAM) [I-D.ietf-ippm-ioam-data] and Service Function Chaining (SFC) [RFC7665]. New applications are emerging. It is possible that the instructions and/or metadata for multiple applications are stacked together in one packet to support a compound service.

Conceivably, such instructions and/or metadata would be encoded as new headers and encapsulated in user packets. Such headers may require to be processed in fast path or in slow path. Moreover, such headers may require being attended at each hop on the forwarding path (i.e., hop-by-hop or HBH) or at designated end nodes (i.e., end-to-end or E2E).
The encapsulation of the new header(s) poses some challenges to MPLS networks, because the MPLS protocol header contains no explicit indicator for the upper layer protocols by design. We leave the discussion on the indicator of new header(s) in an MPLS packet to another companion document [I-D.song-mpls-eh-indicator]. In this document, we focus on the encode and encapsulation of new headers in an MPLS packet.

The similar problem has been tackled for some particular application before. However, these solutions have some drawbacks:

* The solutions rely on either the built-in next-protocol indicator in the header or the knowledge of the format and size of the header to access the following packet data. The node is required to be able to parse the new header, which is unrealistic in an incremental deployment environment.

* These works only provide piecemeal solution which assumes the new header is the only extra header and its location in the packet is fixed by default. It is impossible or difficult to support multiple new headers in one packet due to the conflicted assumption.

* Some previous work such as G-ACH [RFC5586] was explicitly defined for control channel only but what we need is the user packet service.

To solve these problems, we introduce extension header as a general and extensible means to support new in-network functions and applications in MPLS networks. The idea is similar to IPv6 extension headers which offer a huge innovation potential (e.g, network security, SRv6 [RFC8754], network programming [I-D.ietf-spring-srv6-network-programming], SFC [I-D.xu-clad-spring-sr-service-chaining], etc.). Thanks to the existence of extension headers, it is straightforward to introduce new in-network services into IPv6 networks. For example, it has been proposed to carry IOAM header [I-D.brockners-inband-oam-transport] as a new extension header option in IPv6 networks.

Nevertheless, IPv6 EH is not perfect either. It has three issues:

* IPv6’s header is large compared to MPLS, claiming extra bandwidth overhead and complicating the packet processing. We prefer to retain the header compactness in MPLS networks.

* IPv6’s extension headers are chained with the original upper layer protocol headers in a flat stack. One must scan all the extension headers to access the upper layer protocol headers and the
payload. This is inconvenient and raises some performance concerns for some applications (e.g., DPI and ECMP). The new scheme for MPLS header extension needs to address these issues too.

* [RFC8200] enforces many constraints to IPv6 extension headers (e.g., EH can only be added or deleted by the end nodes specified by the IP addresses in the IPv6 header, and there is only one Hop-by-Hop EH that can be processed on the path nodes), which are not suitable for MPLS networks. For example, MPLS label stacks are added and changed in network, and there could be tunnel within tunnel, so the extension headers need more flexibility.

2. MPLS Extension Header

From the previous discussion, we have laid out the design requirements to support extension headers in MPLS networks:

Performance: Unnecessary label stack scanning for a label and the full extension header stack scanning for the upper layer protocol should be avoided. The extension headers a node needs to process should be located as close to the MPLS label stack as possible. Each extension header is better to serve only one application to avoid the need of packing multiple TLV options in one extension header.

Scalability: New applications can be supported by introducing new extension headers. Multiple extension headers can be easily stacked together to support multiple services simultaneously.

Backward Compatibility: Legacy devices which do not recognize the extension header option should still be able to forward the packets as usual. If a device recognize some of the extension headers but not the others in an extension header stack, it can process the known headers only while ignoring the others.

Flexibility: A node can be configured to process or not process any EH. Any tunnel end nodes in the MPLS domain can add new EH to the packets which shall be removed on the other end of the tunnel.

We assume the MPLS label stack has included some indicator of the extension header(s). The actual extension headers are inserted between the MPLS label stack and the original upper layer packet header. The format of the MPLS packets with extension headers is shown in Figure 1.
Following the MPLS label stack is the 4-octet Header of Extension Headers (HEH), which indicates the total number of extension headers in this packet, the overall length of the extension headers, the type of the original upper layer header, and the type of the next header. The format of the HEH is shown in Figure 2.

![HEH Format](image)

The meaning of the fields in an HEH is as follows:

- **R**: 4-bit reserved. The nibble value means to avoid conflicting with
IP version numbers.

EHC: 4-bit unsigned integer for the Extension Header Counter. This field keeps the total number of extension headers included in this packet. It does not count the original upper layer protocol headers. At most 15 EHs are allowed in one packet.

EHTL: 8-bit unsigned integer for the Extension Header Total Length in 4-octet units. This field keeps the total length of the extension headers in this packet, not including the HEH itself.

OUL: 8-bit Original Upper Layer protocol number indicating the original upper layer protocol type. It can be set to "UNKNOWN" if unknown.

NH: 8-bit selector for the Next Header. This field identifies the type of the header immediately following the HEH.

The value of the reserved nibble needs further consideration. The EHC field can be used to keep track of the number of extension headers when some headers are inserted or removed at some network nodes. The EHTL field can help to skip all the extension headers in one step if the original upper layer protocol headers or payload need to be accessed. The OUL field can help identify the type of the original upper layer protocol.

The format of an Extension Header (EH) is shown in Figure 3.

```
0 1 2 3
01234567 89012345 67890123 45678901
+--------+--------+--------+-------+
|  NH    |  HLEN  | EXT     |       |
+--------+--------+--------+       |
        |                                  |
+--------+--------+----------------+
        |                                  |
        |                                  |
        +--------+----------------+++++
                       Header Specific Data
                       +----------------+++++
```

Figure 3: EH Format

The meaning of the fields in an EH is as follows:

NH: 8-bit indicator for the Next Header. This field identifies the type of the EH immediately following this EH.

HLEN: 8-bit unsigned integer for the Extension Header Length in 4-octet units, not including the first 4 octets.
EXT: 8-bit optional type extension. To save the Next Header numbers and extend the number space, it is possible to use one "Next Header" code to cover a set of sub-types. Each sub-type is assigned a new code in a different name space. This field is optional and it is only specified for some specific EH type.

Header Specific Data: Variable length field for the specification of the EH. This field is 4-octet aligned.

The extension headers as well as the first original upper layer protocol header are chained together through the NH field in HEH and EHs. The encoding of NH can share the same value registry for IPv4/IPv6 protocol numbers. Values for new EH types shall be assigned by IANA.

Specifically, the NH field of the last EH in a chain can have some special values, which shall be assigned by IANA as well:

NONE (No Next Header): Indicates that there is no other header and payload after this header. This can be used to transport packets with only extension header(s), for example, the control packets for control or the probe packets for measurements. Note that value 59 was reserved for "IPv6 No Next Header" indicator. It may be possible for MPLS EH to share this value.

UNKNOWN (Unknown Next Header): Indicates that the type of the header after this header is unknown. This is intended to be compatible with the original MPLS design in which the upper layer protocol type is unknown from the MPLS header alone.

MPLS: Indicates that the original upper layer protocol is still MPLS and another MPLS label stack follows.

Note that the original upper layer protocol can be of type "MPLS", which implies that in a packet there might be multiple label stacks separated by EHs. Having more than one independent label stack is not new. For example, A Bier header could separate the transport/bier labels and the payload labels; An MPLS PW network could be implemented on the top of another infrastructure MPLS network. In such cases, we have the flexibility to apply different services to different label stacks.
3. Type of MPLS Extension Headers

Basically, there are two types of MPLS EHs: HBH and E2E. E2E means that the EH is only supposed to be inserted/removed and processed at the MPLS tunnel end points where the MPLS header is inserted or removed. The EHs that need to be processed on path nodes within the MPLS tunnel are of the HBH type. However, any node in the tunnel can be configured to ignore an HBH EH, even if it is capable of processing it.

If there are two types of EHs in a packet, the HBH EHs must take precedence over the E2E EHs.

Making a distinction of the EH types and ordering the EHs in a packet help improve the forwarding performance. For example, if a node within an MPLS tunnel finds only E2E EHs in a packet, it can avoid scanning the EH list.

4. Operation on MPLS Extension Headers

When the first EH X needs to be added to an MPLS packet, an EH indicator is inserted into the proper location in the MPLS label stack. A HEH is then inserted after the MPLS label stack, in which EHCNT is set to 1, EHTLEN is set to the length of X in 4-octet units, and NH is set to the header value of X. At last, X is inserted after the HEH, in which NH and HELN are set accordingly. Note that if this operation happens at a PE device, the upper layer protocol is known before the MPLS encapsulation, so its value can be saved in the NH field if desired. Otherwise, the NH field is filled with the value of "UNKNOWN".

When an EH Y needs to be added to an MPLS packet which already contains extension header(s), the EHCNT and EHTLEN in the HEH are updated accordingly (i.e., EHCNT is incremented by 1 and EHTLEN is incremented by the size of Y in 4-octet units). Then a proper location for Y in the EH chain is located. Y is inserted at this location. The NH field of Y is copied from the previous EH’s NH field (or from the HEH’s NH field, if Y is the first EH in the chain). The previous EH’s NH value, or, if Y is the first EH in the chain, the HEH’s NH, is set to the header value of Y.

Deleting an EH simply reverses the above operation. If the deleted EH is the last one, the EH indicator and HEH can also be removed.

When processing an MPLS packet with extension headers, the node needs to scan through the entire EH chain and process the EH one by one. The node should ignore any unrecognized EH or the EH that is configured as "No Processing".
The EH can be categorized into HBH or E2E. Since EHs are ordered based on their type (i.e., HBH EHs are located before E2E EHs), a node can avoid some unnecessary EH scan.

5. Use Cases

In this section, we show how MPLS extension header can be used to support several new network applications.

In-situ OAM: In-situ OAM (IOAM) records flow OAM information within user packets while the packets traverse a network. The instruction and collected data are kept in an IOAM header [I-D.ietf-ippm-ioam-data]. When applying IOAM in an MPLS network, the IOAM header can be encapsulated as an MPLS extension header.

Network Telemetry and Measurement: A network telemetry and instruction header can be carried as an extension header to instruct a node what type of network measurements should be done. For example, the method described in [RFC8321] can be implemented in MPLS networks since the EH provides a natural way to color MPLS packets.

Network Security: Security related functions often require user packets to carry some metadata. In a DoS limiting network architecture, a "packet passport" header is used to embed packet authentication information for each node to verify.

Segment Routing and Network Programming: MPLS extension header can support the implementation of a new flavor of the MPLS-based segment routing, with better performance and richer functionalities. The details will be described in another draft.

With MPLS extension headers, multiple in-network applications can be stacked together. For example, IOAM and SFC can be applied at the same time to support network OAM and service function chaining. A node can stop scanning the extension header stack if all the known headers it can process have been located. For example, if IOAM is the first EH in a stack and a node is configured to process IOAM only, it will stop searching the EH stack when the IOAM EH is found.

6. Security Considerations

TBD
7. IANA Considerations

This document requests IANA to assign two new Internet Protocol Numbers from the "Protocol Numbers" Registry to indicate "No Next Header" and "Unknown Next Header".

This document does not create any other new registries.

8. Contributors

The other contributors of this document are listed as follows.

* James Guichard
* Stewart Bryant
* Andrew Malis

9. Acknowledgments

TBD.

10. References

10.1. Normative References


10.2. Informative References

[I-D.brockners-inband-oam-transport]

[I-D.ietf-ippm-ioam-data]

[I-D.ietf-spring-srv6-network-programming]

[I-D.song-mpls-eh-indicator]

[I-D.xu-clad-spring-sr-service-chaining]


Authors’ Addresses

Haoyu Song (editor)
Futurewei Technologies
Santa Clara,
United States of America
Email: haoyu.song@futurewei.com

Zhenbin Li
Huawei
Beijing
P.R. China
Email: lizhenbin@huawei.com

Tianran Zhou
Huawei
Beijing
P.R. China
Email: zhoutianran@huawei.com

Loa Andersson
Bronze Dragon Consulting
Stockholm
Sweden
Email: loa@pi.nu
Zhachui Zhang
Juniper Networks
Boston,
United States of America

Email: zzhang@juniper.net
Label Switched Path (LSP) Ping for Segment Routing (SR) Path Segment Identifiers (SIDs) with MPLS Data Planes
draft-xp-mpls-spring-lsp-ping-path-sid-03

Abstract

Path Segment is a type of SR segment, which is used to identify an SR path. This document provides Target Forwarding Equivalence Class (FEC) stack TLV definitions for Path Segment Identifiers.

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 September 2022.

Copyright Notice

Copyright (c) 2022 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 (https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.
Table of Contents

1. Introduction ........................................... 2
2. Conventions ........................................... 2
   2.1. Requirements Language ............................. 2
   2.2. Terminology ...................................... 3
3. Path Segment ID Sub-TLV .................................. 3
   3.1. SR Policy’s Path SID ............................... 3
   3.2. SR Candidate Path’s Path SID ...................... 5
   3.3. SR Segment List’s Path SID ....................... 7
4. Path-SID FEC Validation .................................. 9
5. Security Considerations ................................ 13
6. IANA Considerations ................................... 14
7. Acknowledgements ...................................... 14
8. References ........................................... 14
   8.1. Normative References .............................. 14
   8.2. Informative References ............................ 15
Authors’ Addresses ....................................... 17

1. Introduction

Path Segment is a type of SR segment, which is used to identify an SR path. Path Segment in MPLS based segment routing network is defined in [I-D.ietf-spring-mpls-path-segment].

When Path Segment is used, it’s inserted by the ingress node of the SR path, and then processed by the egress node of the SR path. The position of Path Segment Label within the MPLS label stack is immediately following the segment list of the SR path. Note that the Path Segment would not be popped up until it reaches the egress node.

This document provides Target Forwarding Equivalence Class (FEC) stack TLV definitions for Path-SIDs. Procedures for LSP Ping as defined in [RFC8287] and [RFC8690] are applicable to Path-SIDs as well.

2. Conventions

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.
2.2. Terminology

This document uses the terminology defined in [RFC8402] and [RFC8029], readers are expected to be familiar with those terms.

3. Path Segment ID Sub-TLV

Analogous to what’s defined in Section 5 of [RFC8287] and Section 4 of [I-D.ietf-mpls-sr-epe-oam], three new sub-TLVs are defined for the Target FEC Stack TLV (Type 1), the Reverse-Path Target FEC Stack TLV (Type 16), and the Reply Path TLV (Type 21).

<table>
<thead>
<tr>
<th>Sub-Type</th>
<th>Sub-TLV Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD1</td>
<td>SR Policy’s Path SID</td>
</tr>
<tr>
<td>TBD2</td>
<td>SR Candidate Path’s Path SID</td>
</tr>
<tr>
<td>TBD3</td>
<td>SR Segment List’s Path SID</td>
</tr>
</tbody>
</table>

As specified in Section 2 of [I-D.ietf-spring-mpls-path-segment], the Path Segment may be used to identify an SR Policy, its Candidate Path, or a Segment List, so three different Target FEC sub-TLVs need to be defined for Path Segment ID. When a Path Segment is used to identify an SR Policy, the Target FEC sub-TLV of SR Policy’s Path SID would be used to validate the control plane to forwarding plane synchronization for this Path-SID; When a Path Segment is used to identify an SR Candidate Path, the Target FEC sub-TLV of SR Candidate Path’s Path SID would be used to validate the control plane to forwarding plane synchronization for this Path-SID; When a Path Segment is used to identify a Segment List, the Target FEC sub-TLV of SR Segment List’s Path SID would be used to validate the control plane to forwarding plane synchronization for this Path-SID. Note that the three new Target FEC sub-TLVs are mutual exclusive and they wouldn’t be present in one message simultaneously.

3.1. SR Policy’s Path SID

The format of SR Policy’s Path SID sub-TLV is as specified below:
Type

This field is set to the value (TBD1) which indicates that it’s an SR Policy’s Path SID sub-TLV.

Length

This field is set to the length of the sub-TLV’s Value field in octets. If Headend and Endpoint fields are in IPv4 address format which is 4 octets long, it MUST be set to 12; If Headend and Endpoint fields are in IPv6 address format which is 16 octets long, it MUST be set to 36.

Headend

This field identifies the headend of an SR Policy, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The headend is a 4-octet IPv4 address or a 16-octet IPv6 address.

Color

This field associates the SR Policy with an intent, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The color is a 4-octet numerical value.

Endpoint

This field identifies the endpoint of an SR Policy, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The endpoint is a 4-octet IPv4 address or a 16-octet IPv6 address.
3.2. SR Candidate Path’s Path SID

The format of SR Candidate Path’s Path SID sub-TLV is as specified below:

```
  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 = TBD2          |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                     Headend (4/16 octets)                    
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
              Color (4 octets)                       
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                     Endpoint (4/16 octets)                    
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Protocol-Origin|                    Reserved                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Originator (20 octets)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Discriminator (4 octets)
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: SR Candidate Path’s Path SID sub-TLV

Type

This field is set to the value (TBD2) which indicates that it’s an SR Candidate Path’s Path SID sub-TLV.

Length

This field is set to the length of the sub-TLV’s Value field in octets. If Headend and Endpoint fields are in IPv4 address format which is 4 octets long, it MUST be set to 40; If Headend and Endpoint fields are in IPv6 address format which is 16 octets long, it MUST be set to 64.

Headend
This field identifies the headend of an SR Policy, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The headend is a 4-octet IPv4 address or a 16-octet IPv6 address.

Color

This field associates the SR Policy with an intent, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The color is a 4-octet numerical value.

Endpoint

This field identifies the endpoint of an SR Policy, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The endpoint is a 4-octet IPv4 address or a 16-octet IPv6 address.

Protocol-Origin

This field identifies the component or protocol that originates or signals the candidate path for an SR Policy, the same as defined in Section 2.3 of [I-D.ietf-spring-segment-routing-policy]. The protocol-origin is a 1-octet value that follows the recommendation from Table 1 of Section 2.3 of [I-D.ietf-spring-segment-routing-policy], which specifies value 10 for "PCEP", value 20 for "BGP SR Policy" and value 30 for "Via Configuration".

Originator

This field identifies the node which provisioned or signaled the candidate path for an SR Policy, the same as defined in Section 2.4 of [I-D.ietf-spring-segment-routing-policy]. The originator is a 20-octet numerical value formed by the concatenation of the fields of the tuple <ASN, node-address>, among which ASN is a 4-octet number and node address is a 16-octet value (an IPv6 address or an IPv4 address encoded in the lowest 4 octets). When Protocol-Origin is respectively "Via Configuration", or "PCEP", or "BGP SR Policy", the values of ASN and node address follow the specification in Section 2.4 of [I-D.ietf-spring-segment-routing-policy].

Discriminator
This field uniquely identifies a candidate path within the context of an SR policy, the same as defined in Section 2.5 of [I-D.ietf-spring-segment-routing-policy]. The discriminator is a 4-octet value. When Protocol-Origin is respectively "Via Configuration", or "PCEP", or "BGP SR Policy", the value of discriminator follows the specification in Section 2.5 of [I-D.ietf-spring-segment-routing-policy].

3.3. SR Segment List’s Path SID

The format of SR Segment List’s Path SID sub-TLV is as specified below:

```
+-------------+-------------+-------------+-------------+
|             |             |             |             |
| 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 |
+-------------+-------------+-------------+-------------+
| Type = TBD3 | Length      |
|             |             |             |             |
|             | Headend (4/16 octets) |             |
|             | Color (4 octets) |
|             | Endpoint (4/16 octets) |
| Protocol-Origin | Reserved |
|               |             |
|               |             |
|               |               |
|               | Reserve      |
|               |             |
|               |             |
|               | Originator (20 octets) |
|               |               |
|               | Discriminator (4 octets) |
|               |               |
|               | Segment-List-ID (4 octets) |
|               |               |
|               |               |
+-------------+-------------+-------------+-------------+
```

Figure 3: SR Segment List’s Path SID sub-TLV

Type

This field is set to the value (TBD3) which indicates that it’s an SR Segment List’s Path SID sub-TLV.

Length
This field is set to the length of the sub-TLV’s Value field in octets. If Headend and Endpoint fields are in IPv4 address format which is 4 octets long, it MUST be set to 44; if Headend and Endpoint fields are in IPv6 address format which is 16 octets long, it MUST be set to 68.

**Headend**

This field identifies the headend of an SR Policy, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The headend is a 4-octet IPv4 address or a 16-octet IPv6 address.

**Color**

This field associates the SR Policy with an intent, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The color is a 4-octet numerical value.

**Endpoint**

This field identifies the endpoint of an SR Policy, the same as defined in Section 2.1 of [I-D.ietf-spring-segment-routing-policy]. The endpoint is a 4-octet IPv4 address or a 16-octet IPv6 address.

**Protocol-Origin**

This field identifies the component or protocol that originates or signals the candidate path for an SR Policy, the same as defined in Section 2.3 of [I-D.ietf-spring-segment-routing-policy]. The protocol-origin is a 1-octet value that follows the recommendation from Table 1 of Section 2.3 of [I-D.ietf-spring-segment-routing-policy], which specifies value 10 for "PCEP", value 20 for "BGP SR Policy" and value 30 for "Via Configuration".

**Originator**

This field identifies the node which provisioned or signaled the candidate path for an SR Policy, the same as defined in Section 2.4 of [I-D.ietf-spring-segment-routing-policy]. The originator is a 20-octet numerical value formed by the concatenation of the fields of the tuple <ASN, node-address>, among which ASN is a 4-octet number and node address is a 16-octet value (an IPv6 address or an IPv4 address encoded in the lowest 4 octets). When Protocol-Origin is respectively "Via
Configuration", or "PCEP", or "BGP SR Policy", the values of ASN and node address follow the specification in Section 2.4 of [I-D.ietf-spring-segment-routing-policy].

Discriminator

This field uniquely identifies a candidate path within the context of an SR policy, the same as defined in Section 2.5 of [I-D.ietf-spring-segment-routing-policy]. The discriminator is a 4-octet value. When Protocol-Origin is respectively "Via Configuration", or "PCEP", or "BGP SR Policy", the value of discriminator follows the specification in Section 2.5 of [I-D.ietf-spring-segment-routing-policy].

Segment-List-ID

This field identifies an SR path within the context of a candidate path of an SR Policy, the same as "Path ID" defined in Section 4.2 of [I-D.ietf-pce-multipath], or "List Identifier" defined in Section 2.2 of [I-D.lp-idr-sr-path-protection]. The segment-list-ID is a 4-octet identifier of the corresponding segment list.

4. Path-SID FEC Validation

The MPLS LSP Ping procedures MAY be initiated by the headend of the Segment Routing path or a centralized topology-aware data plane monitoring system as described in [RFC8403]. For the Path-SID, the responder nodes that receive echo request and send echo reply MUST be the endpoint of the Segment Routing path.

When an endpoint receives the LSP echo request packet with top FEC being the Path-SID, it SHOULD perform validity checks on the content of the Path-SID FEC sub-TLV. The basic length check should be performed on the received FEC.

SR Policy’s Path SID
---------------------
Length = 12 or 36

SR Candidate Path’s Path SID
---------------------
Length = 40 or 64

SR Segment List’s Path SID
---------------------
Length = 44 or 68
If a malformed FEC sub-TLV is received, then a return code of 1, "Malformed echo request received" as defined in [RFC8029] SHOULD be sent. The below section augments the section 7.4 of [RFC8287].

4a. Segment Routing Path-SID Validation:

If the Label-stack-depth is 0 and the Target FEC Stack sub-TLV at FEC-stack-depth is TBD1 (SR Policy’s Path SID sub-TLV), {

- Set the Best-return-code to 10, "Mapping for this FEC is not the given label at stack-depth <RSC>" if any below conditions fail:

  o Validate that the Path Segment ID is signaled or provisioned for the SR Policy {

    + Validate that the signaled or provisioned headend, color and end-point for the Path SID, matches with the corresponding fields in the received SR Policy’s Path SID sub-TLV.

  }

- If all the above validations have passed, set the return code to 3 "Replying router is an egress for the FEC at stack-depth <RSC>".

- Set FEC-Status to 1 and return.

}

Else, if the Label-stack-depth is 0 and the Target FEC Stack sub-TLV at FEC-stack-depth is TBD2 (SR Candidate Path’s Path SID sub-TLV), {

- Set the Best-return-code to 10, "Mapping for this FEC is not the given label at stack-depth <RSC>" if any below conditions fail:

  o Validate that the Path Segment ID is signaled or provisioned for the SR Candidate Path {


+ When the Protocol-Origin field in the received SR Candidate Path’s Path SID sub-TLV is 10, "PCEP" is used as the signaling protocol. And then validate that the Path Segment ID matches with the tuple identifying the SR Candidate Path within PCEP {

    * Validate that the signaled headend, color, end-point, originator ASN, originator address and discriminator defined in [I-D.ietf-pce-segment-routing-policy-cp] and [I-D.ietf-pce-sr-path-segment], for the Path SID, matches with the corresponding fields in the received SR Candidate Path’s Path SID sub-TLV.

} 

+ When the Protocol-Origin field in the received SR Candidate Path’s Path SID sub-TLV is 20, "BGP SR Policy" is used as the signaling protocol. And then validate that the Path Segment ID matches with the tuple identifying the SR Candidate Path within BGP SR Policy {

    * Validate that the signaled headend, policy color, endpoint, ASN, BGP Router-ID and distinguisher defined in [I-D.ietf-idr-segment-routing-te-policy] and [I-D.ietf-idr-sr-policy-path-segment], for the Path SID, matches with the corresponding fields in the received SR Candidate Path’s Path SID sub-TLV.

} 

+ When the Protocol-Origin field in the received SR Candidate Path’s Path SID sub-TLV is 30, "Via Configuration" is used. And then validate that the Path Segment ID matches with the tuple identifying the SR Candidate Path within Configuration {

    * Validate that the provisioned headend, color, endpoint, originator and discriminator defined in [I-D.ietf-spring-sr-policy-yang], for the Path SID, matches with the corresponding fields in the received SR Candidate Path’s Path SID sub-TLV.

} 

}
- If all the above validations have passed, set the return code to 3 "Replingy router is an egress for the FEC at stack-depth <RSC>".

- Set FEC-Status to 1 and return.

}

Else, if the Label-stack-depth is 0 and the Target FEC Stack sub-TLV at FEC-stack-depth is TBD3 (SR Segment List’s Path SID sub-TLV), {

- Set the Best-return-code to 10, "Mapping for this FEC is not the given label at stack-depth <RSC>" if any below conditions fail:

  o Validate that the Path Segment ID is signaled or provisioned for the SR Segment List {

    + When the Protocol-Origin field in the received SR Segment List’s Path SID sub-TLV is 10, "PCEP" is used as the signaling protocol. And then validate that the Path Segment ID matches with the tuple identifying the SR Segment List within PCEP {

      * Validate that the signaled headend, color, end-point, originator ASN, originator address and discriminator defined in [I-D.ietf-pce-segment-routing-policy-cp] and [I-D.ietf-pce-sr-path-segment], and the signaled Path ID defined in [I-D.ietf-pce-multipath], for the Path SID, matches with the corresponding fields in the received SR Segment List’s Path SID sub-TLV.

    }

    + When the Protocol-Origin field in the received SR Segment List’s Path SID sub-TLV is 20, "BGP SR Policy" is used as the signaling protocol. And then validate that the Path Segment ID matches with the tuple identifying the SR Segment List within BGP SR Policy {


* Validate that the signaled headend, policy color, endpoint, ASN, BGP Router-ID and distinguisher defined in [I-D.ietf-idr-segment-routing-te-policy] and [I-D.ietf-idr-sr-policy-path-segment], and the signaled List Identifier defined in [I-D.lp-idr-sr-path-protection], for the Path SID, matches with the corresponding fields in the received SR Segment List’s Path SID sub-TLV.

} 

+ When the Protocol-Origin field in the received SR Segment List’s Path SID sub-TLV is 30, "Via Configuration" is used. And then validate that the Path Segment ID matches with the tuple identifying the SR Segment List within Configuration {

* Validate that the provisioned headend, color, endpoint, originator, discriminator and Segment-List-ID defined in [I-D.ietf-spring-sr-policy-yang], for the Path SID, matches with the corresponding fields in the received SR Segment List’s Path SID sub-TLV.

}

} 

- If all the above validations have passed, set the return code to 3 "Replied router is an egress for the FEC at stack-depth <RSC>".

- Set FEC-Status to 1 and return.

}

5. Security Considerations

This document defines additional MPLS LSP Ping sub-TLVs and follows the mechanisms defined in [RFC8029]. All the security considerations defined in [RFC8029] will be applicable for this document and, in addition, they do not impose any additional security challenges to be considered.
6. IANA Considerations

IANA is requested to assign three new sub-TLVs from the "sub-TLVs for TLV Types 1, 16, and 21" subregistry of the "Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters" registry [IANA].

<table>
<thead>
<tr>
<th>Sub-Type</th>
<th>Sub-TLV Name</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD1</td>
<td>SR Policy’s Path SID</td>
<td>Section 3.1</td>
</tr>
<tr>
<td>TBD2</td>
<td>SR Candidate Path’s Path SID</td>
<td>Section 3.2</td>
</tr>
<tr>
<td>TBD3</td>
<td>SR Segment List’s Path SID</td>
<td>Section 3.3</td>
</tr>
</tbody>
</table>

7. Acknowledgements

The authors would like to acknowledge Detao Zhao for his thorough review and very helpful comments.

The authors would like to acknowledge Yao Liu for the very helpful f2f discussion.

8. References

8.1. Normative References

[I-D.ietf-spring-mpls-path-segment]


8.2. Informative References

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P.,
Jain, D., and S. Lin, "Advertising Segment Routing
Policies in BGP", Work in Progress, Internet-Draft, draft-
ietf-idr-segment-routing-te-policy-14, 10 November 2021,
<https://www.ietf.org/archive/id/draft-ietf-idr-segment-
routing-te-policy-14.txt>.

[I-D.ietf-idr-sr-policy-path-segment]
Li, C., Li, Z., Yin, Y., Cheng, W., and K. Talaulikar, "SR
Policy Extensions for Path Segment and Bidirectional
Path", Work in Progress, Internet-Draft, draft-ietf-idr-
sr-policy-path-segment-05, 23 January 2022,
<https://www.ietf.org/archive/id/draft-ietf-idr-
sr-policy-path-segment-05.txt>.

[I-D.ietf-mpls-sr-epe-oam]
Hegde, S., Arora, K., Srivastava, M., Ninan, S., and X.
Xu, "Label Switched Path (LSP) Ping/Traceroute for Segment
Routing (SR) Egress Peer Engineering Segment Identifiers
(SIDs) with MPLS Data Planes", Work in Progress, Internet-
Draft, draft-ietf-mpls-sr-epe-oam-04, 8 November 2021,
<https://www.ietf.org/archive/id/draft-ietf-mpls-sr-epe-
oam-04.txt>.

[I-D.ietf-pce-multipath]
Koldychev, M., Sivabalan, S., Saad, T., Beeram, V. P.,
Bidgoli, H., Yadav, B., Peng, S., and G. Mishra, "PCEP
Extensions for Signaling Multipath Information", Work in
Progress, Internet-Draft, draft-ietf-pce-multipath-04, 25
February 2022, <https://www.ietf.org/archive/id/draft-
ietf-pce-multipath-04.txt>.


IANA


Authors’ Addresses

Xiao Min
ZTE Corp.
Nanjing
China
Phone: +86 25 88013062
Email: xiao.min2@zte.com.cn

Shaofu Peng
ZTE Corp.
Nanjing
China
Email: peng.shaofu@zte.com.cn