RSVP Standards Today and the Path Towards a Generic Messenger

Xiaoming Fu[1] and Jukka Manner[2]

[1] Telematics Group, Institute for Informatics, University of Goettingen
[2] Department of Computer Science, University of Helsinki

1 Introduction to the Base Standard and its Extensions

The Resource Reservation Protocol (RSVP) specified in RFC2205[3] has evolved from ST-II (RFC1819) to provide end-to-end QoS signaling services for application data streams. Hosts use RSVP to request a specific quality of service (QoS) reservation from the network for particular application flows. RSVP maintains and refreshes reservation states in routers for a requested QoS application flow. By original design, RSVP fits well into the framework of the Integrated Services (IntServ) of RFC2210 with certain modularity and scalability.

The fundamental concepts of RSVP include soft state management, two-pass signaling message exchanges, receiver-based resource reservation, and separation of QoS signaling from routing. Most of the functionality designed into RSVP has emerged from the original goal to support multicast reservations: the (multicast) receiver-based resource reservation, reservations styles and reservation merging, and soft state to support changes in the multicast routing tree.

RSVP was originally designed to support real-time applications over the Internet. Over the past several years, a tremendous demand for multicast-capable real-time applications, which many people had envisioned to be a killer application that could benefit from network-wide deployment of RSVP, has never materialized. Instead, RSVP-TE (RFC3209), an RSVP extension for traffic engineering, has been widely deployed by a large number of network providers to support label distribution in MPLS networks. GMPLS RSVP-TE (RFC3473) further extends RSVP-TE, by enabling the provisioning of data-paths within networks supporting a variety of switching types including packet and cell switching networks, layer two networks, TDM networks and photonic networks.

Various other extensions have been designed to extend the use of RSVP. RFCs 2379 and 2380 define RSVP over ATM implementation guidelines and requirements to interwork with the ATM UNI 3.x/4.0. RFC2996 introduces a DCLASS Object to carry DSCPs in RSVP message objects. The Null Service Type in RFC2997 allows applications to identify themselves to network policy agents using RSVP, and leaves resource allocations up to the network policies.

RFC2746 allows RSVP to make reservations across all IP-in-IP tunnels, basically, by recursively applying RSVP over the tunnel portion of the path. RFC2207 extends RSVP by using the IPsec SPI in place of the UDP/TCP-like

[3] All RFC documents available from http://www.ietf.org/rfc/rfcNUMBER.txt.

ports. As RFC2205 leaves the policy component open, later RFCs 2749, 2750, and 3181 specify POLICY_DATA objects, handling of RSVP policy events by COPS-aware nodes, and a preemption priority policy. RFC2961 describes mechanisms to reduce processing overhead of refresh messages, eliminate the state synchronization latency incurred when an RSVP message is lost and, refreshing state without the transmission of whole refresh messages. Aggregation of reservations is specified in RFC3175. RSVP diagnostic messages are defined in RFC2745 to collect and report RSVP state information.

2 Analysis of the Current RSVP

A good signaling protocol should be transparent to the applications. RSVP has proven to be a very well designed protocol. However, it has a number of fundamental protocol design issues that requires more careful re-evaluation.

The design of RSVP was originally targeted at multicast applications. The result is that the message processing within nodes is somewhat heavy, mainly due to flow merging. Still, merging rules should not appear in the specification as they are QoS-specific. Also, the QoS objects should be more general.

From the security point of view, RSVP does provide the basic building blocks to protect the messages from forgery and modification in various deployment environments. However, current RSVP security mechanisms do not provide non-repudiation and protection against message deletion; the two-way peer authentication and key management procedures are still missing.

Domains not supporting RSVP are traversed transparently by default. Unfortunately, like other IP options, RSVP messages implemented by way of IP alert option may result in themselves being dropped by some routers. Also, RSVP does not support message fragmentation and reassembly at protocol level. If the size of an RSVP message is larger than the link MTU, e.g., from carried large security-related objects, the message may be fragmented by IP. However, RSVP routers simply cannot detect and process message fragments.

The state machine of RSVP is complex, mainly due to the focus on multicast, which complicates message processing and per-session state maintenance. Moreover, the order and existence of objects can vary, which increases the complexity in message parsing and internal message and state representation. RFC2961 tries to lower the bandwidth consumption of RSVP, and provide better reliability. However, a lot of effort has to be spent on per-session timer maintenance, message retransmission (e.g., avoid message bursts), and message sequencing.

Although RSVP uses soft state mechanism and is independent of underlying routing protocols, a mobile node's movement may not properly trigger a reservation refresh for the new path and a mobile node may be left without a downlink reservation up to the lifetime of the refresh timer. This can happen because only Path messages can repair the routing path of the reservation messages, and a receiving node must wait for a Path message from the sender. Furthermore, RSVP does not work properly when the mobile node's IP address changes, since the filters will not identify the flow that had a reservation.

Moreover, to be useful, RSVP needs support from both communicating end hosts, and the underlying network. In many deployment scenarios, it would be most beneficial, if a reservation could be applied to only the local domain. Ideas from different points of view have been discussed in the IETF, e.g., the RSVP Proxy and the Localized RSVP.

It is expected that the development of future signaling protocols should learn from the lessons of existing ones. A thorough evaluation of Internet QoS signaling protocols can be found in *Analysis of Existing Quality of Service Signaling Protocols*, Internet Draft (work in progress), December 2004.

3 Path Towards a Generic Messenger

As observed in RFCs 3234 and 3724, the rise of various middleboxes including QoS boxes and stateful packet filter firewalls, has put in question the general applicability of the end-to-end principle. Unlike SIP (RFC3261), which is designed under an architecture with registrars and proxies for end-to-end session level signaling between both communication endpoints, middleboxes require session-related state installation and maintenance along the communication path. Due to the variety of middleboxes, there has been a strong need for a generic signaling service for delivery of control information into these boxes. A big question is, can RSVP be changed to handle present and tomorrow's deployment scenarios and requirements? In our view, the new signaling requirements, with network security requirements, and with MTU problems, will prevent a direct re-use of the existing RSVP. A new, generic messenger is needed: the protocol must be catered primarily for unicast applications, must be able to handle reliable and secure messaging, message packing, the MTU problem, small triggered message volumes, and changes in IP addresses and effective re-routing during the lifetime of a reservation. Moreover, moving a signaling protocol for QoS resource reservations into a generic messenger can provide much adoption. Towards this, several design choices have been identified in the IETF NSIS Working Group:

- The messenger is separated from signaling applications, dedicates to message delivery, and is responsible for only transporting signaling data into relevant middleboxes.
- It decouples the discovery of next signaling hop and signaling message transport, which allows rich security protocol for transport while keeping the discovery component rather simple.
- It reuses reliable transport protocols, such as TCP and SCTP, which also support fragmentation and other features, as well as unreliable ones, for message transport depending on the application requirements and availability.

Given the amount of legal RSVP implementations, the transition path from RSVP to such a generic messenger may be gradual. For example, if fragmentation or strong security for signaling messages is not required, unreliable transport and coupling next hop discovery within signaling message transport can still be used as in RSVP, especially in intra-domain cases.