Network Working Group
Request for Comments: 5495
Category: Informational

D. Li
J. Gao
Huawei
A. Satyanarayana
Cisco
S. Bardalai
Fujitsu

March 2009

Description of the Resource Reservation Protocol - Traffic-Engineered (RSVP-TE) Graceful Restart Procedures

Status of This Memo

Copyright Notice

Abstract

The Hello message for the Resource Reservation Protocol (RSVP) has been defined to establish and maintain basic signaling node adjacencies for Label Switching Routers (LSRs) participating in a Multiprotocol Label Switching (MPLS) traffic-engineered (TE) network. The Hello message has been extended for use in Generalized MPLS (GMPLS) networks for state recovery of control channel or nodal faults.

The GMPLS protocol definitions for RSVP also allow a restarting node to learn which label it previously allocated for use on a Label Switched Path (LSP).

Further RSVP protocol extensions have been defined to enable a restarting node to recover full control plane state by exchanging RSVP messages with its upstream and downstream neighbors.

This document provides an informational clarification of the control plane procedures for a GMPLS network when there are multiple node failures, and describes how full control plane state can be recovered in different scenarios where the order in which the nodes restart is different.

This document does not define any new processes or procedures. All protocol mechanisms are already defined in the referenced documents.

Table of Contents

1. Introduction

The Hello message for the Resource Reservation Protocol (RSVP) has been defined to establish and maintain basic signaling node adjacencies for Label Switching Routers (LSRs) participating in a Multiprotocol Label Switching (MPLS) traffic-engineered (TE) network [RFC3209]. The Hello message has been extended for use in Generalized MPLS (GMPLS) networks for state recovery of control channel or nodal faults through the exchange of the Restart_Cap Object [RFC3473].

The GMPLS protocol definitions for RSVP [RFC3473] also allow a restarting node to learn which label it previously allocated for use on a Label Switched Path (LSP) through the Recovery_Label Object carried on a Path message sent to a restarting node from its upstream neighbor.

Further RSVP protocol extensions have been defined [RFC5063] to perform graceful restart and to enable a restarting node to recover full control plane state by exchanging RSVP messages with its upstream and downstream neighbors. State previously transmitted to the upstream neighbor (principally, the downstream label) is recovered from the upstream neighbor on a Path message (using the

Recovery_Label Object as described in [RFC3473]). State previously transmitted to the downstream neighbor (including the upstream label, interface identifiers, and the explicit route) is recovered from the downstream neighbor using a RecoveryPath message.

[RFC5063] also extends the Hello message to exchange information about the ability to support the RecoveryPath message.

The examples and procedures in [RFC3473] and [RFC5063] focus on the description of a single node restart when adjacent network nodes are operative. Although the procedures are equally applicable to multi-node restarts, no detailed explanation is provided for such a case.

This document provides an informational clarification of the control plane procedures for a GMPLS network when there are multiple node failures, and describes how full control plane state can be recovered in different scenarios where the order in which the nodes restart is different.

This document does not define any new processes or procedures. All protocol mechanisms already defined in [RFC3473] and [RFC5063] are definitive.

2. Existing Procedures for Single Node Restart

This section documents for information the existing procedures defined in [RFC3473] and [RFC5063]. Those documents are definitive, and the description here is non-normative. It is provided for informational clarification only.

2.1. Procedures Defined in RFC 3473

In the case of nodal faults, the procedures for the restarting node and the procedures for the neighbor of a restarting node are applied to the corresponding nodes. These procedures, described in [RFC3473], are summarized as follows:

For the Restarting Node:

1) Tells its neighbors that state recovery is supported using the Hello message.

2) Recovers its RSVP state with the help of a Path message, received from its upstream neighbor, that carries the Recovery_Label Object.

3) For bidirectional LSPs, uses the Upstream_Label Object on the received Path message to recover the corresponding RSVP state.

4) If the corresponding forwarding state in the data plane does not exist, the node treats this as a setup for a new LSP. If the forwarding state in the data plane does exist, the forwarding state is bound to the LSP associated with the message, and the related forwarding state should be considered as valid and refreshed. In addition, if the node is not the tail-end of the LSP, the incoming label on the downstream interface is retrieved from the forwarding state on the restarting node and set in the Upstream_Label Object in the Path message sent to the downstream neighbor.

For the Neighbor of a Restarting Node:

1) Sends a Path message with the Recovery_Label Object containing a label value corresponding to the label value received in the most recently received corresponding Resv message.

2) Resumes refreshing Path state with the restarting node.

3) Resumes refreshing Resv state with the restarting node.

2.2. Procedures Defined in RFC 5063

A new message is introduced in [RFC5063] called the RecoveryPath message. This message is sent by the downstream neighbor of a restarting node to convey the contents of the last received Path message back to the restarting node.

The restarting node will receive the Path message with the Recovery_Label Object from its upstream neighbor and/or the RecoveryPath message from its downstream neighbor. The full RSVP state of the restarting node can be recovered from these two messages.

The following state can be recovered from the received Path message:

- Upstream data interface (from RSVP_Hop Object)
- Label on the upstream data interface (from Recovery_Label Object)
- Upstream label for bidirectional LSP (from Upstream_Label Object)

The following state can be recovered from the received RecoveryPath message:

- Downstream data interface (from RSVP_Hop Object)
- Label on the downstream data interface (from Recovery_Label Object)

o Upstream direction label for bidirectional LSP (from Upstream_Label Object)

The other objects originally exchanged on Path and Resv messages can be recovered from the regular Path and Resv refresh messages, or from the RecoveryPath.

3. Multiple Node Restart Scenarios

We define the following terms for the different node types:

Restarting - The node has restarted. Communication with its neighbor nodes is restored, and its RSVP state is under recovery.

Delayed Restarting - The node has restarted, but the communication with a neighbor node is interrupted (for example, the neighbor node needs to restart).

Normal - The normal node is the fully operational neighbor of a restarting or delayed restarting node.

There are five scenarios for multi-node restart. We will focus on the different positions of a restarting node. As shown in Figure 1, an LSP starts from Node A, traverses Nodes B and C, and ends at Node D.

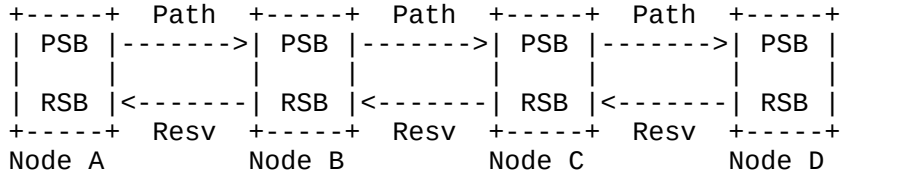

Figure 1: Two Neighbor Nodes Restart

1) A restarting node with downstream delayed restarting node. For example, in Figure 1, Nodes A and D are normal nodes, Node B is a restarting node, and Node C is a delayed restarting node.

2) A restarting node with upstream delayed restarting node. For example, in Figure 1, Nodes A and D are normal nodes, Node B is a delayed restarting node, and Node C is a restarting node.

3) A restarting node with downstream and upstream delayed restarting nodes. For example, in Figure 1, Node A is a normal node, Nodes B and D are delayed restarting nodes, and Node C is a restarting node.

4) A restarting ingress node with downstream delayed restarting node. For example, in Figure 1, Node A is a restarting node and Node B is a delayed restarting node. Nodes C and D are normal nodes.

5) A restarting egress node with upstream delayed restarting node. For example, in Figure 1, Nodes A and B are normal nodes, Node C is a delayed restarting node, and Node D is a restarting node.

If the communication between two nodes is interrupted, the upstream node may think the downstream node is a delayed restarting node, or vice versa.

Note that if multiple nodes that are not neighbors are restarted, the restart procedures could be applied as multiple separated restart procedures that are exactly the same as the procedures described in [RFC3473] and [RFC5063]. Therefore, these scenarios are not described in this document. For example, in Figure 1, Node A and Node C are normal nodes, and Node B and Node D are restarting nodes; therefore, Node B could be restarted through Node A and Node C, while Node D could be restarted through Node C separately.

4. RSVP State

For each scenario, the RSVP state that needs to be recovered at the restarting nodes are the Path State Block (PSB) and Resv State Block (RSB), which are created when the node receives the corresponding Path message and Resv message.

According to [RFC2209], how to construct the PSB and RSB is really an implementation issue. In fact, there is no requirement to maintain separate PSB and RSB data structures. In GMPLS, there is a much closer tie between Path and Resv state so it is possible to combine the information into a single state block (the LSP state block). On the other hand, if point-to-multipoint is supported, it may be convenient to maintain separate upstream and downstream state. Note that the PSB and RSB are not upstream and downstream state since the PSB is responsible for receiving a Path from upstream and sending a Path to downstream.

Regardless of how the RSVP state is implemented, on recovery there are two logical pieces of state to be recovered and these correspond to the PSB and RSB.

5. Procedures for Multiple Node Restart

In this document, all the nodes are assumed to have the graceful restart capabilities that are described in [RFC3473] and [RFC5063].

5.1. Procedures for the Normal Node

When the downstream normal node detects its neighbor restarting, it must send a RecoveryPath message for each LSP associated with the restarting node for which it has previously sent a Resv message and which has not been torn down.

When the upstream normal node detects its neighbor restarting, it must send a Path message with a Recovery_Label Object containing a label value corresponding to the label value received in the most recently received corresponding Resv message.

This document does not modify the procedures for the normal node, which are described in [RFC3473] and [RFC5063].

5.2. Procedures for the Restarting Node

This document does not modify the procedures for the restarting node, which are described in [RFC3473] and [RFC5063].

5.2.1. Procedures for Scenario 1

After the restarting node restarts, it starts a Recovery Timer. Any RSVP state that has not been resynchronized when the Recovery Timer expires should be cleared.

At the restarting node (Node B in the example), full resynchronization with the upstream neighbor (Node A) is possible because Node A is a normal node. The upstream Path information is recovered from the Path message received from Node A. Node B also recovers the upstream Resv information (that it had previously sent to Node A) from the Recovery_Label Object carried in the Path message received from Node A, but, obviously, some information (like the Recorded_Route Object) will be missing from the new Resv message generated by Node B and cannot be supplied until the downstream delayed restarting node (Node C) restarts and sends a Resv.

After the upstream Path information and upstream Resv information have been recovered by Node B, the normal refresh procedure with upstream Node A should be started.

As per [RFC5063], the restarting node (Node B) would normally expect to receive a RecoveryPath message from its downstream neighbor (Node C). It would use this to recover the downstream Path information, and would subsequently send a Path message to its downstream neighbor and receive a Resv message. But in this scenario, because the downstream neighbor has not restarted yet, Node B detects the communication with

Node C is interrupted and must wait before resynchronizing with its downstream neighbor.

In this case, the restarting node (Node B) follows the procedures in Section 9.3 of [RFC3473] and may run a Restart Timer to wait for the downstream neighbor (Node C) to restart. If its downstream neighbor (Node C) has not restarted before the timer expires, the corresponding LSPs may be torn down according to local policy [RFC3473]. Note, however, that the Restart Time value suggested in [RFC3473] is based on the previous Hello message exchanged with the node that has not restarted yet (Node C). Since this time value is unlikely to be available to the restarting node (Node B), a configured time value must be used if the timer is operated.

The RSVP state must be reconciled with the retained data plane state if the cross-connect information can be retrieved from the data plane. In the event of any mismatches, local policy will dictate the action that must be taken, which could include:

- reprogramming the data plane

- sending an alert to the management plane

- tearing down the control plane state for the LSP

In the case that the delayed restarting node never comes back and a Restart Timer is not used to automatically tear down LSPs, the LSPs can be tidied up through the control plane using a PathTear from the upstream node (Node A). Note that if Node C restarts after this operation, the RecoveryPath message that it sends to Node B will not be matched with any state on Node B and will receive a PathTear as its response, resulting in the teardown of the LSP at all downstream nodes.

5.2.2. Procedures for Scenario 2

In this case, the restarting node (Node C) can recover full downstream state from its downstream neighbor (Node D), which is a normal node. The downstream Path state can be recovered from the RecoveryPath message, which is sent by Node D. This allows Node C to send a Path refresh message to Node D, and Node D will respond with a Resv message from which Node C can reconstruct the downstream Resv state.

After the downstream Path information and downstream Resv information have been recovered in Node C, the normal refresh procedure with downstream Node D should be started.

The restarting node would normally expect to resynchronize with its upstream neighbor to re-learn the upstream Path and Resv state, but in this scenario, because the upstream neighbor (Node B) has not restarted yet, the restarting node (Node C) detects that the communication with upstream neighbor (Node B) is interrupted. The restarting node (Node C) follows the procedures in Section 9.3 of [RFC3473] and may run a Restart Timer to wait for the upstream neighbor (Node B) to restart. If its upstream neighbor (Node B) has not restarted before the Restart Timer expires, the corresponding LSPs may be torn down according to local policy [RFC3473]. Note, however, that the Restart Time value suggested in [RFC3473] is based on the previous Hello message exchanged with the node that has not restarted yet (Node B). Since this time value is unlikely to be available to the restarting node (Node C), a configured time value must be used if the timer is operated.

Note that no Resv message is sent to the upstream neighbor (Node B), because it has not restarted.

The RSVP state must be reconciled with the retained data plane state if the cross-connect information can be retrieved from the data plane.

In the event of any mismatches, local policy will dictate the action that must be taken, which could include:

- reprogramming the data plane

- sending an alert to the management plane

- tearing down the control plane state for the LSP

In the case that the delayed restarting node never comes back and a Restart Timer is not used to automatically tear down LSPs, the LSPs cannot be tidied up through the control plane using a PathTear from the upstream node (Node A), because there is no control plane connectivity to Node C from the upstream direction. There are two possibilities in [RFC3473]:

- Management action may be taken at the restarting node to tear the LSP. This will result in the LSP being removed from Node C and a PathTear being sent downstream to Node D.

- Management action may be taken at any downstream node (for example, Node D), resulting in a PathErr message with the Path_State_Removed flag set being sent to Node C to tear the LSP state.

Note that if Node B restarts after this operation, the Path message that it sends to Node C will not be matched with any state on Node C and will be treated as a new Path message, resulting in LSP setup. Node C should use the labels carried in the Path message (in the Upstream_Label Object and in the Recovery_Label Object) to drive its label allocation, but may use other labels according to normal LSP setup rules.

5.2.3. Procedures for Scenario 3

In this example, the restarting node (Node C) is isolated. Its upstream and downstream neighbors have not restarted.

The restarting node (Node C) follows the procedures in Section 9.3 of [RFC3473] and may run a Restart Timer for each of its neighbors (Nodes B and D). If a neighbor has not restarted before its Restart Timer expires, the corresponding LSPs may be torn down according to local policy [RFC3473]. Note, however, that the Restart Time values suggested in [RFC3473] are based on the previous Hello message exchanged with the nodes that have not restarted yet. Since these time values are unlikely to be available to the restarting node (Node C), a configured time value must be used if the timer is operated.

During the Recovery Time, if the upstream delayed restarting node has restarted, the procedure for scenario 1 can be applied.

During the Recovery Time, if the downstream delayed restarting node has restarted, the procedure for scenario 2 can be applied.

In the case that neither delayed restarting node ever comes back and a Restart Timer is not used to automatically tear down LSPs, management intervention is required to tidy up the control plane and the data plane on the node that is waiting for the failed device to restart.

If the downstream delayed restarting node restarts after the cleanup of LSPs at Node C, the RecoveryPath message from Node D will be responded to with a PathTear message. If the upstream delayed restarting node restarts after the cleanup of LSPs at Node C, the Path message from Node B will be treated as a new LSP setup request, but the setup will fail because Node D cannot be reached; Node C will respond with a PathErr message. Since this happens to Node B during its restart processing, it should follow the rules of [RFC5063] and tear down the LSP.

5.2.4. Procedures for Scenario 4

When the ingress node (Node A) restarts, it does not know which LSPs it caused to be created. Usually, however, this information is retrieved from the management plane or from the configuration requests stored in non-volatile form in the node in order to recover the LSP state.

Furthermore, if the downstream node (Node B) is a normal node, according to the procedures in [RFC5063], the ingress will receive a RecoveryPath message and will understand that it was the ingress of the LSP.

However, in this scenario, the downstream node is a delayed restarting node, so Node A must either rely on the information from the management plane or stored configuration, or it must wait for Node B to restart.

In the event that Node B never restarts, management plane intervention is needed at Node A to clean up any LSP control plane state restored from the management plane or from local configuration, and to release any data plane resources.

5.2.5. Procedures for Scenario 5

In this scenario, the egress node (Node D) restarts, and its upstream neighbor (Node C) has not restarted. In this case, the egress node may have no control plane state relating to the LSPs. It has no downstream neighbor to help it and no management plane or configuration information, although there will be data plane state for the LSP. The egress node must simply wait until its upstream neighbor restarts and gives it the information in Path messages carrying Recovery_Label Objects.

5.3. Consideration of the Reuse of Data Plane Resources

Fundamental to the processes described above is an understanding that data plane resources may remain in use (allocated and cross-connected) when control plane state has not been fully resynchronized because some control plane nodes have not restarted.

It is assumed that these data plane resources might be carrying traffic and should not be reconfigured except through application of operator-configured policy, or as a direct result of operator action.

In particular, new LSP setup requests from the control plane or the management plane should not be allowed to use data plane resources

that are still in use. Specific action must first be taken to release the resources.

5.4. Consideration of Management Plane Intervention

The management plane must always retain the ability to control data plane resources and to override the control plane. In this context, the management plane must always be able to release data plane resources that were previously in place for use by control-plane-established LSPs. Further, the management plane must always be able to instruct any control plane node to tear down any LSP.

Operators should be aware of the risks of misconnection that could be caused by careless manipulation from the management plane of in-use data plane resources.

6. Clarification of Restarting Node Procedure

According to the current graceful restart procedure [RFC3473], after a node restarts its control plane, it needs its upstream node to send a PATH message with a recovery label in order to synchronize its RSVP state. If the restarted control plane becomes operational quickly, the upstream node may not detect the restarting of the downstream node and, therefore, may send a PATH message without a recovery label, causing errors and unwanted connection deletion.

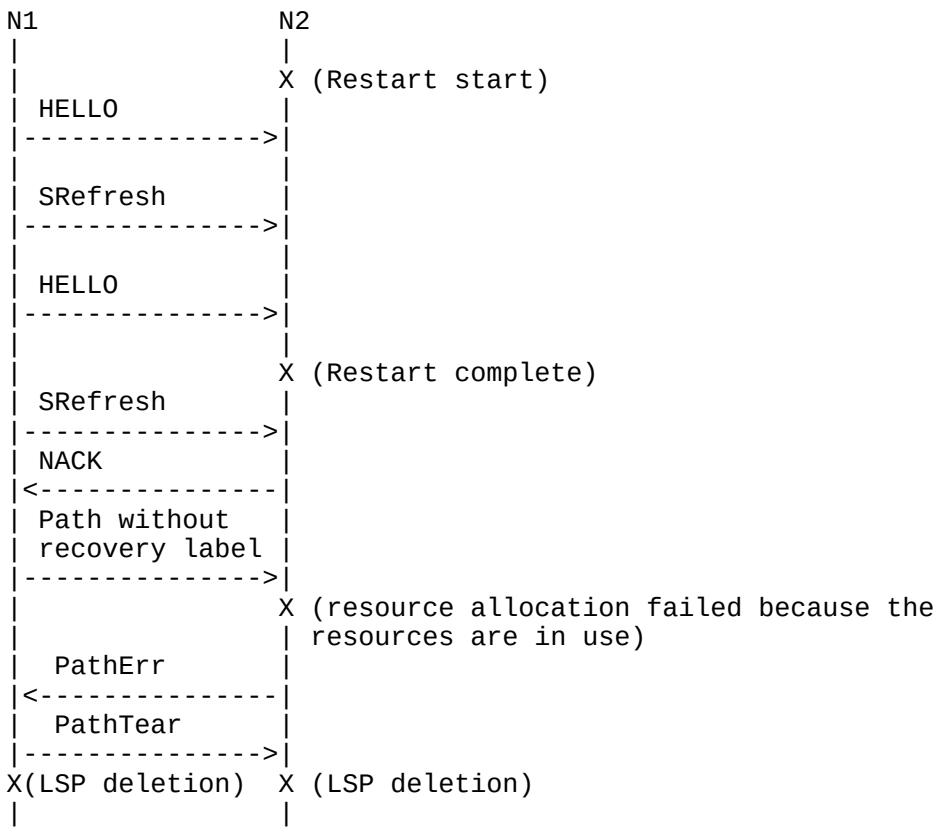

Figure 2: Message Flow for Accidental LSP Deletion

The sequence diagram above depicts one scenario where the LSP may get deleted.

In this sequence, N1 does not detect Hello failure and continues sending SRefreshes, which may get NACK'ed by N2 once restart completes because there is no Path state corresponding to the SRefresh message. This NACK causes a Path refresh message to be generated, but there is no Recovery_Label because N1 does not yet detect that N2 has restarted, as Hello exchanges have not yet started. The Path message is treated as "new" and fails to allocate the resources because they are still in use. This causes a PathErr message to be generated, which may lead to the teardown of the LSP.

To resolve the aforementioned problem, the following procedures, which are implicit in [RFC3473] and [RFC5063], should be followed. These procedures work together with the recovery procedures documented in [RFC3473]. Here, it is assumed that the restarting

node and the neighboring node(s) support the Hello extension as documented in [RFC3209] as well as the recovery procedures documented in [RFC3473].

After a node restarts its control plane, it should ignore and silently drop all RSVP-TE messages (except Hello messages) it receives from any neighbor to which no HELLO session has been established.

The restarting node should follow [RFC3209] to establish Hello sessions with its neighbors, after its control plane becomes operational.

The restarting node resumes processing of RSVP-TE messages sent from each neighbor to which the Hello session has been established.

7. Security Considerations

This document clarifies the procedures defined in [RFC3473] and [RFC5063] to be performed on RSVP agents that neighbor one or more restarting RSVP agents. It does not introduce any new procedures and, therefore, does not introduce any new security risks or issues.

In the case of the control plane in general, and the RSVP agent in particular, where one or more nodes carrying one or more LSPs are restarted due to external attacks, the procedures defined in [RFC5063] and described in this document provide the ability for the restarting RSVP agents to recover the RSVP state in each restarting node corresponding to the LSPs, with the least possible perturbation to the rest of the network. These procedures can be considered to provide mechanisms by which the GMPLS network can recover from physical attacks or from attacks on remotely controlled power supplies.

The procedures described are such that only the neighboring RSVP agents should notice the restart of a node, and hence only they 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 and without propagating the error condition in the control or data plane. In other words, the effect of the restart (which might be the result of an attack) does not spread into the network.

Note that concern has been expressed about the vulnerability of a restarting node to false messages received from its neighbors. For example, a restarting node might receive a false Path message with a

Recovery_Label Object from an upstream neighbor, or a false RecoveryPath message from its downstream neighbor. This situation might arise in one of four cases:

- The message is spoofed and does not come from the neighbor at all.

- The message has been modified as it was traveling from the neighbor.

- The neighbor is defective and has generated a message in error.

- The neighbor has been subverted and has a "rogue" RSVP agent.

The first two cases may be handled using standard RSVP authentication and integrity procedures [RFC3209], [RFC3473]. If the operator is particularly worried, the control plane may be operated using IPsec [RFC4301], [RFC4302], [RFC4835], [RFC4306], and [RFC2411].

Protection against defective or rogue RSVP implementations is generally hard-to-impossible. Neighbor-to-neighbor authentication and integrity validation is, by definition, ineffective in these situations. For example, if a neighbor node sends a Resv during normal LSP setup, and if that message carries a Generalized_Label Object carrying an incorrect label value, then the receiving LSR will use the supplied value and the LSP will be set up incorrectly. Alternatively, if a Path message is modified by an upstream LSR to change the destination and explicit route, there is no way for the downstream LSR to detect this, and the LSP may be set up to the wrong destination. Furthermore, the upstream LSR could disguise this fact by modifying the recorded route reported in the Resv message. Thus, these issues are in no way specific to the restart case, do not cause any greater or different problems from the normal case, and do not warrant specific security measures applicable to restart scenarios.

Note that the RSVP Policy_Data Object [RFC2205] provides a scope by which secure end-to-end checks could be applied. However, very little definition of the use of this object has been made to date.

See [MPLS-SEC] for a wider discussion of security in MPLS and GMPLS networks.

8. Acknowledgments

We would like to thank Adrian Farrel, Dimitri Papadimitriou, and Lou Berger for their useful comments.

9. References

9.1. Normative References

[RFC2209] Braden, R. and L. Zhang, "Resource ReSerVation Protocol (RSVP) -- Version 1 Message Processing Rules", RFC 2209, September 1997.

[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.

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

[RFC5063] Satyanarayana, A., Ed., and R. Rahman, Ed., "Extensions to GMPLS Resource Reservation Protocol (RSVP) Graceful Restart", RFC 5063, October 2007.

9.2. Informative References

[MPLS-SEC] Fang, L., "Security Framework for MPLS and GMPLS Networks", Work in Progress, November 2008.

[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.

[RFC2411] Thayer, R., Doraswamy, N., and R. Glenn, "IP Security Document Roadmap", RFC 2411, November 1998.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

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

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

[RFC4835] Manral, V., "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 4835, April 2007.

Authors' Addresses

Dan Li
Huawei Technologies
F3-5-B R&D Center, Huawei Base,
Shenzhen 518129, China

Phone: +86 755 28970230
EMail: danli@huawei.com

Jianhua Gao
Huawei Technologies
F3-5-B R&D Center, Huawei Base,
Shenzhen 518129, China

Phone: +86 755 28972902
EMail: gjhhit@huawei.com

Arun Satyanarayana
Cisco Systems
170 West Tasman Dr
San Jose, CA 95134, USA

Phone: +1 408 853-3206
EMail: asatyana@cisco.com

Snigdho C. Bardalai
Fujitsu Network Communications
2801 Telecom Parkway
Richardson, Texas 75082, USA

Phone: +1 972 479 2951
EMail: snigdho.bardalai@us.fujitsu.com