Network Working Group
Request for Comments: 4974
Updates: 3473
Category: Standards Track

D. Papadimitriou
Alcatel
A. Farrel
Old Dog Consulting

August 2007

Generalized MPLS (GMPLS) RSVP-TE Signaling Extensions in Support of Calls

Status of This Memo

Copyright Notice

Abstract

In certain networking topologies, it may be advantageous to maintain associations between endpoints and key transit points to support an instance of a service. Such associations are known as Calls.

A Call does not provide the actual connectivity for transmitting user traffic, but only builds a relationship by which subsequent Connections may be made. In Generalized MPLS (GMPLS) such Connections are known as Label Switched Paths (LSPs).

This document specifies how GMPLS Resource Reservation Protocol - Traffic Engineering (RSVP-TE) signaling may be used and extended to support Calls. These mechanisms provide full and logical Call/Connection separation.

The mechanisms proposed in this document are applicable to any environment (including multi-area), and for any type of interface: packet, layer-2, time-division multiplexed, lambda, or fiber switching.

Table of Contents

1. Introduction

This document defines protocol procedures and extensions to support Calls within Generalized MPLS (GMPLS).

A Call is an association between endpoints and possibly between key transit points (such as network boundaries) in support of an instance of a service. The end-to-end association is termed a "Call", and the association between two transit points or between an endpoint and a transit point is termed a "Call Segment". An entity that processes a Call or Call Segment is called a "Call Manager".

A Call does not provide the actual connectivity for transmitting user traffic, but only builds a relationship by which subsequent Connections may be made. In GMPLS, such Connections are known as Label Switched Paths (LSPs). This document does not modify Connection setup procedures defined in [RFC3473], [RFC4208], and [STITCH]. Connections set up as part of a Call follow the rules defined in these documents.

A Call may be associated with zero, one, or more than one Connection, and a Connection may be associated with zero or one Call. Thus, full and logical Call/Connection separation is needed.

An example of the requirements for Calls can be found in the ITU-T's Automatically Switched Optical Network (ASON) architecture [G.8080] and specific requirements for support of Calls in this context can be found in [RFC4139]. Note, however, that while the mechanisms described in this document meet the requirements stated in [RFC4139], they have wider applicability.

The mechanisms defined in this document are equally applicable to any packet (PSC) interface, layer-2 interfaces (L2SC), TDM capable interfaces, LSC interfaces, or FSC interfaces. The mechanisms and protocol extensions are backward compatible, and can be used for Call management where only the Call Managers need to be aware of the protocol extensions.

1.1. Applicability to ASON

[RFC4139] details the requirements on GMPLS signaling to satisfy the ASON architecture described in [G.8080]. The mechanisms described in this document meet the requirements for Calls as described in Sections 4.2 and 4.3 of [RFC4139] and the additional Call-related requirements in Sections 4.4, 4.7, 5, and 6 of [RFC4139].

[ASON-APPL] describes the applicability of GMPLS protocols to the ASON architecture.

2. Conventions Used in This document

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

In addition, the reader is assumed to be familiar with the terminology used in [RFC3471], [RFC3473], [RFC3477], and [RFC3945].

3. Requirements

3.1. Basic Call Function

The Call concept is used to deliver the following capabilities:

- Verification and identification of the Call initiator (prior to LSP setup).

- Support of virtual concatenation with diverse path component LSPs.

- Association of multiple LSPs with a single Call (note aspects related to recovery are detailed in [RFC4426] and [GMPLS-E2E]).

- Facilitation of control plane operations by allowing an operational status change of the associated LSP.

Procedures and protocol extensions to support Call setup, and the association of Calls with Connections are described in Section 5 and onwards of this document.

3.2. Call/Connection Separation

Full and logical Call and Connection separation is required. That is:

- It MUST be possible to establish a Connection without dependence on a Call.
- It MUST be possible to establish a Call without any associated Connections.
- It MUST be possible to associate more than one Connection with a Call.
- Removal of the last Connection associated with a Call SHOULD NOT result in the automatic removal of the Call except as a matter of local policy at the ingress of the Call.
- Signaling of a Connection associated with a Call MUST NOT require the distribution or retention of Call-related information (state) within the network.

3.3. Call Segments

Call Segment capabilities MUST be supported.

Procedures and (GMPLS) RSVP-TE signaling protocol extensions to support Call Segments are described in Section 7.3.1 of this document.

4. Concepts and Terms

The concept of a Call and a Connection are also discussed in the ASON architecture [G.8080] and [RFC4139]. This section is not intended as a substitute for those documents, but is a brief summary of the key terms and concepts.

4.1. What Is a Call?

A Call is an agreement between endpoints possibly in cooperation with the nodes that provide access to the network. Call setup may include capability exchange, policy, authorization, and security.

A Call is used to facilitate and manage a set of Connections that provide end-to-end data services. While Connections require state to be maintained at nodes along the data path within the network, Calls do not involve the participation of transit nodes except to forward the Call management requests as transparent messages.

A Call may be established and maintained independently of the Connections that it supports.

4.2. A Hierarchy of Calls, Connections, Tunnels, and LSPs

Clearly, there is a hierarchical relationship between Calls and Connections. One or more Connections may be associated with a Call. A Connection may not be part of more than one Call. A Connection may, however, exist without a Call.

In GMPLS RSVP-TE [RFC3473], a Connection is identified with a GMPLS TE Tunnel. Commonly, a Tunnel is identified with a single LSP, but it should be noted that for protection, load balancing, and many other functions, a Tunnel may be supported by multiple parallel LSPs. The following identification reproduces this hierarchy.

- Call IDs are unique within the context of the pair of addresses that are the source and destination of the Call.

- Tunnel IDs are unique within the context of the Session (that is the destination of the Tunnel). Applications may also find it convenient to keep the Tunnel ID unique within the context of a Call.

- LSP IDs are unique within the context of a Tunnel.

Note that the Call_ID value of zero is reserved and MUST NOT be used during LSP-independent Call establishment.

Throughout the remainder of this document, the terms LSP and Tunnel are used interchangeably with the term Connection. The case of a Tunnel that is supported by more than one LSP is covered implicitly.

4.3. Exchanging Access Link Capabilities

In an overlay model, it is useful for the ingress node of an LSP to know the link capabilities of the link between the network and the remote overlay network. In the language of [RFC4208], the ingress node can make use of information about the link between the egress core node (CN) and the remote edge node (EN). We call this link the egress network link. This information may allow the ingress node to tailor its LSP request to fit those capabilities and to better utilize network resources with regard to those capabilities.

For example, this might be used in transparent optical networks to supply information on lambda availability on egress network links, or, where the egress CN is capable of signal regeneration, it might provide a mechanism for negotiating signal quality attributes (such

as bit error rate). Similarly, in multi-domain routing environments, it could be used to provide end-to-end selection of component links (i.e., spatial attribute negotiation) where TE links have been bundled based on technology specific attributes.

In some circumstances, the Traffic Engineering Database (TED) may contain sufficient information for decisions to be made about which egress network link to use. In other circumstances, the TED might not contain this information and Call setup may provide a suitable mechanism to exchange information for this purpose. The Call-responder may use the Call parameters to select a subset of the available egress network links between the egress CN and the remote EN, and may report these links and their capabilities on the Call response so that the Call-initiator may select a suitable link.

The sections that follow indicate the cases where the TED may be used, and those where Call parameter exchange may be appropriate.

4.3.1. Network-Initiated Calls

Network-initiated Calls arise when the ingress (and correspondingly the egress) lie within the network and there may be no need to distribute additional link capability information over and above the information distributed by the TE and GMPLS extensions to the IGP. Further, it is possible that future extensions to these IGPs will allow the distribution of more detailed information including optical impairments.

4.3.2. User-Initiated Calls

User-initiated Calls arise when the ingress (and correspondingly the egress) lie outside the network. Edge link information may not be visible within the core network, nor (and specifically) at other edge nodes. This may prevent an ingress from requesting suitable LSP characteristics to ensure successful LSP setup.

Various solutions to this problem exist, including the definition of static TE links (that is, not advertised by a routing protocol) between the CNs and ENs. Nevertheless, special procedures may be necessary to advertise to the edge nodes outside of the network information about egress network links without also advertising the information specific to the contents of the network.

In the future, when the requirements on the information that needs to be supported are better understood, TE extensions to EGPs may be defined to provide this function, and new rules for leaking TE information between routing instances may be used.

4.3.3. Utilizing Call Setup

When IGP and EGP solutions are not available at the User-to-Network Interface (UNI), there is still a requirement to have the knowledge of the remote edge link capabilities at the local edge nodes.

The Call setup procedure provides an opportunity to discover edge link capabilities of remote edge nodes before LSP setup is attempted.

- The Call-responder can return information on one or more egress network links. The Call-responder could return a full list of the available links with information about the link capabilities, or it could filter the list to return information about only those links that might be appropriate to support the Connections needed by the Call. To do this second option, the Call-responder must determine such appropriate links from information carried in the Call request including destination of the Call, and the level of service (bandwidth, protection, etc.) required.

- On receiving a Call response, the Call-initiator must determine paths for the Connections (LSPs) that it will set up. The way that it does this is out of scope for this document since it is an implementation-specific, algorithmic process. However, it can take as input the information about the available egress network links as supplied in the Call response.

The LINK_CAPABILITY object is defined to allow this information to be exchanged. The information that is included in this object is similar to that distributed by GMPLS-capable IGPs (see [RFC4202]).

5. Protocol Extensions for Calls and Connections

This section describes the protocol extensions needed in support of Call identification and management of Calls and Connections. Procedures for the use of these protocol extensions are described in Section 6.

5.1. Call Setup and Teardown

Calls are established independently of Connections through the use of the Notify message. The Notify message is a targeted message and does not need to follow the path of LSPs through the network.

Simultaneous Call and Connection establishment (sometimes referred to as piggybacking) is not supported.

5.2. Call Identification

As soon as the concept of a Call is introduced, it is necessary to support some means of identifying the Call. This becomes particularly important when Calls and Connections are separated and Connections must contain some reference to the Call.

A Call may be identified by a sequence of bytes that may have considerable (but not arbitrary) length. A Call ID of 40 bytes would not be unreasonable. It is not the place of this document to supply rules for encoding or parsing Call IDs, but it must provide a suitable means to communicate Call IDs within the protocol. The full Call identification is referred to as the long Call ID.

The Call_ID is only relevant at the sender and receiver nodes. Maintenance of this information in the signaling state is not mandated at any intermediate node. Thus, no change in [RFC3473] transit implementations is required and there are no backward compatibility issues. Forward compatibility is maintained by using the existing default values to indicate that no Call processing is required.

Further, the long Call ID is not required as part of the Connection (LSP) state even at the sender and receiver nodes so long as some form of correlation is available. This correlation is provided through the short Call ID.

5.2.1. Long Form Call Identification

The long Call ID is only required on the Notify message used to establish the Call. It is carried in the "Session Name" field of the SESSION_ATTRIBUTE object on the Notify message.

A unique value per Call is inserted in the "Session Name" field by the initiator of the Call. Subsequent core nodes MAY inspect this object and MUST forward this object transparently across network interfaces until reaching the egress node. Note that the structure of this field MAY be the object of further formatting depending on the naming convention(s). However, [RFC3209] defines the "Session Name" field as a Null padded display string, so any formatting conventions for the Call ID must be limited to this scope.

5.2.2. Short Form Call Identification

The Connections (LSPs) associated with a Call need to carry a reference to the Call - the short Call ID. A new field is added to the signaling protocol to identify an individual LSP with the Call to which it belongs.

The new field is a 16-bit identifier (unique within the context of the address pairing provided by the Tunnel_End_Point_Address and the Sender_Address of the SENDER_TEMPLATE object) that MUST be exchanged on the Notify message during Call initialization and is used on all subsequent LSP messages that are associated with the Call. This identifier is known as the short Call ID and is encoded as described in Section 5.2.3. The Call ID MUST NOT be used as part of the processing to determine the session to which an RSVP signaling message applies. This does not generate any backward compatibility issue since the reserved field of the SESSION object defined in [RFC3209] MUST NOT be examined on receipt.

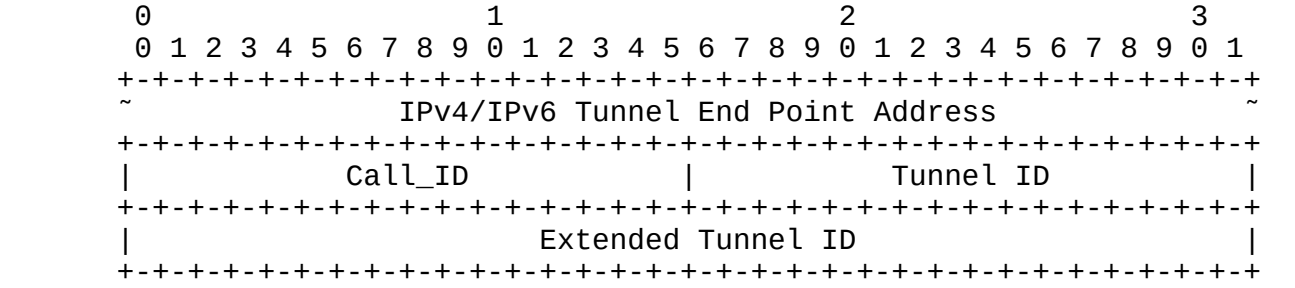
In the unlikely case of short Call_ID exhaustion, local node policy decides upon specific actions to be taken, but might include the use of second Sender_Address. Local policy details are outside of the scope of this document.

5.2.3. Short Form Call ID Encoding

The short Call ID is carried in a 16-bit field in the SESSION object carried on the Notify message used during Call setup, and on all messages during LSP setup and management. The field used was previously reserved (MUST be set to zero on transmission and ignored on receipt). This ensures backward compatibility with nodes that do not utilize Calls.

The figure below shows the new version of the object.

Class = SESSION, Class-Num = 1, C-Type = 7(IPv4)/8(IPv6)



IPv4/IPv6 Tunnel End Point Address: 32 bits/128 bits (see [RFC3209])

Call_ID: 16 bits

A 16-bit identifier used in the SESSION object that remains constant over the life of the Call. The Call_ID value MUST be set to zero when there is no corresponding Call.

Tunnel ID: 16 bits (see [RFC3209])

Extended Tunnel ID: 32 bits/128 bits (see [RFC3209])

5.3. LINK_CAPABILITY Object

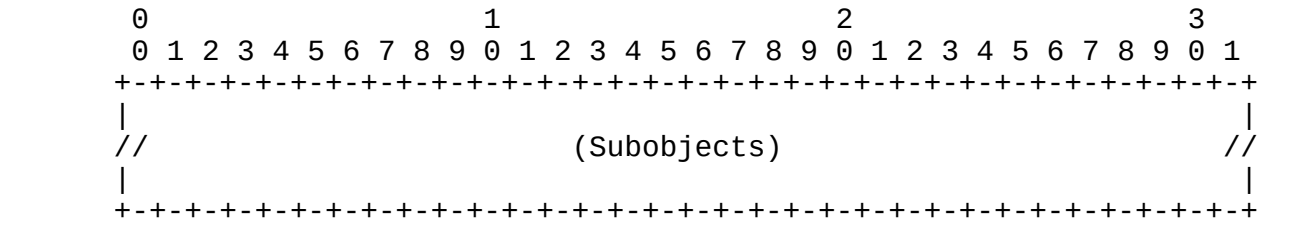
The LINK_CAPABILITY object is introduced to support link capability exchange during Call setup and MAY be included in a Notify message used for Call setup. This optional object includes the link-local capabilities of a link joining the Call-initiating node (or Call-terminating node) to the network. The specific node is indicated by the source address of the Notify message.

The link reported can be a single link or can be a bundled link [RFC4201].

The Class Number is selected so that the nodes that do not recognize this object drop it silently. That is, the top bit is set and the next bit is clear.

This object has the following format:

Class-Num = 133 (form 10bbbbbb), C_Type = 1



The contents of the LINK_CAPABILITY object is defined as a series of variable-length data items called subobjects. The subobject format is defined in [RFC3209].

The following subobjects are currently defined.

- Type 1: the link local IPv4 address of a link or a numbered bundle using the format defined in [RFC3209].

- Type 2: the link local IPv6 address of a link or a numbered bundle using the format defined in [RFC3209].

- Type 4: the link local identifier of an unnumbered link or bundle using the format defined in [RFC3477].

- Type 64: the Maximum Reservable Bandwidth corresponding to this link or bundle (see [RFC4201]).

- Type 65: the interface switching capability descriptor (see [RFC4202]) corresponding to this link or bundle (see also [RFC4201]).

Note: future revisions of this document may extend the above list.

A single instance of this object MAY be used to exchange capability information relating to more than one link or bundled link. In this case, the following ordering MUST be used:

- each link MUST be identified by an identifier subobject (Type 1, 2, or 4)

- capability subobjects (Type 64 or 65, and future subobjects) MUST be placed after the identifier subobject for the link or bundle to which they refer.

Multiple instances of the LINK_CAPABILITY object within the same Notify message are not supported by this specification. In the event that a Notify message contains multiple LINK_CAPABILITY objects, the receiver SHOULD process the first one as normal and SHOULD ignore subsequent instances of the object.

5.4. Revised Message Formats

The Notify message is enhanced to support Call establishment and teardown of Calls. See Section 6 for a description of the procedures.

5.4.1. Notify Message

The Notify message is modified in support of Call establishment by the optional addition of the LINK_CAPABILITY object. Further, the SESSION_ATTRIBUTE object is added to the <notify session> sequence to carry the long Call ID. The presence of the SESSION_ATTRIBUTE object MAY be used to distinguish a Notify message used for Call management, but see Section 5.5 for another mechanism. The <notify session list> MAY be used to simultaneously set up multiple Calls.

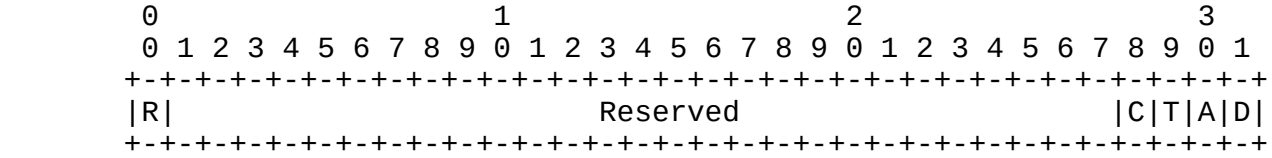
The format of the Notify Message is as follows:

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

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

<notify session>  ::= <SESSION> [ <ADMIN_STATUS> ]
                      [ <POLICY_DATA>...]
                      [ <LINK_CAPABILITY> ]
                      [ <SESSION_ATTRIBUTE> ]
                      [ <sender descriptor> | <flow descriptor> ]

<sender descriptor> ::= see [RFC3473]

<flow descriptor> ::= see [RFC3473]
```

5.5. ADMIN_STATUS Object

Notify messages exchanged for Call control and management purposes carry a specific new bit (the Call Management or C bit) in the ADMIN_STATUS object.

[RFC3473] indicates that the format and contents of the ADMIN_STATUS object are as defined in [RFC3471]. The new "C" bit is added for Call control as shown below.



Reflect (R): 1 bit - see [RFC3471]
Testing (T): 1 bit - see [RFC3471]
Administratively down (A): 1 bit - see [RFC3471]
Deletion in progress (D): 1 bit - see [RFC3471]
Call Management (C): 1 bit

This bit is set when the message is being used to control and manage a Call.

The procedures for the use of the C bit are described in Section 6.

6. Procedures in Support of Calls and Connections

6.1. Call/Connection Setup Procedures

This section describes the processing steps for Call and Connection setup.

There are three cases considered:

- A Call is set up without any associated Connection. It is assumed that Connections will be added to the Call at a later time, but this is neither a requirement nor a constraint.

- A Connection may be added to an existing Call. This may happen if the Call was set up without any associated Connections, or if another Connection is added to a Call that already has one or more associated Connections.

- A Connection may be established without any reference to a Call (see Section 6.4). This encompasses the previous LSP setup procedure.

Note that a Call MUST NOT be imposed upon a Connection that is already established. To do so would require changing the short Call ID in the SESSION object of the existing LSPs and this would constitute a change in the Session Identifier. This is not allowed by existing protocol specifications.

Call and Connection teardown procedures are described later in Section 6.6.

6.2. Call Setup

A Call is set up before, and independent of, LSP (i.e., Connection) setup.

Call setup MAY necessitate verification of the link status and link capability negotiation between the Call ingress node and the Call egress node. The procedure described below is applied only once for a Call and hence only once for the set of LSPs associated with a Call.

The Notify message (see [RFC3473]) is used to signal the Call setup request and response. The new Call Management (C) bit in the ADMIN_STATUS object is used to indicate that this Notify is managing a Call. The Notify message is sent with source and destination IPv4/IPv6 addresses set to any of the routable ingress/egress node addresses respectively.

At least one session MUST be listed in the <notify session list> of the Notify message. In order to allow for long identification of the Call, the SESSION_ATTRIBUTE object is added as part of the <notify session list>. Note that the ERROR_SPEC object is not relevant in Call setup and MUST carry the Error Code zero ("Confirmation") to indicate that there is no error.

During Call setup, the ADMIN_STATUS object is sent with the following bits set. Bits not listed MUST be set to zero.

R - to cause the egress to respond
C - to indicate that the Notify message is managing a Call.

The SESSION, SESSION_ATTRIBUTE, SENDER_TEMPLATE, SENDER_TSPEC objects included in the <notify session> of the Notify message are built as follows.

- The SESSION object includes as Tunnel_End_Point_Address any of the Call-terminating (egress) node's IPv4/IPv6 routable addresses. The Call_ID is set to a non-zero value unique within the context of the address pairing provided by the Tunnel_End_Point_Address and the Sender_Address from the SENDER_TEMPLATE object (see below). This value will be used as the short Call ID carried on all messages for LSPs associated with this Call.

  Note that the Call_ID value of zero is reserved and MUST NOT be used since it will be present in SESSION objects of LSPs that are not associated with Calls. The Tunnel_ID of the SESSION object is not relevant for this procedure and SHOULD be set to zero. The Extended_Tunnel_ID of the SESSION object is not relevant for this procedure and MAY be set to zero or to an address of the ingress node.

- The SESSION_ATTRIBUTE object contains priority flags. Currently no use of these flags is envisioned, however, future work may identify value in assigning priorities to Calls; accordingly the Priority fields MAY be set to non-zero values. None of the Flags in the SESSION_ATTRIBUTE object is relevant to this process and this field SHOULD be set to zero. The Session Name field is used to carry the long Call Id as described in Section 5.

- The SENDER_TEMPLATE object includes as Sender Address any of the Call-initiating (ingress) node's IPv4/IPv6 routable addresses. The LSP_ID is not relevant and SHOULD be set to zero.

- The bandwidth value inserted in the SENDER_TSPEC and FLOWSPEC objects MUST be ignored upon receipt and SHOULD be set to zero when sent.

Additionally, ingress/egress nodes that need to communicate their respective link local capabilities may include a LINK_CAPABILITY object in the Notify message.

The receiver of a Notify message may identify whether it is part of Call management or reporting an error by the presence or absence of the SESSION_ATTRIBUTE object in the <notify session list>. Full clarity, however, may be achieved by inspection of the new Call Management (C) bit in the ADMIN_STATUS object.

Note that the POLICY_DATA object may be included in the <notify session list> and MAY be used to identify requestor credentials, account numbers, limits, quotas, etc. This object is opaque to RSVP, which simply passes it to policy control when required.

Message IDs MUST be used during Call setup.

6.2.1. Accepting Call Setup

A node that receives a Notify message carrying the ADMIN_STATUS object with the R and C bits set is being requested to set up a Call. The receiver MAY perform authorization and policy according to local requirements.

If the Call is acceptable, the receiver responds with a Notify message reflecting the information from the Call request with two exceptions.

- The responder removes any LINK_CAPABLITY object that was received and MAY insert a LINK_CAPABILITY object that describes its own access link.

- The ADMIN_STATUS object is sent with only the C bit set. All other bits MUST be set to zero.

The responder MUST use the Message ID object to ensure reliable delivery of the response. If no Message ID Acknowledgement is received after the configured number of retries, the responder SHOULD continue to assume that the Call was successfully established. Call liveliness procedures are covered in Section 6.7.

6.2.2. Call Setup Failure and Rejection

Call setup may fail or be rejected.

If the Notify message can not be delivered, no Message ID acknowledgement will be received by the sender. In the event that the sender has retransmitted the Notify message a configurable number

of times without receiving a Message ID Acknowledgement (as described in [RFC2961]), the initiator SHOULD declare the Call failed and SHOULD send a Call teardown request (see Section 6.6).

It is also possible that a Message ID Acknowledgement is received but no Call response Notify message is received. In this case, the initiator MAY re-send the Call setup request a configurable number of times (see Section 6.7) before declaring that the Call has failed. At this point, the initiator MUST send a Call teardown request (see Section 6.6).

If the Notify message cannot be parsed or is in error, it MAY be responded to with a Notify message carrying the error code 13 ("Unknown object class") or 14 ("Unknown object C-Type") if appropriate to the error detected.

The Call setup MAY be rejected by the receiver because of security, authorization, or policy reasons. Suitable error codes already exist [RFC2205] and can be used in the ERROR_SPEC object included in the Notify message sent in response.

Error response Notify messages SHOULD also use the Message ID object to achieve reliable delivery. No action should be taken on the failure to receive a Message ID Acknowledgement after the configured number of retries.

6.3. Adding a Connections to a Call

Once a Call has been established, LSPs can be added to the Call. Since the short Call ID is part of the SESSION object, any LSP that has the same Call ID value in the SESSION object belongs to the same Call, and the Notify message used to establish the Call carried the same Call ID in its SESSION object.

There will be no confusion between LSPs that are associated with a Call and those which are not, since the Call ID value MUST be equal to zero for LSPs that are not associated with a Call, and MUST NOT be equal to zero for a valid Call ID.

LSPs for different Calls can be distinguished because the Call ID is unique within the context of the source address (in the SENDER_TEMPLATE object) and the destination address (in the SESSION object).

Ingress and egress nodes MAY group together LSPs associated with the same Call and process them as a group according to implementation requirements. Transit nodes need not be aware of the association of multiple LSPs with the same Call.

The ingress node MAY choose to set the "Session Name" of an LSP to match the long Call ID of the associated Call.

The C bit of the ADMIN_STATUS object MUST NOT be set on LSP messages including on Notify messages that pertain to the LSP and MUST be ignored.

6.3.1. Adding a Reverse Direction LSP to a Call

Note that once a Call has been established, it is symmetric. That is, either end of the Call may add LSPs to the Call.

Special care is needed when managing LSPs in the reverse direction since the addresses in the SESSION and SENDER_TEMPLATE are reversed. However, since the short Call ID is unique in the context of a given ingress-egress address pair, it may safely be used to associate the LSP with the Call.

Note that since Calls are defined here to be symmetrical, the issue of potential Call ID collision arises. This is discussed in Section 6.5.

6.4. Call-Free Connection Setup

It continues to be possible to set up LSPs as per [RFC3473] without associating them with a Call. If the short Call ID in the SESSION object is set to zero, there is no associated Call and the Session Name field in the SESSION_ATTRIBUTE object MUST be interpreted simply as the name of the session (see [RFC3209]).

The C bit of the ADMIN_STATUS object MUST NOT be set on messages for LSP control, including on Notify messages that pertain to LSPs, and MUST be ignored when received on such messages.

6.5. Call Collision

Since Calls are symmetrical, it is possible that both ends of a Call will attempt to establish Calls with the same long Call IDs at the same time. This is only an issue if the source and destination address pairs match. This situation can be avoided by applying some rules to the contents of the long Call ID, but such mechanisms are outside the scope of this document.

If a node that has sent a Call setup request and has not yet received a response itself receives a Call setup request with the same long Call ID and matching source/destination addresses, it SHOULD process as follows:

- If its source address is numerically greater than the remote source address, it MUST discard the received message and continue to wait for a response to its setup request.

- If its source address is numerically smaller than the remote source address, it MUST discard state associated with the Call setup that it initiated, and MUST respond to the received Call setup.

If a node receives a Call setup request carrying an address pair and long Call ID that match an existing Call, the node MUST return an error message (Notify message) with the new Error Code "Call Management" and the new Error Value "Duplicate Call" in response to the new Call request, and MUST NOT make any changes to the existing Call.

A further possibility for contention arises when short Call IDs are assigned by a pair of nodes for two distinct Calls that are set up simultaneously using different long Call IDs. In this event, a node receives a Call setup request carrying a short Call ID that matches one that it previously sent for the same address pair. The following processing MUST be followed:

- If the receiver's source address is numerically greater than the remote source address, the receiver returns an error (Notify message) with the new Error Code "Call Management" and the new Error Value "Call ID Contention".

- If the receiver's source address is numerically less than the remote source address, the receiver accepts and processes the Call request. It will receive an error message sent as described above, and at that point, it selects a new short Call ID and re-sends the Call setup request.

6.6. Call/Connection Teardown

As with Call/Connection setup, there are several cases to consider.

- Removal of a Connection from a Call
- Removal of the last Connection from a Call
- Teardown of an "empty" Call

The case of tearing down an LSP that is not associated with a Call does not need to be examined as it follows exactly the procedures described in [RFC3473].

6.6.1. Removal of a Connection from a Call

An LSP that is associated with a Call may be deleted using the standard procedures described in [RFC3473]. No special procedures are required.

Note that it is not possible to remove an LSP from a Call without deleting the LSP. It is not valid to change the short Call ID from non-zero to zero since this involves a change to the SESSION object, which is not allowed.

6.6.2. Removal of the Last Connection from a Call

When the last LSP associated with a Call is deleted, the question arises as to what happens to the Call. Since a Call may exist independently of Connections, it is not always acceptable to say that the removal of the last LSP from a Call removes the Call.

The removal of the last LSP does not remove the Call and the procedures described in the next Section MUST be used to delete the Call.

6.6.3. Teardown of an "Empty" Call

When all LSPs have been removed from a Call, the Call may be torn down or left for use by future LSPs.

Deletion of Calls is achieved by sending a Notify message just as for Call setup, but the ADMIN_STATUS object carries the R, D, and C bits on the teardown request and the D and C bits on the teardown response. Other bits MUST be set to zero.

When a Notify message is sent for deleting a Call and the initiator does not receive the corresponding reflected Notify message (or possibly even the Message ID Ack), the initiator MAY retry the deletion request using the same retry procedures as used during Call establishment. If no response is received after full retry, the node deleting the Call MAY declare the Call deleted, but under such circumstances the node SHOULD avoid re-using the long or short Call IDs for at least five times the Notify refresh period.

6.6.4. Attempted Teardown of a Call with Existing Connections

If a Notify request with the D bit of the ADMIN_STATUS object set is received for a Call for which LSPs still exist, the request MUST be rejected with the Error Code "Call Management" and Error Value "Connections Still Exist". The state of the Call MUST NOT be changed.

6.6.5. Teardown of a Call from the Egress

Since Calls are symmetric, they may be torn down from the ingress or egress.

When the Call is "empty" (has no associated LSPs), it may be deleted by the egress sending a Notify message just as described above.

Note that there is a possibility that both ends of a Call initiate Call deletion at the same time. In this case, the Notify message acting as teardown request MAY be interpreted by its recipient as a teardown response. But since the Notify messages acting as teardown requests carry the R bit in the ADMIN_STATUS object, they MUST be responded to anyway. If a teardown request Notify message is received for an unknown Call ID, it is, nevertheless, responded to in the affirmative.

6.7. Control Plane Survivability

Delivery of Notify messages is secured using Message ID Acknowledgements as described in previous sections.

Notify messages provide end-to-end communication that does not rely on constant paths through the network. Notify messages are routed according to IGP routing information. No consideration is, therefore, required for network resilience (for example, make-before-break, protection, fast re-route), although end-to-end resilience is of interest for node restart and completely disjoint networks.

Periodic Notify messages SHOULD be sent by the initiator and terminator of the Call to keep the Call alive and to handle ingress or egress node restart. The time period for these retransmissions is a local matter, but it is RECOMMENDED that this period should be twice the shortest refresh period of any LSP associated with the Call. When there are no LSPs associated with a Call, an LSR is RECOMMENDED to use a refresh period of no less than one minute. The Notify messages are identical to those sent as if establishing the Call for the first time, except for the LINK_CAPABILITY object, which may have changed since the Call was first established, due to, e.g., the establishment of Connections, link failures, or the addition of new component links. The current link information is useful for the establishment of subsequent Connections. A node that receives a refresh Notify message carrying the R bit in the ADMIN_STATUS object MUST respond with a Notify response. A node that receives a refresh Notify message (response or request) MAY reset its timer - thus, in normal processing, Notify refreshes involve a single exchange once per time period.

A node (sender or receiver) that is unsure of the status of a Call MAY immediately send a Notify message as if establishing the Call for the first time.

Failure to receive a refresh Notify request has no specific meaning. A node that fails to receive a refresh Notify request MAY send its own refresh Notify request to establish the status of the Call. If a node receives no response to a refresh Notify request (including no Message ID Acknowledgement), a node MAY assume that the remote node is unreachable or unavailable. It is a local policy matter whether this causes the local node to teardown associated LSPs and delete the Call.

In the event that an edge node restarts without preserved state, it MAY relearn LSP state from adjacent nodes and Call state from remote nodes. If a Path or Resv message is received with a non-zero Call ID but without the C bit in the ADMIN_STATUS, and for a Call ID that is not recognized, the receiver is RECOMMENDED to assume that the Call establishment is delayed and ignore the received message. If the Call setup never materializes, the failure by the restarting node to refresh state will cause the LSPs to be torn down. Optionally, the receiver of such an LSP message for an unknown Call ID may return an error (PathErr or ResvErr message) with the error code "Call Management" and Error Value "Unknown Call ID".

7. Applicability of Call and Connection Procedures

This section considers the applicability of the different Call establishment procedures at the NNI and UNI reference points. This section is informative and is not intended to prescribe or prevent other options.

7.1. Network-Initiated Calls

Since the link properties and other traffic-engineering attributes are likely known through the IGP, the LINK_CAPABILITY object is not usually required.

In multi-domain networks, it is possible that access link properties and other traffic-engineering attributes are not known since the domains do not share this sort of information. In this case, the Call setup mechanism may include the LINK_CAPABILITY object.

7.2. User-Initiated Calls

It is possible that the access link properties and other traffic-engineering attributes are not shared across the core network. In this case, the Call setup mechanism may include the LINK_CAPABILITY object.

Further, the first node within the network may be responsible for managing the Call. In this case, the Notify message that is used to set up the Call is addressed by the user network edge node to the first node of the core network. Moreover, neither the long Call ID nor the short Call ID is supplied (the Session Name Length is set to zero and the Call ID value is set to zero). The Notify message is passed to the first core node, which is responsible for generating the long and short Call IDs before dispatching the message to the remote Call end point (which is known from the SESSION object).

Further, when used in an overlay context, the first core node is allowed (see [RFC4208]) to replace the Session Name assigned by the ingress node and passed in the Path message. In the case of Call management, the first core node:

1) MAY insert a long Call ID in the Session Name of a Path message.

2) MUST replace the Session Name with that originally issued by the user edge node when it returns the Resv message to the ingress node.

7.3. External Call Managers

Third party Call management agents may be used to apply policy and authorization at a point that is neither the initiator nor terminator of the Call. The previous example is a particular case of this, but the process and procedures are identical.

7.3.1. Call Segments

Call Segments exist between a set of default and configured External Call Managers along a path between the ingress and egress nodes, and use the protocols described in this document.

The techniques that are used by a given service provider to identify which External Call Managers within its network should process a given Call are beyond the scope of this document.

An External Call Manager uses normal IP routing to route the Notify message to the next External Call Manager. Notify messages (requests

and responses) are therefore encapsulated in IP packets that identify the sending and receiving External Call Managers, but the addresses used to identify the Call (the Sender Address in the SENDER_TEMPLATE object and the Tunnel Endpoint Address in the SESSION object) continue to identify the endpoints of the Call.

8. Non-Support of Call ID

It is important that the procedures described above operate as seamlessly as possible with legacy nodes that do not support the extensions described.

Clearly, there is no need to consider the case where the Call initiator does not support Call setup initiation.

8.1. Non-Support by External Call Managers

It is unlikely that a Call initiator will be configured to send Call establishment Notify requests to an external Call manager, including the first core node, if that node does not support Call setup.

A node that receives an unexpected Call setup request will fall into one of the following categories.

- Node does not support RSVP. The message will fail to be delivered or responded to. No Message ID Acknowledgement will be sent. The initiator will retry and then give up.

- Node supports RSVP or RSVP-TE but not GMPLS. The message will be delivered but not understood. It will be discarded. No Message ID Acknowledgement will be sent. The initiator will retry and then give up.

- Node supports GMPLS but not Call management. The message will be delivered, but parsing will fail because of the presence of the SESSION_ATTRIBUTE object. A Message ID Acknowledgement may be sent before the parse fails. When the parse fails, the Notify message may be discarded in which case the initiator will retry and then give up; alternatively, a parse error may be generated and returned in a Notify message which will indicate to the initiator that Call management is not supported.

8.2. Non-Support by Transit Node

Transit nodes SHOULD NOT examine Notify messages that are not addressed to them. However, they will see short Call IDs in all messages for all LSPs associated with Calls.

Previous specifications state that these fields SHOULD be ignored on receipt and MUST be transmitted as zero. This might be interpreted by some implementations as meaning that the fields should be zeroed before the objects are forwarded. If this happens, LSP setup will not be possible. If either of the fields is zeroed either on the Path or the Resv message, the Resv message will reach the initiator with the field set to zero - this is an indication to the initiator that some node in the network is preventing Call management. Use of Explicit Routes may help to mitigate this issue by avoiding such nodes. Ultimately, however, it may be necessary to upgrade the offending nodes to handle these protocol extensions.

8.3. Non-Support by Egress Node

It is unlikely that an attempt will be made to set up a Call to a remote node that does not support Calls.

If the egress node does not support Call management through the Notify message, it will react (as described in Section 8.1) in the same way as an External Call Manager.

9. Security Considerations

Please refer to each of the documents referenced in the following sections for a description of the security considerations applicable to the features that they provide.

9.1. Call and Connection Security Considerations

Call setup is vulnerable to attacks both of spoofing and denial of service. Since Call setup uses Notify messages, the process can be protected by the use of the INTEGRITY object to secure those messages as described in [RFC2205] and [RFC3473]. Deployments where security is a concern SHOULD use this mechanism.

Implementations and deployments MAY additionally protect the Call setup exchange using end-to-end security mechanisms such as those provided by IPsec (see [RFC4302] and [RFC4303]), or using RSVP security [RFC2747].

Note, additionally, that it would be desirable to use the process of independent Call establishment, where the Call is set up separately from the LSPs, to apply an extra level of authentication and policy for the end-to-end LSPs above that which is available with Call-less, hop-by-hop LSP setup. However doing so will require additional work to set up security associations between the peer and the call manager that meet the requirements of [RFC4107]. The mechanism described in this document is expected to meet this use case when combined with

this additional work. Application of this mechanism to the authentication and policy use case prior to standardization of a security solution is inappropriate and outside the current applicability of the mechanism.

The frequency of Call establishment is application dependent and hard to generalize. Key exchange for Call-related message exchanges is therefore something that should be configured or arranged dynamically in different deployments according to the advice in [RFC4107]. Note that the remote RSVP-TE signaling relationship between Call endpoints is no different from the signaling relationship between LSRs that establish an LSP. That is, the LSRs are not necessarily IP-adjacent in the control plane in either case. Thus, key exchange should be regarded as a remote procedure, not a single hop procedure. There are several procedures for automatic remote exchange of keys, and IKEv2 [RFC4306] is particularly suggested in [RFC3473].

10. IANA Considerations

10.1. RSVP Objects

A new RSVP object is introduced. IANA has made an assignment from the "RSVP Parameters" registry using the sub-registry "Class Names, Class Numbers, and Class Types".

- LINK_CAPABILITY object

  Class-Num = 133 (form 10bbbbbb)

  The Class Number is selected so that nodes not recognizing this object drop it silently. That is, the top bit is set and the next bit is cleared.

  C-Type = 1 (TE Link Capabilities)

  The LINK_CAPABILITY object is only defined for inclusion on Notify messages.

  Refer to Section 5.3 of this document.

  IANA maintains a list of subobjects that may be carried in this object. This list is maintained in the registry entry for the LINK_CAPABILITY object as is common practice for the subobjects of other RSVP objects. For each subobject, IANA lists:

  - subobject type number
  - subobject name
  - reference indicating where subobject is defined.

The initial list of subobjects is provided in Section 5.3 of this document.

10.2. RSVP Error Codes and Error Values

A new RSVP Error Code and new Error Values are introduced. IANA has made assignments from the "RSVP Parameters" registry using the sub-registry "Error Codes and Globally-Defined Error Value Sub-Codes".

- Error Codes:
  - Call Management (value 32)

- Error Values:
  - Call Management/Call ID Contention (value 1)
  - Call Management/Connections Still Exist (value 2)
  - Call Management/Unknown Call ID (value 3)
  - Call Management/Duplicate Call (value 4)

10.3. RSVP ADMIN_STATUS Object Bits

[GMPLS-E2E] requested that IANA manage the bits of the RSVP ADMIN_STATUS object. A new "Administrative Status Information Flags" sub-registry of the "GMPLS Signaling Parameters" registry was created.

This document defines one new bit, the C bit, to be tracked in that sub-registry. Bit number 28 has been assigned. See Section 5.5 of this document.

11. Acknowledgements

The authors would like to thank George Swallow, Yakov Rekhter, Lou Berger, Jerry Ash, and Kireeti Kompella for their very useful input to, and comments on, an earlier revision of this document.

Thanks to Lyndon Ong and Ben Mack-Crane for lengthy discussions during and after working group last call, and to Deborah Brungard for a final, detailed review.

Thanks to Suresh Krishnan for the GenArt review, and to Magnus Nystrom for discussions about security.

Useful comments were received during IESG review from Brian Carpenter, Lars Eggert, Ted Hardie, Sam Hartman, and Russ Housley.

12. References

12.1. Normative References

[GMPLS-E2E] Lang, J., Ed., Rekhter, Y., Ed., and D. Papadimitriou, Ed., "RSVP-TE Extensions in Support of End-to-End Generalized Multi-Protocol Label Switching (GMPLS) Recovery", RFC 4872, May 2007.

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

[RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[RFC2747] Baker, F., Lindell, B., and M. Talwar, "RSVP Cryptographic Authentication", RFC 2747, January 2000.

[RFC2961] Berger, L., Gan, D., Swallow, G., Pan, P., Tommasi, F., and S. Molendini, "RSVP Refresh Overhead Reduction Extensions", RFC 2961, April 2001.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3471] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description", RFC 3471, January 2003.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

[RFC3477] Kompella, K. and Y. Rekhter, "Signalling Unnumbered Links in Resource ReSerVation Protocol - Traffic Engineering (RSVP-TE)", RFC 3477, January 2003.

[RFC3945] Mannie, E., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", RFC 3945, October 2004.

[RFC4201] Kompella, K., Rekhter, Y., and L. Berger, "Link Bundling in MPLS Traffic Engineering (TE)", RFC 4201, October 2005.

[RFC4202] Kompella, K., Ed., and Y. Rekhter, Ed., "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4202, October 2005.

[RFC4208] Swallow, G., Drake, J., Ishimatsu, H., and Y. Rekhter, "Generalized Multiprotocol Label Switching (GMPLS) User-Network Interface (UNI): Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Support for the Overlay Model", RFC 4208, October 2005.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC4306] Kaufman, C., Ed., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

[RFC4426] Lang, J., Ed., Rajagopalan, B., Ed., and D. Papadimitriou, Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Recovery Functional Specification", RFC 4426, March 2006.

12.2. Informative References

[ASON-APPL] Drake, J., Papadimitriou, D., Farrel, A., Brungard, D., Ali, Z., Ayyangar, A., Ould-Brahim, H., and D. Fedyk, "Generalized MPLS (GMPLS) RSVP-TE Signalling in support of Automatically Switched Optical Network (ASON), Work in Progress, July 2005.

[RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, June 2005.

[RFC4139] Papadimitriou, D., Drake, J., Ash, J., Farrel, A., and L. Ong, "Requirements for Generalized MPLS (GMPLS) Signaling Usage and Extensions for Automatically Switched Optical Network (ASON)", RFC 4139, July 2005.

[STITCH] Ayyangar, A., Kompella, K., Vasseur, JP., and A. Farrel, "Label Switched Path Stitching with Generalized Multiprotocol Label Switching Traffic Engineering (GMPLS TE)", Work in Progress, April 2007.

For information on the availability of the following document, please see http://www.itu.int.

[G.8080] ITU-T, "Architecture for the Automatically Switched Optical Network (ASON)," Recommendation G.8080/ Y.1304, November 2001 (and Revision, January 2003).

Authors' Addresses

John Drake
Boeing Satellite Systems
2300 East Imperial Highway
El Segundo, CA 90245
EMail: John.E.Drake2@boeing.com

Deborah Brungard (AT&T)
Rm. D1-3C22 - 200 S. Laurel Ave.
Middletown, NJ 07748, USA
EMail: dbrungard@att.com

Zafar Ali (Cisco)
100 South Main St. #200
Ann Arbor, MI 48104, USA
EMail: zali@cisco.com

Arthi Ayyangar (Nuova Systems)
2600 San Tomas Expressway
Santa Clara, CA 95051
EMail: arthi@nuovasystems.com

Don Fedyk (Nortel Networks)
600 Technology Park Drive
Billerica, MA, 01821, USA
EMail: dwfedyk@nortel.com

Contact Addresses

Dimitri Papadimitriou
Alcatel-Lucent,
Fr. Wellesplein 1,
B-2018 Antwerpen, Belgium
Phone: +32 3 240-8491
EMail: dimitri.papadimitriou@alcatel-lucent.be

Adrian Farrel
Old Dog Consulting
Phone: +44 (0) 1978 860944
EMail: adrian@olddog.co.uk

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.