Network Working Group
Internet Draft
Expiration Date: September 2005

A. Satyanarayana (Cisco Systems)
R. Rahman (Cisco Systems)
Editors

March 2005

Extensions to GMPLS RSVP Graceful Restart

draft-ietf-ccamp-rsvp-restart-ext-02.txt

Status of this Memo

Abstract

This document describes extensions to the 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. These mechanisms do not fully support signaling state recovery on ingress nodes or recovery of all RSVP objects. The presented extensions use the RSVP Hello extensions defined in RFC 3209, and extensions for state recovery on nodal faults defined in RFC 3473. With the presented extensions the restarting node can recover all previously transmitted Path state including the ERO and the downstream (outgoing) interface identifiers. The extensions can also be used to recover signaling state after the restart of an ingress node. 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 LSP's.

Contents

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 RFC 2119 [RFC2119].

1. Introduction

RSVP Graceful Restart is defined in [RFC3473] and uses mechanisms defined in [RFC3209]. [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 ERO, from its downstream neighbors. The extensions also enable graceful restart of an ingress node that does not preserve full RSVP state across restarts.

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, 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, after reestablishing RSVP communication with the restarted node, sending a new message for every Path message it has previously received from the restarting node.

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.

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 LSP's 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 LSP's.

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

2. Extensions to Nodal Fault Handling

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

2.1. RecoveryPath Message Format

The format of a RecoveryPath message is the same as the format of a Path message as defined in [RFC3473]:

<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.

2.2. Related Procedures

This document does not modify existing procedures for sending and receiving RSVP Hello messages as defined in [RFC3209] and the Restart_Caps 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_Caps 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.

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

2.3. Procedures For The Downstream Neighbor

After a downstream RSVP neighbor has detected that its upstream node has restarted and is capable of recovery as defined in [RFC3473], 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.

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 within approximately 1/2 of the Recovery Time advertised by the restarted neighbor. If there are many LSP's 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.

After sending a RecoveryPath message and during the Recovery Period, the node SHOULD periodically re-send 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 re-send attempt is at the end of any Message ID rapid retransmissions, if the Message ID mechanism is used. If the Message ID mechanim is not in use, the period between re-send attempts SHOULD be such that at least 3 attempts are completed before the expiry of 1/2 the Recovery Time

interval. Each such re-send 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.

2.4. Procedures for the Restarting Node

These procedures apply during the "state recovery process" and "Recovery Period" as defined in Section 9.5.2 in [RFC3473]. Any RecoveryPath message received after the Recovery Period has expired MUST be discarded. If 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 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.

2.4.1. 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 triggered Path message, and, consider the message as processed.

If 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 2.4.2.

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 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 2.4.2.

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 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.

2.4.2. 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 triggered 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 re-processing 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 path message as a setup 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.

2.4.3. 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 have not been resynchronized, SHOULD 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.

2.5. Compatibility

This document introduces a new RSVP signaling message to be generated by the downstream RSVP neighbor of a restarting node.

If the restarting node does not support the RecoveryPath message and associated procedures, it will discard all received RecoveryPath messages, and revert to recovery processing as defined in [RFC3473].

If the downstream RSVP neighbor does not support the RecoveryPath message and associated procedures, the restarting node processes received Path messages as defined in Section 2.4.3, which essentially reverts to the processing defined in [RFC3473].

3. 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 a new object 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 a new object in Hello messages it sends to its RSVP neighbors. The new object is defined below in Section 3.2 and is called the Capability object. A bit carried within the Capability object indicates when a restarting node desires its downstream neighbor to use the mechanisms described in this section. This bit is called the RecoveryPath Srefresh Capable bit.

When a neighbor of the restarting node detects a restart, see [RFC3209], it detects that the restarted node is requesting RecoveryPath Srefresh messages by the presence of the Capability object with the RecoveryPath Srefresh Capable bit set. 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, 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 2.4. 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 above in Section 2 are then followed by the restarted node and the downstream RSVP neighbor.

3.1. 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.

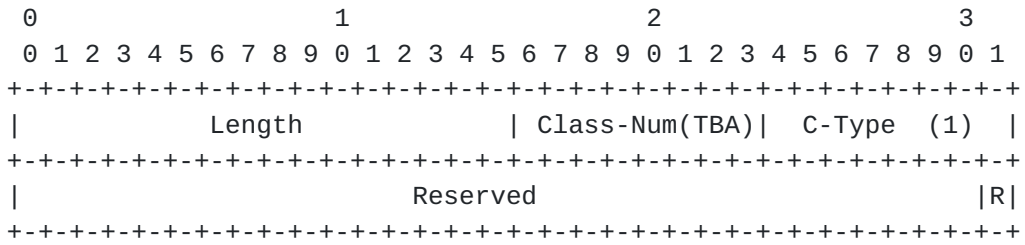
3.2. Capability Object

Capability objects are carried in RSVP Hello messages. The Capability object uses Class-Number TBA (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:



RecoveryPath Srefresh Capable (R): 1 bit

When set (1), 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 R-bit is cleared (0).

Reserved bits

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

3.2.1. Procedures

The Capability object is sent within a Hello message to indicate that a node supports selective transmission of RecoveryPath messages. To indicate to a neighbor that selective transmission of RecoveryPath messages is desired, a restarting node MUST include the Capability object with the R-bit set (1) in all Hello messages it sends during the Recovery Period to the neighbor. When either the Restart_Cap Object is not present in a Hello message or when the Recovery Time is zero (0), the Capability object MUST be omitted or the R-bit MUST be cleared (0).

On the downstream neighbor, the R-bit is checked upon detecting a restart of a neighbor that supports state recovery. Detection of neighbor restarts with state recovery is defined in [RFC3473]. When a node supports RecoveryPath Summary Refresh, and it detects a restart of a neighbor that supports state recovery, it MUST check to see if the received Hello message contains a Capability object with the R-bit set. If the R-bit is set, then the procedures defined below in Section 3.3.1 MUST be followed. If the Capability object is not found or the R-bit not set, then the node MUST revert to normal recovery procedures as defined above in Section 2.3.

3.2.2. Compatibility

There are no compatibility issues introduced in this section. The use of the Capability object will not cause any issues on a non-supporting receiver as it uses a Class-Number of form 10bbbbbb. The object will simply be ignored, and normal processing will continue. Normal processing includes procedures defined above in Section 2, in [RFC3473] and in [RFC3209].

The sender of the Capability object 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 above in Section 2.

3.3. RecoveryPath Summary Refresh Procedures

Related processing occurs in the following logical order:

- Generation of RecoveryPath-related Srefresh messages
- RecoveryPath-related Srefresh message receive processing and NACK generation
- Message ID NACK receive processing and generation of RecoveryPath messages
- Receive processing of RecoveryPath messages

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

3.3.1. Generation of RecoveryPath-related Srefresh Messages

A neighbor of a restarting node generates one or more RecoveryPath-related Srefresh messages when the R-bit is set in the restarted node's Hello messages as described above in Section 3.2.1. The procedures for generating an Srefresh message are defined in [RFC2961]. Only modifications to these procedures are described 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 R-bit as described in Section 3.2.1, the node identifies all state associated with Path messages received from the restarted neighbor. Only 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 2.3.

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 same algorithm used for retransmitting RecoveryPath messages during the Recovery Period (see section 2.3). 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.

3.3.2. 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 2.4. Specifically, it MUST generate a triggered Path message for the LSP as defined in Section 2.4.2. The restarted node MAY spread the transmission of such triggered 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 NACK's into a single ACK message, per [RFC2961].

3.3.3. RecoveryPath-related MESSAGE_ID NACK Receive Processing

This section defines the procedures associated with the processing of received MESSAGE_ID NACK's which have the RecoveryPath Flag set (1). Procedures for processing of MESSAGE_ID NACK's without the RecoveryPath Flag present are defined in [RFC2961] and not modified in this document. Processing of MESSAGE_ID NACK's 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 2.3. 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.

4. Acknowledgments

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

5. 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.

6. IANA Considerations

A new RSVP message type is defined in this document. The RSVP message type is TBA by IANA.

A new RSVP object of form 10bbbbbb is defined in this document. The Class-Num is TBA by IANA.

7. References

7.1. Normative References

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

[RFC2205] "Resource ReSerVation Protocol (RSVP) - Version 1, Functional Specification", RFC 2205, Braden, et al, September 1997.

[RFC2747] "RSVP Cryptographic Authentication", RFC 2747, F. Baker, et al, January 2000.

[RFC2961] "RSVP Refresh Overhead Reduction Extensions", RFC 2961, L. Berger, et al, April 2001.

[RFC3209] "Extensions to RSVP for LSP Tunnels", D. Awduche, et al, RFC 3209, December 2001.

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

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

7.2. Informative References

[RESTART] "RSVP Graceful Restart Extensions", draft-rahman-rsvp-restart-extensions-00, R. Rahman, et al, October 2003

8. Authors' Addresses

Arun Satyanarayana
Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA 95134
Phone: +1 408 853-3206
Email: asatyana@cisco.com

Lou Berger
Movaz Networks, Inc.
7926 Jones Branch Drive
Suite 615
McLean VA, 22102
Phone: +1 703 847-1801
Email: lberger@movaz.com

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

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

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

9. Intellectual Property Considerations

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

10. Disclaimer of Validity

11. Full Copyright Statement