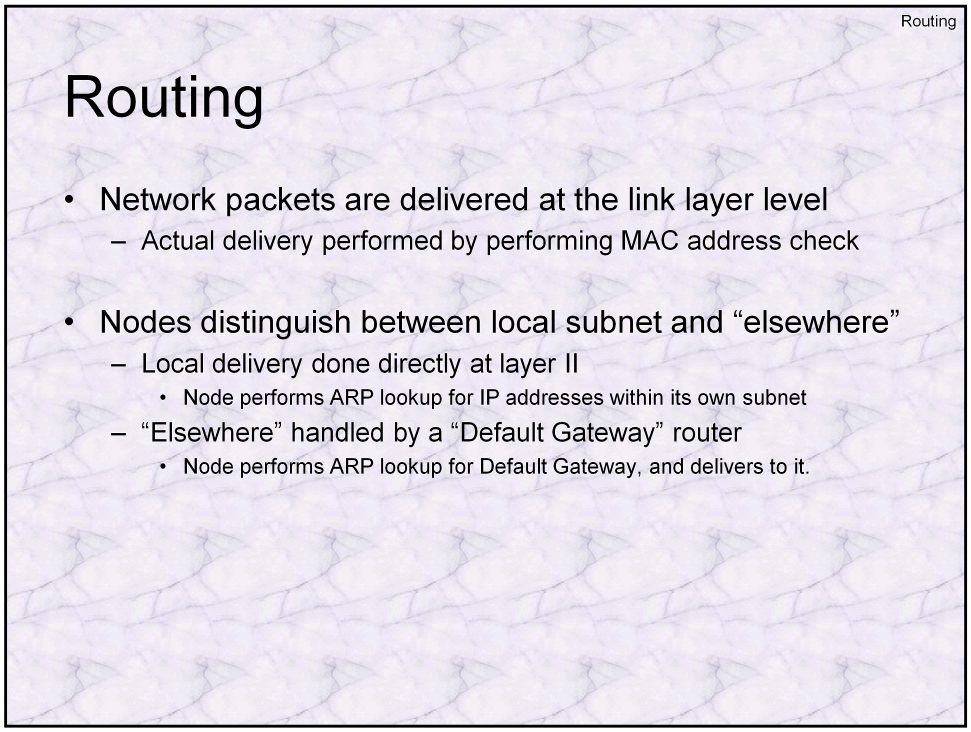
## Table of Contents

## About the Author:

**J**onathan Levin specializes in training and consulting services. This, and many other training materials, are created and constantly updated to reflect the ever changing environment of the IT industry.

To report errata, or for more details, feel free to email **JL@HisOwn.com**



Routing

## Routing

- Network packets are delivered at the link layer level
  - Actual delivery performed by performing MAC address check
- Nodes distinguish between local subnet and "elsewhere"
  - Local delivery done directly at layer II
    - Node performs ARP lookup for IP addresses within its own subnet
  - "Elsewhere" handled by a "Default Gateway" router
    - Node performs ARP lookup for Default Gateway, and delivers to it.

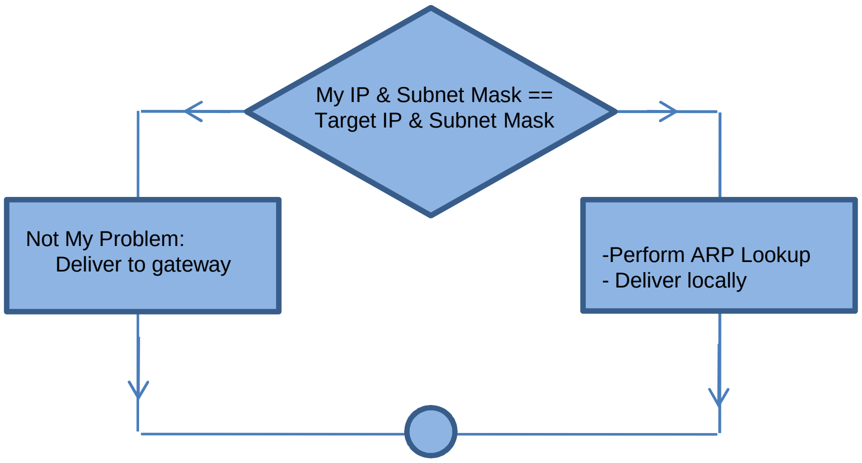
Getting packets from their source to their destination involves finding the path, or **route**, between them. For most nodes, this involves answering a simple question: Is the packet destined to an immediately connected neighbor on the same link, or destined elsewhere.

Every host, in the most minimal of configurations for network connectivity, must have at least three parameters defined, as this example from a Windows XP host shows:

```
C:\> ipconfig
Windows IP Configuration
Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : hisown.com
        IP Address. . . . . . . . . . . . : 10.24.19.75
        Subnet Mask . . . . . . . . . . . : 255.255.0.0
        Default Gateway . . . . . . . . . : 10.24.20.99
```

These three parameters enable the host to perform the routing decision by examining the destination IP address against its own, with the value called the "Subnet Mask": This is a bit mask (specified in decimal octets (=bytes)) with which the IP Addresses are masked by means of a bitwise "and" (&) operation in this way:

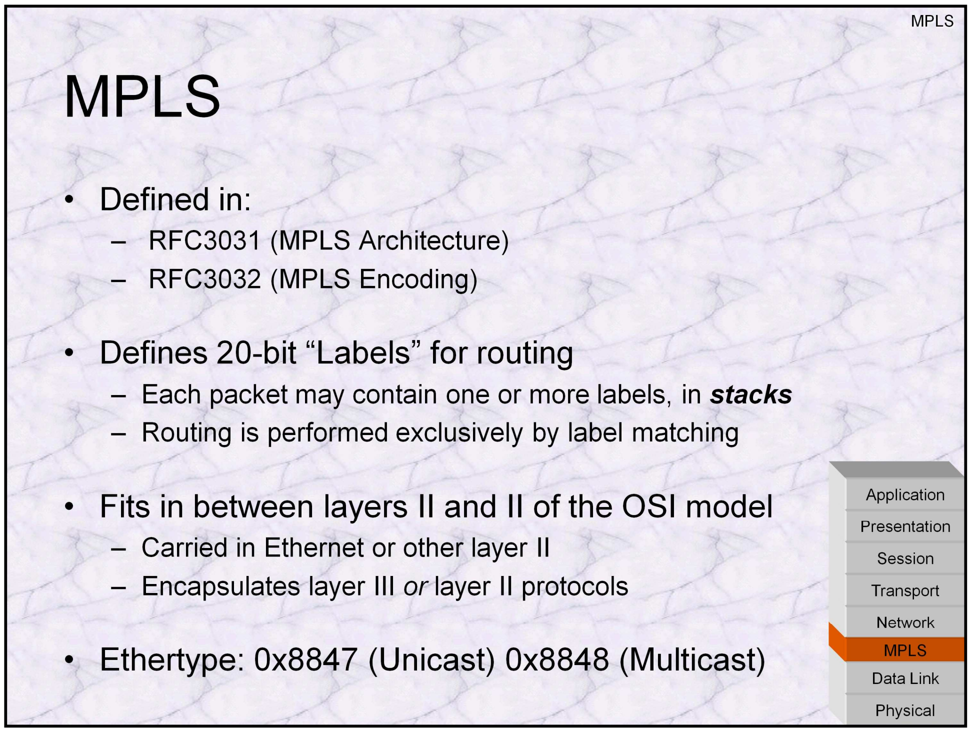

My IP & Subnet Mask == Target IP & Subnet Mask
Not My Problem: Deliver to gateway
-Perform ARP Lookup
- Deliver locally

The “&” of a “1” bit with any bit value (0 or 1) retains the bit value, whereas an “&” of a “0” with any bit value (again, 0 or 1) sets it to 0, regardless. “255” is 0xFF, or a “1” in all 8-bits of the octet (i.e. “11111111”) . So, in the example above, any address beginning with “10.24” is considered local, since the application of the subnet mask of “255” on an octet retains the octet value, whereas the application of a “0” nullifies it. Thus, 10.24.19.75 & 255.255.0.0 yields “10.24.0.0”.

For local addresses, the node can issue an ARP request* to query for the target’s link address. For non-local, however, the node needs assistance in routing the packet to its destination. This assistance comes from a “default gateway” – a router that has multiple interfaces (network connections) – one of the local subnet, and at least one more on some other subnet. The node issues an ARP request for the default gateway’s local interface address, and delivers it. From that point on, the packet becomes the gateway’s problem, and it must route it somehow.

The corollary from the above is that if a host is configured without a default gateway, or if the default gateway is incorrectly configured, the host will still maintain local connectivity, but not external connectivity.

* - ARP is an extremely loud, broadcast based protocol – meaning it is visible to all hosts on the subnet and interrupts them, so hosts aim to use it at the bare minimum. The actual ARP lookup via broadcast are actually performed only once, and the results are cached for a short period (usually 60-120 seconds). This saves the need for subsequent ARP requests for each packet, and so long as packets are transferred between the hosts the caching timeout period resets.



MPLS

# MPLS

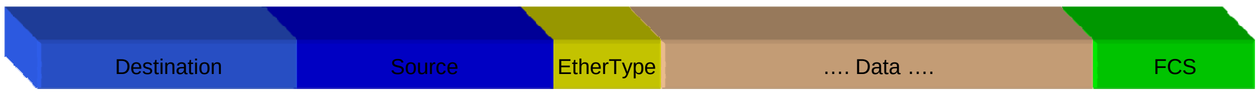
- Defined in:
  - RFC3031 (MPLS Architecture)
  - RFC3032 (MPLS Encoding)
- Defines 20-bit "Labels" for routing
  - Each packet may contain one or more labels, in ***stacks***
  - Routing is performed exclusively by label matching
- Fits in between layers II and II of the OSI model
  - Carried in Ethernet or other layer II
  - Encapsulates layer III *or* layer II protocols
- Ethertype: 0x8847 (Unicast) 0x8848 (Multicast)

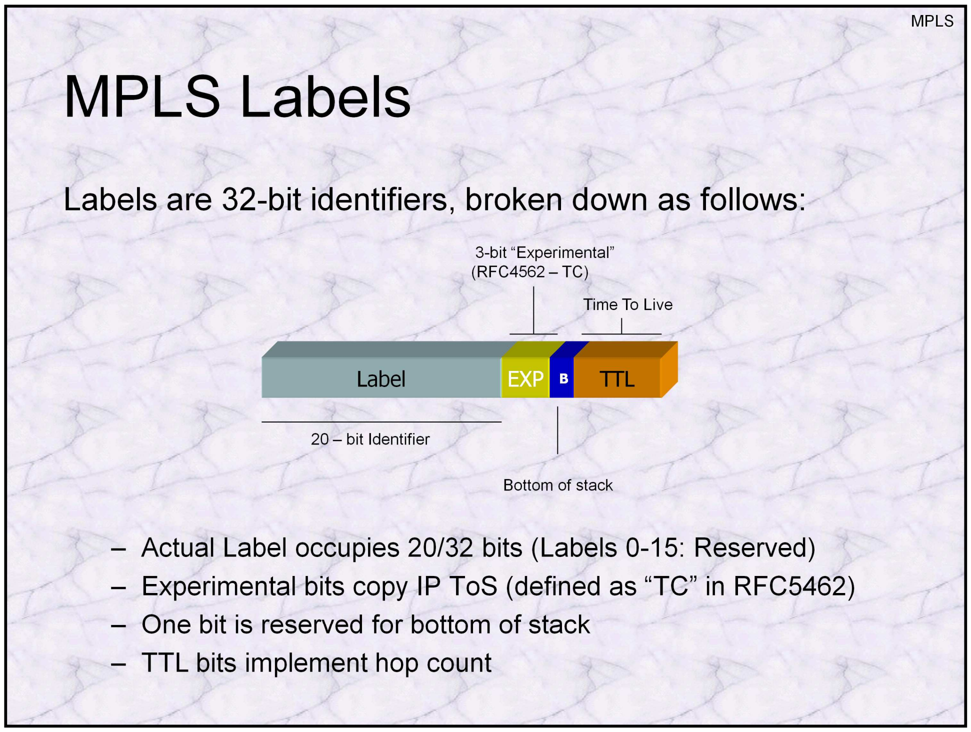
Multi-Protocol Label Switching or, in short, MPLS, is defined in two core RFCs. RFC3031 discusses the architecture, whereas RFC3032 discusses encoding of labels and packet formats. The idea is simple – packets are prepended by a 32-bit field that contains a 20-bit "flow label". This enables the routing to be performed by the MPLS label, often a simple lookup in a hash table, rather than through a costly IPv4 or IPv6 address lookup.

MPLS can either encapsulate, or be encapsulated in other protocols. MPLS is commonly used to tunnel IPv4/IPv6 traffic, but it's not uncommon to see layer II VPNs (i.e., tunneling entire Ethernet frames in MPLS). Nor is it unusual for MPLS to be tunneled in IP, ad we will see later.

When transported in Ethernet or HDLC, MPLS is identified by the Ethertype/protocol value of 0x8847. MPLS Multicast packets use 0x8848. MPLS can also be transported in other Layer II protocols: In PPP, it is identified by PPP protocol type 0x0281. In Frame Relay, its NPLID is 0x80.

**Ethernet II**:





## MPLS Labels

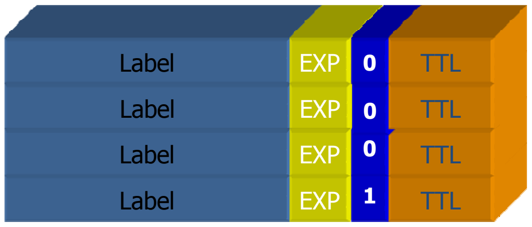
Labels are 32-bit identifiers, broken down as follows:

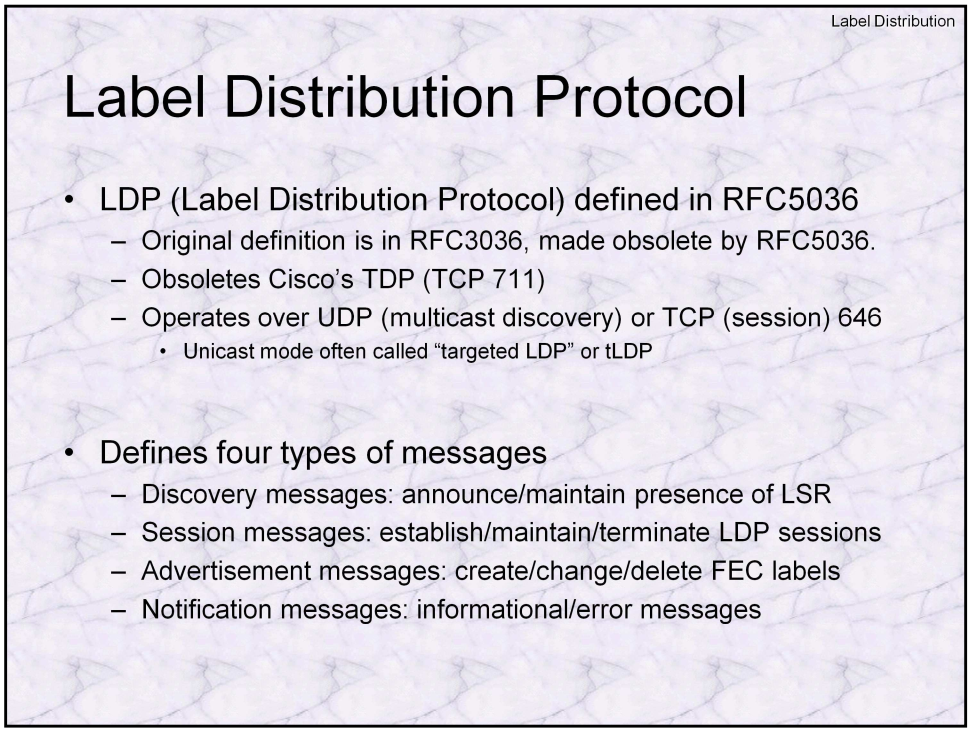
3-bit "Experimental" (RFC4562 – TC)
Time To Live
Label
EXP
B
TTL
20 – bit Identifier
Bottom of stack

- Actual Label occupies 20/32 bits (Labels 0-15: Reserved)
- Experimental bits copy IP ToS (defined as "TC" in RFC5462)
- One bit is reserved for bottom of stack
- TTL bits implement hop count

Labels can be best thought of as "colors" for the packet. Each packet is assigned a 20-bit value by means of which it will be routed. The label additionally contains 3 bits designated as "Experimental", which are currently used for QoS implementations – copying the IP ToS bits into the MPLS label. A fairly recent RFC (**RFC5462**) finally renames the "EXP" to "Traffic Class". Another bit marks the bottom of the label stack (see below), and an 8-bit TTL or hop count value, much like IP.

A packet can have more than one label. This idea is known as label stacking: Labels are simply put one on top of the other, in a stack formation: Last In, First Out. The very first label (at the bottom of the stack) has its "B" bit (Bottom-of-Stack) set to "1".





Label Distribution

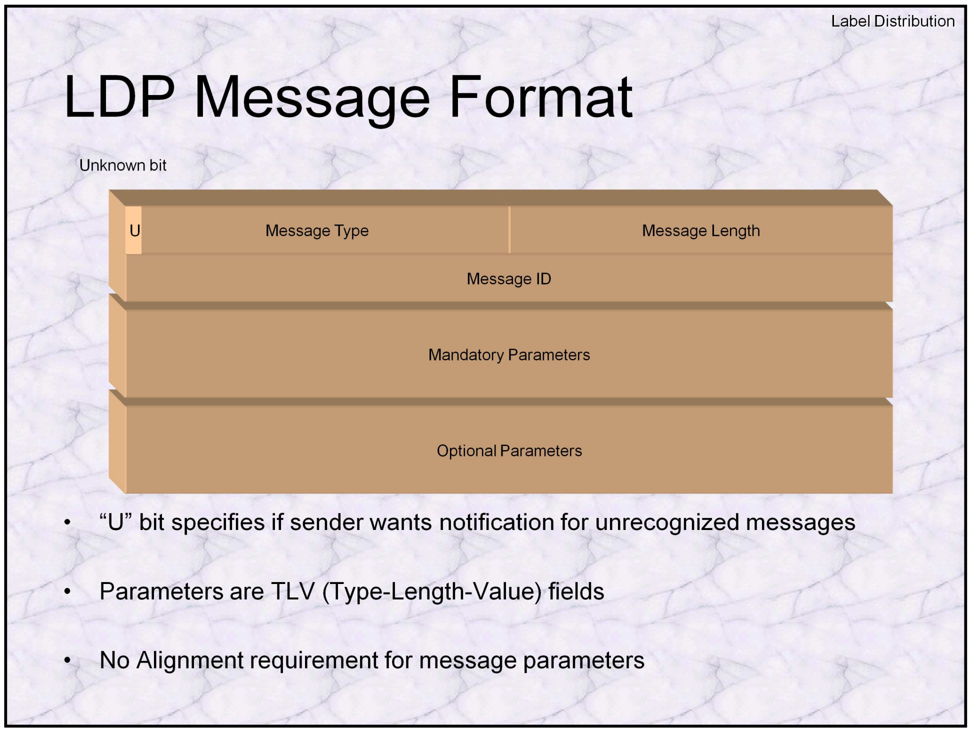
## Label Distribution Protocol

- LDP (Label Distribution Protocol) defined in RFC5036
  - Original definition is in RFC3036, made obsolete by RFC5036.
  - Obsoletes Cisco's TDP (TCP 711)
  - Operates over UDP (multicast discovery) or TCP (session) 646
    - Unicast mode often called "targeted LDP" or tLDP
- Defines four types of messages
  - Discovery messages: announce/maintain presence of LSR
  - Session messages: establish/maintain/terminate LDP sessions
  - Advertisement messages: create/change/delete FEC labels
  - Notification messages: informational/error messages

The most common (but not only) method to distribute Labels in between routers in an MPLS environment is the **Label Distribution Protocol** or **LDP**. This protocol, specified in RFC3036 and revised in RFC5036, is the pretty much the de facto standard, although in some cases Labels may be distributed using RSVP-TE (more on that later).

LDP uses a combination of UDP and TCP. UDP is used for the initial router discovery, via "Hello" messages, which are multicast to 224.0.0.2 – The Multicast "All Routers" group. Once routers establish the identity of their peers (only routers are supposed to reply to this multicast address), unicast TCP sessions my be formed. These are often called "tLDP" sessions (i.e. "targeted LDP").

Prior to LDP, Cisco tried to push its own "Tag Distribution Protocol" (TDP) – A Cisco proprietary solution, that used TCP port 711 for its sessions. This has been deprecated by the introduction of LDP by the IETF.



## LDP Message Format

- "U" bit specifies if sender wants notification for unrecognized messages
- Parameters are TLV (Type-Length-Value) fields
- No Alignment requirement for message parameters

The LDP messages (PDUs) themselves, following the header, may be of variable length. Because they follow a 10-byte header (or other messages), there is no guarantee for the messages to be 32-bit aligned. The message format always follows a simple structure:

The **Message Type** is specified in the first 16-bits of the message. Message types are listed in the table, on the following page.

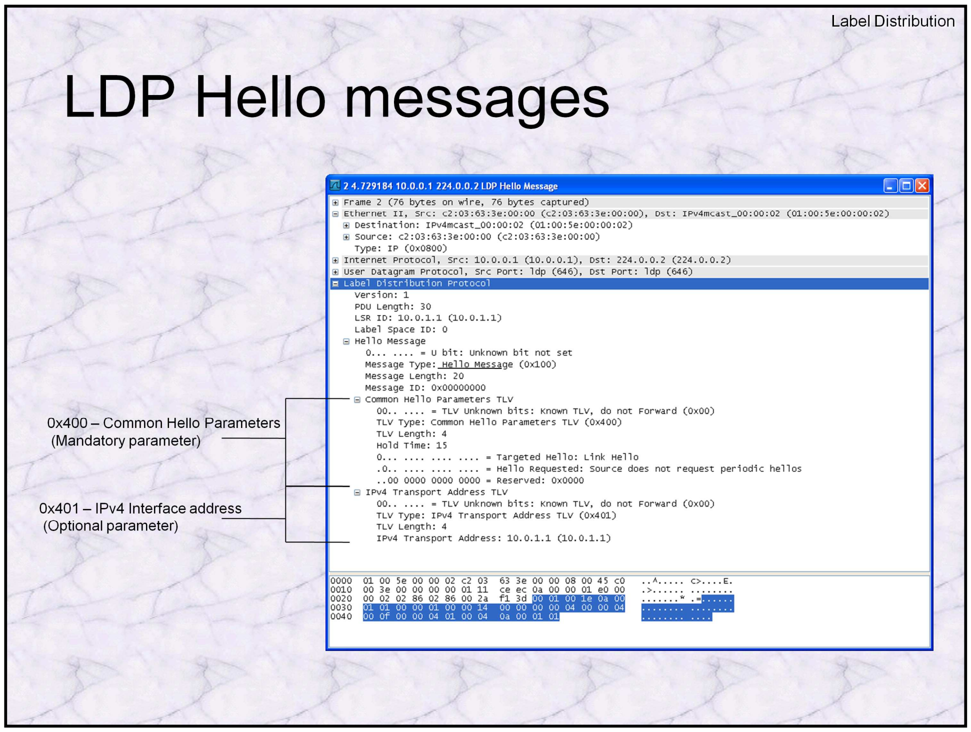
The first bit of the message type is known as the "Unknown bit". If it is set to "1", this means the sender requests notification in case the message is not recognized by the target LSR. This bit is always zero for the RFC-specified messages (since, by default, they MUST be implemented), but may be "1" for vendor extensions or experimental messages.

Messages are of variable length, specified by the **Message Length** field. Messages carry parameters – some mandatory, and some optional (also listed in the table). Parameters are in 'TLV's – Type-Length-Value: that is, a type code, following by a parameter length field, and a value field (of length bytes).

The RFC lists the following messages:

| Message | Code | Purpose | Mandatory TLVs |
|---|---|---|---|
| Notification | 0x0001 | Event notification | Status |
| Hello | 0x0100 | Discovery message | Common Hello params |
| Initialization | 0x0200 | Session establishment | Common Session Params |
| Keep-Alive | 0x0201 | Periodic keep-alive | -- |
| Address | 0x0300 | Advertising Interface | Address List |
| Address-Withdraw | 0x0301 | Removing interface | Address List |
| Label Mapping | 0x0400 | FEC-Label Binding | FEC, Label |
| Label Request | 0x0401 | Request FEC binding | FEC |
| Label Withdraw | 0x0402 | Break FEC-label binding | FEC, Label (optional) |
| Label Release | 0x0403 | Deprecate binding | FEC, Label (optional) |
| Label Abort | 0x0404 | Abort Label-Request | FEC, Message-ID |
| Vendor-Private | 0x3E00- | Open to vendors | |
| Experimental | 0x3F00- | Mostly unused | |

The table above lists the mandatory parameters for each message, but each message may also have optional parameters, as well



Label Distribution

## LDP Hello messages

0x400 – Common Hello Parameters
(Mandatory parameter)

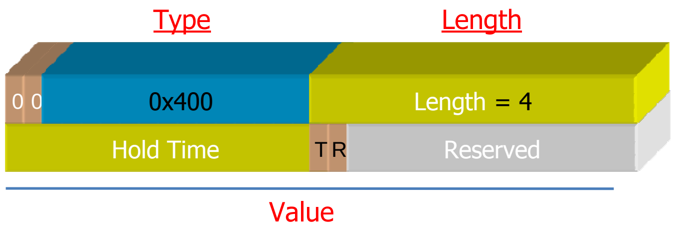
0x401 – IPv4 Interface address
(Optional parameter)

```
2 4.729184 10.0.0.1 224.0.0.2 LDP Hello Message
⊞ Frame 2 (76 bytes on wire, 76 bytes captured)
⊟ Ethernet II, Src: c2:03:63:3e:00:00 (c2:03:63:3e:00:00), Dst: IPv4mcast_00:00:02 (01:00:5e:00:00:02)
  ⊞ Destination: IPv4mcast_00:00:02 (01:00:5e:00:00:02)
  ⊞ Source: c2:03:63:3e:00:00 (c2:03:63:3e:00:00)
    Type: IP (0x0800)
⊞ Internet Protocol, Src: 10.0.0.1 (10.0.0.1), Dst: 224.0.0.2 (224.0.0.2)
⊞ User Datagram Protocol, Src Port: ldp (646), Dst Port: ldp (646)
⊟ Label Distribution Protocol
    Version: 1
    PDU Length: 30
    LSR ID: 10.0.1.1 (10.0.1.1)
    Label Space ID: 0
  ⊟ Hello Message
      0... .... = U bit: Unknown bit not set
      Message Type: Hello Message (0x100)
      Message Length: 20
      Message ID: 0x00000000
    ⊟ Common Hello Parameters TLV
        00.. .... = TLV Unknown bits: Known TLV, do not Forward (0x00)
        TLV Type: Common Hello Parameters TLV (0x400)
        TLV Length: 4
        Hold Time: 15
        0... .... .... .... = Targeted Hello: Link Hello
        .0.. .... .... .... = Hello Requested: Source does not request periodic hellos
        ..00 0000 0000 0000 = Reserved: 0x0000
    ⊟ IPv4 Transport Address TLV
        00.. .... = TLV Unknown bits: Known TLV, do not Forward (0x00)
        TLV Type: IPv4 Transport Address TLV (0x401)
        TLV Length: 4
        IPv4 Transport Address: 10.0.1.1 (10.0.1.1)

0000  01 00 5e 00 00 02 c2 03  63 3e 00 00 08 00 45 c0   ..^..... c>....E.
0010  00 3e 00 00 00 00 01 11  ce ec 0a 00 00 01 e0 00   .>...... ........
0020  00 02 02 86 02 86 00 2a  f1 3d 00 01 00 1e 0a 00   .......* .=......
0030  01 01 00 00 01 00 00 14  00 00 00 00 04 00 00 04   ........ ........
0040  00 0f 00 00 04 01 00 04  0a 00 01 01               ........ ....
```

The basic "Hello" message in LDP is a UDP multicast sent to the 224.0.0.2 ("All Routers on this subnet") group. This is a link-local multicast, meant to advertise the presence of an MPLS and LDP enabled router – the LSR - to any potential peers.

The hello message (0x100) defines as mandatory parameters a special TLV called the **<u>"Common Hello Parameters" (0x401) TLV:</u>**



This TLV has three fields, specifying:

- <u>Hold time</u>**:** Retain hello messages for 0 (default – 15 seconds for link hellos, 45 seconds for targeted hellos – see following), 0xFFFF (forever), or any value in between.

- <u>Targeted hello</u>: If this bit is set, this is a targeted (i.e. unicast) hello. In multicast packets, this bit is obviously never set.

- <u>Reserved:</u> the last 16-bit are currently set to zeros.