Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 25, 2019

Anil Kumar
Deepak J
Basil Saji
RtBrick India

September 21, 2018

MPLS LDP Identifier Name
draft-abd-mpls-ldp-identifier-name-00

Abstract

This document introduces a new optional LDP Identifier Name TLV that allows LDP routers to inform their LDP-Identifier-to-Name mapping in LDP Hello messages as part of the LDP Discovery Mechanism. This document describes a mechanism to provide a simple and dynamic mechanism for ldp routers to learn about symbolic LDP Identifier Names.

Requirements Language

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

Status of This Memo

This Internet-Draft 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 March 25, 2019.

Copyright Notice

Table of Contents

1. Introduction

The Label Distribution Protocol (LDP) [RFC5036] sets up LDP sessions that run between LDP peers. The peers could either be directly connected at the link level or could be multiple hops away. An LDP Label Switching Router (LSR) could either be configured with the identity of its peers or could discover them using LDP Hello messages. These messages are sent encapsulated in UDP addressed to "all routers on this subnet" or to a specific IP address. Periodic Hello messages are also used to maintain the relationship between LDP peers necessary to keep the LDP session active.

[RFC5036] states that an LDP Identifier is a six octet quantity used to identify an LSR label space. The first four octets identify the LSR and must be a globally unique value, such as a 32-bit router Id assigned to the LSR. The last two octets identify a specific label space within the LSR. The last two octets of LDP Identifiers for platform-wide label spaces are always both zero.

An LSR that manages and advertises multiple label spaces uses a different LDP Identifier for each such label space.

[RFC5036] specifies using following representation for LDP Identifiers: <LSR Id> : <label space id> e.g., lsr171 : 0 lsr19 : 2

For management and operational reasons, network operators need to check the status of LDP adjacencies, entries in Label Information Base (LIB), and next hop address for the prefix to an LDP Identifier mapping table. When looking at diagnostic information, numerical representations of LDP Identifiers (e.g., dotted-decimal or hexadecimal representations) are less clear to humans than symbolic names.

LDP is increasingly used inside the data center. Due to the sheer scale of devices(ldp peers) involved, simplifying troubleshooting LDP would be very useful. One way to overcome this problem is to define a LDP-Identifier-to-Name mapping table on a router. This mapping can be used bidirectionally (e.g., to find symbolic names for LDP-Identifiers and to find LDP-Identifiers for symbolic names) or unidirectionally (e.g., to find symbolic names for LDP-Identifiers). Thus, every router has to maintain a table with mappings between names and LDP-Identifiers.

If these mapping tables are built by static definitions, it can currently become a manual and tedious process in operational networks; modifying these static mapping entries when additions, deletions, or changes occur becomes a non-scalable process very prone to error.

This document provides a way to populate tables by defining a new optional LDP Identifier Name TLV for LDP which will be exchanged in LDP Hello messages.

2. LDP Identifier Name TLV

2.1. Optional Parameter for Hello Message

There are various approaches to providing a LDP-Identifier-to-Name mapping service.

One way to build this table of mappings is by static definitions. The problem with static definitions is that the network administrator needs to keep updating the mapping entries manually as the network changes; this approach does not scale as the network grows, since there needs to be an entry in the mapping table for each LDP peer. Thus, this approach greatly suffers from maintainability and scalability considerations.

Another approach is having a centralized location where the name-to-LDP-Identifier mapping can be kept. The DNS could be used for this. A disadvantage with this centralized solution is that it is a single point of failure; and although enhanced availability of the central mapping service can be designed, it may not be able to resolve the LDP Identifier name in the event of reachability or network problems, which can be particularly problematic in times of problem resolution. Also, the response time can be an issue with the centralized solution, which can be equally problematic. If the DNS is used as the centralized mapping table, a network operator may desire a different name mapping than the existing mapping in the DNS, or new routers may not yet be in the DNS.

The third solution that we have defined in this document is to make use of the protocol itself to carry the LDP-Identifier-to-Name mapping in a TLV. Routers that understand this TLV can use it to create the symbolic LDP-Identifier-to-Name mapping, and routers that don't understand it can simply ignore it. This specification provides these semantics and mapping mechanisms for LDP, leveraging the LDP Optional TLVs exchanged in LDP Hellos. ([RFC5036]).

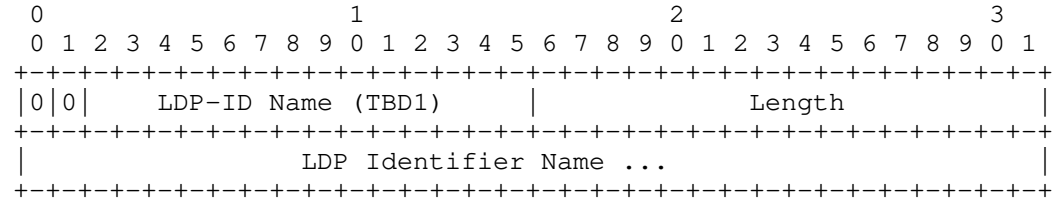
[RFC5036] defines the encoding for the Hello message. Each Hello message contains zero or more Optional Parameters, each encoded as a TLV. Three Optional Parameters are defined by [RFC5036].

[RFC7349] defines optional Cryptographic Authentication TLV.This document defines a new Optional Parameter: the LDP Identifier Name parameter.

Optional Parameter	Type
IPv4 Transport Address	0x0401 (RFC5036)
Configuration Sequence Number	0x0402 (RFC5036)
IPv6 Transport Address	0x0403 (RFC5036)
Cryptographic Authentication	0x0405 (RFC7349)
LDP Identifier Name	TBD1 (this document, TBD1 by IANA)

The LDP Identifier Name TLV Encoding is described in section 2.2.

2.2. LDP Identifier Name TLV Encoding



- Type: TBD1, LDP Identifier Name Type

- Length: Total length of the LDP Identifier Name (Value field) in octets, not including the optional padding.

- LDP Identifier Name: LDP Identifier Name, a string of 1 to 255 octets, padded with zeroes to 4-octet alignment, encoded in the US-ASCII charset.

Routers that do not recognize the LDP Identifier Name TLV Type ignore the TLV

The Value field identifies the symbolic LDP Identifier Name of the router originating the LSA. This symbolic name can be the Fully Qualified Domain Name (FQDN) for the LDP Identifier, it can be a subset of the FQDN, or it can be any string that operators want to use for the router. The use of FQDN or a subset of it is strongly recommended since it can be beneficial to correlate the LDP Identifier Name and the corresponding DNS entry. If there is no DNS LDP Identifier Name for the LDP Identifier, if the LDP Identifier does not map to an IPv4-addressed entity or if an alternate LDP Identifier Name naming convention is desired, any string with significance can be used. The string is not null-terminated. The LDP Identifier of this router is derived from the LDP header. The Value field is encoded in 7-bit ASCII. If a user-interface for configuring or displaying this field permits Unicode characters, that user-interface is responsible for applying the ToASCII and/or ToUnicode algorithm as described in [RFC3490] to achieve the correct format for transmission or display.

3. Security Considerations

Since the LDP-Identifier-to-Name mapping relies on information provided by the routers themselves, a misconfigured or compromised router can inject false mapping information, including a duplicate LDP-Identifier Name for different LDP-Identifiers. Thus, this information needs to be treated with suspicion when, for example, doing diagnostics about a suspected security incident.

There is potential confusion from name collisions if two routers use and advertise the same LDP-Identifier names. Name conflicts are not crucial, and therefore there is no generic conflict detection or resolution mechanism in the protocol. However, a router that detects that a received LDP-Identifier name is the same as the local one can issue a notification or a management alert.

This document raises no other new security issues for LDP.

4. IANA Considerations

The IANA is requested to as assign a new TLV from the "Label Distribution Protocol (LDP) Parameters" registry, "TLV Type Name Space".

Value	Meaning	Reference
TBD1	LDP Identifier Name TLV	this document (sect 2.2)

Note to the RFC Editor and IANA (to be removed before publication):

The new value should be assigned from the range 0x400 - 0x4ff using the first free value.

5. Acknowledgements

we would also thank Hannes Gredler for his invaluable guidance.

6. References

6.1. Normative References

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[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>.

[RFC3490] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, DOI 10.17487/RFC3490, March 2003, <https://www.rfc-editor.org/info/rfc3490>.

[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>.

6.2. Informative References

[RFC7349] Zheng, L., Chen, M., and M. Bhatia, "LDP Hello Cryptographic Authentication", RFC 7349, DOI 10.17487/RFC7349, August 2014, <https://www.rfc-editor.org/info/rfc7349>.

Authors' Addresses

Anil Kumar S N
RtBrick India
HSR Layout
Bengaluru, Karnataka 560102
India

Email: anil.ietf@gmail.com

Deepak J Gowda
RtBrick India
HSR Layout
Bengaluru, Karnataka 560102
India

Email: deepakj4982@gmail.com

Basil Saji
RtBrick India
HSR Layout
Bengaluru, Karnataka 560102
India

Email: sajibasil@gmail.com

Routing area
Internet-Draft
Intended status: Standards Track
Expires: August 25, 2019

K. Arora
S. Hegde
Juniper Networks Inc.
S. Aldrin
Google
S. Litkowski
Orange Business Service
M. Durrani
Equinix

February 21, 2019

TTL Procedures for SR-TE Paths in Label Switched Path Traceroute Mechanisms
draft-arora-mpls-spring-ttl-procedures-srte-paths-01

Abstract

Segment routing supports the creation of explicit paths using adjacency-sids, node-sids, and anycast-sids. The SR-TE paths are built by stacking the labels that represent the nodes and links in the explicit path. A very useful Operations And Maintenance requirement is to be able to trace these paths as defined in [RFC8029]. This document specifies a uniform mechanism to support MPLS traceroute for the SR-TE paths when the nodes in the network are following uniform mode or short-pipe mode [RFC3443].

Requirements Language

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

Status of This Memo

This Internet-Draft 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 August 25, 2019.

Table of Contents

1. Introduction

The mechanisms to handle TTL procedures for SR-TE paths are described in ([RFC8287]). Section 7.5 of ([RFC8287]) defines the TTL manipulation procedures for short pipe model as below.The LSR

initiating the traceroute SHOULD start by setting the TTL to 1 for the tunnel in the LSP's label stack it wants to start the tracing from, the TTL of all outer labels in the stack to the max value, and the TTL of all the inner labels in the stack to zero. However this mechanism has issues when the constituent tunnels are penultimate-hop-popping(PHP). This document does not propose any change to ([RFC8287]) if the constituent tunnels are ultimate-hop-popping (UHP) or Egress LSR advertizes explicit NULL.

Section 2 describes problems in tracing SR-TE paths and the need for a specialized mechanism to trace SR-TE paths. Section 3 describes the solution applied to mpls echo request/response to trace adjacency-sids and node-sids trace SR-TE path in uniform model and short pipe model.

2. Problem with SR-TE Paths

The topology shown in Figure 1. illustrates a example network topology with SPRING enabled on each node.

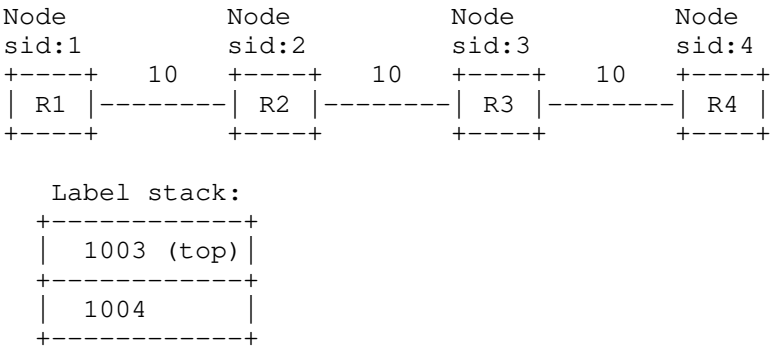


Figure 1: Example topology with SRGB 1000-2000

Consider an explicit path in the topology in Figure 1 from R1->R4 via R1->R2->R3->R4. The label stack to instantiate this path contains two node-sids 1003 and 1004. The 1003 label will take the packet from R1 to R3. The next label in the stack 1004 will take the packet from R3 to the destination R4. consider the mechanism below for the TTL procedures specified in RFC 8287 for short pipe model and uniform model for PHP LSPs.

Notation: ((X,Y),(Z,W)) refers to a label stack whose top label stack entry has the label corresponding to the node-SID of X, with TTL Y, and whose second label stack entry has the label corresponding to the node-SID of Z, with TTL W.

According to the procedure in Section 7.5 of [RFC8287], the LSP traceroute is done as follows in short pipe model and uniform model:

2.1. Short Pipe model

Refer the diagram in Figure 1.

1. Ingress R1 sends mpls LSP Echo Request with label stack of ((1003,1),(1004,0)) to R2.

2. Since R2 receives mpls LSP Echo Request with TTL as 1 for outer most label, R2's local software processes the Lsp traceroute packet and R2 sends an echo reply to R1 with return code as 'transit'.

3. R1 receives the LSP Echo Reply from R2, and then sends next LSP Echo Request with label stack ((1003,2),(1004,0)).

4. R2 forwards packet to R3 as ((1004,0)) (i.e. R2 being PHP, pops the label 1003 and does not propagate TTL)

5. R3 receives a packet with TTL=0 at the top of the stack. Receipt of a packet with TTL=0 may cause R3 to drop the packet or rate limit it.

6. Even if R3's local software processes the packet and validates the FEC for 1003 and sends egress code in echo-reply, the next packet will have ((1003,255), (1004, 1)) which causes TTL to expire again on R3 as the 1003 label is popped at the penultimate.

RFC 8287 suggests that when R1's LSP Echo Request has reached the egress of the outer tunnel, R1 should begin to trace the inner tunnel by sending a LSP Echo Request with label stack ((1003,255),(1004,1)). However, as explained in step 6, the traceroute procedure does not work correctly.

2.2. Uniform Model

1. Ingress R1 sends mpls LSP Echo Request with label stack of ((1003,1),(1004,0)) to R2.

2. Since R2 receives mpls LSP Echo Request with TTL as 1 for outer most label, R2's local software processes the Lsp ping packet and R2 sends an echo reply to R1 with return code as 'transit'.

3. R1 receives the LSP Echo Reply from R2, and then sends next LSP Echo Request with label stack ((1003,2),(1004,0)).

4. It is expected that R2 should propogate the TTL of outer label to inner label before forwarding the packet to R3. However most of the PFEs implementations generally do not increase a label stack entry's TTL when they do TTL propagation. So when (1003,2) is popped, we might still end up with (1004,0) at R3, even if we have TTL propagation configured. Increasing the TTL of a packet is not a good practice as it can result in forwarding loops.

5. R3 receives a packet with TTL=0 at the top of the stack. Receipt of a packet with TTL=0 will cause R3 to drop the packet or rate limit it.

6. Even if R3's local software processes the packet and validates the FEC for 1003 and sends egress code in echo-reply, the next packet will have ((1003,255), (1004, 1)) which causes TTL to expire again on R3 as the 1003 label is popped at the penultimate.

So in either case (uniform model or short pipe model) traceroute may not work for SR-TE paths with PHP Lsps.

3. Detailed Solution For TTL procedures for SR-TE paths

3.1. P bit in DDMT TLV

DS flags has 4 unused bits from position '0' to '3'. This document uses bit '3' in DS flags of downstream mapping TLV.

3.1.1. Procedures for a PHP router of the tunnel being traced

When a LSR receives an echo request it MUST validate the outermost FEC in the echo request. LSR SHOULD set the 'P' bit in the DS flags of downstream mapping TLV if its a PHP router for the outermost FEC. Other cases it should work as explained in [RFC8029] and [RFC8287].

3.1.2. Procedures for a egress router of the tunnel being traced

When a LSR receives an echo request it MUST validate the outermost FEC in the echo request. Egress cases should work as explained in [RFC8029] and [RFC8287].

3.1.3. Procedures for a ingress router of the SR-TE path

When an ingress LSR receives an echo response it MUST behave as defined below depending on the return code in the echo response.

1. When an ingress LSR receives an echo response with return code as 8 (Label switched at stack-depth), Ingress LSR MUST check if the LSR that sent the echo response is PHP for the outermost FEC in the FEC

stack. If the LSR that sent the echo response is PHP for the outermost FEC then while sending next echo request Ingress LSR MUST increase the TTL value of inner label also (if exists) in addition to increasing the TTL value of the tunnel it is tracing. Ingress LSR can detect that LSR that sent the echo response is a PHP router for the outermost FEC, either by looking at 'P' bit set in the DS flags of downstream mapping TLV or if Ingress LSR has received LABEL '3' in the label stack TLV of downstream detailed mapping TLV. For all other cases ingress should work as explained in [RFC8029] and [RFC8287].

2. When an Ingress LSR receives an echo response with return code as 3 (Replying router is an egress for the FEC at stack-depth) for the outermost FEC and this is not the only FEC in the FEC stack, then ingress LSR SHOULD remove the outermost FEC from the FEC stack and send the next traceroute request with the same TTL value for all the labels in the label stack as the previous echo request. This will ensure the egress of the tunnel is visited twice, once as egress for top label and again as a transit for next tunnel.

3.1.4. Example describing the solution

This section provides a detailed description of how PHP router helps ingress in handling TTL procedures for SR-TE paths. Below are the procedures performed by PHP router and ingress router to perform TTL procedure for mpls traceroute for SR-TE paths. Below solution works for both uniform model and short pipe model.

1. Ingress R1 sends mpls LSP Echo Request with label stack of ((1003,1),(1004,0)) to R2.

2. Since R2 receives mpls LSP Echo Request with TTL as 1 for outer most label, R2's local software processes the Lsp ping packet. R2's local software validates the outermost FEC and looking at the FEC R2 knows that its the PHP router for outermost FEC (Node-Sid R3).

3. R2 sets a bit in the DS flags in the DDMT TLV in echo response (P bit, One of the reserved bits).

4. When R1 looks at the echo response from R2 it sees P bit in DDMT TLV.

5. So R1 increments the TTL value of Node-R3 by 1 (make it 2) and TTL value of next element in the label stack also

6. R1 should send the next mpls LSP Echo Request with label stack ((1003,2),(1004,1)).

7. R2 being PHP pops the outermost label from the label stack and forwards the packet to R3 with with label (1004, 1)

8. R3 receives mpls LSP Echo Request with TTL as 1 for outer most label, R3's local software processes the echo request.

9. R3 validates the outermost FEC and sends echo response to R1 with return code as the egress for outermost FEC (Node-Sid R3).

10. When R1 receives echo response with return code as egress, R1 should remove outermost FEC (Node-Sid R3) from the FEC stack and send the next echo request with the same TTL value as the previous one i.e ((1003,2),(1004,1)).

11. Since R3 is the PHP router for FEC (Node-Sid R4) in the label stack. R3 should set 'P' bit in the in the DS flags in the DDMT TLV in echo response with return code as Transit.

12. R1 should send the next mpls LSP Echo Request with label stack ((1003,2),(1004,2)) with FEC Node-Sid-R4 .

13. R2 pops the first label from the label stack and R3 pops the second label from the label stack.

14. R4 receives an unlabelled packet with RA bit set in ip options. R4 delivers the packet to local software for processing.

15. R4's local software validates the ouetmost FEC as 'egress' and sends an echo reply with return code as egress.

17. R1 receives an echo reply with return code as egress for the last FEC in the FEC stack TLV and completes the traceroute.

3.2. Procedures for handling binding-sids

Inorder to provide greater scalability, network opacity, and service independence, SR architecture [RFC8402] defines a Binding SID (BSID). A Binding SID is bound to an SR policy which typically involves a list of SIDs. These Binding SIDs may appear in another SR Policy or may be used to steer service traffic from the service origin. The TTL handling mechanisms for MPLS traceroute procedures involving Binding SIDs is described below.

3.2.1. Uniform Model

When the node advertising the Binding SID is operating in uniform mode [RFC3443], it SHOULD send FEC stack change sub-TLV as in sec 4.5.1 of [RFC8029]. The ingress node SHOULD increment the TTL of

Binding SID label at every step until "egress" return code is sent for all the new FECs included due to FEC stack change and all the Tunnels replaced by the Binding SID are completely traced. It is required that all the label popping nodes involved in these tunnels MUST support uniform model and copy the TTL to bottom label when the label is popped.

3.2.2. Shortpipe Model

When the node advertising the Binding SID is operating in short pipe model [RFC3443], it SHOULD not send FEC stack change sub-TLV. The Binding SID is treated as single hop and the nodes internal to the Tunnel represented by Binding SID SHOULD NOT be traced.

4. Backward Compatibility

The extension proposed in this document is backward compatible with procedures described in [RFC8029] and [RFC8287]. If the LSR with the proposed solution is the Ingress and all other LSR in the SR tunnel are not with the extension, Then no LSR is going to set 'P' bit so ingress LSR with new extension will work as per [RFC8029] and [RFC8287].If the LSR with the proposed extension is the one of the transit router and if its the PHP then it may set 'P' bit based on the section 3. Ingress may not react to the 'P' bit and traceroute will continue to work as per [RFC8029] and [RFC8287].

5. Security Considerations

TBD

6. IANA Considerations

IANA has created and now maintains a registry entitled "DS Flags". The registration policy for this registry is Standards Action [RFC5226]. IANA has made the following assignments:

Bit Number Name Reference

---------- ---------- ---------

7 N: Treat as a Non-IP Packet [RFC8029]

6 I: Interface and Label Stack Object Request [RFC8029]

5 E: ELI/EL push indicator [RFC8012]

4 L: Label-based load balance indicator [RFC8012]

3 P: Penulimate Hop router

2-0 Unassigned

7. Acknowledgements

Thanks to Przemyslaw Krol for careful review and comments.

8. References

8.1. 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>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/info/rfc5226>.

[RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <https://www.rfc-editor.org/info/rfc8029>.

[RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <https://www.rfc-editor.org/info/rfc8287>.

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

8.2. Informative References

[RFC3443] Agarwal, P. and B. Akyol, "Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks", RFC 3443, DOI 10.17487/RFC3443, January 2003, <https://www.rfc-editor.org/info/rfc3443>.

Authors' Addresses

Kapil Arora
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: kapilaro@juniper.net

Shraddha Hegde
Juniper Networks Inc.
Exora Business Park
Bangalore, KA 560103
India

Email: shraddha@juniper.net

Sam Aldrin
Google

Email: aldrin.ietf@gmail.com

Stephane Litkowski
Orange Business Service

Email: stephane.litkowski@orange.com

Muhammad Durrani
Equinix

Email: mdurrani@equinix.com

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 21, 2020

K. Raza, Ed.
R. Asati
Cisco Systems

X. Liu
Volta Networks

S. Esale
Juniper Networks

X. Chen
Huawei Technologies

H. Shah
Ciena Corporation

March 20, 2020

YANG Data Model for MPLS LDP
draft-ietf-mpls-ldp-yang-09

Abstract

This document describes a YANG data model for Multi-Protocol Label Switching (MPLS) Label Distribution Protocol (LDP). The model also serves as the base model to define Multipoint LDP (mLDP) model.

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA).

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 September 21, 2020.

Copyright Notice

Table of Contents

1. Introduction

The Network Configuration Protocol (NETCONF) [RFC6241] is one of the network management protocols that defines mechanisms to manage network devices. YANG [RFC6020] [RFC7950] is a modular language that represents data structures in an XML tree format, and is used as a data modelling language for the NETCONF.

This document introduces a YANG data model for MPLS Label Distribution Protocol (LDP) [RFC5036]. This model also covers LDP IPv6 [RFC7552] and LDP capabilities [RFC5561] specifications.

The data model is defined for the following constructs that are used for managing the protocol:

* Configuration

* Operational State

* Executables (Actions)

* Notifications

This document is organized to define the data model for each of the above constructs in the sequence as listed above.

1.1. Base and Extended

The configuration and state items are divided into the following two broad categories:

* Base

* Extended

The "base" category contains the basic and fundamental features that are covered in LDP base specification [RFC5036] and constitute the minimum requirements for a typical base LDP deployment. Whereas, the "extended" category contains other non-base features. All the items in a base category are mandatory and hence no "if-feature" is allowed under the "base" category. The base and extended categories are defined in their own modules as described later.

The example of base feature includes the configuration of LDP lsr-id, enabling LDP interfaces, setting password for LDP session etc., whereas the examples of extended feature include inbound/outbound label policies, igp sync [RFC5443], downstream-on-demand etc. It is

worth highlighting that LDP IPv6 [RFC7552] is also categorized as an extended feature.

While "base" model support will suffice for small deployments, it is expected that large deployments will require both the "base" and "extended" models support from the vendors.

2. Specification of Requirements

In this document, the word "IP" is used to refer to both IPv4 and IPv6, unless otherwise explicitly stated. For example, "IP address family" should be read as "IPv4 and/or IPv6 address family".

3. Overview

This document defines two new modules for LDP YANG support:

* "ietf-mpls-ldp" module that specifies the base LDP features and augments /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol defined in [RFC8349]. We define new identity 'mpls-ldp' for LDP and the model allows only a single instance of 'mpls-ldp'.

* "ietf-mpls-ldp-extended" module that specifies the extended LDP features and augments the base LDP module.

It is to be noted that mLDP YANG model [I-D.ietf-mpls-mldp-yang] augments LDP base and extended modules to specify the mLDP specific base and extended features.

There are four types of containers in our module(s):

* Read-Write parameters for configuration (Section 5)

* Read-only parameters for operational state (Section 6)

* Notifications for events (Section 7)

* RPCs for executing commands to perform some action (Section 8)

The modules in this document conforms to the Network Management Datastore Architecture (NMDA) defined in [RFC8342]. The operational state data is combined with the associated configuration data in the same hierarchy [RFC8407]. When protocol states are retrieved from the NMDA operational state datastore, the returned states cover all "config true" (rw) and "config false" (ro) nodes defined in the schema.

Following diagram depicts high level LDP YANG tree organization and hierarchy:

```
          +-- rw routing
             +-- rw control-plane-protocols
              +-- rw control-plane-protocol
               +-- rw mpls-ldp
                   +-- rw ...
                       +-- rw ...                  // base
                       |   +-- rw ...
                       |   +-- ro ...
                       |   +--
                       +-- ro ...
                       |   +-- ro ...
                       |   +-- ro ...
                       |   +--
                       +-- rw ldp-ext: ....        // extended
                       |   +-- rw ...
                       |   +-- ro ...
                       |   +--
                       +-- ro ...
                           +-- ro ...
                           +-- ro ...


      rpcs:
         +-- x mpls-ldp-some_action
         +-- x . . . . .

      notifications:
         +--- n mpls-ldp-some_event
         +--- n ...
```

Figure 1: LDP YANG tree organization

Before going into data model details, it is important to take note of the following points:

* This model aims to address only the core LDP parameters as per RFC specification, as well as well-known and widely deployed manageability controls (such as label filtering policies to apply filtering rules on the assignment, advertisement, and acceptance for label bindings). Any vendor specific feature should be defined in a vendor-specific augmentation of this model.

* Multi-topology LDP [RFC7307] is beyond the scope of this document.

* This model does not cover any applications running on top of LDP, nor does it cover any OAM procedures for LDP.

* This model is a VPN Routing and Forwarding (VRF)-centric model. It is important to note that [RFC4364] defines VRF tables and default forwarding tables as different, however from a YANG modelling perspective this introduces unnecessary complications, hence we are treating the default forwarding table as just another VRF.

* A "network-instance", as defined in [RFC8529], refers to a VRF instance (both default and non-default) within the scope of this model.

* This model supports two address-families, namely "ipv4" and "ipv6".

* This model assumes platform-wide label space (i.e. label space Id of zero). However, when Upstream Label assignment [RFC6389] is in use, an upstream assigned label is looked up in a Context-Specific label space as defined in [RFC5331].

* The label and peer policies (including filters) are defined using prefix-set and neighbor-set respectively as defined in routing-policy model [I-D.ietf-rtgwg-policy-model].

* This model uses the terms LDP "neighbor"/"adjacency", "session", and "peer" with the following semantics:

  - Neighbor/Adjacency: An LDP enabled LSR that is discovered through LDP discovery mechanisms.

  - Session: An LDP neighbor with whom a TCP connection has been established.

  - Peer: An LDP session which has successfully progressed beyond its initialization phase and is either already exchanging the bindings or is ready to do so.

  It is to be noted that LDP Graceful Restart (GR) mechanisms defined in [RFC3478] allow keeping the exchanged bindings for some time after a session goes down with a peer. We call such a state belonging to a "stale" peer -- i.e. keeping peer bindings from a peer with whom currently there is either no connection established or connection is established but GR session is in recovery state. When used in this document, the above terms will refer strictly to the semantics and definitions defined for them.

A simplified graphical tree representation of base and extended LDP YANG data model is presented in Figure 2. The meaning of the symbols in these tree diagrams is defined in [RFC8340].

The actual YANG specification for base and extended modules is captured in Section 9.

While presenting the YANG tree view and actual specification, this document assumes readers' familiarity with the concepts of YANG modeling, its presentation and its compilation.

4. The Complete Tree

Following is a complete tree representation of configuration, state, notification, and RPC items under LDP base and extended modules.

```
module: ietf-mpls-ldp
  augment /rt:routing/rt:control-plane-protocols
            /rt:control-plane-protocol:
    +--rw mpls-ldp
       +--rw global
          |  +--rw capability
          |  |  +--rw ldp-ext:end-of-lib {capability-end-of-lib}?
          |  |  |  +--rw ldp-ext:enabled?   boolean
          |  |  +--rw ldp-ext:typed-wildcard-fec
          |  |  |       {capability-typed-wildcard-fec}?
          |  |  |  +--rw ldp-ext:enabled?   boolean
          |  |  +--rw ldp-ext:upstream-label-assignment
          |  |  |       {capability-upstream-label-assignment}?
          |  |  |  +--rw ldp-ext:enabled?   boolean
          |  +--rw graceful-restart
          |  |  +--rw enabled?                  boolean
          |  |  +--rw reconnect-time?           uint16
          |  |  +--rw recovery-time?            uint16
          |  |  +--rw forwarding-holdtime?      uint16
          |  |  +--rw ldp-ext:helper-enabled?   boolean
          |  |  |       {graceful-restart-helper-mode}?
          |  +--rw lsr-id?
          |  |     rt-types:router-id
          |  +--rw address-families
          |  |  +--rw ipv4!
          |  |  |  +--rw enabled?                              boolean
          |  |  |  +--ro label-distribution-control-mode?   enumeration
          |  |  |  +--ro bindings
          |  |  |  |  +--ro address* [address]
          |  |  |  |  |  +--ro address               inet:ipv4-address
          |  |  |  |  |  +--ro advertisement-type?   advertised-received
          |  |  |  |  |  +--ro peer
```

```
|  |  |  |  |  |     +--ro lsr-id?           leafref
|  |  |  |  |  |     +--ro label-space-id?   leafref
|  |  |  |  |  +--ro fec-label* [fec]
|  |  |  |  |     +--ro fec     inet:ipv4-prefix
|  |  |  |  |     +--ro peer*
|  |  |  |  |              [lsr-id label-space-id advertisement-type]
|  |  |  |  |        +--ro lsr-id                leafref
|  |  |  |  |        +--ro label-space-id        leafref
|  |  |  |  |        +--ro advertisement-type
|  |  |  |  |        |       advertised-received
|  |  |  |  |        +--ro label?
|  |  |  |  |        |       rt-types:mpls-label
|  |  |  |  |        +--ro used-in-forwarding?   boolean
|  |  |  |  +--rw ldp-ext:label-policy
|  |  |  |  |  +--rw ldp-ext:advertise
|  |  |  |  |  |  +--rw ldp-ext:egress-explicit-null
|  |  |  |  |  |  |  +--rw ldp-ext:enabled?   boolean
|  |  |  |  |  |  +--rw ldp-ext:prefix-list?
|  |  |  |  |  |          prefix-list-ref
|  |  |  |  |  +--rw ldp-ext:accept
|  |  |  |  |  |  +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  |  |  |  +--rw ldp-ext:assign
|  |  |  |  |           {policy-label-assignment-config}?
|  |  |  |  |     +--rw ldp-ext:independent-mode
|  |  |  |  |     |  +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  |  |  |     +--rw ldp-ext:ordered-mode
|  |  |  |  |              {policy-ordered-label-config}?
|  |  |  |  |        +--rw ldp-ext:egress-prefix-list?
|  |  |  |  |                 prefix-list-ref
|  |  |  |  +--rw ldp-ext:transport-address?
|  |  |  |            inet:ipv4-address
|  |  |  +--rw ldp-ext:ipv6!
|  |  |     +--rw ldp-ext:enabled?
|  |  |     |       boolean
|  |  |     +--rw ldp-ext:label-policy
|  |  |     |  +--rw ldp-ext:advertise
|  |  |     |  |  +--rw ldp-ext:egress-explicit-null
|  |  |     |  |  |  +--rw ldp-ext:enabled?   boolean
|  |  |     |  |  +--rw ldp-ext:prefix-list?
|  |  |     |  |          prefix-list-ref
|  |  |     |  +--rw ldp-ext:accept
|  |  |     |  |  +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  |     |  +--rw ldp-ext:assign
|  |  |     |           {policy-label-assignment-config}?
|  |  |     |     +--rw ldp-ext:independent-mode
|  |  |     |     |  +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  |     |     +--rw ldp-ext:ordered-mode
|  |  |     |              {policy-ordered-label-config}?
```

```
|  |  |     |        +--rw ldp-ext:egress-prefix-list?
|  |  |     |                 prefix-list-ref
|  |  |     +--rw ldp-ext:transport-address
|  |  |     |       inet:ipv6-address
|  |  |     +--ro ldp-ext:label-distribution-control-mode?
|  |  |     |       enumeration
|  |  |     +--ro ldp-ext:bindings
|  |  |        +--ro ldp-ext:address* [address]
|  |  |        |  +--ro ldp-ext:address
|  |  |        |  |       inet:ipv6-address
|  |  |        |  +--ro ldp-ext:advertisement-type?
|  |  |        |  |       advertised-received
|  |  |        |  +--ro ldp-ext:peer
|  |  |        |     +--ro ldp-ext:lsr-id?           leafref
|  |  |        |     +--ro ldp-ext:label-space-id?   leafref
|  |  |        +--ro ldp-ext:fec-label* [fec]
|  |  |           +--ro ldp-ext:fec     inet:ipv6-prefix
|  |  |           +--ro ldp-ext:peer*
|  |  |                    [lsr-id label-space-id advertisement-type]
|  |  |              +--ro ldp-ext:lsr-id                leafref
|  |  |              +--ro ldp-ext:label-space-id        leafref
|  |  |              +--ro ldp-ext:advertisement-type
|  |  |              |       advertised-received
|  |  |              +--ro ldp-ext:label?
|  |  |              |       rt-types:mpls-label
|  |  |              +--ro ldp-ext:used-in-forwarding?   boolean
|  |  +--rw ldp-ext:forwarding-nexthop
|  |  |       {forwarding-nexthop-config}?
|  |  |  +--rw ldp-ext:interfaces
|  |  |     +--rw ldp-ext:interface* [name]
|  |  |        +--rw ldp-ext:name              if:interface-ref
|  |  |        +--rw ldp-ext:address-family* [afi]
|  |  |           +--rw ldp-ext:afi            identityref
|  |  |           +--rw ldp-ext:ldp-disable?   boolean
|  |  +--rw ldp-ext:igp-synchronization-delay?   uint16
|  +--rw discovery
|  |  +--rw interfaces
|  |  |  +--rw hello-holdtime?   uint16
|  |  |  +--rw hello-interval?   uint16
|  |  |  +--rw interface* [name]
|  |  |     +--rw name
|  |  |     |       if:interface-ref
|  |  |     +--ro next-hello?                          uint16
|  |  |     +--rw address-families
|  |  |     |  +--rw ipv4!
|  |  |     |  |  +--rw enabled?                     boolean
|  |  |     |  |  +--ro hello-adjacencies
|  |  |     |  |  |  +--ro hello-adjacency* [adjacent-address]
```

```
|  |  |  |  |  |     +--ro adjacent-address
|  |  |  |  |  |     |       inet:ipv4-address
|  |  |  |  |  |     +--ro flag*               identityref
|  |  |  |  |  |     +--ro hello-holdtime
|  |  |  |  |  |     |  +--ro adjacent?     uint16
|  |  |  |  |  |     |  +--ro negotiated?   uint16
|  |  |  |  |  |     |  +--ro remaining?    uint16
|  |  |  |  |  |     +--ro next-hello?         uint16
|  |  |  |  |  |     +--ro statistics
|  |  |  |  |  |     |  +--ro discontinuity-time
|  |  |  |  |  |     |  |       yang:date-and-time
|  |  |  |  |  |     |  +--ro hello-received?
|  |  |  |  |  |     |  |       yang:counter64
|  |  |  |  |  |     |  +--ro hello-dropped?
|  |  |  |  |  |     |          yang:counter64
|  |  |  |  |  |     +--ro peer
|  |  |  |  |  |        +--ro lsr-id?           leafref
|  |  |  |  |  |        +--ro label-space-id?   leafref
|  |  |  |  |  +--rw ldp-ext:transport-address?   union
|  |  |  |  +--rw ldp-ext:ipv6!
|  |  |  |     +--rw ldp-ext:enabled?             boolean
|  |  |  |     +--ro ldp-ext:hello-adjacencies
|  |  |  |     |  +--ro ldp-ext:hello-adjacency*
|  |  |  |     |          [adjacent-address]
|  |  |  |     |     +--ro ldp-ext:adjacent-address
|  |  |  |     |     |       inet:ipv6-address
|  |  |  |     |     +--ro ldp-ext:flag*
|  |  |  |     |     |       identityref
|  |  |  |     |     +--ro ldp-ext:hello-holdtime
|  |  |  |     |     |  +--ro ldp-ext:adjacent?     uint16
|  |  |  |     |     |  +--ro ldp-ext:negotiated?   uint16
|  |  |  |     |     |  +--ro ldp-ext:remaining?    uint16
|  |  |  |     |     +--ro ldp-ext:next-hello?         uint16
|  |  |  |     |     +--ro ldp-ext:statistics
|  |  |  |     |     |  +--ro ldp-ext:discontinuity-time
|  |  |  |     |     |  |       yang:date-and-time
|  |  |  |     |     |  +--ro ldp-ext:hello-received?
|  |  |  |     |     |  |       yang:counter64
|  |  |  |     |     |  +--ro ldp-ext:hello-dropped?
|  |  |  |     |     |          yang:counter64
|  |  |  |     |     +--ro ldp-ext:peer
|  |  |  |     |        +--ro ldp-ext:lsr-id?           leafref
|  |  |  |     |        +--ro ldp-ext:label-space-id?   leafref
|  |  |  |     +--rw ldp-ext:transport-address?   union
|  |  |  +--rw ldp-ext:hello-holdtime?              uint16
|  |  |  |       {per-interface-timer-config}?
|  |  |  +--rw ldp-ext:hello-interval?              uint16
|  |  |          {per-interface-timer-config}?
```

```
|  |     +--rw ldp-ext:igp-synchronization-delay?   uint16
|  |               {per-interface-timer-config}?
|  +--rw targeted
|     +--rw hello-holdtime?     uint16
|     +--rw hello-interval?     uint16
|     +--rw hello-accept
|     |  +--rw enabled?                 boolean
|     |  +--rw ldp-ext:neighbor-list?   neighbor-list-ref
|     |           {policy-targeted-discovery-config}?
|     +--rw address-families
|        +--rw ipv4!
|        |  +--ro hello-adjacencies
|        |  |  +--ro hello-adjacency*
|        |  |           [local-address adjacent-address]
|        |  |     +--ro local-address       inet:ipv4-address
|        |  |     +--ro adjacent-address    inet:ipv4-address
|        |  |     +--ro flag*               identityref
|        |  |     +--ro hello-holdtime
|        |  |     |  +--ro adjacent?     uint16
|        |  |     |  +--ro negotiated?   uint16
|        |  |     |  +--ro remaining?    uint16
|        |  |     +--ro next-hello?         uint16
|        |  |     +--ro statistics
|        |  |     |  +--ro discontinuity-time
|        |  |     |  |       yang:date-and-time
|        |  |     |  +--ro hello-received?
|        |  |     |  |       yang:counter64
|        |  |     |  +--ro hello-dropped?
|        |  |     |           yang:counter64
|        |  |     +--ro peer
|        |  |        +--ro lsr-id?           leafref
|        |  |        +--ro label-space-id?   leafref
|        |  +--rw target* [adjacent-address]
|        |     +--rw adjacent-address    inet:ipv4-address
|        |     +--rw enabled?            boolean
|        |     +--rw local-address?      inet:ipv4-address
|        +--rw ldp-ext:ipv6!
|           +--ro ldp-ext:hello-adjacencies
|           |  +--ro ldp-ext:hello-adjacency*
|           |           [local-address adjacent-address]
|           |     +--ro ldp-ext:local-address
|           |     |       inet:ipv6-address
|           |     +--ro ldp-ext:adjacent-address
|           |     |       inet:ipv6-address
|           |     +--ro ldp-ext:flag*
|           |     |       identityref
|           |     +--ro ldp-ext:hello-holdtime
|           |     |  +--ro ldp-ext:adjacent?     uint16
```

```
|           |     |  +--ro ldp-ext:negotiated?   uint16
|           |     |  +--ro ldp-ext:remaining?    uint16
|           |     +--ro ldp-ext:next-hello?         uint16
|           |     +--ro ldp-ext:statistics
|           |     |  +--ro ldp-ext:discontinuity-time
|           |     |  |       yang:date-and-time
|           |     |  +--ro ldp-ext:hello-received?
|           |     |  |       yang:counter64
|           |     |  +--ro ldp-ext:hello-dropped?
|           |     |          yang:counter64
|           |     +--ro ldp-ext:peer
|           |        +--ro ldp-ext:lsr-id?           leafref
|           |        +--ro ldp-ext:label-space-id?   leafref
|           +--rw ldp-ext:target* [adjacent-address]
|              +--rw ldp-ext:adjacent-address
|              |       inet:ipv6-address
|              +--rw ldp-ext:enabled?           boolean
|              +--rw ldp-ext:local-address?
|                      inet:ipv6-address
+--rw peers
   +--rw authentication
   |  +--rw (authentication-type)?
   |     +--:(password)
   |     |  +--rw key?                 string
   |     |  +--rw crypto-algorithm?    identityref
   |     +--:(ldp-ext:key-chain) {key-chain}?
   |        +--rw ldp-ext:key-chain?   key-chain:key-chain-ref
   +--rw session-ka-holdtime?                        uint16
   +--rw session-ka-interval?                        uint16
   +--rw peer* [lsr-id label-space-id]
      |  +--rw lsr-id                          rt-types:router-id
      |  +--rw label-space-id                  uint16
      |  +--rw authentication
      |  |  +--rw (authentication-type)?
      |  |     +--:(password)
      |  |     |  +--rw key?                 string
      |  |     |  +--rw crypto-algorithm?    identityref
      |  |     +--:(ldp-ext:key-chain) {key-chain}?
      |  |        +--rw ldp-ext:key-chain?
      |  |                key-chain:key-chain-ref
      |  +--rw address-families
      |  |  +--rw ipv4!
      |  |  |  +--ro hello-adjacencies
      |  |  |  |  +--ro hello-adjacency*
      |  |  |  |          [local-address adjacent-address]
      |  |  |  |     +--ro local-address       inet:ipv4-address
      |  |  |  |     +--ro adjacent-address    inet:ipv4-address
      |  |  |  |     +--ro flag*               identityref
```

```
|  |  |  |  |     +--ro hello-holdtime
|  |  |  |  |     |  +--ro adjacent?     uint16
|  |  |  |  |     |  +--ro negotiated?   uint16
|  |  |  |  |     |  +--ro remaining?    uint16
|  |  |  |  |     +--ro next-hello?         uint16
|  |  |  |  |     +--ro statistics
|  |  |  |  |     |  +--ro discontinuity-time
|  |  |  |  |     |  |       yang:date-and-time
|  |  |  |  |     |  +--ro hello-received?
|  |  |  |  |     |  |       yang:counter64
|  |  |  |  |     |  +--ro hello-dropped?
|  |  |  |  |     |          yang:counter64
|  |  |  |  |     +--ro interface?          if:interface-ref
|  |  |  |  +--rw ldp-ext:label-policy
|  |  |  |     +--rw ldp-ext:advertise
|  |  |  |     |  +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  |  |     +--rw ldp-ext:accept
|  |  |  |        +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  |  +--rw ldp-ext:ipv6!
|  |  |     +--ro ldp-ext:hello-adjacencies
|  |  |     |  +--ro ldp-ext:hello-adjacency*
|  |  |     |           [local-address adjacent-address]
|  |  |     |     +--ro ldp-ext:local-address
|  |  |     |     |       inet:ipv6-address
|  |  |     |     +--ro ldp-ext:adjacent-address
|  |  |     |     |       inet:ipv6-address
|  |  |     |     +--ro ldp-ext:flag*
|  |  |     |     |       identityref
|  |  |     |     +--ro ldp-ext:hello-holdtime
|  |  |     |     |  +--ro ldp-ext:adjacent?     uint16
|  |  |     |     |  +--ro ldp-ext:negotiated?   uint16
|  |  |     |     |  +--ro ldp-ext:remaining?    uint16
|  |  |     |     +--ro ldp-ext:next-hello?         uint16
|  |  |     |     +--ro ldp-ext:statistics
|  |  |     |     |  +--ro ldp-ext:discontinuity-time
|  |  |     |     |  |       yang:date-and-time
|  |  |     |     |  +--ro ldp-ext:hello-received?
|  |  |     |     |  |       yang:counter64
|  |  |     |     |  +--ro ldp-ext:hello-dropped?
|  |  |     |     |          yang:counter64
|  |  |     |     +--ro ldp-ext:interface?
|  |  |     |             if:interface-ref
|  |  |     +--rw ldp-ext:label-policy
|  |  |        +--rw ldp-ext:advertise
|  |  |        |  +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  |        +--rw ldp-ext:accept
|  |  |           +--rw ldp-ext:prefix-list?   prefix-list-ref
|  |  +--ro label-advertisement-mode
```

```
|  |  +--ro local?        label-adv-mode
|  |  +--ro peer?         label-adv-mode
|  |  +--ro negotiated?   label-adv-mode
|  +--ro next-keep-alive?              uint16
|  +--ro received-peer-state
|  |  +--ro graceful-restart
|  |  |  +--ro enabled?          boolean
|  |  |  +--ro reconnect-time?   uint16
|  |  |  +--ro recovery-time?    uint16
|  |  +--ro capability
|  |     +--ro end-of-lib
|  |     |  +--ro enabled?   boolean
|  |     +--ro typed-wildcard-fec
|  |     |  +--ro enabled?   boolean
|  |     +--ro upstream-label-assignment
|  |        +--ro enabled?   boolean
|  +--ro session-holdtime
|  |  +--ro peer?         uint16
|  |  +--ro negotiated?   uint16
|  |  +--ro remaining?    uint16
|  +--ro session-state?                enumeration
|  +--ro tcp-connection
|  |  +--ro local-address?    inet:ip-address
|  |  +--ro local-port?       inet:port-number
|  |  +--ro remote-address?   inet:ip-address
|  |  +--ro remote-port?      inet:port-number
|  +--ro up-time?
|  |       rt-types:timeticks64
|  +--ro statistics
|  |  +--ro discontinuity-time          yang:date-and-time
|  |  +--ro received
|  |  |  +--ro total-octets?          yang:counter64
|  |  |  +--ro total-messages?        yang:counter64
|  |  |  +--ro address?               yang:counter64
|  |  |  +--ro address-withdraw?      yang:counter64
|  |  |  +--ro initialization?        yang:counter64
|  |  |  +--ro keepalive?             yang:counter64
|  |  |  +--ro label-abort-request?   yang:counter64
|  |  |  +--ro label-mapping?         yang:counter64
|  |  |  +--ro label-release?         yang:counter64
|  |  |  +--ro label-request?         yang:counter64
|  |  |  +--ro label-withdraw?        yang:counter64
|  |  |  +--ro notification?          yang:counter64
|  |  +--ro sent
|  |  |  +--ro total-octets?          yang:counter64
|  |  |  +--ro total-messages?        yang:counter64
|  |  |  +--ro address?               yang:counter64
|  |  |  +--ro address-withdraw?      yang:counter64
```

```
            |  |  |  +--ro initialization?        yang:counter64
            |  |  |  +--ro keepalive?             yang:counter64
            |  |  |  +--ro label-abort-request?   yang:counter64
            |  |  |  +--ro label-mapping?         yang:counter64
            |  |  |  +--ro label-release?         yang:counter64
            |  |  |  +--ro label-request?         yang:counter64
            |  |  |  +--ro label-withdraw?        yang:counter64
            |  |  |  +--ro notification?          yang:counter64
            |  |  +--ro total-addresses?            uint32
            |  |  +--ro total-labels?               uint32
            |  |  +--ro total-fec-label-bindings?   uint32
            |  +--rw ldp-ext:admin-down?            boolean
            |  |       {per-peer-admin-down}?
            |  +--rw ldp-ext:graceful-restart
            |  |       {per-peer-graceful-restart-config}?
            |  |  +--rw ldp-ext:enabled?          boolean
            |  |  +--rw ldp-ext:reconnect-time?   uint16
            |  |  +--rw ldp-ext:recovery-time?    uint16
            |  +--rw ldp-ext:session-ka-holdtime?   uint16
            |  |       {per-peer-session-attributes-config}?
            |  +--rw ldp-ext:session-ka-interval?   uint16
            |          {per-peer-session-attributes-config}?
            +--rw ldp-ext:session-downstream-on-demand
            |       {session-downstream-on-demand-config}?
            |  +--rw ldp-ext:enabled?     boolean
            |  +--rw ldp-ext:peer-list?   peer-list-ref
            +--rw ldp-ext:dual-stack-transport-preference
                    {peers-dual-stack-transport-preference}?
               +--rw ldp-ext:max-wait?      uint16
               +--rw ldp-ext:prefer-ipv4!
                  +--rw ldp-ext:peer-list?   peer-list-ref

rpcs:
  +---x mpls-ldp-clear-peer
  |  +---w input
  |     +---w protocol-name?    leafref
  |     +---w lsr-id?           leafref
  |     +---w label-space-id?   leafref
  +---x mpls-ldp-clear-hello-adjacency
  |  +---w input
  |     +---w hello-adjacency
  |        +---w protocol-name?    leafref
  |        +---w (hello-adjacency-type)?
  |           +--:(targeted)
  |           |  +---w targeted!
  |           |     +---w target-address?   inet:ip-address
  |           +--:(link)
  |              +---w link!
```

```
      |                  +---w next-hop-interface?   leafref
      |                  +---w next-hop-address?     inet:ip-address
      +---x mpls-ldp-clear-peer-statistics
         +---w input
            +---w protocol-name?    leafref
            +---w lsr-id?           leafref
            +---w label-space-id?   leafref

    notifications:
      +---n mpls-ldp-peer-event
      |  +--ro event-type?   oper-status-event-type
      |  +--ro peer
      |     +--ro protocol-name?    leafref
      |     +--ro lsr-id?           leafref
      |     +--ro label-space-id?   leafref
      +---n mpls-ldp-hello-adjacency-event
      |  +--ro event-type?       oper-status-event-type
      |  +--ro protocol-name?    leafref
      |  +--ro (hello-adjacency-type)?
      |     +--:(targeted)
      |     |  +--ro targeted
      |     |     +--ro target-address?   inet:ip-address
      |     +--:(link)
      |        +--ro link
      |           +--ro next-hop-interface?   if:interface-ref
      |           +--ro next-hop-address?     inet:ip-address
      +---n mpls-ldp-fec-event
         +--ro event-type?      oper-status-event-type
         +--ro protocol-name?   leafref
         +--ro fec?             inet:ip-prefix
```

Figure 2: Complete Tree

5. Configuration

This specification defines the configuration parameters for base LDP as specified in [RFC5036] and LDP IPv6 [RFC7552]. Moreover, it incorporates provisions to enable LDP Capabilities [RFC5561], and defines some of the most significant and commonly used capabilities such as Typed Wildcard FEC [RFC5918], End-of-LIB [RFC5919], and LDP Upstream Label Assignment [RFC6389].

This model augments /rt:routing/rt:control-plane-protocols/ rt:control-plane-protocol that is defined in [RFC8349] and follows NMDA as mentioned earlier.

Following is the high-level configuration organization for base LDP module:

```
        augment /rt:routing/rt:control-plane-protocols:
                  /rt:control-plane-protocol:
              +-- mpls-ldp
                  +-- global
                  |    +-- ...
                  |    +-- ...
                  |    +-- address-families
                  |    |   +-- ipv4
                  |    |     +-- . . .
                  |    |     +-- . . .
                  |    +-- capability
                  |        +-- ...
                  |        +-- ...
                  +-- discovery
                  |    +-- interfaces
                  |    |   +-- ...
                  |    |   +-- ...
                  |    |   +-- interface* [interface]
                  |    |     +-- ...
                  |    |     +-- address-families
                  |    |       +-- ipv4
                  |    |         +-- ...
                  |    |         +-- ...
                  |    +-- targeted
                  |        +-- ...
                  |        +-- address-families
                  |            +-- ipv4
                  |                +- target* [adjacent-address]
                  |                  +- ...
                  |                  +- ...
                  +-- peers
                       +-- ...
                       +-- ...
                       +-- peer* [lsr-id label-space-id]
                           +-- ...
                           +-- ...
```

Figure 3: Base Configuration organization

Following is the high-level configuration organization for extended LDP:

```
           augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protoc
ol
                 +-- mpls-ldp
                     +-- global
                     |    +-- ...
                     |    +-- ...
                     |    +-- address-families
                     |    |   +-- ipv4
                     |    |   | +-- . . .
                     |    |   | +-- . . .
                     |    |   | +-- label-policy
                     |    |   |   +-- ...
                     |    |   |   +-- ...
                     |    |   +-- ipv6
                     |    |     +-- . . .
                     |    |     +-- . . .
                     |    |     +-- label-policy
                     |    |       +-- ...
                     |    |       +-- ...
                     |    +-- capability
                     |    |   +-- ...
                     |    |   +-- ...
                     |    +-- discovery
                     |        +-- interfaces
                     |        |   +-- ...
                     |        |   +-- ...
                     |        |   +-- interface* [interface]
                     |        |     +-- ...
                     |        |     +-- address-families
                     |        |       +-- ipv4
                     |        |       | +-- ...
                     |        |       | +-- ...
                     |        |       +-- ipv6
                     |        |         +-- ...
                     |        |         +-- ...
                     |        +-- targeteted
                     |            +-- ...
                     |            +-- address-families
                     |              +-- ipv6
                     |                +- target* [adjacent-address]
                     |                  +- ...
                     |                  +- ...
                     +-- forwarding-nexthop
                     |   +-- ...
                     |   +-- ...
                     +-- peers
                          +-- ...
                          +-- ...
                          +-- peer*
```

```
        +-- ...
        +-- ...
        +-- label-policy
        |   +-- ..
        +-- address-families
           +-- ipv4
           |   +-- ...
           +-- ipv6
               +-- ...
```

Figure 4: Extended Configuration organization

Given the configuration hierarchy, the model allows inheritance such that an item in a child tree is able to derive value from a similar or related item in one of the parents. For instance, hello holdtime can be configured per-VRF or per-VRF-interface, thus allowing inheritance as well flexibility to override with a different value at any child level.

5.1. Configuration Hierarchy

LDP module resides under a network-instance and the scope of any LDP configuration defined under this tree is per network-instance (per-VRF). This configuration is further divided into sub categories as follows.

* Global parameters
* Per-address-family parameters
* LDP Capabilities parameters
* Hello Discovery parameters
  - interfaces
    - Global
    - Per-interface: Global
    - Per-interface: Per-address-family
  - targeted
    - Global

- o Per-address-family: Per-target

* Peer parameters
  - Global
  - Per-peer: Global
  - Per-peer: Per-address-family
* Forwarding parameters

Following subsections briefly explain these configuration areas.

5.1.1. Global parameters

There are configuration items that are available directly under a VRF instance and do not fall under any other sub tree. Example of such a parameter is LDP LSR Id that is typically configured per VRF. To keep legacy LDP features and applications working in an LDP IPv4 networks with this model, this document recommends an operator to pick a routable IPv4 unicast address (within a routing domain) as an LSR Id.

5.1.2. Capabilities parameters

This container falls under the global tree and holds the LDP capabilities that are to be enabled for certain features. By default, an LDP capability is disabled unless explicitly enabled. These capabilities are typically used to negotiate with LDP peer(s) the support/non-support related to a feature and its parameters. The scope of a capability enabled under this container applies to all LDP peers in the given VRF instance. There is also a peer level capability container that is provided to override a capability that is enabled/specified at VRF level.

5.1.3. Per-Address-Family parameters

Any LDP configuration parameter related to IP address family (AF) whose scope is VRF wide is configured under this tree. The examples of per-AF parameters include enabling LDP for an address family, prefix-list based label policies, and LDP transport address.

5.1.4. Hello Discovery parameters

This container is used to hold LDP configuration related to Hello and discovery process for both basic (link) and extended (targeted) discovery.

The "interfaces" is a container to configure parameters related to VRF interfaces. There are parameters that apply to all interfaces (such as hello timers), as well as parameters that can be configured per-interface. Hence, an interface list is defined under "interfaces" container. The model defines parameters to configure per-interface non AF related items, as well as per-interface per-AF items. The example of the former is interface hello timers, and example of the latter is enabling hellos for a given AF under an interface.

The "targeted" container under a VRF instance allows to configure LDP targeted discovery related parameters. Within this container, the "target" list provides a means to configure multiple target addresses to perform extended discovery to a specific destination target, as well as to fine-tune the per-target parameters.

5.1.5. Peer parameters

This container is used to hold LDP configuration related to LDP sessions and peers under a VRF instance. This container allows to configure parameters that either apply on VRF's all peers or a subset (peer-list) of VRF peers. The example of such parameters include authentication password, session KA timers etc. Moreover, the model also allows per-peer parameter tuning by specifying a "peer" list under the "peers" container. A peer is uniquely identified by its LSR Id.

Like per-interface parameters, some per-peer parameters are AF-agnostic (i.e. either non AF related or apply to both IP address families), and some that belong to an AF. The example of the former is per-peer session password configuration, whereas the example of the latter is prefix-list based label policies (inbound and outbound) that apply to a given peer.

5.1.6. Forwarding parameters

This container is used to hold configuration used to control LDP forwarding behavior under a VRF instance. One example of a configuration under this container is when a user wishes to enable neighbor discovery on an interface but wishes to disable use of the same interface as forwarding nexthop. This example configuration makes sense only when there are more than one LDP enabled interfaces towards the neighbor.

6. Operational State

Operational state of LDP can be queried and obtained from read-only state containers that fall under the same tree (/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol) as the configuration.

Following are main areas for which LDP operational state is defined:

* Neighbor Adjacencies
* Peer
* Bindings (FEC-label and address)
* Capabilities

6.1. Adjacency state

Neighbor adjacencies are per address-family hello adjacencies that are formed with neighbors as result of LDP basic or extended discovery. In terms of organization, there is a source of discovery (e.g. interface or target address) along with its associated parameters and one or more discovered neighbors along with neighbor discovery related parameters. For the basic discovery, there could be more than one discovered neighbor for a given source (interface), whereas there is at most one discovered neighbor for an extended discovery source (local-address and target-address). It is also to be noted that the reason for a targeted neighbor adjacency could be either an active source (locally configured targeted) or passive source (to allow any incoming extended/targeted hellos). A neighbor/adjacency record also contains session-state that helps highlight whether a given adjacency has progressed to subsequent session level or to eventual peer level.

Following captures high level tree hierarchy for neighbor adjacency state. The tree is shown for ipv4 address-family only; a similar tree exists for ipv6 address-family as well.

```
+--rw mpls-ldp!
   +--rw discovery
      +--rw interfaces
      |  +--rw interface* [interface]
      |     +--rw address-families
      |        +--rw ipv4
      |           +--ro hello-adjacencies
      |              +--ro hello-adjacencies* [adjacent-address]
      |                 +--ro adjacent-address
      |                    . . . .
      |                    . . . .
      +--rw targeted
         +--rw address-families
            +--rw ipv4
               +--ro hello-adjacencies
                  +--ro hello-adjacencies*
                  |                  [local-address adjacent-address]
                  +--ro local-address
                     +--ro adjacent-address
                        . . . .
                        . . . .
```

Figure 5: Adjacency state

6.2. Peer state

Peer related state is presented under peers tree. This is one of the core state that provides info on the session related parameters (mode, authentication, KA timeout etc.), TCP connection info, hello adjacencies for the peer, statistics related to messages and bindings, and capabilities exchange info.

Following captures high level tree hierarchy for peer state. The peer's hello adjacencies tree is shown for ipv4 address-family only; a similar tree exists for ipv6 address-family as well.

```
      +--rw mpls-ldp!
          +--rw peers
             +--rw peer* [lsr-id label-space-id]
                 +--rw lsr-id
                 +--rw label-space-id
                 +--ro label-advertisement-mode
                 +--ro session-state
                 +--ro tcp-connection
                 +--ro session-holdtime?
                 +--ro up-time
                 +-- . . . .
                 +--ro address-families
                 |  +--ro ipv4
                 |     +--ro hello-adjacencies
                 |        +--ro hello-adjacencies*
                 |                          [local-address adjacent-address]
                 |           . . . .
                 |           . . . .
                 +--ro received-peer-state
                 |  +--ro . . . .
                 |  +--ro capability
                 |     +--ro . . . .
                 +--ro statistics
                    +-- . . . .
                    +-- received
                    |   +-- ...
                    +-- sent
                        +-- ...
```

Figure 6: Peer state

6.3. Bindings state

Binding state provides information on LDP FEC-label bindings as well as address binding for both inbound (received) as well as outbound (advertised) direction. FEC-label bindings are presented as a FEC-centric view, and address bindings are presented as an address-centric view:

```
        FEC-Label bindings:
            FEC 203.0.113.1/32:
              advertised: local-label 16000
                peer 192.0.2.1:0
                peer 192.0.2.2:0
                peer 192.0.2.3:0
              received:
                peer 192.0.2.1:0, label 16002, used-in-forwarding=Yes
                peer 192.0.2.2:0, label 17002, used-in-forwarding=No
            FEC 203.0.113.2/32:
               . . . .
            FEC 198.51.100.0/24:
               . . . .
            FEC 2001:db8:0:2::
               . . . .
            FEC 2001:db8:0:3::
               . . . .

        Address bindings:
            Addr 192.0.2.10:
               advertised
            Addr 2001:db8:0:10::
               advertised

            Addr 192.0.2.1:
              received, peer 192.0.2.1:0
            Addr 192.0.2.2:
              received, peer 192.0.2.2:0
            Addr 192.0.2.3:
              received, peer 192.0.2.3:0
            Addr 2001:db8:0:2::
              received, peer 192.0.2.2:0
            Addr 2001:db8:0:3::
              received, peer 192.0.2.3:0
```

Figure 7: Example Bindings

Note that all local addresses are advertised to all peers and hence no need to provide per-peer information for local address advertisement. Furthermore, note that it is easy to derive a peer-centric view for the bindings from the information already provided in this model.

Following captures high level tree hierarchy for bindings state. The tree shown below is for ipv4 address-family only; a similar tree exists for ipv6 address-family as well.

```
        +--rw mpls-ldp!
           +--rw global
              +--rw address-families
                 +--rw ipv4
                     +--ro bindings
                        +--ro address* [address]
                        |  +--ro address (ipv4-address or ipv6-address)
                        |  +--ro advertisement-type?   advertised-received
                        |  +--ro peer?        leafref
                        +--ro fec-label* [fec]
                           +--ro fec     (ipv4-prefix or ipv6-prefix)
                           +--ro peer* [peer advertisement-type]
                              +--ro peer                  leafref
                              +--ro advertisement-type? advertised-received
                              +--ro label?                mpls:mpls-label
                              +--ro used-in-forwarding?   boolean
```

Figure 8: Bindings state

6.4. Capabilities state

LDP capabilities state comprise two types of information - global information (such as timer etc.), and per-peer information.

Following captures high level tree hierarchy for LDP capabilities state.

```
      +--rw mpls-ldp!
         +--rw peers
            +--rw peer* [lsr-id label-space-id]
               +--rw lsr-id    yang:dotted-quad
               +--rw label-space-id
               +--ro received-peer-state
                  +--ro capability
                     +--ro . . . .
                     +--ro . . . .
```

Figure 9: Capabilities state

7. Notifications

This model defines a list of notifications to inform client of important events detected during the protocol operation. These events include events related to changes in the operational state of an LDP peer, hello adjacency, and FEC etc. It is to be noted that an LDP FEC is treated as operational (up) as long as it has at least 1 NHLFE (Next Hop Label Forwarding Entry) with outgoing label.

A simplified graphical representation of the data model for LDP notifications is shown in Figure 2.

8. Action

This model defines a list of rpcs that allow performing an action or executing a command on the protocol. For example, it allows to clear (reset) LDP peers, hello-adjacencies, and statistics. The model makes an effort to provide different level of control so that a user is able to either clear all, or clear all for a given type, or clear a specific entity.

A simplified graphical representation of the data model for LDP actions is shown in Figure 2.

9. YANG Specification

Following sections specify the actual YANG (module) specification for LDP constructs defined earlier in the document.

9.1. Base

This YANG module imports types defined in [RFC6991], [RFC8349], [RFC8294], [RFC8343], and [RFC8344].

```
<CODE BEGINS> file "ietf-mpls-ldp@2020-02-25.yang"

// RFC Editor: replace the above date 2020-02-25 with the date of
// publication and remove this note.

module ietf-mpls-ldp {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-ldp";
  prefix "ldp";

  import ietf-inet-types {
```

```
      prefix "inet";
      reference "RFC 6991: Common YANG Data Types";
    }

    import ietf-yang-types {
      prefix "yang";
      reference "RFC 6991: Common YANG Data Types";
    }

    import ietf-routing {
      prefix "rt";
      reference
        "RFC 8349: A YANG Data Model for Routing Management (NMDA
         version)";
    }

    import ietf-routing-types {
      prefix "rt-types";
      reference
        "RFC 8294: Common YANG Data Types for the Routing Area";
    }

    import ietf-interfaces {
      prefix "if";
      reference "RFC 8343: A YANG Data Model for Interface Management";
    }

    import ietf-ip {
      prefix "ip";
      reference "RFC 7277: A YANG Data Model for IP Management";
    }

    import ietf-key-chain {
      prefix "key-chain";
      reference "RFC 8177: YANG Data Model for Key Chains";
    }

    organization
      "IETF MPLS Working Group";
    contact
      "WG Web:   <http://tools.ietf.org/wg/mpls/>
       WG List:  <mailto:mpls@ietf.org>

       Editor:   Kamran Raza
                 <mailto:skraza@cisco.com>

       Editor:   Rajiv Asati
                 <mailto:rajiva@cisco.com>
```

```
        Editor:   Xufeng Liu
                  <mailto:xufeng.liu.ietf@gmail.com>

        Editor:   Santosh Esale
                  <mailto:sesale@juniper.net>

        Editor:   Xia Chen
                  <mailto:jescia.chenxia@huawei.com>

        Editor:   Himanshu Shah
                  <mailto:hshah@ciena.com>";

    description
      "This YANG module defines the essential components for the
       management of Multi-Protocol Label Switching (MPLS) Label
       Distribution Protocol (LDP). It is also the base model to
       be augmented for Multipoint LDP (mLDP).

       Copyright (c) 2020 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Simplified BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (http://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX; see the
       RFC itself for full legal notices.";

       // RFC Editor: replace XXXX with actual RFC number and remove
       // this note

    revision 2020-02-25 {
      description
        "Initial revision.";
      reference
        "RFC XXXX: YANG Data Model for MPLS LDP.";
        // RFC Editor: replace XXXX with actual RFC number and remove
        // this note
    }

    /*
     * Typedefs
     */
    typedef advertised-received {
      type enumeration {
```

```
        enum advertised {
          description "Advertised information.";
        }
        enum received {
          description "Received information.";
        }
      }
      description
        "Received or advertised.";
    }

    typedef downstream-upstream {
      type enumeration {
        enum downstream {
          description "Downstream information.";
        }
        enum upstream {
          description "Upstream information.";
        }
      }
      description
        "Downstream or upstream.";
    }

    typedef label-adv-mode {
      type enumeration {
        enum downstream-unsolicited {
          description "Downstream Unsolicited.";
        }
        enum downstream-on-demand {
          description "Downstream on Demand.";
        }
      }
      description
        "Label Advertisement Mode.";
    }

    typedef oper-status-event-type {
      type enumeration {
        enum up {
          value 1;
          description
            "Operational status changed to up.";
        }
        enum down {
          value 2;
          description
            "Operational status changed to down.";
```

```
        }
      }
      description "Operational status event type for notifications.";
    }

    /*
     * Identities
     */
    identity mpls-ldp {
      base rt:control-plane-protocol;
      description
        "LDP protocol.";
      reference
        "RFC 5036: LDP Specification";
    }

    identity adjacency-flag-base {
      description "Base type for adjacency flags.";
    }

    identity adjacency-flag-active {
      base adjacency-flag-base;
      description
        "This adjacency is configured and actively created.";
    }

    identity adjacency-flag-passive {
      base adjacency-flag-base;
      description
        "This adjacency is not configured and passively accepted.";
    }

    /*
     * Groupings
     */

    grouping adjacency-state-attributes {
      description
        "The operational state attributes of an LDP Hello adjacency,
         which can used for basic and extended discoveris, in IPv4 and
         IPv6 address families.";

      leaf-list flag {
        type identityref {
          base adjacency-flag-base;
        }
        description
          "On or more flags to indicate whether the adjacency is
```

```
           actively created, passively accepted, or both.";
        }
        container hello-holdtime {
          description
            "Containing Hello holdtime state information.";
          leaf adjacent {
            type uint16;
            units seconds;
            description
              "The holdtime value learned from the adjacent LSR.";
          }
          leaf negotiated {
            type uint16;
            units seconds;
            description
              "The holdtime negotiated between this LSR and the adjacent
               LSR.";
          }
          leaf remaining {
            type uint16;
            units seconds;
            description
              "The time remaining until the holdtime timer expires.";
          }
        }

        leaf next-hello {
          type uint16;
          units seconds;
          description
            "The time when the next Hello message will be sent.";
        }

        container statistics {
          description
            "Statistics objects.";

          leaf discontinuity-time {
            type yang:date-and-time;
            mandatory true;
            description
              "The time on the most recent occasion at which any one or
               more of this interface's counters suffered a
               discontinuity.  If no such discontinuities have occurred
               since the last re-initialization of the local management
               subsystem, then this node contains the time the local
               management subsystem re-initialized itself.";
          }
```

```
        leaf hello-received {
          type yang:counter64;
          description
            "The number of Hello messages received.";
        }
        leaf hello-dropped {
          type yang:counter64;
          description
            "The number of Hello messages dropped.";
        }
      } // statistics
    } // adjacency-state-attributes

    grouping basic-discovery-timers {
      description
        "The timer attributes for basic discovery, used in the
         per-interface setting and in the all-interface setting.";

      leaf hello-holdtime {
        type uint16 {
          range 15..3600;
        }
        units seconds;
        description
          "The time interval for which a LDP link Hello adjacency
           is maintained in the absence of link Hello messages from
           the LDP neighbor.
           This leaf may be configured at the per-interface level or
           the global level, with precedence given to the value at the
           per-interface level.  If the leaf is not configured at
           either level, the default value at the global level is
           used.";
      }
      leaf hello-interval {
        type uint16 {
          range 5..1200;
        }
        units seconds;
        description
          "The interval between consecutive LDP link Hello messages
           used in basic LDP discovery.
           This leaf may be configured at the per-interface level or
           the global level, with precedence given to the value at the
           per-interface level.  If the leaf is not configured at
           either level, the default value at the global level is
           used.";
      }
    } // basic-discovery-timers
```

```
    grouping binding-address-state-attributes {
      description
        "Operational state attributes of an address binding, used in
         IPv4 and IPv6 address families.";

      leaf advertisement-type {
        type advertised-received;
        description
          "Received or advertised.";
      }
      container peer {
        when "../advertisement-type = 'received'" {
          description
            "Applicable for received address.";
        }
        description
          "LDP peer from which this address is received.";
        uses ldp-peer-ref-from-binding;
      }
    } // binding-address-state-attributes

    grouping binding-label-state-attributes {
      description
        "Operational state attributes for a FEC-label binding, used in
         IPv4 and IPv6 address families.";

      list peer {
        key "lsr-id label-space-id advertisement-type";
        description
          "List of advertised and received peers.";
        uses ldp-peer-ref-from-binding {
          description
            "The LDP peer from which this binding is received, or to
             which this binding is advertised.
             The peer is identified by its LDP ID, which consists of
             the LSR ID and the Label Space ID.";
        }
        leaf advertisement-type {
          type advertised-received;
          description
            "Received or advertised.";
        }
        leaf label {
          type rt-types:mpls-label;
          description
            "Advertised (outbound) or received (inbound)
             label.";
        }
```

```
        leaf used-in-forwarding {
          type boolean;
          description
            "'true' if the label is used in forwarding.";
        }
      } // peer
    } // binding-label-state-attributes

    grouping graceful-restart-attributes-per-peer {
      description
        "Per peer graceful restart attributes.
         On the local side, these attributes are configuration and
         operational state data. One the peer side, these attributes
         are operational state data received from the peer.";

      container graceful-restart {
        description
          "Attributes for graceful restart.";
        leaf enabled {
          type boolean;
          description
            "Enable or disable graceful restart.
             This leaf may be configured at the per-peer level or the
             global level, with precedence given to the value at the
             per-peer level.  If the leaf is not configured at either
             level, the default value at the global level is used.";
        }
        leaf reconnect-time {
          type uint16 {
            range 10..1800;
          }
          units seconds;
          description
            "Specifies the time interval that the remote LDP peer
             must wait for the local LDP peer to reconnect after the
             remote peer detects the LDP communication failure.
             This leaf may be configured at the per-peer level or the
             global level, with precedence given to the value at the
             per-peer level.  If the leaf is not configured at either
             level, the default value at the global level is used.";
        }
        leaf recovery-time {
          type uint16 {
            range 30..3600;
          }
          units seconds;
          description
            "Specifies the time interval, in seconds, that the remote
```

```
            LDP peer preserves its MPLS forwarding state after
            receiving the Initialization message from the restarted
            local LDP peer.
            This leaf may be configured at the per-peer level or the
            global level, with precedence given to the value at the
            per-peer level.  If the leaf is not configured at either
            level, the default value at the global level is used.";
        }
      } // graceful-restart
    } // graceful-restart-attributes-per-peer

    grouping ldp-interface-ref {
      description
        "Defining a reference to LDP interface.";

      leaf name {
        type if:interface-ref;
        must "(/if:interfaces/if:interface[if:name=current()]/ip:ipv4)"
          + " or "
          + "(/if:interfaces/if:interface[if:name=current()]/ip:ipv6)"
        {
          description "Interface is IPv4 or IPv6.";
        }
        description
          "The name of an LDP interface.";
      }
    }

    grouping ldp-peer-ref-absolute {
      description
        "An absolute reference to an LDP peer, by the LDP ID, which
         consists of the LSR ID and the Label Space ID.";

      leaf protocol-name {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
            + "rt:control-plane-protocol/rt:name";
        }
        description
          "The name of the LDP protocol instance.";
      }
      leaf lsr-id {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
            + "rt:control-plane-protocol"
            + "[rt:name=current()/../protocol-name]/"
            + "ldp:mpls-ldp/ldp:peers/ldp:peer/ldp:lsr-id";
        }
```

```
        description
          "The LSR ID of the peer, as a portion of the peer LDP ID.";
      }
      leaf label-space-id {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
            + "rt:control-plane-protocol"
            + "[rt:name=current()/../protocol-name]/"
            + "ldp:mpls-ldp/ldp:peers/"
            + "ldp:peer[ldp:lsr-id=current()/../lsr-id]/"
            + "ldp:label-space-id";
        }
        description
          "The Label Space ID of the peer, as a portion of the peer
           LDP ID.";
      }
    } // ldp-peer-ref-absolute

    grouping ldp-peer-ref-from-binding {
      description
        "A relative reference to an LDP peer, by the LDP ID, which
         consists of the LSR ID and the Label Space ID.";

      leaf lsr-id {
        type leafref {
          path "../../../../../../../ldp:peers/ldp:peer/ldp:lsr-id";
        }
        description
          "The LSR ID of the peer, as a portion of the peer LDP ID.";
      }
      leaf label-space-id {
        type leafref {
          path "../../../../../../../ldp:peers/"
            + "ldp:peer[ldp:lsr-id=current()/../lsr-id]/"
            + "ldp:label-space-id";
        }
        description
          "The Label Space ID of the peer, as a portion of the peer
             LDP ID.";
      }
    } // ldp-peer-ref-from-binding

    grouping ldp-peer-ref-from-interface {
      description
        "A relative reference to an LDP peer, by the LDP ID, which
         consists of the LSR ID and the Label Space ID.";

      container peer {
```

```
        description
          "Reference to an LDP peer, by the LDP ID, which consists of
           the LSR ID and the Label Space ID.";
        leaf lsr-id {
          type leafref {
            path "../../../../../../../../../ldp:peers/ldp:peer/"
              + "ldp:lsr-id";
          }
          description
            "The LSR ID of the peer, as a portion of the peer LDP ID.";
        }
        leaf label-space-id {
          type leafref {
            path "../../../../../../../../../ldp:peers/"
              + "ldp:peer[ldp:lsr-id=current()/../lsr-id]/"
              + "ldp:label-space-id";
          }
          description
            "The Label Space ID of the peer, as a portion of the peer
             LDP ID.";
        }
      } // peer
    } // ldp-peer-ref-from-interface

    grouping ldp-peer-ref-from-target {
      description
        "A relative reference to an LDP peer, by the LDP ID, which
         consists of the LSR ID and the Label Space ID.";

      container peer {
        description
          "Reference to an LDP peer, by the LDP ID, which consists of
           the LSR ID and the Label Space ID.";
        leaf lsr-id {
          type leafref {
            path "../../../../../../../../../ldp:peers/ldp:peer/"
              + "ldp:lsr-id";
          }
          description
            "The LSR ID of the peer, as a portion of the peer LDP ID.";
        }
        leaf label-space-id {
          type leafref {
            path "../../../../../../../../../ldp:peers/"
              + "ldp:peer[ldp:lsr-id=current()/../lsr-id]/"
              + "ldp:label-space-id";
          }
          description
```

```
            "The Label Space ID of the peer, as a portion of the peer
             LDP ID.";
        }
      } // peer
    } // ldp-peer-ref-from-target

    grouping peer-attributes {
      description
        "Peer configuration attributes, used in the per-peer setting
         can in the all-peer setting.";

      leaf session-ka-holdtime {
        type uint16 {
          range 45..3600;
        }
        units seconds;
        description
          "The time interval after which an inactive LDP session
           terminates and the corresponding TCP session closes.
           Inactivity is defined as not receiving LDP packets from the
           peer.
           This leaf may be configured at the per-peer level or the
           global level, with precedence given to the value at the
           per-peer level.  If the leaf is not configured at either
           level, the default value at the global level is used.";
      }
      leaf session-ka-interval {
        type uint16 {
          range 15..1200;
        }
        units seconds;
        description
          "The interval between successive transmissions of keepalive
           packets. Keepalive packets are only sent in the absence of
           other LDP packets transmitted over the LDP session.
           This leaf may be configured at the per-peer level or the
           global level, with precedence given to the value at the
           per-peer level.  If the leaf is not configured at either
           level, the default value at the global level is used.";
      }
    } // peer-attributes

    grouping peer-authentication {
      description
        "Peer authentication container, used in the per-peer setting
         can in the all-peer setting.";

      container authentication {
```

```
        description
          "Containing authentication information.";
        choice authentication-type {
          description
            "Choice of authentication.";
          case password {
            leaf key {
              type string;
              description
                "This leaf specifies the authentication key. The length
                 of the key may be dependent on the cryptographic
                 algorithm.";
            }
            leaf crypto-algorithm {
              type identityref {
                base key-chain:crypto-algorithm;
              }
              description
                "Cryptographic algorithm associated with key.";
             }
          }
        }
      }
    } // peer-authentication

    grouping peer-state-derived {
      description
        "The peer state information derived from the LDP protocol
         operations.";

      container label-advertisement-mode {
        config false;
        description "Label advertisement mode state.";
        leaf local {
          type label-adv-mode;
          description
            "Local Label Advertisement Mode.";
        }
        leaf peer {
          type label-adv-mode;
          description
            "Peer Label Advertisement Mode.";
        }
        leaf negotiated {
          type label-adv-mode;
          description
            "Negotiated Label Advertisement Mode.";
        }
```

```
      }
      leaf next-keep-alive {
        type uint16;
        units seconds;
        config false;
        description
          "Time duration from now until sending the next KeepAlive
           message.";
      }

      container received-peer-state {
        config false;
        description
          "Operational state information learned from the peer.";

        uses graceful-restart-attributes-per-peer;

        container capability {
          description "Peer capability information.";
          container end-of-lib {
            description
              "Peer's end-of-lib capability.";
            leaf enabled {
              type boolean;
              description
                "'true' if peer's end-of-lib capability is enabled.";
            }
          }
          container typed-wildcard-fec {
            description
              "Peer's typed-wildcard-fec capability.";
            leaf enabled {
              type boolean;
              description
                "'true' if peer's typed-wildcard-fec capability is
                  enabled.";
            }
          }
          container upstream-label-assignment {
            description
              "Peer's upstream label assignment capability.";
            leaf enabled {
              type boolean;
              description
                "'true' if peer's upstream label assignment is
                 enabled.";
            }
          }
```

```
        } // capability
      } // received-peer-state

      container session-holdtime {
        config false;
        description "Session holdtime state.";
        leaf peer {
          type uint16;
          units seconds;
          description "Peer holdtime.";
        }
        leaf negotiated {
          type uint16;
          units seconds;
          description "Negotiated holdtime.";
        }
        leaf remaining {
          type uint16;
          units seconds;
          description "Remaining holdtime.";
        }
      } // session-holdtime

      leaf session-state {
        type enumeration {
          enum non-existent {
            description "NON EXISTENT state. Transport disconnected.";
          }
          enum initialized {
            description "INITIALIZED state.";
          }
          enum openrec {
            description "OPENREC state.";
          }
          enum opensent {
            description "OPENSENT state.";
          }
          enum operational {
            description "OPERATIONAL state.";
          }
        }
        config false;
        description
          "Representing the operational status of the LDP session.";
        reference
          "RFC5036, Sec. 2.5.4.";
      }
```

```
      container tcp-connection {
        config false;
        description "TCP connection state.";
        leaf local-address {
          type inet:ip-address;
          description "Local address.";
        }
        leaf local-port {
          type inet:port-number;
          description "Local port number.";
        }
        leaf remote-address {
          type inet:ip-address;
          description "Remote address.";
        }
        leaf remote-port {
          type inet:port-number;
          description "Remote port number.";
        }
      } // tcp-connection

      leaf up-time {
        type rt-types:timeticks64;
        config false;
        description
          "The number of time ticks (hundredths of a second) since the
           the state of the session with the peer changed to
           OPERATIONAL.";
      }

      container statistics {
        config false;
        description
          "Statistics objects.";

        leaf discontinuity-time {
          type yang:date-and-time;
          mandatory true;
          description
            "The time on the most recent occasion at which any one or
             more of this interface's counters suffered a
             discontinuity.  If no such discontinuities have occurred
             since the last re-initialization of the local management
             subsystem, then this node contains the time the local
             management subsystem re-initialized itself.";
        }

        container received {
```

```
          description "Inbound statistics.";
          uses statistics-peer-received-sent;
        }
        container sent {
          description "Outbound statistics.";
          uses statistics-peer-received-sent;
        }

        leaf total-addresses {
          type uint32;
          description
            "The number of learned addresses.";
        }
        leaf total-labels {
          type uint32;
          description
            "The number of learned labels.";
        }
        leaf total-fec-label-bindings {
          type uint32;
          description
            "The number of learned label-address bindings.";
        }
      } // statistics
    } // peer-state-derived

    grouping statistics-peer-received-sent {
      description
        "Inbound and outbound statistic counters.";
      leaf total-octets {
        type yang:counter64;
        description
          "The total number of octets sent or received.";
      }
      leaf total-messages {
        type yang:counter64;
        description
          "The number of messages sent or received.";
      }
      leaf address {
        type yang:counter64;
        description
          "The number of address messages sent or received.";
      }
      leaf address-withdraw {
        type yang:counter64;
        description
          "The number of address-withdraw messages sent or received.";
```

```
      }
      leaf initialization {
        type yang:counter64;
        description
          "The number of initialization messages sent or received.";
      }
      leaf keepalive {
        type yang:counter64;
        description
          "The number of keepalive messages sent or received.";
      }
      leaf label-abort-request {
        type yang:counter64;
        description
          "The number of label-abort-request messages sent or
           received.";
      }
      leaf label-mapping {
        type yang:counter64;
        description
          "The number of label-mapping messages sent or received.";
      }
      leaf label-release {
        type yang:counter64;
        description
          "The number of label-release messages sent or received.";
      }
      leaf label-request {
        type yang:counter64;
        description
          "The number of label-request messages sent or received.";
      }
      leaf label-withdraw {
        type yang:counter64;
        description
          "The number of label-withdraw messages sent or received.";
      }
      leaf notification {
        type yang:counter64;
        description
          "The number of notification messages sent or received.";
      }
    } // statistics-peer-received-sent

    /*
     * Configuration data and operational state data nodes
     */
```

```
    augment "/rt:routing/rt:control-plane-protocols/"
          + "rt:control-plane-protocol" {
      when "derived-from-or-self(rt:type, 'ldp:mpls-ldp')" {
        description
          "This augmentation is only valid for a control-plane
           protocol instance of LDP (type 'mpls-ldp').";
      }
      description
        "LDP augmentation to routing control-plane protocol
         configuration and state.";

      container mpls-ldp {
        must "not (../../rt:control-plane-protocol"
          + "[derived-from-or-self(rt:type, 'ldp:mpls-ldp')]"
          + "[rt:name!=current()/../rt:name])"
        {
          description "Only one LDP instance is allowed.";
        }

        description
          "Containing configuration and operational data for the LDP
           protocol.";

        container global {
          description
            "Global attributes for LDP.";

          container capability {
            description
              "Containing the LDP capability data. The container is
               used for augmentations.";
            reference
              "RFC5036: Sec. 1.5.";
          }

          container graceful-restart {
            description
              "Attributes for graceful restart.";
            leaf enabled {
              type boolean;
              default false;
              description
                "Enable or disable graceful restart.";
            }
            leaf reconnect-time {
              type uint16 {
                range 10..1800;
              }
```

```
            units seconds;
            default 120;
            description
              "Specifies the time interval that the remote LDP peer
               must wait for the local LDP peer to reconnect after
               the remote peer detects the LDP communication
               failure.";
          }
          leaf recovery-time {
            type uint16 {
              range 30..3600;
            }
            units seconds;
            default 120;
            description
              "Specifies the time interval, in seconds, that the
               remote LDP peer preserves its MPLS forwarding state
               after receiving the Initialization message from the
               restarted local LDP peer.";
          }
          leaf forwarding-holdtime {
            type uint16 {
              range 30..3600;
            }
            units seconds;
            default 180;
            description
              "Specifies the time interval, in seconds, before the
               termination of the recovery phase.";
          }
        } // graceful-restart

        leaf lsr-id {
          type rt-types:router-id;
          description
            "Specify the value to act as the LDP LSR ID.
             If this attribute is not specified, LDP uses the router
             ID as determined by the system.";
        }

        container address-families {
          description
            "Per address family confgiration and operational state.
             The address family can be either IPv4 or IPv6.";
          container ipv4 {
            presence
              "Present if IPv4 is enabled, unless the 'enabled'
               leaf is set to 'false'";
```

```
            description
              "Containing data related to the IPv4 address family.";

            leaf enabled {
              type boolean;
              default true;
              description
              "'false' to disable the address family.";
            }

            leaf label-distribution-control-mode {
              type enumeration {
                enum independent {
                  description
                  "Independent label distribution control.";
                }
                enum ordered {
                  description
                  "Ordered label distribution control.";
                }
              }
              config false;
              description
                "Label distribution control mode.";
              reference
                "RFC5036: LDP Specification. Sec 2.6.";
            }

            // ipv4 bindings
            container bindings {
              config false;
              description
                "LDP address and label binding information.";
              list address {
                key "address";
                description
                  "List of address bindings learned by LDP.";
                leaf address {
                  type inet:ipv4-address;
                  description
                    "The IPv4 address learned from an Address
                     message received from or advertised to a peer.";
                }
                uses binding-address-state-attributes;
              }

              list fec-label {
                key "fec";
```

```
                description
                  "List of FEC-label bindings learned by LDP.";
                leaf fec {
                  type inet:ipv4-prefix;
                  description
                    "The prefix FEC value in the FEC-label binding,
                     learned in a Label Mapping message received from
                     or advertised to a peer.";
                }
                uses binding-label-state-attributes;
              }
            } // bindings
          } // ipv4
        } // address-families
      } // global

      container discovery {
        description
          "Neighbor discovery configuration and operational state.";

        container interfaces {
          description
            "A list of interfaces for LDP Basic Discovery.";
          reference
            "RFC5036: LDP Specification. Sec 2.4.1.";

          uses basic-discovery-timers {
            refine "hello-holdtime" {
              default 15;
            }
            refine "hello-interval" {
              default 5;
            }
          }

          list interface {
            key "name";
            description
              "List of LDP interfaces used for LDP Basic Discovery.";
            uses ldp-interface-ref;
            leaf next-hello {
              type uint16;
              units seconds;
              config false;
              description "Time to send the next Hello message.";
            }

            container address-families {
```

```
                description
                  "Container for address families.";
                container ipv4 {
                  presence
                    "Present if IPv4 is enabled, unless the 'enabled'
                     leaf is set to 'false'";
                  description
                    "IPv4 address family.";

                  leaf enabled {
                    type boolean;
                    default true;
                    description
                    "Set to false to disable the address family on
                     the interface.";
                  }

                  container hello-adjacencies {
                    config false;
                    description
                      "Containing a list of Hello adjacencies.";

                    list hello-adjacency {
                      key "adjacent-address";
                      config false;
                      description "List of Hello adjacencies.";

                      leaf adjacent-address {
                        type inet:ipv4-address;
                        description
                          "Neighbor address of the Hello adjacency.";
                      }

                      uses adjacency-state-attributes;
                      uses ldp-peer-ref-from-interface;
                    }
                  }
                } // ipv4
              } // address-families
            } // interface
          } // interfaces

          container targeted
          {
            description
              "A list of targeted neighbors for extended discovery.";

            leaf hello-holdtime {
```

```
            type uint16 {
              range 15..3600;
            }
            units seconds;
            default 45;
            description
              "The time interval for which LDP targeted Hello
               adjacency is maintained in the absence of targeted
               Hello messages from an LDP neighbor.";
          }
          leaf hello-interval {
            type uint16 {
              range 5..3600;
            }
            units seconds;
            default 15;
            description
              "The interval between consecutive LDP targeted Hello
               messages used in extended LDP discovery.";
          }

          container hello-accept {
            description
              "LDP policy to control the acceptance of extended
               neighbor discovery Hello messages.";

            leaf enabled {
              type boolean;
              default false;
              description
                "'true' to accept; 'false' to deny.";
            }
          }

          container address-families {
            description
              "Container for address families.";
            container ipv4 {
              presence
                "Present if IPv4 is enabled.";
              description
                "IPv4 address family.";

              container hello-adjacencies {
                config false;
                description
                  "Containing a list of Hello adjacencies.";
```

```
      list hello-adjacency {
        key "local-address adjacent-address";
        description "List of Hello adjacencies.";

        leaf local-address {
          type inet:ipv4-address;
          description
            "Local address of the Hello adjacency.";
        }
        leaf adjacent-address {
          type inet:ipv4-address;
          description
            "Neighbor address of the Hello adjacency.";
        }

        uses adjacency-state-attributes;
        uses ldp-peer-ref-from-target;
      }
    }

    list target {
      key "adjacent-address";
      description
        "Targeted discovery params.";

      leaf adjacent-address {
        type inet:ipv4-address;
        description
          "Configures a remote LDP neighbor for the
           extended LDP discovery.";
      }

      leaf enabled {
        type boolean;
        default true;
        description
          "'true' to enable the target.";
      }
      leaf local-address {
        type inet:ipv4-address;
        description
          "The local address used as the source address to
           send targeted Hello messages.
           If the value is not specified, the
           transport-address is used as the source
           address.";
      }
    } // target
```

```
            } // ipv4
          } // address-families
        } // targeted
      } // discovery

      container peers {
        description
          "Peers configuration attributes.";

        uses peer-authentication;
        uses peer-attributes {
          refine session-ka-holdtime {
            default 180;
          }
          refine session-ka-interval {
            default 60;
          }
        }

        list peer {
          key "lsr-id label-space-id";
          description
            "List of peers.";

          leaf lsr-id {
            type rt-types:router-id;
            description
              "The LSR ID of the peer, to identify the globally
               unique LSR. This is the first four octets of the LDP
               ID. This leaf is used together with the leaf
               'label-space-id' to form the LDP ID.";
            reference
              "RFC5036. Sec 2.2.2.";
          }
          leaf label-space-id {
            type uint16;
            description
              "The Label Space ID of the peer, to identify a specific
               label space within the LSR. This is the last two
               octets of the LDP ID. This leaf is used together with
               the leaf 'lsr-id' to form the LDP ID.";
            reference
              "RFC5036. Sec 2.2.2.";
          }

          uses peer-authentication;

          container address-families {
```

```
            description
              "Per-vrf per-af params.";
            container ipv4 {
              presence
                "Present if IPv4 is enabled.";
              description
                "IPv4 address family.";

              container hello-adjacencies {
                config false;
                description
                  "Containing a list of Hello adjacencies.";

                list hello-adjacency {
                  key "local-address adjacent-address";
                  description "List of Hello adjacencies.";

                  leaf local-address {
                    type inet:ipv4-address;
                    description
                      "Local address of the Hello adjacency.";
                  }
                  leaf adjacent-address {
                    type inet:ipv4-address;
                    description
                      "Neighbor address of the Hello adjacency.";
                  }

                  uses adjacency-state-attributes;

                  leaf interface {
                    type if:interface-ref;
                    description "Interface for this adjacency.";
                  }
                }
              }
            } // ipv4
          } // address-families

          uses peer-state-derived;
        } // list peer
      } // peers
    } // container mpls-ldp
  }

  /*
   * RPCs
   */
```

```
    rpc mpls-ldp-clear-peer {
      description
        "Clears the session to the peer.";
      input {
        uses ldp-peer-ref-absolute {
          description
            "The LDP peer to be cleared. If this is not provided
             then all peers are cleared.
             The peer is identified by its LDP ID, which consists of
             the LSR ID and the Label Space ID.";
        }
      }
    }

    rpc mpls-ldp-clear-hello-adjacency {
      description
        "Clears the hello adjacency";
      input {
        container hello-adjacency {
          description
            "Link adjacency or targettted adjacency. If this is not
             provided then all Hello adjacencies are cleared";
          leaf protocol-name {
            type leafref {
              path "/rt:routing/rt:control-plane-protocols/"
                + "rt:control-plane-protocol/rt:name";
            }
            description
              "The name of the LDP protocol instance.";
          }
          choice hello-adjacency-type {
            description "Adjacency type.";
            case targeted {
              container targeted {
                presence "Present to clear targeted adjacencies.";
                description
                  "Clear targeted adjacencies.";
                leaf target-address {
                  type inet:ip-address;
                  description
                    "The target address. If this is not provided then
                     all targeted adjacencies are cleared";
                }
              }
            }
            case link {
              container link {
                presence "Present to clear link adjacencies.";
```

```
                description
                  "Clear link adjacencies.";
                leaf next-hop-interface {
                  type leafref {
                    path "/rt:routing/rt:control-plane-protocols/"
                      + "rt:control-plane-protocol/mpls-ldp/discovery/"
                      + "interfaces/interface/name";
                  }
                  description
                    "Interface connecting to next-hop. If this is not
                     provided then all link adjacencies are cleared.";
                }
                leaf next-hop-address {
                  type inet:ip-address;
                  must "../next-hop-interface" {
                    description
                      "Applicable when interface is specified.";
                  }
                  description
                    "IP address of next-hop. If this is not provided
                     then adjacencies to all next-hops on the given
                     interface are cleared.";
                }
              }
            }
          } // hello-adjacency-type
        } // hello-adjacency
      } // input
    } // mpls-ldp-clear-hello-adjacency

    rpc mpls-ldp-clear-peer-statistics {
      description
        "Clears protocol statistics (e.g. sent and received
         counters).";
      input {
        uses ldp-peer-ref-absolute {
          description
            "The LDP peer whose statistics are to be cleared.
             If this is not provided then all peers' statistics are
             cleared.
             The peer is identified by its LDP ID, which consists of
             the LSR ID and the Label Space ID.";
        }
      }
    }

    /*
     * Notifications
```

```
     */
    notification mpls-ldp-peer-event {

      description
        "Notification event for a change of LDP peer operational
         status.";
      leaf event-type {
        type oper-status-event-type;
        description "Event type.";
      }
      container peer {
        description
          "Reference to an LDP peer, by the LDP ID, which consists of
           the LSR ID and the Label Space ID.";
        uses ldp-peer-ref-absolute;
      }
    }

    notification mpls-ldp-hello-adjacency-event {
      description
        "Notification event for a change of LDP adjacency operational
         status.";
      leaf event-type {
        type oper-status-event-type;
        description "Event type.";
      }
      leaf protocol-name {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
            + "rt:control-plane-protocol/rt:name";
        }
        description
          "The name of the LDP protocol instance.";
      }
      choice hello-adjacency-type {
        description
          "Interface or targeted adjacency.";
        case targeted {
          container targeted {
            description
              "Targeted adjacency through LDP extended discovery.";
            leaf target-address {
              type inet:ip-address;
              description
                "The target adjacent address learned.";
            }
          }
        }
```

```
        case link {
          container link {
            description
              "Link adjacency through LDP basic discovery.";
            leaf next-hop-interface {
              type if:interface-ref;
              description
                "The interface connecting to the adjacent next hop.";
            }
            leaf next-hop-address {
              type inet:ip-address;
              must "../next-hop-interface" {
                description
                  "Applicable when interface is specified.";

              }
              description
                "IP address of the next hop. This can be IPv4 or IPv6
                 address.";
            }
          }
        }
      } // hello-adjacency-type
    } // mpls-ldp-hello-adjacency-event

    notification mpls-ldp-fec-event {
      description
        "Notification event for a change of FEC status.";
      leaf event-type {
        type oper-status-event-type;
        description "Event type.";
      }
      leaf protocol-name {
        type leafref {
          path "/rt:routing/rt:control-plane-protocols/"
            + "rt:control-plane-protocol/rt:name";
        }
        description
          "The name of the LDP protocol instance.";
      }
      leaf fec {
        type inet:ip-prefix;
        description
          "The address prefix element of the FEC whose status
           has changed.";
      }
    }
  }
```

<CODE ENDS>

Figure 10: LDP base module

9.2. Extended

This YANG module imports types defined in [RFC6991], [RFC8349], [RFC8177], and [RFC8343].

<CODE BEGINS> file "ietf-mpls-ldp-extended@2020-02-25.yang"

```
// RFC Editor: replace the above date 2020-02-25  with the date of
// publication and remove this note.

module ietf-mpls-ldp-extended {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-ldp-extended";
  prefix "ldp-ext";

  import ietf-inet-types {
    prefix "inet";
    reference "RFC 6991: Common YANG Data Types";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
       version)";
  }

  import ietf-key-chain {
    prefix "key-chain";
    reference "RFC 8177: YANG Data Model for Key Chains";
  }

  import ietf-mpls-ldp {
    prefix "ldp";
    reference "RFC XXXX: YANG Data Model for MPLS LDP";
    // RFC Editor: replace XXXX with actual RFC number and remove
    // this note
  }
```

```
  import ietf-interfaces {
    prefix "if";
    reference "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-routing-policy {
    prefix rt-pol;
    reference
      "I-D.ietf-rtgwg-policy-model: A YANG Data Model for Routing
       Policy Management";
  }

  organization
    "IETF MPLS Working Group";
  contact
    "WG Web:   <http://tools.ietf.org/wg/mpls/>
     WG List:  <mailto:mpls@ietf.org>

     Editor:   Kamran Raza
               <mailto:skraza@cisco.com>

     Editor:   Rajiv Asati
               <mailto:rajiva@cisco.com>

     Editor:   Xufeng Liu
               <mailto:xufeng.liu.ietf@gmail.com>

     Editor:   Santosh Esale
               <mailto:sesale@juniper.net>

     Editor:   Xia Chen
               <mailto:jescia.chenxia@huawei.com>

     Editor:   Himanshu Shah
               <mailto:hshah@ciena.com>";

  description
    "This YANG module defines the extended components for the
     management of Multi-Protocol Label Switching (MPLS) Label
     Distribution Protocol (LDP). It is also the model to
     be augmented for extended Multipoint LDP (mLDP).

     Copyright (c) 2020 IETF Trust and the persons identified as
     authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with or
     without modification, is permitted pursuant to, and subject to
     the license terms contained in, the Simplified BSD License set
```

```
        forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (http://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX; see the
        RFC itself for full legal notices.";

        // RFC Editor: replace XXXX with actual RFC number and remove
        // this note

      revision 2020-02-25 {
        description
          "Initial revision.";
        reference
          "RFC XXXX: YANG Data Model for MPLS LDP.";

          // RFC Editor: replace XXXX with actual RFC number and remove
          // this note
      }

      /*
       * Features
       */
      feature capability-end-of-lib {
        description
          "This feature indicates that the system allows to configure
           LDP end-of-lib capability.";
      }

      feature capability-typed-wildcard-fec {
        description
          "This feature indicates that the system allows to configure
           LDP typed-wildcard-fec capability.";
      }

      feature capability-upstream-label-assignment {
        description
          "This feature indicates that the system allows to configure
           LDP upstream label assignment capability.";
      }

      feature forwarding-nexthop-config {
        description
          "This feature indicates that the system allows to configure
           forwarding nexthop on interfaces.";
      }

      feature graceful-restart-helper-mode {
```

```
      description
        "This feature indicates that the system supports graceful
         restart helper mode. We call an LSR to be operating in GR
         helper mode when it advertises 0 as its FT Reconnect Timeout
         in the FT Session TLV.
         Please refer RFC3478 section 2 for details.";
    }

    feature key-chain {
      description
        "This feature indicates that the system supports keychain for
         authentication.";
    }

    feature peers-dual-stack-transport-preference {
      description
        "This feature indicates that the system allows to configure
         the transport connection preference in a dual-stack setup
         for peers.";
    }

    feature per-interface-timer-config {
      description
        "This feature indicates that the system allows to configure
         interface Hello timers at the per-interface level.";
    }

    feature per-peer-admin-down {
      description
        "This feature indicates that the system allows to
         administratively disable a peer.";
    }

    feature per-peer-graceful-restart-config {
      description
        "This feature indicates that the system allows to configure
         graceful restart at the per-peer level.";
    }

    feature per-peer-session-attributes-config {
      description
        "This feature indicates that the system allows to configure
         session attributes at the per-peer level.";
    }

    feature policy-label-assignment-config {
      description
        "This feature indicates that the system allows to configure
```

```
          policies to assign labels according to certain prefixes.";
      }

      feature policy-ordered-label-config {
        description
          "This feature indicates that the system allows to configure
           ordered label policies.";
      }

      feature policy-targeted-discovery-config {
        description
          "This feature indicates that the system allows to configure
           policies to control the acceptance of targeted neighbor
           discovery Hello messages.";
      }

      feature session-downstream-on-demand-config {
        description
          "This feature indicates that the system allows to configure
           session downstream-on-demand";
      }

      /*
       * Typedefs
       */
      typedef neighbor-list-ref {
        type leafref {
          path "/rt-pol:routing-policy/rt-pol:defined-sets/"
            + "rt-pol:neighbor-sets/rt-pol:neighbor-set/rt-pol:name";
        }
        description
          "A type for a reference to a neighbor address list.
           The string value is the name identifier for uniquely
           identifying the referenced address list, which contains a list
           of addresses that a routing policy can applied.";
        reference
          "I-D.ietf-rtgwg-policy-model: A YANG Data Model for Routing
           Policy Management";
      }

      typedef prefix-list-ref {
        type leafref {
          path "/rt-pol:routing-policy/rt-pol:defined-sets/"
            + "rt-pol:prefix-sets/rt-pol:prefix-set/rt-pol:name";
        }
        description
          "A type for a reference to a prefix list.
           The string value is the name identifier for uniquely
```

```
          identifying the referenced prefix set, which contains a list
          of prefixes that a routing policy can applied.";
        reference
          "I-D.ietf-rtgwg-policy-model: A YANG Data Model for Routing
          Policy Management";
      }

      typedef peer-list-ref {
        type leafref {
          path "/rt-pol:routing-policy/rt-pol:defined-sets/"
            + "rt-pol:neighbor-sets/rt-pol:neighbor-set/rt-pol:name";
        }
        description
          "A type for a reference to a peer address list.
          The string value is the name identifier for uniquely
          identifying the referenced address list, which contains a list
          of addresses that a routing policy can applied.";
        reference
          "I-D.ietf-rtgwg-policy-model: A YANG Data Model for Routing
          Policy Management";
      }

      /*
       * Identities
       */

      /*
       * Groupings
       */
      grouping address-family-ipv4-augment {
        description "Augmentation to address family IPv4.";

        uses policy-container;

        leaf transport-address {
          type inet:ipv4-address;
          description
            "The transport address advertised in LDP Hello messages.
            If this value is not specified, the LDP LSR ID is used as
            the transport address.";
          reference
            "RFC5036. Sec. 3.5.2.";
        }
      }

      grouping authentication-keychain-augment {
        description "Augmentation to authentication to add keychain.";
```

```
      leaf key-chain {
        type key-chain:key-chain-ref;
        description
          "key-chain name.
           If not specified, no key chain is used.";
      }
    }

    grouping capability-augment {
      description "Augmentation to capability.";

      container end-of-lib {
        if-feature capability-end-of-lib;
        description
          "Configure end-of-lib capability.";
        leaf enabled {
          type boolean;
          default false;
          description
            "'true' to enable end-of-lib capability.";
        }
      }
      container typed-wildcard-fec {
        if-feature capability-typed-wildcard-fec;
        description
          "Configure typed-wildcard-fec capability.";
        leaf enabled {
          type boolean;
          default false;
          description
            "'true' to enable typed-wildcard-fec capability.";
        }
      }
      container upstream-label-assignment {
        if-feature capability-upstream-label-assignment;
        description
          "Configure upstream label assignment capability.";
        leaf enabled {
          type boolean;
          default false;
          description
            "'true' to enable upstream label assignment.";
        }
      }
    } // capability-augment

    grouping global-augment {
      description "Augmentation to global attributes.";
```

```
      leaf igp-synchronization-delay {
        type uint16 {
          range "0 | 3..300";
        }
        units seconds;
        default 0;
        description
          "Sets the interval that the LDP waits before notifying the
           Interior Gateway Protocol (IGP) that label exchange is
           completed so that IGP can start advertising the normal
           metric for the link.
           If the value is not specified, there is no delay.";
      }
    }

    grouping global-forwarding-nexthop-augment {
      description
        "Augmentation to global forwarding nexthop interfaces.";

      container forwarding-nexthop {
        if-feature forwarding-nexthop-config;
        description
          "Configuration for forwarding nexthop.";

        container interfaces {
          description
            "Containing a list of interfaces on which forwarding can be
             disabled.";

          list interface {
            key "name";
            description
              "List of LDP interfaces on which forwarding can be
               disabled.";
            uses ldp:ldp-interface-ref;
            list address-family {
              key "afi";
              description
                "Per-vrf per-af params.";
              leaf afi {
                type identityref {
                  base rt:address-family;
                }
                description
                  "Address family type value.";
              }
              leaf ldp-disable {
                type boolean;
```

```
                default false;
                description
                  "'true' to disable LDP forwarding on the interface.";
              }
            }
          } // interface
        } // interfaces
      } // forwarding-nexthop
    } // global-forwarding-nexthop-augment

    grouping graceful-restart-augment {
      description "Augmentation to graceful restart.";

      leaf helper-enabled {
        if-feature graceful-restart-helper-mode;
        type boolean;
        default false;
        description
          "Enable or disable graceful restart helper mode.";
      }
    }

    grouping interface-address-family-ipv4-augment {
      description "Augmentation to interface address family IPv4.";

      leaf transport-address {
        type union {
          type enumeration {
            enum "use-global-transport-address" {
              description
                "Use the transport address set at the global level
                 common for all interfaces for this address family.";
            }
            enum "use-interface-address" {
              description
                "Use interface address as the transport address.";
            }
          }
          type inet:ipv4-address;
        }
        default "use-global-transport-address";
        description
          "IP address to be advertised as the LDP transport address.";
      }
    }

    grouping interface-address-family-ipv6-augment {
      description "Augmentation to interface address family IPv6.";
```

```
        leaf transport-address {
          type union {
            type enumeration {
              enum "use-global-transport-address" {
                description
                  "Use the transport address set at the global level
                   common for all interfaces for this address family.";
              }
              enum "use-interface-address" {
                description
                  "Use interface address as the transport address.";
              }
            }
            type inet:ipv6-address;
          }
          default "use-global-transport-address";
          description
            "IP address to be advertised as the LDP transport address.";
        }
      }

      grouping interface-augment {
        description "Augmentation to interface.";

        uses ldp:basic-discovery-timers {
          if-feature per-interface-timer-config;
        }
        leaf igp-synchronization-delay {
          if-feature per-interface-timer-config;
          type uint16 {
            range "0 | 3..300";
          }
          units seconds;
          description
            "Sets the interval that the LDP waits before notifying the
             Interior Gateway Protocol (IGP) that label exchange is
             completed so that IGP can start advertising the normal
             metric for the link.
             This leaf may be configured at the per-interface level or
             the global level, with precedence given to the value at the
             per-interface level.  If the leaf is not configured at
             either level, the default value at the global level is
             used.";
        }
      }

      grouping peer-af-policy-container {
        description
```

```
        "LDP policy attribute container under peer address-family.";
      container label-policy {
        description
          "Label policy attributes.";
        container advertise {
          description
            "Label advertising policies.";
          leaf prefix-list {
            type prefix-list-ref;
              description
                "Applies the prefix list to filter outgoing label
                 advertisements.
                 If the value is not specified, no prefix filter
                 is applied.";
          }
        }
        container accept {
          description
            "Label advertisement acceptance policies.";
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Applies the prefix list to filer incoming label
               advertisements.
               If the value is not specified, no prefix filter
               is applied.";
          }
        }
      }
    } // peer-af-policy-container

    grouping peer-augment {
      description "Augmentation to each peer list entry.";

      leaf admin-down {
        if-feature per-peer-admin-down;
        type boolean;
        default false;
        description
          "'true' to disable the peer.";
      }

      uses ldp:graceful-restart-attributes-per-peer {
        if-feature per-peer-graceful-restart-config;
      }

      uses ldp:peer-attributes {
        if-feature per-peer-session-attributes-config;
```

```
      }
    }

    grouping peers-augment {
      description "Augmentation to peers container.";

      container session-downstream-on-demand {
        if-feature session-downstream-on-demand-config;
        description
          "Session downstream-on-demand attributes.";
        leaf enabled {
          type boolean;
          default false;
          description
            "'true' if session downstream-on-demand is enabled.";
        }
        leaf peer-list {
          type peer-list-ref;
          description
            "The name of a peer ACL, to be applied to the
             downstream-on-demand sessions.
             If this value is not specified, no filter is applied to
             any downstream-on-demand sessions.";
        }
      }
      container dual-stack-transport-preference {
        if-feature peers-dual-stack-transport-preference;
        description
          "The settings of peers to establish TCP connection in a
           dual-stack setup.";
        leaf max-wait {
            type uint16 {
              range "0..60";
            }
            default 30;
            description
              "The maximum wait time in seconds for preferred transport
               connection establishment. 0 indicates no preference.";
        }
        container prefer-ipv4 {
          presence
            "Present if IPv4 is prefered for transport connection
             establishment, subject to the 'peer-list' in this
             container.";
          description
            "Uses IPv4 as the prefered address family for transport
             connection establishment, subject to the 'peer-list' in
             this container.
```

```
            If this container is not present, as a default, IPv6 is
            the prefered address family for transport connection
            establishment.";
          leaf peer-list {
            type peer-list-ref;
            description
              "The name of a peer ACL, to be applied to the IPv4
               transport connections.
               If this value is not specified, no filter is applied,
               and the IPv4 is prefered for all peers.";
          }
        }
      }
    } // peers-augment

    grouping policy-container {
      description
        "LDP policy attributes.";
      container label-policy {
        description
          "Label policy attributes.";
        container advertise {
          description
            "Label advertising policies.";
          container egress-explicit-null {
            description
              "Enables an egress router to advertise an
               explicit null label (value 0) in place of an
               implicit null label (value 3) to the
               penultimate hop router.";
            leaf enabled {
              type boolean;
              default false;
              description
                "'true' to enable explicit null.";
            }
          }
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Applies the prefix list to filter outgoing label
               advertisements.
               If the value is not specified, no prefix filter
               is applied.";
          }
        }
        container accept {
          description
```

```
            "Label advertisement acceptance policies.";
          leaf prefix-list {
            type prefix-list-ref;
            description
              "Applies the prefix list to filter incoming label
               advertisements.
               If the value is not specified, no prefix filter
               is applied.";
          }
        }
        container assign {
          if-feature policy-label-assignment-config;
          description
            "Label assignment policies";
          container independent-mode {
            description
              "Independent label policy attributes.";
            leaf prefix-list {
              type prefix-list-ref;
              description
                "Assign labels according to certain prefixes.
                 If the value is not specified, no prefix filter
                 is applied (labels are assigned to all learned
                 routes).";

            }
          }
          container ordered-mode {
            if-feature policy-ordered-label-config;
            description
              "Ordered label policy attributes.";
            leaf egress-prefix-list {
              type prefix-list-ref;
              description
                "Assign labels according to certain prefixes for
                 egress LSR.";
            }
          }
        } // assign
      } // label-policy
    } // policy-container

    /*
     * Configuration and state data nodes
     */
    // Forwarding nexthop augmentation to the global tree
    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:global" {
```

```
      description "Forwarding nexthop augmentation.";
      uses global-forwarding-nexthop-augment;
    }

    // global/address-families/ipv6
    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:global/"
      + "ldp:address-families" {
      description "Global IPv6 augmentation.";

      container ipv6 {
        presence
          "Present if IPv6 is enabled, unless the 'enabled'
           leaf is set to 'false'";
        description
          "Containing data related to the IPv6 address family.";

        leaf enabled {
          type boolean;
          default true;
          description
            "'false' to disable the address family.";
        }

        uses policy-container;

        leaf transport-address {
          type inet:ipv6-address;
          mandatory true;
          description
            "The transport address advertised in LDP Hello messages.";
        }

        leaf label-distribution-control-mode {
          type enumeration {
            enum independent {
              description
              "Independent label distribution control.";
            }
            enum ordered {
              description
              "Ordered label distribution control.";
            }
          }
          config false;
          description
            "Label distribution control mode.";
          reference
```

```
            "RFC5036: LDP Specification. Sec 2.6.";
        }

        // ipv6 bindings
        container bindings {
          config false;
          description
            "LDP address and label binding information.";
          list address {
            key "address";
            description
              "List of address bindings learned by LDP.";
            leaf address {
              type inet:ipv6-address;
              description
                "The IPv6 address learned from an Address
                 message received from or advertised to a peer.";
            }
            uses ldp:binding-address-state-attributes;
          }

          list fec-label {
            key "fec";
            description
              "List of FEC-label bindings learned by LDP.";
            leaf fec {
              type inet:ipv6-prefix;
              description
                "The prefix FEC value in the FEC-label binding,
                 learned in a Label Mapping message received from
                 or advertised to a peer.";
            }
            uses ldp:binding-label-state-attributes;
          }
        } // bindings
      } // ipv6
    }

    // discovery/interfaces/interface/address-families/ipv6
    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:discovery/"
      + "ldp:interfaces/ldp:interface/"
      + "ldp:address-families" {
      description "Interface IPv6 augmentation.";

      container ipv6 {
        presence
          "Present if IPv6 is enabled, unless the 'enabled'
```

```
           leaf is set to 'false'";
        description
          "IPv6 address family.";

        leaf enabled {
          type boolean;
          default true;
          description
            "'false' to disable the address family on the interface.";
        }

        container hello-adjacencies {
          config false;
          description
            "Containing a list of Hello adjacencies.";

          list hello-adjacency {
            key "adjacent-address";
            config false;
            description "List of Hello adjacencies.";

            leaf adjacent-address {
              type inet:ipv6-address;
              description
                "Neighbor address of the Hello adjacency.";
            }

            uses ldp:adjacency-state-attributes;
            uses ldp:ldp-peer-ref-from-interface;
          }
        }
      } // ipv6
    }

    // discovery/targeted/address-families/ipv6
    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:discovery/"
      + "ldp:targeted/ldp:address-families" {
      description "Targeted discovery IPv6 augmentation.";

      container ipv6 {
        presence
          "Present if IPv6 is enabled.";
        description
          "IPv6 address family.";

        container hello-adjacencies {
          config false;
```

```
          description
            "Containing a list of Hello adjacencies.";

          list hello-adjacency {
            key "local-address adjacent-address";
            config false;
            description "List of Hello adjacencies.";

            leaf local-address {
              type inet:ipv6-address;
              description
                "Local address of the Hello adjacency.";
            }
            leaf adjacent-address {
              type inet:ipv6-address;
              description
                "Neighbor address of the Hello adjacency.";
            }

            uses ldp:adjacency-state-attributes;
            uses ldp:ldp-peer-ref-from-target;
          }
        }

        list target {
          key "adjacent-address";
          description
            "Targeted discovery params.";

          leaf adjacent-address {
            type inet:ipv6-address;
            description
              "Configures a remote LDP neighbor for the
               extended LDP discovery.";
          }
          leaf enabled {
            type boolean;
            default true;
            description
              "'true' to enable the target.";
          }
          leaf local-address {
            type inet:ipv6-address;
            description
              "The local address used as the source address to send
               targeted Hello messages.
               If the value is not specified, the transport-address
               is used as the source address.";
```

```
          }
        } // target
      } // ipv6
    }

    // /peers/peer/state/address-families/ipv6
    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:peers/"
      + "ldp:peer/ldp:address-families" {
      description "Peer state IPv6 augmentation.";

      container ipv6 {
        presence
          "Present if IPv6 is enabled.";
        description
          "IPv6 address family.";

        container hello-adjacencies {
          config false;
          description
            "Containing a list of Hello adjacencies.";

          list hello-adjacency {
            key "local-address adjacent-address";
            description "List of Hello adjacencies.";

            leaf local-address {
              type inet:ipv6-address;
              description
                "Local address of the Hello adjacency.";
            }
            leaf adjacent-address {
              type inet:ipv6-address;
              description
                "Neighbor address of the Hello adjacency.";
            }

            uses ldp:adjacency-state-attributes;

            leaf interface {
              type if:interface-ref;
              description "Interface for this adjacency.";
            }
          }
        }
      } // ipv6
    }
```

```
    /*
     * Configuration data and operational state data nodes
     */
    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:global" {
      description "Graceful restart augmentation.";
      uses global-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:global/"
      + "ldp:capability" {
      description "Capability augmentation.";
      uses capability-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:global/"
      + "ldp:graceful-restart" {
      description "Graceful restart augmentation.";
      uses graceful-restart-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:global/"
      + "ldp:address-families/ldp:ipv4" {
      description "Address family IPv4 augmentation.";
      uses address-family-ipv4-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:discovery/"
      + "ldp:interfaces/ldp:interface" {
      description "Interface augmentation.";
      uses interface-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:discovery/"
      + "ldp:interfaces/ldp:interface/ldp:address-families/"
      + "ldp:ipv4" {
      description "Interface address family IPv4 augmentation.";
      uses interface-address-family-ipv4-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:discovery/"
      + "ldp:interfaces/ldp:interface/ldp:address-families/"
```

```
      + "ldp-ext:ipv6" {
      description "Interface address family IPv6 augmentation.";
      uses interface-address-family-ipv6-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:discovery/"
      + "ldp:targeted/ldp:hello-accept" {
      description "Targeted discovery augmentation.";
      leaf neighbor-list {
        if-feature policy-targeted-discovery-config;
        type neighbor-list-ref;
        description
          "The name of a neighbor ACL, to accept Hello messages from
           LDP peers as permitted by the neighbor-list policy.
           If this value is not specified, targeted Hello messages from
           any source are accepted.";
      }
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:peers" {
      description "Peers augmentation.";
      uses peers-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:peers/"
      + "ldp:authentication/ldp:authentication-type" {
      if-feature key-chain;
      description "Peers authentication augmentation.";
      case key-chain {
        uses authentication-keychain-augment;
      }
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:peers/ldp:peer" {
      description "Peer list entry augmentation.";
      uses peer-augment;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:peers/ldp:peer/"
      + "ldp:authentication/ldp:authentication-type" {
      if-feature key-chain;
      description "Peer list entry authentication augmentation.";
      case key-chain {
```

```
        uses authentication-keychain-augment;
      }
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:peers/ldp:peer/"
      + "ldp:address-families/ldp:ipv4" {
      description
        "Peer list entry IPv4 augmentation.";
      uses peer-af-policy-container;
    }

    augment "/rt:routing/rt:control-plane-protocols/"
      + "rt:control-plane-protocol/ldp:mpls-ldp/ldp:peers/ldp:peer/"
      + "ldp:address-families/ldp-ext:ipv6" {
      description
        "Peer list entry IPv6 augmentation.";
      uses peer-af-policy-container;
    }
  }

  <CODE ENDS>
```

Figure 11: LDP extended module

10. Security Considerations

This specification inherits the security considerations captured in [RFC5920] and the LDP protocol specification documents, namely base LDP [RFC5036], LDP IPv6 [RFC7552], LDP Capabilities [RFC5561], Typed Wildcard FEC [RFC5918], LDP End-of-LIB [RFC5919], and LDP Upstream Label Assignment [RFC6389].

10.1. YANG model

The YANG modules specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or

RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

10.1.1. Writable nodes

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations.

For LDP, the ability to modify MPLS LDP configuration may allow the entire MPLS LDP domain to be compromised including forming LDP adjacencies and/or peer sessions with unauthorized routers to mount a massive Denial-of-Service (DoS) attack. In particular, folling are the subtrees and data nodes that are sensitivite and vulnerable:

* /mpls-ldp/discovery/interfaces/interface: Adding LDP on any unprotected interface could allow an LDP hello adjacency to be formed with an unauthorized and malicious neighbor. Once an hello adjacency is formed, a peer session could progress with this neighbor.

* /mpls-ldp/discovery/targeted/hello-accept: Allowing acceptance of targeted-hellos could open LDP to DoS attacks related to incoming targeted hellos from malicious sources.

* /mpls-ldp/peers/authentication: Allowing a peer session establishement is typically controlled via LDP authentication where a proper and secure authentication password/key management is warranted.

* /mpls-ldp/peers/peer/authentication: Same as above.

10.1.2. Readable nodes

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

The exposure of LDP databases (such as hello adjacencies, peers, address bindings, and fec-label bindings) beyond the scope of the LDP admin domain may be undesirable. The relevant subtrees and data nodes are as follows:

* /mpls-ldp/global/address-families/ipv4/bindings/address

* /mpls-ldp/global/address-families/ipv6/bindings/address

* /mpls-ldp/global/address-families/ipv4/bindings/fec-label

* /mpls-ldp/global/address-families/ipv6/bindings/fec-label

* /mpls-ldp/discovery/interfaces/interface/address-families/ipv4/hello-adjacencies

* /mpls-ldp/discovery/interfaces/interface/address-families/ipv6/hello-adjacencies

* /mpls-ldp/discovery/targeted/address-families/ipv4/hello-adjacencies

* /mpls-ldp/discovery/targeted/address-families/ipv6/hello-adjacencies

* /mpls-ldp/peers

The configuration for LDP peer authentication is supported via the specification of key-chain [RFC8040], or via direct specification of a key associated with a crypto algorithm (such as MD5). The relevant subtrees and data nodes are as follows:

* /mpls-ldp/peers/authentication

* /mpls-ldp/peers/peer/authentication

The actual authentication key data (whether locally specified or part of a key-chain) is sensitive and needs to be kept secret from unauthorized parties. For key-chain based authentication, this model inherits the security considerations of [RFC8040] (that includes the considerations with respect to the local storage and handling of authentication keys). A similar procedure for storage and access to direct key is warranted.

10.1.3. RPC operations

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations otherwise control plane flaps, network outages, and DoS attacks are possible. The RPC operations are:

* mpls-ldp-clear-peer

* mpls-ldp-clear-hello-adjacency

10.1.4. Notifications

The model describes several notifications. The implementations must rate-limit the generation of these notifications to avoid creating significant notification load and possible side effects on the system stability.

11. IANA Considerations

This document requests the registration of the following URIs in the IETF "XML registry" [RFC3688]:

URI	Registrant	XML
urn:ietf:params:xml:ns:yang:ietf-mpls-ldp	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-mpls-ldp-extended	The IESG	N/A

Table 1: URIs

This document requests the registration of the following YANG modules in the "YANG Module Names" registry [RFC6020]:

Name	Namespace	Prefix	Reference
ietf-mpls-ldp	urn:ietf:params:xml:ns:yang:ietf-mpls-ldp	ldp	This document
ietf-mpls-ldp-extended	urn:ietf:params:xml:ns:yang:ietf-mpls-ldp-extended	ldp-ext	This document

Table 2: YANG Modules

-- RFC Editor: Replace "this document" with the document RFC number at time of publication, and remove this note.

12. Acknowledgments

The authors would like to acknowledge Eddie Chami, Nagendra Kumar, Mannan Venkatesan, and Pavan Beeram for their contribution to this document.

We also acknowledge Ladislav Lhotka, Jan Lindblad, Tom Petch, Yingzhen Qu, and Benjamin Kaduk for their detailed review of the model during WG and IESG.

13. Contributors

Danial Johari
Cisco Systems
Email: dajohari@cisco.com

Loa Andersson
Huawei Technologies
Email: loa@pi.nu

Jeff Tantsura
Apstra
Email: jefftant.ietf@gmail.com

Matthew Bocci
Nokia
Email: matthew.bocci@nokia.com

Reshad Rahman
Cisco Systems
Email: rrahman@cisco.com

Stephane Litkowski
Cisco Systems
Email: slitkows@cisco.com

14. Normative References

[I-D.ietf-rtgwg-policy-model]
Qu, Y., Tantsura, J., Lindem, A., and X. Liu, "A YANG Data Model for Routing Policy Management", Work in Progress, Internet-Draft, draft-ietf-rtgwg-policy-model-09, 4 March 2020, <https://tools.ietf.org/html/draft-ietf-rtgwg-policy-model-09>.

[RFC3478] Leelanivas, M., Rekhter, Y., and R. Aggarwal, "Graceful Restart Mechanism for Label Distribution Protocol", RFC 3478, DOI 10.17487/RFC3478, February 2003, <https://www.rfc-editor.org/info/rfc3478>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,

DOI 10.17487/RFC3688, January 2004,
<https://www.rfc-editor.org/info/rfc3688>.

[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>.

[RFC5331] Aggarwal, R., Rekhter, Y., and E. Rosen, "MPLS Upstream Label Assignment and Context-Specific Label Space", RFC 5331, DOI 10.17487/RFC5331, August 2008, <https://www.rfc-editor.org/info/rfc5331>.

[RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", RFC 5443, DOI 10.17487/RFC5443, March 2009, <https://www.rfc-editor.org/info/rfc5443>.

[RFC5561] Thomas, B., Raza, K., Aggarwal, S., Aggarwal, R., and JL. Le Roux, "LDP Capabilities", RFC 5561, DOI 10.17487/RFC5561, July 2009, <https://www.rfc-editor.org/info/rfc5561>.

[RFC5918] Asati, R., Minei, I., and B. Thomas, "Label Distribution Protocol (LDP) 'Typed Wildcard' Forward Equivalence Class (FEC)", RFC 5918, DOI 10.17487/RFC5918, August 2010, <https://www.rfc-editor.org/info/rfc5918>.

[RFC5919] Asati, R., Mohapatra, P., Chen, E., and B. Thomas, "Signaling LDP Label Advertisement Completion", RFC 5919, DOI 10.17487/RFC5919, August 2010, <https://www.rfc-editor.org/info/rfc5919>.

[RFC5920] Fang, L., Ed., "Security Framework for MPLS and GMPLS Networks", RFC 5920, DOI 10.17487/RFC5920, July 2010, <https://www.rfc-editor.org/info/rfc5920>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6389] Aggarwal, R. and JL. Le Roux, "MPLS Upstream Label Assignment for LDP", RFC 6389, DOI 10.17487/RFC6389, November 2011, <https://www.rfc-editor.org/info/rfc6389>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7277] Bjorklund, M., "A YANG Data Model for IP Management", RFC 7277, DOI 10.17487/RFC7277, June 2014, <https://www.rfc-editor.org/info/rfc7277>.

[RFC7552] Asati, R., Pignataro, C., Raza, K., Manral, V., and R. Papneja, "Updates to LDP for IPv6", RFC 7552, DOI 10.17487/RFC7552, June 2015, <https://www.rfc-editor.org/info/rfc7552>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8177] Lindem, A., Ed., Qu, Y., Yeung, D., Chen, I., and J. Zhang, "YANG Data Model for Key Chains", RFC 8177, DOI 10.17487/RFC8177, June 2017, <https://www.rfc-editor.org/info/rfc8177>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8344] Bjorklund, M., "A YANG Data Model for IP Management", RFC 8344, DOI 10.17487/RFC8344, March 2018, <https://www.rfc-editor.org/info/rfc8344>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8529] Berger, L., Hopps, C., Lindem, A., Bogdanovic, D., and X. Liu, "YANG Data Model for Network Instances", RFC 8529, DOI 10.17487/RFC8529, March 2019, <https://www.rfc-editor.org/info/rfc8529>.

15. Informative References

[I-D.ietf-mpls-mldp-yang] Raza, K., Liu, X., Esale, S., Andersson, L., Tantsura, J., and S. Krishnaswamy, "YANG Data Model for MPLS mLDP", Work in Progress, Internet-Draft, draft-ietf-mpls-mldp-yang-06, 31 May 2019, <https://tools.ietf.org/html/draft-ietf-mpls-mldp-yang-06>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[RFC7307] Zhao, Q., Raza, K., Zhou, C., Fang, L., Li, L., and D. King, "LDP Extensions for Multi-Topology", RFC 7307, DOI 10.17487/RFC7307, July 2014, <https://www.rfc-editor.org/info/rfc7307>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

Appendix A. Data Tree Example

This section contains an example of an instance data tree in the JSON encoding [RFC7951], containing both configuration and state data.

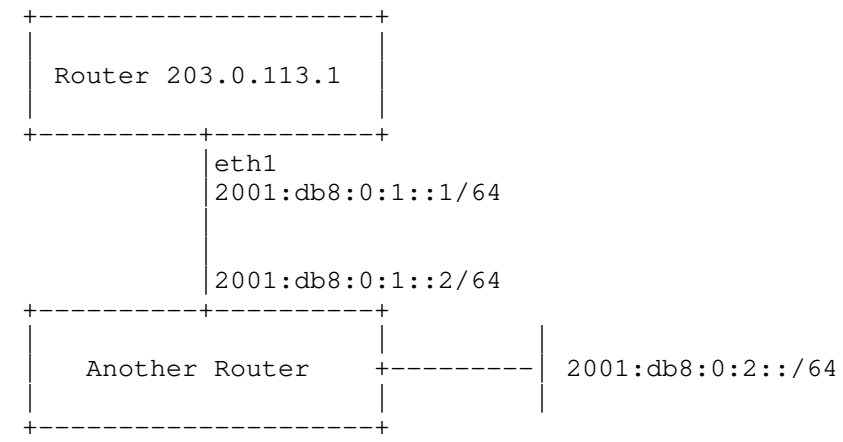


Figure 12: Example topology

The configuration instance data tree for Router 203.0.113.1 in the above figure could be as follows:

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth1",
        "description": "An interface with LDP enabled.",
        "type": "iana-if-type:ethernetCsmacd",
        "ietf-ip:ipv6": {
          "address": [
            {
              "ip": "2001:db8:0:1::1",
              "prefix-length": 64
            }
          ],
          "forwarding": true
        }
      }
    ]
  },
  "ietf-routing:routing": {
    "router-id": "203.0.113.1",
    "control-plane-protocols": {
```

```
        "control-plane-protocol": [
          {
            "type": "ietf-mpls-ldp:mpls-ldp",
            "name": "ldp-1",
            "ietf-mpls-ldp:mpls-ldp": {
              "global": {
                "address-families": {
                  "ietf-mpls-ldp-extended:ipv6": {
                    "enabled": true,
                    "transport-address": "2001:db8:0:1::1"
                  }
                }
              },
              "discovery": {
                "interfaces": {
                  "interface": [
                    {
                      "name": "eth1",
                      "address-families": {
                        "ietf-mpls-ldp-extended:ipv6": {
                          "enabled": true
                        }
                      }
                    }
                  ]
                }
              }
            }
          }
        ]
      }
    }
  }
```

Figure 13: Example Configuration data in JSON

The corresponding operational state data for Router 203.0.113.1 could be as follows:

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth1",
        "description": "An interface with LDP enabled.",
        "type": "iana-if-type:ethernetCsmacd",
        "phys-address": "00:00:5e:00:53:01",
```

```
          "oper-status": "up",
          "statistics": {
            "discontinuity-time": "2018-09-10T15:16:27-05:00"
          },
          "ietf-ip:ipv6": {
            "forwarding": true,
            "mtu": 1500,
            "address": [
              {
                "ip": "2001:db8:0:1::1",
                "prefix-length": 64,
                "origin": "static",
                "status": "preferred"
              },
              {
                "ip": "fe80::200:5eff:fe00:5301",
                "prefix-length": 64,
                "origin": "link-layer",
                "status": "preferred"
              }
            ],
            "neighbor": [
              {
                "ip": "2001:db8:0:1::2",
                "link-layer-address": "00:00:5e:00:53:02",
                "origin": "dynamic",
                "is-router": [null],
                "state": "reachable"
              },
              {
                "ip": "fe80::200:5eff:fe00:5302",
                "link-layer-address": "00:00:5e:00:53:02",
                "origin": "dynamic",
                "is-router": [null],
                "state": "reachable"
              }
            ]
          }
        }
      ]
    },
    "ietf-routing:routing": {
      "router-id": "203.0.113.1",
      "interfaces": {
        "interface": [
          "eth1"
        ]
      },
```

```
      "control-plane-protocols": {
        "control-plane-protocol": [
          {
            "type": "ietf-mpls-ldp:mpls-ldp",
            "name": "ldp-1",
            "ietf-mpls-ldp:mpls-ldp": {
              "global": {
                "address-families": {
                  "ietf-mpls-ldp-extended:ipv6": {
                    "enabled": true,
                    "transport-address": "2001:db8:0:1::1"
                  }
                }
              },
              "discovery": {
                "interfaces": {
                  "interface": [
                    {
                      "name": "eth1",
                      "address-families": {
                        "ietf-mpls-ldp-extended:ipv6": {
                          "enabled": true,
                          "hello-adjacencies": {
                            "hello-adjacency": [
                              {
                                "adjacent-address":
                                "fe80::200:5eff:fe00:5302",
                                "flag": ["adjacency-flag-active"],
                                "hello-holdtime": {
                                  "adjacent": 15,
                                  "negotiated": 15,
                                  "remaining": 9
                                },
                                "next-hello": 3,
                                "statistics": {
                                  "discontinuity-time":
                                  "2018-09-10T15:16:27-05:00"
                                },
                                "peer": {
                                  "lsr-id": "203.0.113.2",
                                  "label-space-id": 0
                                }
                              }
                            ]
                          }
                        }
                      }
                    }
```

```
                ]
              }
            },
            "peers": {
              "peer": [
                {
                  "lsr-id": "203.0.113.2",
                  "label-space-id": 0,
                  "label-advertisement-mode": {
                    "local": "downstream-unsolicited",
                    "peer": "downstream-unsolicited",
                    "negotiated": "downstream-unsolicited"
                  },
                  "next-keep-alive": 5,
                  "session-holdtime": {
                    "peer": 180,
                    "negotiated": 180,
                    "remaining": 78
                  },
                  "session-state": "operational",
                  "tcp-connection": {
                    "local-address": "fe80::200:5eff:fe00:5301",
                    "local-port": 646,
                    "remote-address": "fe80::200:5eff:fe00:5302",
                    "remote-port": 646
                  },
                  "up-time": 3438100,
                  "statistics": {
                    "discontinuity-time": "2018-09-10T15:16:27-05:00"
                  }
                }
              ]
            }
          }
        }
      ]
    }
  }
}
```


Figure 14: Example Operational data in JSON

Authors' Addresses

Kamran Raza (editor)
Cisco Systems

Canada
Email: skraza@cisco.com

Rajiv Asati
Cisco Systems
United States of America
Email: rajiva@cisco.com

Xufeng Liu
Volta Networks
United States of America
Email: xufeng.liu.ietf@gmail.com

Santosh Esale
Juniper Networks
United States of America
Email: sesale@juniper.net

Xia Chen
Huawei Technologies
China
Email: jescia.chenxia@huawei.com

Himanshu Shah
Ciena Corporation
United States of America
Email: hshah@ciena.com

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 14, 2021

K. Raza
Cisco Systems

X. Liu
Volta Networks

S. Esale
Juniper Networks

L. Andersson
Huawei Technologies

J. Tantsura
Nuage Networks

S. Krishnaswamy
Individual

July 13, 2020

YANG Data Model for MPLS mLDP
draft-ietf-mpls-mldp-yang-07

Abstract

This document describes a YANG data model for Multi-Protocol Label Switching (MPLS) Multipoint Label Distribution Protocol (mLDP). The mLDP data model augments the LDP data model.

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA).

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 January 14, 2021.

Table of Contents

1. Introduction

This document introduces a YANG data model for MPLS Multipoint Label Distribution Protocol (mLDP). The mLDP model being defined here is dependent on the LDP YANG data model [I-D.ietf-mpls-ldp-yang]. This implies that an operator will need to use the base LDP module to configure and manage the control plane for mLDP. For example, an operator would enable LDP discovery on MPLS interface to establish LDP/mLDP peering on which mLDP bindings could be exchanged. Similarly, an operator could query state information for an LDP peer in order to verify peering attributes, etc.

Moreover, it is important to note here that any assumptions made in the LDP model also hold true in this document, unless otherwise explicitly stated.

Like its parent LDP data model, this mLDP model also defines the following constructs for managing the mLDP protocol:

- Configuration
- Operational State
- Executables (Actions)
- Notifications

The modeling in this document complies with the Network Management Datastore Architecture (NMDA) [RFC8342]. The operational state data is combined with the associated configuration data in the same hierarchy [RFC8407]. When protocol states are retrieved from the NMDA operational state datastore, the returned states cover all "config true" (rw) and "config false" (ro) nodes defined in the schema.

This document is organized to define the data model for each of the above constructs in the sequence as listed above.

1.1. Base and Extended

Like the LDP model, the configuration and state items are divided into the following two broad categories:

- Base
- Extended

The "base" category contains the basic and fundamental features that are covered in the mLDP base specification [RFC6388] alongwith few significant extension like targeted mLDP [RFC7060], constituting the minumum requirements for an mLDP deployment. Whereas, the "extended" category contains all other non-base features (such as recursive FEC support, protection etc.). All the items in the base category are mandatory and hence no "if-feature" is allowed under the "base" category. While "base" model support will suffice for small deployments, large deployments will require not only the "base" module support but also "extended" support for some selected and required features.

The base and extended categories are defined in their own modules ietf-mpls-mldp and ietf-mpls-mldp-extended respectively, each of which augments the LDP base model as defined within the ietf-mpls-ldp module [I-D.ietf-mpls-ldp-yang].

Like LDP, the mLDP "base" model configuration and state covers ipv4 address-family only, with ipv6 address-family related configuration and state be covered in the "extended" model.

In this document, when a simplified graphical representation of YANG model is presented in a tree diagrams, the meaning of the symbols in these tree diagrams is defined in [RFC8340].

2. 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Overview

This document defines a YANG module named "ietf-mpls-mldp" for the mLDP YANG base data model that augments /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp defined in [I-D.ietf-mpls-ldp-yang]. The

document also defines the "ietf-mpls-mldp-extended" YANG module that models the extended mLDP features.

The following diagram depicts high level mLDP yang tree organization and hierarchy with respect to LDP:

```
          +-- rw routing
            +-- rw control-plane-protocols
               +-- rw mpls-ldp
                   +-- rw some_ldp_container
                   |   +-- rw mldp
                   |       +-- rw ...                  // mldp base
                   |       |   +-- rw ...
                   |       |   +-- ro ...
                   |       |   +--
                   |       +-- rw mldp-ext:...         // mldp extended
                   |       |   +-- rw ...
                   |       |   +-- ro ...
                   |       |   +--
                   |   +-- ro some_ldp_container
                   |       +-- ro mldp
                   |           +-- ro ...              // mldp base
                   |           |   +-- ro ...
                   |           |   +--
                   |           +-- ro mldp-ext:...     // mldp extended
                   |               +-- ro ...
                   |               +--

       notifications:
          +--- n mpls-mldp-some_event
          +--- n ...
```

Figure 1

3.1. Scope

The main mLDP areas and features that are within the scope of this model are as follows:

- Base:
  - mLDP Base Specification [RFC6388]
  - Targeted mLDP [RFC7060]
  - Configured Leaf LSPs (manually provisioned)

- Extended:
  - mLDP Recursive FEC [RFC6512]
  - mLDP Fast-Reroute (FRR):
    - Node Protection [RFC7715]
    - Multicast-only [RFC7431]
  - In-band Signaling:
    - mLDP In-band Signaling [RFC6826]
    - mLDP In-band signaling in a VRF [RFC7246]
    - mLDP In-band Signaling with Wildcards [RFC7438]
  - Hub-and-Spoke Multipoint LSPs [RFC7140]

[Ed Note: Some of the topics in the above list are to be addressed/ extended in a later revision of this document].

3.2. FEC Types

The FEC for Multipoint LSP is presented as (root-address, opaque-element). The following table lists various type of MP opaque elements with their keys, as covered in the configuration and state model:

Opaque Type	Key	RFC
Generic LSP Identifier	LSP Id	[RFC6388]
Transit IPv4 Source	Source, Group	[RFC6826]
Transit IPv6 Source	Source, Group	[RFC6826]
Transit IPv4 Bidir	RP, Group	[RFC6826]
Transit IPv6 Bidir	RP, Group	[RFC6826]
Transit VPNv4 Source	Source, Group, RD	[RFC7246]
Transit VPNv6 Source	Source, Group, RD	[RFC7246]
Transit VPNv4 Bidir	RP, Group, RD	[RFC7246]
Transit VPNv6 Bidir	RP, Group, RD	[RFC7246]
Recursive Opaque	Root	[RFC6512]
VPN-Recursive Opaque	Root, RD	[RFC6512]

Table 1: MP Opaque Types and keys

It should be noted that there are three basic types (LSP Id, Source, and Bidir) and then there are variants (VPN, recursive, VPN-recursive) on top of these basic types.

The "base" model includes only the "Generic LSP Identifier" opaque type (for ipv4), while rest of the above types are covered by the "extended" model.

4. Configuration

4.1. Configuration Hierarchy

The high-level configuration organization for the base and extended mLDP follows:

```
augment /rt:routing/rt:control-plane-protocols/rt:control-plane-protocol:
      +-- mpls-ldp
          +-- global
               +-- ...
               +-- ...
               +-- mldp
               |   +-- ...
               |   +-- ...
               |   +-- address-families
               |       +-- ipv4
               |       |   +-- ...
               |       |   +-- mldp-ext: ...
               |       |   +-- ...
               |       |   +-- configured-leaf-lsps
               |       |       +-- ...
               |       |       +-- ...
               |       |       +-- mldp-ext: ...
               |       |       +-- ...
               |       +-- mldp-ext: ipv6
               |           +-- ...
               |           +-- ...
               |           +-- configured-leaf-lsps
               |               +-- ...
               |               +-- ...
               +-- capability
               |   +-- mldp
               |       +-- ...
               |       +-- mldp-ext: ...
               |       +-- ...
               +-- forwarding-nexthop
                   +--- interfaces
                        +--- interface* [name]
                             +--- mldp-ext: ...
```

Figure 2

From above hierarchy, we can categorize mLDP configuration parameters into two types:

- Parameters that are mLDP specific
- Parameters that leverage/extend LDP containers and parameters

The following subsections first describe the mLDP specific configuration parameters, followed by those leveraging LDP. It should be noted that these parameters are defined under their respective base or extended module as per their categorization.

4.2. mldp global container

mldp container is an augmentation of LDP global container and holds the configuration related to items that are mLDP specific. The main items under this container are:

- mLDP enablement: To enable mLDP under a (VRF) routing instance, mldp is enabled in the mldp container under LDP. Given that mLDP requires LDP signaling, it is not sensible to allow disabling the LDP control plane under a (VRF) network-instance while requiring mLDP to be enabled for the same. However, if a user wants to only allow signaling for multipoint FECs on an LDP/mLDP enabled VRF instance, he/she can use LDP label-policies to disable unicast FECs under the VRF.

- mLDP per-AF features: mLDP manages its own list of IP address-families and the features enabled underneath. The per-AF mLDP configuration items include:

  * Multicast-only FRR: This enables Multicast-only FRR functionality for a given AF under mLDP. The feature allows route-policy to be configured for finer control/applicability of the feature.

  * Recursive FEC: The recursive-fec feature [RFC6512] can be enabled per-AF with a route-policy.

  * Configured Leaf LSPs: To provision multipoint leaf LSPs manually, a per-AF container is provided under LDP. The configuration is flexible and allows a user to specify MP LSPs of type p2mp or mp2mp with IPv4 or IPv6 root address(es) by using either LSP-Id or (S,G).

Targeted mLDP feature specification [RFC7060] does not require any mLDP specific configuration. It, however, requires LDP upstream-label-assignment capability [RFC6389] to be enabled.

4.3. Leveraging LDP containers

The mLDP configuration model leverages following configuration areas and containers that are already defined for LDP:

- Capabilities: A new container "mldp" is defined that augments LDP's capabilities container. This new container specifies any mLDP specific capabilities and their parameters. Moreover, a new container "mldp" is also added by augmenting LDP per-peer capability container to override/control mLDP specific capabilities on a peer level. In the scope of this document, the

most important capabilities related to mLDP are p2mp, mp2mp, make-before-break, hub-and-spoke, and node-protection.

- Discovery and Peering: mLDP requires LDP discovery and peer procedures to form mLDP peering. A peer is treated as an mLDP peer only when either P2MP or MP2MP capabilities have been successfully exchanged with the peer. If a user wish to selectively enable or disable mLDP with a LDP-enabled peer, he/she may use per-peer mLDP capabilities configuration. In most common deployments, it is desirable to disable mLDP (capabilities announcements) on a targeted-only LDP peering, where targeted-only peer is the one whose discovery sources are the targeted type only.

- Forwarding: By default, mLDP is allowed to select any of the LDP enabled interface as a downstream interface towards a next-hop (LDP/mLDP peer) for MP LSP programming. However, a configuration option is provided to allow mLDP to exclude a given interface from such a selection. Note that such a configuration option will be useful only when there are more than one interface available for the downstream selection.

4.4. Configuration Tree

4.4.1. Base

A simplified graphical representation of the data model for mLDP base configuration follows:

```
module: ietf-mpls-mldp
  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:global/ldp:capa
bility:
    +--rw mldp
       +--rw p2mp
       |  +--rw enable?   boolean
       +--rw mp2mp
       |  +--rw enable?   boolean
       +--rw make-before-break
          +--rw enable?             boolean
          +--rw switchover-delay?   uint16
          +--rw timeout?            uint16

  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:global:
    +--rw mldp
       +--rw enable?             boolean
       +--rw address-families
          +--rw ipv4
             +--rw configured-leaf-lsps
                +--rw opaque-element-lspid
                   +--rw fec-label* [root-address lsp-id]
                      +--rw root-address              inet:ipv4-address
                      +--rw lsp-id                    uint32
                      +--rw multipoint-type?          multipoint-type
```

Figure 3

4.4.2. Extended

A simplified graphical representation of the data model for mLDP extended configuration follows:

```
module: ietf-mpls-mldp
  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:global/ldp:capa
bility:
    +--rw mldp
       +--rw mldp-ext:hub-and-spoke {capability-mldp-hsmp}?
       |  +--rw mldp-ext:enable?   boolean
       +--rw mldp-ext:node-protection {capability-mldp-node-protection}?
          +--rw mldp-ext:plr?           boolean
          +--rw mldp-ext:merge-point
             +--rw mldp-ext:enable?                              boolean
             +--rw mldp-ext:targeted-session-teardown-delay?   uint16

  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:global:
    +--rw mldp
       +--rw enable?             boolean
```

```
       +--rw address-families
          +--rw ipv4
          |  +--rw configured-leaf-lsps
          |  |  +--rw mldp-ext:opaque-element-transit
          |  |  |  +--rw mldp-ext:fec-label* [root-address source-address group-a
ddress rd recur-root-address recur-rd]
          |  |  |     +--rw mldp-ext:root-address          inet:ipv4-address
          |  |  |     +--rw mldp-ext:source-address        inet:ip-address
          |  |  |     +--rw mldp-ext:group-address         inet:ip-address-no-zon
e
          |  |  |     +--rw mldp-ext:rd                    route-distinguisher
          |  |  |     +--rw mldp-ext:recur-root-address    inet:ip-address
          |  |  |     +--rw mldp-ext:recur-rd              route-distinguisher
          |  |  |     +--rw mldp-ext:multipoint-type?      mldp:multipoint-type
          |  |  +--rw mldp-ext:opaque-element-bidir
          |  |     +--rw mldp-ext:fec-label* [root-address rp group-address rd re
cur-root-address recur-rd]
          |  |        +--rw mldp-ext:root-address          inet:ipv4-address
          |  |        +--rw mldp-ext:rp                    inet:ip-address
          |  |        +--rw mldp-ext:group-address         inet:ip-address-no-zon
e
          |  |        +--rw mldp-ext:rd                    route-distinguisher
          |  |        +--rw mldp-ext:recur-root-address    inet:ip-address
          |  |        +--rw mldp-ext:recur-rd              route-distinguisher
          |  |        +--rw mldp-ext:multipoint-type?      mldp:multipoint-type
          |  +--rw mldp-ext:multicast-only-frr {mldp-mofrr}?
          |  |  +--rw mldp-ext:prefix-list?   ldp-ext:prefix-list-ref
          |  +--rw mldp-ext:recursive-fec
          |     +--rw mldp-ext:prefix-list?   ldp-ext:prefix-list-ref
          +--rw mldp-ext:ipv6
             +--rw mldp-ext:configured-leaf-lsps
             |  +--rw mldp-ext:opaque-element-lspid
             |  |  +--rw mldp-ext:fec-label* [root-address lsp-id]
             |  |     +--rw mldp-ext:root-address       inet:ipv6-address
             |  |     +--rw mldp-ext:lsp-id             uint32
             |  |     +--rw mldp-ext:multipoint-type?   mldp:multipoint-type
             |  |     +--rw mldp-ext:recursive-fec* [recur-root-address recur-rd]
             |  |        +--rw mldp-ext:recur-root-address    inet:ip-address
             |  |        +--rw mldp-ext:recur-rd              route-distinguisher
             |  |        +--rw mldp-ext:multipoint-type?      mldp:multipoint-typ
e
             |  +--rw mldp-ext:opaque-element-transit
             |  |  +--rw mldp-ext:fec-label* [root-address source-address group-a
ddress rd recur-root-address recur-rd]
             |  |     +--rw mldp-ext:root-address          inet:ipv6-address
             |  |     +--rw mldp-ext:source-address        inet:ip-address
             |  |     +--rw mldp-ext:group-address         inet:ip-address-no-zon
e
             |  |     +--rw mldp-ext:rd                    route-distinguisher
             |  |     +--rw mldp-ext:recur-root-address    inet:ip-address
             |  |     +--rw mldp-ext:recur-rd              route-distinguisher
             |  |     +--rw mldp-ext:multipoint-type?      mldp:multipoint-type
             |  +--rw mldp-ext:opaque-element-bidir
             |     +--rw mldp-ext:fec-label* [root-address rp group-address rd re
cur-root-address recur-rd]
             |        +--rw mldp-ext:root-address          inet:ipv6-address
```

```
              |        +--rw mldp-ext:rp                    inet:ip-address
              |        +--rw mldp-ext:group-address         inet:ip-address-no-zon
e
              |        +--rw mldp-ext:rd                    route-distinguisher
              |        +--rw mldp-ext:recur-root-address    inet:ip-address
              |        +--rw mldp-ext:recur-rd              route-distinguisher
              |        +--rw mldp-ext:multipoint-type?      mldp:multipoint-type
              +--rw mldp-ext:multicast-only-frr {mldp-mofrr}?
              |  +--rw mldp-ext:prefix-list?   ldp-ext:prefix-list-ref
              +--rw mldp-ext:recursive-fec
                 +--rw mldp-ext:prefix-list?   ldp-ext:prefix-list-ref

  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:peers/ldp:peer/
ldp:capability:
    +--rw mldp {per-peer-capability}?
       +--rw p2mp
       |  +--rw enable?   boolean
       +--rw mp2mp
       |  +--rw enable?   boolean
       +--rw make-before-break
          +--rw enable?             boolean
          +--rw switchover-delay?   uint16
          +--rw timeout?            uint16

  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:global/ldp-ext:
forwarding-nexthop/ldp-ext:interfaces/ldp-ext:interface/ldp-ext:address-family:
  +--rw mldp-disable?   boolean
```

Figure 4

5. Operational State

The operational state of mLDP can be queried and obtained from various read-only mdlp "state" containers that augment ldp containers.

5.1. Base

A simplified graphical representation of the data model for mLDP base operational state follows:

```
module: ietf-mpls-mldp
  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:peers/ldp:peer/
ldp:received-peer-state/ldp:capability:
    +--ro mldp
       +--ro p2mp
       |  +--ro enable?   boolean
       +--ro mp2mp
       |  +--ro enable?   boolean
       +--ro make-before-break
          +--ro enable?   boolean

  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:global:
    +--rw mldp
       +--rw enable?             boolean
       +--rw address-families
          +--rw ipv4
             +--ro roots
                +--ro root* [root-address]
                   +--ro root-address    inet:ipv4-address
                   +--ro is-self?        boolean
                   +--ro reachability* [address interface]
                   |  +--ro address      inet:ipv4-address
                   |  +--ro interface    if:interface-ref
                   |  +--ro peer?        -> ../../../../../../../../ldp:peers/pee
r/lsr-id
                   +--ro bindings
                      +--ro opaque-element-lspid
                         +--ro fec-label* [lsp-id]
                            +--ro lsp-id                     uint32
                            +--ro multipoint-type?           multipoint-type
                            +--ro peer* [direction peer advertisement-type]
                               +--ro direction               ldp:downstream-upstre
am
                               +--ro peer                    -> /rt:routing/contro
l-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
                               +--ro advertisement-type      ldp:advertised-receiv
ed
                               +--ro label?                  rt-types:mpls-label
                               +--ro mbb-role?               enumeration
                               +--ro mldp-ext:mofrr-role?    mofrr-role
```

Figure 5

5.2. Extended

A simplified graphical representation of the data model for mLDP extended operational state follows:

```
module: ietf-mpls-mldp

  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:peers/ldp:peer/
ldp:received-peer-state/ldp:capability:
```

```
    +--ro mldp
       +--ro mldp-ext:hub-and-spoke
       |  +--ro mldp-ext:enable?   boolean
       +--ro mldp-ext:node-protection
          +--ro mldp-ext:plr?           boolean
          +--ro mldp-ext:merge-point?   boolean

  augment /rt:routing/rt:control-plane-protocols/ldp:mpls-ldp/ldp:global:
    +--rw mldp
       +--rw enable?             boolean
       +--rw address-families
          +--rw ipv4
          |  +--ro roots
          |     +--ro root* [root-address]
          |        +--ro root-address    inet:ipv4-address
          |        +--ro bindings
          |           +--ro opaque-element-lspid
          |           |     +--ro mldp-ext:recursive-fec* [recur-root-address rec
ur-rd]
          |           |        +--ro mldp-ext:recur-root-address    inet:ip-addre
ss
          |           |        +--ro mldp-ext:recur-rd              route-disting
uisher
          |           |        +--ro mldp-ext:multipoint-type?      mldp:multipoi
nt-type
          |           |        +--ro mldp-ext:peer* [direction peer advertisement
-type]
          |           |           +--ro mldp-ext:direction             ldp:downst
ream-upstream
          |           |           +--ro mldp-ext:peer                  -> /rt:rou
ting/control-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
          |           |           +--ro mldp-ext:advertisement-type    ldp:advert
ised-received
          |           |           +--ro mldp-ext:label?                rt-types:m
pls-label
          |           |           +--ro mldp-ext:mbb-role?             enumeratio
n
          |           |           +--ro mldp-ext:mofrr-role?           mofrr-role
          |           +--ro mldp-ext:opaque-element-transit
          |           |  +--ro mldp-ext:fec-label* [source-address group-address
rd recur-root-address recur-rd]
          |           |     +--ro mldp-ext:source-address        inet:ip-address
          |           |     +--ro mldp-ext:group-address         inet:ip-address-
no-zone
          |           |     +--ro mldp-ext:rd                    route-distinguis
her
          |           |     +--ro mldp-ext:recur-root-address    inet:ip-address
          |           |     +--ro mldp-ext:recur-rd              route-distinguis
her
          |           |     +--ro mldp-ext:multipoint-type?      mldp:multipoint-
type
          |           |     +--ro mldp-ext:peer* [direction peer advertisement-ty
pe]
          |           |        +--ro mldp-ext:direction             ldp:downstrea
m-upstream
          |           |        +--ro mldp-ext:peer                  -> /rt:routin
g/control-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
          |           |        +--ro mldp-ext:advertisement-type    ldp:advertise
```

```
d-received
          |           |        +--ro mldp-ext:label?                rt-types:mpls
-label
          |           |        +--ro mldp-ext:mbb-role?             enumeration
          |           |        +--ro mldp-ext:mofrr-role?           mofrr-role
          |           +--ro mldp-ext:opaque-element-bidir
          |              +--ro mldp-ext:fec-label* [rp group-address rd recur-roo
t-address recur-rd]
          |                 +--ro mldp-ext:rp                    inet:ip-address
          |                 +--ro mldp-ext:group-address         inet:ip-address-
no-zone
          |                 +--ro mldp-ext:rd                    route-distinguis
her
```

```
          |                  +--ro mldp-ext:recur-root-address    inet:ip-address
          |                  +--ro mldp-ext:recur-rd              route-distinguis
her
          |                  +--ro mldp-ext:multipoint-type?      mldp:multipoint-
type
          |                  +--ro mldp-ext:peer* [direction peer advertisement-ty
pe]
          |                     +--ro mldp-ext:direction             ldp:downstrea
m-upstream
          |                     +--ro mldp-ext:peer                  -> /rt:routin
g/control-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
          |                     +--ro mldp-ext:advertisement-type    ldp:advertise
d-received
          |                     +--ro mldp-ext:label?                rt-types:mpls
-label
          |                     +--ro mldp-ext:mbb-role?             enumeration
          |                     +--ro mldp-ext:mofrr-role?           mofrr-role
          +--rw mldp-ext:ipv6
             +--ro mldp-ext:roots
                +--ro mldp-ext:root* [root-address]
                   +--ro mldp-ext:root-address    inet:ipv6-address
                   +--ro mldp-ext:is-self?        boolean
                   +--ro mldp-ext:reachability* [address interface]
                   |  +--ro mldp-ext:address      inet:ipv6-address
                   |  +--ro mldp-ext:interface    if:interface-ref
                   |  +--ro mldp-ext:peer?        -> ../../../../../../../../ldp:
peers/peer/lsr-id
                   +--ro mldp-ext:bindings
                      +--ro mldp-ext:opaque-element-lspid
                      |  +--ro mldp-ext:fec-label* [lsp-id]
                      |     +--ro mldp-ext:lsp-id             uint32
                      |     +--ro mldp-ext:multipoint-type?   mldp:multipoint-typ
e
                      |     +--ro mldp-ext:peer* [direction peer advertisement-ty
pe]
                      |     |  +--ro mldp-ext:direction             ldp:downstrea
m-upstream
                      |     |  +--ro mldp-ext:peer                  -> /rt:routin
g/control-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
                      |     |  +--ro mldp-ext:advertisement-type    ldp:advertise
d-received
                      |     |  +--ro mldp-ext:label?                rt-types:mpls
-label
                      |     |  +--ro mldp-ext:mbb-role?             enumeration
                      |     |  +--ro mldp-ext:mofrr-role?           mofrr-role
                      |     +--ro mldp-ext:recursive-fec* [recur-root-address rec
ur-rd]
                      |        +--ro mldp-ext:recur-root-address    inet:ip-addre
ss
                      |        +--ro mldp-ext:recur-rd              route-disting
uisher
                      |        +--ro mldp-ext:multipoint-type?      mldp:multipoi
nt-type
                      |        +--ro mldp-ext:peer* [direction peer advertisement
-type]
                      |           +--ro mldp-ext:direction             ldp:downst
ream-upstream
                      |           +--ro mldp-ext:peer                  -> /rt:rou
```

```
ting/control-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
                        |           +--ro mldp-ext:advertisement-type    ldp:advert
ised-received
                        |           +--ro mldp-ext:label?                rt-types:m
pls-label
                        |           +--ro mldp-ext:mbb-role?             enumeratio
n
                        |           +--ro mldp-ext:mofrr-role?           mofrr-role
                        +--ro mldp-ext:opaque-element-transit
                        |  +--ro mldp-ext:fec-label* [source-address group-address
rd recur-root-address recur-rd]
                        |     +--ro mldp-ext:source-address        inet:ip-address
                        |     +--ro mldp-ext:group-address         inet:ip-address-
no-zone
                        |     +--ro mldp-ext:rd                    route-distinguis
her
                        |     +--ro mldp-ext:recur-root-address    inet:ip-address
```

```
                        |     +--ro mldp-ext:recur-rd              route-distinguis
her
                        |     +--ro mldp-ext:multipoint-type?      mldp:multipoint-
type
                        |     +--ro mldp-ext:peer* [direction peer advertisement-ty
pe]
                        |        +--ro mldp-ext:direction             ldp:downstrea
m-upstream
                        |        +--ro mldp-ext:peer                  -> /rt:routin
g/control-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
                        |        +--ro mldp-ext:advertisement-type    ldp:advertise
d-received
                        |        +--ro mldp-ext:label?                rt-types:mpls
-label
                        |        +--ro mldp-ext:mbb-role?             enumeration
                        |        +--ro mldp-ext:mofrr-role?           mofrr-role
                        +--ro mldp-ext:opaque-element-bidir
                           +--ro mldp-ext:fec-label* [rp group-address rd recur-roo
t-address recur-rd]
                              +--ro mldp-ext:rp                     inet:ip-address
                              +--ro mldp-ext:group-address          inet:ip-address-
no-zone
                              +--ro mldp-ext:rd                     route-distinguis
her
                              +--ro mldp-ext:recur-root-address     inet:ip-address
                              +--ro mldp-ext:recur-rd               route-distinguis
her
                              +--ro mldp-ext:multipoint-type?       mldp:multipoint-
type
                              +--ro mldp-ext:peer* [direction peer advertisement-ty
pe]
                                 +--ro mldp-ext:direction             ldp:downstrea
m-upstream
                                 +--ro mldp-ext:peer                  -> /rt:routin
g/control-plane-protocols/ldp:mpls-ldp/peers/peer/lsr-id
                                 +--ro mldp-ext:advertisement-type    ldp:advertise
d-received
                                 +--ro mldp-ext:label?                rt-types:mpls
-label
                                 +--ro mldp-ext:mbb-role?             enumeration
                                 +--ro mldp-ext:mofrr-role?           mofrr-role
```

Figure 6

5.3. Derived states

The main areas for which mLDP operational derived state is defined are:

- Root
- Bindings (FEC-label)
- Capabilities

5.3.1. Root state

The root address is a fundamental construct for MP FEC bindings and LSPs. The root state provides information on all the known roots in a given address-family and their root reachability information (as learnt from RIB). In case of multi-path reachability to a root, the selection of the upstream path is done on per-LSP basis at the time of LSP setup. Similarly, when protection mechanisms like Make-

before-break (MBB) or Multicast-only FRR (MoFRR) are in place, the path designation as active/standby or primary/backup is also done on per-LSP basis. It should be noted that a given root can be shared amongst multiple P2MP and/or MP2MP LSPs. Moreover, an LSP can be signaled to more than one root for Root Node Redundancy (RNR) purposes.

The following diagram illustrates a root database on a branch/transit LSR:

```
root 203.0.113.1:
    path1:
        RIB: GigEthernet 1/0, 198.51.100.1;
        LDP: peer 192.0.2.1:0
    path2:
        RIB: GigEthernet 2/0, 198.51.100.16;
        LDP: peer 192.0.2.2:0

root 203.0.113.2:
    path1:
        RIB: 198.51.100.100;      (NOTE: This is a recursive path)
        LDP: peer 192.0.2.100:0   (NOTE: T-mLDP peer)

root . . . .
```

Figure 7

A root entry on a root LSR itself will be presented as follows:

```
root 203.0.113.10:
    is-self
```

Figure 8

5.3.2. Bindings state

Binding state provides information on mLDP FEC-label bindings for both the P2MP and MP2MP FEC types. Like LDP, the FEC-label binding derived state is presented in a FEC-centric view per address-family, and provides information on both inbound (received) and outbound (advertised) bindings. The FEC is presented as (root-address, opaque-element-data) as described earlier in section Section 3.2, and the direction (upstream or downstream) is picked with respect to root reachability. In case of MBB or/and MoFRR, the role of a given peer

binding is also provided with respect to MBB (active or standby) or/ and MoFRR (primary or backup).

A high-level tree hierarchy for mLDP bindings state follows:

```
+--rw mpls-ldp!
 +--rw global
   +--rw mldp
      +--rw address-families
         +--rw ipv4 (or ipv6)
            +--ro state
               +--ro roots
                  +--ro root* [root-address]
                     +--ro ....
                     +--ro bindings
                        +--ro opaque-element-xxx
                        |  +--ro fec-label* [type-specific-key]
                        |     +--ro some_key_1  ...
                        |     +--ro some_key_2  ...
                        |     +--ro multipoint-type?          multipoint-type
                        |     +--ro peer* [direction peer advertisement-type]
                        |     |  +--ro direction              ldp:downstream-upst
ream
                        |     |  +--ro peer                   leafref
                        |     |  +--ro advertisement-type     ldp:advertised-rece
ived
                        |     |  +--ro label?                 mpls:mpls-label
                        |     |  +--ro mbb-role?              enumeration
                        |     |  +--ro mldp-ext:mofrr-role?   mofrr-role
                        +--ro opaque-element-yyy
                        |  +--ro fec-label* [type-specific-key]
                        |     +--ro some_key_1  ...
                        ...
```

Figure 9

mLDP binding state is organized and presented per root address, and hence the bindings container in under a root node in the model. The bindings state is made available for FECs pertaining to different types of opaque elements, with some state avaiable under the "base" tree and the rest under the "extended" tree.

In the above tree, the various opaque types along with their type specific key(s) refer to the table Table 1, as captured earlier in the document. For example, if the opaque type is a Generic LSP Identifier, then the type-specific-key will be a uint32 LSP-Id key. Please see the complete model for all other types.

It is important to note the following:

- The address-family ipv4/ipv6 applies to "root" address in the mLDP binding tree. The other addresses (source, group, Rendezvous-Point etc.) do not have to be of the same address family type as the root.

- The "recur-root-address" field applies to the Recursive opaque type, and the (recur-root-address, recur-rd) fields applies to the VPN-Recursive opaque types as defined in [RFC6512].

- In case of a recursive FEC, the address-family of the recur-root-address could be different than the address-family of the root address of the original encapsulated MP FEC.

The following diagram illustrates the FEC-label binding information structure for a P2MP (Transit IPv4 Source type) LSP on a branch/ transit LSR:

```
FEC (root 203.0.113.1, S=198.51.100.1, G=224.1.1.1):
  type: p2mp
  upstream:
    advertised:
      peer 192.0.2.1:0, label 16000 (local)
  downstream:
    received:
      peer 192.0.2.2:0, label 17000 (remote)
      peer 192.0.2.3:0, label 18000  (remote)
```

Figure 10

The following diagram illustrates the FEC-label binding information structure for a similar MP2MP LSP on a branch/transit LSR:

```
          FEC (root 203.0.113.2, RP=198.51.100.2, G=224.1.1.1):
            type: mp2mp
            upstream:
              advertised:
                peer 192.0.2.1:0, label 16000 (local)
              received:
                peer 192.0.2.1:0, label 17000 (remote)
            downstream:
              advertised:
                peer 192.0.2.2:0, label 16001 (local), MBB role=active
                peer 192.0.2.3:0, label 16002 (local), MBB role=standby
              received:
                peer 192.0.2.2:0, label 17001 (remote)
                peer 192.0.2.3:0, label 18001 (remote)
```

Figure 11

5.3.3. Capabilities state

Like LDP, mLDP capabilities state comprise two types of information:

- global: augments ldp:global/ldp:state/ldp:capability.
- per-peer: augments ldp:peers/ldp:peer/ldp:state/ldp:capability

6. Notifications

The mLDP notification module consists of notifications related to changes in the operational state of an mLDP FEC.

6.1. Base

A simplified graphical representation of the base data model for mLDP notifications follows:

```
module: ietf-mpls-mldp
notifications:
    +---n mpls-mldp-fec-event
       +--ro event-type?                      ldp:oper-status-event-type
       +--ro (opaque-element)?
          +--:(opaque-element-lspid)
             +--ro opaque-element-lspid
                +--ro root-address?             inet:ip-address
                +--ro lsp-id?                   uint32
                +--ro multipoint-type?          multipoint-type
                +--ro mldp-ext:recursive-fec
                   +--ro mldp-ext:recur-root-address?   inet:ip-address
                   +--ro mldp-ext:recur-rd?             route-distinguisher
                   +--ro mldp-ext:multipoint-type?      mldp:multipoint-type
```

Figure 12

6.2. Extended

A simplified graphical representation of the extended data model for mLDP notifications follows:

```
module: ietf-mpls-mldp
notifications:
    +---n mpls-mldp-fec-event
       +--ro event-type?                    ldp:oper-status-event-type
       +--ro (opaque-element)?
          +--:(mldp-ext:opaque-element-transit)
          |  +--ro mldp-ext:opaque-element-transit
          |     +--ro mldp-ext:root-address?         inet:ip-address
          |     +--ro mldp-ext:source-address?       inet:ip-address
          |     +--ro mldp-ext:group-address?        inet:ip-address-no-zone
          |     +--ro mldp-ext:rd?                   route-distinguisher
          |     +--ro mldp-ext:recur-root-address?   inet:ip-address
          |     +--ro mldp-ext:recur-rd?             route-distinguisher
          |     +--ro mldp-ext:multipoint-type?      mldp:multipoint-type
          +--:(mldp-ext:opaque-element-bidir)
             +--ro mldp-ext:opaque-element-bidir
                +--ro mldp-ext:root-address?         inet:ip-address
                +--ro mldp-ext:rp?                   inet:ip-address
                +--ro mldp-ext:group-address?        inet:ip-address-no-zone
                +--ro mldp-ext:rd?                   route-distinguisher
                +--ro mldp-ext:recur-root-address?   inet:ip-address
                +--ro mldp-ext:recur-rd?             route-distinguisher
                +--ro mldp-ext:multipoint-type?      mldp:multipoint-type
```

Figure 13

7. Actions

Currently, no RPCs/actions are defined for mLDP.

8. Open Items

A list of open items that are to be addressed in future revisions of this document follows:

- Specify default values for configuration parameters

9. YANG Specification

The YANG definition, i.e., the modules, for mLDP constructs defined earlier in this document are includind the subsections below.

9.1. Base

This YANG module imports types defined in [RFC6991], [RFC8343], [RFC8349], [I-D.ietf-mpls-ldp-yang], and [RFC8294].

```
<CODE BEGINS> file "ietf-mpls-mldp@2018-10-22.yang"
// RFC Editor: replace the above date with the date of
// publication and remove this note.

module ietf-mpls-mldp {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-mldp";
  prefix "mldp";

  import ietf-inet-types {
    prefix "inet";
    reference "RFC 6991: Common YANG Data Types";
  }

  import ietf-interfaces {
    prefix "if";
    reference "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-mpls-ldp {
    prefix "ldp";
    reference "RFC XXXX: A YANG Data Model for MPLS LDP";
// RFC Editor: replace the XXXX with actual LDP YANG RFC number at
// time of publication and remove this note.
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
       version)";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

  organization
    "IETF MPLS Working Group";
  contact
    "WG Web:   <http://tools.ietf.org/wg/mpls/>
     WG List:  <mailto:mpls@ietf.org>

     Editor:   Kamran Raza
               <mailto:skraza@cisco.com>
```

```
        Editor:   Sowmya Krishnaswamy
                  <mailto:krishnaswamy.sowmya@gmail.com>

        Editor:   Xufeng Liu
                  <mailto:xufeng.liu.ietf@gmail.com>

        Editor:   Santosh Esale
                  <mailto:sesale@juniper.net>

        Editor:   Loa Andersson
                  <mailto:loa@pi.nu>

        Editor:   Jeff Tantsura
                  <mailto:jefftant.ietf@gmail.com>";

    description
      "This YANG module defines the essential components for the
       management of Multi-Protocol Label Switching (MPLS) Multipoint
       LDP (mLDP).

       Copyright (c) 2018 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

       Redistribution and use in source and binary forms, with or
       without modification, is permitted pursuant to, and subject to
       the license terms contained in, the Simplified BSD License set
       forth in Section 4.c of the IETF Trust's Legal Provisions
       Relating to IETF Documents
       (http://trustee.ietf.org/license-info).

       This version of this YANG module is part of RFC XXXX; see the
       RFC itself for full legal notices.";

       // RFC Editor: replace XXXX with actual RFC number and remove
       // this note

    revision 2018-10-22 {
      // RFC Editor: replace the above date 2018-10-22 with the date of
      // publication and remove this note.

      description
        "Initial revision.";
      reference
        "RFC XXXX: Base YANG Data Model for MPLS mLDP";
       // RFC Editor: replace XXXX with actual RFC number and remove
       // this note
    }
```

```
     /*
      * Typedefs
      */
     typedef multipoint-type {
       type enumeration {
         enum p2mp {
           description "Point to multipoint";
         }
         enum mp2mp {
           description "Multipoint to multipoint";
         }
       }
       description
         "The type of a multipoint LSP: either Point to multipoint
          (p2mp) or Multipoint to multipoint (mp2mp)";
     }

     /*
      * Groupings
      */
     grouping mldp-capabilities {
       description
         "A grouping describing the protocol capabilities of mLDP";
       container p2mp {
         description
           "Configuration and state information for the
            point-to-multipoint capability";
         leaf enable {
           type boolean;
           description
             "'true' to enable the point-to-multipoint capability";
         }
       }
       container mp2mp {
         description
           "Configuration and state information for the
            multipoint-to-multipoint capability";
         leaf enable {
           type boolean;
           description
             "'true' to enable the multipoint-to-multipoint capability";
         }
       }
       container make-before-break {
         description
           "Configuration and state information for the
            make-before-break capability.";
         leaf enable {
```

```
          type boolean;
          description
            "'true' to enable the make-before-break capability";
        }
        leaf switchover-delay {
          type uint16;
          units seconds;
          description
            "Switchover delay in seconds";
        }
        leaf timeout {
          type uint16;
          units seconds;
          description
            "Timeout in seconds";
        }
      }
    } // mldp-capabilities

    grouping  mldp-binding-label-peer-state-attributes {
      description
        "mLDP label binding per peer attributes";
      leaf direction {
        type ldp:downstream-upstream;
        description
          "Downstream or upstream";
      }
      leaf peer {
        type leafref {
          path
            "/rt:routing/rt:control-plane-protocols/"
          + "ldp:mpls-ldp/ldp:peers/ldp:peer/ldp:lsr-id";
        }
        description
          "LDP peer from which this binding is received,
           or to which this binding is advertised.";
      }
      leaf advertisement-type {
        type ldp:advertised-received;
        description
          "Advertised or received";
      }
      leaf label {
        type rt-types:mpls-label;
        description
          "Advertised (outbound) or received (inbound) label";
      }
      leaf mbb-role {
```

```
        when "../direction = 'upstream'" {
          description
            "This leaf is used for upstream only.";
        }
        type enumeration {
          enum none {
            description "Make-Before-Break (MBB) is not enabled";
          }
          enum active {
            description "This LSP is active.";
          }
          enum inactive {
            description "This LSP is inactive.";
          }
        }
        description
          "The MBB status of this LSP";
      }
    } // mldp-binding-label-peer-state-attributes

    grouping  mldp-binding-label-state-attributes {
      description
        "mLDP label binding attributes";
      list peer {
        key "direction peer advertisement-type";
        description
          "List of advertised and received peers";
        uses mldp-binding-label-peer-state-attributes;
      } // peer
    } // mldp-binding-label-state-attributes

    /*
     * Configuration data and operational state data nodes
     */
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:global/ldp:capability" {
      description "Augmentation for MLDP global capability";
      container mldp {
        description
          "This container contains the configruation and state
           information for multipoint LDP capabilities.";
        uses mldp-capabilities;
      }
    }

    /*
     * Operational state data nodes
     */
```

```
  augment "/rt:routing/rt:control-plane-protocols/"
    + "ldp:mpls-ldp/ldp:peers/ldp:peer/ldp:received-peer-state/"
    + "ldp:capability" {
    description
      "Augmentation for MLDP received peer state capability";
    container mldp {
      description
        "Operational state information for the protocol capabilities
         of mLDP";

      container p2mp {
        description
          "Operational state information for the point-to-multipoint
           capability";
        leaf enable {
          type boolean;
          description
            "'true' to enable the point-to-multipoint capability";
        }
      }
      container mp2mp {
        description
          "Operational state information for the
           multipoint-to-multipoint capability";
        leaf enable {
          type boolean;
          description
            "'true' to enable the multipoint-to-multipoint
             capability";
        }
      }
      container make-before-break {
        description
          "Operational state information for the make-before-break
           capability";
        leaf enable {
          type boolean;
          description
            "'true' to enable the make-before-break capability";
        }
      }
    } // mldp
  }

  /*
   * Global augmentation
   */
  augment "/rt:routing/rt:control-plane-protocols/"
```

```
       + "ldp:mpls-ldp/ldp:global" {
       description "MLDP global augmentation.";
       container mldp {
         description
           "mLDP attributes at per instance level. Defining
            attributes here does not enable any MP capabilities.
            MP capabilities need to be explicitly enabled under
            container capability.";

         leaf enable {
           type boolean;
           description
             "'true' to enable mLDP";
         }

         container address-families {
           description
             "Per address family parameters";

           container ipv4 {
             description
                 "IPv4 information";
             container roots {
               config false;
               description
                 "IPv4 multicast LSP roots";
               list root {
                 key "root-address";
                 description
                   "List of roots for configured multicast LSPs";

                 leaf root-address {
                   type inet:ipv4-address;
                   description
                     "Root address.";
                 }

                 leaf is-self {
                   type boolean;
                   description
                     "I am the root node.";
                 }

                 list reachability {
                   key "address interface";
                   description
                     "A next-hop for reachability to root,
                      as a RIB view";
```

```
        leaf address {
          type inet:ipv4-address;
          description
            "The next-hop address to reach root";
        }
        leaf interface {
          type if:interface-ref;
          description
            "Interface connecting to next-hop";
        }
        leaf peer {
          type leafref {
            path
              "../../../../../../../ldp:peers/"
              + "ldp:peer/ldp:lsr-id";
          }
          description
            "LDP peer from which this next-hop can be
             reached";
        }
      }

      container bindings {
        description
          "mLDP FEC to label bindings";
        container opaque-element-lspid {
          description
            "The type of opaque value element is the generic
             LSP identifier";
          reference
            "RFC6388: Label Distribution Protocol
             Extensions for Point-to-Multipoint and
             Multipoint-to-Multipoint Label Switched
             Paths.";
          list fec-label {
            key
              "lsp-id";
            description
              "List of FEC to label bindings";
            leaf lsp-id {
              type uint32;
              description "ID to identify the LSP";
            }
            leaf multipoint-type {
              type multipoint-type;
              description
                "The type of mutipoint: p2mp or mp2mp";
            }
```

```
                    uses mldp-binding-label-state-attributes;
                  } // fec-label
                } // opaque-element-lspid
              } // bindings
            } // list root
          } // roots

          container configured-leaf-lsps {
            description
              "Configured multicast LSPs.";
            container opaque-element-lspid {
              description
                "The type of opaque value element is
                 the generic LSP identifier";
              reference
                "RFC6388: Label Distribution Protocol
                 Extensions for Point-to-Multipoint and
                 Multipoint-to-Multipoint Label Switched
                 Paths.";
              list fec-label {
                key
                  "root-address lsp-id";
                description
                  "List of FEC to label bindings.";
                leaf root-address {
                  type inet:ipv4-address;
                  description
                    "Root address";
                }
                leaf lsp-id {
                  type uint32;
                  description "ID to identify the LSP";
                }
                leaf multipoint-type {
                  type multipoint-type;
                  description
                    "The type of mutipoint: p2mp or mp2mp";
                }
              } // fec-label
            } // opaque-element-lspid
          } // configured-leaf-lsps
        } // ipv4
      } // list address-family
    } // mldp
  }

  /*
   * Notifications
```

```
     */
    notification mpls-mldp-fec-event {
      description
        "Notification event for a change of FEC status";
      leaf event-type {
        type ldp:oper-status-event-type;
        description "Event type";
      }
      choice opaque-element {
        description
          "The type of opaque value element";
        case opaque-element-lspid {
          container opaque-element-lspid {
            description
              "The type of opaque value element is
               the generic LSP identifier";
            reference
              "RFC6388: Label Distribution Protocol
               Extensions for Point-to-Multipoint and
               Multipoint-to-Multipoint Label Switched
               Paths.";
            leaf root-address {
              type inet:ip-address;
              description
                "Root address.";
            }
            leaf lsp-id {
              type uint32;
              description "ID to identify the LSP";
            }
            leaf multipoint-type {
              type multipoint-type;
              description
                "The type of mutipoint: p2mp or mp2mp";
            }
          } // container opaque-element-lspid
        }
      }
    }
  }
  <CODE ENDS>
```

Figure 14

9.2. Extended

This YANG module imports types defined in [RFC6991], [RFC8343], [RFC8349], [I-D.ietf-mpls-ldp-yang], and [RFC8294].

```
<CODE BEGINS> file "ietf-mpls-mldp-extended@2018-10-22.yang"
// RFC Editor: replace the above date with the date of
// publication and remove this note.

module ietf-mpls-mldp-extended {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-mpls-mldp-extended";
  prefix "mldp-ext";

  import ietf-inet-types {
    prefix "inet";
    reference "RFC 6991: Common YANG Data Types";
  }

  import ietf-interfaces {
    prefix "if";
    reference "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
       version)";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-mpls-ldp {
    prefix "ldp";
    reference "RFC XXXX: A YANG Data Model for MPLS LDP";
// RFC Editor: replace the XXXX with actual LDP YANG RFC number at
// time of publication and remove this note.
  }

  import ietf-mpls-ldp-extended {
```

```
       prefix "ldp-ext";
      reference "RFC XXXX: A YANG Data Model for MPLS LDP";
    // RFC Editor: replace the XXXX with actual LDP YANG RFC number at
    // time of publication and remove this note.
      }
      import ietf-mpls-mldp {
        prefix "mldp";
       reference "RFC XXXX: Base YANG Data Model for MPLS mLDP";
    // RFC Editor: replace the XXXX with actual mLDP YANG RFC number at
    // time of publication and remove this note.
      }

      organization
        "IETF MPLS Working Group";
      contact
        "WG Web:   <http://tools.ietf.org/wg/mpls/>
         WG List:  <mailto:mpls@ietf.org>

         Editor:   Kamran Raza
                   <mailto:skraza@cisco.com>

         Editor:   Sowmya Krishnaswamy
                   <mailto:krishnaswamy.sowmya@gmail.com>

         Editor:   Xufeng Liu
                   <mailto:xufeng.liu.ietf@gmail.com>

         Editor:   Santosh Esale
                   <mailto:sesale@juniper.net>

         Editor:   Loa Andersson
                   <mailto:loa@pi.nu>

         Editor:   Jeff Tantsura
                   <mailto:jefftant.ietf@gmail.com>";

      description
        "This YANG module defines the extended components for the
         management of Multi-Protocol Label Switching (MPLS) Multipoint
         LDP (mLDP).

         Copyright (c) 2018 IETF Trust and the persons identified as
         authors of the code.  All rights reserved.

         Redistribution and use in source and binary forms, with or
         without modification, is permitted pursuant to, and subject to
         the license terms contained in, the Simplified BSD License set
         forth in Section 4.c of the IETF Trust's Legal Provisions
```

```
        Relating to IETF Documents
        (http://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX; see the
        RFC itself for full legal notices.";

        // RFC Editor: replace XXXX with actual RFC number and remove
        // this note

      revision 2018-10-22 {
        // RFC Editor: replace the above date 2018-10-22 with the date of
        // publication and remove this note.

        description
          "Initial revision.";
        reference
          "RFC XXXX: Extended YANG Data Model for MPLS mLDP";
         // RFC Editor: replace XXXX with actual RFC number and remove
         // this note
      }

      /*
       * Features
       */
      feature capability-mldp-hsmp {
        description
          "This feature indicates that the system allows to configure
           mLDP hub-and-spoke-multipoint capability.";
      }

      feature capability-mldp-node-protection {
        description
          "This feature indicates that the system allows to configure
           mLDP node-protection capability.";
      }

      feature mldp-mofrr {
        description
          "This feature indicates that the system supports mLDP
           Multicast only FRR (MoFRR).";
      }

      feature per-peer-capability {
        description
          "This feature indicates that the system allows to configure
           mLDP capabilities at the per peer level.";
      }
```

```
    /*
     * Typedefs
     */
    typedef mofrr-role {
      type enumeration {
        enum none {
          description "MOFRR is not enabled.";
        }
        enum primary {
          description "This LSP is primary.";
        }
        enum backup {
          description "This LSP is backup.";
        }
      }
      description
        "This type represents the MOFRR (Multicast only FRR) role
         status of a LSP.";
    }

    /*
     * Groupings
     */
    grouping  mldp-ext-binding-label-state-attributes {
      description
        "mLDP label binding attributes";

      list peer {
        key "direction peer advertisement-type";
        description
          "List of advertised and received peers";
        uses mldp:mldp-binding-label-peer-state-attributes;

        leaf mofrr-role {
          when "../direction = 'upstream'" {
            description
              "For upstream.";
          }
          type mofrr-role;
          description
            "The MOFRR status of this LSP";
        }
      } // peer
    } // mldp-ext-binding-label-state-attributes

    grouping mldp-ext-capabilities {
      description
        "mLDP extended capabilities";
```

```
      container hub-and-spoke {
        if-feature capability-mldp-hsmp;
        description
          "Configure hub-and-spoke-multipoint capability";
        reference
          "RFC7140: LDP Extensions for Hub and Spoke Multipoint
           Label Switched Path";
        leaf enable {
          type boolean;
          description
            "Enable hub-and-spoke-multipoint";
        }
      }
      container node-protection {
        if-feature capability-mldp-node-protection;
        description
          "Configure node-protection capability.";
        reference
          "RFC7715: mLDP Node Protection.";
        leaf plr {
          type boolean;
          description
            "Point of Local Repair (PLR) capable for Multipoimt LSP
             node protection";
        }
        container merge-point {
          description
            "Merge Point capable for Multipoint LSP node protection";
          leaf enable {
            type boolean;
            description
              "Enable merge point capability";
          }
          leaf targeted-session-teardown-delay {
            type uint16;
            units seconds;
            description
              "Targeted session teardown delay";
          }
        } // merge-point
      }
    } // mldp-ext-capabilities

    grouping mldp-ext-per-af-config-attibutes {
      description
        "mLDP per address family configuration attibutes";
      container multicast-only-frr {
        if-feature mldp-mofrr;
```

```
        description
          "Multicast-only FRR (MoFRR) policy";
        leaf prefix-list {
          type ldp-ext:prefix-list-ref;
          description
            "Enables Multicast-only FRR (MoFRR) for the specified
             access list";
        }
      } // multicast-only-frr
      container recursive-fec {
        description
          "Recursive FEC policy";
        leaf prefix-list {
          type ldp-ext:prefix-list-ref;
          description
            "Enables recursive FEC for the specified prefix-list";
        }
      } // recursive-fec
    } // mldp-ext-per-af-config-attibutes

    grouping recursive-fec-attibutes {
      description
        "mLDP recursive FEC attibutes.";
      leaf recur-root-address {
        type inet:ip-address;
        description
          "Recursive root address";
        reference
          "RFC6512: Using Multipoint LDP When the
           Backbone Has No Route to the Root";
      }
      leaf recur-rd {
        type rt-types:route-distinguisher;
        description
          "Route Distinguisher in the VPN-Recursive
           Opaque Value";
        reference
          "RFC6512: Using Multipoint LDP When the
           Backbone Has No Route to the Root";
      }
      leaf multipoint-type {
        type mldp:multipoint-type;
        description
          "The type of mutipoint: p2mp or mp2mp";
      }
    } // recursive-fec-attibutes

    /*
```

```
     * Configuration data and operational state data nodes
     */
    // Global capability
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:global/ldp:capability/mldp:mldp" {
      description "Augmentation for MLDP global capability.";

      uses mldp-ext-capabilities;
    }

    // Peer capability
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:peers/ldp:peer/ldp:capability" {
      description "Augmentation for MLDP peer capability.";
      container mldp {
        if-feature per-peer-capability;
        description
          "mLDP capabilities";
        uses mldp:mldp-capabilities;
      }
    }

    // IPv4 config
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
      + "mldp:ipv4" {
      description "Augmentation for MLDP IPv4 configuration";
      uses mldp-ext-per-af-config-attibutes;
    }

    // IPv4 configured-leaf-lsps config
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
      + "mldp:ipv4/mldp:configured-leaf-lsps/"
      + "mldp:opaque-element-lspid/mldp:fec-label" {
      description
        "Augmentation for MLDP IPv4 configured-leaf-lsps
         configuration for opaque-element-lspid";
      list recursive-fec {
        key
          "recur-root-address recur-rd";
        description
          "List of recursive opaque values";
        uses recursive-fec-attibutes;
      } // fec-label
    }
```

```
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
      + "mldp:ipv4/mldp:configured-leaf-lsps" {
      description
        "Augmentation for MLDP IPv4 configured-leaf-lsps
         configuration";

      container opaque-element-transit {
        description
          "The type of opaque value element is the transit IPv4
           source.";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched Paths.";
        list fec-label {
          key
            "root-address source-address group-address " +
            "rd recur-root-address recur-rd";
          description
            "List of FEC to label bindings";
          leaf root-address {
            type inet:ipv4-address;
            description
              "Root address";
          }
          leaf source-address {
            type inet:ip-address;
            description
              "Source address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
              "Group address";
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
               Routing and Forwarding (VRF) Table
               Context.";
          }
          uses recursive-fec-attibutes;
        } // fec-label
```

```
      } // opaque-element-transit

      container opaque-element-bidir {
        description
          "The type of opaque value element is
           the generic LSP identifier";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched
           Paths.";
        list fec-label {
          key
            "root-address rp group-address rd recur-root-address "
          + "recur-rd";
          description
            "List of FEC to label bindings";
          leaf root-address {
            type inet:ipv4-address;
            description
              "Root address";
          }
          leaf rp {
            type inet:ip-address;
            description
              "Rendezvous-Point (RP) address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
              "Group address";
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
               Routing and Forwarding (VRF) Table
               Context.";
          }
          uses recursive-fec-attibutes;
        } // fec-label
      } // opaque-element-bidir
    }

    // IPv6 config
```

```
augment "/rt:routing/rt:control-plane-protocols/"
  + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
  + "ipv6" {
  description "Augmentation for MLDP IPv4 configuration";
  uses mldp-ext-per-af-config-attibutes;
}

// Global forwarding-nexthop
augment "/rt:routing/rt:control-plane-protocols/"
  + "ldp:mpls-ldp/ldp:global/ldp-ext:forwarding-nexthop/"
  + "ldp-ext:interfaces/ldp-ext:interface/ldp-ext:address-family" {
  description
    "Augmentation for MLDP nexthop forwarding interface";
  leaf mldp-disable {
    type boolean;
    description
      "Disable mLDP forwarding on this interface";
  }
}

/*
 * Operational state data nodes
 */
// IPv4 state for per peer bindings
augment "/rt:routing/rt:control-plane-protocols/"
  + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
  + "mldp:ipv4/mldp:roots/mldp:root/mldp:bindings/"
  + "mldp:opaque-element-lspid/mldp:fec-label/mldp:peer" {
  description "Augmentation for MLDP IPv4 state";

  leaf mofrr-role {
    when "../mldp:direction = 'upstream'" {
      description
        "For upstream";
    }
    type mofrr-role;
    description
      "The MOFRR status of this LSP";
  }
}

// Peer capability state
augment "/rt:routing/rt:control-plane-protocols/"
  + "ldp:mpls-ldp/ldp:peers/ldp:peer/ldp:received-peer-state/"
  + "ldp:capability/mldp:mldp" {
  description
    "Augmentation for MLDP received peer state capability.";
  container hub-and-spoke {
```

```
        description
          "Configure hub-and-spoke-multipoint capability.";
        reference
          "RFC7140: LDP Extensions for Hub and Spoke Multipoint
           Label Switched Path";
        leaf enable {
          type boolean;
          description
            "Enable hub-and-spoke-multipoint";
        }
      }
      container node-protection {
        description
          "Configure node-protection capability";
        reference
          "RFC7715: mLDP Node Protection.";
        leaf plr {
          type boolean;
          description
            "Point of Local Repair (PLR) capable for Multipoint LSP
             node protection";
        }
        leaf merge-point {
          type boolean;
          description
            "Merge Point capable for Multipoint LSP node protection";
        } // merge-point
      } // node-protection
    }

    // IPv4 bindings state
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
      + "mldp:ipv4/mldp:roots/mldp:root/mldp:bindings" {
      description "Augmentation for MLDP IPv4 bindings.";
      container opaque-element-transit {
        description
          "The type of opaque value element is the transit IPv4
           source.";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched Paths.";
        list fec-label {
          key
            "source-address group-address "
          + "rd recur-root-address recur-rd";
          description
```

```
            "List of FEC to label bindings";
          leaf source-address {
            type inet:ip-address;
            description
              "Source address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
              "Group address";
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
               Routing and Forwarding (VRF) Table
               Context.";
          }
          uses recursive-fec-attibutes;
          uses mldp-ext-binding-label-state-attributes;
        } // fec-label
      } // opaque-element-transit

      container opaque-element-bidir {
        description
          "The type of opaque value element is
           the generic LSP identifier.";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched
           Paths.";
        list fec-label {
          key
            "rp group-address rd recur-root-address recur-rd";
          description
            "List of FEC to label bindings";
          leaf rp {
            type inet:ip-address;
            description
              "Rendezvous Point (RP) address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
```

```
              "Group address";
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
               Routing and Forwarding (VRF) Table
               Context.";
          }
          uses recursive-fec-attibutes;
          uses mldp-ext-binding-label-state-attributes;
        } // fec-label
      } // opaque-element-bidir
    }

    // IPv6 bindings state
    augment "/rt:routing/rt:control-plane-protocols/"
      + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
      + "ipv6/roots/root/bindings" {
      description "Augmentation for MLDP IPv6 bindings.";
      container opaque-element-transit {
        config false;
        description
          "The type of opaque value element is the transit IPv6
           source.";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched
           Paths.";
        list fec-label {
          key
            "source-address group-address "
          + "rd recur-root-address recur-rd";
          description
            "List of FEC to label bindings";
          leaf source-address {
            type inet:ip-address;
            description
              "Source address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
              "Group address";
```

```
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
               Routing and Forwarding (VRF) Table
               Context.";
          }
          uses recursive-fec-attibutes;
          uses mldp-ext-binding-label-state-attributes;
        } // fec-label
      } // opaque-element-transit

      container opaque-element-bidir {
        config false;
        description
          "The type of opaque value element is
           the generic LSP identifier";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched
           Paths.";
        list fec-label {
          key
            "rp group-address rd recur-root-address recur-rd";
          description
            "List of FEC to label bindings";
          leaf rp {
            type inet:ip-address;
            description
              "Rendezvous Point (RP) address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
              "Group address";
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
```

```
                Routing and Forwarding (VRF) Table
                Context.";
            }
            uses recursive-fec-attibutes;
            uses mldp-ext-binding-label-state-attributes;
          } // fec-label
        } // opaque-element-bidir
      }

      // IPv4 bindings opaque-element-lspid state
      augment "/rt:routing/rt:control-plane-protocols/"
        + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
        + "mldp:ipv4/mldp:roots/mldp:root/mldp:bindings/"
        + "mldp:opaque-element-lspid/mldp:fec-label" {
        description
          "Augmentation for MLDP IPv4 bindings with opaque type LSP ID.";
        list recursive-fec {
          key
            "recur-root-address recur-rd";
          description
            "List of recursive opaque values";
          uses recursive-fec-attibutes;
          uses mldp-ext-binding-label-state-attributes;
        } // fec-label
      }

      // IPv6 bindings opaque-element-lspid state
      augment "/rt:routing/rt:control-plane-protocols/"
        + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families/"
        + "ipv6/roots/root/bindings/opaque-element-lspid/fec-label" {
        description
          "Augmentation for MLDP IPv6 bindings with opaque type LSP ID.";
        list recursive-fec {
          key "recur-root-address recur-rd";
          config false;
          description
            "List of recursive opaque values";
          uses recursive-fec-attibutes;
          uses mldp-ext-binding-label-state-attributes;
        } // fec-label
      }

      /*
       * Per AF augmentation
       */
      // IPv6 augmentation
      augment "/rt:routing/rt:control-plane-protocols/"
        + "ldp:mpls-ldp/ldp:global/mldp:mldp/mldp:address-families" {
```

```
      description "Augmentation for MLDP IPv6 address family.";
      container ipv6 {
        description
          "IPv6 information";

        container roots {
          config false;
          description
            "IPv6 multicast LSP roots";
          list root {
            key "root-address";
            description
              "List of roots for configured multicast LSPs";

            leaf root-address {
              type inet:ipv6-address;
              description
                "Root address";
            }

            leaf is-self {
              type boolean;
              description
                "This is the root";
            }

            list reachability {
              key "address interface";
              description
                "A next-hop for reachability to root,
                 as a RIB view";
              leaf address {
                type inet:ipv6-address;
                description
                  "The next-hop address to reach root";
              }
              leaf interface {
                type if:interface-ref;
                description
                  "Interface connecting to next-hop";
              }
              leaf peer {
                type leafref {
                  path
                    "../../../../../../../../ldp:peers/"
                    + "ldp:peer/ldp:lsr-id";
                }
                description
```

```
                "LDP peer from which this next-hop can be
                 reached";
            }
          }

          container bindings {
            description
              "mLDP FEC to label bindings";
            container opaque-element-lspid {
              description
                "The type of opaque value element is
                 the generic LSP identifier";
              reference
                "RFC6388: Label Distribution Protocol
                 Extensions for Point-to-Multipoint and
                 Multipoint-to-Multipoint Label Switched
                 Paths.";
              list fec-label {
                key
                  "lsp-id";
                description
                  "List of FEC to label bindings";
                leaf lsp-id {
                  type uint32;
                  description "ID to identify the LSP";
                }
                leaf multipoint-type {
                  type mldp:multipoint-type;
                  description
                    "The type of mutipoint: p2mp or mp2mp";
                }

                uses mldp-ext-binding-label-state-attributes;
              } // fec-label
            } // opaque-element-lspid
          } // bindings
        } // list root
      } // roots

      container configured-leaf-lsps {
        description
          "Configured multicast LSPs";

        container opaque-element-lspid {
          description
            "The type of opaque value element is
             the generic LSP identifier";
          reference
```

```
            "RFC6388: Label Distribution Protocol
             Extensions for Point-to-Multipoint and
             Multipoint-to-Multipoint Label Switched
             Paths.";
          list fec-label {
            key
              "root-address lsp-id";
            description
              "List of FEC to label bindings";
            leaf root-address {
              type inet:ipv6-address;
              description
                "Root address";
            }
            leaf lsp-id {
              type uint32;
              description "ID to identify the LSP";
            }
            leaf multipoint-type {
              type mldp:multipoint-type;
              description
                "The type of mutipoint: p2mp or mp2mp";
            }
            list recursive-fec {
              key
                "recur-root-address recur-rd";
              description
                "List of recursive opaque values";
              uses recursive-fec-attibutes;
            } // fec-label
          } // fec-label
        } // opaque-element-lspid

        container opaque-element-transit {
          description
            "The type of opaque value element is the transit IPv4
             source.";
          reference
            "RFC6826: Multipoint LDP In-Band Signaling for
             Point-to-Multipoint and
             Multipoint-to-Multipoint Label Switched Paths.";
          list fec-label {
            key
              "root-address source-address group-address "
              + "rd recur-root-address recur-rd";
            description
              "List of FEC to label bindings";
            leaf root-address {
```

```
            type inet:ipv6-address;
            description
              "Root address";
          }
          leaf source-address {
            type inet:ip-address;
            description
              "Source address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
              "Group address";
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
               Routing and Forwarding (VRF) Table
               Context.";
          }
          uses recursive-fec-attibutes;
        } // fec-label
      } // opaque-element-transit

      container opaque-element-bidir {
        description
          "The type of opaque value element is
           the generic LSP identifier";
        reference
          "RFC6826: Multipoint LDP In-Band Signaling for
           Point-to-Multipoint and
           Multipoint-to-Multipoint Label Switched
           Paths.";
        list fec-label {
          key
            "root-address rp group-address rd recur-root-address "
          + "recur-rd";
          description
            "List of FEC to label bindings.";
          leaf root-address {
            type inet:ipv6-address;
            description
              "Root address";
          }
```

```
            leaf rp {
              type inet:ip-address;
              description
                "Rendezvous Point (RP) address";
            }
            leaf group-address {
              type inet:ip-address-no-zone;
              description
                "Group address";
            }
            leaf rd {
              type rt-types:route-distinguisher;
              description
                "Route Distinguisher";
              reference
                "RFC7246: Multipoint Label Distribution
                 Protocol In-Band Signaling in a Virtual
                 Routing and Forwarding (VRF) Table
                 Context.";
            }
            uses recursive-fec-attibutes;
          } // fec-label
        } // opaque-element-bidir
      } // configured-leaf-lsps
    } // ipv6
  }

  /*
   * Global augmentation
   */
  /*
   * Notifications
   */
  augment "/mldp:mpls-mldp-fec-event/mldp:opaque-element/"
    + "mldp:opaque-element-lspid/mldp:opaque-element-lspid" {
    description
      "Augmentation for MLDP notification for opaque-element-lspid.";
    container recursive-fec {
      description
        "Container of recursive opaque values";
      uses recursive-fec-attibutes;
    } // fec-label
  }

  augment "/mldp:mpls-mldp-fec-event/mldp:opaque-element" {
    description
      "Augmentation for MLDP notification.";
    case opaque-element-transit {
```

```
          container opaque-element-transit {
            description
              "The type of opaque value element is the transit IPv4
               source.";
            reference
              "RFC6826: Multipoint LDP In-Band Signaling for
               Point-to-Multipoint and
               Multipoint-to-Multipoint Label Switched Paths.";
            leaf root-address {
              type inet:ip-address;
              description
                "Root address";
            }
            leaf source-address {
              type inet:ip-address;
              description
                "Source address";
            }
            leaf group-address {
              type inet:ip-address-no-zone;
              description
                "Group address";
            }
            leaf rd {
              type rt-types:route-distinguisher;
              description
                "Route Distinguisher";
              reference
                "RFC7246: Multipoint Label Distribution
                 Protocol In-Band Signaling in a Virtual
                 Routing and Forwarding (VRF) Table
                 Context.";
            }
            uses recursive-fec-attibutes;
          } // opaque-element-transit
        } // opaque-element-transit

        case opaque-element-bidir {
          container opaque-element-bidir {
            description
              "The type of opaque value element is
               the generic LSP identifier";
            reference
              "RFC6826: Multipoint LDP In-Band Signaling for
               Point-to-Multipoint and
               Multipoint-to-Multipoint Label Switched
               Paths.";
            leaf root-address {
```

```
            type inet:ip-address;
            description
              "Root address";
          }
          leaf rp {
            type inet:ip-address;
            description
              "Rendezvous Point (RP) address";
          }
          leaf group-address {
            type inet:ip-address-no-zone;
            description
              "Group address";
          }
          leaf rd {
            type rt-types:route-distinguisher;
            description
              "Route Distinguisher";
            reference
              "RFC7246: Multipoint Label Distribution
               Protocol In-Band Signaling in a Virtual
               Routing and Forwarding (VRF) Table
               Context.";
          }
          uses recursive-fec-attibutes;
        } // opaque-element-bidir
      } // opaque-element-bidir
    }
  }
  <CODE ENDS>
```

Figure 15

10. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes.

It goes without saying that this specification also inherits the security considerations captured in the actual protocol specification documents, namely base mLDP [RFC6388], targeted mLDP [RFC7060], mLDP Recursive FEC [RFC6512], Multicast-only FRR [RFC7431], mLDP Node Protection [RFC7715], mLDP In-band Signaling [RFC6826] [RFC7246] [RFC7438], and Hub-and-Spoke Multipoint LSPs [RFC7140].

11. IANA Considerations

This document requests the registration of the following URIs in the IETF "XML registry" [RFC3688]:

URI	Registrant	XML
urn:ietf:params:xml:ns:yang:ietf-mpls-mldp	The IESG	N/A
urn:ietf:params:xml:ns:yang:ietf-mpls-mldp-extended	The IESG	N/A

This document requests the registration of the following YANG modules in the "YANG Module Names" registry [RFC6020]:

Name	Namespace	Prefix	Reference
ietf-mpls-mldp	urn:ietf:params:xml:ns:yang:ietf-mpls-mldp	mldp	This document
ietf-mpls-mldp-extended	urn:ietf:params:xml:ns:yang:ietf-mpls-mldp-extended	mldp-ext	This document

12. Acknowledgments

The authors would like to acknowledge Ladislav Lhotka and Acee Lindem for their review and comments.

13. References

13.1. Normative References

[I-D.ietf-mpls-ldp-yang]
Raza, K., Asati, R., Liu, X., Esale, S., Chen, X., and H. Shah, "YANG Data Model for MPLS LDP", draft-ietf-mpls-ldp-yang-09 (work in progress), March 2020.

[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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011, <https://www.rfc-editor.org/info/rfc6388>.

[RFC6389] Aggarwal, R. and JL. Le Roux, "MPLS Upstream Label Assignment for LDP", RFC 6389, DOI 10.17487/RFC6389, November 2011, <https://www.rfc-editor.org/info/rfc6389>.

[RFC6512] Wijnands, IJ., Rosen, E., Napierala, M., and N. Leymann, "Using Multipoint LDP When the Backbone Has No Route to the Root", RFC 6512, DOI 10.17487/RFC6512, February 2012, <https://www.rfc-editor.org/info/rfc6512>.

[RFC6826] Wijnands, IJ., Ed., Eckert, T., Leymann, N., and M. Napierala, "Multipoint LDP In-Band Signaling for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6826, DOI 10.17487/RFC6826, January 2013, <https://www.rfc-editor.org/info/rfc6826>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7060] Napierala, M., Rosen, E., and IJ. Wijnands, "Using LDP Multipoint Extensions on Targeted LDP Sessions", RFC 7060, DOI 10.17487/RFC7060, November 2013, <https://www.rfc-editor.org/info/rfc7060>.

[RFC7140] Jin, L., Jounay, F., Wijnands, IJ., and N. Leymann, "LDP Extensions for Hub and Spoke Multipoint Label Switched Path", RFC 7140, DOI 10.17487/RFC7140, March 2014, <https://www.rfc-editor.org/info/rfc7140>.

[RFC7246] Wijnands, IJ., Ed., Hitchen, P., Leymann, N., Henderickx, W., Gulko, A., and J. Tantsura, "Multipoint Label Distribution Protocol In-Band Signaling in a Virtual Routing and Forwarding (VRF) Table Context", RFC 7246, DOI 10.17487/RFC7246, June 2014, <https://www.rfc-editor.org/info/rfc7246>.

[RFC7431] Karan, A., Filsfils, C., Wijnands, IJ., Ed., and B. Decraene, "Multicast-Only Fast Reroute", RFC 7431, DOI 10.17487/RFC7431, August 2015, <https://www.rfc-editor.org/info/rfc7431>.

[RFC7438] Wijnands, IJ., Ed., Rosen, E., Gulko, A., Joorde, U., and J. Tantsura, "Multipoint LDP (mLDP) In-Band Signaling with Wildcards", RFC 7438, DOI 10.17487/RFC7438, January 2015, <https://www.rfc-editor.org/info/rfc7438>.

[RFC7715] Wijnands, IJ., Ed., Raza, K., Atlas, A., Tantsura, J., and Q. Zhao, "Multipoint LDP (mLDP) Node Protection", RFC 7715, DOI 10.17487/RFC7715, January 2016, <https://www.rfc-editor.org/info/rfc7715>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[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>.

[RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <https://www.rfc-editor.org/info/rfc8294>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

[RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <https://www.rfc-editor.org/info/rfc8349>.

[RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <https://www.rfc-editor.org/info/rfc8407>.

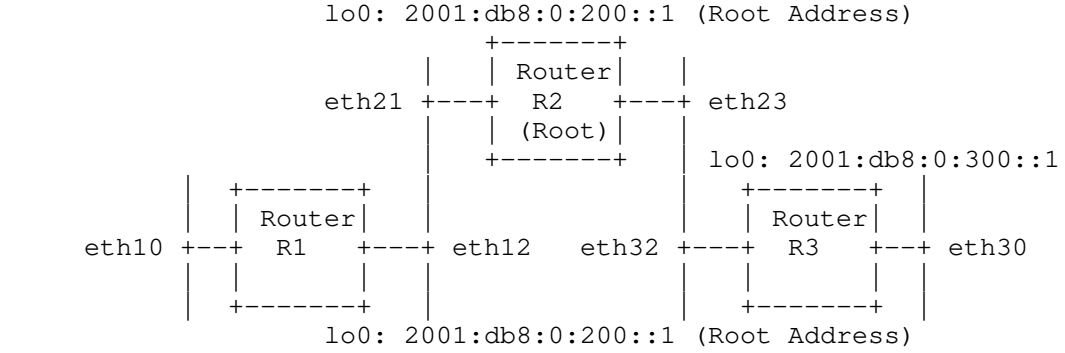
[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

13.2. Informative References

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

Appendix A. Data Tree Example

This section contains an example of an instance data tree in the JSON encoding [RFC7951], containing both configuration and state data.



The configuration instance data tree for Router R3 in the above figure could be as follows:

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "lo0",
        "description": "R3 loopback interface.",
        "type": "iana-if-type:softwareLoopback",
        "ietf-ip:ipv6": {
          "address": [
            {
              "ip": "2001:db8:0:300::1",
              "prefix-length": 64
            }
          ]
        }
      },
      {
        "name": "eth30",
        "description": "An interface connected to client routers.",
```

```
          "type": "iana-if-type:ethernetCsmacd",
          "ietf-ip:ipv6": {
            "forwarding": true
          }
        },
        {
          "name": "eth32",
          "description": "An interface connected to root (R2).",
          "type": "iana-if-type:ethernetCsmacd",
          "ietf-ip:ipv6": {
            "forwarding": true
          }
        }
      ]
    },
    "ietf-routing:routing": {
      "router-id": "203.0.113.3",
      "control-plane-protocols": {
        "ietf-mpls-ldp:mpls-ldp": {
          "global": {
            "address-families": {
              "ietf-mpls-ldp-extended:ipv6": {
                "enable": true
              }
            },
            "capability": {
              "ietf-mpls-mldp:mldp": {
                "mp2mp": {
                  "enable": true
                }
              }
            },
            "ietf-mpls-mldp:mldp": {
              "enable": true,
              "address-families": {
                "ietf-mpls-mldp-extended:ipv6": {
                  "configured-leaf-lsps": {
                    "opaque-element-lspid": {
                      "fec-label": [
                        {
                          "root-address": "2001:db8:0:200::1",
                          "lsp-id": 201,
                          "multipoint-type": "mp2mp"
                        }
                      ]
                    }
                  }
                }
```

```
              }
            }
          },
          "discovery": {
            "interfaces": {
              "interface": [
                {
                  "name": "eth30",
                  "address-families": {
                    "ietf-mpls-ldp-extended:ipv6": {
                      "enable": true
                    }
                  }
                },
                {
                  "name": "eth32",
                  "address-families": {
                    "ietf-mpls-ldp-extended:ipv6": {
                      "enable": true
                    }
                  }
                }
              ]
            }
          }
        }
      }
    }
```

The cooresponding operational state data for Router R3 could be as follows:

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "lo0",
        "description": "R3 loopback interface.",
        "type": "iana-if-type:softwareLoopback",
        "phys-address": "00:00:5e:00:53:03",
        "oper-status": "up",
        "statistics": {
          "discontinuity-time": "2018-10-15T12:34:56-05:00"
        },
        "ietf-ip:ipv6": {
          "mtu": 1500,
          "address": [
```

```
            {
              "ip": "2001:db8:0:300::1",
              "prefix-length": 64,
              "origin": "static",
              "status": "preferred"
            },
            {
              "ip": "fe80::200:5eff:fe00:5303",
              "prefix-length": 64,
              "origin": "link-layer",
              "status": "preferred"
            }
          ],
          "neighbor": [
          ]
        }
      },
      {
        "name": "eth30",
        "description": "An interface connected to client routers.",
        "type": "iana-if-type:ethernetCsmacd",
        "phys-address": "00:00:5e:00:53:30",
        "oper-status": "up",
        "statistics": {
          "discontinuity-time": "2018-10-15T12:34:56-05:00"
        },
        "ietf-ip:ipv6": {
          "forwarding": true,
          "mtu": 1500,
          "address": [
            {
              "ip": "fe80::200:5eff:fe00:5330",
              "prefix-length": 64,
              "origin": "link-layer",
              "status": "preferred"
            }
          ],
          "neighbor": [
          ]
        }
      },
      {
        "name": "eth32",
        "description": "An interface connected to root (R2).",
        "type": "iana-if-type:ethernetCsmacd",
        "phys-address": "00:00:5e:00:53:32",
        "oper-status": "up",
        "statistics": {
```

```
            "discontinuity-time": "2018-10-15T12:34:56-05:00"
          },
          "ietf-ip:ipv6": {
            "forwarding": true,
            "mtu": 1500,
            "address": [
              {
                "ip": "fe80::200:5eff:fe00:5332",
                "prefix-length": 64,
                "origin": "link-layer",
                "status": "preferred"
              }
            ],
            "neighbor": [
              {
                "ip": "fe80::200:5eff:fe00:5323",
                "link-layer-address": "00:00:5e:00:53:23",
                "origin": "dynamic",
                "is-router": [null],
                "state": "reachable"
              }
            ]
          }
        }
      ]
    },
    "ietf-routing:routing": {
      "router-id": "203.0.113.3",
      "interfaces": {
        "interface": [
          "lo0",
          "eth30",
          "eth32"
        ]
      },
      "control-plane-protocols": {
        "ietf-mpls-ldp:mpls-ldp": {
          "global": {
            "address-families": {
              "ietf-mpls-ldp-extended:ipv6": {
                "enable": true
              }
            },
            "capability": {
              "ietf-mpls-mldp:mldp": {
                "mp2mp": {
                  "enable": true
                }
```

```
            }
          },
          "ietf-mpls-mldp:mldp": {
            "enable": true,
            "address-families": {
              "ietf-mpls-mldp-extended:ipv6": {
                "configured-leaf-lsps": {
                  "opaque-element-lspid": {
                    "fec-label": [
                      {
                        "root-address": "2001:db8:0:200::1",
                        "lsp-id": 201,
                        "multipoint-type": "mp2mp"
                      }
                    ]
                  }
                },
                "roots": {
                  "root": [
                    {
                      "root-address": "2001:db8:0:200::1",
                      "is-self": false,
                      "reachability": [
                        {
                          "address": "fe80::200:5eff:fe00:5323",
                          "interface": "eth32",
                          "peer": "203.0.113.2"
                        }
                      ],
                      "bindings": {
                        "opaque-element-lspid": {
                          "fec-label": [
                            {
                              "lsp-id": 201,
                              "multipoint-type": "mp2mp",
                              "peer": [
                                {
                                  "direction": "upstream",
                                  "peer": "203.0.113.2",
                                  "advertisement-type": "advertised",
                                  "label": 3201
                                },
                                {
                                  "direction": "upstream",
                                  "peer": "203.0.113.2",
                                  "advertisement-type": "received",
                                  "label": 2301
                                }
```

```
                                ]
                              }
                            ]
                          }
                        }
                      }
                    ]
                  }
                }
              }
            }
          },
          "discovery": {
            "interfaces": {
              "interface": [
                {
                  "name": "eth30",
                  "address-families": {
                    "ietf-mpls-ldp-extended:ipv6": {
                      "enable": true,
                      "hello-adjacencies": {
                        "hello-adjacency": [
                        ]
                      }
                    }
                  }
                },
                {
                  "name": "eth32",
                  "address-families": {
                    "ietf-mpls-ldp-extended:ipv6": {
                      "enable": true,
                      "hello-adjacencies": {
                        "hello-adjacency": [
                          {
                            "adjacent-address":
                              "fe80::200:5eff:fe00:5323",
                            "flag": ["adjacency-flag-active"],
                            "hello-holdtime": {
                              "adjacent": 15,
                              "negotiated": 15,
                              "remaining": 9
                            },
                            "next-hello": 3,
                            "statistics": {
                              "discontinuity-time":
                                "2018-10-15T12:34:56-05:00"
                            },
```

```
                            "peer": {
                              "lsr-id": "203.0.113.2",
                              "label-space-id": 0
                            }
                          }
                        ]
                      }
                    }
                  }
                }
              ]
            }
          },
          "peers": {
            "peer": [
              {
                "lsr-id": "203.0.113.2",
                "label-space-id": 0,
                "label-advertisement-mode": {
                  "local": "downstream-unsolicited",
                  "peer": "downstream-unsolicited",
                  "negotiated": "downstream-unsolicited"
                },
                "next-keep-alive": 5,
                "session-holdtime": {
                  "peer": 180,
                  "negotiated": 180,
                  "remaining": 78
                },
                "session-state": "operational",
                "tcp-connection": {
                  "local-address": "fe80::200:5eff:fe00:5332",
                  "local-port": 646,
                  "remote-address": "fe80::200:5eff:fe00:5323",
                  "remote-port": 646
                },
                "up-time": "P2H33M5S",
                "statistics": {
                  "discontinuity-time": "2018-10-15T12:34:56-05:00"
                },
                "received-peer-state": {
                  "capability": {
                    "ietf-mpls-mldp:mldp": {
                      "mp2mp": {
                        "enable": true
                      }
                    }
                  }
```

```
                }
              }
            ]
          }
        }
      }
    }
  }
```

Appendix B. Additional Contributors

Matthew Bocci
Nokia
Email: matthew.bocci@nokia.com

Authors' Addresses

Kamran Raza
Cisco Systems
Email: skraza@cisco.com

Xufeng Liu
Volta Networks
Email: xufeng.liu.ietf@gmail.com

Santosh Esale
Juniper Networks
Email: santosh_easale@berkeley.edu

Loa Andersson
Huawei Technologies
Email: loa@pi.nu

Jeff Tantsura
Nuage Networks
Email: jefftant.ietf@gmail.com

Sowmya Krishnaswamy
Individual
Email: krishnaswamy.sowmya@gmail.com

Rajiv Asati
Cisco Systems, Inc.
Email: rajiva@cisco.com

Xia Chen
Huawei Technologies
Email: jescia.chenxia@huawei.com

Himanshu Shah
Ciena Corporation
Email: hshah@ciena.com

MPLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 24, 2020

G. Mirsky
ZTE Corp.

October 22, 2019

Operations, Administration, and Maintenance for MPLS-SR over IP
draft-mirsky-mpls-oam-mpls-sr-ip-03

Abstract

Segment routing uses source routing paradigm to traffic engineering by specifying segments a packet traverses through the network. MPLS Segment Routing applies that paradigm to an MPLS data plane-based networks. SR-MPLS over IP uses MPLS label stack as a source routing instruction set and uses IP encapsulation/tunneling such as MPLS-in-UDP as defined in RFC 7510 to realize a source routing mechanism across MPLS, IPv4, and IPv6 data planes. This document describes Operations, Administration, and Maintenance operations in SR-MPLS over IP environment.

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 April 24, 2020.

Copyright Notice

Table of Contents

1. Introduction

Segment routing [RFC8402] uses source routing paradigm to traffic engineering by specifying segments a packet traverses through the network. MPLS Segment Routing (SR-MPLS) [I-D.ietf-spring-segment-routing-mpls] applies that paradigm to an MPLS data plane-based networks. SR-MPLS over IP uses MPLS label stack as a source routing instruction set and uses IP encapsulation/ tunneling such as MPLS-in-UDP as defined in [RFC7510] to realize a source routing mechanism across MPLS, IPv4, and IPv6 data planes. This document describes Operations, Administration, and Maintenance (OAM) operations in SR-MPLS over IP environment.

2. Conventions used in this document

2.1. Terminology

MPLS: Multiprotocol Label Switching

LSP: Label Switched Path

BFD: Bidirectional Forwarding Detection

SR Segment Routing

SR-MPLS Segment Routing in MPLS data plane

FEC: Forwarding Equivalence Class

G-ACh: Generic Associated Channel

ACH: Associated Channel Header

GAL: G-ACh Label

OAM Operations, Administration, and Maintenance

2.2. Requirements Language

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

3. OAM in SR-MPLS over IP

OAM operations support Fault Management and Performance Monitoring components of FCAPS framework for network management. To achieve its objectives, Fault Management OAM includes proactive and on-demand protocols to provide constant monitoring of the network to detect the failure in combination with on-demand tools to efficiently localize and characterize the defect. Performance Monitoring OAM protocols support measurement of packet loss and packet delay that enables calculation of performance metrics, e.g., packet loss ration, inter-packet delay variation, that are useful in monitoring the quality of service in the network, detect and quantify the service degradation.

3.1. Fault Management OAM in SR-MPLS over IP

Fault management OAM toolset includes protocols to perform on-demand failure detection and localization as well as proactively monitor path continuity. An example of the former is echo request/reply, e.g., Label Switched Path (LSP) Ping [RFC8029]. An example of the latter - Bidirectional Forwarding Detection (BFD) over MPLS LSP [RFC5884]. For SR-MPLS environment applicability and use of these OAM tools defined in [RFC8287] and [I-D.mirsky-spring-bfd] respectively. Both LSP Ping and BFD can be used either with IP/UDP encapsulation or in Generic Associated Channel (G-ACh) [RFC5586]. The use of IP/UDP encapsulation is well-understood and has been defined in [RFC8029]:

> The IP header is set as follows: the source IP address is a routable address of the sender; the destination IP address is a (randomly chosen) IPv4 address from the range 127/8 or an IPv6

address from the range 0:0:0:0:0:FFFF:7F00:0/104. The IP TTL is set to 1. The source UDP port is chosen by the sender.

Using the sender's routable address enables the receiver to send an echo reply or BFD control packets over the IP network. In some environments, the overhead of extra IP/UDP encapsulations may be considered as overburden and make to use more compact G-ACh encapsulation instead. In such a case, the OAM control packet MUST be immediately followed by the IP Address TLV [I-D.mirsky-mpls-p2mp-bfd] with its Value field containing one of the routable IP addresses of the sender.

3.2. Performance Monitoring OAM in SR-MPLS over IP

Performance monitoring in SR-MPLS over IP may be performed using mechanisms defined in [RFC6374]. Unlike FM OAM protocols for MPLS, [RFC6374] does not define the use of IP encapsulation. Instead, the addressing object of the type Return Address MUST be used in two-way measurements or queries.

4. Security Considerations

This document does not introduce new security aspects but inherits all security considerations from [RFC8287], [RFC8029], [RFC5884], [I-D.mirsky-spring-bfd].

5. IANA Considerations

5.1. Source MEP ID IP Address Type

TBD.

6. Acknowledgements

TBD

7. Normative References

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-22 (work in progress), May 2019.

[I-D.mirsky-mpls-p2mp-bfd]
Mirsky, G., "BFD for Multipoint Networks over Point-to-Multi-Point MPLS LSP", draft-mirsky-mpls-p2mp-bfd-07 (work in progress), June 2019.

[I-D.mirsky-spring-bfd]
Mirsky, G., Tantsura, J., Varlashkin, I., and M. Chen, "Bidirectional Forwarding Detection (BFD) in Segment Routing Networks Using MPLS Dataplane", draft-mirsky-spring-bfd-08 (work in progress), August 2019.

[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>.

[RFC5884] Aggarwal, R., Kompella, K., Nadeau, T., and G. Swallow, "Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs)", RFC 5884, DOI 10.17487/RFC5884, June 2010, <https://www.rfc-editor.org/info/rfc5884>.

[RFC6374] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, DOI 10.17487/RFC6374, September 2011, <https://www.rfc-editor.org/info/rfc6374>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <https://www.rfc-editor.org/info/rfc7510>.

[RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <https://www.rfc-editor.org/info/rfc8029>.

[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>.

[RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <https://www.rfc-editor.org/info/rfc8287>.

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

Author's Address

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Network Work group
Internet-Draft
Intended status: Standards Track
Expires: April 26, 2019

N. Kumar, Ed.
C. Pignataro, Ed.
F. Iqbal
Z. Ali
Cisco

October 23, 2018

Label Switched Path (LSP) Ping/Traceroute for Segment Routing SIDs with MPLS Data-plane
draft-nainar-mpls-spring-lsp-ping-sids-00

Abstract

RFC8402 introduces Segment Routing architecture that leverages source routing and tunneling paradigms and can be directly applied to the Multi Protocol Label Switching (MPLS) data plane. A node steers a packet through a controlled set of instructions called segments, by prepending the packet with Segment Routing header. SR architecture defines different types of segments with different forwarding semantics associated.

RFC8287 defines the extensions to MPLS LSP Ping and Traceroute for Segment Routing IGP-Prefix and IGP-Adjacency Segment Identifier (SIDs) with an MPLS data plane. RFC8287 defines the Target FEC Stack Sub-TLVs and the procedures to apply RFC8029 on SR architecture with MPLS data plane.

This document defines the Target FEC Stack Sub-TLVs and the extension required for other SR Segments.

Table of Contents

1. Introduction

[RFC8402] introduces and describes a Segment Routing architecture that leverages the source routing and tunneling paradigms. A node steers a packet through a controlled set of instructions called segments, by prepending the packet with Segment Routing header. A detailed definition of the Segment Routing architecture is available in [RFC8402]

As described in [RFC8402] and [I-D.ietf-spring-segment-routing-mpls], the Segment Routing architecture can be directly applied to an MPLS data plane, the Segment identifier (Segment ID) will be of 20-bits size and the Segment Routing header is the label stack.

[RFC8287] defines the mechanism to perform LSP Ping and Traceroute for Segment Routing with MPLS data plane. [RFC8287] defines the Target FEC Stack Sub-TLVs for IGP-Prefix Segment ID and IGP-Adjacency Segment ID.

There are various other Segment IDs proposed by different documents that are applicable for SR architecture.
[I-D.ietf-idr-bgp-prefix-sid] defines BGP Prefix Segment ID,
[I-D.ietf-idr-bgpls-segment-routing-epe] defines BGP Peering Segment ID such as Peer Node SID, Peer Adj SID and Peer Set SID.
[I-D.sivabalan-pce-binding-label-sid] defines Path Binding Segment ID.

As above Segment IDs get deployed in the field, operators require corresponding MPLS OAM procedures for the SIDs. This document describes the target FEC Stack Sub-TLVs and the procedure to use LSP Ping and Traceroute for the above defined Segment IDs to support path validation and fault isolation.

2. Requirements notation

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].

3. Terminology

This document uses the terminologies defined in [RFC8402], [RFC8029], readers are expected to be familiar with it.

The term "BGP EPE node" is used to refer to node assigning and advertising BGP Peering Segment SIDs to steer traffic towards a BGP peer, as described in [I-D.ietf-spring-segment-routing-central-epe].

4. Segment ID sub-TLV

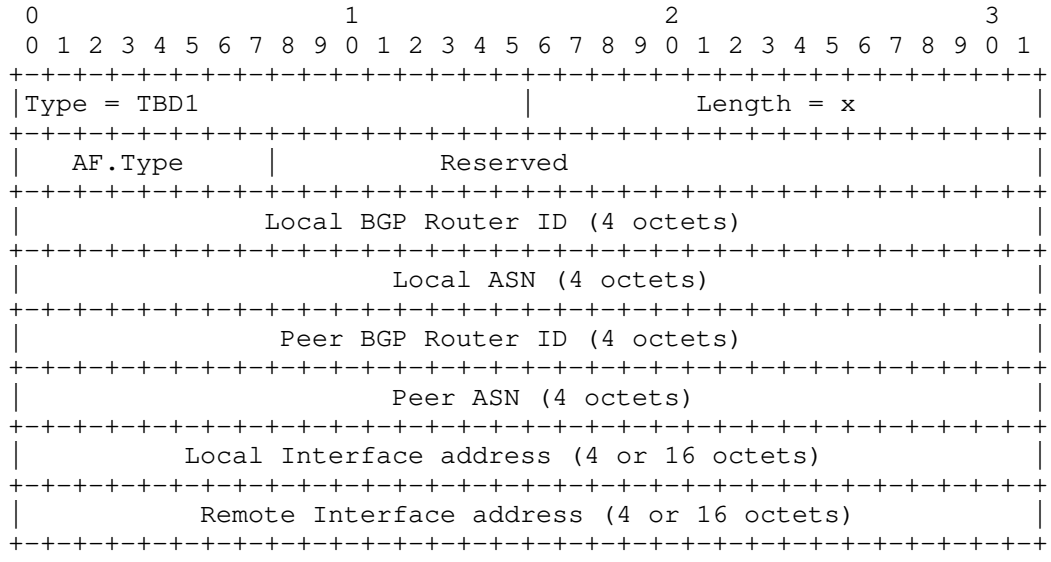
As defined in Section 5 of [RFC8287], the format of the following Segment ID sub-TLVs defined in this document follows the philosophy of Target FEC Stack TLV carrying FECs corresponding to each label in the label stack.

4.1. BGP Prefix Segment ID

Section 3.2.13 and 3.2.14 of [RFC8029] defines the Sub-TLV for BGP labeled IPv4 and IPv6 prefix respectively. This document proposes the use of the same Sub-TLV for IPv4 and IPv6 BGP Prefix SID without any change.

4.2. BGP Peering Segment - Peer-Node-SID

Peer-Node-SID identifies the peer node in the BGP Peering Segment. The sub-TLV format for Peer-Node-SID of BGP Peering Segment MUST be set as shown in the below TLV format:



AF.Type

Set to 4 if the address in Local/Remote Interface address field is IPv4 and set to 6 if the address in Local/Remote Interface address field is IPv6.

Reserved

MUST be set to 0 on send and MUST be ignored on receipt.

Local BGP Router ID

4-octet BGP Router ID of the node that assigns the Peer-Node-SID.

Local ASN

4-octet local ASN number of the node that assigns the Peer-Node-SID.

Peer BGP Router ID

4-octet BGP Router ID of the peer node.

Peer ASN

4-octet ASN number of the peer node.

Local Interface Address

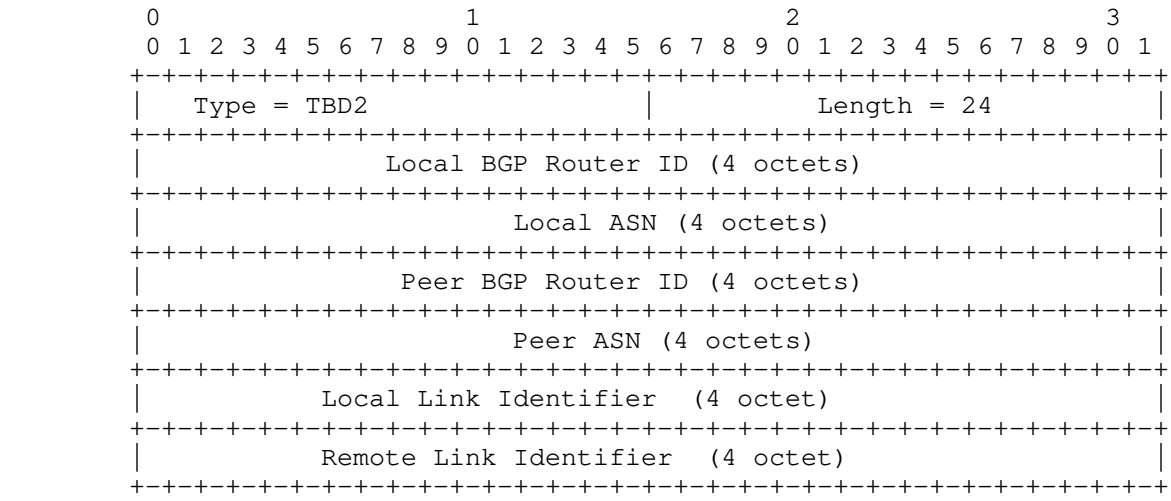
Set to the address used by the local node for BGP session peering. When AF.Type is set to 4, this address is 4-octet IPv4 address and when AF.Type is set to 6, this address is 16-octet IPv6 address.

Remote Interface Address

Set to the address used by the peer node for BGP session peering. When AF.Type is set to 4, this address is 4-octet IPv4 address and when AF.Type is set to 6, this address is 16-octet IPv6 address.

4.3. BGP Peering Segment - Peer-Adj-SID

Peer-Adj-SID identifies the underlying link to the BGP peer node. The sub-TLV format for Peer-Adj-SID of BGP Peering Segment MUST be set as shown in the below TLV format:



Local BGP Router ID

4-octet BGP Router ID of the node that assigns the Peer-Node-SID.

Local ASN

4-octet local ASN number of the node that assigns the Peer-Node-SID.

Peer BGP Router ID

4-octet BGP Router ID of the peer node.

Peer ASN

4-octet ASN number of the peer node.

Local Link Identifier

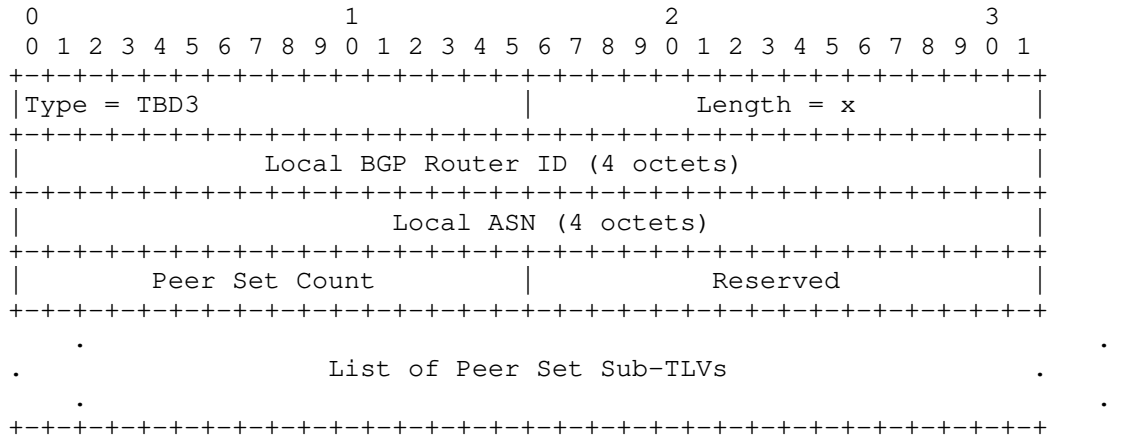
Set to 4-octet link identifier of the local interface to which Peer-Adj-SID is assigned to.

Remote Link Identifier

Set to 4-octet link identifier of the peer interface to which Peer-Adj-SID is assigned to. Set to all-zeros when this identifier is unknown.

4.4. BGP Peering Segment - Peer-Set-SID

The sub-TLV format for Peer-Node-SID of BGP Peering Segment MUST be set as shown in the below TLV format:



Local BGP Router ID

4-octet BGP Router ID of the node that assigns the Peer-Set-SID.

Local ASN

4-octet local ASN number of the node that assigns the Peer-Set-SID.

Peer Set Count

Set to the number of Peer Sub-TLVs included.

Sub-TLV Length

Total length in octets of the sub-TLVs associated with this TLV.

Peer Set Sub-TLV

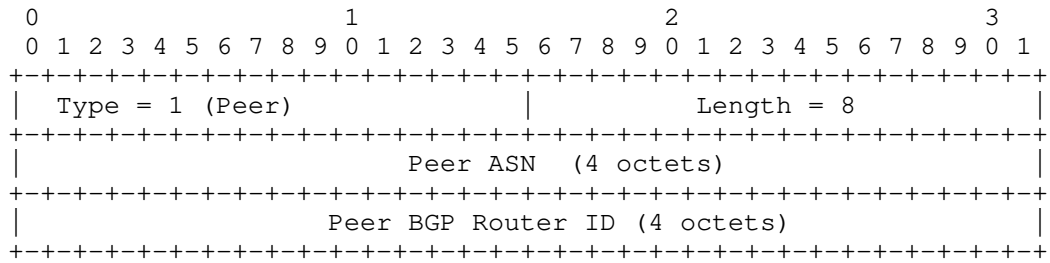
Carries the Sub-TLVs defined in section 4.4.1.

4.4.1. Peer Set Sub-TLV

As defined in section 5.3 of [I-D.ietf-idr-bgpls-segment-routing-epe], Peer-Set-SID can identify the set where the members can be Peer-Node or Peer-Adj from same or different ASN. The format of the Peer Set Sub-TLV will identify each such member.

4.4.1.1. Peer Node

The format for this sub-TLV MUST be set as below:



Peer ASN

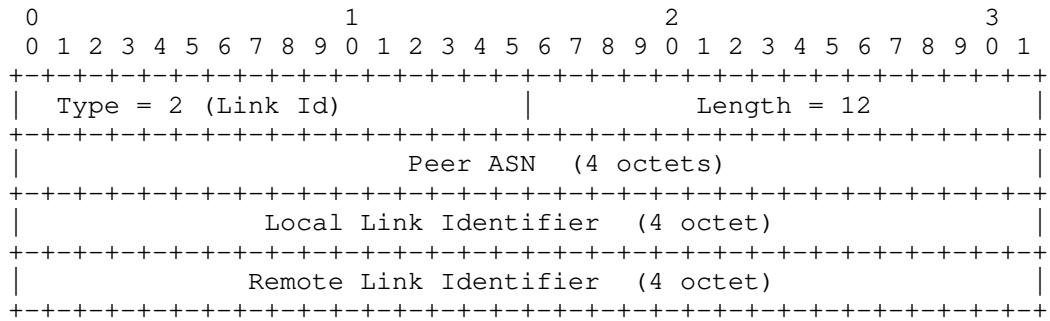
4-octet ASN number of the peer node.

Peer Router ID

4-octet BGP Router ID of the peer node.

4.4.1.2. Link Identifier

The format for this sub-TLV is as below:



Peer ASN

4-octet ASN number of the peer node.

Local Link Identifier

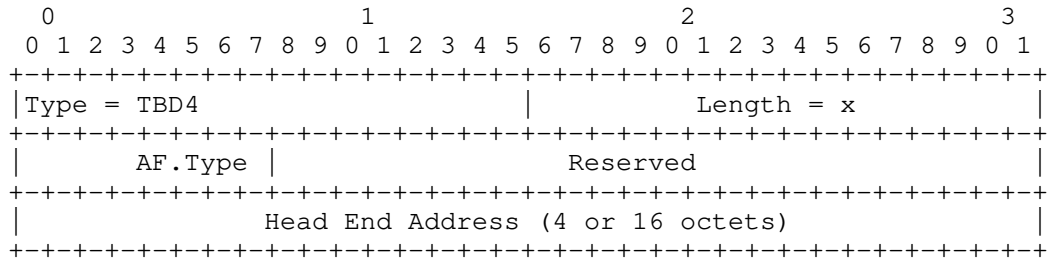
Set to 4-octet link identifier of the local interface to which Peer-Adj-SID is assigned to.

Remote Link Identifier

Set to 4-octet link identifier of the peer interface to which Peer-Adj-SID is assigned to. Set to all-zeros when this identifier is unknown.

4.5. Path Binding SID

Path Binding SID identifies the Binding Segment Identifier associated with an RSVP-TE or SR-TE path. The format for this sub-TLV is as below:



AF.Type

Set to 4 if the address in Head End Address field is IPv4 and set to 6 if the address in Head End address field is IPv6.

Reserved

MUST be set to 0 on send and MUST be ignored on receipt.

Head End Address

Set to the address of the head end node to which the policy is assigned. When AF.Type is 4, this address is IPv4 and when AF.Type is 6, it is IPv6.

4.6. Multicast Replication

[I-D.voyer-spring-sr-p2mp-policy] describes Segment Routing Multicast Replication Policy and introduces the notion of Tree SID to achieve this. A future version of this document will describe LSP Ping and Traceroute Target FEC Stack sub-TLV and procedures for Tree SID validation.

5. Procedures

This section describes the aspects of LSP Ping and Traceroute operations that require further considerations beyond [RFC8029] and [RFC8287].

5.1. BGP Prefix SID

The procedures described in [RFC8029] are sufficient for MPLS Ping and Traceroute operations for BGP Prefix SID using the FEC definitions from Section 3.2.13 and 3.2.14 of [RFC8029].

5.2. BGP Peering Segment Sub-TLVs

BGP Peering Segment sub-TLVs (BGP-Node-SID, BGP-Adj-SID, Peer-Set-SID) are assigned by BGP EPE node for a particular BGP neighbor, and advertised to the peer nodes. Any LSP Ping and Traceroute operation MUST be performed on the BGP EPE node, and not the remote neighbor node, as only the BGP EPE node can validate the contents of BGP Peering Segment sub-TLVs. Additionally, leaking the echo packet to the peer node may not be desirable for network operators.

5.2.1. Initiator Node Procedures

If the bottom-most label in the label stack is BGP Peer Segment label, the initiating node MUST set the TTL of the bottom-most label to 1 to ensure that MPLS TTL expires at the BGP EPE node, and the

echo packet does not leak to the BGP peer node. Echo packet MUST include one of BGP-Node-SID, BGP-Adj-SID, or Peer-Set-SID sub-TLV in the Target FEC Stack TLV corresponding to the BGP Peer Segment label. Operator MAY push one or more transport labels on top of the BGP Peer Segment label to forward the echo packet to the BGP EPE node.

5.2.2. Responder Node Procedures

In addition to procedures defined in [RFC8029], the responding node, upon TTL expiry of the echo packet, MUST process the incoming BGP Peer Segment sub-TLV of the Target FEC Stack. It MUST validate that contents of the sub-TLV and ensure the incoming label is advertised for the processed BGP Peer Segment sub-TLV.

5.3. Path Binding SID

5.3.1. Initiator Node Procedures

Similar to BGP Peering Segment sub-TLVs, Path Binding SID sub-TLV MUST be validated at the node assigning and advertising the Binding SID, instead of the endpoint of the path associated with the Binding SID. The initiating node MUST set the TTL of the Binding SID label to 1 and include the associated Path Binding SID TLV in the Target FEC Stack TLV of the echo request. Operator MAY push one or more transport labels on top of Binding SID label to forward echo packet from initiating node to the assigning node.

5.3.2. Responder Node Procedures

In addition to procedures defined in [RFC8029], the responding node, upon TTL expiry of the echo packet, MUST process the incoming Path Binding SID sub-TLV of the Target FEC Stack. The responding node MUST ensure that it is the advertising node specified in the Path Bindng SID sub-TLV, and the incoming Binding SID label matches the advertised label value.

6. IANA Considerations

To be Updated.

7. Security Considerations

To be Updated

8. Acknowledgement

TBD

9. Contributors

TBD

10. References

10.1. Normative References

[I-D.ietf-idr-bgp-prefix-sid]
Previdi, S., Filsfils, C., Lindem, A., Sreekantiah, A., and H. Gredler, "Segment Routing Prefix SID extensions for BGP", draft-ietf-idr-bgp-prefix-sid-27 (work in progress), June 2018.

[I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-15 (work in progress), March 2018.

[I-D.sivabalan-pce-binding-label-sid]
Sivabalan, S., Tantsura, J., Filsfils, C., Previdi, S., Hardwick, J., and D. Dhody, "Carrying Binding Label/ Segment-ID in PCE-based Networks.", draft-sivabalan-pce-binding-label-sid-04 (work in progress), March 2018.

[I-D.voyer-spring-sr-p2mp-policy]
daniel.voyer@bell.ca, d., Hassen, C., Gillis, K., Filsfils, C., Parekh, R., and H. Bidgoli, "SR Replication Policy for P2MP Service Delivery", draft-voyer-spring-sr-p2mp-policy-01 (work in progress), October 2018.

[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>.

[RFC3443] Agarwal, P. and B. Akyol, "Time To Live (TTL) Processing in Multi-Protocol Label Switching (MPLS) Networks", RFC 3443, DOI 10.17487/RFC3443, January 2003, <https://www.rfc-editor.org/info/rfc3443>.

[RFC4203] Kompella, K., Ed. and Y. Rekhter, Ed., "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4203, DOI 10.17487/RFC4203, October 2005, <https://www.rfc-editor.org/info/rfc4203>.

[RFC5307] Kompella, K., Ed. and Y. Rekhter, Ed., "IS-IS Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 5307, DOI 10.17487/RFC5307, October 2008, <https://www.rfc-editor.org/info/rfc5307>.

[RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <https://www.rfc-editor.org/info/rfc8029>.

[RFC8287] Kumar, N., Ed., Pignataro, C., Ed., Swallow, G., Akiya, N., Kini, S., and M. Chen, "Label Switched Path (LSP) Ping/Traceroute for Segment Routing (SR) IGP-Prefix and IGP-Adjacency Segment Identifiers (SIDs) with MPLS Data Planes", RFC 8287, DOI 10.17487/RFC8287, December 2017, <https://www.rfc-editor.org/info/rfc8287>.

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

10.2. Informative References

[I-D.ietf-isis-segment-routing-extensions] Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-19 (work in progress), July 2018.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions] Psenak, P., Filsfils, C., Previdi, S., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-15 (work in progress), August 2018.

[I-D.ietf-ospf-segment-routing-extensions]
Psenak, P., Previdi, S., Filsfils, C., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF Extensions for Segment Routing", draft-ietf-ospf-segment-routing-extensions-25 (work in progress), April 2018.

[I-D.ietf-spring-segment-routing-central-epe]
Filsfils, C., Previdi, S., Dawra, G., Aries, E., and D. Afanasiev, "Segment Routing Centralized BGP Egress Peer Engineering", draft-ietf-spring-segment-routing-central-epe-10 (work in progress), December 2017.

[I-D.ietf-spring-segment-routing-ldp-interop]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., and S. Litkowski, "Segment Routing interworking with LDP", draft-ietf-spring-segment-routing-ldp-interop-15 (work in progress), September 2018.

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-14 (work in progress), June 2018.

[IANA-MPLS-LSP-PING]
IANA, "Multi-Protocol Label Switching (MPLS) Label Switched Paths (LSPs) Ping Parameters", <http://www.iana.org/assignments/mpls-lsp-ping-parameters/mpls-lsp-ping-parameters.xhtml>.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

Authors' Addresses

Nagendra Kumar (editor)
Cisco Systems, Inc.
7200-12 Kit Creek Road
Research Triangle Park, NC 27709-4987
US

Email: naikumar@cisco.com

Carlos Pignataro (editor)
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709-4987
US

Email: cpignata@cisco.com

Faisal Iqbal
Cisco Systems, Inc.

Email: faiqbal@cisco.com

Zafar Ali
Cisco Systems, Inc.

Email: zali@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 9, 2019

J. Xie
Huawei Technologies
M. Chen
R. Li
Huawei

September 5, 2018

Multipoint LDP Extension for P2MP-based BIER
draft-xie-mpls-ldp-bier-extension-01

Abstract

Bit Index Explicit Replication (BIER) is a new architecture that provides optimal multicast forwarding without requiring intermediate routers to maintain any per-flow state by using a multicast-specific BIER header. An extension to the Label Distribution Protocol (LDP) defined in [RFC5036] for the setup of point-to-multipoint (P2MP) is described in [RFC6388] is called mLDP, and is used for multicast-specific tree building. This document describes mLDP extensions to setup a P2MP with BIER information, which is called P2MP based BIER in [I-D.xie-bier-mvpn-mpls-p2mp].

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 March 9, 2019.

Table of Contents

1. Introduction

Bit Index Explicit Replication (BIER) is a new architecture that provides optimal multicast forwarding without requiring intermediate routers to maintain any per-flow state by using a multicast-specific BIER header. An extension to the Label Distribution Protocol (LDP) defined in [RFC5036] for the setup of point-to-multipoint (P2MP) is described in [RFC6388] is called mLDP, and is used for multicast-specific tree building. This document describes mLDP extensions to

setup a P2MP with BIER information, which is called a P2MP based BIER in [I-D.xie-bier-mvpn-mpls-p2mp] .

Related documents that may be of interest include [RFC5561], and [RFC3988].

2. Terminology

Readers of this document are assumed to be familiar with the terminology and concepts of the documents listed as Normative References. For convenience, some of the more frequently used terms and new terms list below.

- mLDP: Multipoint extensions for LDP.
- P2MP LSP: An LSP that has one Ingress LSR and one or more Egress LSRs.
- Ingress LSR: An Ingress LSR for a particular LSP is an LSR that can send a data packet along the LSP. MP2MP LSPs can have multiple Ingress LSRs, P2MP LSPs have just one, and that node is often referred to as the "root node".
- Egress LSR: An Egress LSR for a particular LSP is an LSR that can remove a data packet from that LSP for further processing. P2P and MP2P LSPs have only a single egress node, but P2MP and MP2MP LSPs can have multiple egress nodes.
- Transit LSR: An LSR that has reachability to the root of the MP LSP via a directly connected upstream LSR and one or more directly connected downstream LSRs.
- Bud LSR: An LSR that is an egress but also has one or more directly connected downstream LSRs.
- Leaf node: A leaf node can be either an Egress or Bud LSR when referred to in the context of a P2MP LSP.
- FEC: Forwarding Equivalence Class
- P2MP FEC: defined in RFC6388.
- F-BM: Forwarding Bit Mask
- BSL: Bit String Length, that is 64, 128, 256, etc (per [RFC8279]).

3. MLDP P2MP BIER Signalling

3.1. Definitions

F-BM: Forwarding Bit Mask, An array of Bit, which is defined in [RFC8279].

Peer F-BM: For LSR A and P2MP FEC<Root,N>, this is the F-BM that included in Label Mapping from a downstream LSR for P2MP FEC<Root,N> to A.

Downstream F-BM: For LSR A and P2MP FEC<Root,N>, this is the OR'ing result of each of the F-BM included in Label Mapping from downstream LSR for P2MP FEC<Root,N> to A. A use this Downstream F-BM in its Lable Mapping to upstream node for P2MP FEC<Root,N>. In other words, A's Downstream F-BM is A's upstream node's Peer F-BM.

3.2. Example

Consider LSRs A - F, interconnected as follows:

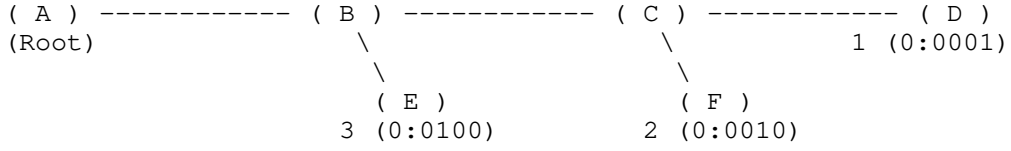


Figure 1: P2MP-based BIER Topology

Say that the node D has a BFR-id 1, F has a BFR-id 2, and E has a BFR-id 3, and we use a Bit String Length 4 (which is not valid per [RFC8296]) as an example.

Consider an P2MP FEC<Root=A,N=10> for which A is the Root, and say that D,E,F are all the Leafs of this P2MP FEC<Root=A, N=10>.

While D send LDP Mapping to C, it includes a F-BM 0001. Say that C got a Peer<D> F-BM 0001, and then C form a Downstream F-BM 0001.

While F send LDP Mapping to C, it includes a F-BM 0010. Say that C got a Peer<F> F-BM 0010, and then C form a Downstream F-BM 0011.

While C send LDP Mapping to B, it includes a F-BM 0010. Say that B got a Peer<C> F-BM 0011, and then B form a Downstream F-BM 0011.

While E send LDP Mapping to B, it includes a F-BM 0100. Say that B got a Peer<E> F-BM 0100, and then B form a Downstream F-BM 0111.

While B send LDP Mapping to A, it includes a F-BM 0111. Say that A got a Peer<B> F-BM 0111, and then A form a Downstream F-BM 0111.

This memo describes how every nodes in a P2MP tree get Peer F-BM from every downstream LSR, form a Downstream F-BM by OR'ing all it's Downstream Peer F-BM, and send a Mapping to it's upstream node using the Downstream F-BM.

3.3. Signaling the P2MP BIER

The procedure for signalling the P2MP BIER is performed hop-by-hop by each LSR L along an P2MP LSP for a given P2MP FEC<Root,N>. The steps are as follows:

1. First, L computes its Downstream F-BM for P2MP FEC<Root,N>:

If L is a leaf for P2MP FEC<Root,N>, L sets the F-BM with it's BFRID's BitPosition to 1.

Otherwise (L is not a Leaf), L computes the Downstream F-BM by OR'ing all it's downstream Node's F-BM, as described above.

2. For each LDP neighbor of L to which L decides to send a Mapping for FEC F, L attaches an BIER TLV with the F-BM that it computed for this P2MP FEC.

3. When a new BIER TLV is received for P2MP FEC<Root,N> from a downstream LSR or the set of downstream LSRs, L returns to step 1. If the newly computed Downstream F-BM is unchanged, L SHOULD NOT advertise new information to its upstream neighbor. Otherwise, L readvertises its Mappings to its upstream neighbor with an updated BIER TLV.

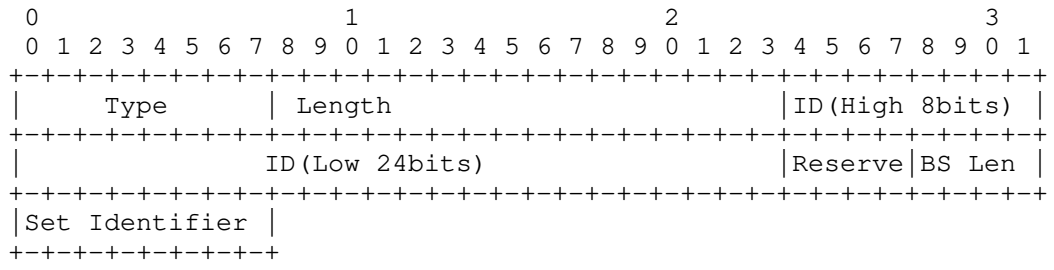
This behavior is standard for attributes such as path vector, hop count, and MTU, and the same rules apply, as specified in [RFC5036].

If the Downstream F-BM changes, L MAY readvertise the new F-BM immediately, or hold down the readvertisement for a while.

3.4. The P2MP BIER LSP Identifier

[RFC6388] defined the P2MP FEC Element, which include a LDP MP Opaque Value Element. It also defined a Generic LSP Identifier as the LDP MP Opaque Value Element, with a TLV of Type 1. This document defined a new type of LDP MP Opaque Value Element, called the P2MP BIER LSP Identifier.

The encoding for the P2MP BIER LSP Identifer is as follows:



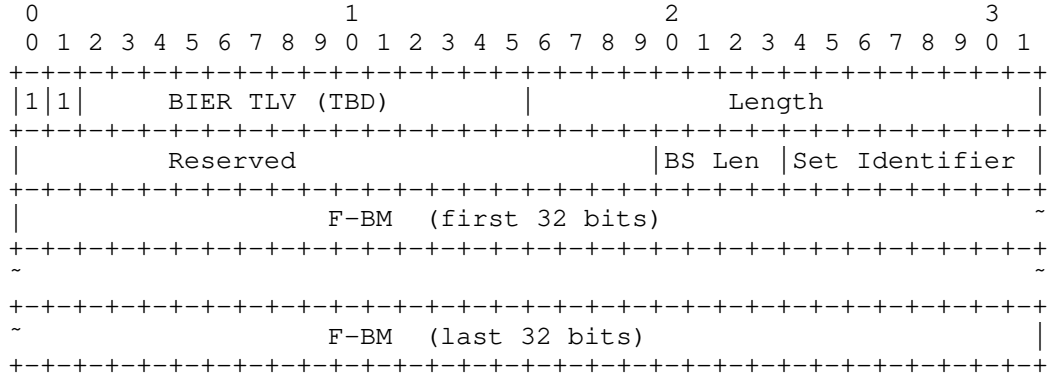
Type: TBD. Need to be less than 255.
Length: 6
ID: A 32-bit integer, same as RFC6388.
BS Len: Bit String Length
Set Identifier: as defined in RFC8279.

Figure 2: P2MP BIER LSP Identifer

3.5. BIER TLV

The BIER TLV encodes information on the F-BM for an LSP, from the advertising LSR to the egress(es) over all valid paths.

The encoding for the BIER TLV is as follows:



Type: TBD.
Length: Length
BSL: Bit String Length
Set Identifier: as defined in RFC8279.
F-BM: Forwarding Bit Mask

Figure 3: BIER TLV

3.6. Make Before Break (MBB)

The Make Before Break (MBB) mechanism for mLDP P2MP defined in RFC6388 also applies.

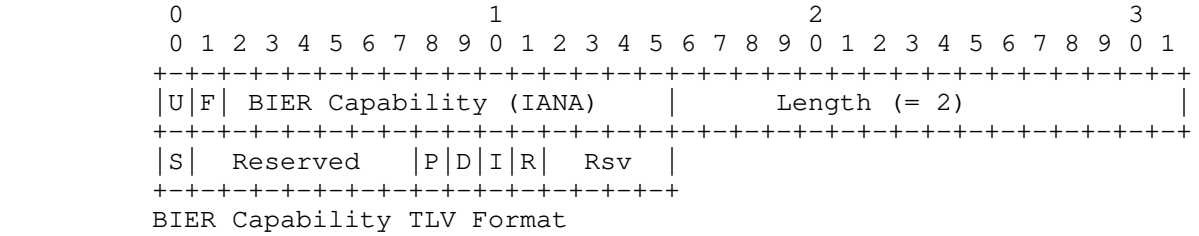
4. Capability and Error handling

The extensions defined in this document utilize the existing LDP error handling defined in [RFC5036]. If an LSR receives an error notification from a peer for a session, it terminates the LDP session by closing the TCP transport connection for the session and discarding all multi-topology label mappings learned via the session.

An LSR should respond with an LDP MP Status in LDP Notification Messages when it receives an LDP Label Mapping message with a P2MP FEC element specifying an BIER TLV that is not locally known or not supported. The LSR MUST also discard the entire message before sending the Notification message.

4.1. BIER Capability

A new optional capability parameter TLV, RMR Capability, is defined. Following is the format of the RMR Capability Parameter:



BIER Capability TLV Format

U: set to 1. Ignore, if not known.
F: Set to 0. Do not forward.
S: MUST be set to 1 to advertise the BIER TLV.
P: The node has BIER P-Capability.
D: The node has BIER D-Capability.
I: The node Ignore the BIER Header except the Label.
R: The node Require a packet without BIER Header except the Label.

Figure 4: BIER Capability

The BIER Capability TLV MUST be advertised in the LDP Initialization message. If the peer has not advertised the BIER capability, then label messages including a BIER TLV MUST NOT be sent to the peer.

If an LSR has not advertised that it is BIER capable, its LDP peers MUST NOT send it messages that include BIER TLV. If an LSR receives

a Label Mapping message with an BIER TLV from downstream LSR-D and its upstream LSR-U has not advertised that it is BIER capable, the LSR MUST send an BIER notification immediately to LSR-D. If this happens, an P2MP BIER LSP will not be established, a normal P2MP LSP will not be established either.

P-Capability indicate a complete BIER function, which includes P-Capability and D-Capability. If a node support P-Capability, then it support the whole BIER function, which means it support both P-capability and D-capability.

D-Capability indicate a subset of BIER function, to Disposition some length of a packet from some offset. For example, from BIER Label and a whole (BIER Header Length) , or from the position after BIER Label and a length of (BIER Header Length - BIER Label Length) . If a node don't support P-Capability, it may still support D-Capability. If a node don't support D-capability, it must not support P-Capability.

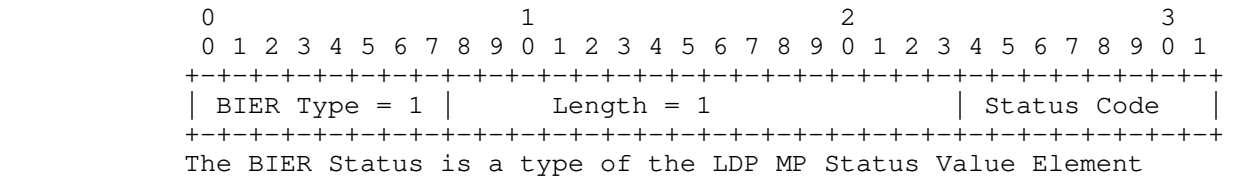
If a Node doesn't have P-Capability, then P flag MUST be cleared. Whether the node will be a Branch or BUD or Leaf, the I flag SHOULD be set.

if a node doesn't have D-Capability, then P and D flag MUST be cleared. If the node will be a BUD or Leaf then R flag SHOULD be set. if the node will be a Branch then R flag MAY not be set.

if a node doesn't have P-Capability but does have D-Capability, then D flag SHOULD be set, but R flag MAY be set or not be set.

4.2. BIER Status

A new optional LDP MP Status Value Element (RFC6388), BIER Status, is defined. Following is the format of the BIER Status:



The BIER Status is a type of the LDP MP Status Value Element

LDP MP Status Value Element TLV Type(RFC6388):

- Type 0: Reserved
- Type TBD: BIER Status

Status Code:

- 1 = BIER TLV not supported
- 2 = Leaf or Bud don't have D-capability, must set R flag
- 3 = Branch or Bud don't have P-capability, must set I flag
- 4 = Node must set R flag when Upstream has R flag
- 5 = Node must clear R flag when any of downstream not have R flag

Figure 5: BIER Status TLV

4.3. Check Capability and Use Status for Error Handling

In order to deploy P2MP BIER on a network with some legacy nodes, which may not support P-capability or D-capability, some Capability flags need to be checked and notification messages may be generated.

If all edge nodes support D-capability, but some nodes don't support P-capability and they set a I flag as required, then deployment of P2MP BIER is fine, and a BIER packet can walk from Root to all Leaf(s) without any change except the BIER Label.

If an LSR support P-capability, and it's upstream node also support P-capability, and when the LSR receives a Label Mapping message with an BIER TLV and R flag set from downstream LSR-D, the LSR will accept such Label Mapping message. If receives a Label Mapping wiht an BIER TLV and R flag cleaned another downstream LSR-D', the LSR will accept too.

If an LSR receives a Label Mapping message with an BIER TLV from downstream LSR-D with a R flag, and its upstream LSR-U has no P-capability, the LSR MUST send an BIER notification immediately to LSR-D. If this happens, an P2MP BIER LSP will not be established, a normal P2MP LSP will not be established either.

When an LSR receives a Label Mapping message, it need to do some check before process and build P2MP BIER forwarding table. Such check includes:

1) Check if the node's P-capability and D-capability conflict with it's I flag and R flag.

The following table list the whole check matrix.

NODE's Role	NODE's PDIR-flag	Check Result	Rule
Leaf	[PDxx]	OK	(*) Comment 1
Leaf	[-Dxx]	OK	
Leaf	[--xR]	OK	Rule 1
Leaf	[--x-]	ERR	Rule 1
Branch	[PDxx]	OK	(*) Comment 1
Branch	[-DIx]	OK	Rule 2
Branch	[-D-x]	ERR	Rule 2
Branch	[--Ix]	OK	(*) Comment 2
Branch	[---x]	ERR	Rule 2
BUD	[PDxx]	OK	(*) Comment 1
BUD	[-DIx]	OK	Rule 2
BUD	[-D-x]	ERR	Rule 2
BUD	[--IR]	OK	
BUD	[--I-]	ERR	Rule 1
BUD	[---R]	ERR	Rule 2
BUD	[----]	ERR	Rule 1 & 2

(*) Comment 1: In some cases, set a R flag is useful
(*) Comment 2: In some cases, Clear R flag on a Branch node is useful
Rule 1: Leaf don't have D-capability, must set R flag
Rule 2: Branch don't have P-capability, must set I flag

Figure 6: BIER Self Check Matrix

2) Check the node's R flag, the node's upstream R flag, the node's downstream R flag.

The following table list the whole check martrix about the R flag of node, the upstream node, the downstream nodes.

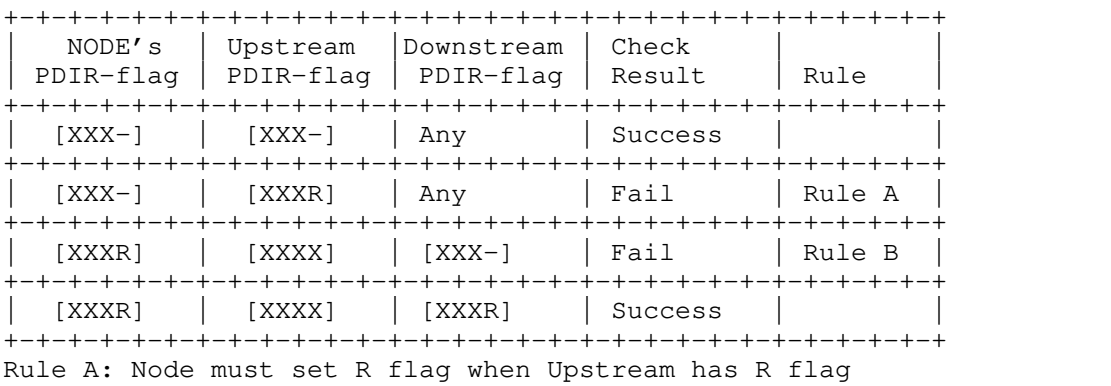


Figure 7: BIER upstream and downstream check

When the above two checks fail, a notification message with a reason code is sent to the downstream node which send the Label Mapping message.

5. IANA Considerations

Allocation is expected from IANA for a new type codepoints for "P2MP BIER LSP Identifier" from the "LDP MP Opaque Value Element basic type" registry of the "Label Distribution Protocol (LDP) Parameters" registry.

Allocation is expected from IANA for a new type codepoints for "BIER TLV" from the "TLV Type Name Space" registry of the "Label Distribution Protocol (LDP) Parameters" registry.

Allocation is expected from IANA for a new type codepoints for "BIER capability TLV" from the "TLV Type Name Space" registry of the "Label Distribution Protocol (LDP) Parameters" registry.

Allocation is expected from IANA for a new type codepoints for "BIER Status" from the "LDP MP Status Value Element type" registry of the "Label Distribution Protocol (LDP) Parameters" registry.

6. Security Considerations

TBD

7. Acknowledgements

TBD

8. References

8.1. Normative References

[I-D.ietf-bier-mvpn]
Rosen, E., Sivakumar, M., Aldrin, S., Dolganow, A., and T. Przygienda, "Multicast VPN Using BIER", draft-ietf-bier-mvpn-11 (work in progress), March 2018.

[I-D.xie-bier-mvpn-mpls-p2mp]
Xie, J., McBride, M., Chen, M., and L. Geng, "Multicast VPN Using MPLS P2MP and BIER", draft-xie-bier-mvpn-mpls-p2mp-02 (work in progress), July 2018.

[RFC3988] Black, B. and K. Kompella, "Maximum Transmission Unit Signalling Extensions for the Label Distribution Protocol", RFC 3988, DOI 10.17487/RFC3988, January 2005, <https://www.rfc-editor.org/info/rfc3988>.

[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>.

[RFC5561] Thomas, B., Raza, K., Aggarwal, S., Aggarwal, R., and JL. Le Roux, "LDP Capabilities", RFC 5561, DOI 10.17487/RFC5561, July 2009, <https://www.rfc-editor.org/info/rfc5561>.

[RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011, <https://www.rfc-editor.org/info/rfc6388>.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <https://www.rfc-editor.org/info/rfc8279>.

[RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <https://www.rfc-editor.org/info/rfc8296>.

8.2. Informative 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>.

Authors' Addresses

Jingrong Xie
Huawei Technologies
Q15 Huawei Campus, No.156 Beiqing Rd.
Beijing 100095
China

Email: xiejingrong@huawei.com

Mach Chen
Huawei

Email: mach.chen@huawei.com

Robin Li
Huawei

Email: lizhenbin@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 9, 2019

J. Xie
Huawei Technologies
M. Chen
R. Li
Huawei

September 5, 2018

RSVP-TE Extensions for P2MP-based BIER
draft-xie-mpls-rsvp-bier-extension-01

Abstract

Bit Index Explicit Replication (BIER) is a new architecture that provides optimal multicast forwarding without requiring intermediate routers to maintain any per-flow state by using a multicast-specific BIER header. This document describes extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for the set up of Traffic Engineered (TE) point-to-multipoint (P2MP) Label Switched Paths (LSPs) with BIER infomation, which is called P2MP based BIER in [I-D.xie-bier-mvpn-mpls-p2mp].

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 March 9, 2019.

Table of Contents

1. Introduction

Bit Index Explicit Replication (BIER) is a new architecture that provides optimal multicast forwarding without requiring intermediate routers to maintain any per-flow state by using a multicast-specific BIER header. [RFC4875] defines extensions to the RSVP-TE protocol ([RFC3209] and [RFC3473]) to support P2MP TE LSPs satisfying the set of requirements described in [RFC4461] .

This document extends RSVP-TE to establish P2MP TE LSPs with BIER information, which is called P2MP based BIER in [I-D.xie-bier-mvpn-mpls-p2mp].

2. Terminology

Readers of this document are assumed to be familiar with the terminology and concepts of the documents listed as Normative References. For convenience, some of the more frequently used terms and new terms list below.

- LSP: Label Switch Path
- LSR: Label Switching Router
- BFR: BIER Forwarding Router
- BFR-ID: BIER Forwarding Router IDentify.
- P2MP: Point to Multi-point
- P2MP based BIER: BIER using P2MP as topology

3. RSVP Extensions

RSVP Extensions to setup a P2MP-based BIER is very similar to the setup of a P2MP LSP described in [RFC4875]. Most of the structure and description are borrowed from RFC4875, and a precursive example is put in the beginning to give a whole picture of building the forwarding state of P2MP based BIER.

3.1. Example of signaling the P2MP-BIER

Consider LSRs A - F, interconnected as follows:

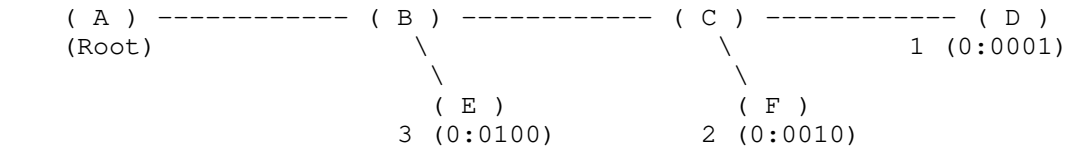


Figure 1: P2MP-based BIER Topology

Say that the node D has a BFR-id 1, F has a BFR-id 2, and E has a BFR-id 3, and we use a Bit String Length 4.

Consider an P2MP SESSION<P2MPID, TunnelID, ExtTunnelID=RootAddr>, for which A is the Root, and say that D,E,F are all the Leafs of this P2MP SESSION.

There are 3 Sub-LSPs: A-->B-->E, A-->B-->C-->D, A-->B-->C-->F.

PATH message: When PATH message walk through A-->B-->E, it include an session attribute that identify ths session is to establish a P2MP-based BIER LSP. The same to A-->B-->C-->D and A-->B-->C-->F.

RESV message: When RESV message work throuth A<--B<--E, it include an Object that identify BFR-ID of E. The same to A<--B<--C<--D and A<--B<--C<--F.

Procedure: B learns that it's downstream endpoint has a BFR-ID<3> after a RSVP message passes through A<--B<--E. B also learns a BFR-ID<1> after a RSVP message passes throuth A<--B<--C<--D, and a BFR-ID<2> after a RSVP message passes through A<--B<--C<--D.

3.2. PATH Message

This section describes modifications made to the Path message format as specified in [RFC4875]. The Path message is enhanced to signal one or more S2L sub-LSPs with BIER information. This is done by including the S2L BIER sub-LSP descriptor list in the Path message as shown below.

```
<Path Message> ::=     <Common Header> [ <INTEGRITY> ]
                      [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ...]
                      [ <MESSAGE_ID> ]
                      <SESSION> <RSVP_HOP>
                      <TIME_VALUES>
                      [ <EXPLICIT_ROUTE> ]
                      <LABEL_REQUEST>
                      [ <PROTECTION> ]
                      [ <LABEL_SET> ... ]
                      [ <SESSION_ATTRIBUTE> ]
                      [ <NOTIFY_REQUEST> ]
                      [ <ADMIN_STATUS> ]
                      [ <POLICY_DATA> ... ]
                      <sender descriptor>
                      [<S2L BIER sub-LSP descriptor list>]
<S2L BIER sub-LSP descriptor list> ::= <S2L BIER sub-LSP descriptor>
                                    [ <S2L BIER sub-LSP descriptor list> ]
<S2L BIER sub-LSP descriptor> ::= <S2L_BIER_SUB_LSP>
                             [ <P2MP SECONDARY_EXPLICIT_ROUTE> ]
```

Figure 2: PATH Message

3.3. RESV Message

The Resv message follows the [RFC4875] format:

```
<Resv Message> ::=    <Common Header> [ <INTEGRITY> ]
                      [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                      [ <MESSAGE_ID> ]
                      <SESSION> <RSVP_HOP>
                      <TIME_VALUES>
                      [ <RESV_CONFIRM> ]  [ <SCOPE> ]
                      [ <NOTIFY_REQUEST> ]
                      [ <ADMIN_STATUS> ]
                      [ <POLICY_DATA> ... ]
                      <STYLE> <flow descriptor list>

<flow descriptor list> ::= <FF flow descriptor list>
                         | <SE flow descriptor>

<FF flow descriptor list> ::= <FF flow descriptor>
                            | <FF flow descriptor list>
                              <FF flow descriptor>

<SE flow descriptor> ::= <FLOWSPEC> <SE filter spec list>

<SE filter spec list> ::= <SE filter spec>
                        | <SE filter spec list> <SE filter spec>

<FF flow descriptor> ::= [ <FLOWSPEC> ] <FILTER_SPEC> <LABEL>
                         [ <RECORD_ROUTE> ]
                         [ <S2L BIER sub-LSP flow descriptor list> ]

<SE filter spec> ::=     <FILTER_SPEC> <LABEL> [ <RECORD_ROUTE> ]
                         [ <S2L BIER sub-LSP flow descriptor list> ]

<S2L BIER sub-LSP flow descriptor list> ::=
                           <S2L BIER sub-LSP flow descriptor>
                           [ <S2L BIER sub-LSP flow descriptor list> ]

<S2L BIER sub-LSP flow descriptor> ::= <S2L_BIER_SUB_LSP>
                                  [ <P2MP_SECONDARY_RECORD_ROUTE> ]
```

Figure 3: RESV Message

FILTER_SPEC is defined in below section.

The S2L BIER sub-LSP flow descriptor has the same format as S2L BIER sub-LSP descriptor in previous section with the difference that a P2MP_SECONDARY_RECORD_ROUTE object is used in place of a P2MP SECONDARY_EXPLICIT_ROUTE object.

Note that a Resv message can signal multiple S2L BIER sub-LSPs that may belong to the same FILTER_SPEC object or different FILTER_SPEC

objects. The same label SHOULD be allocated if the <Sender Address, LSP-ID> fields of the FILTER_SPEC object are the same.

However different labels MUST be allocated if the <Sender Address, LSP-ID> of the FILTER_SPEC object is different, as that implies that the FILTER_SPEC refers to a different P2MP BIER LSP.

3.4. SESSION Object

A P2MP BIER LSP SESSION object is used. This object uses the existing SESSION C-Num. New C-Types are defined to accommodate a logical P2MP destination identifier of the P2MP BIER tunnel. This SESSION object has a similar structure as the existing point-to-multipoint RSVP-TE SESSION object. However the C-Types is different. All S2L BIER sub-LSPs that are part of the same P2MP BIER LSP share the same SESSION object. This SESSION object identifies the P2MP BIER tunnel.

The combination of the SESSION object, the SENDER_TEMPLATE object and the S2L_BIER_SUB_LSP object identifies each S2L BIER sub-LSP. This follows the existing P2MP RSVP-TE notion of using the SESSION object for identifying a P2MP Tunnel, which in turn can contain multiple LSPs, each distinguished by a unique SENDER_TEMPLATE object.

3.4.1. P2MP BIER Tunnel IPv4 SESSION Object

Class = SESSION, P2MP_BIER_TUNNEL_IPv4 C-Type = TBD

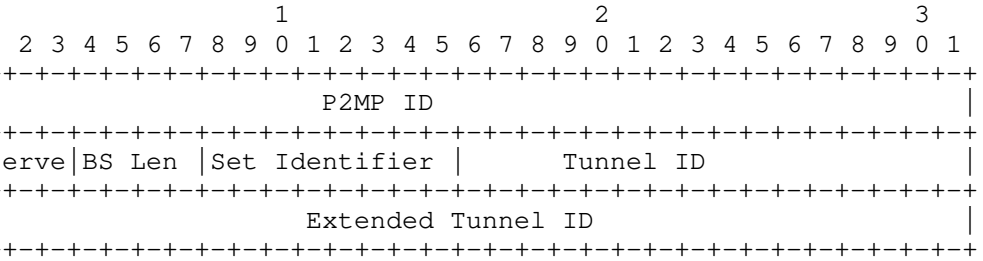


Figure 4: P2MP BIER Tunnel IPv4 SESSION Object

P2MP ID: A 32-bit identifier used in the SESSION object that remains constant over the life of the P2MP BIER tunnel. It encodes the P2MP Identifier that is unique within the scope of the ingress LSR.

BS Len: A 4 bits field encoding the supported BitString length associated with this BFR-prefix. The values allowed in this field are specified in section 2 of [RFC8296].

Set Identifier: A 8 bits fields encoding the Set Identifier (section 1 of [RFC8279])

Tunnel ID: A 16-bit identifier used in the SESSION object that remains constant over the life of the P2MP BIER tunnel.

Extended Tunnel ID: A 32-bit identifier used in the SESSION object that remains constant over the life of the P2MP BIER tunnel. Ingress LSRs that wish to have a globally unique identifier for the P2MP BIER tunnel SHOULD place their tunnel sender address here. A combination of this address, P2MP ID, and Tunnel ID provides a globally unique identifier for the P2MP BIER tunnel.

3.4.2. P2MP BIER Tunnel IPv6 SESSION Object

Class = SESSION, P2MP_BIER_TUNNEL_IPv6 C-Type = TBD

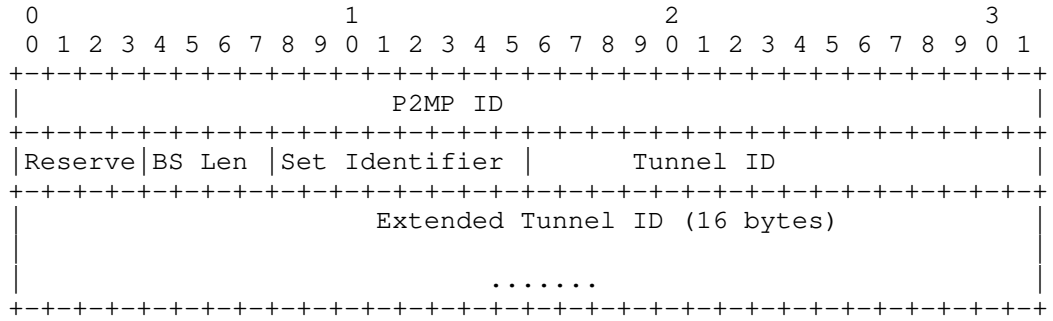


Figure 5: P2MP BIER Tunnel IPv6 SESSION Object

This is the same as the P2MP BIER Tunnel IPv4 LSP SESSION object with the difference that the extended tunnel ID may be set to a 16-byte identifier [RFC3209].

3.5. SENDER_TEMPLATE Object

The SENDER_TEMPLATE object contains the ingress LSR source address. The LSP ID can be changed to allow a sender to share resources with itself. Thus, multiple instances of the P2MP BIER tunnel can be created, each with a different LSP ID. The instances can share resources with each other. The S2L BIER sub-LSPs corresponding to a particular instance use the same LSP ID.

The combination of the SESSION object, the SENDER_TEMPLATE object and the S2L_BIER_SUB_LSP object identifies each S2L BIER sub-LSP. This follows the existing P2MP RSVP-TE notion of using the SESSION object

for identifying a P2MP Tunnel, which in turn can contain multiple LSPs, each distinguished by a unique SENDER_TEMPLATE object.

3.5.1. P2MP BIER Tunnel IPv4 SENDER_TEMPLATE Object

Class = SENDER_TEMPLATE, P2MP_BIER_TUNNEL_IPv4 C-Type = TBD

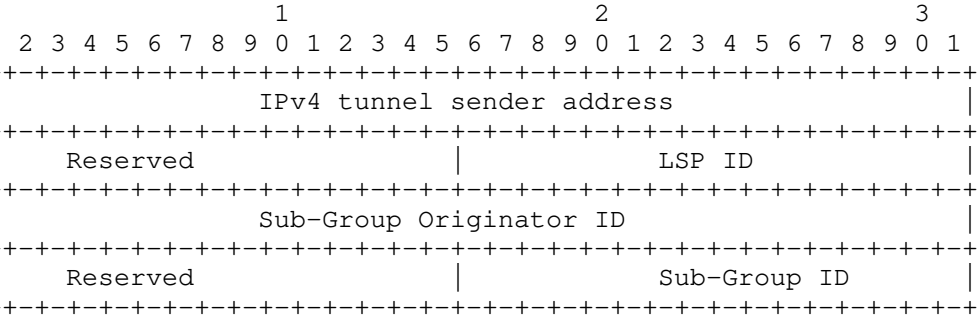


Figure 6: P2MP BIER Tunnel IPv4 SENDER_TEMPLATE Object

IPv4 tunnel sender address

See [RFC3209].

Sub-Group Originator ID

The Sub-Group Originator ID is set to the TE Router ID of the LSR that originates the Path message. This is either the ingress LSR or an LSR which re-originates the Path message with its own Sub- Group Originator ID.

Sub-Group ID

An identifier of a Path message used to differentiate multiple Path messages that signal state for the same P2MP BIER LSP. This may be seen as identifying a group of one or more egress nodes targeted by this Path message.

LSP ID

See [RFC3209].

3.5.2. P2MP BIER Tunnel IPv6 SENDER_TEMPLATE Object

Class = SENDER_TEMPLATE, P2MP_BIER_TUNNEL_IPv6 C-Type = TBD

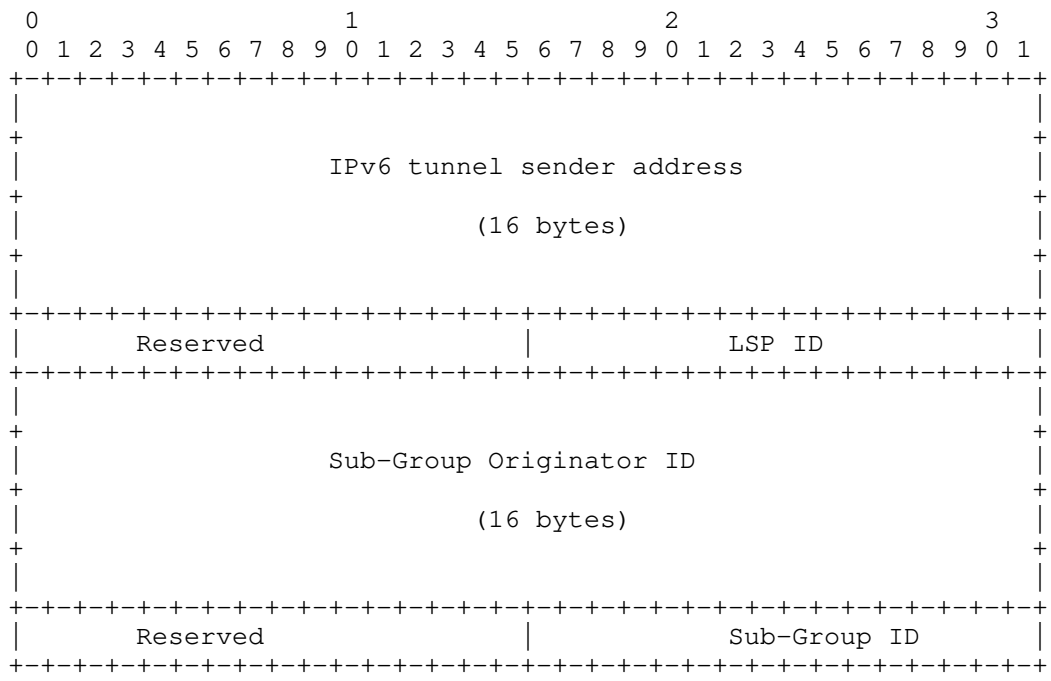


Figure 7: P2MP BIER Tunnel IPv6 SENDER_TEMPLATE Object

This is the same as the P2MP BIER Tunnel IPv4 SENDER_TEMPLATE Object with the difference that the sender address and Sub-Group Originator ID may be set to a 16-byte identifier [RFC3209].

3.6. S2L_BIER_SUB_LSP Object

An S2L_BIER_SUB_LSP object identifies a particular S2L BIER sub-LSP belonging to the P2MP BIER LSP.

3.6.1. S2L_BIER_SUB_LSP IPv4 Object

S2L_BIER_SUB_LSP Class = TBD, S2L_BIER_SUB_LSP_IPv4 C-Type = TBD

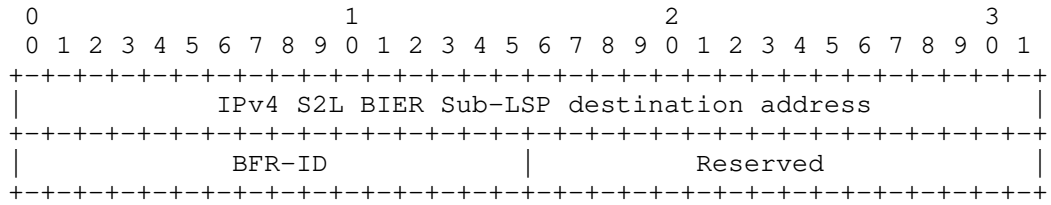


Figure 8: S2L_BIER_SUB_LSP IPv4 Object

IPv4 BIER Sub-LSP destination address

IPv4 address of the S2L BIER sub-LSP destination.

3.6.2. S2L_BIER_SUB_LSP IPv6 Object

S2L_BIER_SUB_LSP Class = TBD, S2L_BIER_SUB_LSP_IPv6 C-Type = TBD

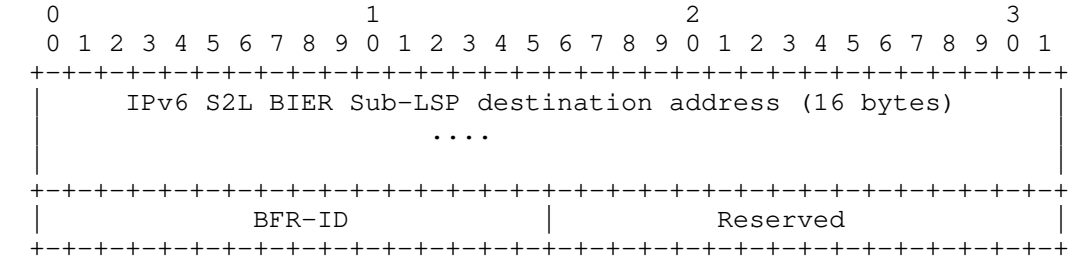


Figure 9: S2L_BIER_SUB_LSP IPv6 Object

This is the same as the S2L BIER IPv4 Sub-LSP object, with the difference that the destination address is a 16-byte IPv6 address.

3.7. FILTER_SPEC Object

The FILTER_SPEC object is canonical to the P2MP BIER SENDER_TEMPLATE object.

3.7.1. P2MP BIER_IPv4 FILTER_SPEC Object

Class = FILTER_SPEC, P2MP BIER LSP_IPv4 C-Type = TBD

The format of the P2MP BIER_IPv4 FILTER_SPEC object is identical to the P2MP BIER_IPv4 SENDER_TEMPLATE object.

3.7.2. P2MP BIER_IPv6 FILTER_SPEC Object

Class = FILTER_SPEC, P2MP BIER LSP_IPv6 C-Type = TBD

The format of the P2MP BIER_IPv6 FILTER_SPEC object is identical to the P2MP BIER_IPv6 SENDER_TEMPLATE object.

4. Capability and Error Handling

TBD.

5. IANA Considerations

Allocation is expected from IANA for codepoints from the "Class Names, Class Numbers, and Class Types" registry.

6. Security Considerations

TBD

7. Acknowledgements

TBD

8. References

8.1. Normative References

[I-D.ietf-bier-mvpn]
Rosen, E., Sivakumar, M., Aldrin, S., Dolganow, A., and T. Przygienda, "Multicast VPN Using BIER", draft-ietf-bier-mvpn-11 (work in progress), March 2018.

[I-D.xie-bier-mvpn-mpls-p2mp]
Xie, J., McBride, M., Chen, M., and L. Geng, "Multicast VPN Using MPLS P2MP and BIER", draft-xie-bier-mvpn-mpls-p2mp-02 (work in progress), July 2018.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <https://www.rfc-editor.org/info/rfc3209>.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <https://www.rfc-editor.org/info/rfc3473>.

[RFC4461] Yasukawa, S., Ed., "Signaling Requirements for Point-to-Multipoint Traffic-Engineered MPLS Label Switched Paths (LSPs)", RFC 4461, DOI 10.17487/RFC4461, April 2006, <https://www.rfc-editor.org/info/rfc4461>.

[RFC4875] Aggarwal, R., Ed., Papadimitriou, D., Ed., and S. Yasukawa, Ed., "Extensions to Resource Reservation Protocol - Traffic Engineering (RSVP-TE) for Point-to-Multipoint TE Label Switched Paths (LSPs)", RFC 4875, DOI 10.17487/RFC4875, May 2007, <https://www.rfc-editor.org/info/rfc4875>.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <https://www.rfc-editor.org/info/rfc8279>.

[RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", RFC 8296, DOI 10.17487/RFC8296, January 2018, <https://www.rfc-editor.org/info/rfc8296>.

8.2. Informative 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>.

Authors' Addresses

Jingrong Xie
Huawei Technologies
Q15 Huawei Campus, No.156 Beiqing Rd.
Beijing 100095
China

Email: xiejingrong@huawei.com

Mach Chen
Huawei

Email: mach.chen@huawei.com

Robin Li
Huawei

Email: lizhenbin@huawei.com