               Extensions to GMPLS Resource Reservation Protocol (RSVP)
                                Graceful Restart

Status of This Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

Abstract

   This document describes extensions to the Resource Reservation
   Protocol (RSVP) Graceful Restart mechanisms defined in RFC 3473. The
   extensions enable the recovery of RSVP signaling state based on the
   Path message last sent by the node being restarted.

   Previously defined Graceful Restart mechanisms, also called recovery
   from nodal faults, permit recovery of signaling state from adjacent
   nodes when the data plane has retained the associated forwarding
   state across a restart. Those mechanisms do not fully support
   signaling state recovery on ingress nodes or recovery of all RSVP
   objects.

   The extensions defined in this document build on the RSVP Hello
   extensions defined in RFC 3209, and extensions for state recovery on
   nodal faults defined in RFC 3473. Using these extensions, the
   restarting node can recover all previously transmitted Path state,
   including the Explicit Route Object and the downstream (outgoing)
   interface identifiers. The extensions can also be used to recover
   signaling state after the restart of an ingress node.

   These extensions are not used to create or restore data plane state.

   The extensions optionally support the use of Summary Refresh, defined
   in RFC 2961, to reduce the number of messages exchanged during the
   Recovery Phase when the restarting node has recovered signaling state
   locally for one or more Label Switched Paths (LSPs).

Table of Contents

1. Introduction

   RSVP Graceful Restart is defined in [RFC3473] and uses mechanisms
   defined in [RFC3209]. When data/forwarding plane state can be
   retained across the restart of the RSVP agent that established such
   state, RSVP Graceful Restart provides the ability for the RSVP agent
   to resynchronize its state based on updates received from its
   neighboring RSVP agents, and, reconcile such state with the retained
   data/forwarding plane state. [RFC3209] describes a mechanism, using
   RSVP Hello messages, to detect the state of an adjacent RSVP agent.
   [RFC3473] extends this mechanism to advertise the capability of
   retaining data/forwarding plane state across the restart of a node or
   a "nodal fault". [RFC3473] also defines the Recovery Label object
   for use in the Path message of the RSVP neighbor upstream of a
   restarting node, to indicate that the Path message is for existing
   data plane state.

   This document presents extensions to address two aspects of graceful
   restart not previously supported. The presented extensions enable a
   restarting node to recover all objects in previously transmitted Path
   messages, including the Explicit Route Object (ERO), from its
   downstream neighbors, thus recovering the control plane state. The
   extensions do not facilitate the recovery or creation of
   data/forwarding plane state, and can only be used to reestablish
   control plane state that matches in-place data/forwarding state. The
   extensions also enable graceful restart of an ingress node that does
   not preserve full RSVP state across restarts. The presented
   extensions are equally applicable to LSPs of various switching types
   as defined in [RFC3471].

   Per [RFC3473], a restarting node can distinguish Path messages
   associated with LSPs being recovered by the presence of the Recovery
   Label object. To determine the downstream (outgoing) interface and
   associated label(s), the restarting node must consult the data plane.
   This may not be possible for all types of nodes. Furthermore, data
   plane information is not sufficient to reconstruct all previously
   transmitted Path state. In these cases, the only source of RSVP
   state is the downstream RSVP neighbor.

   For example, when the restarting node is an ingress node, all
   previously transmitted Path state may need to be recovered. Such
   Path state may include (but is not restricted to) the Protection
   object, the Admin Status object, the Session Attribute object, the
   Notify Request object, and the Sender Tspec object. A restarting
   transit node may have modified received Path state in its previously
   transmitted Path message, which cannot be reconstructed internally
   during recovery.

Another example of state that cannot be completely recovered from the data plane in some cases is the previously transmitted ERO. Recovery of the previously transmitted ERO minimizes subsequent change of downstream LSP state. On a restarting ingress node, the ERO may have been based on configuration or the result of a previous path computation. A restarting transit node may have previously performed some form of path computation as a result of not receiving an ERO or receiving a loose hop in the ERO. In addition to the ERO, the restarting node may have modified other received Path state in its previously transmitted Path state, which cannot be reconstructed internally during recovery.

The defined extensions provide a restarting upstream node with all information previously transmitted by the node in Path messages. This is accomplished by the downstream RSVP neighbor sending a new message for every Path message it has previously received from the restarting node, after reestablishing RSVP communication with a restarted node that supports the recovery procedures defined in Section 4.5.2 of this document.

The new message is called the RecoveryPath message. The message conveys the contents of the last received Path message back to the restarting node. The restarting node can use the RecoveryPath message, along with the state in the received Path message to associate control and data plane state and to validate the forwarding state with the state presented by the neighboring RSVP nodes.

The restarting node indicates its desire to receive and process the RecoveryPath message by including a new object called the Capability object with the RecoveryPath Desired bit set, in its Hello messages sent to the downstream RSVP neighbor. The downstream RSVP neighbor can indicate its ability to send RecoveryPath messages by including the Capability object with the RecoveryPath Transmit Enabled set in its Hello messages to the restarting node. Thus, both the restarting node and its RSVP neighbor, with the help of the Capability object, can detect if the RecoveryPath message extensions defined in this document can be used to recover signaling state after a restart.

If the restarting node is a transit node, it will receive a Path message with a Recovery Label object from its upstream RSVP neighbor. In addition, the RecoveryPath message allows such transit nodes to reconstruct any state that was previously dynamically constructed by the node, e.g., ERO sub-objects. If the restarting node is an ingress node, all significant signaling state can be recovered based on the RecoveryPath message.

Selective transmission of the RecoveryPath message is supported by
enhancing the Summary Refresh mechanisms defined in [RFC2961]. When
Recovery Summary Refresh is supported, the restarting node can select
the LSPs for which it would like to receive RecoveryPath messages.
This is useful when the restarting node is able to locally recover
the signaling state for a subset of the previously active LSPs.

Restarting egress nodes, and Resv message processing are not impacted
by the presented extensions, see [RFC3473] for details.

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].

3. Terminology

The reader is assumed to be familiar with the terminology defined in
[RFC3209] and [RFC3473].

Throughout this document, the term "node", when used in the context
of a restarting or restarted node, generally refers to the control
plane component, which is the signaling controller for a data plane
switch.

4. Extensions to Nodal Fault Handling

This section presents the protocol modifications to Section 9 of
[RFC3473].

4.1. RecoveryPath Message Format

The format of a RecoveryPath message is the same as the format of a
Path message, as defined in [RFC3473], but uses a new message number
(30) so that it can be identified correctly.

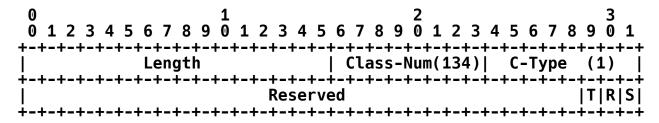   <RecoveryPath Message> ::= <Path Message>

The destination address used in the IP header of a RecoveryPath
message MUST be the same as the destination address used in the IP
header of the corresponding Resv message last generated by the
sending node. Except as specified below, all objects in a
RecoveryPath message are identical to the objects in the
corresponding Path message last received by the sending node.

4.2. Capability Object

Capability objects are carried in RSVP Hello messages. The
Capability object uses Class-Number 134 (of form 10bbbbbb) and C-Type
of 1.

The message format of a Hello message is modified to be:

```
<Hello Message> ::= <Common Header> [ <INTEGRITY> ] <HELLO>
                    [ <RESTART_CAP> ] [ <CAPABILITY> ]
```

The format of a Capability object 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(134)| C-Type  (1)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Reserved                         |T|R|S|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

RecoveryPath Transmit Enabled (T): 1 bit

   When set (1), indicates that the sending node is enabled to
   send RecoveryPath messages. Absence of the Capability object
   MUST be treated as if the T-bit is cleared (0).

RecoveryPath Desired (R): 1 bit

   When set (1), indicates that the sending node desires to
   receive RecoveryPath messages. Absence of the Capability
   object MUST be treated as if the R-bit is cleared (0).

RecoveryPath Srefresh Capable (S): 1 bit

   When set (1), along with the R-bit, indicates that the sending
   node is capable of receiving and processing Srefresh messages
   with the RecoveryPath Flag set (1) in the MESSAGE_ID LIST
   object. Absence of the Capability object MUST be treated as if
   the S-bit is cleared (0). Related procedures are defined in
   Section 5.2.1.

Reserved bits

   Reserved bits MUST be set to zero on transmission and MUST be
   ignored on receipt.

4.2.1. Conformance

All nodes supporting the extensions defined in this document MUST be
able to transmit, and properly receive and process RecoveryPath
messages. All nodes MUST be able to set both the T and R bits. Both
the T and R bits SHOULD be set (1) by default. A node MAY allow
RecoveryPath message transmission and reception to be independently
disabled based on local policy. When RecoveryPath message
transmission is disabled, the T-bit MUST be set to zero (0). When
RecoveryPath message reception is not desired, the R-bit MUST be set
to zero (0).

Any node that supports the extensions defined in this document and
sets the Refresh-Reduction-Capable bit [RFC2961] SHOULD support
setting of the S-bit and support the mechanisms defined in Section 5.

4.3. Related Procedures

This document does not modify existing procedures for sending and
receiving RSVP Hello messages, as defined in [RFC3209], and the
Restart_Cap object in RSVP Hello messages as defined in [RFC3473].
The procedures for control channel faults are defined in [RFC3473]
and are not changed by this document.

The presented extensions require the use of RSVP Hellos, as defined
in [RFC3209], and the use of the Restart_Cap object extension as
defined in [RFC3473]. The presented extensions address only "Nodal
Faults" as defined in [RFC3473]. Control channel faults are fully
addressed in [RFC3473].

Note: There are no changes to the procedures defined in Section 9.5.3
in [RFC3473] (Procedures for the Neighbor of a Restarting node).
There are no changes to the procedures defined in Section 9.5.2 in
[RFC3473] if the restarting node is an egress node.

There are no changes to the procedures with respect to the
data/forwarding plane as described in [RFC3473]. In particular, a
restarting node MUST NOT create data/forwarding plane state as the
result of any of the extensions defined in this document.

The following sections assume previously defined procedures are
followed, except where explicitly modified.

4.4. Procedures for the Capability Object

4.4.1. Procedures for the Downstream Neighbor

If a node is capable of sending RecoveryPath messages, it MUST
include the Capability object with the RecoveryPath Transmit Enabled
(T) bit set (1) in all its Hello messages.

If the downstream RSVP neighbor receives Hello messages from a
restarting node, with the Restart_Cap object, as defined in
[RFC3473], and with the Capability object with the RecoveryPath
Desired (R) bit set (1), it MUST treat the restarting node as capable
of receiving and processing RecoveryPath messages as defined in this
document.

If the downstream RSVP neighbor receives a Capability object in a
Hello message with the RecoveryPath Desired (R) bit set (1), but
without the Restart_Cap object, it MUST process the Hello message as
if the RecoveryPath Receive Desired (R) bit is cleared (0) in the
Hello message.

If the downstream RSVP neighbor does not receive the Capability
object in Hello messages sent by the restarting node or the
RecoveryPath Desired (R) bit is cleared (0) in the Capability object,
it MUST treat the restarting node as not capable of supporting the
RecoveryPath message procedures defined in this document, and MUST
revert to recovery procedures as defined in [RFC3473].

4.4.2. Procedures for the Restarting Node

A node that expects to recover RSVP state by the receipt and
processing of RecoveryPath messages according to procedures defined
in this document, MUST include the Capability object with the
RecoveryPath Desired (R) bit set (1) in its RSVP Hello messages to
its neighbors. The node MUST also include the Restart_Cap object, as
defined in [RFC3473], in all those Hello messages.

If the Recovery Time is zero (0) or the restarting node does not
support/desire the use of RecoveryPath messages, the RecoveryPath
Desired (R) bit MUST be cleared (0) in the Capability object included
in Hello messages, or the Capability object MAY be omitted from Hello
messages sent by the restarting node.

During the Recovery Period, if the restarting node receives Hello
messages from a downstream RSVP neighbor with the RecoveryPath
Transmit Enabled (T) bit set (1) in the Capability object and the
Restart_Cap object, as defined in [RFC3473], it MUST treat the
downstream RSVP neighbor as capable of sending RecoveryPath messages

according to procedures defined in Section 4.5.1. If the restarting
node expects to recover RSVP state by the receipt and processing of
RecoveryPath messages, it MUST follow procedures defined in Section
4.5.2, with the downstream RSVP neighbor.

During the Recovery Period, if the restarting node receives Hello
messages from a downstream RSVP neighbor with the RecoveryPath
Transmit Enabled (T) bit cleared (0) in the Capability object, or,
with the Capability object not present, it MUST treat the downstream
RSVP neighbor as not capable of the RecoveryPath message procedures
defined in this document, and, it MUST revert to the recovery
procedures defined in [RFC3473] immediately, with the downstream RSVP
neighbor.

4.5. Procedures for the RecoveryPath Message

4.5.1. Procedures for the Downstream Neighbor

After a downstream RSVP neighbor has detected that its upstream node
has restarted, is capable of recovery as defined in [RFC3473], and,
is capable of receiving RecoveryPath messages as defined in Section
4.4, the downstream RSVP neighbor MUST send a RecoveryPath message
for each LSP associated with the restarting node for which it has
sent a Resv message. During the Recovery Period, if the downstream
RSVP neighbor detects that the restarting node is not capable of
receiving RecoveryPath messages by the absence of the Capability
object or the RecoveryPath Desired (R) bit cleared (0) in the
Capability object in the restarting node's Hello messages, the
downstream RSVP neighbor SHOULD NOT send the RecoveryPath messages to
the restarting node.

The RecoveryPath message is constructed by copying all objects from
the last received associated Path message, with the following
exceptions:

   The MESSAGE_ID, MESSAGE_ID_ACK and MESSAGE_ID_NACK objects are not
   copied. Any MESSAGE_ID, MESSAGE_ID_ACK and MESSAGE_ID_NACK
   objects used in RecoveryPath messages are generated based on
   procedures defined in [RFC2961].

   The Integrity object is not copied. Any Integrity objects used in
   RecoveryPath messages are generated based on procedures defined in
   [RFC2747].

   The RSVP Hop object is copied from the most recent associated Resv
   message sent to the restarted node for the LSP being recovered.

In the sender descriptor, the Recovery Label object MUST be
included, with the label value copied from the label value in the
Label object in the most recent associated Resv message sent to
the restarted node, for the LSP being recovered.

All other objects from the most recent received Path message MUST be
included in the RecoveryPath message.

All RecoveryPath messages SHOULD be sent at least once within
approximately 1/2 of the Recovery Time advertised by the restarted
neighbor. If there are many LSPs to be recovered by the restarted
node, the downstream RSVP neighbor should avoid sending RecoveryPath
messages in a short time interval to avoid overloading the restarted
node's CPU. Instead, it should spread the messages across 1/2 the
Recovery Time interval. The range of Recovery Time is dependent on
many factors including, but not limited to, the CPU processing power
on the restarting node as well as the upstream and downstream
neighbors, the amount of CPU available for processing RSVP recovery
procedures, and the implementation specifics that affect the amount
of time taken to verify the received recovery state against existing
forwarding plane state. Such discussion is out of scope of this
document.

After sending a RecoveryPath message and during the Recovery Period,
the node SHOULD periodically resend the RecoveryPath message until it
receives a corresponding response. A corresponding response is a
Message ID acknowledgment or a Path message for the LSP the
RecoveryPath message represents. Each such resend attempt is at the
end of any Message ID rapid retransmissions, if the Message ID
mechanism is used. If the Message ID mechanism is not in use, the
period between resend attempts SHOULD be such that at least 3
attempts are completed before the expiry of 3/4 the Recovery Time
interval. Each such resend attempt MUST treat the RecoveryPath
message as a new message and update the MESSAGE_ID object according
to procedures defined in [RFC2961]. Note, per [RFC3473], Resv
messages are suppressed during this recovery period until a
corresponding Path message is received.

4.5.2. Procedures for the Restarting Node

These procedures apply during the "state recovery process" and
"Recovery Period" as defined in Section 9.5.2 of [RFC3473]. Any
RecoveryPath message received after the Recovery Period has expired
SHOULD be matched against local LSP state. If matching fully
resynchronized state is located, the node SHOULD send a Path message
downstream. If non-resynchronized or no LSP state matching the
RecoveryPath message is located, the restarted node MAY send a
PathTear message constructed from the RecoveryPath message to

expedite the cleanup of unrecovered RSVP and associated forwarding
state downstream of the restarted node. The restarting node MUST NOT
create data plane or forwarding state to match the received
RecoveryPath message.

The remaining procedures are broken down into three sub-sections.
The term "resynchronized state", originally defined in [RFC3473], is
used and modified in these sections. This term refers to LSP state
that is fully recovered.

Signaling state may be recovered from sources other than the
mechanisms defined in this document. The restarting node SHOULD
consider signaling state as resynchronized for all such LSPs and
follow corresponding procedures defined below. Further, recovery
procedures defined below may be overridden by local policy.

Again, there are no changes to the procedures defined in Section
9.5.2 in [RFC3473] if the restarting node is an egress node.

4.5.2.1. Path and RecoveryPath Message Procedures

When a node receives a RecoveryPath message during the Recovery
Period, the node first checks if it has resynchronized RSVP state
associated with the message. If there is resynchronized state, and
when both reliable message delivery [RFC2961] is supported and a
MESSAGE_ID object is present in the RecoveryPath message, the node
MUST follow Message ID acknowledgment procedures, as defined in
[RFC2961], and consider the message as processed. If there is
resynchronized state and there is no MESSAGE_ID object or reliable
message delivery [RFC2961] is not supported, the node SHOULD send a
trigger Path message, and, consider the message as processed.

If a non-resynchronized state is found or the node is the ingress,
the node saves the information contained in the RecoveryPath message
and continues with processing as defined in Section 4.5.2.2.

If no associated RSVP state is found and the node is not the ingress
node, the node saves the information contained in the RecoveryPath
message for later use.

Note the following modifies Section 9.5.2 of [RFC3473]:

When a node receives a Path message during the Recovery Period, the
node first locates any RSVP state associated with the message. If
resynchronized RSVP state is found, then the node handles this
message according to previously defined procedures.

If a non-resynchronized state is found, the node saves the
information contained in the Path message, including the
Recovery_Label object, and continues with processing as defined in
Section 4.5.2.2.

Per [RFC3473], if matching RSVP state is not found, and the message
does not carry a Recovery_Label object, the node treats this as a
setup for a new LSP, and handles it according to previously defined
procedures.

If a matching RSVP state is not found and the message carries a
Recovery_Label object, the node saves the information contained in
the Path message, including the Recovery_Label object for later use.

4.5.2.2. Re-Synchronization Procedures

After receipt of the RecoveryPath message and, for non-ingress LSPs,
the corresponding Path message with a Recovery Label object, the
restarting node SHOULD locate and associate corresponding forwarding
state using the received information. The restarting node associates
the corresponding active forwarding plane state from the following
signaled information:

   The upstream data interface is recovered from the RSVP HOP object
   in the received Path message.

   The label on the upstream data interface is recovered from the
   Recovery Label object in the received Path message. If the LSP is
   bidirectional, the label for the upstream direction is recovered
   from the Upstream Label object in the received Path message.

   The downstream data interface is recovered from the RSVP HOP
   object in the received RecoveryPath message.

   The label on the downstream data interface is recovered from the
   Recovery Label object in the received RecoveryPath message. If
   the LSP is bidirectional, the label for the upstream direction is
   recovered from the Upstream Label object in the RecoveryPath
   message.

If complete forwarding state is located, the restarted node MUST
treat the LSP as resynchronized and MUST send a trigger Path message
downstream. The Explicit Route object in the Path message SHOULD
match the Explicit Route object received in the RecoveryPath message.
In addition, the restarted node SHOULD recover state from the other
objects received in the RecoveryPath message. Optimally, the
resulting Path message should not cause any redundant or unnecessary
reprocessing of state along the remaining downstream nodes. Ideally,

except for MESSAGE_ID processing and recovery processing, the
transmitted Path message will be treated as a refresh by the
downstream RSVP neighbor (and hence, should not trigger any
generation of Path messages with changed state further downstream).

If no forwarding state is located, the node treats the received Path
message as a setup request for a new LSP. The outgoing interface and
label(s) indicated in the RecoveryPath message SHOULD be reused when
possible. All other information contained in the RecoveryPath
message MAY also be used. That is, forwarding state MUST NOT be
created except after receipt of a Path message from upstream or, at
an ingress node, the receipt of a command from the management plane.
Further, the forwarding state created is subject to local policy and
the information received from downstream in the RecoveryPath message
is treated only as advisory.

4.5.2.3. Procedures on Expiration of Recovery Period

There are several cleanup steps to follow at the end of the Recovery
Period. At the end of the Recovery Period, any state that was
installed as the result of a received RecoveryPath message that is
not resynchronized SHOULD be discarded.

Any Path messages that were received containing a Recovery_Label that
has not been resynchronized, MUST be treated as being received during
the Recovery Period and processed as per [RFC3473].

Per [RFC3473], any other state that is not resynchronized during the
Recovery Period SHOULD be removed at the end of the Period.

4.6. Compatibility

This document introduces a new RSVP signaling message called the
RecoveryPath message to be generated by the downstream RSVP neighbor
of a restarting node. To advertise the capability of sending and
receiving RecoveryPath messages, this document introduces the
Capability object to be included in Hello messages by a restarting
node and its downstream RSVP neighbors.

If a restarting node does not support the Capability object, it will
discard the object, as the Class-Number is of the form 10bbbbbb, and
revert to recovery processing as defined in [RFC3473]. The
restarting node will not include the Capability object in its Hello
messages. Hence, all downstream RSVP neighbors that detect that the
restarting node is not capable of supporting the extensions defined
in this document will not send the RecoveryPath messages to the
restarting node and will revert to recovery processing as defined in
[RFC3473].

If a downstream RSVP neighbor does not support the Capability object,
it will discard the object received in Hello messages and revert to
recovery processing as defined in [RFC3473]. The downstream RSVP
neighbor will not include the Capability object in its Hello
messages. Hence, the restarting node will detect that the downstream
RSVP neighbor is not capable of supporting the extensions defined in
this document and will revert to recovery processing as defined in
[RFC3473].

5. RecoveryPath Summary Refresh

   This section describes a mechanism to control which LSP state is
   communicated in RecoveryPath messages. This mechanism enhances the
   Summary Refresh mechanism defined in [RFC2961], and uses the
   RecoveryPath Srefresh Capable (S) bit in the Capability object, as
   defined in Section 4.2, carried in the Hello message defined in
   [RFC3209] and [RFC3473]. The described mechanism is referred to as
   RecoveryPath Summary Refresh.

   Selective transmission of RecoveryPath messages is controlled much
   the same way transmission of Path or Resv messages is controlled with
   standard Summary Refresh, see [RFC2961]. In standard Summary
   Refresh, an Srefresh message is sent by a node to identify to its
   neighbor about Path and Resv state that is locally installed and
   available. The receiver of the Srefresh message can then attempt to
   locate matching Path and Resv state. If no matching state is found,
   the receiver can request that the missing state be sent to it by
   sending an Srefresh NACK to the sender of the Srefresh message. When
   the Srefresh NACK is received, the corresponding Path or Resv message
   is sent. MESSAGE_ID information is used to identify Path and Resv
   state in this process.

   The mechanism described in this section extends the Summary Refresh
   process to the Path state that can be represented in RecoveryPath
   messages. In this case, the Srefresh messages represent previously
   received Path messages, rather than previously transmitted Path
   messages. This is the primary difference between standard Summary
   Refresh and RecoveryPath Summary Refresh described in this section.

   When a node restarts, and is capable of supporting the mechanisms
   described in this section, it includes the Capability object with the
   RecoveryPath Desired (R) bit set and the RecoveryPath Srefresh
   Capable (S) bit set in Hello messages it sends to its RSVP neighbors.

   When a neighbor of the restarting node detects a restart (see
   [RFC3209]), it detects that the restarted node is capable of
   receiving RecoveryPath messages, as defined in Section 4.4, and that
   the restarted node is requesting RecoveryPath Srefresh messages by

the RecoveryPath Srefresh Capable (S) bit set in the Capability
object. When such an indication is found, the neighbor generates one
or more Srefresh messages. Each message indicates the Path state
that can be represented in a RecoveryPath message. Within such
Srefresh messages, the Path state that can be represented in
RecoveryPath messages is represented using MESSAGE_ID information,
and this information is communicated within MESSAGE_ID LIST objects.
To indicate that the MESSAGE_ID LIST object is for recovery purposes,
a new flag is set in the MESSAGE_ID LIST object. This flag is called
the RecoveryPath Flag and is defined below.

The restarted node can then use the Srefresh message and the
MESSAGE_ID LIST object to try to identify matching transmitted Path
state. The node identifies local state by matching Epoch and Message
ID tuples against Path messages transmitted downstream prior to the
restart.

If matching state is located, then the restarted node operates as if
a RecoveryPath message has been received, per Section 4.5.2. If no
matching state can be located, the restarted node generates a
Srefresh NACK, see Section 5.4 of [RFC2961]. The Srefresh NACK is
also marked with the new RecoveryPath Flag to indicate that the NACK
is related to RecoveryPath messages.

Upon receiving a Srefresh NACK, the downstream node generates a
RecoveryPath message for the Path state indicated by each entry in
the MESSAGE_ID LIST. The procedures defined in Section 4 above are
then followed by the restarted node and the downstream RSVP neighbor.

5.1. MESSAGE_ID ACK/NACK and MESSAGE_ID LIST Objects

The MESSAGE_ID ACK/NACK objects and the MESSAGE_ID LIST object,
defined in [RFC2961], are updated by this document. A new bit within
the existing Flags field of each object is defined. This bit
indicates that the object carries MESSAGE_ID information related to
Path state that can be recovered using RecoveryPath messages. The
same flag value is used in all the objects for consistency.

MESSAGE_ID_ACK object
MESSAGE_ID_NACK object

See Section 4.3 of [RFC2961] for definition of other fields.

MESSAGE_ID LIST object

See Section 5.1 of [RFC2961] for definition of other fields.

Flags: 8 bits

0x02: RecoveryPath Flag

Indicates that the associated object carries MESSAGE_ID
information related to one or more Path messages that can be
recovered using a RecoveryPath message.

5.2. RecoveryPath Srefresh Capable Bit

The Capability object and the RecoveryPath Srefresh Capable (S) bit
are defined in Section 4.2.

5.2.1. Procedures

To support the selective receipt of RecoveryPath messages as defined
in this section, a restarting node MUST support the receipt and
processing of RecoveryPath messages as defined in Section 4.5.2, and
MUST indicate this capability by including the Capability object with
the RecoveryPath Desired (R) bit set as defined in Section 4.4.2 in
its Hello messages.

To indicate to an RSVP neighbor that selective transmission of
RecoveryPath messages is desired, a node MUST set (1) the S-bit in
the Capability object in all Hello messages it sends. When the
restarting node does not desire the receipt of RecoveryPath messages
(see Section 4.4.2) or the selective transmission mechanism defined
in this section, it MUST clear (0) the S-bit in the Capability object
if included in Hello messages.

The downstream RSVP neighbor checks the R-bit and the S-bit upon
detecting a restart of a node that supports state recovery with
RecoveryPath messages. Detection of neighbor restarts with state
recovery using RecoveryPath messages is defined in Section 4. If
both the R-bit and the S-bit are set, then the procedures defined
below in Section 5.3.1 MUST be followed. If the S-bit is cleared,
the downstream RSVP neighbor MUST revert to normal procedures defined
in Section 4.5.1. If the R-bit is cleared, but the S-bit is set, the

downstream RSVP neighbor MUST treat it as if the Capability object
was received with the S-bit cleared. See Section 4.4 for handling of
Hello messages without the Capability object.

5.2.2. Compatibility

There are no compatibility issues introduced in the procedures
defined in Section 5.2.1.

The restarting node will detect that its neighbor does not support
selective transmission of RecoveryPath messages when a RecoveryPath
message is received prior to the receipt of a Srefresh message
containing a MESSAGE_ID LIST object with the RecoveryPath Flag set
(1). When this occurs, any received RecoveryPath messages MUST be
processed as defined in Section 4.

5.3. RecoveryPath Summary Refresh Procedures

Related processing occurs in the following logical order:

o Generation of RecoveryPath-related Srefresh messages

o RecoveryPath-related Srefresh message receive processing and NACK
   generation

o Message ID NACK receive processing and generation of RecoveryPath
   messages

o Receive processing of RecoveryPath messages

Actual processing MAY result in the above occurring in an interlaced
fashion when multiple LSPs are being recovered. Both the restarted
node and the downstream RSVP neighbor MUST be able to process in this
fashion.

5.3.1. Generation of RecoveryPath-Related Srefresh Messages

A neighbor of a restarting node generates one or more RecoveryPath-
related Srefresh messages when the S-bit is set in the restarted
node's Hello messages as described in Section 5.2.1. The procedures
for generating an Srefresh message are defined in [RFC2961]. Only
modifications to these procedures are described in this section.
Also, Srefresh message transmit and receive processing may occur
simultaneously during the Recovery Period and are not impacted by the
procedures defined in this section.

To generate RecoveryPath-related Srefresh messages, a node must
identify which Path state can be represented in RecoveryPath messages

and which Srefresh message or messages can be used to carry the
related information. As previously mentioned, the Path state that
can be represented in RecoveryPath messages is indicated in Srefresh
messages using the MESSAGE_ID information from the most recently
received Path message associated with the state.

After processing the S-bit as described in Section 5.2.1, the node
identifies all state associated with Path messages received from the
restarted neighbor. Only a Path state that has not been updated
since the restart may be represented in the Srefresh messages.
Received Path state containing a MESSAGE_ID object whose Epoch value
matches the Epoch received in the most recent Hello message is
considered as updated after the upstream neighbor has restarted.
Such Path state MUST NOT be represented in the Srefresh messages.
Each Srefresh message contains one or more MESSAGE_ID LIST objects.
Each such MESSAGE_ID LIST object MUST have the RecoveryPath Flag set
(1).

Multiple MESSAGE_ID LIST objects MAY be included in order to
accommodate multiple Epoch values. The MESSAGE_ID LIST objects
represent the identified, non-updated, Path state. A
Message_Identifier field created for each identified, non-updated
Path state MUST be included in an appropriate MESSAGE_ID LIST object.
The Message_Identifier field is created based on the MESSAGE_ID
object from the most recently received Path message associated with
identified Path state. If any identified Path state does not have an
associated MESSAGE_ID object, this state MUST be processed as defined
above in Section 4.5.1.

The source IP address for the Srefresh message SHOULD be the source
IP address in the IP header of the corresponding Resv messages
previously sent to the restarted node. The Srefresh message SHOULD
be destined to the IP address in the HOP object in the corresponding
Path messages. This may result in multiple Srefresh messages being
generated. Per [RFC2961], implementations may choose to limit each
Srefresh message to the MTU size of the outgoing link, and to not
bundle Srefresh messages. RecoveryPath-related Srefresh messages
SHOULD be sent using reliable delivery, as defined in [RFC2961].

During the Recovery Period, unacknowledged RecoveryPath-related
Srefresh messages SHOULD be periodically transmitted. The
retransmission algorithm used can be the same algorithm used for
retransmitting RecoveryPath messages during the Recovery Period (see
Section 4.5.1). Note that prior to each such periodic
retransmission, the Srefresh message SHOULD be updated to exclude the
Message ID's of Path state that has been updated by the receipt of a
Path message.

To allow sufficient processing time for the restarted node, the
downstream RSVP neighbor MAY choose to generate multiple
RecoveryPath-related Srefresh messages containing partial but
mutually exclusive sets of Message Identifiers spread across 1/4 of
the Recovery Time advertised by the restarted node.

5.3.2. RecoveryPath-Related Srefresh Receive Processing and NACK
        Generation

Upon receiving an Srefresh message containing a MESSAGE_ID LIST
object with the RecoveryPath Flag set), the restarted node attempts
to locate matching previously transmitted Path state. The Epoch in
the MESSAGE_ID LIST object, along with each Message Identifier in the
object, is used to match against the MESSAGE_ID object in Path
messages previously transmitted to the downstream RSVP neighbor. For
each Message Identifier in the MESSAGE_ID LIST:

   If matching transmitted Path state is found, the restarting node
   treats the corresponding LSP state as having received and
   processed a RecoveryPath message and perform any further
   processing necessary as defined in Section 4.5.2. Specifically,
   it MUST generate a trigger Path message for the LSP as defined in
   Section 4.5.2.2. The restarted node MAY spread the transmission
   of such trigger Path messages across 1/2 of the remaining Recovery
   Period to allow the downstream RSVP neighbor sufficient processing
   time.

   If matching transmitted Path state is not found, the restarting
   node MUST generate a MESSAGE_ID NACK as defined in [RFC2961].
   Each generated MESSAGE_ID NACK MUST have the RecoveryPath Flag set
   (1).

It is recommended that the restarted node combine multiple such
MESSAGE_ID NACKs into a single ACK message, per [RFC2961].

5.3.3. RecoveryPath-Related MESSAGE_ID NACK Receive Processing

This section defines the procedures associated with the processing of
received MESSAGE_ID NACKs that have the RecoveryPath Flag set (1).
Procedures for processing of MESSAGE_ID NACKs without the
RecoveryPath Flag present are defined in [RFC2961] and not modified
in this document. Processing of MESSAGE_ID NACKs with the
RecoveryPath Flag set (1) also follows procedures defined in
[RFC2961] unless explicitly modified in this section.

For each MESSAGE_ID NACK with the RecoveryPath Flag set (1), the
downstream RSVP neighbor must locate the matching received Path
message. If a matching Path message is found, the downstream RSVP

neighbor MUST generate a RecoveryPath message as defined in Section
4.5.1. If a matching Path message is not found, the MESSAGE_ID NACK
is ignored. An example where this may occur is when the restarted
node has already generated an updated Path message after its restart.

6. Security Considerations

This document introduces a new RSVP message that is restricted to one
RSVP hop. This document introduces no new security considerations
beyond those already addressed for existing RSVP hop-by-hop messages.

This document introduces a new RSVP object to be included in RSVP
Hello messages. This document introduces no new security
considerations beyond those already addressed for existing objects in
RSVP Hello messages.

This document introduces new procedures to be performed on RSVP
agents that neighbor a restarting RSVP agent. In situations where
the control plane in general, and the RSVP agent in particular, of a
node carrying one or more LSPs is restarted due to external attacks,
the procedures introduced in this document provide the ability for
the restarting RSVP agent to recover the RSVP state corresponding to
the LSPs with the least possible perturbation to the rest of the
network. Ideally, only the neighboring RSVP agents should notice the
restart and hence need to perform additional processing. This allows
for a network with active LSPs to recover LSP state gracefully from
an external attack without perturbing the data/forwarding plane
state.

[RFC2747] provides mechanisms to protect against external agents
compromising the RSVP signaling state in an RSVP agent. These
mechanisms, when used with the new message and procedures introduced
in this document, provide the same degree of protection to the
restarting RSVP agent against installing compromised signaling state
from an external agent during its RSVP signaling state recovery.

Note that the procedures assume a full trust model between RSVP
neighbors. That is, although the protocol exchanges before and after
restart can be secured, and although it is possible to authenticate
the identity of the neighbors, no mechanism is provided to verify
that the restart information is correctly mapped from the protocol
information exchanged before the restart. This is considered
acceptable because a similar trust model is required for normal
operation of the protocol.

The procedures defined in this document introduce additional
processing overhead for the RSVP agents that neighbor a restarting
RSVP agent. If an RSVP agent restarts due to external attacks, such

added processing on the neighboring RSVP agents may impact their
ability to perform other control plane tasks, including any
processing for other LSPs that do not involve the restarting node.
Such impact can be minimalized by the restarting RSVP agent using a
large enough Recovery Time, so that its neighbors are provided
sufficient time to handle the additional processing involved while
continuing to perform their other control plane functions normally
during the Recovery Period.

Note that the procedures defined in this document cannot be used to
create false forwarding state. The restarting node that receives a
RecoveryPath message that does not match the existing forwarding
state MUST NOT create or modify its forwarding state to match. A
restarting node SHOULD log such an event or otherwise notify the
operator since it might represent an attack.

7. Acknowledgments

The authors would like to thank participants of the CCAMP WG for
comments and suggestions. Also thanks to Arthi Ayyangar, Adrian
Farrel, Nick Neate, and Pavan Beeram for their helpful comments and
feedback.

Derek Atkins provided useful discussion during SecDir review. Sam
Hartman gave careful scrutiny of the security considerations and the
potential impact on the RSVP-TE security trust model.

Adrian Farrel edited the final revisions of this document as it
progressed through IESG review.

8. IANA Considerations

[RFC2205] defines the Class-Number name space for RSVP objects. The
name space is managed by IANA.

A new RSVP object using a Class-Number of form 10bbbbbb called the
Capability Object is defined in Section 4.2 in this document. The
Class-Number is 134.

A new RSVP message type called a RecoveryPath message is defined in
Section 4.1 of this document. The RSVP message type is 30.

This document creates a new name space in the Capability object
defined in Section 4.2. The new name space is a 32-bit-wide field.
New registrations in this name space are to be allocated by IANA
through an IETF Consensus action, per [RFC2434]. IANA also serves as
the repository for this name space.

Section 4.2 defines the following bits in the 32-bit field of the
Capability Object (134):

    RecoveryPath Transmit Enabled (T): 1 bit
    RecoveryPath Desired (R): 1 bit
    RecoveryPath Srefresh Capable (S): 1 bit

9. Normative References

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

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

    [RFC2434]   Narten, T. and H. Alvestrand, "Guidelines for Writing an
                IANA Considerations Section in RFCs", BCP 26, RFC 2434,
                October 1998.

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

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

Editors' Addresses

   Arun Satyanarayana (editor)
   Cisco Systems, Inc.
   170 West Tasman Dr.
   San Jose, CA   95134
   USA
   Phone: +1 408 853 3206
   EMail: asatyana@cisco.com

   Reshad Rahman (editor)
   Cisco Systems, Inc.
   2000 Innovation Dr.
   Kanata, Ontario K2K 3E8
   Canada
   Phone: 613 254 3519
   EMail: rrahman@cisco.com

Authors' Addresses

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

   Lou Berger
   LabN Consulting, L.L.C.
   Phone: +1 301 468 9228
   EMail: lberger@labn.net

   Anca Zamfir
   Cisco Systems, Inc.
   2000 Innovation Dr.
   Kanata, Ontario K2K 3E8
   Canada
   Phone: 613 254 3484
   EMail: ancaz@cisco.com

   Junaid Israr
   Cisco Systems, Inc.
   2000 Innovation Dr.
   Kanata, Ontario K2K 3E8
   Canada
   Phone: 613 254 3693
   EMail: jisrar@cisco.com