Network Working Group   K. Kompella (Juniper)
Internet Draft   P. Pan (Ciena)
draft-ietf-mpls-lsp-ping-05.txt   N. Sheth (Juniper)
Category: Standards Track   D. Cooper (Global Crossing)
Expires: August 2004   G. Swallow (Cisco)
                                                     S. Wadhwa (Juniper)
                                                    R. Bonica (WorldCom)
                                                           February 2004

                     Detecting MPLS Data Plane Failures

                    <draft-ietf-mpls-lsp-ping-05.txt>

Status of this Memo

   This document is an Internet-Draft and is in full conformance with
   all provisions of Section 10 of RFC2026.

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF), its areas, and its working groups. Note that
   other groups may also distribute working documents as Internet-
   Drafts.

   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.''

   The list of current Internet-Drafts can be accessed at
   http://www.ietf.org/ietf/1id-abstracts.txt

   The list of Internet-Draft Shadow Directories can be accessed at
   http://www.ietf.org/shadow.html.

Copyright Notice

Abstract

   This document describes a simple and efficient mechanism that can be
   used to detect data plane failures in Multi-Protocol Label Switching
   (MPLS) Label Switched Paths (LSPs). There are two parts to this
   document: information carried in an MPLS "echo request" and "echo
   reply" for the purposes of fault detection and isolation; and
   mechanisms for reliably sending the echo reply.

Changes since last revision

   (This section to be removed before publication.)

   *** Changed the format of an L2 circuit ID FEC. Added a sender's PE
   address field to uniquely identify the VC ID ***

   Further clarified that an MPLS echo request/reply can be either an
   IPv4 or an IPv6 packet.

   Added format pictures for LDP IPv4/IPv6 prefixes.

   Clarified the section on Receiving an MPLS Echo Request.

Issues

   (This section to be removed before publication.)

   Need to address issues with pinging L3VPN FECs.

   Need to add new FEC type for "type 129" L2 circuits.

1. Introduction

   This document describes a simple and efficient mechanism that can be
   used to detect data plane failures in MPLS LSPs. There are two parts
   to this document: information carried in an MPLS "echo request" and
   "echo reply"; and mechanisms for transporting the echo reply. The
   first part aims at providing enough information to check correct
   operation of the data plane, as well as a mechanism to verify the
   data plane against the control plane, and thereby localize faults.

The second part suggests two methods of reliable reply channels for
the echo request message, for more robust fault isolation.

An important consideration in this design is that MPLS echo requests
follow the same data path that normal MPLS packets would traverse.

MPLS echo requests are meant primarily to validate the data plane,
and secondarily to verify the data plane against the control plane.
Mechanisms to check the control plane are valuable, but are not
covered in this document.

To avoid potential Denial of Service attacks, it is recommended to
regulate the MPLS ping traffic going to the control plane. A rate
limiter should be applied to the well-known UDP port defined below.

1.1. Conventions

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 [KEYWORDS].

1.2. Structure of this document

The body of this memo contains four main parts: motivation, MPLS echo
request/reply packet format, MPLS ping operation, and a reliable
return path. It is suggested that first-time readers skip the actual
packet formats and read the Theory of Operation first; the document
is structured the way it is to avoid forward references.

The last section (reliable return path for RSVP LSPs) may be removed
in a future revision.

2. Motivation

When an LSP fails to deliver user traffic, the failure cannot always
be detected by the MPLS control plane. There is a need to provide a
tool that would enable users to detect such traffic "black holes" or
misrouting within a reasonable period of time; and a mechanism to
isolate faults.

In this document, we describe a mechanism that accomplishes these

goals. This mechanism is modeled after the ping/traceroute paradigm:
ping (ICMP echo request [ICMP]) is used for connectivity checks, and
traceroute is used for hop-by-hop fault localization as well as path
tracing. This document specifies a "ping mode" and a "traceroute"
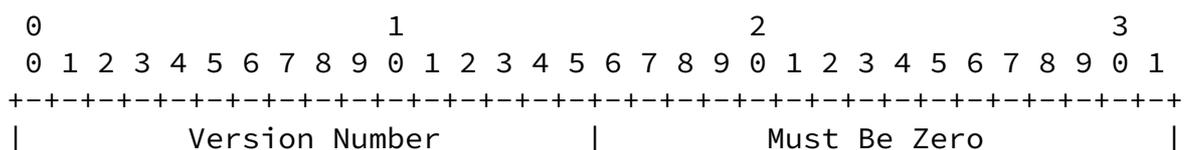mode for testing MPLS LSPs.

The basic idea is to test that packets that belong to a particular
Forwarding Equivalence Class (FEC) actually end their MPLS path on an
LSR that is an egress for that FEC. This document proposes that this
test be carried out by sending a packet (called an "MPLS echo
request") along the same data path as other packets belonging to this
FEC. An MPLS echo request also carries information about the FEC

whose MPLS path is being verified. This echo request is forwarded
just like any other packet belonging to that FEC. In "ping" mode
(basic connectivity check), the packet should reach the end of the
path, at which point it is sent to the control plane of the egress
LSR, which then verifies that it is indeed an egress for the FEC. In
"traceroute" mode (fault isolation), the packet is sent to the
control plane of each transit LSR, which performs various checks that
it is indeed a transit LSR for this path; this LSR also returns
further information that helps check the control plane against the
data plane, i.e., that forwarding matches what the routing protocols
determined as the path.

One way these tools can be used is to periodically ping a FEC to
ensure connectivity. If the ping fails, one can then initiate a
traceroute to determine where the fault lies. One can also
periodically traceroute FECs to verify that forwarding matches the
control plane; however, this places a greater burden on transit LSRs
and thus should be used with caution.

3. Packet Format

An MPLS echo request is a (possibly labelled) IPv4 or IPv6 UDP
packet; the contents of the UDP packet have the following format:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |         Version Number        |         Must Be Zero          |
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Message Type |  Reply mode  |  Return Code  | Return Subcode|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Sender's Handle                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Sequence Number                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   TimeStamp Sent (seconds)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                TimeStamp Sent (microseconds)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 TimeStamp Received (seconds)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              TimeStamp Received (microseconds)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         TLVs ...                              |
.                                                               .
.                                                               .
.                                                               .
```

```
      |                                                         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Version Number is currently 1. (Note: the Version Number is to
be incremented whenever a change is made that affects the ability of
an implementation to correctly parse or process an MPLS echo
request/reply. These changes include any syntactic or semantic
changes made to any of the fixed fields, or to any TLV or sub-TLV
assignment or format that is defined at a certain version number.
The Version Number may not need to be changed if an optional TLV or
sub-TLV is added.)

The Message Type is one of the following:

     Value   Meaning
     -----   -------
        1   MPLS Echo Request
        2   MPLS Echo Reply

The Reply Mode can take one of the following values:

     Value   Meaning

```
        -----   -------
          1     Do not reply
          2     Reply via an IPv4/IPv6 UDP packet
          3     Reply via an IPv4/IPv6 UDP packet with Router Alert
          4     Reply via application level control channel
```

   An MPLS echo request with "Do not reply" may be used for one-way
   connectivity tests; the receiving router may log gaps in the sequence
   numbers and/or maintain delay/jitter statistics. An MPLS echo
   request would normally have "Reply via an IPv4/IPv6 UDP packet"; if
   the normal IP return path is deemed unreliable, one may use "Reply
   via an IPv4/IPv6 UDP packet with Router Alert" (note that this
   requires that all intermediate routers understand and know how to
   forward MPLS echo replies). The echo reply uses the same IP version
   number as the received echo request, i.e., an IPv4 encapsulated echo
   reply is sent in response to an IPv4 encapsulated echo request.

   Any application which supports an IP control channel between its
   control entities may set the Reply Mode to 4 to ensure that replies
   use that same channel. Further definition of this codepoint is
   application specific and thus beyond the scope of this docuemnt.

   Return Codes and Subcodes are described in the next section.

   The Sender's Handle is filled in by the sender, and returned
   unchanged by the receiver in the echo reply (if any). There are no

   semantics associated with this handle, although a sender may find
   this useful for matching up requests with replies.

   The Sequence Number is assigned by the sender of the MPLS echo
   request, and can be (for example) used to detect missed replies.

   The TimeStamp Sent is the time-of-day (in seconds and microseconds,
   wrt the sender's clock) when the MPLS echo request is sent. The
   TimeStamp Received in an echo reply is the time-of-day (wrt the
   receiver's clock) that the corresponding echo request was received.

   TLVs (Type-Length-Value tuples) have the following format:

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

```
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |              Type             |             Length            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                            Value                              |
     .                                                               .
     .                                                               .
     .                                                               .
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
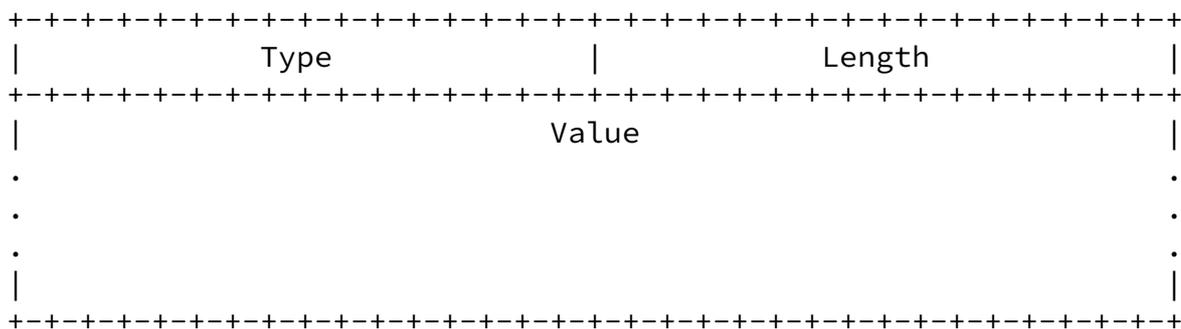
   Types are defined below; Length is the length of the Value field in
   octets. The Value field depends on the Type; it is zero padded to
   align to a four-octet boundary.

```
          Type #                       Value Field
          ------                       -----------
             1                         Target FEC Stack
             2                         Downstream Mapping
             3                         Pad
             4                         Error Code
             5                         Vendor Enterprise Code
```

3.1. Return Codes

   The Return Code is set to zero by the sender. The receiver can set
   it to one of the values listed below. The notation <RSC> refers to
   the Return Subcode. This field is filled in with the stack-depth for
   those codes which specify that. For all other codes the Return
   Subcode MUST be set to zero.

```
          Value    Meaning
          -----    -------
```

```
            0      No return code or return code contained in the Error
                   Code TLV

            1      Malformed echo request received

            2      One or more of the TLVs was not understood

            3      Replying router is an egress for the FEC at stack
```

```
                depth <RSC>

          4     Replying router has no mapping for the FEC at stack
                depth <RSC>

          5     Reserved

          6     Reserved

          7     Reserved

          8     Label switched at stack-depth <RSC>

          9     Label switched but no MPLS forwarding at stack-depth
                <RSC>

         10     Mapping for this FEC is not the given label at stack
                depth <RSC>

         11     No label entry at stack-depth <RSC>

         12     Protocol not associated with interface at FEC stack
                depth <RSC>
```

3.2. Target FEC Stack

   A Target FEC Stack is a list of sub-TLVs. The number of elements is
   determined by the looking at the sub-TLV length fields.

```
       Sub-Type #        Length              Value Field
       ----------        ------              -----------
               1            5                LDP IPv4 prefix
               2           17                LDP IPv6 prefix
               3           20                RSVP IPv4 Session Query
               4           56                RSVP IPv6 Session Query
               5                             Reserved; see Appendix
               6           13                VPN IPv4 prefix
               7           25                VPN IPv6 prefix
               8           14                L2 VPN endpoint
               9           10                L2 circuit ID
```

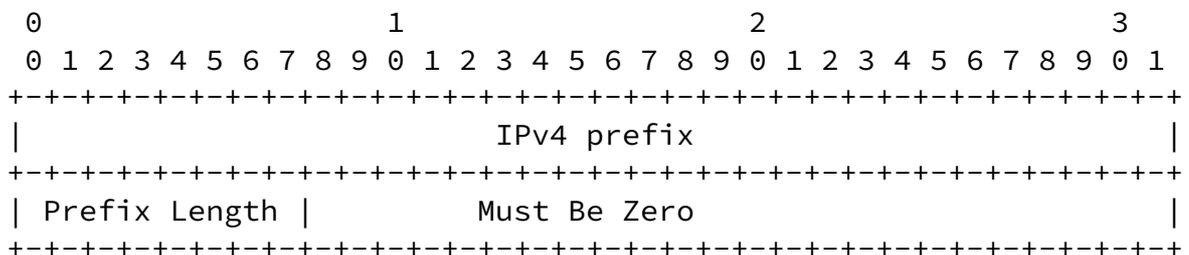   Other FEC Types will be defined as needed.

Note that this TLV defines a stack of FECs, the first FEC element
corresponding to the top of the label stack, etc.

An MPLS echo request MUST have a Target FEC Stack that describes the
FEC stack being tested. For example, if an LSR X has an LDP mapping
for 192.168.1.1 (say label 1001), then to verify that label 1001 does
indeed reach an egress LSR that announced this prefix via LDP, X can
send an MPLS echo request with a FEC Stack TLV with one FEC in it,
namely of type LDP IPv4 prefix, with prefix 192.168.1.1/32, and send
the echo request with a label of 1001.

Say LSR X wanted to verify that a label stack of <1001, 23456> is the
right label stack to use to reach a VPN IPv4 prefix of 10/8 in VPN
foo. Say further that LSR Y with loopback address 192.168.1.1
announced prefix 10/8 with Route Distinguisher RD-foo-Y (which may in
general be different from the Route Distinguisher that LSR X uses in
its own advertisements for VPN foo), label 23456 and BGP nexthop
192.168.1.1. Finally, suppose that LSR X receives a label binding of
1001 for 192.168.1.1 via LDP. X has two choices in sending an MPLS
echo request: X can send an MPLS echo request with a FEC Stack TLV
with a single FEC of type VPN IPv4 prefix with a prefix of 10/8 and a
Route Distinguisher of RD-foo-Y. Alternatively, X can send a FEC
Stack TLV with two FECs, the first of type LDP IPv4 with a prefix of
192.168.1.1/32 and the second of type of IP VPN with a prefix 10/8
with Route Distinguisher of RD-foo-Y. In either case, the MPLS echo
request would have a label stack of <1001, 23456>. (Note: in this
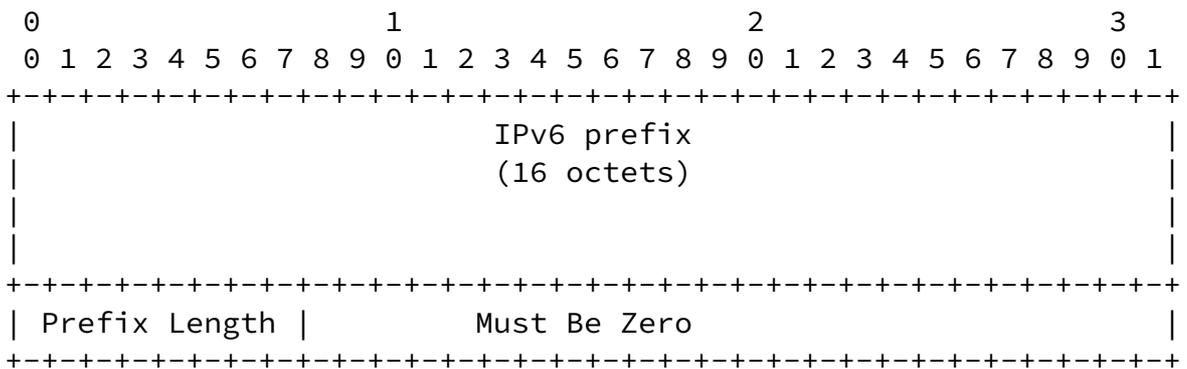example, 1001 is the "outer" label and 23456 is the "inner" label.)

3.2.1. LDP IPv4 Prefix

The value consists of four octets of an IPv4 prefix followed by one
octet of prefix length in bits; the format is given below. The IPv4
prefix is in network byte order; if the prefix is shorter than 32
bits, trailing bits SHOULD be set to zero. See [LDP] for an example
of a Mapping for an IPv4 FEC.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          IPv4 prefix                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Prefix Length |              Must Be Zero                     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
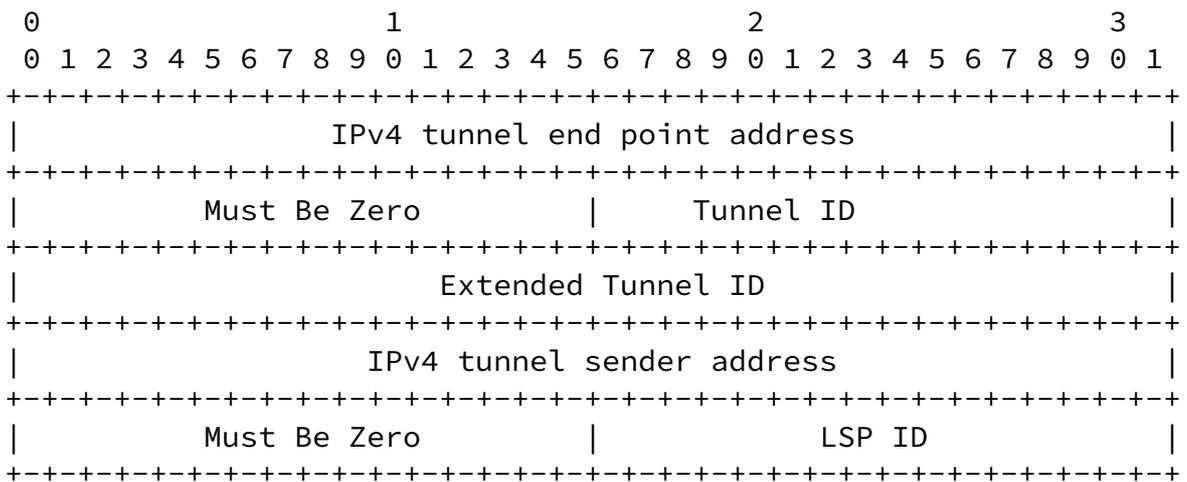
3.2.2. LDP IPv6 Prefix

   The value consists of sixteen octets of an IPv6 prefix followed by
   one octet of prefix length in bits; the format is given below. The
   IPv6 prefix is in network byte order; if the prefix is shorter than
   128 bits, the trailing bits SHOULD be set to zero. See [LDP] for an
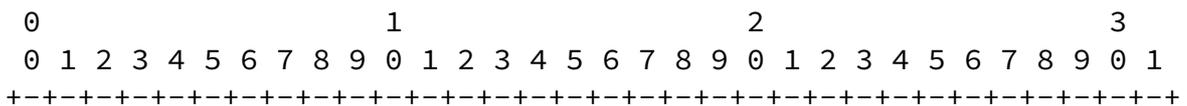   example of a Mapping for an IPv6 FEC.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                          IPv6 prefix                          |
    |                          (16 octets)                          |
    |                                                               |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Prefix Length |             Must Be Zero                      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.3. RSVP IPv4 Session

   The value has the format below. The value fields are taken from
   [RFC3209, sections 4.6.1.1 and 4.6.2.1].

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                  IPv4 tunnel end point address                |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Must Be Zero          |            Tunnel ID          |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                       Extended Tunnel ID                      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   IPv4 tunnel sender address                  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Must Be Zero          |              LSP ID          |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
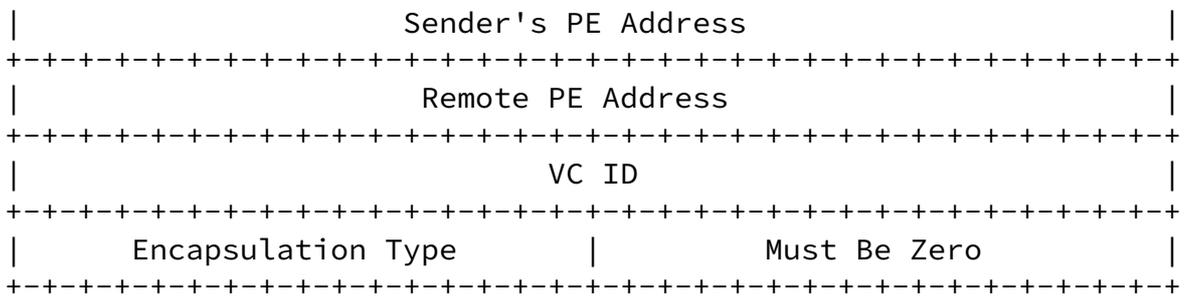
3.2.4. RSVP IPv6 Session

   The value has the format below. The value fields are taken from
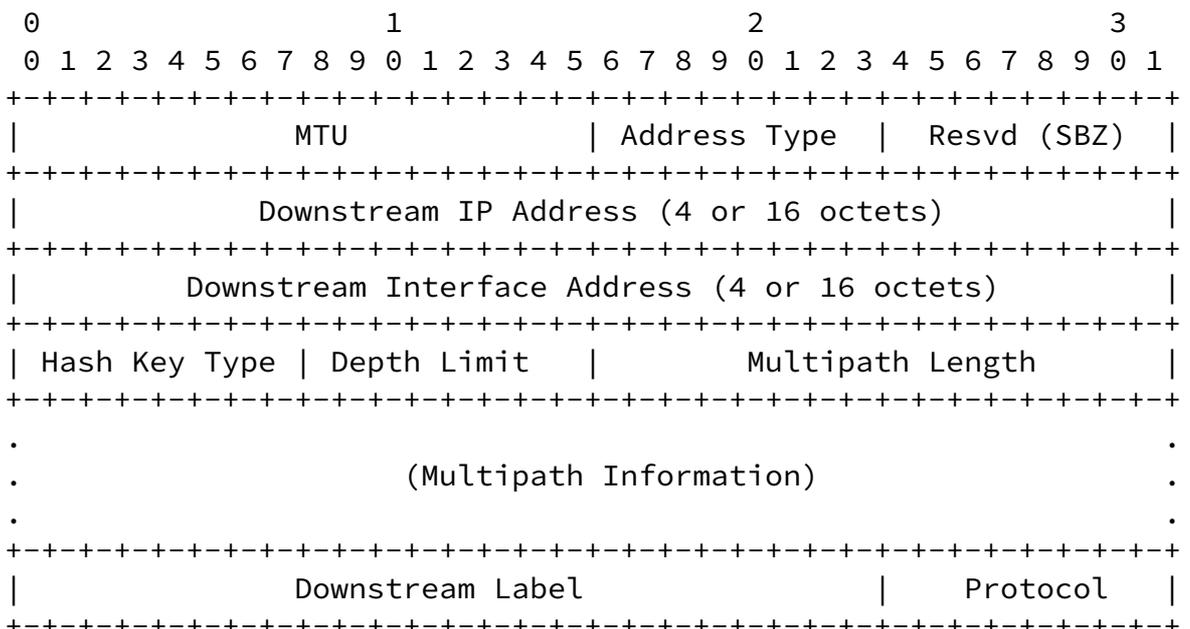   [RFC3209, sections 4.6.1.2 and 4.6.2.2].

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
           |                 IPv6 tunnel end point address            |
           |                                                           |
```

```
           |                                                           |
           |                                                           |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |            Must Be Zero         |          Tunnel ID      |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |                      Extended Tunnel ID                   |
           |                                                           |
           |                                                           |
           |                                                           |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |                   IPv6 tunnel sender address              |
           |                                                           |
           |                                                           |
           |                                                           |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
           |            Must Be Zero         |            LSP ID       |
           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.5. VPN IPv4 Prefix

   The value field consists of the Route Distinguisher advertised with
   the VPN IPv4 prefix, the IPv4 prefix and a prefix length, as follows:

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        |                     Route Distinguisher                      |
        |                        (8 octets)                            |
        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        |                        IPv4 prefix                           |
        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        | Prefix Length |                Must Be Zero                  |
        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.6. VPN IPv6 Prefix

   The value field consists of the Route Distinguisher advertised with
   the VPN IPv6 prefix, the IPv6 prefix and a prefix length, as follows:

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                    Route Distinguisher                       |
       |                        (8 octets)                            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                       IPv6 prefix                            |
       |                                                              |
       |                                                              |
```

```
       |                                                              |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       | Prefix Length |              Must Be Zero                    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

[3.2.7](). L2 VPN Endpoint

   The value field consists of a Route Distinguisher (8 octets), the
   sender (of the ping)'s CE ID (2 octets), the receiver's CE ID (2
   octets), and an encapsulation type (2 octets), formatted as follows:

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                    Route Distinguisher                       |
       |                        (8 octets)                            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |         Sender's CE ID        |       Receiver's CE ID       |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |       Encapsulation Type      |          Must Be Zero        |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

[3.2.8](). L2 Circuit ID

   The value field consists of the sender's PE address (the source
   address of the targetted LDP session), the remote PE address (the
   destination address of the targetted LDP session), a VC ID and an
   encapsulation type, as follows:

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
|                    Sender's PE Address                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                    Remote PE Address                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                          VC ID                                |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Encapsulation Type       |          Must Be Zero         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.3. Downstream Mapping

   The Downstream Mapping object is an optional TLV. Only one
   Downstream Mapping request may appear in and echo request. The
   presence of a Downstream Mapping object is a request that Downstream
   Mapping objects be included in the echo reply. If the replying
   router is the destination of the FEC, then a Downstream Mapping TLV

   SHOULD NOT be included in the echo reply. Otherwise Downstream
   Mapping objects SHOULD include a Downstream Mapping object for each
   interface over which this FEC could be forwarded.

   The Length is 16 + M + 4*N octets, where M is the Multipath Length,
   and N is the number of Downstream Labels. The Value field of a
   Downstream Mapping has the following format:

```
        0                   1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |               MTU             | Address Type  |  Resvd (SBZ)  |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |           Downstream IP Address (4 or 16 octets)             |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |        Downstream Interface Address (4 or 16 octets)         |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       | Hash Key Type | Depth Limit   |       Multipath Length       |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       .                                                              .
       .                  (Multipath Information)                     .
       .                                                              .
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                 Downstream Label              |   Protocol    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
        .                                                      .
        .                                                      .
        .                                                      .
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                 Downstream Label              |   Protocol    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Maximum Transmission Unit (MTU)

      The MTU is the largest MPLS frame (including label stack) that
      fits on the interface to the Downstream LSR.

   Address Type

      The Address Type indicates if the interface is numbered or
      unnumbered and is set to one of the following values:

                   Type #   Address Type
                   ------   ------------
                      1   IPv4
                      2   Unnumbered
                      3   IPv6

      The field marked SBZ SHOULD be set to zero when sending and
      SHOULD be ignored on receipt.

   Downstream IP Address and Downstream Interface Address

      If the interface to the downstream LSR is numbered, then the
      Address Type MUST be set to IPv4 or IPv6, the Downstream IP
      Address MUST be set to either the downstream LSR's Router ID or
      the interface address of the downstream LSR, and the Downstream
      Interface Address MUST be set to the downstream LSR's interface
      address.

      If the interface to the downstream LSR is unnumbered, the Address
      Type MUST be Unnumbered, the Downstream IP Address MUST be the
      downstream LSR's Router ID (4 octets), and the Downstream
      Interface Address MUST be set to the index assigned by the
      upstream LSR to the interface.

Multipath Length

     The length in octets of the Multipath Information.

Downstream Label(s)

     The set of labels in the label stack as it would have appeared if
     this router were forwarding the packet through this interface.
     Any Implicit Null labels are explicitly inluded. Labels are
     treated as numbers, i.e. they are right justified in the field.

Protocol

     The Protocol is taken from the following table:

          Protocol #   Signaling Protocol
          ----------   ------------------
                  0   Unknown
                  1   Static
                  2   BGP
                  3   LDP
                  4   RSVP-TE
                  5   Reserved; see Appendix

     [T](T)he notion of "downstream router" and "downstream interface"
     should be explained. Consider an LSR X. If a packet that was
     originated with TTL n>1 arrived with outermost label L at LSR X,
     X must be able to compute which LSRs could receive the packet if
     it was originated with TTL=n+1, over which interface the request
     would arrive and what label stack those LSRs would see. (It is

Kompella et al   Standards Track   [Page 13]

     outside the scope of this document to specify how this
     computation is done.) The set of these LSRs/interfaces are the
     downstream routers/interfaces (and their corresponding labels)
     for X with respect to L. Each pair of downstream router and
     interface requires a separate Downstream Mapping to be added to
     the reply. (Note that there are multiple Downstream Label fields
     in each TLV as the incoming label L may be swapped with a label
     stack.)

     The case where X is the LSR originating the echo request is a
     special case. X needs to figure out what LSRs would receive the

MPLS echo request for a given FEC Stack that X originates with
TTL=1.

The set of downstream routers at X may be alternative paths (see
the discussion below on ECMP) or simultaneous paths (e.g., for
MPLS multicast). In the former case, the Multipath sub-field is
used as a hint to the sender as to how it may influence the
choice of these alternatives. The "No of Multipaths" is the
number of IP Address/Next Label fields. The Hash Key Type is
taken from the following table:

```
    Key    Type                  Multipath Information
    ---    ----------------      ---------------------
     0     no multipath          (empty; M = 0)
     1     label                 labels
     2     IP address            IP addresses
     3     label range           low/high label pairs
     4     IP address range      low/high address pairs
     5     no more labels        (empty; M = 0)
     6     All IP addresses      (empty; M = 0)
     7     no match              (empty; M = 0)
     8     Bit-masked IPv4       IP address prefix and bit mask
             address set
     9     Bit-masked label set  Label prefix and bit mask
```

Type 0 indicates that all packets will be forwarded out this one
interface.

Types 1, 2, 3, 4, 8 and 9 specify that the supplied Multipath
Information will serve to execise this path.

Types 5 and 6 are TBD.

Type 7 indicates that no matches are possible given the Multipath
Information in the received DS mapping information.

  Depth Limit

The Depth Limit is applicable only to a label stack, and is the
maximum number of labels considered in the hash; this SHOULD be
set to zero if unspecified or unlimited.

Multipath Information

   The multipath information encodes labels or addresses which will
   exercise this path. The multipath informaiton depends on the
   hash key type. The contents of the field are shown in the table
   above. IP addresses are drawn from the range 127/8. Labels are
   treated as numbers, i.e. they are right justified in the field.
   Label and Address pairs MUST NOT overlap and MUST be in ascending
   sequence.

   Hash key 8 allows a denser encoding of IP address. The IPv4
   prefix is formatted as a base IPv4 address with the non-prefix
   low order bits set to zero. The maximum prefix length is 27.
   Following the prefix is a mask of length $2^{(32-prefix\ length)}$
   bits. Each bit set to one represents a valid address. The
   address is the base IPv4 address plus the position of the bit in
   the mask where the bits are numbered left to right begining with
   zero.

   Hash key 9 allows a denser encoding of Labels. The label prefix
   is formatted as a base label value with the non-prefix low order
   bits set to zero. The maximum prefix (including leading zeros
   due to encoding) length is 27. Following the prefix is a mask of
   length $2^{(32-prefix\ length)}$ bits. Each bit set to one represents
   a valid Label. The label is the base label plus the position of
   the bit in the mask where the bits are numbered left to right
   begining with zero.

   If the received DS mapping information is non-null the labels and
   IP addresses MUST be picked from the set provided or the Hash Key
   Type MUST be set to 7.

   For example, suppose LSR X at hop 10 has two downstream LSRs Y
   and Z for the FEC in question. X could return Hash Key Type 4,
   with low/high IP addresses of 1.1.1.1->1.1.1.255 for downstream
   LSR Y and 2.1.1.1->2.1.1.255 for downstream LSR Z. The head end
   reflects this information to LSR Y. Y, which has three
   downstream LSRs U, V and W, computes that 1.1.1.1->1.1.1.127
   would go to U and 1.1.1.128-> 1.1.1.255 would go to V. Y would
   then respond with 3 Downstream Mappings: to U, with Hash Key Type
   4 (1.1.1.1->1.1.1.127); to V, with Hash Key Type 4
   (1.1.1.127->1.1.1.255); and to W, with Hash Key Type 7.

3.4. Pad TLV

   The value part of the Pad TLV contains a variable number (>= 1) of
   octets. The first octet takes values from the following table; all
   the other octets (if any) are ignored. The receiver SHOULD verify
   that the TLV is received in its entirety, but otherwise ignores the
   contents of this TLV, apart from the first octet.

                   Value   Meaning
                   -----   -------
                       1   Drop Pad TLV from reply
                       2   Copy Pad TLV to reply
                   3-255   Reserved for future use

3.5. Error Code

   The Error Code TLV is currently not defined; its purpose is to
   provide a mechanism for a more elaborate error reporting structure,
   should the reason arise.

3.6. Vendor Enterprise Code

   The Length is always 4; the value is the SMI Enterprise code, in
   network octet order, of the vendor with a Vendor Private extension to
   any of the fields in the fixed part of the message, in which case
   this TLV MUST be present. If none of the fields in the fixed part of
   the message have vendor private extensions, this TLV is OPTIONAL.

4. Theory of Operation

   An MPLS echo request is used to test a particular LSP. The LSP to be
   tested is identified by the "FEC Stack"; for example, if the LSP was
   set up via LDP, and is to an egress IP address of 10.1.1.1, the FEC
   stack contains a single element, namely, an LDP IPv4 prefix sub-TLV
   with value 10.1.1.1/32. If the LSP being tested is an RSVP LSP, the
   FEC stack consists of a single element that captures the RSVP Session
   and Sender Template which uniquely identifies the LSP.

   FEC stacks can be more complex. For example, one may wish to test a
   VPN IPv4 prefix of 10.1/8 that is tunneled over an LDP LSP with
   egress 10.10.1.1. The FEC stack would then contain two sub-TLVs, the
   first being a VPN IPv4 prefix, and the second being an LDP IPv4
   prefix. If the underlying (LDP) tunnel were not known, or was
   considered irrelevant, the FEC stack could be a single element with
   just the VPN IPv4 sub-TLV.

When an MPLS echo request is received, the receiver is expected to do

a number of tests that verify that the control plane and data plane
are both healthy (for the FEC stack being pinged), and that the two
planes are in sync.

4.1. Dealing with Equal-Cost Multi-Path (ECMP)

LSPs need not be simple point-to-point tunnels. Frequently, a single
LSP may originate at several ingresses, and terminate at several
egresses; this is very common with LDP LSPs. LSPs for a given FEC
may also have multiple "next hops" at transit LSRs. At an ingress,
there may also be several different LSPs to choose from to get to the
desired endpoint. Finally, LSPs may have backup paths, detour paths
and other alternative paths to take should the primary LSP go down.

To deal with the last two first: it is assumed that the LSR sourcing
MPLS echo requests can force the echo request into any desired LSP,
so choosing among multiple LSPs at the ingress is not an issue. The
problem of probing the various flavors of backup paths that will
typically not be used for forwarding data unless the primary LSP is
down will not be addressed here.

Since the actual LSP and path that a given packet may take may not be
known a priori, it is useful if MPLS echo requests can exercise all
possible paths. This, while desirable, may not be practical, because
the algorithms that a given LSR uses to distribute packets over
alternative paths may be proprietary.

To achieve some degree of coverage of alternate paths, there is a
certain lattitude in choosing the destination IP address and source
UDP port for an MPLS echo request. This is clearly not sufficient;
in the case of traceroute, more lattitude is offered by means of the
"Multipath Exercise" sub-TLV of the Downstream Mapping TLV. This is
used as follows. An ingress LSR periodically sends an MPLS
traceroute message to determine whether there are multipaths for a
given LSP. If so, each hop will provide some information how each of
its downstreams can be exercised. The ingress can then send MPLS
echo requests that exercise these paths. If several transit LSRs
have ECMP, the ingress may attempt to compose these to exercise all
possible paths. However, full coverage may not be possible.

. Sending an MPLS Echo Request

   An MPLS echo request is a (possibly) labelled UDP packet. 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)
   address from 127/8; the IP TTL is set to 1. The source UDP port is
   chosen by the sender; the destination UDP port is set to 3503
   (assigned by IANA for MPLS echo requests). The Router Alert option

   is set in the IP header.

   If the echo request is labelled, one may (depending on what is being
   pinged) set the TTL of the innermost label to 1, to prevent the ping
   request going farther than it should. Examples of this include
   pinging a VPN IPv4 or IPv6 prefix, an L2 VPN end point or an L2
   circuit ID. This can also be accomplished by inserting a router
   alert label above this label; however, this may lead to the undesired
   side effect that MPLS echo requests take a different data path than
   actual data.

   In "ping" mode (end-to-end connectivity check), the TTL in the
   outermost label is set to 255. In "traceroute" mode (fault isolation
   mode), the TTL is set successively to 1, 2, ...

   The sender chooses a Sender's Handle, and a Sequence Number. When
   sending subsequent MPLS echo requests, the sender SHOULD increment
   the sequence number by 1. However, a sender MAY choose to send a
   group of echo requests with the same sequence number to improve the
   chance of arrival of at least one packet with that sequence number.

   The TimeStamp Sent is set to the time-of-day (in seconds and
   microseconds) that the echo request is sent. The TimeStamp Received
   is set to zero.

   An MPLS echo request MUST have a FEC Stack TLV. Also, the Reply Mode
   must be set to the desired reply mode; the Return Code and Subcode
   are set to zero.

   In the "traceroute" mode, the echo request SHOULD contain one or more
   Downstream Mapping TLVs. For TTL=1, all the downstream routers (and
   corresponding labels) for the sender with respect to the FEC Stack
   being pinged SHOULD be sent in the echo request. For n>1, the

Downstream Mapping TLVs from the echo reply for TTL=(n-1) are copied
to the echo request with TTL=n; the sender MAY choose to reduce the
size of a "Downstream Multipath Mapping TLV" when copying into the
next echo request as long as the Hash Key Type matching the label or
IP address used to exercise the current MP is still present.

4.3. Receiving an MPLS Echo Request

An LSR X that receives an MPLS echo request first parses the packet
to ensure that it is a well-formed packet, and that the TLVs that are
not marked "Ignore" are understood. If not, X SHOULD send an MPLS
echo reply with the Return Code set to "Malformed echo request
received" or "TLV not understood" (as appropriate), and the Subcode
set to zero. In the latter case, the misunderstood TLVs (only) are
included in the reply.

If the echo request is good, X notes the interface I over which the
echo was received, and the label stack with which it came.

X matches up the labels in the received label stack with the FECs
contained in the FEC stack. The matching is done beginning at the
bottom of both stacks, and working up. For reporting purposes the
bottom of stack is consided to be stack-depth of 1. This is to
establish an absolute reference for the case where the stack may have
more labels than are in the FEC stack.

If there are more FECs than labels, the extra FECs are assumed to
correspond to Implicit Null Labels. Thus for the processing below,
there is never the case where there is a FEC with no corresponding
label. Further the label operation associated with an assumed Null
Label is 'pop and continue processing'.

Note: in all the error codes listed in this draft a stack-depth of 0
means "no value specified". This allows compatibility with existing
implementations which do not use the Return Subcode field.

X sets a variable, call it current-stack-depth, to the number of
labels in the received label stack. Processing now continues with
the following steps:

1. Check if there is a FEC corresponding to the current-stack-
    depth. If there is, go to step 2. If not, check if the label is

valid on interface I. If it is, continue with step 4. Otherwise
X MUST send an MPLS echo reply with a Return Code 11, "No label
entry at stack-depth" and a Return Subcode set to current-stack-
depth.

2. Check the FEC at the current-stack-depth to determine what
    protocol would be used to advertise it. If it can determine that
    no protocol associated with interface I, would have advertised a
    FEC of that FEC-Type, X MUST send an MPLS echo reply with a
    Return Code 12, "Protocol not associated with interface at FEC
    stack-depth" and a Return Subcode set to current-stack-depth.

3. Check that the mapping for the FEC at the current-stack-depth is
    the corresponding label.

    If no mapping for the FEC exists, X MUST send an MPLS echo reply
    with a Return Code 4, "Replying router has no mapping for the FEC
    at stack-depth" and a Return Subcode set to current- stack-depth.

    If a mapping is found, but the mapping is not the corresponding
    label, X MUST send an MPLS echo reply with a Return Code 10,
    "Mapping for this FEC is not the given label at stack-depth" and

Kompella et al   Standards Track   [Page 19]

    a Return Subcode set to current-stack-depth.

4. X determines the label operation. If the operation is to pop and
    continue processing, X checks the current-stack-depth. If it is
    one, X MUST send an MPLS echo reply with a Return Code 3,
    "Replying router is an egress for the FEC at stack depth" and a
    Return Subcode set to one. Otherwise, X decrements current-stack-
    depth and goes back to step 1.

    If the label operation is pop and switch based on the popped
    label, X then checks if it is valid to forward a labelled packet.
    If it is, X MUST send an MPLS echo reply with a Return Code 8,
    "Label switched at stack-depth" and a Return Subcode set to
    current-stack-depth. If it is not valid to forward a labelled
    packet, X MUST send an MPLS echo reply with a Return Code 9,
    "Label switched but no MPLS forwarding at stack-depth" and a
    Return Subcode set to current-stack-depth. This return code is
    sent even if current-stack-depth is one.

If the label operation is swap, X MUST send an MPLS echo reply
with a Return Code 8, "Label switched at stack-depth" and a
Return Subcode set to current-stack-depth.

If the MPLS echo request contains a downstream mapping TLV, and the
MPLS echo reply has either a Return Code of 8, or a Return Code of 9
with a Return Subcode of 1 then Downstream mapping TLVs SHOULD be
included for each multipath.

X uses the procedure in the next subsection to send the echo reply.

4.4. Sending an MPLS Echo Reply

An MPLS echo reply is a UDP packet. It MUST ONLY be sent in response
to an MPLS echo request. The source IP address is a routable address
of the replier; the source port is the well-known UDP port for MPLS
ping. The destination IP address and UDP port are copied from the
source IP address and UDP port of the echo request. The IP TTL is
set to 255. If the Reply Mode in the echo request is "Reply via an
IPv4 UDP packet with Router Alert", then the IP header MUST contain
the Router Alert IP option. If the reply is sent over an LSP, the
topmost label MUST in this case be the Router Alert label (1) (see
[LABEL-STACK]).

The format of the echo reply is the same as the echo request. The
Sender's Handle, the Sequence Number and TimeStamp Sent are copied
from the echo request; the TimeStamp Received is set to the time-of-
day that the echo request is received (note that this information is
most useful if the time-of-day clocks on the requestor and the

replier are synchronized). The FEC Stack TLV from the echo request
MAY be copied to the reply.

The replier MUST fill in the Return Code and Subcode, as determined
in the previous subsection.

If the echo request contains a Pad TLV, the replier MUST interpret
the first octet for instructions regarding how to reply.

If the echo request contains a Downstream Mapping TLV, the replier
SHOULD compute its downstream routers and corresponding labels for
the incoming label, and add Downstream Mapping TLVs for each one to

the echo reply it sends back.

4.5. Receiving an MPLS Echo Reply

   An LSR X should only receive an MPLS Echo Reply in response to an
   MPLS Echo Request that it sent. Thus, on receipt of an MPLS Echo
   Reply, X should parse the packet to assure that it is well-formed,
   then attempt to match up the Echo Reply with an Echo Request that it
   had previously sent, using the destination UDP port and the Sender's
   Handle. If no match is found, then X jettisons the Echo Reply;
   otherwise, it checks the Sequence Number to see if it matches. Gaps
   in the Sequence Number MAY be logged and SHOULD be counted. Once an
   Echo Reply is received for a given Sequence Number (for a given UDP
   port and Handle), the Sequence Number for subsequent Echo Requests
   for that UDP port and Handle SHOULD be incremented.

   If the Echo Reply contains Downstream Mappings, and X wishes to
   traceroute further, it SHOULD copy the Downstream Mappings into its
   next Echo Request (with TTL incremented by one).

4.6. Non-compliant Routers

   If the egress for the FEC Stack being pinged does not support MPLS
   ping, then no reply will be sent, resulting in possible "false
   negatives". If in "traceroute" mode, a transit LSR does not support
   MPLS ping, then no reply will be forthcoming from that LSR for some
   TTL, say n. The LSR originating the echo request SHOULD try sending
   the echo request with TTL=n+1, n+2, ..., n+k in the hope that some
   transit LSR further downstream may support MPLS echo requests and
   reply. In such a case, the echo request for TTL>n MUST NOT have
   Downstream Mapping TLVs, until a reply is received with a Downstream
   Mapping.

Kompella et al   Standards Track   [Page 21]

Internet Draft   Detecting MPLS Data Plane Failures   February 2004

Normative References

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

   [LABEL-STACK] Rosen, E., et al, "MPLS Label Stack Encoding", RFC
      3032, January 2001.

   [RSVP] Braden, R. (Editor), et al, "Resource ReSerVation protocol
      (RSVP) -- Version 1 Functional Specification," RFC 2205,
      September 1997.

   [RSVP-REFRESH] Berger, L., et al, "RSVP Refresh Overhead Reduction
      Extensions", RFC 2961, April 2001.

   [RSVP-TE] Awduche, D., et al, "RSVP-TE: Extensions to RSVP for LSP
      tunnels", RFC 3209, December 2001.

Informative References

   [ICMP] Postel, J., "Internet Control Message Protocol", RFC 792.

   [LDP] Andersson, L., et al, "LDP Specification", RFC 3036, January
      2001.

Security Considerations

   There are at least two approaches to attacking LSRs using the
   mechanisms defined here. One is a Denial of Service attack, by
   sending MPLS echo requests/replies to LSRs and thereby increasing
   their workload. The other is obfuscating the state of the MPLS data
   plane liveness by spoofing, hijacking, replaying or otherwise
   tampering with MPLS echo requests and replies.

   Authentication will help reduce the number of seemingly valid MPLS
   echo requests, and thus cut down the Denial of Service attacks;
   beyond that, each LSR must protect itself.

   Authentication sufficiently addresses spoofing, replay and most
   tampering attacks; one hopes to use some mechanism devised or
   suggested by the RPSec WG. It is not clear how to prevent hijacking
   (non-delivery) of echo requests or replies; however, if these
   messages are indeed hijacked, MPLS ping will report that the data
   plane isn't working as it should.

   It doesn't seem vital (at this point) to secure the data carried in

MPLS echo requests and replies, although knowledge of the state of
   the MPLS data plane may be considered confidential by some.

5. IANA Considerations

   The TCP and UDP port number 3503 has been allocated by IANA for LSP
   echo requests and replies.

   The following sections detail the new name spaces to be managed by
   IANA. For each of these name spaces, the space is divided into
   assignment ranges; the following terms are used in describing the
   procedures by which IANA allocates values: "Standards Action" (as
   defined in [IANA]); "Expert Review" and "Vendor Private Use".

   Values from "Expert Review" ranges MUST be registered with IANA, and
   MUST be accompanied by an Experimental RFC that describes the format
   and procedures for using the code point.

   Values from "Vendor Private" ranges MUST NOT be registered with IANA;
   however, the message MUST contain an enterprise code as registered
   with the IANA SMI Network Management Private Enterprise Codes. For
   each name space that has a Vendor Private range, it must be specified
   where exactly the SMI Enterprise Code resides; see below for
   examples. In this way, several enterprises (vendors) can use the
   same code point without fear of collision.

5.1. Message Types, Reply Modes, Return Codes

   It is requested that IANA maintain registries for Message Types,
   Reply Modes, Return Codes and Return Subcodes. Each of these can
   take values in the range 0-255. Assignments in the range 0-191 are
   via Standards Action; assignments in the range 192-251 are made via
   Expert Review; values in the range 252-255 are for Vendor Private
   Use, and MUST NOT be allocated.

   If any of these fields fall in the Vendor Private range, a top-level
   Vendor Enterprise Code TLV MUST be present in the message.

5.2. TLVs

   It is requested that IANA maintain registries for the Type field of
   top-level TLVs as well as for sub-TLVs. The valid range for each of
   these is 0-65535. Assignments in the range 0-32767 are made via
   Standards Action; assignments in the range 32768-64511 are made via
   Expert Review; values in the range 64512-65535 are for Vendor Private
   Use, and MUST NOT be allocated.

    If a TLV or sub-TLV has a Type that falls in the range for Vendor
    Private Use, the Length MUST be at least 4, and the first four octets
    MUST be that vendor's SMI Enterprise Code, in network octet order.
    The rest of the Value field is private to the vendor.

Acknowledgments

    This document is the outcome of many discussions among many people,
    that include Manoj Leelanivas, Paul Traina, Yakov Rekhter, Der-Hwa
    Gan, Brook Bailey, Eric Rosen and Ina Minei.

    The description of the Multipath Information sub-field of the
    Downstream Mapping TLV was adapted from text suggested by Curtis
    Villamizar.

Appendix

    This appendix specifies non-normative aspects of detecting MPLS data
    plane liveness.

5.1. CR-LDP FEC

    This section describes how a CR-LDP FEC can be included in an Echo
    Request using the following FEC subtype:

        Sub-Type #   Length   Value Field
        ----------   ------   -------------
             5   6   CR-LDP LSP ID

    The value consists of the LSPID of the LSP being pinged. An LSPID is
    a four octet IPv4 address (a local address on the ingress LSR, for
    example, the Router ID) plus a two octet identifier that is unique
    per LSP on a given ingress LSR.

        0   1   2   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+
       | Ingress LSR Router ID |
       +-+
       | Must Be Zero | LSP ID |

```
        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
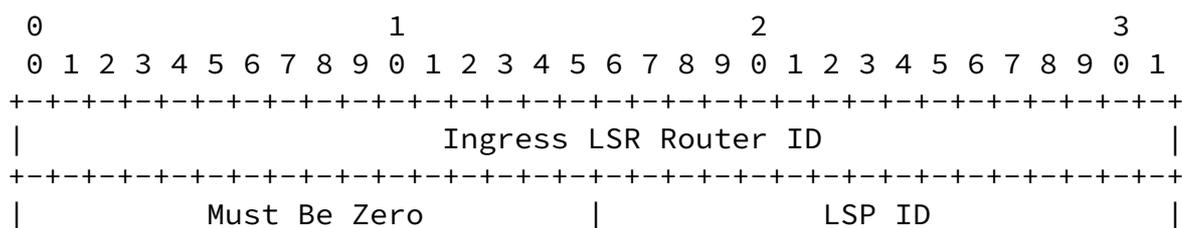
5.2. Downstream Mapping for CR-LDP

   If a label in a Downstream Mapping was learned via CR-LDP, the
   Protocol field in the Mapping TLV can use the following entry:

           Protocol #   Signaling Protocol
           ----------   ------------------
                   5   CR-LDP

Authors' Addresses

   Kireeti Kompella
   Nischal Sheth
   Juniper Networks
   1194 N.Mathilda Ave
   Sunnyvale, CA 94089
   e-mail: kireeti@juniper.net
   e-mail: nsheth@juniper.net

   Ping Pan
   Ciena
   10480 Ridgeview Court
   Cupertino, CA 95014
   e-mail: ppan@ciena.com
   phone: +1 408.366.4700

   Dave Cooper
   Global Crossing
   960 Hamlin Court
   Sunnyvale, CA 94089
   email: dcooper@gblx.net
   phone: +1 916.415.0437

   George Swallow

    Cisco Systems, Inc.
    250 Apollo Drive
    Chelmsford, MA 01824
    e-mail: swallow@cisco.com
    phone: +1 978.497.8143

    Sanjay Wadhwa
    Juniper Networks
    10 Technology Park Drive
    Westford, MA 01886-3146
    email: swadhwa@unispherenetworks.com
    phone: +1 978.589.0697

    Ronald P. Bonica
    WorldCom
    22001 Loudoun County Pkwy
    Ashburn, Virginia, 20147
    email: ronald.p.bonica@wcom.com
    phone: +1 703.886.1681

Intellectual Property Rights Notices

    The IETF takes no position regarding the validity or scope of any
    intellectual property or other rights that might be claimed to
    pertain to the implementation or use of the technology described in
    this document or the extent to which any license under such rights
    might or might not be available; neither does it represent that it
    has made any effort to identify any such rights. Information on the
    IETF's procedures with respect to rights in standards-track and
    standards-related documentation can be found in BCP-11. Copies of
    claims of rights made available for publication and any assurances of
    licenses to be made available, or the result of an attempt made to
    obtain a general license or permission for the use of such
    proprietary rights by implementors or users of this specification can
    be obtained from the IETF Secretariat.

    The IETF invites any interested party to bring to its attention any
    copyrights, patents or patent applications, or other proprietary
    rights which may cover technology that may be required to practice

this standard. Please address the information to the IETF Executive
Director.