Contribution Number: oif2000.286

Working Group: Signaling Working Group

TITLE: RSVP Extensions for Optical UNI Signaling

SOURCE: John Yu, jzyu@zaffire.com
Fong Liaw, fliaw@zaffire.com
Zaffire, Inc.
.

Jonathan Lang, jplang@calient.net
Ayan Banerjee, abanerjee@calient.net
John Drake, jdrake@calient.net
Calient Networks

Greg Bernstein, GregB@ciena.com
Ciena

George Swallow, swallow@cisco.com
Yakov Rekhter, yakov@cisco.com
Cisco Systems, Inc

Kireeti Kompella, kireeti@juniper.net
Juniper Networks

Robert Rennison, robren@laurelnetworks.com
Laurel Networks, Inc.

Dimitrios Pendarakis, dpendarakis@tellium.com
Nooshin Komaee, nkomaee@tellium.com
Tellium, Inc.

Raj Jain, raj@nayna.com
Nayna Networks, Inc.

Zhensheng Zhang
zzhang@sorrentonet.com
Sorrento Networks

DATE: November 7, 2000

For additional information contact:
The Optical Internetworking Forum, 39355 California Street,
Suite 307, Fremont, CA 94538
510-608-5990 phone ✦ info@oiforum.com

Abstract

This draft defines a signaling mechanism based on RSVP-TE ([2]) to support an Optical User Network Interface (UNI). This effort is in part driven by work in the OIF as well as the recent draft on signaling requirements for the optical UNI ([3]), and is consistent with recent work on Generalized MPLS (see [4], [5], [6], and [7]) in IETF. The main function of this draft is to identify the relevant mechanisms in RSVP-TE (including further extensions) to satisfy functional requirements for an Optical UNI. This draft reflects ongoing work at the Optical Interworking Forum (OIF), however, not all of the concepts/requirements have been approved by the OIF.

Table of Contents

4 References

1. Introduction

There has recently been a significant effort amongst carriers, service providers, and vendors in the optical networking space to eliminate proprietary control protocols and develop a common control plane. The Optical Internetworking Forum (OIF) has initiated work on an Optical User-Network Interface (O-UNI) as a step in this direction. Recently, a draft [3] was submitted to the IETF defining proposed requirements and abstract messages for the Optical UNI.

This document describes how a signaling mechanism based on RSVP-TE [2] may be used for an Optical UNI. In particular, we identify the mechanisms already defined for RSVP-TE that can be used to satisfy the proposed requirements of [3]. This work is also in alignment with the recent Internet Drafts defining Generalized MPLS signaling (e.g., [4]).

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 [8].

2. Overview

The purpose of this document is to describe the use of RSVP [15], RSVP-TE [2], and GMPLS [4] to manage lightpaths at an optical User-Network Interface (O-UNI). The intent is to sufficiently describe information of the objects, packet formats, and procedures required to realize interoperable implementations, with the principle of leveraging the existing specifications as much as possible. A few new objects are defined to indicate lightpath attributes that are unique to O-UNI.

This specification supports only signaling messages and procedures to establish point-to-point, uni-directional and bi-directional, lightpaths. Specification for any other type of lightpath is for further study.

In this document and in [2, 4, 15], we define the directional terms “source” vs. “destination”, “originating” vs. “terminating”, "upstream" vs. "downstream", "previous hop" vs. "next hop", and "incoming interface" vs. "outgoing interface" with respect to the direction of light.

Procedures different from [2,4] are <u>underlined</u> to highlight the differences between this document and [2,4].

2.1 Optical UNI (O-UNI) signaling requirements and RSVP objects

As part of an optical UNI, the signaling protocol must have the capability to create, delete, and modify connections across a network, as well as query the status of an existing lightpath. Most of these capabilities may be directly supported by re-using existing procedures, messages, and objects defined in RSVP-TE [2] and in Generalized MPLS Signaling [4].

In particular, the optical UNI signaling requirements [3, 17] specify a set of attributes to be signaled during lightpath creation and modification. The following table summarizes the optical signaling attributes and the corresponding RSVP objects. Specific O-UNI related object formats and usage are described in Section 3.

OPTICAL signaling attribute	RSVP object
Bandwidth	Sender_Tspec [4]
Contract ID	Policy Data [16]
Destination address	Session [2]
Destination Port/Channel	Label ERO [4]
Directionality	Upstream Label [4]
Diversity	Session Attribute [2] and Generalized Label Request Object [4]
Framing Type	Generalized Label Request [4]
Light_Path ID	Session and SENDER_TEMPLATE [2]
Propagation Delay	Sender_Tspec [4]
Return code	Error Spec [15]
Reversion strategy	See note (1)
Service priority	Session Attribute [2]
Source address	Sender Template [2]
Source port/channel	Label Set/Suggest Label [4]
Transparency	Generalized Label Request [4]
User group ID	Policy Data [16]

Note:

(1) The support of Reversion strategy is not yet defined in [4]. OIF must first establish the exact semantic to allow clear signaling specification.

2.2 Basic RSVP protocol operation

There are two fundamental RSVP message types: Resv and Path [15]. An originating user node originates a lightpath establishment request by transmitting RSVP "Path" messages downstream along the direction of a lightpath. These Path messages create "path state" in each node along the way. In response to a Path message, the lightpath terminating user node sends RSVP reservation request (Resv) messages upstream towards the lightpath originator. These messages must follow exactly the reverse of the path(s) that the light will travel. They create and maintain "reservation state" in each node along the path(s). Resv messages will finally be delivered to the lightpath originating user node itself. This completes the establishment of a lightpath. Removal of a reservation state does not automatically removes its associated path state, but removal of a path state will remove the reservation states that are associated with it as well.

Path messages are sent with the same source and destination addresses as the lightpath. On the other hand, Resv messages are sent hop-by-hop; each RSVP-speaking node forwards a Resv message to the unicast address of a previous RSVP hop.

For each RSVP message type, there is a set of rules for the permissible choice of object types. These rules are specified using Backus-Naur Form (BNF) augmented with square brackets surrounding optional sub-sequences. The BNF implies an order for the objects in a message. However, in many (but not all) cases, object order makes no logical difference. An implementation should create messages with the objects in the order shown for each message, but accept the objects in any permissible order.

2.3 Use of RSVP-TE and Generalized MPLS signaling for O-UNI

2.3.1 O-UNI interfaces, control channels, and addressing

An O-UNI interface may identify one/multiple regular link(s), or one/multiple bundled link(s). Within a bundled link, there may be one or more component links that have the same characteristics. Generalized MPLS signaling [4] defines several types of labels that may be represented in a Generalized Label object. For optical applications a label may be arbitrarily associated with all or part of a component link, or may be a superset of multiple component links. A common understanding of the meaning of the component interface identifier [11] must exist between the user and the network across any O-UNI. This common understanding may be dynamically derived (e.g. using LMP [5]), or pre-configured.

In an O-UNI interface, each bundle link is associated with a control channel. Signaling protocol messages are exchanged over each control channel which are associated with a control channel interface. The control channel interface maybe numbered (with an IP address) or maybe unnumbered (in which case it is identified by a node's IP address and a unique identifier such as an ifindex value). The support for unnumbered O-UNI interface is optional. Furthermore, one or more control channels may transmit and receive their protocol messages via the same signaling transport control channel interface, which can be direct IP link (single IP hop) or over an IP network (multiple IP hop), as specified in [14].

2.3.2 Sending O-UNI RSVP messages

When sending an RSVP message over a directly connected signaling transport channel [14], the message format defined in 3.3. of [15] is used, i.e. sent as "raw" IP datagrams with protocol number 46.

RSVP Path, PathTear, and ResvConf messages are sent with an IP router alert option. The IP destination and source address of Path, PathTear, and ResvConf messages are the lightpath’s source and destination addresses.

RSVP Resv, ResvTear, ResvErr, PathErr, Ack, Srefresh messages are routed hop-by-hop, and their IP destination address is set to the previous RSVP hop. The source address is the previous hop that sends the message, not the originating hop.

The Notify message can be generated at any time to allow expedited notification of change in the status of a lightpath. The IP destination address is set to the IP address of the intended receiver. The Notify message is sent without the router alert option.

When the UNI RSVP messages are delivered over a non-directly connected signaling transport channel (out-of-fiber, out-of-band, co-located or non-co-located), the messages shall be encapsulated in IP-in-IP header before the transmission [14].

2.3.3 Receiving O-UNI RSVP messages

A node shall verify a received O-UNI RSVP message as specified in [2,4]. In addition, if a Path message is to be processed by a receiving node, the receiving RSVP node shall verify that the IP address in the RSVP_HOP object matches one of its adjacent node’s O-UNI interface identifiers and associates the message with the matched O-UNI interface. If a match does not exist, an error code “routing problem” SHALL be reported in a PathErr Message.

2.3.4 Reliable messaging

To support reliable messaging across the O-UNI, the Message_ID object and Ack desired flag of [10] MUST be used in every O-UNI RSVP message.

Message identification and acknowledgment is done on a per hop basis. All types of MESSAGE_ID objects contain a message identifier. The identifier MUST be unique on a per object generator's IP address basis. No more than one MESSAGE_ID object may be included in an RSVP message. Each message containing a MESSAGE_ID object may be acknowledged via a MESSAGE_ID_ACK object, when so indicated. MESSAGE_ID_ACK and MESSAGE_ID_NACK objects may be sent piggybacked in unrelated RSVP messages or in RSVP Ack messages. RSVP messages carrying any of the three object types may be included in an RSVP Bundled message. When included, each object is treated as if it were contained in a standard, non-bundled, RSVP message.

If a MESSAGE_ID_ACK object of a RSVP Message cannot be piggybacked in a pending RSVP message immediately, a downstream (upstream) node SHALL generate an Ack message including the MESSAGE_ID_ACK object to its upstream (downstream) node. This allows a faster confirmation for the message.

2.3.5 Reservation style

A reservation request includes a set of options that are collectively called the reservation "style" [15]. One reservation option concerns the treatment of reservations for different traffic sources within the same session: make a "distinct" reservation for each upstream traffic source, or make a single reservation that is "shared" among all selected upstream traffic sources.

The supported reservation styles at the O-UNI interface are Fixed Format (FF) style and Shared-Explicit (SE) style. A FF-style reservation reserves the resource for a distinctive upstream traffic source. An FF-style reservation request creates a distinct reservation for traffic from a particular upstream source, not sharing them with other upstream traffic sources for the same session. An SE-style reservation reserves the resource for one or more upstream traffic sources. An SE-style reservation creates a single reservation shared by selected upstream traffic sources. It allows a receiver to explicitly specify the set of upstream traffic sources to be included.

2.3.6 Lightpath identification

The SESSION Object plus SENDER_TEMPLATE Object [2] uniquely identifies a lightpath in RSVP at an Optical UNI.

2.3.7 Lightpath creation

To create a lightpath, the user O-UNI node creates an RSVP Path message with a session type of LSP_TUNNEL_IPv4 and inserts a GENERALIZED LABEL_REQUEST object into the Path message. The GENERALIZED LABEL_REQUEST object indicates that a label binding for this lightpath is requested and also provides an indication of the characteristics, such as desired link protection, encoding, of the light path.

To create a bi-directional lightpath, an UPSTREAM LABEL Object MUST be inserted in the Path Message. <u>Therefore, if the encoding type in a GENERALIZED LABEL REQUEST object indicates a bi-directional medium type, an UPSTREAM LABEL object MUST be inserted in the Path message; if no such an UPSTREAM LABEL object is inserted, an error code “incompatible” SHALL be reported in PathErr Message</u>.

Furthermore, during a lightpath creation, in order to provide the capability of lightpath modification later on after the ligthpath is established, it MUST insert a SESSION_ATTRIBUTE Object and indicates “SE Style desired” in the Flag field.

2.3.8 Lightpath modification

Lightpath modification can only be initiated by a lightpath’s originating user node on a lightpath that is established with a “SE Style desired” flag in SESSION ATTRIBUTE Object. It uses the bandwidth-increase “make before break” procedure as described in [2]. The following describes the procedure in more detail.

To effect a modification, the lightpath originating user node picks a new LSP ID and forms a new SENDER_TEMPLATE. Thereafter the node sends a new Path Message using the original SESSION object and the new SENDER_TEMPLATE with the new characteristics of the lightpath. It continues to use the old lightpath and refresh the old Path message. The shared SESSION object and SE style allow the LSP to be established sharing resources with the old LSP. Once the user node receives a Resv message for the new LSP, it MUST transition traffic to it and tear down the old LSP by sending a PathTear message toward the network node.

2.3.9 Lightpath deletion

A lightpath originating user node shall send a PathTear message toward the network to delete a lightpath.

A terminating user node SHALL send a ResvTear message toward its upstream nodes to delete a lightpath. Once a user node receives a ResvTear message for a lightpath, it MUST send a PathTear message toward the network to completely delete the lightpath.

A network node MAY send a PathErr message with error code set to “network initiated deletion, normal” to request the originating user node to delete the lightpath. If the network also removes its path state, it MUST set the Path_State_Removed flag in the PathErr message and also send a PathTear message toward downstream.

2.3.10 Lightpath status enquiry and response

The purpose of lightpath status enquiry and response identified so far has been to allow adjacent user and network nodes to re-synchronize their lightpath states when necessary, in particular after a link or node failure. Potentially, there are a large number of lightpath states to be re-synchronized. Therefore the procedure must allow the re-synchronization to occur efficiently. This specification uses the Srefresh message [10] for efficiency.

When a node decides that it is necessary to resynchronize its lightpath state with its adjacent node, it shall send Srefresh messages to refresh the Message_IDs of some or all active Resv and Path Messages. If a lightpath state has been deleted from the adjacent node before the re-synchronization, the adjacent node responds with a Message_Id_Nack Object for the deleted lightpath. Once a user node receives a Message_Id_Nack, it shall send a Resv or Path message toward its adjacent node. This would trigger the standard RSVP resource reservation and recovery procedure.

The above procedure may change the reservation states in a user or network node. The need for a non-state affecting procedure is for further study.

2.3.11 Node failure detection

RSVP HELLO procedure [2] may be used when a node’s link failure detection mechanism cannot detect node failure. The RSVP Hello extension enables RSVP nodes to detect when a neighboring node is not reachable. The mechanism provides node to node failure detection. When such a failure is detected it is handled much the same as a link layer communication failure. This mechanism is intended to be used when notification of link layer failure is not available and unnumbered links are not used, or when the failure detection mechanisms provided by the link layer are not sufficient for timely node failure detection.

This procedure is optional and does not require adjacent nodes to generate HELLO messages at the same time.

3. RSVP Messages and Objects for O-UNI signaling

The following sections describe messages, procedures, and objects from [2, 4, 16, 15] that are O-UNI related. If this document does not explicit specify some aspects of messages, procedures, and objects related to the O-UNI interface, the procedure defined in [4], [2], [10], [16], and [15] shall apply in the order listed.

An optical UNI node must support the following RSVP messages:

> Path, PathTear, PathErr, Resv, ResvTear, ResvConf, ResvErr, Ack, Srefresh, Notify

An optical UNI node may support the following RSVP messages:

> Hello

An optical UNI node must support the following RSVP objects:

> Label ERO [4], Error Spec [2, 4, 15], Flow Spec [15], Filter Spec [2], Generalized Label [4], Generalized Label Request [4], Component_interface_ID [11], Message_Id [10], Message_Id_Ack [10], Message_Id_Nack [10], Message_ID_List [10], Session [2], RSVP HOP [15], Sender Template [2], Session Attribute [2], Sender_Tsepc [15], Time Value [15], Upstream Label [4]

An optical UNI node may support the following RSVP objects:

> Integrity [15], Label Set [4], Policy Data [15], Record route [2], Suggested Label [4], Notify Request [4]

3.1 O-UNI RSVP Messages

3.1.1 ACK Message

The ACK message is for reliable messaging across an O-UNI. It has the following format:

```
<ACK message> ::= <Common Header> [ <INTEGRITY> ]
            <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>
            [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
```

3.1.2 Hello Message

Hello message is for node failure detection. It has the following format:

```
<Hello message> ::= <Common Header> [ <INTEGRITY> ]
            [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
            <HELLO>
```

3.1.3 Notify Message

The Notify message provides a mechanism to inform non-adjacent nodes of LSP related events. Notify messages are only generated after a Notify Request object has been received. The Notify message differs from the currently defined error messages (i.e., PathErr and ResvErr messages of RSVP) in that it can be "targeted" to a node other than the immediate upstream or downstream neighbor and that it is a generalized notification mechanism.

The Notify message can be generated at any time to allow expedited notification of change in the status of a lightpath. Consequently, both the user and network sides of the Optical UNI MUST be prepared to receive a Notify message. The Notify message is a generalized notification message. The IP destination address is set to the IP address of the intended receiver. The Notify message is sent without the router alert option. The format of the Notify message is as follows ([4]):

```
<Notify message> ::= <Common Header> [<INTEGRITY>] <MESSAGE_ID>
                  <ERROR_SPEC> <notify session list>

<notify session list> ::==
                  [<notify session list>] <notify session>

<notify session> ::== <SESSION> [<POLICY_DATA>...]
                  <sender descriptor>
```

The ERROR_SPEC object specifies the error and includes the IP address of either the node that detected the error or the link that has failed.

3.1.4 Path Message

The format for an O-UNI Path message is shown as follows:

```
<Path Message> ::=
      <Common Header>
      [ <INTEGRITY> ]
      [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
```

```
          <MESSAGE_ID>
          <SESSION> <RSVP_HOP>
          <TIME_VALUES>
          [ <EXPLICIT ROUTE> ]
          <GENERALIZED LABEL_REQUEST>
          [ <LABEL SET> ]
          [ <SESSION_ATTRIBUTE> ]
          [ <NOTIFY_REQUEST> ]
          [ <POLICY_DATA> ... ]
          <sender descriptor>
```

The format of the sender descriptor for unidirectional lightpath is:

```
     <sender descriptor> ::=
          <SENDER_TEMPLATE> <SENDER TSPEC>
          [ <RECORD_ROUTE> ]
          [ <DOWNSTREAM_COMPONENT_INTERFACE_ID> ]
          [ <SUGGESTED LABEL> ]
```

The format of the sender descriptor for bi-directional lightpath is:

```
     <sender descriptor> ::=
          <SENDER_TEMPLATE> <SENDER TSPEC>
          [ <RECORD_ROUTE> ]
          [ <DOWNSTREAM_COMPONENT_INTERFACE_ID> ]
          [ <SUGGESTED LABEL> ]
          [ <UPSTREAM_COMPONENT_INTERFACE_ID> ]
          <UPSTREAM LABEL>
```

3.1.5 PathErr Message

The format of an O-UNI PathErr Message is shown as follows:

```
     <PathErr message> ::= <Common Header> [ <INTEGRITY> ]
          [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
          <MESSAGE_ID>
          <SESSION>
          <ERROR_SPEC>
          [ <sender description> ]
```

3.1.6 PathTear Message

The format of an O-UNI PathTear Message is shown as follows:

```
     <PathTear Message> ::= <Common Header> [ <INTEGRITY> ]
          [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
          <MESSAGE_ID>
          <SESSION> <RSVP HOP>
          [ <sender descriptor> ]

     <sender descriptor> ::= (see earlier definition)
```

3.1.7 Resv Message

The format for an O-UNI Resv Message is as shown as follows:

```
<Resv Message> ::= <Common Header> [ <INTEGRITY> ]
```

```
        [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
        <MESSAGE_ID>
        <SESSION>  <RSVP_HOP>
        <TIME_VALUES>
        [ <RESV_CONFIRM> ]
        [ <NOTIFY_REQUEST> ]
        [<POLICY_DATA> ... ]
        <STYLE> <flow descriptor list>

  <flow descriptor list> ::=
        <FF flow descriptor list> | <SE flow descriptor>


  <FF flow descriptor list> ::=
        <FLOWSPEC> <FF flow descriptor>
        | <FF flow descriptor list> <FF flow descriptor>

  <FF flow descriptor> ::=
        [ <FLOWSPEC> ] <FILTER_SPEC>
        <GENERALIZED LABEL>
        [ <RECORD_ROUTE> ]

  <SE flow descriptor> ::= <FLOWSPEC> <SE filter spec list>

  <SE filter spec list> ::=
        <SE filter spec> | <SE filter spec list> <SE filter spec>

  <SE filter spec> ::=
        <FILTER SPEC> <GENERALIZED LABEL>
        [ <RECORD ROUTE> ]
```

3.1.8 ResvConf Message

The format of an O-UNI ResvConf Message is shown as follows:

```
  <ResvConf message> ::= <Common Header> [ <INTEGRITY> ]
        [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
        <MESSAGE_ID>
        <SESSION> <ERROR_SPEC>
        <RESV_CONFIRM>
        <STYLE> <flow descriptor list>

  <flow descriptor list> ::= (see earlier definition)
```

3.1.9 ResvErr Message

The format of an O-UNI ResvErr Message is shown as follows:

```
  <ResvErr message> ::= <Common Header> [ <INTEGRITY> ]
        [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
        <MESSAGE_ID>
        <SESSION> <RSVP_HOP>
        <ERROR_SPEC>
        [ <POLICY_DATA> ... ]
        <STYLE> <flow error description>
```

3.1.10 ResvTear Message

The format of an O-UNI ResvTear Message is shown as follows:

```
<ResvTear Message> ::=

      <Common Header> [ <INTEGRITY> ]
      [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
      <MESSAGE_ID>
      <SESSION> <RSVP HOP>
      [ <SCOPE> ] <STYLE>
      <flow description list>

<flow description list> ::= (see earlier list)
```

3.1.11 Srefresh Message

The format of an O-UNI Srefresh Message is shown below:

```
<Srefresh message> ::= <Common Header>  [ <INTEGRITY> ]
      [ [ <MESSAGE_ID_ACK> | <MESSAGE_ID_NACK> ] ... ]
      [ <MESSAGE_ID> ]
      <srefresh list>
<srefresh list> ::= <MESSAGE_ID_LIST>
      [ <srefrsh list> ]
```

3.2 O-UNI RSVP objects format

This section serves as a place holder for O-UNI RSVP objects.

3.2.1 Sender Template Object

LSP_TUNNEL_IPv4 Sender Template Object [2] has the following format:

```
Class = SENDER_TEMPLATE, LSP_TUNNEL_IPv4 C-Type = 7

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 sender address                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  MUST be zero                 |            LSP ID             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv4 tunnel sender address is the IPv4 address for a sender node. LSP ID is a 16-bit identifier used in the SENDER_TEMPLATE and the FILTER_SPEC that can be changed to allow a sender to share resources with itself.

LSP_TUNNEL_IPv6 Sender Template Object [2] has the following format:

```
Class = SENDER_TEMPLATE, LSP_TUNNEL_IPv6 C_Type = 8

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 sender address                  |
+                                                               +
|                            (16 bytes)                         |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  MUST be zero                 |            LSP ID             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 tunnel sender address is the IPv6 address for a sender node. LSP ID a 16-bit identifier used in the SENDER_TEMPLATE and the FILTER_SPEC that can be changed to allow a sender to share resources with itself.

3.2.2 Session Object

LSP_TUNNEL_IPv4 Session Object [2] has the following format:

Class = SESSION, LSP_TUNNEL_IPv4 C_Type = 7

```
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 end point address is the IPv4 address of the egress node for the tunnel. Tunnel ID a 16-bit identifier used in the SESSION that remains constant over the life of the tunnel. Extended Tunnel ID is a 32-bit identifier used in the SESSION that remains constant over the life of the tunnel. Extended Tunnel ID MUST be set to the source IP address or a pre-assigned lightpath identifier of a lightpath. This will uniquely identify a lightpath.

LSP_TUNNEL_IPv6 Session Object [2] has the following format:

Class = SESSION, LSP_TUNNEL_IPv6 C_Type = 8

```
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               |
+                                                               +
|                            (16 bytes)                         |
+                                                               +
```

```
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |  MUST be zero                 |      Tunnel ID                |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      +                                                               +
      |                       Extended Tunnel ID                      |
      +                                                               +
      |                            (16 bytes)                         |
      +                                                               +
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 tunnel end point address is the IPv6 address of the egress node for the tunnel. Tunnel ID is a 16-bit identifier used in the SESSION that remains constant over the life of the tunnel. Extended Tunnel ID a 16-byte identifier used in the SESSION that remains constant over the life of the tunnel. Extended Tunnel ID MUST be set to the source IP address of a lightpath. This will uniquely identify a lightpath in source and destination user node.

3.2.3 Session Attribute Object

The Session Attribute class is 207 [2]. Two C_Types are defined, LSP_TUNNEL, C-Type = 7 and LSP_TUNNEL_RA, C-Type = 1. The LSP_TUNNEL_RA C-Type includes all the same fields as the LSP_TUNNEL C-Type. Additionally it carries resource affinity information. The formats are as follows:

3.2.3.1 Format without resource affinities

```
      SESSION_ATTRIBUTE class = 207, LSP_TUNNEL C-Type = 7

      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
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |   Setup Prio  | Holding Prio  |     Flags     |  Name Length  |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                               |
      //          Session Name      (NULL padded display string)      //
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Setup Priority is the priority of the session with respect to taking resources, in the range of 0 to 7. The value 0 is the highest priority. The Setup Priority is used in deciding whether this session can preempt another session.

- Holding Priority the priority of the session with respect to holding resources, in the range of 0 to 7. The value 0 is the highest priority. Holding Priority is used in deciding whether this session can be preempted by another session.

- Flags indicate Local protection desired, or Label recording desired, or SE Style desired.

- Name Length is the length of the display string before padding, in bytes.

- Session Name is null padded string of characters.

3.2.3.2 Format with resource affinities

SESSION_ATTRIBUTE class = 207, LSP_TUNNEL_RA C-Type = 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Exclude-any                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Include-any                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Include-all                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Setup Prio  | Holding Prio  |     Flags     |  Name Length  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//          Session Name      (NULL padded display string)      //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Exclude-any represents a set of attribute filters associated with a tunnel any of which renders a link unacceptable.

- Include-any represents a set of attribute filters associated with a tunnel any of which renders a link acceptable (with respect to this test).

- Include-all represents a set of attribute filters associated with a tunnel all of which must be present for a link to be acceptable (with respect to this test).

Both the LSP_TUNEL or LSP_TUNNEL_RA subobjects shall be supported over the O-UNI. The usage of SESSION_ATTRIBUTE object in an O-UNI interface is as follows:

- The setup priority field indicates a lightpath’s service priority [3] during signaling.

- The holding priority field serves the purpose to indicate restoration priority [3].

- The Exclude-any, Include-any, and Include-all fields are used to indicate diverse lightpath and user-specified routing requirements [3] from user nodes.

3.2.4 GENERALIZED LABEL Object

The Generalized Label [4] extends the traditional Label Object in that it allows the representation of not only labels which travel in-band with associated data packets, but also labels which identify time-slots, wavelengths, or space division multiplexed positions.

The format of a Generalized Label is [4]:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class Num (16)|   C_Type (2)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Label                             |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Refer to [4] for detailed format for SONET/SDH label and port/wavelength label.

3.2.5 GENERALIZED LABEL REQUEST Object

GENERALIZED LABEL REQUEST object [4] is used to indicate a lightpath's bandwidth, diversion (protection) type, and framing type. Regarding support for diversion indication, this object specifies only the protection type. If explicit link inclusion or exclusion is desirable, SESSION ATRRIBUTE object shall be used.

The format of a Generalized Label Request with SONET/SDH Label range is:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class Num (19)|C_Type (5)[TBA]|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| LSP Enc. Type |Link Prot.Flags|             G-PID             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  RGT  |   RT  |    Reserved   |              RNC              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- LSP Encoding Type indicates the encoding of the LSP being requested.

- Link Protection Flags indicate the desired protection level(s) for each link along the LSP. Note that the flags are distinct from MPLS-level LSP protection. More than one bit may be set to indicate when multiple protection types are acceptable. When multiple bits are set and multiple protection types are available, the choice of protection type is a local (policy) decision.

- Generalized PID (G-PID) is an identifier of the payload carried by an LSP, i.e. an identifier of the client layer of that LSP. This must be interpreted according to the technology encoding type of the LSP and is used by the nodes at the endpoints of the LSP.

- Requested Grouping Type (RGT) indicates the SDH/SONET type of grouping requested for the LSP, it is used to constraint the type of concatenation.

- Requested Transparency (RT) indicates the type of SDH/SONET transparency ("emulation") requested for that LSP.

- Requested Number of Components (RNC) indicates the number of identical SDH/SONET signal types that are requested to be concatenated or inverse multiplexed in that LSP, as specified in the previous field. In these cases, the bandwidth of each component of that concatenation/bundling is obtained by dividing the aggregate bandwidth by the number of components requested. It is assumed that all these components have identical characteristics. This field is set to zero if non-concatenation or bundling is requested.

3.2.6 LABEL SET Object

The Label Set is used to limit the label choices of a downstream node to a set of acceptable labels. This limitation applies on a per hop basis. LABEL SET object is used to specify the port and channel to be used by a lightpath at the source. The support of this object is optional.

The format of a Label_Set is shown below and described in [4]:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             | Class Num (xx)|C_Type (xx)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Reserved                   |     Type      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Subchannel                           |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.7 SUGGESTED LABEL Object

The Suggested Label [4] is used to provide a downstream node with the upstream node's label preference. This permits the upstream node to start configuring it's hardware with the proposed label before the label is communicated by the downstream node. SUGGESTED LABEL object is used by a user (network) node to suggest a label to a network (user) node. The specified label is a suggestion, and downstream node may select a different label in its Resv message.

The format of a suggested label is identical to a generalized label. It is used in Path/REQUEST messages. In RSVP the Suggested Label uses a new class number (TBD of form 10bbbbbb) and the C-type of the label being suggested.

3.2.8 UPSTREAM LABEL Class

For bidirectional LSPs, two labels must be allocated. Bi-directional LSP setup is indicated by the presence of an Upstream Label in the REQUEST/Path message [4]. Lightpath is typical uni-directional. However, the directionality of some framing type is implied [17], e.g. SONET/SDH and G.E. is bi-directional. Therefore, <u>when a Framing Type implies bi-directional lightpath, this object SHALL be included in the Path Message</u>.

An Upstream Label has the same format as the generalized label. It uses a new class number (TBD of form 0bbbbbbb) and the C-type of the label being suggested.

3.2.9 ERROR_SPEC Object

ERROR_SPEC object has the following format [15]:

- o IPv4 ERROR_SPEC object: Class = 6, C-Type = 1

```
+-------------+-------------+-------------+-------------+
|           IPv4 Error Node Address (4 bytes)           |
+-------------+-------------+-------------+-------------+
|    Flags    |  Error Code |        Error Value        |
+-------------+-------------+-------------+-------------+
```

- o IPv6 ERROR_SPEC object: Class = 6, C-Type = 2

```
+-------------+-------------+-------------+-------------+
|                                                       |
+                                                       +
|                                                       |
+           IPv6 Error Node Address (16 bytes)          +
|                                                       |
+                                                       +
|                                                       |
+-------------+-------------+-------------+-------------+
|    Flags    |  Error Code |        Error Value        |
+-------------+-------------+-------------+-------------+
```

- Error Node Address is the IP address of the node in which the error was detected. This field may be set to zero if the address of the failed node is not known or its disclosure is not desirable.

- Flags is used only for an ERROR_SPEC object in a ResvErr message. If it on, this flag indicates that there was, and still is, a reservation in place at the failure point, or indicates that the FLOWSPEC that failed was strictly greater than the FLOWSPEC requested by this receiver.

- Error Code is for error description.

- Error Value contains additional information about the error.

3.2.10 Filter Specification Object

FILTER_SPEC [2] is used to identify the LSP, per RSVP-TE, required to identify the LSP in Resv.

LSP_TUNNEL_IPv4 Filter Specification Object:

Class = FILTER SPECIFICATION, LSP_TUNNEL_IPv4 C-Type = 7,

and the format of the object is identical to the LSP_TUNNEL_IPv4 SENDER_TEMPLATE object.

LSP_TUNNEL_IPv6 Filter Specification Object:

Class = FILTER SPECIFICATION, LSP_TUNNEL_IPv6 C_Type = 8.

and the format of the object is identical to the LSP_TUNNEL_IPv6 SENDER_TEMPLATE object.

3.2.11 FLOWSPEC Object

An OIF FLOWSPEC object may be defined and used to carry the peak bandwidth and the maximum propagation delay information and shall be used only if a limit on maximum delay propagation is applicable.

OIF_FLOWSPE object: Class = 9, C-Type = 0bbbbbbb.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Bandwidth                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Delay                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Bandwidth is carried in 32 bit number in IEEE floating point format (the unit is bytes per second).

- Delay indicates the maximum delay the lightpath may experience. It is expressed in 32 bit unsigned integer in unit of one microsecond.

3.2.12 SENDER_TSPEC object

An OIF SENDER_TSPEC object may be defined and used to carry the peak bandwidth and the maximum propagation delay information and shall be used only if a limit on maximum delay propagation is applicable.

OIF_SENDER_TSPEC object: Class = 12, C-Type = 0bbbbbbb.

OIF_SENDER_TSPEC Object has the same format as OIF_FLOWSPEC Object.

3.2.13 INTEGRITY Object

The INTEGRITY Object [18] of an RSVP control message contains cryptographic data to authenticate the originating node and to verify the contents of an RSVP message.

INTEGRITY Object: Class = 4, C-Type = 1

```
+-------------+-------------+-------------+-------------+
|    Flags    | 0 (Reserved)|                           |
+-------------+-------------+                           +
|                     Key Identifier                    |
+-------------+-------------+-------------+-------------+
|                     Sequence Number                   |
|                                                       |
+-------------+-------------+-------------+-------------+
|                                                       |
+                                                       +
|                                                       |
+                  Keyed Message Digest                 |
|                                                       |
+                                                       +
|                                                       |
+-------------+-------------+-------------+-------------+
```

Refer to [18] for the detailed usage and code point.

3.2.14. COMPONENT_INTERFACE_ID Object Class

The COMPONENT_INTERFACE_ID object class’s length field is set to 8. The Class Num is TBD of form 0bbbbbbb. The DOWNSTREAM_COMPONENT_INTERFACE_ID object, which has a C_Type of 1, is used to indicate the component interface to be used for traffic flowing in the downstream direction. The

UPSTREAM_COMPONENT_INTERFACE_ID object, which has a C_Type of 2, is used to indicate the component interface to be used for traffic flowing in the upstream direction. Both objects have the same format and carry a 16-bit Component Interface Identifier. The format of the objects are:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             |Class Num (TBD)|  C_Type (1|2) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Reserved               | Component Interface Identifier|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.2.15 MESSAGE_ID Object

MESSAGE_ID object [10] has the following format:

Class = MESSAGE_ID Class, C_Type = 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |                    Epoch                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Message_Identifier                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Flags indicate that the sender requests the receiver to send an acknowledgment for the message.

- Epoch indicates when the Message_Identifier sequence has reset. SHOULD be randomly generated each time a node reboots or the RSVP agent is restarted. The value SHOULD NOT be the same as was used when the node was last operational. This value MUST NOT be changed during normal operation.

- Message_Identifier: When combined with the message generator's IP address, the Message_Identifier field uniquely identifies a message. The values placed in this field change incrementally and only decrease when the Epoch changes or when the value wraps.

3.2.16 MESSAGE_ID_ACK Object

MESSAGE_ID_ACK object [10] has the following format:

Class = MESSAGE_ID_ACK Class, C_Type = 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
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |                     Epoch                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Message_Identifier                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Flags: No flags are currently defined. This field MUST be zero on transmission and ignored on receipt.
- Epoch: The Epoch field copied from the message being acknowledged.
- Message_Identifier: The Message_Identifier field copied from the message being acknowledged.

3.2.17 MESSAGE_ID_NAC object

MESSAGE_ID_NACK object [10]:

Class = MESSAGE_ID_ACK Class, C_Type = 2

with the same format as the MESSAGE_ID_ACK object.

3.2.18 MESSAGE_ID_LIST Object

MESSAGE_ID_LIST object [10] has the following format:

Class = MESSAGE_ID_LIST Class, C_Type = 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Flags     |                     Epoch                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Message_Identifier                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                :                              |
//                               :                             //
|                                :                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Message_Identifier                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- Flags: No flags are currently defined. This field MUST be zero on transmission and ignored on receipt.
- Epoch: The Epoch field from the MESSAGE_ID object corresponding to the trigger message that advertised the state being refreshed.
- Message_Identifier: The Message_Identifier field from the MESSAGE_ID object corresponding to the trigger message that

advertised the state being refreshed. One or more Message_Identifiers may be included.

3.2.19 POLICY_DATA Class

POLICY_DATA objects [16] contain policy information and are carried by RSVP messages. Policy_Data class can be used to convey Contract ID and User group ID as required by [3]. This specification does not mandate the support of specific Policy Data sub-objects, they are left to the network providers and vendors to decide.

3.2.20 EXPLICIT ROUTE Object & explicit egress label control

The EXPLICIT ROUTE object may be included to provide an EXPLICIT LABEL subobject [4] to indicate the desirable destination port and channel. An O-UNI node may ignore all subobjects, except the Label ERO sub-object, in an Explicit Route Object. The EXPLICIT_ROUTE object has the following format [2]:

```
    Class = 20, C_Type = 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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 //                        (Subobjects)                         //
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Label ERO subobject has the follow format [4]:

```
 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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |L|    Type     |     Length    |U|   Reserved  |    C-Type     
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                             Label                             |
 |                              ...                              |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Refer to [4] for the detailed usage and code point.

3.2.21 RECORD ROUTE Object (RRO)

Routes can be recorded via the RECORD_ROUTE object (RRO) [2]. Optionally, labels may also be recorded. The RECORD_ROUTE object has the following format:

Class = 21, C_Type = 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                        (Subobjects)                         //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The contents of a RRO are a series of variable-length data items called subobjects. It can be present in both RSVP Path and Resv messages. If Path message contains multiple RROs, only the first RRO is meaningful. Subsequent RROs SHOULD be ignored and SHOULD NOT be propagated. Similarly, if in a Resv message multiple RROs are encountered following a FILTER_SPEC before another FILTER_SPEC is encountered, only the first RRO is meaningful. Subsequent RROs SHOULD be ignored and SHOULD NOT be propagated.

Network may ignore the RRO by removing it from the Path message. The reason to list it here (as an option) is that a UNI interface may be owned by network providers, named as private-UNI, in which RRO may be meaningful.

3.2.22 RSVP HOP Object

RSVP_HOP Carries the IP address of the RSVP-capable node that sent this message and a logical outgoing interface handle. It refers to a RSVP_HOP object as a PHOP ("previous hop") object for downstream messages or as a NHOP ("next hop") object for upstream messages [15]. The RSVP_HOP object of each Path message contains the previous hop address, i.e., the IP address of the interface through which the Path message was most recently sent.

When a node forwards a Path message out a logical outgoing interface, it includes in the message some encoding of the identity of logical outgoing interface, called the "logical interface handle", or LIH. For an O-UNI interface, the LIH MUST carry a control channel’s identity over which the lightpath is to be established. The LIH value is carried in the RSVP_HOP object. The RSVP_HOP object has the following format [15]:

o   IPv4 RSVP_HOP object: Class = 3, C-Type = 1

```
+-------------+-------------+-------------+---------------+
|              IPv6 Next/Previous Hop Address             |
+-------------+-------------+-------------+---------------+
|                 Logical Interface Handle                |
+-------------+-------------+-------------+---------------+
```

o   IPv6 RSVP_HOP object: Class = 3, C-Type = 2

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

```
            |                                                           |
            +              IPv6 Next/Previous Hop Address               +
            |                                                           |
            +                                                           +
            |                                                           |
            +-------------+-------------+-------------+----------------+
            |                 Logical Interface Handle                  |
            +-------------+-------------+-------------+----------------+
```

This object carries the IP address of the interface through which the last RSVP-knowledgeable hop forwarded this message. The LIH is used to distinguish logical outgoing interfaces. A node receiving an LIH in a Path message saves its value and returns it in the HOP objects of subsequent Resv messages sent to the node that originated the LIH. LIH shall be set to ifindex value. It is used to identify an O-UNI when a non-directly connected signaling transport channel is used between the user and network node.

3.2.23 TIME_VALUES Object

The TIME_VALUES object [15] in an RSVP control message specifies the time period timer used for refreshing the state in this message. It indicates the refresh rate of a Path or Resv message that carries this object.

TIME_VALUES Object: Class = 5, C-Type = 1

```
            +-------------+-------------+-------------+-------------+
            |                    Refresh Period R                   |
            +-------------+-------------+-------------+-------------+
```

Refresh Period is the refresh timeout period R used to generate this message; in milliseconds.

3.2.24 Notify Request Object

Notifications may be sent via the Notify message defined below. The Notify Request object is used to request the generation of notifications. Notifications, i.e., the sending of a Notify message, may be requested in both the upstream and downstream directions.

The Notify Request Object [4] may be carried in Path or Resv Messages, see Section 7. The NOTIFY_REQUEST class number is TBA (of form 11bbbbbb). The format of a Notify Request is:

```
       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
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            Length             | Class Num(TBD)|  C_Type (1)   |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                    IPv4 Notify Node Address                   |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv4 Notify Node Address: 32 bits, set to the IP address of the node that should be notified when generating an error message.

4. References

[1] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[2] Awduche, D. O., Berger, L., Gan, D.-H., Li, T., Swallow, G., Srinivasan, V., "Extensions to RSVP for LSP Tunnels", Work in Progress (Internet Draft), draft-ietf-mpls-rsvp-lsp-tunnel-07.txt, August 2000.

[3] Aboul-Magd, O., Aparicio, O., Barry, R., Bernstein, G., Jain, R., Jia, L., Dulepet, R., Lazer, M., Yates, J., Pendarakis, D., Rajagopalan, B., Rennison, R., Xu, Y., Xue, Y., Yu, J. and Zhang, Z., "Signaling Requirements at the Optical UNI", Work in Progress (Internet Draft), July 2000.

[4] Ashwood-Smith, P., Fan, Y., Banerjee, A., Drake, J., Lang, J., Berger, L., Bernstein, G., Kompella, K., Mannie, E., Rajagopalan, B., Saha, S., Tang, Z., Rekhter, Y. and Sharma, V., "Generalized MPLS - Signaling Functional Description", Work in Progress (Internet Draft), draft-ietf-mpls-generialized-signaling-00.txt, October 2000.

[5] Lang, J.P., Mitra, K., Drake, J., Kompella, K., Rekhter, Y., Saha, D., Berger, L., Basak, D., and Sandick, H., "Link Management Protocol (LMP)", Work in Progress (Internet Draft), draft-lang-mpls-lmp-01.txt, July 2000.

[6] Kompella, K., Rekhter, Y., Banerjee, A., Drake, J., Bernstein, G., Fedyk, D., Mannie, E., Saha, D., and Sharma, V., "OSPF Extensions in Support of MPL(ambda)S", Work in Progress (Internet Draft), draft-kompella-ospf-ompls-extensions-00.txt, July 2000.

[7] Kompella, K., Rekhter, Y., Banerjee, A., Drake, J., Bernstein, G., Fedyk, D., Mannie, E., Saha, D., and Sharma, V., "IS-IS Extensions in Support of MPL(ambda)S", Work in Progress (Internet Draft), draft-ietf-gmpls-extensions-00.txt, September 2000.

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

[9] Terzis, A., et al., "RSVP Diagnostic Messages", RFC 2745, January 2000.

[10] Berger, L. et al., "RSVP Refresh Overhead Reduction Extensions", work in progress, June 2000.

[11] Kompella, K. et al., "Link Bundling in MPLS Traffic Engineering", work in progress, July 2000.

[12] Kompella, K. et al., "Traffic Engineering with Unnumbered Links", work in progress, Expire March 2001.

[13] Larry McAdams, Jennifer Yates, "User to Network Interface (UNI) Service Definition and Lightpath Attributes", OIF2000.0611, February 2000.

[14] D. Pendarakis et al. “Signaling Transport Mechanisms for UNI 1.0”, oif2000.xxx, Nov. 2000

[15] R. Braden, Ed., “Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification”, RFC 2205, September 1997.

[16] S. Yadav, “Identity Representation for RSVP”, RFC 2752, January 2000.

[17] L. McAdams, “User to Network Interface Service Definition and Lightpath Attributes”, OIF2000.061, February 2000.

[18] F. Baker et al. “RSVP Cryptographic Authentication”, RFC 2747, January 2000.