

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 30, 2015

X. Chen  
Z. Li  
S. Hares  
Huawei Technologies  
R. White  
J. Tantsura  
Ericsson  
October 27, 2014

I2RS Information Model for MPLS LDP  
draft-chen-i2rs-mpls-ldp-info-model-00

## Abstract

The Label Distribution Protocol (LDP) ([\[RFC5036\]](#)) is a protocol defined for distributing labels in MPLS domain. Traditionally LDP protocol may be managed via CLI, SNMP or NETCONF. The Interface to the Routing System's (I2RS) Programmatic interface ([draft-ietf-i2rs-architecture](#)) provides an alternate way to control the configuration and diagnose the operation of the LDP protocol.

This document specifies an information model for the LDP protocol to facilitate the definition of a standardized data model, which can be used to define interfaces to the LDP from an entity that may even be external to the routing system. Based on standardized data model and interfaces, use cases of I2RS interface to LDP protocol ([draft-chen-i2rs-mpls-ldp-usecases](#)) can be supported.

## 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 <http://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

Internet-Draft

I2RS Information Model for MPLS LDP

October 2014

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 30, 2015.

## Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">2.</a>	LDP Data . . . . .	<a href="#">3</a>
<a href="#">2.1.</a>	LDP Instance . . . . .	<a href="#">4</a>
<a href="#">2.1.1.</a>	LDP Instance Parameters . . . . .	<a href="#">4</a>
<a href="#">2.1.2.</a>	LDP Interface . . . . .	<a href="#">4</a>
<a href="#">2.1.3.</a>	LDP Remote Peer . . . . .	<a href="#">5</a>
<a href="#">2.1.4.</a>	LDP Peer . . . . .	<a href="#">7</a>
<a href="#">2.1.5.</a>	LDP Peer Information . . . . .	<a href="#">7</a>
<a href="#">2.1.6.</a>	LDP Session . . . . .	<a href="#">9</a>
<a href="#">2.1.7.</a>	LDP LSP . . . . .	<a href="#">11</a>
<a href="#">2.1.8.</a>	LDP Policy . . . . .	<a href="#">12</a>
<a href="#">2.1.9.</a>	LDP Status . . . . .	<a href="#">13</a>
<a href="#">3.</a>	LDP notification . . . . .	<a href="#">14</a>
<a href="#">4.</a>	I2RS YANG model of LDP . . . . .	<a href="#">15</a>
<a href="#">5.</a>	IANA Considerations . . . . .	<a href="#">17</a>
<a href="#">6.</a>	Security Considerations . . . . .	<a href="#">17</a>
<a href="#">7.</a>	Acknowledgements . . . . .	<a href="#">17</a>
<a href="#">8.</a>	Normative References . . . . .	<a href="#">18</a>
	Authors' Addresses . . . . .	<a href="#">19</a>

## [1.](#) Introduction

The Label Distribution Protocol (LDP) ([\[RFC5036\]](#)) is a protocol defined for distributing labels in MPLS domain. Traditionally LDP protocol may be managed via CLI, SNMP or NETCONF. With the expansion and complication of modern networks, the necessity for rapid and

dynamic control has been increased. The Interface to the Routing System's (I2RS) Programmatic interface ([\[I-D.ietf-i2rs-architecture\]](#)) provides an alternate way to control the configuration and diagnose the operation of the LDP protocol and achieve this goal.

This document specifies an information model for the LDP protocol to facilitate the definition of a standardized data model, which can be used to define interfaces to the LDP from an entity that may even be external to the routing system. Based on standardized data model and interfaces, use cases and requirements for I2RS interface to LDP Protocol [\[I-D.chen-i2rs-mpls-ldp-usecases\]](#) can be supported.

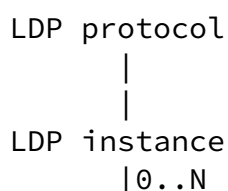
Please note I2RS utilizes ephemeral configuration plus status information. This draft proposes needs of this ephemeral configuration, and the authors of this draft intent to collaborate with related work on yang configuration for MPLS LDP.

## [2.](#) LDP Data

This section describes the data involved in the LDP information model in detail. LDP data includes information related to LDP instances, LDP entities, LDP peers, LDP sessions, LDP LSPs and LDP policies.

There are two kinds of LDP entities which are used for LDP discovery. One kind of LDP entity uses interface to engage in LDP basic discovery which is to set up the sessions between directly connected LSRs. The other uses remote peer to engage in extended discovery which is to set up the sessions between non-directly connected LSRs.

A high-level architecture of the LDP contents is shown as below.



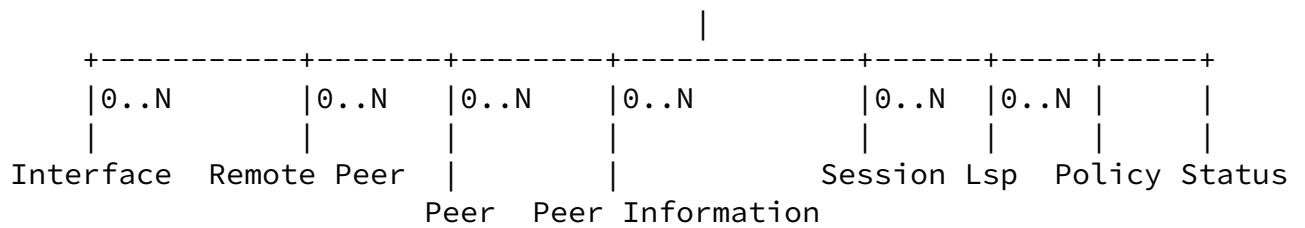


Figure 1: Architecture of LDP information model

## 2.1. LDP Instance

In the context of LDP information model, LDP instance is virtual private network (VPN) instance or public network instance which contains the instance name of VPN or public network, the parameters of LDP instance and the data of LDP instance. Multiple instances MAY be supported in one network device.

The corresponding YANG description of the top level is below.

```

module: i2rs-mplsldp
  +--rw mplsLdp
    +--rw ldpInstances
      +--rw ldpInstance* [vrfName]
        +--rw vrfName          string
        +--rw lsrid            inet:ipv4-address
        +--rw ldpInterfaces
          | ...
        +--rw ldpRemotePeers
          | ...
        +--rw ldpPeers
          | ...
        +--rw ldpPeerInfos
          | ...
        +--rw ldpSessions
          | ...
        +--rw ldpLsps
          | ...
        +--rw ldpPolicy

```

```

|      ...
+---ro ldpInstanceStatus
|      ...

```

### [2.1.1.](#) LDP Instance Parameters

- o vrfName: A name uniquely identifying LDP instance which is from the name of VPN instance. If the name string is empty the instance means a public instance whose name is \_public\_.
- o lsrid: LSR ID of a LDP instance.

### [2.1.2.](#) LDP Interface

LDP interface is one kind of LDP entity which is engaged in LDP basic discovery which is to set up the sessions between directly connected LSRs in LDP instance.

This section describes the information model related to LDP interface which is shown in the Yang high-level description and interprets the meaning of each element.

```

+---rw ldpInterfaces
|  +---rw ldpInterface* [ifName]
|    +---rw ifName                               ifName
|    +---rw helloSendTime?                       uint16
|    +---rw helloHoldTime?                       uint16
|    +---rw keepaliveSendTime?                   uint16
|    +---rw keepaliveHoldTime?                   uint16
|    +---rw igpSyncDelayTime?                    uint32
|    +---rw transportAddrInterface?              ifName
|    +---rw localLsrIdAddrInterface?             ifName
|    +---rw labelAdvMode?                        ldpLabelDistMode

```

- o ifName: the name of interface which is enabled as an LDP entity.
- o helloSendTime: the value of the timer for periodically sending hello packet. The value is in seconds.
- o helloHoldTime: the interval value of the Hello hold timer. The

value is in seconds.

o keepaliveSendTime: the value of the timer for periodically sending keepalive packet. The value is in seconds.

o keepaliveHoldTime: the interval value of the keepalive hold timer. The value is in seconds.

o igpSyncDelayTime: the interval value at which an interface waits to establish an LSP after an LDP session is set up. The value is in seconds.

o transportAddrInterface: an interface address to be used as the transport address.

o localLsrIdAddrInterface: an interface address to be used as local LSR-ID address.

o labelAdvMode: the label distribution mode used by a session which is DU or DoD.

### [2.1.3.](#) LDP Remote Peer

LDP Remote Peer is one kind of LDP entity which is engaged in extended discovery which is to set up the sessions between non-directly connected LSRs in LDP instance.

This section describes the information model related to LDP remote peer which is shown in the Yang high-level description and interprets the meaning of each element.

```
+--rw ldpRemotePeers
|   +--rw ldpRemotePeer* [remotePeerName]
|       +--rw remotePeerName          string
|       +--rw remoteIp?                inet:ipv4-address
|       +--rw helloSendTime?           uint16
|       +--rw helloHoldTime?           uint16
|       +--rw keepaliveSendTime?       uint16
|       +--rw keepaliveHoldTime?       uint16
|       +--rw igpSyncDelayTime?        uint32
|       +--rw localLsrIdAddrInterface? ifName
|       +--rw labelAdvMode?            ldpLabelDistMode
```

|       +---rw remoteIpAutoDoDRequest?       boolean

o remotePeerName: the name of a remote neighbor which is as an remote entity.

o remoteIp: the IPv4 address of a remote neighbor which the LDP targeted hello packet will be sent to.

o helloSendTime: the value of the timer for periodically sending hello packet. The value is in seconds.

o helloHoldTime: the interval value of the Hello hold timer. The value is in seconds.

o keepaliveSendTime: the value of the timer for periodically sending keepalive packet. The value is in seconds.

o keepaliveHoldTime: the interval value of the keepalive hold timer. The value is in seconds.

o igpSyncDelayTime: the interval value at which a remote peer waits to establish an LSP after an LDP session is set up. The value is in seconds.

o localLsrIdAddrInterface: an interface address to be used as local LSR-ID address.

o labelAdvMode: the label distribution mode used by a session which is DU or DoD.

o remoteIpAutoDoDRequest: the policy of triggering LDP DoD requests for some prefixes.

#### [2.1.4.](#) LDP Peer

LDP Peer is used to configure local parameters which are used or sent to peer LSR for negotiation in LDP session establishment.

This section describes the information model related to local parameters of specified LDP peer which is shown in the yang high-level description and interprets the meaning of each element.

```

+--rw ldpPeers
|   +--rw ldpPeer* [peerid]
|       +--rw peerid                               inet:ipv4-address
|       +--rw labelAdvMode?                         ldpLabelDistMode
|       +--rw localLsrIdAddrInterface?              ifName
|       +--rw announcementCapability?               boolean
|       +--rw mldpP2mpCapability?                   boolean
|       +--rw mldpMBBCapability?                   boolean
|       +--rw ldpSACCapability?                     uint32

```

o peerid: LDP peer ip address which composes the LDP LSR ID.

o labelAdvMode: the label distribution mode used by a session which is DU or DoD.

o localLsrIdAddrInterface: an interface address to be used as local lsr-id address.

o announcementCapability: specify that if the announcement capability is supported locally which will be used to negotiate with peer.

o mldpP2mpCapability: specify that if the p2mp capability is supported locally which will be used to negotiate with peer.

o mldpMBBCapability: specify that if the mldp MBB capability is supported locally which will be used to negotiate with peer.

o ldpSACCapability: specify that the application types of ldp SAC(State Advertisement Control) capability supported locally which will be used to negotiate with peer. SAC capability is defined in [[I-D.ietf-mpls-ldp-ip-pw-capability](#)].

#### [2.1.5.](#) LDP Peer Information

LDP Peer information is peer parameters which are received from peer LSR or configured for the peer LSR.

This section describes the information model related to information



of LDP peer which is shown in the Yang high-level description and interprets the meaning of each element.

```

+--rw ldpPeerInfos
|   +--rw ldpPeerInfo* [peerLsrid]
|       +--rw peerLsrid                               string
|       +--rw peerMaxPduLen                           uint16
|       +--rw peerLoopDetect                         boolean
|       +--rw peerPVLimit                             uint16
|       +--rw peerFtFlag?                             boolean
|       +--rw peerTportAddr?                         inet:ipv4-address
|       +--rw keepaliveHoldTime?                     uint16
|       +--rw recoveryTime?                           uint16
|       +--rw reconnectTime?                         uint16
|       +--rw labelAdvMode?                           ldpLabelDistMode
|       +--rw announcementCapability?                 boolean
|       +--rw mldpP2mpCapability?                     boolean
|       +--rw mldpMBBCapability?                     boolean
|       +--rw ldpSACCapability?                       uint32

```

o peerLsrid: peer LDP identifier which 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.

o peerMaxPduLen: max length of the PDU peer sends

o peerLoopDetect: specify that if the peer support the loop detect capability.

o peerPVLimit: specify that the path vector limit supported by the peer.

o peerFtFlag: specify that if the peer support the FT(Fault Tolerance) capability.

o peerTportAddr: transport address of peer used for transport connection.

o keepaliveHoldTime: keepalive hold time of peer.

o recoveryTime: recovery time used for graceful restart of peer.

o reconnectTime: reconnect time used for graceful restart of peer.

- o labelAdvMode: the label distribution mode used by a session which is DU or DoD.
- o announcementCapability: specify that if the peer support the announcement capability.
- o mldpP2mpCapability: specify that if the peer support the p2mp capability.
- o mldpMBBCapability: specify that if the peer support the mldp MBB(Make Before Break) capability.
- o ldpSACCapability: specify the application types of ldp SAC capability of peer.

#### [2.1.6.](#) LDP Session

LDP session is established over a TCP transport connection. Normally in the LDP session initialization phase the session parameters are negotiated. LDP Capabilities in [[RFC5661](#)] defines a mechanism for advertising LDP enhancements at session initialization time, as well as a mechanism to enable and disable enhancements after LDP session establishment.

This section describes the information model related to LDP session which is shown in the Yang high-level description and interprets the meaning of each element.

```

+--rw ldpSessions
|   +--rw ldpSession* [peerLsrid]
|       +--rw peerLsrid                string
|       +--rw localLsrid?              string
|       +--rw tcpSourceAddr?           inet:ipv4-address
|       +--rw tcpDestAddr?             inet:ipv4-address
|       +--ro sessionState?            enumeration
|       +--ro sessionRole?             enumeration
|       +--ro sessionType?             enumeration
|       +--rw negotiatedKaHoldTime?    uint32
|       +--ro kaSent?                  uint32
|       +--ro kaReceived?              uint32
|       +--rw sessionDistMode?         enumeration
|       +--rw ftFlag?                  boolean
|       +--ro md5Flag?                 boolean
|       +--rw reconnetTime?            uint32
|       +--rw recoveryTime?            uint32
|       +--ro sessionAge?              uint32
|       +--rw announcementCapability?  boolean
|       +--rw mldpP2mpCapability?      boolean
|       +--rw mldpMBBCapability?      boolean
|       +--rw ldpSACCapability?        uint32

```

o peerLsrid: peer LDP identifier which 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.

o localLsrid: local LDP identifier.

o tcpSourceAddr: TCP connection source address used by a session.

o tcpDestAddr: destination address of the TCP connection used by a session.

o sessionState: session state according to the state machine.

o sessionRole: session role of the LSR which is active or passive.

- o sessionType: session type which is local, remote or both.
- o negotiatedKaHoldTime: the value of the Keepalive hold timer negotiated by local and peer LSR.
- o kaSent: the number of keepalive messages which are sent.
- o kaReceived: the number of keepalive messages which are received.

- o sessionDistMode: the label distribution mode negotiated by local and peer LSR.
- o ftFlag: specify if the session support the FT capability.
- o md5Flag: specify if the session support the MD5 capability.
- o reconnetTime: reconnect time used for graceful restart.
- o recoveryTime: recovery time used for graceful restart.
- o sessionAge: duration since the session is set up.
- o announcementCapability: specify if the session support the announcement capability.
- o mldpP2mpCapability: specify if the session support the p2mp capability.
- o mldpMBBCapability: specify if the session support the mldp MBB capability.
- o ldpSACCapability: specify the application types of ldp SAC(State Advertisement Control ) capability of session.

#### [2.1.7.](#) LDP LSP

LDP LSP is the LSP established according to the LDP protocol. It has three types which are ingress, transit and egress LSP. The FEC may has multiple ingress or transit LSPs which have different outgoing interface and next hop.

This section describes the information model related to LDP LSP which is shown in the Yang high-level description and interprets the meaning of each element.

```

+--rw ldpLsps
|   +--rw ldpLsp* [lspAddr prefixLength lspIndex lspType outIfaceName]
|       +--ro lspAddr                inet:ipv4-address
|       +--ro prefixLength            uint32
|       +--ro lspIndex                uint32
|       +--ro lspType                 enumeration
|       +--ro outIfaceName            ifName
|       +--ro nextHop                 inet:ipv4-address
|       +--ro inLabel?                uint32
|       +--ro outLabel?               uint32
|       +--ro isFrrLsp?               boolean
|       +--ro lspMtu?                 uint32
|       +--ro lspAge?                 uint32

```

- o lspAddr: prefix address of an LDP LSP.
- o prefixLength: prefix length of the prefix address of an LSP.
- o lspIndex: an LSP index.
- o lspType: LSP type which is ingress, transit or egress.
- o outIfaceName: outgoing interface.
- o nextHop: next hop address.

- o inLabel: the value of incoming label.
- o outLabel: the value of outgoing label.
- o isFrrLsp: specify if the LDP LSP is FRR LSP.
- o lspMtu: specifies an LSP MTU for the outgoing packet.
- o lspAge: duration since the LSP is setup.

#### [2.1.8.](#) LDP Policy

LDP policy can be used to limit the set up of LDP LSP by controlling the advertisement of LDP label mappings or other method. The outbound policy can be used for specified peer, a group of peers or all the peers.

This section describes the information model related to LDP policy which is shown in the Yang high-level description and interprets the meaning of each element.

```

+---rw ldpPolicy
|   +---rw ldpOutBoundFecPeers
|   |   +---rw ldpOutBoundFecPeer* [peerid]
|   |   |   +---rw peerid                inet:ipv4-address
|   |   |   +---rw fecPolicyMode?         fecIpPrefixGroupType
|   |   |   +---rw fecIpPrefixName?      string
|   |   +---rw ldpOutBoundPeerFecGroups
|   |   |   +---rw ldpOutBoundPeerFecGroup* [peerid]
|   |   |   |   +---rw peerGroupName      string
|   |   |   |   +---rw fecPolicyMode?     fecIpPrefixGroupType
|   |   |   |   +---rw fecIpPrefixName?   string
|   |   +---rw ldpOutBoundFecPeerAll
|   |       +---rw fecPolicyMode?         fecIpPrefixGroupType
|   |       +---rw fecIpPrefixName?      string

```

- o peerid: LDP peer IP address which composes the LDP LSR ID.
- o fecPolicyMode: the mode which specifies the scope of FECs which is

none, all or ip-prefix.

- o fecIpPrefixName: the name of IP prefix which represents the group of FECs whose label mappings can be sent. (Why is there only permit?)

- o peerGroupName: the name of IP prefix which represents the group of peers which use the same policy.

#### [2.1.9.](#) LDP Status

LDP Status is about the state or statistics of LDP data.

This section describes the information model related to LDP status which is shown in the Yang high-level description and interprets the meaning of each element.

```

+--ro ldpInstanceStatus
  +--ro sessionNum?          uint32
  +--ro adjacencyNum?        uint32
  +--ro interfaceNum?        uint32
  +--ro fecNum?              uint32
  +--ro lspNum?              uint32
  |   +--ro ingressLsp?      uint32
  |   +--ro egressLsp?       uint32
  |   +--ro transitLsp?      uint32
  |   +--ro totalLsp?        uint32
  +--ro p2mpLspNum
```

++ro ingressLsp?	uint32
++ro egressLsp?	uint32
++ro transitLsp?	uint32
++ro budLsp?	uint32
++ro totalLsp?	uint32

- o sessionNum: number of sessions.
- o adjacencyNum: number of adjacencies.
- o interfaceNum: number of interfaces served as LDP interface entities.
- o fecNum: number of FECs.
- o ingressLsp: number of ingress P2P LDP LSPs or mLDP P2MP LSPs.
- o egressLsp: number of egress P2P LDP LSPs or mLDP P2MP LSPs.
- o transitLsp: number of transit P2P LDP LSPs or mLDP P2MP LSPs.
- o totalLsp: number of total P2P LDP LSPs or mLDP P2MP LSPs.
- o budLsp: number of bud mLDP P2MP LSPs.

### 3. LDP notification

The notification features within the I2RS interface would allow applications associated with an I2RS client to subscribe to a stream of state changes regarding LDP protocol from the I2RS Agent. An LDP protocol data-model MUST support sending asynchronous notifications. A brief list of suggested notifications is as below:

- o LDP session state change(up/down) notification
- o LDP Ingress LSP state change(up/down) notification for ip-prefix with prefix length being 32

- o LDP LSP count(total LSP number) reaches the upper limit notification
- o LDP LSP count(total LSP number) exceeds the threshold(threshold LSP



number) notification

#### [4.](#) I2RS YANG model of LDP

module: i2rs-mplsldp

```
+--rw mplsLdp
  +--rw ldpInstances
    +--rw ldpInstance* [vrfName]
      +--rw vrfName          string
      +--rw lsrid            inet:ipv4-address
      +--rw ldpInterfaces
        +--rw ldpInterface* [ifName]
          +--rw ifName          ifName
          +--rw helloSendTime?  uint16
          +--rw helloHoldTime?  uint16
          +--rw keepaliveSendTime? uint16
          +--rw keepaliveHoldTime? uint16
          +--rw igpSyncDelayTime? uint32
          +--rw transportAddrInterface? ifName
          +--rw localLsrIdAddrInterface? ifName
          +--rw labelAdvMode?    ldpLabelDistMode
      +--rw ldpRemotePeers
        +--rw ldpRemotePeer* [remotePeerName]
          +--rw remotePeerName    string
          +--rw remoteIp?         inet:ipv4-address
          +--rw helloSendTime?    uint16
          +--rw helloHoldTime?    uint16
          +--rw keepaliveSendTime? uint16
          +--rw keepaliveHoldTime? uint16
          +--rw igpSyncDelayTime? uint32
          +--rw localLsrIdAddrInterface? ifName
          +--rw labelAdvMode?      ldpLabelDistMode
          +--rw remoteIpAutoDoDRequest? boolean
      +--rw ldpPeers
        +--rw ldpPeer* [peerid]
          +--rw peerid            inet:ipv4-address
          +--rw labelAdvMode?      ldpLabelDistMode
          +--rw localLsrIdAddrInterface? ifName
          +--rw announcementCapability? boolean
          +--rw mldpP2mpCapability? boolean
          +--rw mldpMBBCapability? boolean
          +--rw ldpSACCapability?  uint32
      +--rw ldpPeerInfos
        +--rw ldpPeerInfo* [peerLsrid]
```

```

|      +---rw peerLsrid                                string
|      +---rw peerMaxPduLen                            uint16
|      +---rw peerLoopDetect                          boolean
|      +---rw peerPVLimit                             uint16
|      +---rw peerFtFlag?                             boolean
|      +---rw peerTportAddr?                          inet:ipv4-address
|      +---rw keepaliveHoldTime?                      uint16
|      +---rw recoveryTimer?                          uint16
|      +---rw reconnectTimer?                        uint16
|      +---rw labelAdvMode?                          ldpLabelDistMode
|      +---rw announcementCapability?                boolean
|      +---rw mldpP2mpCapability?                    boolean
|      +---rw mldpMBBCapability?                    boolean
|      +---rw ldpSACCapability?                      uint32
+---rw ldpSessions
|   +---rw ldpSession* [peerLsrid]
|       +---rw peerLsrid                                string
|       +---rw localLsrid?                            string
|       +---rw tcpSourceAddr?                        inet:ipv4-address
|       +---rw tcpDestAddr?                        inet:ipv4-address
|       +---ro sessionState?                        enumeration
|       +---ro sessionRole?                        enumeration
|       +---ro sessionType?                        enumeration
|       +---rw negotiatedKaHoldTime?                uint32
|       +---ro kaSent?                              uint32
|       +---ro kaReceived?                          uint32
|       +---rw sessionDistMode?                    enumeration
|       +---rw ftFlag?                              boolean
|       +---ro md5Flag?                              boolean
|       +---rw reconnetTime?                        uint32
|       +---rw recoveryTime?                        uint32
|       +---ro sessionAge?                          string
|       +---rw announcementCapability?                boolean
|       +---rw mldpP2mpCapability?                    boolean
|       +---rw mldpMBBCapability?                    boolean
|       +---rw ldpSACCapability?                      uint32
+---rw ldpLsps
|   +---rw ldpLsp* [lspAddr prefixLength lspIndex lspType outIfaceName]
|       +---ro lspAddr                                inet:ipv4-address
|       +---ro prefixLength                          uint32
|       +---ro lspIndex                              uint32
|       +---ro lspType                              enumeration
|       +---ro outIfaceName                          ifName
|       +---ro nextHop                              inet:ipv4-address
|       +---ro inLabel?                              uint32
|       +---ro outLabel?                            uint32
|       +---ro isFrrLsp?                            boolean

```

	+++ro lspMtu?	uint32
--	---------------	--------

```

|      +++ro lspTimeStamp?                uint32
+---rw ldpPolicy
|   +---rw ldpOutBoundFecPeers
|   |   +---rw ldpOutBoundFecPeer* [peerid]
|   |   |   +---rw peerid                inet:ipv4-address
|   |   |   +---rw fecPolicyMode?        fecIpPrefixGroupType
|   |   |   +---rw fecIpPrefixName?     string
|   +---rw ldpOutBoundPeerFecGroups
|   |   +---rw ldpOutBoundPeerFecGroup* [peerid]
|   |   |   +---rw peerGroupName         string
|   |   |   +---rw fecPolicyMode?        fecIpPrefixGroupType
|   |   |   +---rw fecIpPrefixName?     string
|   +---rw ldpOutBoundFecPeerAll
|   |   +---rw fecPolicyMode?            fecIpPrefixGroupType
|   |   +---rw fecIpPrefixName?         string
+---ro ldpInstanceStatus
|   +---ro sessionNum?                   uint32
|   +---ro adjacencyNum?                 uint32
|   +---ro interfaceNum?                 uint32
|   +---ro fecNum?                       uint32
|   +---ro lspNum?                       uint32
|   |   +---ro ingressLsp?               uint32
|   |   +---ro egressLsp?                uint32
|   |   +---ro transitLsp?               uint32
|   |   +---ro totalLsp?                 uint32
|   +---ro p2mpLspNum
|   |   +---ro ingressLsp?               uint32
|   |   +---ro egressLsp?                uint32
|   |   +---ro transitLsp?               uint32
|   |   +---ro budLsp?                   uint32
|   |   +---ro totalLsp?                 uint32

```

Figure 2 The I2RS YANG model of LDP

## 5. IANA Considerations

This draft includes no request to IANA.

## 6. Security Considerations

This document introduces no new security threat and SHOULD follow the security requirements as stated in [[I-D.ietf-i2rs-architecture](#)].

## [7.](#) Acknowledgements

TBD

Chen, et al.

Expires April 30, 2015

[Page 17]

---

Internet-Draft

I2RS Information Model for MPLS LDP

October 2014

## [8.](#) Normative References

[I-D.chen-i2rs-mpls-ldp-usecases]

Chen, X. and Z. Li, "Use Cases for an Interface to LDP Protocol", [draft-chen-i2rs-mpls-ldp-usecases-00](#) (work in progress), October 2013.

[I-D.ietf-i2rs-architecture]

Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", [draft-ietf-i2rs-architecture-05](#) (work in progress), July 2014.

[I-D.ietf-mpls-ldp-ip-pw-capability]

Raza, K. and S. Boutros, "Controlling State Advertisements Of Non-negotiated LDP Applications", [draft-ietf-mpls-ldp-ip-pw-capability-08](#) (work in progress), October 2014.

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

[RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP Specification", [RFC 5036](#), October 2007.

[RFC5443] Jork, M., Atlas, A., and L. Fang, "LDP IGP Synchronization", [RFC 5443](#), March 2009.

[RFC5661] Shepler, S., Eisler, M., and D. Noveck, "Network File System (NFS) Version 4 Minor Version 1 Protocol", [RFC 5661](#), January 2010.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", [RFC 6020](#),

October 2010.

- [RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", [RFC 6241](#), June 2011.
- [RFC6388] Wijnands, IJ., Minei, I., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", [RFC 6388](#), November 2011.
- [RFC6720] Pignataro, C. and R. Asati, "The Generalized TTL Security Mechanism (GTSM) for the Label Distribution Protocol (LDP)", [RFC 6720](#), August 2012.

Chen, et al.

Expires April 30, 2015

[Page 18]

---

Internet-Draft

I2RS Information Model for MPLS LDP

October 2014

#### Authors' Addresses

Xia Chen  
Huawei Technologies  
Huawei Bld., No.156 Beiqing Rd.  
Beijing 100095  
China

Email: [jescia.chenxia@huawei.com](mailto:jescia.chenxia@huawei.com)

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

Email: [lizhenbin@huawei.com](mailto:lizhenbin@huawei.com)

Susan Hares  
Huawei Technologies  
Saline, MI 48176  
US

Email: [shares@ndzh.com](mailto:shares@ndzh.com)

Russ White  
Ericsson  
US

Email: russ.white@ericsson.com

Jeff Tantsura  
Ericsson

Email: jeff.tantsura@ericsson.com