

MPLS Working Group  
Internet-Draft  
Intended status: Informational  
Expires: October 7, 2022

L. Andersson  
Bronze Dragon Consulting  
J. Guichard  
H. Song  
Futurewei Technologies  
S. Bryant  
University of Surrey  
April 5, 2022

MPLS Extension Header Architecture  
draft-andersson-mpls-eh-architecture-03

## 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]

- o In-situ OAM: In-situ OAM (IOAM) records flow OAM information within user packets while the packets traverse a network.
- o 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.
- o Network Security: Security related functions may require user packets to carry some metadata.
- o 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.

- o EH - Extension Header
- o EHI - Extension Header Indicator
- o LDP DoD - LDP Downstream on Demand
- o LDP DU - LDP Downstream Unsolicited
- o 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.

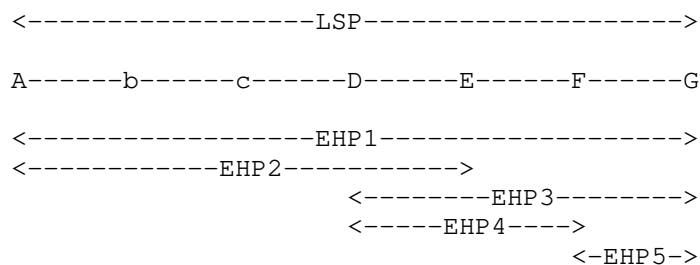
An EH capable node may not add an extension header to a packet if unless it is sure that there is a downstream node that can remove it.

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, 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.

EHP1 - EHP1 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.

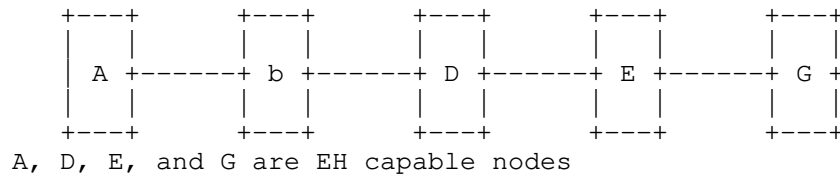
### 3.6. LSP establishment with LDP Downstream on Demand (DoD) in an EH capable network

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.



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:

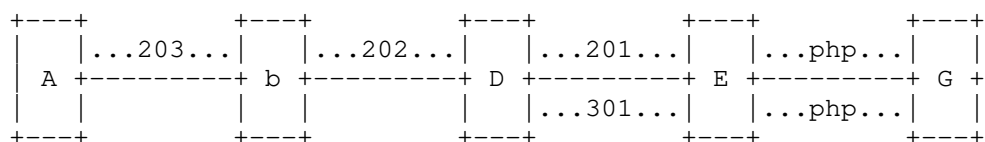
- o A sends an LDP Label Request message to b, indicating that an EH capable LSP should be set up to G. A keeps track of the outstanding request.
- o b is not EH capable and treat the Label Request as a normal request, however, the information indicating that an EH capable LSP is requested is transitive and sent to D.
- o D receives the Label Request, forwards it to E, and keeps track of the outstanding request.
- o E treats the label request the same way as D, and forward it to G.
- o 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.
- o 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.
- o 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.
- o 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.

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

This will result in installed labels like this.



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.

- o 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.
- o 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.
- o 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.

- o 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.
- o A receives the label mapping and installs label value 203 for the IP FEC.
- o 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 folling 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 similzar 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:

Value	Hex	Name	Label Distribution Discipline	Reference	Note/Reg. Date
TBD	TBD	No EH present	DoD or DU	This Document	TBA

Table 1: No EH present

## 10. Acknowledgements

-

-

## 11. References

## 11.1. Normative References

[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, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

## 11.2. Informative References

- [RFC5036] Andersson, L., Ed., Minei, I., Ed., and B. Thomas, Ed., "LDP Specification", RFC 5036, DOI 10.17487/RFC5036, October 2007, <<https://www.rfc-editor.org/info/rfc5036>>.

## Authors' Addresses

Loa Andersson  
Bronze Dragon Consulting

Email: [loa@pi.nu](mailto:loa@pi.nu)

James N Guichard  
Futurewei Technologies

Email: [james.n.guichard@futurewei.com](mailto:james.n.guichard@futurewei.com)

Haoyu Song  
Futurewei Technologies

Email: [haoyu.song@futurewei.com](mailto:haoyu.song@futurewei.com)

Stewart Bryant  
University of Surrey

Email: [stewart.bryant@gmail.com](mailto:stewart.bryant@gmail.com)

MPLS Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: October 7, 2022

L. Andersson  
Bronze Dragon Consulting  
J. Guichard  
H. Song  
Futurewei Technologies  
S. Bryant  
University of Surrey  
April 5, 2022

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 Extetended Spewcial Purpose label called Extention Headder 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

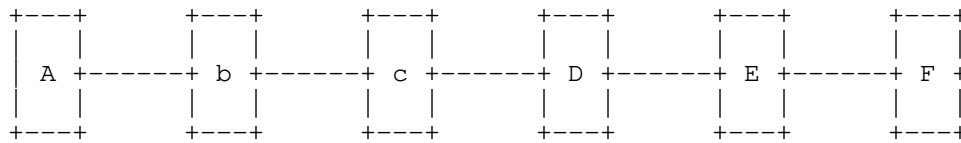
This document provides a set of examples to show the operations performed on MPLS encapsulated packets in a network where MPLS EHs are used. The document does also illustrated the procedures for processing of the information carried within the MPLS label stack to indicate the presence of EHs below the 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.

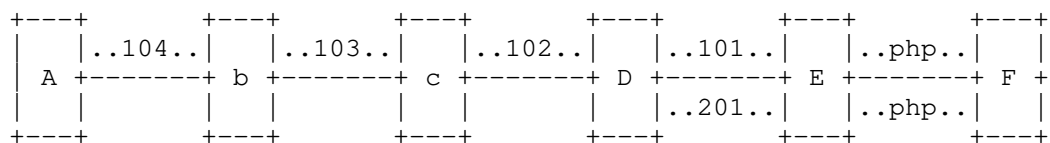
Figure 1: EH topology

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:

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

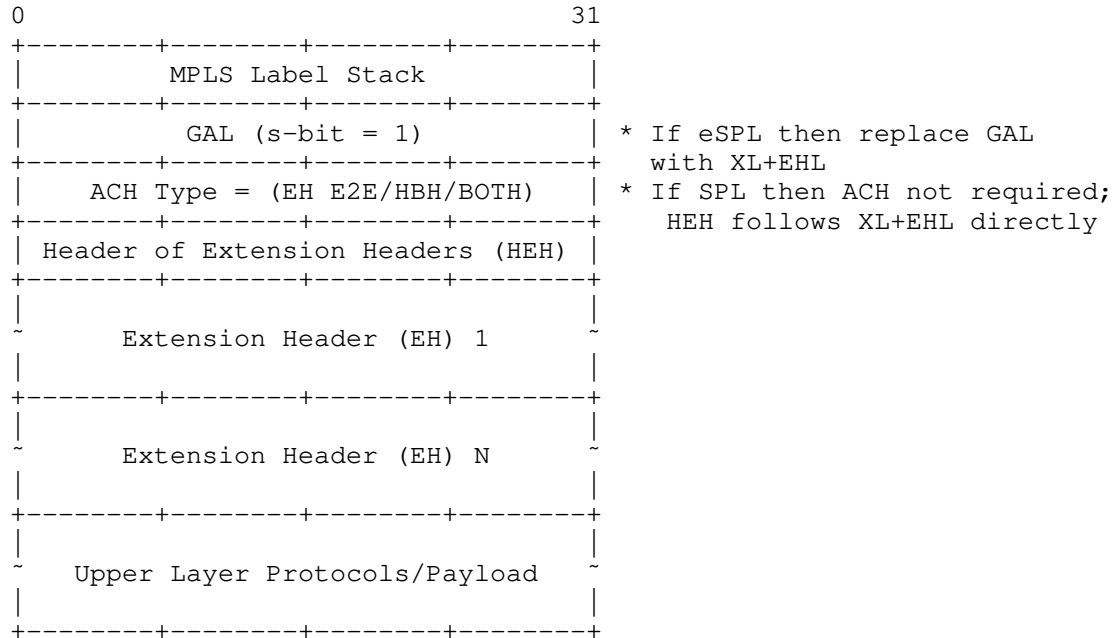


Figure 3: MPLS Extension Header (EH) Stack

#### 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

- o A sends packet to b
  - \* stack = [104, GAL, ACH, HEH, EH, IP]
- o b is a legacy router so just swaps [104] to [103], and sends the packet to c
  - \* stack = [103, GAL, ACH, HEH, EH, IP]
- o 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 send 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 does not include an EH will be sent on the "no EH present" FEC.

- o 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.
- o 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

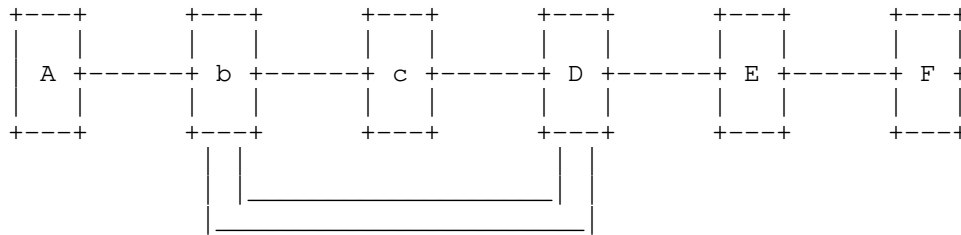
- o A sends packet to b
  - \* stack = [104, VPN, IP]
- o b receives the packet; b is a legacy router and just swaps [104] to [103] and sends the packet to c
  - \* stack = [103, VPN, IP]
- o c receives the packet; c is a legacy router and just swaps [103] to [102] and sends the packet to D
  - \* stack = [102, VPN, IP]
- o 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;
- o 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.

- o F receives and scans the packet for EH; finds no EH and bypasses EH processing. As F is the ultimate hop it processes the VPN label and forwards the packet.

#### 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.



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

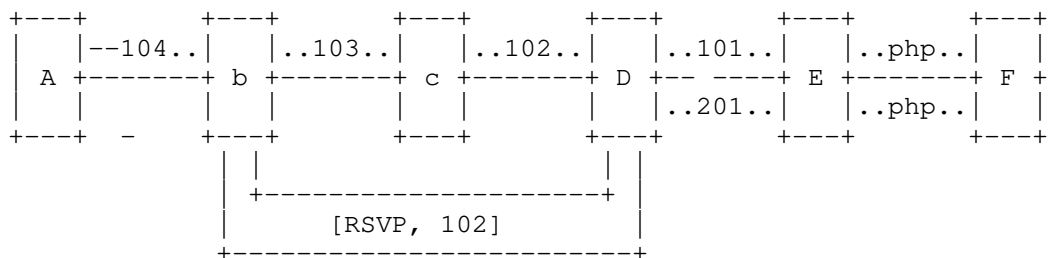
For this example the following assumptions are made:

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

For prefix [F]: The following label mappings are sent by the LSRs in the network.

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

This will result in label mappings like this.



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 labels.

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

- o 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 F 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]; F 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]; b 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]

- o Intermediate tunnel LSRs will forward (swap) based on the RSVP-TE label.
- o 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 send the packet on the standard FEC.
- o 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]
- o 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,  
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5586] Bocci, M., Ed., Vigoureux, M., Ed., and S. Bryant, Ed.,  
"MPLS Generic Associated Channel", RFC 5586,  
DOI 10.17487/RFC5586, June 2009,  
<<https://www.rfc-editor.org/info/rfc5586>>.

### 6.2. Informative References

- [RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating  
and Retiring Special-Purpose MPLS Labels", RFC 7274,  
DOI 10.17487/RFC7274, June 2014,  
<<https://www.rfc-editor.org/info/rfc7274>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC  
2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,  
May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

### Authors' Addresses

Loa Andersson  
Bronze Dragon Consulting  
  
Email: [loa@pi.nu](mailto: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  
Internet-Draft  
Intended status: Standards Track  
Expires: November 19, 2021

S. Bryant  
A. Clemm  
T. Eckert  
Futurewei Technologies, Inc.  
May 18, 2021

Use of an MPLS LSE as an Ancillary Data Pointer  
draft-bryant-mpls-aux-data-pointer-00

Abstract

The purpose of this memo is to describe how Label Stack Entries (LSEs) can be used to point to ancillary or meta-data carried below the MPLS label stack.

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 November 19, 2021.

Copyright Notice

Copyright (c) 2021 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  |
| 2. Background Documents . . . . .                                    | 2  |
| 3. Use of SPLs as Pointers . . . . .                                 | 3  |
| 4. Label Operations: Popping and Swapping . . . . .                  | 5  |
| 5. Use of Multiple Pointers . . . . .                                | 6  |
| 6. Disposition of the Ancillary Data . . . . .                       | 9  |
| 7. Structure of Ancillary Data . . . . .                             | 9  |
| 8. Structure of Pointer Label . . . . .                              | 9  |
| 9. Backward Compatibility . . . . .                                  | 10 |
| 10. Security Considerations . . . . .                                | 10 |
| 11. IANA Considerations . . . . .                                    | 11 |
| 12. Appendix 1: CONTROVERSY ALERT – Use Of Ordinary Labels . . . . . | 11 |
| 13. Appendix 2: Other Issues for Discussion . . . . .                | 12 |
| 14. Appendix 3: Ancillary vs Auxiliary vs Metadata . . . . .         | 12 |
| 15. References . . . . .                                             | 13 |
| 15.1. Normative References . . . . .                                 | 13 |
| 15.2. Informative References . . . . .                               | 13 |
| Authors' Addresses . . . . .                                         | 14 |

## 1. Introduction

There has been significant recent interest in developing the MPLS data plane to address new needs, and in particular to carry ancillary or meta-data below the stack. In this document we consider that this ancillary data is further subdivided into a sequence of blocks. This draft does not prescribe the information or its structure of the ancillary data. For the sake of examples, it could range from a single ancillary data unit to a structured set of ancillary data blocks similar to an IPv6 extension header. There has also been recent interest in carrying additional flags or other indicators to qualify the forwarding operations.

This memo proposes the use of "spare" bits in a Special Purpose Label (SPL) [I-D.kompella-mpls-mspl4fa] be used as a pointer to items of ancillary data carried below the bottom of stack (BoS). Finally we speculate that in certain network scopes we may usefully be able to create pseudo-SPLs from the ordinary label pool.

## 2. Background Documents

[I-D.kompella-mpls-mspl4fa] notes that the forwarder does not need to use the TC, or TTL fields in an LSE [RFC3032] that does not become top of stack (ToS). It proposes to exploit these fields as indicators of forwarding actions, by modifying the semantics of these fields.

There are a number of key proposals in that draft:

- o Using the "spare bits" as forwarding indicator flags to specify actions or in some cases inactions
- o Using the method to multi-purpose SPLs and thus expand the number of single label SPLs available to the IETF.
- o Reuse the Entropy Label (EL) fields to carry additional data needed by the forwarder. This latter point could be adopted by any eSPL. One use for this additional data that was proposed (certainly in discussion but I cannot see it in the draft) was the use of this facility to carry a network slice identifier.

This draft proposes that these "spare" bits in an SPL or pseudo-SPL be used as a pointer to ancillary data below the stack.

This proposal can be used in conjunction with the other indicator proposals, for example by using different SPLs for different options, such as one SPL indicating the presence of a pointer vs one or more other SPLs for the other proposals.

### 3. Use of SPLs as Pointers

Previously it had been proposed to use the "spare" bits in an SPL that is not ToS as a bit field or as an enumerator of a slice. However, it would appear to be an advantage to take things a bit further and use them as a pointer to ancillary data below the BoS. This ancillary data can then be accessed and processed as needed whenever the SPL is being processed.

The advantages of doing this are:

- o The ability to find the ancillary data without scanning the whole stack. Speculatively scanning the label stack can be expensive in Network Processor Unit (NPU) processing time, particularly if the stack is deep.
- o Ability to specify which ancillary data is applicable at the hop being processed.
- o The use of a pointer or set of pointers allows for a simple packet parser.
- o The approach is inherently general and extensible.

This concept is illustrated in Figure 1.

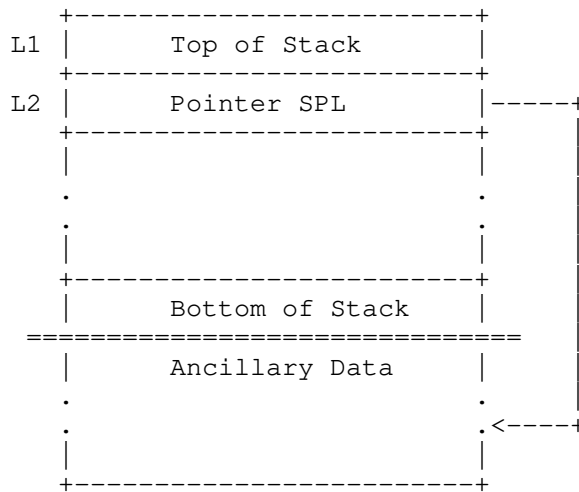


Figure 1: Use of In-stack MPLS pointer

The ToS label (L1) and Pointer SPL (L2) form a tuple with the semantic "process the action that the Forwarding Equivalence Class (FEC) of the ToS label specifies with the assistance of the information pointed to by the following SPL". Ideally L2 is SPL requiring a single LSE rather than an Extended SPL (ESPL) requiring two LSEs. Whilst the additional LSE required for an ESPL may not initially seem significant, the authors imagine that there may be cases where multiple pointer labels will be required.

Let us consider the case when the ToS is not an SPL of any kind. In this case, the forwarder looks at the following LSE (i.e., the LSE that immediately succeeds the ToS). If that LSE is not a pointer SPL, forwarding is performed as normal. If, on the other hand, the following label is a pointer SPL, the forwarder uses the information pointed to by the pointer as assistance in the forwarding operation.

Note that whilst the pointer can be simply point to the end of the stack, which aligns with the other MPLS proposals being made, the ideas discussed here can actually point to a specific item within the MPLS payload i.e. to a specific item of ancillary data. This in turn also means that different LSEs can point to different ancillary data components. This allows the MPLS application or packet designer to express sophisticated behavior in which it is possible to apply different ancillary data to different LSEs, i.e. different network segments.



#### 4. Label Operations: Popping and Swapping

When the ToS is popped, consideration needs to be given to any Pointer SPL immediately following it. In the basic case, a Pointer SPL will simply be popped along with the ToS.

There will be cases in which the same Pointer SPL applies to multiple labels. In those cases, requiring the forwarder to pop the Pointer SPL along with the ToS results in the need to carry multiple instances of the same Pointer SPL, one for each label to which it applies. As an optimization, it will make sense to offer a second behavioral option in which, upon popping the ToS, any subsequent Pointer SPL will be swapped with the next FEC label. This case is depicted in Figure 2 and Figure 3.

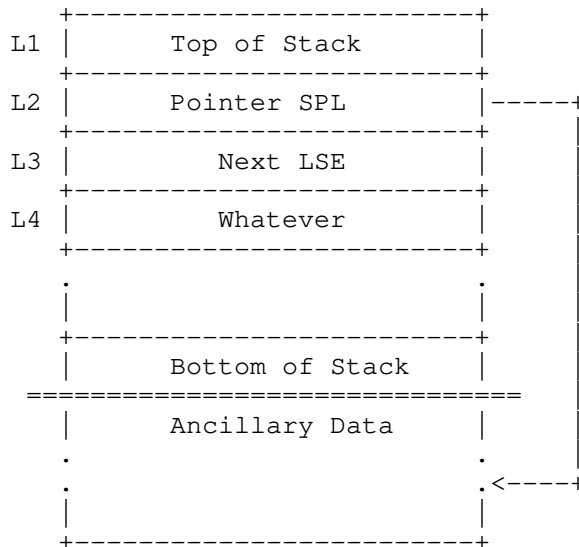


Figure 2: Before Pop - Swap operation:

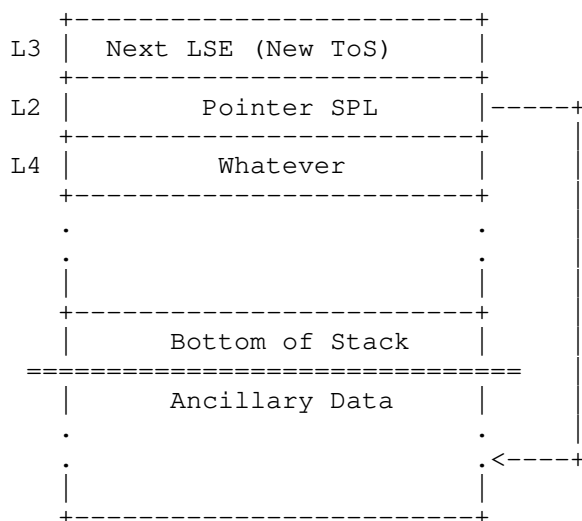


Figure 3: After Pop - Swap operation:

When this optimization is applied, there needs to be a distinction that allows a forwarder to determine whether a Pointer SPL should be popped along with its ToS, or whether it should be swapped with the next FEC label below.

One possibility is to indicate this in the FEC of the LSE that precedes the Pointer SPL, or perhaps it can be indicated by using one of its bits as a corresponding flag. Alternatively a perhaps some method can be found whereby a "TTL" can be associated with the pointer label. How to best indicate this end-of-use distinction is for further study.

## 5. Use of Multiple Pointers

One problem to be solved is how to support multiple independent (sets of) ancillary data in the MPLS header in support of different (forwarding, OAM etc) operations associated with the ancillary data. In IP/IPv6, ancillary data is encoded in the packet header through a sequence of Extension Headers (EH). For IPv6, [RFC8200] defines several EH types, each of which implies a specific set of nodes that have to process the EH and their order. This approach results in a complex parsing requirement for IPv6 packets when multiple extension headers are used and very rigid encoding and EH semantic difficult to extend.

Instead of limiting processing to a single pointer, it is possible to generalize the above concepts to allow for multiple pointers. This increases flexibility by allowing the packet designer to include

pointers to multiple sets of ancillary data, each of which can be potentially used for a different purpose. Therefore, the use of multiple adjacent Pointer SPLs is allowed. This means that ToS processing takes into considerations all Pointer SPLs that immediately follow.

A pointer mechanism in the MPLS label stack provides a method of using multiple pointers to express two (or more) sets of ancillary data, for example a latency object and an iOAM object. An example is shown in Figure 4.

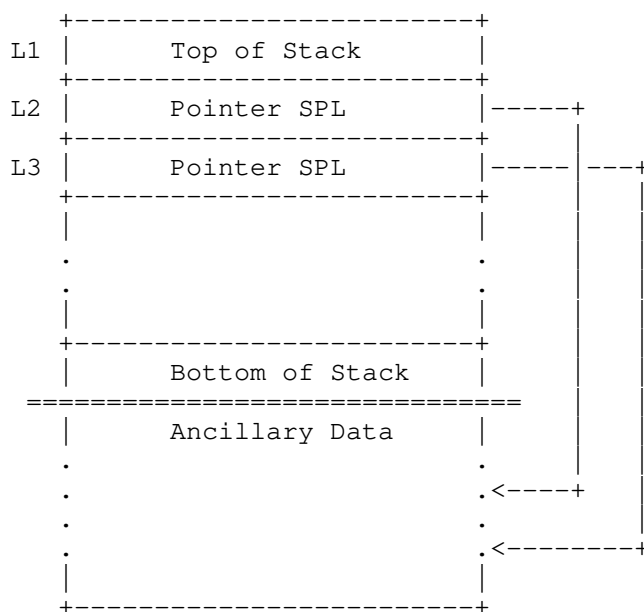


Figure 4: Use of Multiple In-stack MPLS Pointers

As in Figure 1 the top three labels are a tuple that in this case have the semantics "process the action that the FEC of the ToS label specifies with the assistance of the information pointed to by the following SPLs", in this case the label set L1, L2, L3. The tuple terminates when either an "ordinary" label, an SPL that is not a pointer SPL, or a label with the S bit set is encountered.

The Figure 5 further illustrates this capability. Here in this example two LSEs, Li and Lj, are each associated with two pointers. The FEC of Li therefore indicates that execution of Li includes the use of Ancillary Data 1 and 2, and execution of Lj includes the use of ancillary data 2 and 3.

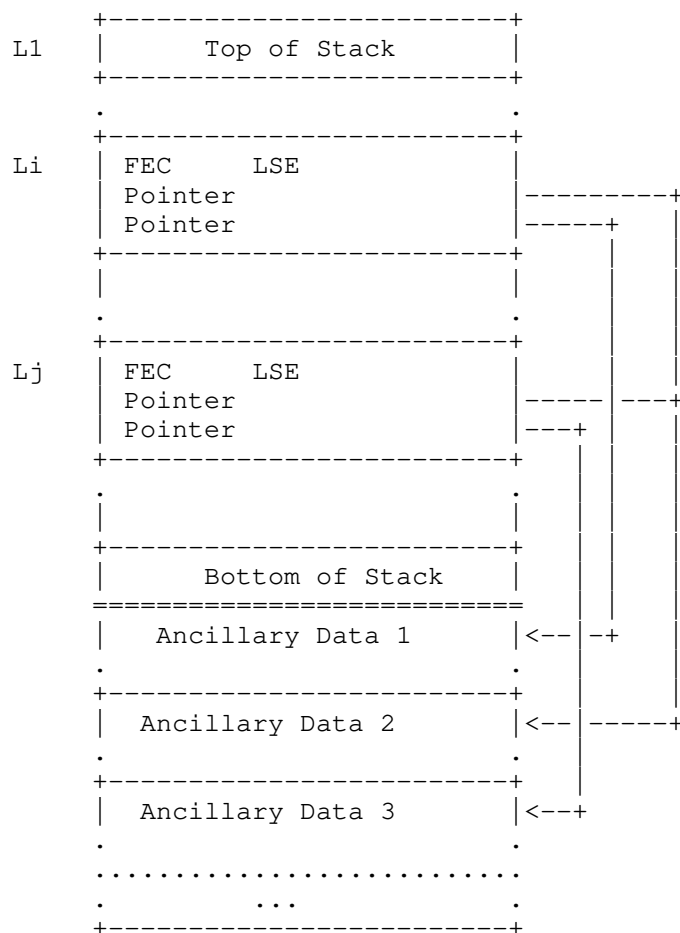


Figure 5: Further Example of Multiple In-stack MPLS Pointers

To support multiple Pointer SPLs, the following additional considerations apply:

The pop operation for the ToS needs to be extended to apply to the entire tuple of Pointer SPLs that are in its scope, i.e. that immediately succeed it. The default pop behavior will be to pop the entire tuple of Pointer SPLs along with the ToS.

As described earlier, as an optimization to reduce the size of the label stack, Pointer SPLs can be designated to not be popped but instead swapped with other LSEs in the stack. This will allow the same Pointer SPL set to be applied to multiple LSEs.

For an in stack swap operation where multiple pointers form a pointer set, the entire tuple, or group, would be swapped with the next FEC label below.

In addition, mixed cases are conceivable in which some Pointer SPLs are popped whereas others are swapped. Whether to pop or swap a Pointer SPL needs to be specified as part of the associated LSE's disposition behavior.

## 6. Disposition of the Ancillary Data

The ancillary data must be removed before the payload is passed out of the MPLS domain. There are three methods whereby the egress PE can know of the presence of the ancillary data:

- o The FEC of the BoS LSE can indicate the need to do this in a manner similar to pseudowires or MPLS VPN.
- o The BoS LSE can be a special purpose label indicating the presence of the ancillary data.
- o The BOS LSE can point to an item of ancillary data that describes the disposition of the ancillary data.

The removal of the ancillary data may be relatively complex depending on its purpose, i.e. it may be more complex than removing some number of bytes, for example, if it is carrying latency or iOAM information.

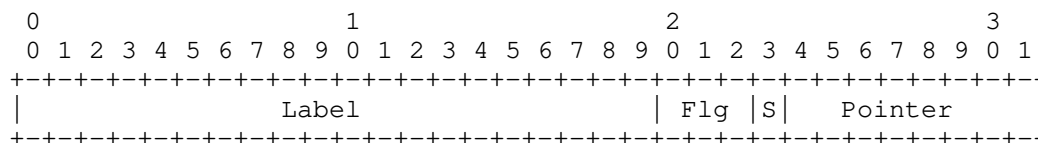
The structure and quantity of ancillary data including any methods whereby the ancillary data points to other ancillary, or whether there are pointer to the payload itself is out of scope for this document. Such information will need to be included in the ancillary data design so that it can be safely processed and/or removed.

## 7. Structure of Ancillary Data

The structure of the ancillary data is outside the scope of this memo.

## 8. Structure of Pointer Label

A possible structure for a pointer LSE is shown in Figure 6.



Label : contains the label that triggers the pointer behavior

Flg (Flags): Contains a number of flags that clarify the pointer

Bit 20: Size of pointer units, Bit 20 = 0 units are octets,  
Bit = 1 units are 16 bit quantities

S : BoS as per {{RFC3032}}

Pointer : Pointer to the start of the specific ancillary data  
block.

Figure 6: MPLS Pointer LSE

The label is recognized by the forwarder as being the trigger for the pointer behavior. The pointer is the offset from the pointer LSE to the start of the auxiliary data that is to be used at this hop to process the ToS label. The pointer may be in units of octets or 16 bit words (or 32 bit words TBD) as specified by the flag. The S bit had its normal meaning in an MPLS LSE.

## 9. Backward Compatibility

If the LSP includes a legacy node that does not understand the pointer SPL it will forward based on the FEC of the ToS alone omitting the feature. If that feature absence results in a service shortfall for traffic on the LSP or MPLS-SR path then obviously the LSP or path has to be constructed to avoid any node that is unable to execute the feature. The various methods of constructing LSPs via LSRs with certain capabilities is well known routing technology and will not be further discussed in this memo.

## 10. Security Considerations

This proposal does not change the security of the MPLS data plane. Normal operational practice is to prohibit the ingress of an MPLS packet from other than a trusted source. An attacker that breaches the physical security of an MPLS domain has many methods of attack by manipulating the label stack, and this mechanism does not significantly increase that risk.

## 11. IANA Considerations

This document has no IANA requests.

## 12. Appendix 1: CONTROVERSY ALERT - Use Of Ordinary Labels

Given the restricted number of Base SPLs [RFC9017] it is interesting to consider whether we might use an ordinary label for this purpose. At the time of writing there are eight out of 16 base SPLs still available. This is a dilemma since there is a protocol need for single label SPLs to support MPLS stack efficiency, but those that we have available must last the development lifetime of MPLS. On the other hand using a non-SPL has potential run-time/hardware issues if we need lots of them. However there probably exists a compromise number where we can safely allocate Base SPLs but not significantly impact the forwarder performance with this approach.

The label becomes a run-time constant that the forwarder needs to check during the parsing of the label stack. This is a 20 bit compare of a run-time constant. This is simple for a software or microcoded forwarder but needs a programmable register in a fully hardware based forwarder. Clearly from a protocol design perspective it is necessary to check the restrictions on the deployed hardware, but this certainly seems feasible. In deployment it will of course be necessary to verify that the routers along the LSP can support this feature before the LSP can be constructed.

If an ordinary label were assigned to this purpose from the 16-104857516 label set, there are two cases to consider: LSRs that have the capability of associating a FEC with a label of this value and LSRs that do not.

If an LSR has the capability to allocate a FEC with the chosen value it is necessary to preallocate this label before any MPLS application takes that value. This may impact a number of MPLS applications, but it seems feasible.

An LSR that does not have the capability to allocate a FEC with this value simply has the issue of adapting the forwarding behavior.

The matter of choosing a suitable value and distributing it is outside the scope of this memo, but is something that is routinely done in the routing system so is not a factor in assessing feasibility.

Clearly the LSP needs to be constructed so as to avoid any LSR that is unable to process a packet with one of these sequestered ordinary labels, but that is no different to the case where an SPL is used.

The final consideration is what happens if the label every becomes ToS. For this to happen the packet must have been incorrectly processed, and that is no different from any other case of a incorrectly processed MPLS packet.

### 13. Appendix 2: Other Issues for Discussion

This appendix briefly describes a number of issue that require further consideration.

1) Pointer labels as described earlier in this document are defined using an offset that is calculated from the pointer label. An alternative approach, given that we may not get rid of scanning for the BoS in MPLS header parsing, is to consider a design in which offsets were relative to the BoS instead. We could use this as a standard method, or we could specify this via a flag in the LSE carrying the offset.

The relative to BoS relative approach can be more efficient in some circumstances, e.g.: When the ancillary data is applicable to multiple hops of a label stack that is indicating a steering path, such as in SR-MPLS, the FEC of every steering hop label could indicate to "reuse" the ancillary data for every hop. The MPLS operation would then consist of a pop of the ToS label followed by a swap of the two top labels with each other, so that the following steering label effectively gets pulled up as ToS.

2) To support pointers being valid across multiple hops, the pointer either needs to be indicated either as an offset relative to BoS, or the value of the pointer in the pointer-SPL needs to be adjusted by a swap operation.

3) The LSE pointer could include a lifetime indicating the number of times it is to be propagated. This raises the following issue.

4) The pointer mechanism has to be able to deal with multiple instances of ancillary data applicable to specific elements within the LSP. So it is important to know when to stop propagating any pointer if that approach is adopted (instead of, for example adopting an approach of one pointer LSE per ToS label.

5) We need to decide of correct name for pointer SPL.

### 14. Appendix 3: Ancillary vs Auxiliary vs Metadata

From the Oxford English Dictionary:



- o Ancillary: Designating activities and services that provide essential support to the functioning of a central service or industry;
- o Auxiliary: Helpful, assistant, affording aid, rendering assistance, giving support or succour(sic).
- o Metadata(sic): data whose purpose is to describe and give information about other data.

The two terms ancillary and auxiliary are similar but the additional qualifier that ancillary is `_essential_` support make it, in the author's view, the preferred term.

Metadata, the term often used in the technical discussions does appear to be sufficiently descriptive of the purpose of this information that is included in the packet.

## 15. References

### 15.1. Normative References

- [RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <<https://www.rfc-editor.org/info/rfc3032>>.
- [RFC9017] Andersson, L., Kompella, K., and A. Farrel, "Special-Purpose Label Terminology", RFC 9017, DOI 10.17487/RFC9017, April 2021, <<https://www.rfc-editor.org/info/rfc9017>>.

### 15.2. Informative References

- [I-D.kompella-mpls-mspl4fa] Kompella, K., Beeram, V. P., Saad, T., and I. Meilik, "Multi-purpose Special Purpose Label for Forwarding Actions", draft-kompella-mpls-mspl4fa-00 (work in progress), February 2021.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.

Authors' Addresses

Stewart Bryant  
Futurewei Technologies, Inc.  
Email: sb@stewartbryant.com

Alexander Clemm  
Futurewei Technologies, Inc.  
Email: ludwig@clemm.org

Toerless Eckert  
Futurewei Technologies, Inc.  
Email: tte@cs.fau.de

MPLS WG  
Internet-Draft  
Intended status: Standards Track  
Expires: 12 August 2022

K. Kompella  
V.P. Beeram  
T. Saad  
Juniper Networks  
I. Meilik  
Broadcom  
8 February 2022

Multi-purpose Special Purpose Label for Forwarding Actions  
draft-kompella-mpls-mspl4fa-02

Abstract

The MPLS architecture introduced Special Purpose Labels (SPLs) to indicate special forwarding actions and offered a few simple examples, such as Router Alert. In the two decades since the original architecture was crafted, the range, complexity and sheer number of such actions has grown; in addition, there now is need for "associated data" for some of the forwarding actions. Likewise, the capabilities and scale of forwarding engines has also improved vastly over the same time period. There is a pressing need to match the needs with the capabilities to deliver the next generation of MPLS architecture.

In this memo, we propose an alternate mechanism whereby a single SPL can encode multiple forwarding actions and carry associated data, some in the label stack and some after the label stack. This proposal also solves the problem of scarcity of base SPLs.

This approach can immediately address several use cases:

- \* to carry a Slice Selector for IETF network slicing;
- \* to signal that further fast reroute may have harmful consequences;
- \* to indicate that there is relevant data after the label stack;
- \* among others.

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 August 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 . . . . .                                         | 3  |
| 1.1. Conventions and Definitions . . . . .                        | 3  |
| 1.2. Revision History . . . . .                                   | 3  |
| 1.2.1. Changes from -00 to -01 . . . . .                          | 4  |
| 1.2.2. Changes from -01 to -02 . . . . .                          | 4  |
| 1.3. Slice Selector . . . . .                                     | 5  |
| 2. Multi-purpose bSPL: the Forwarding Actions Indicator . . . . . | 5  |
| 2.1. The FAI bSPL . . . . .                                       | 6  |
| 2.1.1. ISD vs PSD . . . . .                                       | 6  |
| 2.2. Format of the FAI bSPL . . . . .                             | 6  |
| 2.2.1. Definitions of the FAI Flag Bits . . . . .                 | 7  |
| 2.2.2. Processing the FAI Flags and the ISD . . . . .             | 9  |
| 2.2.3. Example of the FAI . . . . .                               | 9  |
| 3. Issues to be Resolved . . . . .                                | 10 |
| 3.1. Preventing FAI From Reaching Top of Stack . . . . .          | 10 |
| 3.2. Repeating the FAI at "Readable Stack Depth" . . . . .        | 11 |
| 3.3. PSD . . . . .                                                | 11 |
| 4. Contributors . . . . .                                         | 11 |
| 5. Acknowledgments . . . . .                                      | 12 |
| 6. IANA Considerations . . . . .                                  | 12 |

|                                       |    |
|---------------------------------------|----|
| 7. Security Considerations . . . . .  | 12 |
| 8. References . . . . .               | 12 |
| 8.1. Normative References . . . . .   | 12 |
| 8.2. Informative References . . . . . | 13 |
| Authors' Addresses . . . . .          | 13 |

## 1. Introduction

Base Special Purpose Labels (bSPLs) are a precious commodity; there are only 16 such values, of which 8 have already been allocated. There are currently five requests for bSPLs that the authors are aware of; this document proposes another use case for a bSPL, in all consuming nearly all the remaining values. This document suggests a method whereby a single bSPL can be used for all the purposes currently requested. This leads to perhaps the more valuable long-term contribution of this document: an approach to the definition and use of bSPLs (and SPLs in general) whereby a single value can be used for multiple purposes, and provide a flexible yet efficient means of carrying associated data.

### 1.1. Conventions and Definitions

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.

FAI: Forwarding Actions Indicator

FFB: Forwarding Flags Block

ISD: In-Stack Data

sISD: Standard ISD

uISD: User-Defined ISD

PSD: Post-Stack Data

SPL: Special-purpose label

bSPL: Base special-purpose label

### 1.2. Revision History

This section (to be removed before publication) offers highlights from the draft's revision history.

## 1.2.1. Changes from -00 to -01

1. This section added.
2. Added a section discussing when data should be put in the LS FAD vs in the PL FAD.
3. Tweaked the bits in the FAI. Added a field "edist".
4. Elaborated on the use of the H bit and the FAH data.
5. Updated the processing of the LS FAD.
6. Added processing of edist.
7. Updated the FAI example.
8. Updated the Issues section.

## 1.2.2. Changes from -01 to -02

1. Updated Abstract and Introduction to focus on FAI; moved description of use cases to separate section.
2. Added terminology.
3. Changed terminology: LS FAD and PL FAD to ISD and PSD, respectively.
4. Updated text on criteria for putting associated data in ISD.
5. Introduced the terms FAI Block, FFB Block, sISD Block and uISD Block. Introduced an "end of block" bit, s. Updated flag bits; updated processing of ISD.
6. Removed field edist.
7. Updated the section on preventing the FAI from reaching the Top of Stack.
8. Updated the section on Readable Stack Depth

### 1.3. Slice Selector

Network slicing is an important ongoing effort both for network design, as well as for standardization, in particular at the IETF [I-D.nsdtd-teas-ns-framework]. A key issue is identifying which slice a packet belongs to, by means of a "slice selector" carried in the packet header. [I-D.bestbar-teas-ns-packet] describes several such methods for MPLS networks, of which the Global Identifier for Slice Selector (GISS) is one of the more practical solutions. This document shows how to realize the GISS using a base special purpose label (bSPL).

In MPLS networks, a GISS is a data plane construct identifying packets belonging to a slice aggregate (the set of packets that belong to the slice). The GISS dictates forwarding actions for the slice aggregate: QoS behavior and next hop selection. The purpose of the GISS is detailed in [I-D.bestbar-teas-ns-packet]. To embed a GISS in a label stack, one must preface it with a bSPL identifying it as such. For reasons that will become apparent, this bSPL is called the Forwarding Actions Indicator (FAI).

## 2. Multi-purpose bSPL: the Forwarding Actions Indicator

This document proposes the use of a single bSPL to tell routers one or more forwarding actions they should take on a packet, e.g.:

- \* to treat a packet according to its slice, given its GISS;
- \* to load balance a packet, given its entropy;
- \* whether or not to perform fast reroute on a failure [I-D.kompella-mpls-nffrr];
- \* whether or not a packet has metadata relevant to intermediate hops along the path;
- \* and perhaps other functions in the future.

This bSPL is called the "Forwarding Actions Indicator" (FAI). There are other suggestions for this name, including "Network Functions Indicator" and "Network Actions Indicator". We'll let WG consensus determine the final choice of name, but for now, we'll continue to use FAI.

The FAI uses the label's TC bits and TTL field to inform the forwarding plane of the required actions. Each of these actions may have associated data. This data may be carried in the label stack as "In-Stack Data" (ISD) or after the label stack as "Post-Stack Data" (PSD).

## 2.1. The FAI bSPL

The design of the bSPL hinges on two key insights: forwarding engines do not interpret the TC bits or the TTL field for labels that are not at the top of the label stack (ToS); nor do they do so for SPLs. For non-ToS labels, the important bit fields are the label value field (to compute entropy and identify SPLs) and the End of Stack (S) bit (to know when the label stack ends). [If you know of a forwarding engine that looks at other bit fields of labels below the ToS, please contact the authors.] This means that for a bSPL that will never appear at the ToS, the TC bits and the TTL bits can be used to carry additional information. Furthermore, for the ISD, the entire 4-octet label word, the S bit excepted, can be used to carry data. We use this technique to make the FAI bSPL multipurpose, and to make the ISD words compact and efficient.

### 2.1.1. ISD vs PSD

A pertinent question is when one should put data in the ISD versus in the PSD. One alternative is to put all such data in the PSD. However, this would mean that accessing such information would require finding the End of Stack, and parsing the PSD. For certain types of data, this would be a severe burden on the packet forwarding engine. Examples of such data are the Entropy label (needed for efficient load balancing) and the GISS (needed for accurate packet forwarding). Having any of this data in the PSD would hurt forwarding performance.

This memo suggests that data that is required for accurate and optimal forwarding should be put in the ISD, and data that is optional from a forwarding point of view should be put in the PSD. Furthermore, each flag bit should have no more than one word of associated ISD. The EG flag can thus have up to 2 words of associated data.

By the above criteria, this memo suggests that in-situ OAM data and the Flow ID be carried in the PSD.

## 2.2. Format of the FAI bSPL



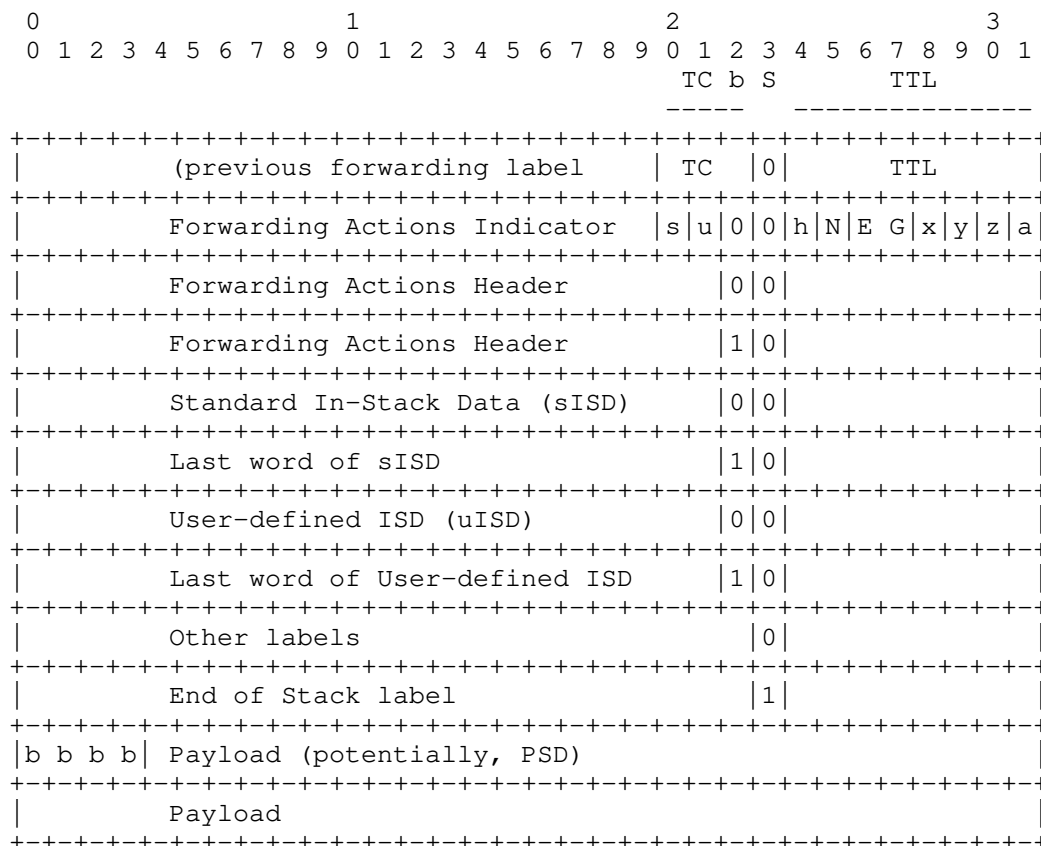


Figure 1: Format for FAI, ISD and PSD

The FAI's label value MUST be the IANA allocated value. The S bit MUST be reflect whether the label stack ends at this label or not.

#### 2.2.1. Definitions of the FAI Flag Bits

The TC and TTL bits are used as flags, defined as follows:

s: sISD is present (1) or not (0).

u: uISD is present (1) or not (0).

b: this is the "end of block" bit that indicates the end of the Forwarding Flags Block and the end of the ISD Block.

S: MUST be set if the FAI is the end of stack, and clear otherwise.

h: If set, the PSD contains hop-by-hop information. Every node in the path SHOULD attempt to process the hop-by-hop information, but not at the expense of exceeding the processing time budget, which could cause this (or other) packets to be dropped. If clear, no hop-by-hop data exists in the PSD: either the PSD is empty, or it contains only end-to-end data (to be processed by the egress).

N: If set, do not do fast reroute (NFFRR).

EG: this is a 2-bit flag indicating whether the ISD carries Entropy and/or GISS information.

The FAI Block consists of a Forwarding Flags Block, an sISD Block and a uISD Block. The two ISD Blocks are optional; their presence is indicated by the s and u bits. Each of these three blocks end when the b bit is set.

The Forwarding Flags Block extends from the FAI bSPL up to (and including) the first label that has the b bit set. If the FFB consists of just the bSPL, then its b bit must be set.

The sISD Block extends from the label after the FFB up to (and including) the label with the b bit set. If there is no sISD, the s bit in the FFB MUST be clear.

The uISD Block extends from the label after the sISD Block up to (and including) the label with the b bit set. If there is no uISD, the u bit in the FFB MUST be clear.

The EG field is used as follows:

00: No Entropy or GISS present

01: ISD 0 contains 16 bits of Entropy in the high order 16 bits and 14 bits of GISS in the low order 16 bits (S and b bits excepted).

10: ISD 0 contains 20 bits of Entropy in the high order 20 bits and 10 bits of GISS in the low order 12 bits (S and b bits excepted).

11: ISD 0 contains the 30-bit Entropy; ISD 1 contains the 30-bit GISS. In ISD 0, the S and b bits MUST be 0; the packet forwarding engine may choose to use the S and b bits as part of the Entropy, as it doesn't affect the outcome. In ISD 1, the S bit may be 0 or 1.

### 2.2.2. Processing the FAI Flags and the ISD

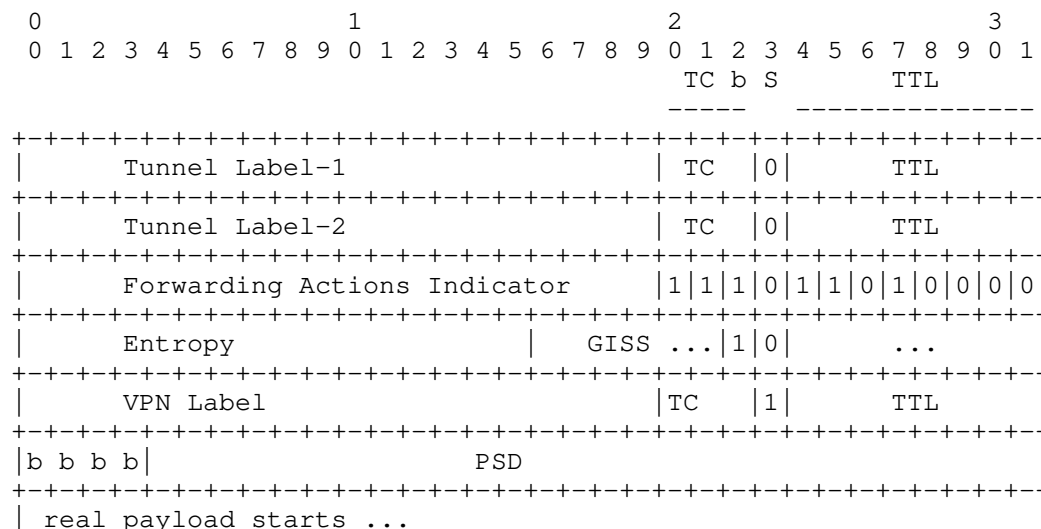
Here's how the Standard ISD is parsed. One must keep track of the s bit to know when the Standard ISD Block ends, and the S bit to know when the stack ends. The Standard ISD data appears in the order of the corresponding flags.

It is an error if the label stack ends while there are more ISD words to process. In particular, it is an error if the FAI's S bit is set, but the b bit is clear.

1. If s and u are both 0, done: there is no associated ISD.
2. Set CL ("current label") to the FAI label. LL is the last label (End of Stack); PL ("payload") is the first 4-octet word of the payload.
3. While b is clear:
  1. increment CL
4. Process N. CL is unchanged.
5. If s is set, Standard ISD is present: process standard flags.
  1. Process EG:
  2. If EG is 00, CL is unchanged.
  3. If EG is 01 or 10, increment CL. CL now contains both GISS and Entropy.
  4. If EG is 11, CL+1 contains Entropy; CL+2 contains GISS. Increment CL by 2.
  5. Process other standard data-bearing flags; increment CL by 1 for each.
6. If u is set, uISD is present.
  1. Process uISD until b is set.

Note that how the uISD is used is not defined here; this is up to the user. All that is included here is how a forwarding engine can tell where the uISD block ends.

### 2.2.3. Example of the FAI



s = 1: there is standard ISD.  
 u = 0: there is no user-defined ISD.  
 N = 1: NFFRR is set.  
 EG = 01: ISD 0 contains Entropy + GISS.  
 h = 1: There is hop-by-hop PSD.

Figure 2: Example of FAI + ISD + hop-by-hop PSD

The real payload starts after the PSD.

### 3. Issues to be Resolved

This section captures issues to be resolved, in this memo and others. As the issues are fixed, they should be removed from here; ideally, this section should be empty before publication.

#### 3.1. Preventing FAI From Reaching Top of Stack

As was said earlier, the FAI MUST NOT be at the top of stack, since its TC and TTL bits have been repurposed. There are two ways to prevent this. If an LSR X pops a label and the next label is the FAI, X can pop the FAI and all ISD words. This version of the memo introduces the "end-of-block" (s) bit, whereby a forwarding engine that knows the FAI can detect the entire FAI block, even if it doesn't know some of the flags. This can be used in conjunction with Section 3.2.

In case it is desired to preserve the FAI+FAD until the egress, X should push an explicit NULL (label value 0 or 2) onto the stack above the FAI, with the correct TC and TTL values.

Other options may be pursued; however, we believe this is an adequate resolution.

### 3.2. Repeating the FAI at "Readable Stack Depth"

For LSRs which cannot parse the entire label stack, or would prefer not to unless needed, it is possible to repeat the FAI at "readable stack depth" (rsd). Say the rsd is 10 labels, and the FAI block is 3 labels. Then, the FAI block can be repeated every 7 labels, allowing all forwarding engines in the path to process it. When a forwarding label is popped and the FAI block exposed, it is deleted in its entirety, since the same (or potentially different) FAI block is again within the rsd.

Note that the s or u bits set to 0 can be used to indicate that the corresponding ISD is absent. Only the last FAI would contain the full information, reducing the size of the label stack. However, in this case, LSRs that don't process the whole stack may not load balance less effectively, and potentially not adhere to the slice service level objectives.

Other options will be described in future versions of this document.

### 3.3. PSD

The format of the PSD, whether or not a Control Word is present, and handling of the first nibble, is outside the scope of this document. The FAI will not contain details about the contents of the PSD, besides the single flag on whether or not the PSD contains information relevant to (most) intermediate hops. It is assumed that another memo will document the format of the PSD, and that that memo will provide a means of parsing the PSD (e.g., a TLV structure) and thus determining its contents.

The PSD memo should also comment on the impact of processing the PSD on forwarding performance, especially in the case of hop-by-hop info.

## 4. Contributors

Many thanks to Colby Barth, Chandra Ramachandran and Srihari Sangli for their contributions to this draft.

## 5. Acknowledgments

We'd like to acknowledge the helpful discussions with Swamy SRK and folks from the Broadcom team on the impacts to existing and future forwarding engines.

The edist field was added thanks to Haoyu Song, who suggested the optimization to find End of Stack.

## 6. IANA Considerations

If this draft is deemed useful and adopted as a WG document, the authors request the allocation of a bSPL for the FAI. We suggest the early allocation of label 8 for this.

## 7. Security Considerations

A malicious or compromised LSR can insert the FAI and associated data into a label stack, preventing (for example) FRR from occurring. If so, protection will not kick in for failures that could have been protected, and there will be unnecessary packet loss. Similarly, inserting or removing a Fragmentation Header means that a packet's contents cannot be accurately reconstructed. Inserting or changing a GISS means that the packet will be misclassified, perhaps leaving or entering a high-value slice and causing damage.

## 8. References

### 8.1. Normative References

[I-D.bestbar-teas-ns-packet]

Saad, T., Beeram, V. P., Wen, B., Ceccarelli, D., Halpern, J., Peng, S., Chen, R., Liu, X., Contreras, L. M., Rokui, R., and L. Jalil, "Realizing Network Slices in IP/MPLS Networks", Work in Progress, Internet-Draft, draft-bestbar-teas-ns-packet-07, 11 January 2022, <<https://www.ietf.org/archive/id/draft-bestbar-teas-ns-packet-07.txt>>.

[I-D.kompella-mpls-nffrr]

Kompella, K. and W. Lin, "No Further Fast Reroute", Work in Progress, Internet-Draft, draft-kompella-mpls-nffrr-02, 12 July 2021, <<https://www.ietf.org/archive/id/draft-kompella-mpls-nffrr-02.txt>>.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

## 8.2. Informative References

- [I-D.nsd-t-teas-ns-framework]  
Gray, E. and J. Drake, "Framework for IETF Network Slices", Work in Progress, Internet-Draft, draft-nsd-t-teas-ns-framework-05, 2 February 2021, <<https://www.ietf.org/archive/id/draft-nsd-t-teas-ns-framework-05.txt>>.

### Authors' Addresses

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

Email: [kireeti.ietf@gmail.com](mailto:kireeti.ietf@gmail.com)

Vishnu Pavan Beeram  
Juniper Networks  
1133 Innovation Way  
Sunnyvale, CA 94089  
United States

Email: [vbeeram@juniper.net](mailto:vbeeram@juniper.net)

Tarek Saad  
Juniper Networks  
1133 Innovation Way  
Sunnyvale, CA 94089  
United States

Email: [tsaad@juniper.net](mailto:tsaad@juniper.net)

Israel Meilik  
Broadcom

Email: [israel.meilik@broadcom.com](mailto:israel.meilik@broadcom.com)



MPLS  
Internet-Draft  
Intended status: Informational  
Expires: 7 July 2022

H. Song, Ed.  
Futurewei Technologies  
Z. Li  
T. Zhou  
Huawei  
L. Andersson  
Bronze Dragon Consulting  
3 January 2022

Options for MPLS Extension Header Indicator  
draft-song-mpls-eh-indicator-04

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.

## 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. Dedicated Extension Header Label . . . . .                         | 3  |
| 2.1. Special Purpose Label . . . . .                                  | 3  |
| 2.2. Extension Label plus an Extended Special Purpose Label . . . . . | 4  |
| 3. Generic Associated Channel Extension . . . . .                     | 4  |
| 3.1. GAL and Associated Channel Header . . . . .                      | 4  |
| 3.2. GAL and a Different Nibble Value . . . . .                       | 5  |
| 4. Extend MPLS Entropy Label . . . . .                                | 6  |
| 5. Configured FEC Labels . . . . .                                    | 7  |
| 6. Summary . . . . .                                                  | 7  |
| 7. Considerations of EHI . . . . .                                    | 9  |
| 8. Security Considerations . . . . .                                  | 9  |
| 9. IANA Considerations . . . . .                                      | 9  |
| 10. Contributors . . . . .                                            | 9  |
| 11. Acknowledgments . . . . .                                         | 9  |
| 12. References . . . . .                                              | 9  |
| 12.1. Normative References . . . . .                                  | 9  |
| 12.2. Informative References . . . . .                                | 10 |
| Authors' Addresses . . . . .                                          | 10 |

## 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.

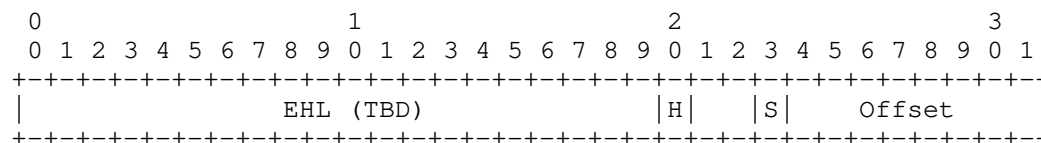


Figure 1: Special EHL with EH Category Encoding

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.

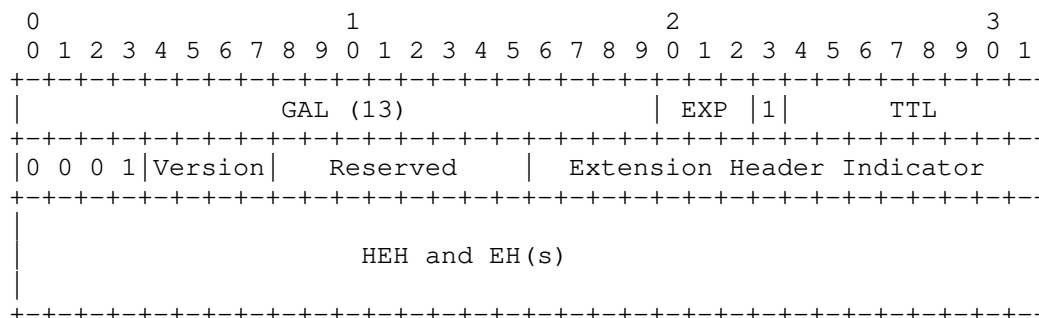


Figure 2: Associated Channel Header as Extension Header Indicator

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.

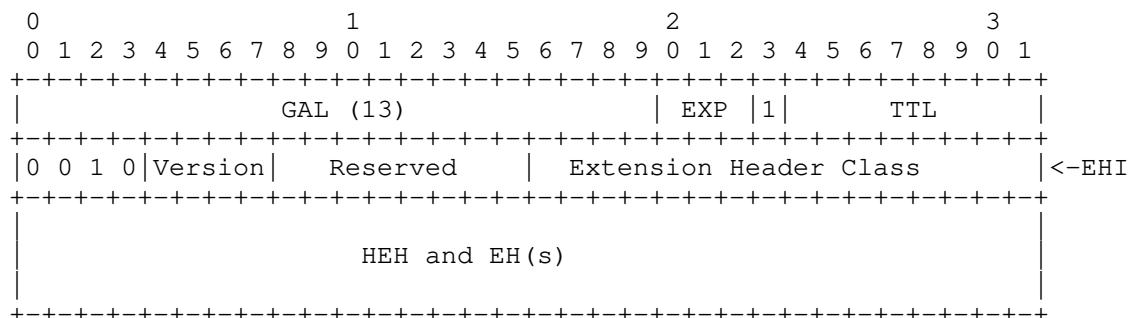


Figure 3: Extension Header Indicator Format

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.

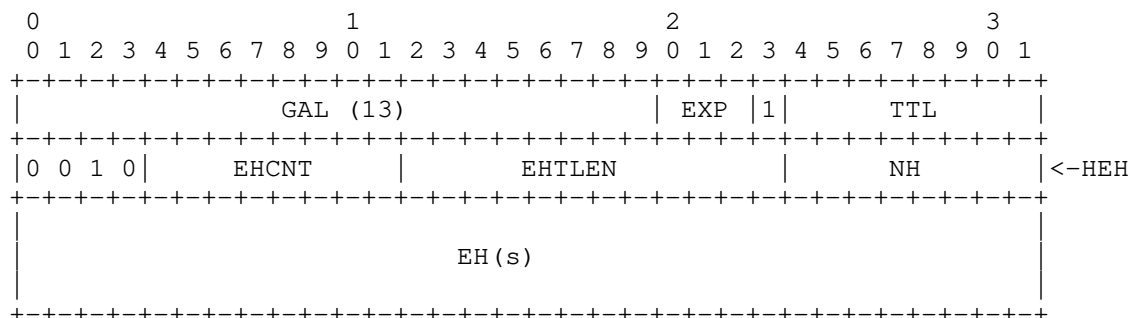


Figure 4: Merge HEH to EHI

#### 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.

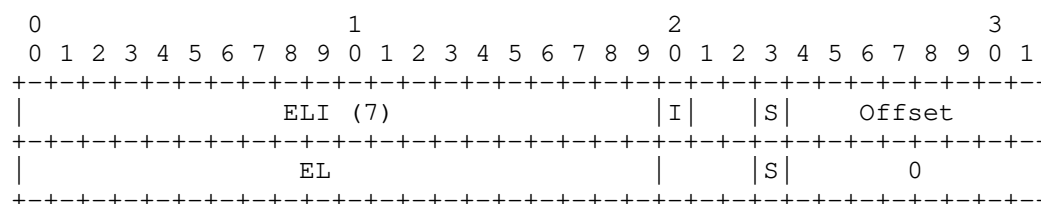


Figure 5: Special EHL with EH Category Encoding

## 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.

| No. | Description                                               | Pros and Cons                                                                                                                                                                                                                                                   |
|-----|-----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1   | Special purpose EHL                                       | <ul style="list-style-type: none"> <li>+ Single label</li> <li>+ Location freedom</li> <li>- Need standard extension</li> <li>- Use scarce resource</li> </ul>                                                                                                  |
| 2   | XL(15) + EHL                                              | <ul style="list-style-type: none"> <li>+ Location freedom</li> <li>+ Established mechanism</li> <li>+ Abundant resource</li> <li>- One extra label than Option 1</li> <li>- Need standard extension</li> </ul>                                                  |
| 3   | GAL + ACH with channel type extension                     | <ul style="list-style-type: none"> <li>+ Reuse existing mechanism</li> <li>+ Abundant resource</li> <li>- Label location limitation</li> <li>- One more word than Option 1</li> <li>- Not for user traffic</li> <li>- Need standard extension/update</li> </ul> |
| 4   | GAL + another nibble value to indicate EHs (e.g., "0010") | <ul style="list-style-type: none"> <li>+ No change to ACH semantics</li> <li>+ Potential overhead saving</li> <li>- Label location limitation</li> <li>- Hack scarce resource (nibble)</li> <li>- Need standard extension</li> </ul>                            |
| 5   | Extend ELI/EL                                             | <ul style="list-style-type: none"> <li>+ No need for new label</li> <li>- Need standard update</li> <li>- Need to make EL mandatory</li> <li>- One extra label than Option 1</li> </ul>                                                                         |
| 6   | FEC label as EH indicator                                 | <ul style="list-style-type: none"> <li>+ No need for header standard</li> <li>- Complex control plane</li> <li>- Cross-domain interoperability</li> <li>- Label space issue</li> <li>- Not for incremental deployment</li> </ul>                                |

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

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

- [RFC5586] Bocci, M., Ed., Vigoureux, M., Ed., and S. Bryant, Ed., "MPLS Generic Associated Channel", RFC 5586, DOI 10.17487/RFC5586, June 2009, <<https://www.rfc-editor.org/info/rfc5586>>.
- [RFC6790] Kompella, K., Drake, J., Amante, S., Henderickx, W., and L. Yong, "The Use of Entropy Labels in MPLS Forwarding", RFC 6790, DOI 10.17487/RFC6790, November 2012, <<https://www.rfc-editor.org/info/rfc6790>>.
- [RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC 7274, DOI 10.17487/RFC7274, June 2014, <<https://www.rfc-editor.org/info/rfc7274>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

## 12.2. Informative References

- [I-D.guichard-sfc-mpls-metadata]  
Guichard, J., Pignataro, C., Spraggs, S., and S. Bryant, "Carrying Metadata in MPLS Networks", Work in Progress, Internet-Draft, draft-guichard-sfc-mpls-metadata-00, 27 September 2013, <<https://www.ietf.org/archive/id/draft-guichard-sfc-mpls-metadata-00.txt>>.
- [I-D.song-mpls-extension-header]  
Song, H., Li, Z., Zhou, T., Andersson, L., and Z. Zhang, "MPLS Extension Header", Work in Progress, Internet-Draft, draft-song-mpls-extension-header-05, 10 July 2021, <<https://www.ietf.org/archive/id/draft-song-mpls-extension-header-05.txt>>.

## Authors' Addresses

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

Email: [haoyu.song@futurewei.com](mailto:haoyu.song@futurewei.com)

Zhenbin Li  
Huawei  
156 Beiqing Road  
Beijing, 100095  
P.R. China

Email: lizhenbin@huawei.com

Tianran Zhou  
Huawei  
156 Beiqing Road  
Beijing, 100095  
P.R. China

Email: zhoutianran@huawei.com

Loa Andersson  
Bronze Dragon Consulting  
Stockholm  
Sweden

Email: loa@pi.nu

MPLS  
Internet-Draft  
Intended status: Standards Track  
Expires: 14 July 2022

H. Song, Ed.  
Futurewei Technologies  
Z. Li  
T. Zhou  
Huawei  
L. Andersson  
Bronze Dragon Consulting  
Z. Zhang  
Juniper Networks  
10 January 2022

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."

This Internet-Draft will expire on 14 July 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. Motivation . . . . .                          | 2  |
| 2. MPLS Extension Header . . . . .               | 4  |
| 3. Type of MPLS Extension Headers . . . . .      | 8  |
| 4. Operation on MPLS Extension Headers . . . . . | 8  |
| 5. Use Cases . . . . .                           | 9  |
| 6. Security Considerations . . . . .             | 9  |
| 7. IANA Considerations . . . . .                 | 10 |
| 8. Contributors . . . . .                        | 10 |
| 9. Acknowledgments . . . . .                     | 10 |
| 10. References . . . . .                         | 10 |
| 10.1. Normative References . . . . .             | 10 |
| 10.2. Informative References . . . . .           | 11 |
| Authors' Addresses . . . . .                     | 12 |

#### 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.

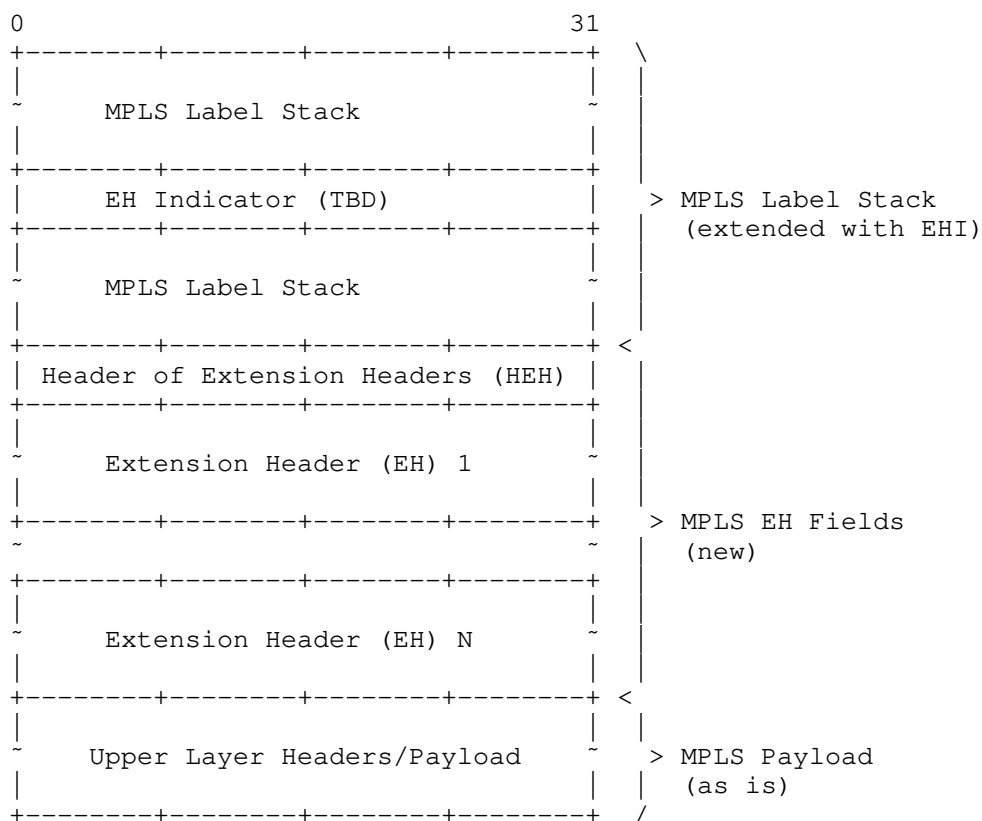


Figure 1: MPLS with Extension Headers

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.

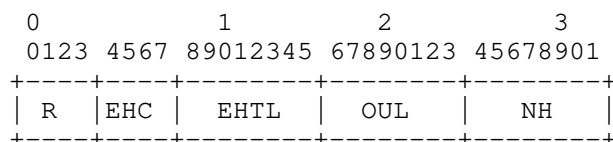


Figure 2: HEH Format

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.

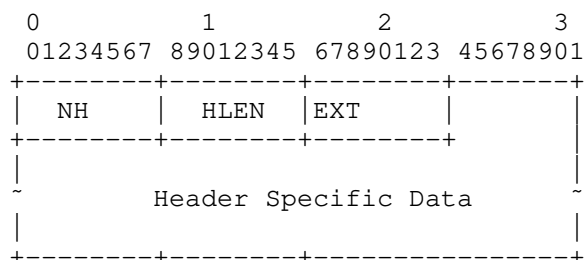


Figure 3: EH Format

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

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <<https://www.rfc-editor.org/info/rfc8321>>.

- [RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <<https://www.rfc-editor.org/info/rfc8754>>.

## 10.2. Informative References

- [I-D.brockners-inband-oam-transport]  
Brockners, F., Bhandari, S., Govindan, V. P., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., and R. Chang, "Encapsulations for In-situ OAM Data", Work in Progress, Internet-Draft, draft-brockners-inband-oam-transport-05, 3 July 2017, <<https://www.ietf.org/archive/id/draft-brockners-inband-oam-transport-05.txt>>.
- [I-D.ietf-ippm-ioam-data]  
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields for In-situ OAM", Work in Progress, Internet-Draft, draft-ietf-ippm-ioam-data-17, 13 December 2021, <<https://www.ietf.org/archive/id/draft-ietf-ippm-ioam-data-17.txt>>.
- [I-D.ietf-spring-srv6-network-programming]  
Filsfils, C., Garvia, P. C., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", Work in Progress, Internet-Draft, draft-ietf-spring-srv6-network-programming-28, 29 December 2020, <<https://www.ietf.org/archive/id/draft-ietf-spring-srv6-network-programming-28.txt>>.
- [I-D.song-mpls-eh-indicator]  
Song, H., Li, Z., Zhou, T., and L. Andersson, "Options for MPLS Extension Header Indicator", Work in Progress, Internet-Draft, draft-song-mpls-eh-indicator-04, 3 January 2022, <<https://www.ietf.org/archive/id/draft-song-mpls-eh-indicator-04.txt>>.
- [I-D.xu-clad-spring-sr-service-chaining]  
Clad, F., Xu, X., Filsfils, C., Bernier, D., Decraene, B., Yadlapalli, C., Henderickx, W., Salsano, S., and S. Ma, "Segment Routing for Service Chaining", Work in Progress, Internet-Draft, draft-xu-clad-spring-sr-service-chaining-00, 22 December 2017, <<https://www.ietf.org/archive/id/draft-xu-clad-spring-sr-service-chaining-00.txt>>.

- [RFC5586] Bocci, M., Ed., Vigoureux, M., Ed., and S. Bryant, Ed.,  
"MPLS Generic Associated Channel", RFC 5586,  
DOI 10.17487/RFC5586, June 2009,  
<<https://www.rfc-editor.org/info/rfc5586>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6  
(IPv6) Specification", STD 86, RFC 8200,  
DOI 10.17487/RFC8200, July 2017,  
<<https://www.rfc-editor.org/info/rfc8200>>.

## Authors' Addresses

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

Zhenbin Li  
Huawei  
Beijing  
P.R. China  
  
Email: [lizhenbin@huawei.com](mailto:lizhenbin@huawei.com)

Tianran Zhou  
Huawei  
Beijing  
P.R. China  
  
Email: [zhoutianran@huawei.com](mailto:zhoutianran@huawei.com)

Loa Andersson  
Bronze Dragon Consulting  
Stockholm  
Sweden  
  
Email: [loa@pi.nu](mailto:loa@pi.nu)

Zhaohui Zhang  
Juniper Networks  
Boston,  
United States of America  
  
Email: zzhang@juniper.net



intarea  
Internet-Draft  
Intended status: Standards Track  
Expires: February 26, 2022

Z. Zhang  
R. Bonica  
K. Kompella  
Juniper Networks  
G. Mirsky  
ZTE  
August 25, 2021

Generic Delivery Functions  
draft-zzhang-intarea-generic-delivery-functions-02

Abstract

Some functionalities (e.g., fragmentation/reassembly and Encapsulating Security Payload) provided by IPv6 can be viewed as delivery functions independent of IPv6 or even IP entirely. This document proposes to provide those functionalities at different layers (e.g., MPLS, BIER or even Ethernet) independent of IP.

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 February 10, 2022.

Copyright Notice

Copyright (c) 2021 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 |
| 2. Specifications . . . . .                              | 4 |
| 2.1. MPLS layer . . . . .                                | 4 |
| 2.2. BIER layer . . . . .                                | 5 |
| 2.3. Other layers . . . . .                              | 5 |
| 2.4. Generic Fragmentation Header (GFH) {#gfh} . . . . . | 6 |
| 3. Security Considerations . . . . .                     | 6 |
| 4. Acknowledgements . . . . .                            | 6 |
| 5. References . . . . .                                  | 7 |
| 5.1. Normative References . . . . .                      | 7 |
| 5.2. Informative References . . . . .                    | 7 |
| Authors' Addresses . . . . .                             | 8 |

## 1. Introduction

Consider an operator providing Ethernet services such as EVPN. The Ethernet frames that a Provider Edge (PE) device receives from a Customer Edge (CE) device may have a larger size than the PE-PE path MTU (PMTU) in the provider network. This could be because

1. the provider network is built upon virtual connections (e.g., pseudowires) provided by another infrastructure provider, or
2. the customer network uses jumbo frames while the provider network does not, or
3. the provider-side overhead for transporting customer packets across the network pushes past the PMTU.

In any case, the provider cannot simply require its customers to change their MTU.

To get those large frames across the provider network, currently, the only workaround is to encapsulate the frames in IP (with or without GRE) and then fragment the IP packets. Even if MPLS is used for service delimiting, IP is used for transportation (MPLS over IP/GRE). This may not be desirable in certain deployment scenarios, where MPLS is the preferred transport or IP encapsulation overhead is deemed excessive.

IPv6 fragmentation and reassembly are based on the IPv6 Fragmentation header below [RFC8200]:

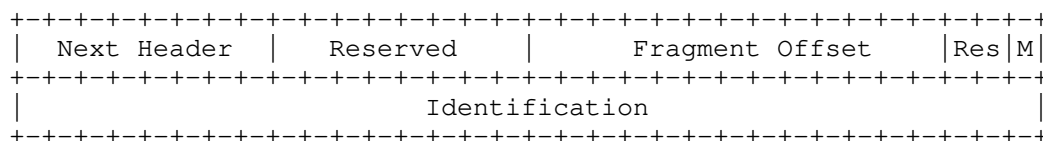


Figure 1: IPv6 Fragmentation Header

This document proposes adapting this header for use in non-IP contexts since the fragmentation/reassembly function is actually independent of IPv6 except for the following aspects:

- o The fragment header is identified as such by the "previous" header.
- o The "Next Header" value is from the "Internet Protocol Numbers" registry.
- o The "Identification" value is unique in the (source, destination) context provided by the IPv6 header.

The "Identification" field, in conjunction with the IPv6 source and destination addresses identifies fragments of the original packet for the purpose of reassembly.

Therefore, the fragmentation/reassembly function can be applied at other layers as long as a) the fragment header is identified as such; and b) the context for packet identification is provided. Examples of such layers include MPLS, BIER, and Ethernet (if IEEE determines it is so desired).

For the same consideration, the IP Encapsulating Security Payload (ESP) [RFC4303] could also be applied at other layers if ESP is desired there. For example, if for whatever reason the Ethernet service provider wants to provide ESP between its PEs, it could do so without requiring IP encapsulation if ESP is applied at non-IP layers.

Similarly, In-Situ OAM (IOAM) functions [I-D.ietf-ippm-ioam-data] can also be applied to many layers.

We refer to these as Generic Delivery Functions (GDFs), which could be achieved at a shim layer between a source and destination delivery points, for example:

- o Source and destination IP/Ethernet nodes
- o Ingress and egress nodes of MPLS Label Switch Paths (LSPs)

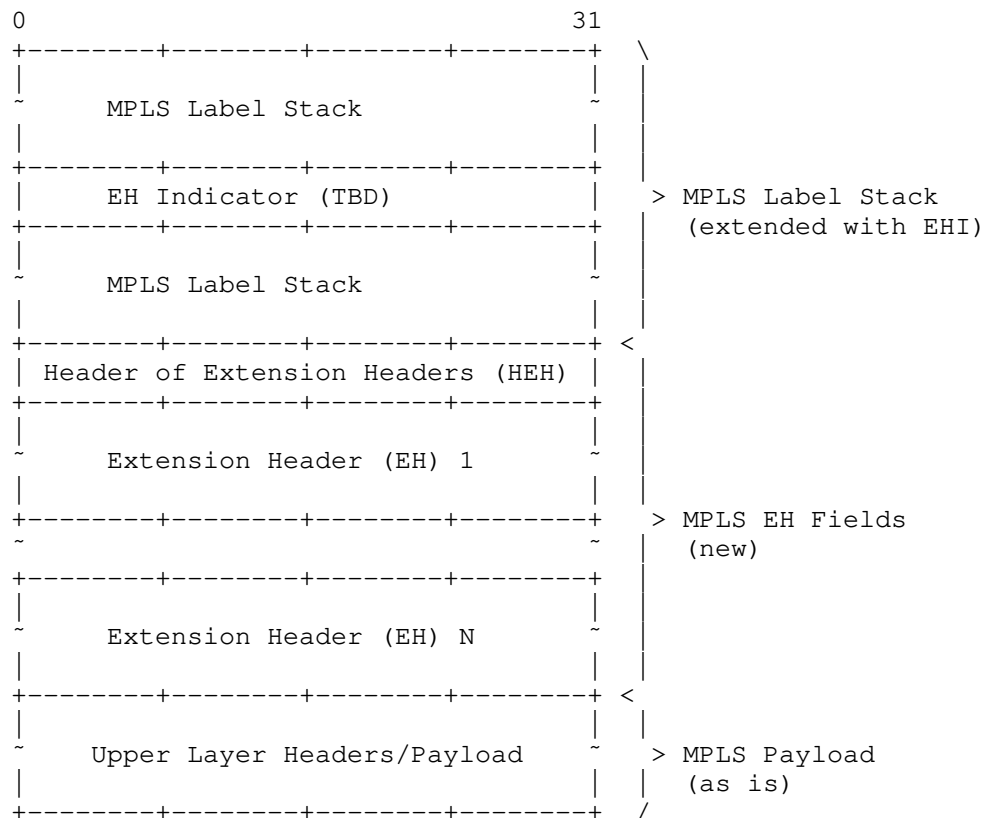
- o BIER Forwarding Ingress Routers (BFIRs) and BIER Forwarding Egress Routers (BFERs)

## 2. Specifications

A Generic Delivery Function, being generic, is likely applicable to IP as well. Therefore, IPv6 Extension Headers (for some GDFs) are directly used at other layers.

### 2.1. MPLS layer

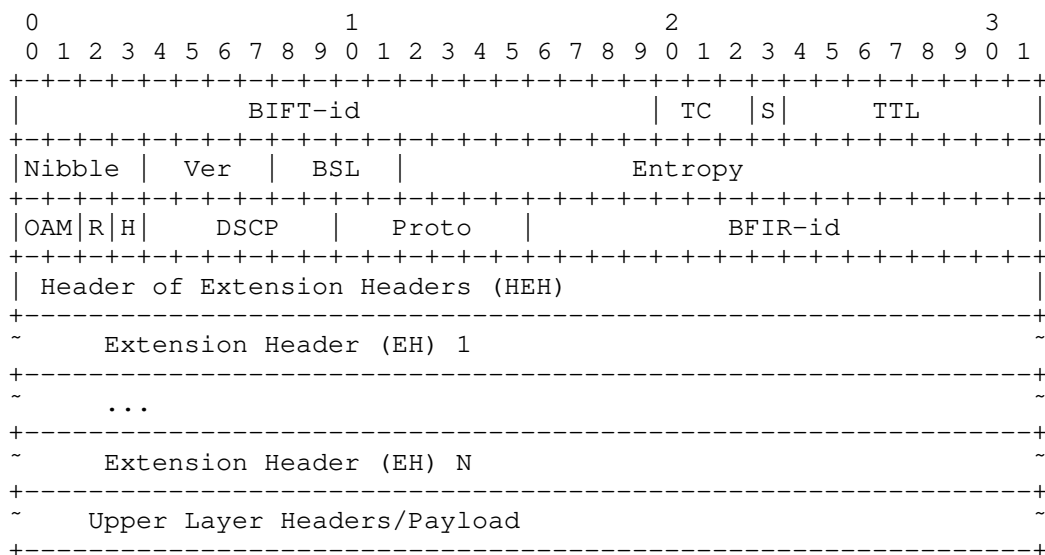
[I-D.song-mpls-extension-header] specifies MPLS Extension Headers encoding. A label entry in the stack indicates the presence of extension headers after the label stack. It starts with a Header of Extension Headers, as depicted in the following excerpt from that specification:



One or more of the EHs in the above can be an IPv6 Extension Header for a GDF.

## 2.2. BIER layer

For BIER layer, a TBD value for the "proto" field in the outer BIER header indicates that some BIER Extension Headers follow the BIER header, including some IPv6 Extension Headers for GDFs.



R: The "R" flag bit is reserved. It MUST be set to 0 on transmit and ignored on receive.

H: If the "H" flag bit, it indicates the presence of at least one extension header that needs to be processed hop by hop even before a BFER is reached. In this case, the Proto field must be set to the TBD value indicating the presence of extension headers.

## 2.3. Other layers

Similarly, any layer can have an indication in its packet header that some GDF extension headers follow, including some IPv6 Extension Headers for GDF purpose.

For example, if the outer header is Ethernet (if IEEE would decide to provide the generic delivery functions on top of Ethernet directly), then a new Ethertype would be assigned by IEEE to indicate the presence of GDF extension headers.

## 2.4. Generic Fragmentation Header (GFH){#gfh}

For generic fragmentation/reassembly functionality, the existing IPv6 Fragment Header needs to be enhanced for MPLS as following:

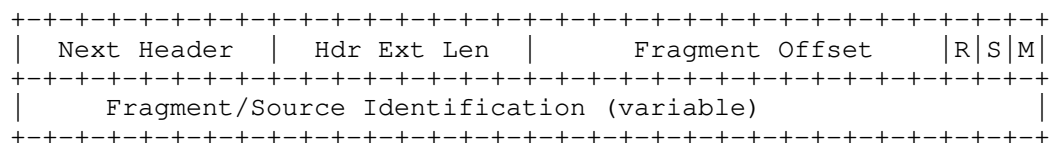


Figure 2: Generic Fragmentation Header

R: The "R" flag bit is reserved. It MUST be set to 0 on transmit and ignored on receive.

S: If the "S" flag bit is clear, the context for the Identification field is provided by the outer header, and only the source-identifying information in the outer header is used.

If the "S" flag bit is set, the variable Identification field encodes both source-identifying information (e.g., the IP address of the node adding the GFH) and an identification number unique within that source. The length of the Fragment header is encoded in the 8-bit "Hdr Ext Len" field (which is a Reserved field in the original IPv6 Fragment Header).

When a GFH is used together with other GDF Headers (GDFH), the GFH SHOULD be the first GDFH.

The above enhancement is not necessary but MAY be used for BIER as well. If the outer header is BIER and the "S" flag bit is clear, the "BFIR-id" field in the BIER header provides the context for the "Identification" field. If the bit is set, then the source information embedded in the source/fragment identification field is used.

## 3. Security Considerations

To be provided.

## 4. Acknowledgements

The authors thank Stewart Bryant and Tony Przygienda for their valuable comments and suggestions.

## 5. References

### 5.1. Normative References

- [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-05 (work in progress), July 2021.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi  
Topology (MT) Routing in Intermediate System to  
Intermediate Systems (IS-IS)", RFC 5120,  
DOI 10.17487/RFC5120, February 2008,  
<<https://www.rfc-editor.org/info/rfc5120>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic  
Engineering", RFC 5305, DOI 10.17487/RFC5305, October  
2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308,  
DOI 10.17487/RFC5308, October 2008,  
<<https://www.rfc-editor.org/info/rfc5308>>.
- [RFC7684] Psenak, P., Gredler, H., Shakir, R., Henderickx, W.,  
Tantsura, J., and A. Lindem, "OSPFv2 Prefix/Link Attribute  
Advertisement", RFC 7684, DOI 10.17487/RFC7684, November  
2015, <<https://www.rfc-editor.org/info/rfc7684>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6  
(IPv6) Specification", STD 86, RFC 8200,  
DOI 10.17487/RFC8200, July 2017,  
<<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8362] Lindem, A., Roy, A., Goethals, D., Reddy Vallem, V., and  
F. Baker, "OSPFv3 Link State Advertisement (LSA)  
Extensibility", RFC 8362, DOI 10.17487/RFC8362, April  
2018, <<https://www.rfc-editor.org/info/rfc8362>>.

### 5.2. Informative References

- [I-D.ietf-ippm-ioam-data]  
Brockners, F., Bhandari, S., and T. Mizrahi, "Data Fields  
for In-situ OAM", draft-ietf-ippm-ioam-data-14 (work in  
progress), June 2021.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)",  
RFC 4303, DOI 10.17487/RFC4303, December 2005,  
<<https://www.rfc-editor.org/info/rfc4303>>.

[RFC6514] Aggarwal, R., Rosen, E., Morin, T., and Y. Rekhter, "BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs", RFC 6514, DOI 10.17487/RFC6514, February 2012, <<https://www.rfc-editor.org/info/rfc6514>>.

#### Authors' Addresses

Zhaohui Zhang  
Juniper Networks

Email: [zzhang@juniper.net](mailto:zzhang@juniper.net)

Ron Bonica  
Juniper Networks

Email: [rbonica@juniper.net](mailto:rbonica@juniper.net)

Kireeti Kompella  
Juniper Networks

Email: [kireeti@juniper.net](mailto:kireeti@juniper.net)

Gregory Mirsky  
ZTE

Email: [gregory.mirsky@ztetx.com](mailto:gregory.mirsky@ztetx.com)