Abstract

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

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 7 January 2023.

Copyright Notice

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

In-situ Operations, Administration, and Maintenance (IOAM) is used for recording and collecting operational and telemetry information while the packet traverses a path between two points in the network. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within the probe packets specifically dedicated to OAM. The IOAM data fields are
defined in [RFC9197]. The IOAM data fields are further updated in
[I-D.ietf-ippm-ioam-direct-export] for direct export use-cases.

This document defines how IOAM data fields are transported with MPLS
data plane encapsulations using MPLS Network Action (MNA) with new
Generic Associated Channel (G-ACh) and updates the [RFC5586].

2. Conventions

2.1. Requirement Language

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

2.2. Abbreviations

The MPLS Network Action (MNA) terminology defined in
[I-D.andersson-mpls-mna-fwk] are used in this document.

Abbreviations used in this document:

PNI Post-Stack Network Action Presence Indicator

ECMP Equal Cost Multi-Path

E2E Edge-To-Edge

G-ACh Generic Associated Channel

HBH Hop-By-Hop

HPI Hop-By-Hop Post-Stack Network Action Processing Indicator

IOAM In-situ Operations, Administration, and Maintenance

MPLS Multiprotocol Label Switching

MNA MPLS Network Action
3. MPLS Extensions for IOAM Data Fields

3.1. IOAM Generic Associated Channel

The IOAM header is added containing different IOAM-Data-Fields in the MPLS header as shown in Figure 1. The IOAM-Data-Fields MUST follow the definitions corresponding to IOAM-Option-Types (e.g. see Section 4.4 of [RFC9197] and Section 3 of [I-D.ietf-ippm-ioam-direct-export]). More than one trace options can be present in the IOAM-Data-Fields.

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

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

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

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

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

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

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

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

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

**IOAM-OPT-Type:** 8-bit field defining the IOAM Option type, as defined in the "IOAM Option-Type Registry" specified in [RFC9197].
IOAM HDR Length: 8-bit unsigned integer. Length of the IOAM Header in 4-octet units.

IOAM Option and Data Space: IOAM-Data-Fields as specified by the IOAM-OPT-Type field. IOAM-Data-Fields are defined corresponding to the IOAM-Option-Type (e.g. see Section 4.4 of [RFC9197] and Section 3 of [I-D.ietf-ippm-ioam-direct-export]).

3.2. IOAM Indicators

A Post-Stack Network Action Presence Indicator MUST be added to indicate the presence of the IOAM-Data-Fields in the MPLS header. If both edge and intermediate nodes need to process IOAM data then both Presence Indicator and HBH Network Action Processing Indicator MUST be added. The HBH Processing Indicator allows to optimize the IOAM processing on intermediate nodes and avoids the need to parse all IOAM-Data-Fields.

A flag called PNI (Post-Stack Network Action Presence Indicator) in the TTL field defined in [I-D.jags-mpls-mna hdr] is used in this document to indicate the presence of IOAM Post-Stack Network Action and Ancillary Data. A flag called HPI (Hop-By-Hop Post-Stack Network Action Processing Indicator) in the TTL field defined in [I-D.jags-mpls-mna hdr] is used to indicate that HBH processing in Post-Stack Network Action and Ancillary Data is required.

The MNA Label used in this document is a Special Purpose Label (value TBA2) assigned by IANA, and is defined as Network Action Sub-Stack Indicator (NASI) in [I-D.jags-mpls-mna hdr].

4. Edge-to-Edge IOAM

4.1. Post-Stack Network Action Presence Indicator

The Post-Stack Network Action Presence Indicator is used to indicate the presence of the IOAM-Data-Fields in the MPLS header as shown in Figure 2.
The E2E IOAM-Data-Fields carry the Option-Type(s) that require processing on the encapsulating and decapsulating nodes only. The IOAM Option-Type carried can be IOAM Edge-to-Edge Option-Type [RFC9197]. The E2E IOAM-Data-Fields SHOULD NOT carry any IOAM Option-Type that require IOAM processing on the intermediate nodes as it will not be processed by them.

4.2. Procedure for Edge-to-Edge IOAM

The E2E IOM procedure is summarized as following:

* The encapsulating node inserts the MNA Label (SPL value TBA2) with the Post-Stack Network Action Presence Indicator (TTL Flag PNI) below the label whose FEC is the end (decapsulating) node and one or more IOAM-Data-Fields in the MPLS header.

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

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

* The decapsulating node MAY punt a copy of the packet with the receive timestamp to the slow path for IOAM-Data-Fields processing when the node recognizes the PNI. The receive timestamp is required by the various E2E OAM use-cases, including streaming telemetry. Note that the packet is not necessarily punted to the control-plane.
* The decapsulating node processes the IOAM-Data-Fields using the procedures defined in [RFC9197]. An example of IOAM processing is to export the IOAM-Data-Fields, send IOAM-Data-Fields via streaming telemetry, etc.

* The decapsulating node MUST remove the IOAM-Data-Fields from the received packet. The decapsulated packet is forwarded downstream or terminated locally similar to the regular IOAM-Data-Fields.

5. Hop-By-Hop IOAM

5.1. Hop-By-Hop Post-Stack Network Action Processing Indicator

The Post-Stack Network Action Presence Indicator (TTL Flag PNI) along with Hop-By-Hop Processing Indicator (TTL Flag HPI) are used to indicate the presence of the HBH IOAM-Data-Fields in the MPLS header as shown in Figure 3.

![Figure 3: Example MPLS Encapsulation for HBH IOAM](image)

The HBH IOAM-Data-Fields carry the Option-Type(s) that require processing at the intermediate and/or encapsulating and decapsulating nodes. The IOAM Option-Type carried can be IOAM Pre-allocated Trace Option-Type, IOAM Incremental Trace Option-Type and IOAM Proof of Transit (POT) Option-Type, as well as Edge-to-Edge Option-Type [RFC9197].

5.2. Procedure for Hop-By-Hop IOAM

The HBH IOAM procedure is summarized as following:
* The encapsulating node inserts the MNA Label with the Post-Stack Network Action Presence Indicator (TTL Flag PNI) and HBH Processing Indicator (TTL Flag HPI) below the label whose FEC is the end (decapsulating) node and one or more IOAM-Data-Fields in the MPLS header.

* The intermediate node enabled with HBH IOAM function processes the data packet including the IOAM-Data-Fields as defined in [RFC9197] when the node recognizes the HBH Processing Indicator in the MPLS header.

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

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

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

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

6. Considerations for IOAM

6.1. Considerations for ECMP

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

6.2. Node Capability

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

The intermediate node that is not capable of supporting the IOAM functions defined in this document, can simply skip the IOAM processing.
The intermediate node that does not understand the MNA Label received at the top of the label stack will drop the packet.

6.3. Nested MPLS Encapsulation

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

7. Security Considerations

The security considerations of IOAM in general are discussed in [RFC9197] and apply to the procedure defined in this document.

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

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

8. IANA Considerations

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

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

Table 1: IOAM G-ACh Type

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

The IOAM-Data-Fields, including IOAM G-ACh header are added in the MPLS encapsulation immediately after the MPLS header. Any Control Word [RFC4385] or another G-ACh [RFC5586] MUST be added after the IOAM-Data-Fields in the packet as shown in the Figure 4 and Figure 5, respectively. This allows the intermediate nodes to easily access the HBH IOAM-Data-Fields located immediately after the MPLS header. The decapsulating node can remove the MPLS encapsulation including the IOAM-Data-Fields and then process the Control Word or another G-ACh following it. The subsequent G-ACh and Control Word are located through the use of the "Length" field in the IOAM G-ACh.

Figure 4: Example MPLS Encapsulation with Generic PW Control Word with HBH IOAM
10. References

10.1. Normative References


10.2. Informative References

[I-D.andersson-mpls-mna-fwk]

[I-D.jags-mpls-mna-hdr]


Acknowledgements

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

Authors’ Addresses

Gandhi, et al.
Abstract

Multi-Topology Routing (MTR) is a technology to enable service differentiation within an IP network. Flexible Algorithm (FA) is another mechanism of creating a sub-topology within a topology using defined topology constraints and computation algorithm. In order to deploy mLDP in a network that supports MTR and/or FA, mLDP is required to become topology and FA aware. This document specifies extensions to mLDP to support MTR with FA such that when building a Multi-Point LSPs it can follow a particular topology and algorithm.

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 December 31, 2022.
1.  Glossary

   MT - Multi-Topology

   MT-ID - Multi-Topology Identifier

2. Introduction

Multi-Topology Routing (MTR) is a technology to enable service differentiation within an IP network. IGP protocols (OSPF and IS-IS) and LDP have already been extended to support MTR. To support MTR, an IGP maintains independent IP topologies, termed as "Multi-Topologies" (MT), and computes installs routes per topology. OSPF extensions [RFC4915] and ISIS extensions [RFC5120] specify the MT extensions under respective IGPs. To support IGP MT, similar LDP extensions [RFC7307] have been specified to make LDP MT-aware and be able to setup unicast Label Switched Paths (LSPs) along IGP MT routing paths.

A more light weight mechanism to define constraint-based topologies is Flexible Algorithm (FA) [I-D.ietf-lsr-flex-algo]. FA can be seen as creating a sub-topology within a topology using defined topology constraints and computation algorithm. This can be done within a MTR topology or just the default Topology. An instance of such a sub-topology is identified by a 1 octet value as documented in [I-D.ietf-lsr-flex-algo]. Flexible Algorithm is a mechanism to create a sub-topology, but in the future different algorithms might be defined on how to achieve that. For that reason, in the remainder of this document we’ll refer to this as the IGP Algorithm (IPA).

Multipoint LDP (mLDP) refers to extensions in LDP to setup multi-point LSPs (point-to-multipoint (P2MP) or multipoint-to-multipoint...
(MP2MP)), by means of set of extensions and procedures defined in [RFC6388]. In order to deploy mLDP in a network that supports MTR and FA, mLDP is required to become topology and algorithm aware. This document specifies extensions to mLDP to support MTR/IPA such that when building a Multi-Point LSPs it can follow a particular topology and algorithm. This means that the identifier for the particular Topology to be used by mLDP have to become a two tuple (MTR Topology Id, IGP Algorithm).

3. Specification of Requirements

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

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

4. MT Scoped mLDP FECs

As defined in [RFC7307], MPLS Multi-Topology Identifier (MT-ID) is an identifier that is used to associate an LSP with a certain MTR topology. In the context of MP LSPs, this identifier is part of the mLDP FEC encoding so that LDP peers are able to setup an MP LSP via their own defined MTR policy. In order to avoid conflicting MTR policies for the same mLDP FEC, the MT-ID needs to be a part of the FEC, so that different MT-ID values will result in unique MP-LSP FEC elements.

The same applies to the IPA. The IPA needs to be encoded as part of the mLDP FEC to create unique MP-LSPs and at the same time is used to signal to mLDP (hop-by-hop) which Algorithm needs to be used to create the MP-LSP.

Since the MT-ID and IPA are part of the FEC, they apply to all the LDP messages that potentially include an mLDP FEC element.

4.1. MP FEC Extensions for MT

Following subsections propose the extensions to bind an mLDP FEC to a topology. The mLDP MT extensions reuse some of the extensions specified in [RFC7307].
4.1.1. MP FEC Element

Base mLDP specification [RFC6388] defines MP FEC Element as follows:

```
+---------------------------------------------------------------+     +
| MP FEC type | Address Family | AF Length |
+---------------------------------------------------------------+     +
| Root Node Address |
+---------------------------------------------------------------+     +
| Opaque Length | Opaque Value |
+---------------------------------------------------------------+     +
```

Figure 1: MP FEC Element Format [RFC6388]

Where "Root Node Address" encoding is as defined for given "Address Family", and whose length (in octets) is specified by the "AF Length" field.

To extend MP FEC elements for MT, the {MT-ID, IPA} is a tuple that is relevant in the context of the root address of the MP LSP. The {MT-ID, IPA} tuple determines in which (sub)-topology the root address needs to be resolved. Since the {MT-ID, IPA} tuple should be considered part of the mLDP FEC, the most natural place to encode this tuple is as part of the root address. While encoding it, we also propose to use "MT IP" Address Families as described in following sub section.

4.1.2. MT IP Address Families

[RFC7307] has specified new address families, named "MT IP" and "MT IPv6", to allow specification of an IP prefix within a topology scope. In addition to using this address family for mLDP, we also use 8 bits of the 16 bits Reserved field to encode the IGP Algorithm (IPA) Registry. The resulting format of the data associated with these new Address Families is as follows:
IPv4/IPv6 Address: An IP address corresponding to "MT IP" and "MT IPv6" address families respectively.

IPA: The IGP Algorithm, values are from the IGP Algorithm registry.

Reserved: This 8-bit field SHOULD be zero.

4.1.3. MT MP FEC Element

By using extended MT IP Address Family, the resultant MT MP FEC element is to be encoded as follows:
In the context of this document, the applicable LDP FECs for MT mLDP include:

- **MP FEC Elements:**
  - P2MP (type 0x6)
  - MP2MP-up (type 0x7)
  - MP2MP-down (type 0x8)

- **Typed Wildcard FEC Element** (type 0x5)

  In case of "Typed Wildcard FEC Element", the sub FEC Element type MUST be one of the MP FECs listed above.

  This specification allows the use of Topology-scoped mLDP FECs in LDP label and notification messages, as applicable.

4.2. Topology IDs

This document assumes the same definitions and procedures associated with MPLS MT-ID as defined in [RFC7307] specification.

5. MT Multipoint Capability

"MT Multipoint Capability" is a new LDP capability, defined in accordance with LDP Capability definition guidelines [RFC5561], that is to be advertised to its peers by an mLDP speaker to announce its capability to support MTR and the procedures specified in this document. This capability MAY be sent either in an Initialization
message at the session establishment time, or in a Capability message
dynamically during the lifetime of a session (only if "Dynamic
Announcement" capability [RFC5561] has been successfully negotiated
with the peer).

The format of this capability is as follows:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|U|F|  MT Multipoint Cap.(IANA) |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|S| Reserved    |
+-+-+-+-+-+-+-+-+-+
```

Figure 4: MT Multipoint Capability TLV Format

Where:

- U- and F-bits: MUST be 1 and 0, respectively, as per Section 3 of
  LDP Capabilities [RFC5561].

- MT Multipoint Capability: TLV type (IANA assigned).

- Length: The length (in octets) of TLV. The value of this field
  MUST be 1 as there is no Capability-specific data [RFC5561] that
  follows in the TLV.

- S-bit: Set to 1 to announce and 0 to withdraw the capability (as
  per [RFC5561]).

An mLDP speaker that has successfully advertised and negotiated "MT
Multipoint" capability MUST support the following:

1. Topology-scoped mLDP FECs in LDP messages (Section 4.1)
2. Topology-scoped mLDP forwarding setup (Section 7)

6. MT Applicability on FEC-based features

6.1. Typed Wildcard MP FEC Elements

[RFC5918] extends base LDP and defines Typed Wildcard FEC Element
framework. Typed Wildcard FEC element can be used in any LDP message
to specify a wildcard operation for a given type of FEC.

The MT extensions proposed in document do not require any extension
in procedures for Typed Wildcard FEC Element support [RFC5918], and
these procedures apply as-is to Multipoint MT FEC wildcarding. Like Typed Wildcard MT Prefix FEC Element, as defined in [RFC7307], the MT extensions allow use of "MT IP" or "MT IPv6" in the Address Family field of the Typed Wildcard MP FEC element in order to use wildcard operations for MP FECs in the context of a given (sub-)topology as identified by the MT-ID and IPA field.

This document proposes following format and encoding for a Typed Wildcard MP FEC element:

```
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Typed Wcard (5)| Type = MP FEC |   Len = 6     |  AF = MT IP ..|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|... or MT IPv6 |    Reserved   |      IPA      |     MT-ID     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|MT ID (contd.) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Typed Wildcard MT MP FEC Element

Where:

- **Type**: One of MP FEC Element type (P2MP, MP2MPup, MP2MP-down).
- **MT ID**: MPLS MT ID
- **IPA**: The IGP Algorithm, values are from the IGP Algorithm registry.

The proposed format allows an LSR to perform wildcard MP FEC operations under the scope of a (sub-)topology.

6.2. End-of-LIB

[RFC5919] specifies extensions and procedures that allows an LDP speaker to signal its End-of-LIB (i.e. convergence) for a given FEC type towards a peer. MT extensions for MP FEC do not require any change in these procedures and they apply as-is to MT MP FEC elements. This means that an MT mLDP speaker MAY signal its convergence per (sub-)topology using MT Typed Wildcard MP FEC element.
7. Topology-Scoped Signaling and Forwarding

Since the \{MT-ID, IPA\} tuple is part of an mLDP FEC, there is no need to support the concept of multiple (sub-)topology forwarding tables in mLDP. Each MP LSP will be unique due to the tuple being part of the FEC. There is also no need to have specific label forwarding tables per topology, and each MP LSP will have its own unique local label in the table. However, In order to implement MTR in an mLDP network, the selection procedures for upstream LSR and downstream forwarding interface need to be changed.

7.1. Upstream LSR selection

The procedures as described in RFC-6388 section-2.4.1.1 depend on the best path to reach the root. When the \{MT-ID, IPA\} tuple is signaled as part of the FEC, this tuple is used to select the (sub-)topology that must be used to find the best path to the root address. Using the next-hop from this best path, a LDP peer is selected following the procedures as defined in [RFC6388].

7.2. Downstream forwarding interface selection

The procedures as described in RFC-6388 section-2.4.1.2 describe how a downstream forwarding interface is selected. In these procedures, any interface leading to the downstream LDP neighbor can be considered as candidate forwarding interface. When the \{MT-ID, IPA\} tuple is part of the FEC, this is no longer true. An interface must only be selected if it is part of the same (sub-)topology that was signaled in the mLDP FEC element. Besides this restriction, the other procedures in [RFC6388] apply.

8. LSP Ping Extensions

[RFC6425] defines procedures to detect data plane failures in Multipoint MPLS LSPs. Section 3.1.2 of [RFC6425] defines new Sub-Types and Sub-TLVs for Multipoint LDP FECs to be sent in "Target FEC Stack" TLV of an MPLS echo request message [RFC4379].

To support LSP ping for MT Multipoint LSPs, this document uses existing sub-types "P2MP LDP FEC Stack" and "MP2MP LDP FEC Stack" defined in [RFC6425]. The proposed extension is to specify "MT IP" or "MT IPv6" in the "Address Family" field, set the "Address Length" field to 8 (for MT IP) or 20 (for MT IPv6), and encode the sub-TLV with additional (MT-ID, IPA) information as an extension to the "Root LSR Address" field. The resultant format of sub-tlv is as follows:
The rules and procedures of using this new sub-TLV in an MPLS echo request message are same as defined for P2MP/MP2MP LDP FEC Stack Sub-TLV in [RFC6425] with only difference being that Root LSR address is now (sub-)topology scoped.

9. Security Considerations

This extension to mLDP does not introduce any new security considerations beyond that already apply to the base LDP specification [RFC5036], base mLDP specification [RFC6388], and MPLS security framework [RFC5920].

10. IANA Considerations

This document defines a new LDP capability parameter TLV. IANA is requested to assign the lowest available value after 0x0500 from "TLV Type Name Space" in the "Label Distribution Protocol (LDP) Parameters" registry within "Label Distribution Protocol (LDP) Name Spaces" as the new code point for the LDP TLV code point.
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
<th>Notes/Registration Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBA</td>
<td>MT Multipoint</td>
<td>This document</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Capability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7: IANA Code Point

11. Acknowledgments

The authors would like to acknowledge Eric Rosen for his input on this specification.

12. References

12.1. Normative References

[I-D.ietf-lsr-flex-algo]


12.2. Informative References


Authors' Addresses

IJsbrand Wijnands
Individual

Email: ice@braindump.be
IANA Registry for the First Nibble Following a Label Stack
draft-kbbma-mpls-1stnibble-02

Abstract

The goal of this memo is to create a new IANA registry (called the MPLS First Nibble registry) for the first nibble (4-bit field) immediately following an MPLS label stack. The memo offers a rationale for such a registry, describes how the registry should be managed, and provides some initial entries. Furthermore, this memo sets out some documentation requirements for registering new values. Finally, it provides some recommendations that makes processing MPLS packets easier and more robust.

There is an important caveat on the use of this registry versus the IP version number registry.

This memo, if published, would update [RFC4928] and [RFC8469].

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 11 January 2023.
1. Introduction

An MPLS packet consists of a label stack, an optional "post-stack header" (PSH) and an optional embedded packet (in that order). By PSH, we mean existing artifacts such as Control Words, BIER headers and the like, as well as new types of PSH being discussed in the MPLS Open Design Team meetings. However, in the data plane, there are scant clues regarding the PSH, and no clue as to the type of embedded packet; this information is communicated via other means, such as the routing protocols that signal the labels in the stack. Nonetheless, in order to better handle an MPLS packet in the data plane, it is common practice for network equipment to "guess" the type of embedded packet. Such equipment may also need to process the post-stack header. Both of these require parsing the data after the label.
stack. To do this, the "first nibble" (the top four bits of the first octet following the label stack) is often used.

The semantics and usage of the first nibble is not well documented, nor are the assignments of values. This memo serves three purposes:

* To document the assignments already made
* To provide for the clear documentation of future assignments through the creation of an "MPLS First Nibble registry"
* Provide a method to tracking usage by requiring more consistent documentation
* To reiterate the importance that any MPLS packet not carrying plain IPv4 or IPv6 packets MUST contain a PSH

There have been suggestions during discussions at the MPLS Open Design Team meetings that this document may serve as a registry for the PSH "headers of headers" types; however, this change needs WG consensus.

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 BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

LSR: label switching router.

MPLS packet: one whose Layer 2 header declares the type to be MPLS. For Ethernet, that means the Ethertype is 0x8847 or 0x8848.

Label Stack: (of an MPLS packet) all labels (four octet fields) after the Layer 2 header, up to and including the label with the BoS bit set ([RFC3032]).

MPLS First Nibble (MFN): the most significant four bits of the first octet following the label stack.

MPLS Payload: all data after the label stack, including the MFN, an optional post-stack header and the embedded packet.

Post-stack Header (PSH): optional field of interest to the egress
LSR (and possibly to transit LSRs). Examples include a control word or an associated channel. The PSH MUST indicate its length, so that a parser knows where the embedded packet starts.

**Embedded Packet:** All octets beyond the PSH (if any). This could be an IPv4 or IPv6 packet (e.g., for traffic engineering of IP packets, or for a Layer 3 VPN [RFC4364]), an Ethernet packet (for VPLS ([RFC4761], [RFC4762]) or EVPN [RFC7432]), or some other type of Layer 2 frame [RFC4446].

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
X | Layer 2 Header |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Y | Label-1 | Label-2 | ... | Label-n |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
A | (MFN) | IP packet | data | ... | end of IP packet |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Example of an MPLS Packet With Label Stack

Figure 2
Figure 1 shows an MPLS packet with Layer 2 header X and a label stack Y ending with Label-n. Then, there are three examples of an MPLS payload. The full MPLS packet thus would consist of [X Y A], or [X Y B], or [X Y C].

A. The first payload is a bare IP packet, i.e., no PSH. The MFN (MPLS First Nibble) in this case overlaps with the IP version number.

B. The next payload is a bare non-IP packet; again, no PSH. The MFN here is the first nibble of the payload, whatever it happens to be.

C. The last example is an MPLS Payload that starts with a PSH followed by the embedded packet. Here, the embedded packet could be IP or non-IP.

2. Rationale

2.1. Why Look at the First Nibble

An MPLS packet can contain many types of embedded packet. The most common types are:

1. An IPv4 packet (whose IP header has version number 4).
2. An IPv6 packet (whose IP header has version number 6).

3. A Layer 2 Ethernet frame (i.e., not including the Preamble or the Start frame delimiter), starting with the destination MAC address.

Many other packet types are possible, and in principle, any Layer 2 embedded packet is permissible; indeed, in the past, PPP, Frame Relay and ATM packets were reasonably common.

In addition, there may be a post-stack header ahead of the embedded packet, and this needs be to parsed. The MPLS First Nibble is currently used for both of these purposes.

2.1.1. Load Balancing

There are four common ways to load balance an MPLS packet:

1. One can use the top label alone.
2. One can do better by using all the (non-SPL) labels in the stack.
3. One can do even better by "divining" the type of embedded packet, and using fields from the guessed header.
4. One can do best by using either an Entropy Label [RFC6790] or a FAT Pseudowire Label [RFC6391]; see Section 2.1.3.)

Load balancing based on just the top label means that all packets with that top label will go the same way -- this is far from ideal. Load balancing based on the entire label stack (not including SPLs) is better, but may still be uneven. If, however, the embedded packet is an IP packet, then the combination of (source IP address, dest IP address, transport protocol, source port, and dest port) from the IP header of the embedded packet forms an excellent basis for load balancing. This is what is typically used for load balancing IP packets.

An MPLS packet doesn't, however, carry a payload type identifier. There is a simple (but dangerous) heuristic that is commonly used to guess the type of the embedded packet. The first nibble, i.e., the four most significant bits of the first octet, of an IP header contains the IP version number. This in turn indicates where to find the relevant fields for load balancing. The heuristic goes roughly as follows:

2.1.1.1. Heuristic for Load Balancing
1. If the MFN is 0x4 (0100b), treat the payload as an IPv4 packet, and find the relevant fields for load balancing on that basis.

2. If the MFN is 0x6 (0101b), treat the payload as an IPv6 packet, and find the relevant fields for load balancing on that basis.

3. If the MFN is anything else, the MPLS payload is not an IP packet; fall back to load balancing using the label stack.

This heuristic has been implemented in many (legacy) routers, and performs well in the case of Figure 1, A. However, this heuristic can work very badly for Figure 1, B. For example, if payload B is an Ethernet frame, then the MFN is the first nibble of the OUI of the destination MAC address, which can be 0x4 or 0x6, and if so would lead to very bad load balancing. This behavior can happen to other types of non-IP payload as well.

This in turn led to the idea of inserting a PSH (e.g., a pseudowire control word [RFC4385], a DetNet control word [RFC8964] or a BIER header [RFC8296]) where the MPLS First Nibble is NOT 0x4 or 0x6, to explicitly prevent forwarding engines from confusing the MPLS payload with an IP packet. [RFC8469] recommends the use of a control word when the embedded packet is an Ethernet frame. RFC 8469 was published at the request of the operator community and the IEEE RAC as a result of operational difficulties with pseudowires that did not contain the control word.

This memo introduces a requirement and a recommendation, the first building on the above; the second deprecating the use of the heuristic in Section 2.1.1.1. The intent of both of these is that legacy routers continue to operate as they have, with no new problems introduced as a result of this memo. However, new implementations SHOULD follow these recommendations for more robust operation.

2.1.2. Requirement

Going forward, network equipment MUST use a post-stack header with an MPLS First Nibble value that is not 0x4 or 0x6 in all cases when the MPLS payload is not an IP packet. Effectively, Figure 1, B is disallowed. [AGREED??]

This replaces the following text from [RFC4928], section 3, paragraph 3:
"It is REQUIRED, however, that applications depend upon in-order packet delivery restrict the first nibble values to 0x0 and 0x1. This will ensure that their traffic flows will not be affected if some future routing equipment does similar snooping on some future version(s) of IP."

This also replaces the following text from [RFC8469], section 4, paragraph 1:

"This document updates [RFC4448] to state that both the ingress provider edge (PE) and the egress PE SHOULD support the Ethernet PW CW and that, if supported, the CW MUST be used."

2.1.3. Recommendation

It is RECOMMENDED that, going forward, if good load balancing of MPLS packets is desired, either an Entropy Label or a FAT Pseudowire Label SHOULD be used; furthermore, going forward, the heuristic in Section 2.1.1.1 MUST NOT be used. [AGREED??]

A consequence of Recommendation 2 is that, while legacy routers may look for a MPLS First Nibble of 0x4 or 0x6, no router will look for a MPLS First Nibble of 0x7 (or whatever the next IP version number will be) for load balancing purposes. This means that the values 0x4 and 0x6 are used to (sometimes incorrectly) identify IPv4 and IPv6 packets, but no other First Nibble values will be used to identify IP packets.

This obviates the need for paragraph 4, section 3 in [RFC4928]:

"This behavior implies that if in the future an IP version is defined with a version number of 0x0 or 0x1, then equipment complying with this BCP would be unable to look past one or more MPLS headers, and loadsplit traffic from a single LSP across multiple paths based on a hash of specific fields in the IPv0 or IPv1 headers. That is, IP traffic employing these version numbers would be safe from disturbances caused by inappropriate loadsplitting, but would also not be able to get the performance benefits."

This also expands the MFN Registry to all 16 possible values, not just 0x0 and 0x1.
2.1.4. Parsing the Post-stack Header

Given the above recommendations on the use of a post-stack header and future non-use of the heuristic (Section 2.1.1.1) via the use of Entropy or FAT Pseudowire Labels, the main reason for creating a First Nibble registry is to document the types of post-stack headers that may follow a label stack, and to simplify their parsing.

2.2. Why Create a Registry

The MPLS WG is currently engaged in updating the MPLS architecture; part of this work involves the use of post-stack headers. This is not possible if post-stack header values are allocated on an ad hoc basis, and their parsing and semantics is ill-specified. Consider that the MPLS First Nibble value of 0x0 has two different formats, depending on whether the post-stack header is a pseudowire control word or a DetNet control word; disambiguation requires the context of the service label. This was a considered decision; documenting this would be helpful to future implementors.

With a registry, post-stack headers become easier to parse; the values are unique, not needing means outside the data plane to interpret them correctly; and their semantics and usage are documented. (Thank you, IANA!)

2.3. Caveat

The use of the MPLS First Nibble stemmed from the desire to heuristically identify IP packets for load balancing purposes. It was then discovered that non-IP packets, misidentified as IP when the heuristic failed, were being badly load balanced, leading to [RFC4928]. This situation may confuse some as to relationship between the MPLS First Nibble Registry and the IP Version Numbers registry. These registries are quite different:

1. The IP Version Numbers registry’s explicit purpose is to track IP version numbers in an IP header.

2. The MPLS First Nibble registry’s purpose is to track post-stack header types.

The only intersection points between the two registries is for values 0x4 and 0x6 (for backward compatibility). There is no need to track future IP version number allocations in the MPLS First Nibble registry.

3. IANA Considerations
3.1. MPLS First Nibble Registry

This memo recommends the creation of an IANA registry called "The MPLS First Nibble Registry" with the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
<th>Reference</th>
<th>Allocation Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0</td>
<td>PW Control Word</td>
<td>RFC 4385</td>
<td></td>
</tr>
<tr>
<td>0x0</td>
<td>DetNet Control Word</td>
<td>RFC 8964</td>
<td></td>
</tr>
<tr>
<td>0x1</td>
<td>PW Assoc Channel</td>
<td>RFC 4385</td>
<td></td>
</tr>
<tr>
<td>0x2</td>
<td>Unallocated</td>
<td></td>
<td>Standards Action</td>
</tr>
<tr>
<td>0x3</td>
<td>Unallocated</td>
<td></td>
<td>Standards Action</td>
</tr>
<tr>
<td>0x4</td>
<td>IPv4 header</td>
<td>RFC 791</td>
<td></td>
</tr>
<tr>
<td>0x5</td>
<td>BIER header</td>
<td>RFC 8296</td>
<td></td>
</tr>
<tr>
<td>0x6</td>
<td>IPv6 header</td>
<td>RFC 8200</td>
<td></td>
</tr>
<tr>
<td>0x7-e</td>
<td>Unallocated</td>
<td></td>
<td>Standards Action</td>
</tr>
<tr>
<td>0xf</td>
<td>Reserved for expansion</td>
<td></td>
<td>Standards Action</td>
</tr>
</tbody>
</table>

Table 1: MPLS First Nibble Values

3.1.1. Allocation Policy

All new values registered here MUST use the Standards Action policy [RFC8126].

4. References

4.1. Normative References


4.2. Informative References


Authors’ Addresses

Kireeti Kompella (editor)
Juniper Networks
1133 Innovation Way
Sunnyvale, 94089
United States of America
Phone: +1-408-745-2000
Email: kireeti.ietf@gmail.com

Stewart Bryant
University of Surrey 5GIC
Email: sb@stewartbryant.com
Matthew Bocci
Nokia
Email: matthew.bocci@nokia.com

Greg Mirsky
Ericsson
Email: gregimirsky@gmail.com

Lars Olaf (Loa) Andersson
Bronze Dragon Consulting
Email: loa@pi.nu
Abstract

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

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

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

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

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

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

1.1. Terminology

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

LSP: Label Switched Path

LSR: Label Switching Router

RAO: Router Alert Option

2. Router Alert for LSP Ping (RFC 8029)

2.1. Echo Request

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

To achieve this, RFC 8029 proposes the following:

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

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

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

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

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

2.2. Echo Reply

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

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

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

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

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

3. Update to RFC 7506

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

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

5. IANA Considerations

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

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

6. Security Considerations

The recommendations this document makes do not compromise security.

7. Normative References


Authors’ Addresses

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

Email: kireeti.ietf@gmail.com

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

Email: rbonica@juniper.net
Abstract

With the development of new services, MPLS faces many problems and technical challenges. This document defines the method to implement MPLS through the GIP6 tunnel.

Requirements Language

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

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 January 2023.

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.
1. Introduction

The GIP6 tunnel is defined in [draft-li-rtgwg-generic-ipv6-tunnel-00] which is to solve the challenges of the existing IP tunnels to support new features such as alternate marking, IOAM, network resource partition and APN.

With the development of new services, MPLS also faces many problems and technical challenges. For example, it is difficult to encapsulate new forwarding attributes because of lack of the metadata extensibility. This document defines how to implement MPLS through the GIP6 tunnel which is to solve the possible problems effectively.

2. Terminology

APN: Application-Aware Networking

GIP6: Generic Ipv6 Tunnel

GRE: Generic Routing Encapsulation

IFIT: In-situ Flow Information Telemetry

MP2P: Multi Point To Point

MPLS: Multiprotocol Label Switching

PM: Performance Monitor
3. Problem Statement

With the development of new services, MPLS faces the following technical problems and challenges of MPLS:

1. MPLS is lack of the source indication and MP2P connections may occur. This causes the difficulty and complex process for OAM over MPLS. Although SFL ([RFC8957]) is defined, there is few implementation.

2. The payload type (for example, L2 or L3 packets) cannot be directly determined because there is no payload indication.

3. There is no metadata extensibility and it is difficult to encapsulate new forwarding attributes for the new features such as IETF network slicing, IFIT, and APN.

4. The process of the ECMP function is complex and affects forwarding performance. Entropy labels or flow labels are placed at the bottom of the label stack for processing and the internal IP header information may have to be parsed for the purpose of ECMP.

4. GIP6 Tunnel for MPLS

[I-D.li-rtgwg-generalized-ipv6-tunnel] defines the GIP6 tunnel to support both new features and the existing functions for the IP tunnels based on the extension of the IPv6 extension header. If the GIP6 tunnel is used for MPLS, there can be the following advantages:

1. The IPv6 source address is used to form a source identifier.

2. The IPv6 NH can indicate the payload type.

3. IPv6 flow labels are used to implement ECMP.

4. The encapsulations for the new features have been defined well in the IPv6 and can be reused easily.

It is simple and can benefit forwarding performance. Moreover, there have been many implementations and deployments.
In order to support MPLS based on the GIP6 tunnel, the method to carry MPLS label stack information is defined as follows:

1. A special IPv6 prefix MUST be used to indicate that it is followed by MPLS label encapsulation. The special IPv6 prefix can be specified or a well-known IPv6 prefix to be assigned.

2. The IPv6 special prefix can be followed by multiple MPLS label encapsulations to form a 128-bit IPv6 MPLS SID (Type 1). The format of the IPv6 MPLS SID (Type 1) is shown in the following figure.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Special Prefix(4 octets)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               MPLS Label Encap (4 octets)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               MPLS Label Encap (4 octets)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               MPLS Label Encap (4 octets)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               MPLS Label Encap (4 octets)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               MPLS Label Encap (4 octets)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1. IPv6 MPLS SID (TYPE 1)

Special prefix: TBD

MPLS Label Encap: For details, please refer to section 2.1 in [RFC3032].

3. IPv6 MPLS SID (Type 1) can be placed in the IPv6 destination address.

Processing of the first label following the special prefix is as follows:

(1) If the local action of the MPLS label is POP, the followed label encapsulations are shifted left by 32 bits after the label is popped. The following figure shows the process.
Before POP MPLS Label Encap:

<table>
<thead>
<tr>
<th>uSID-Block</th>
<th>Active-Label</th>
<th>Next-Label</th>
<th>Last-Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+--------------+------------+------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Special Prefix</td>
<td>Lable-1 Encap</td>
<td>Label-2 Encap</td>
</tr>
<tr>
<td>+-----------+--------------+------------+------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After POP MPLS Label Encap:

<table>
<thead>
<tr>
<th>uSID-Block</th>
<th>Active-Label</th>
<th>Next-Label</th>
<th>Last-Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-----------+--------------+------------+------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Special Prefix</td>
<td>Lable-2 Encap</td>
<td>Label-3 Encap(EOL)</td>
</tr>
<tr>
<td>+-----------+--------------+------------+------------</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. Pop MPLS Label Encapsulation in IPv6 DA

(2) If the local action of the MPLS label is SWAP, the label encapsulation is changed to the new label after swap.

4. If all the MPLS label stack cannot be placed in the IPv6 destination address, IPv6 RH can be used to house the remaining MPLS label stack.

(1) IPv6 MPLS SID (Type 2) is defined to house multiple (≤4) label encapsulations. The format of the IPv6 MPLS SID (Type 2) is shown in the following figure.

```
+-----------+-----------+-----------+-----------+
|            |            |            |            |
| 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 |           |
|                      |            |            |            |
| +-----------+-----------+-----------+-----------+
| | MPLS Label Encap (4 octets) |
| +-----------+-----------+-----------+-----------+
|                      |            |            |            |
| +-----------+-----------+-----------+-----------+
| | MPLS Label Encap (4 octets) |
| +-----------+-----------+-----------+-----------+
|                      |            |            |            |
| +-----------+-----------+-----------+-----------+
| | MPLS Label Encap (4 octets) |
| +-----------+-----------+-----------+-----------+
|                      |            |            |            |
| +-----------+-----------+-----------+-----------+
| | MPLS Label Encap (4 octets) |
| +-----------+-----------+-----------+-----------+
|                      |            |            |            |
| +-----------+-----------+-----------+-----------+
| | MPLS Label Encap (4 octets) |
| +-----------+-----------+-----------+-----------+

MPLS Label Encap: For details, see section 2.1 in [RFC3032].

(2) IPv6 MPLS SID (Type 2) is used as the segment in the RH. After all of the label encapsulations in the IPv6 destination address are popped, the first label encapsulation in the segment indicated by the SL of the RH will be processed as follows:
-- If the local action of the MPLS label is POP, the followed label encapsulations are shifted left by 32 bits after the label is popped.

-- If the local action of the MPLS label is SWAP, the label encapsulation is changed to the new label after swap.

After all the label encapsulations in the segment are popped, SL minus 1. Then the first label encapsulation in the segment indicted by the new SL will go on to be processed as the above procedures.

A new type of RH can be defined to contain IPv6 MPLS SID (Type 2) or SRv6 SRH can be reused for the purpose.

5. When find the S flag of the label encapsulation in the IPv6 destination address or the RH to be processed is set, this means the bottom of the label stack is reached and the process of the label stack in the GIP6 will end after the label is popped.

If an intermediate node requires to push a label or a label stack, there can be two modes: Encap mode and Inserting mode.

1) Encap mode: with this mode, a new IPv6 MPLS packet header is encapsulated outside the original MPLS packet, and the MPLS label (stack) is encapsulated in the new IPv6 MPLS packet header as the above procedures.

2) Inserting mode: All the label encapsulations in the IPv6 destination address and the IPv6 RH (if exist) need to be shifted right and the new label (stack) can be placed immediately following the special prefix in the IPv6 destination address. The process is complex and not recommended.

5. Control Plane Considerations

GIP6 only provides a way to carry MPLS label encapsulations in the data plane. The existing MPLS control plane does not need to be changed. That is, MPLS labels on the control plane can still be distributed for IPv4, IPv6, L2, etc.

6. Security Considerations

TBD.
7. IANA Considerations

TBD.

8. References

[I-D.li-rtgwg-generalized-ipv6-tunnel]


Authors’ Addresses

Zhenbin Li
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China
Email: lizhenbin@huawei.com

Jie Dong
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China
Email: jie.dong@huawei.com