Sonia Fahmy and Raj Jain. “Resource ReSerVation Protocol (RSVP).”
Handbook of Emerging Communications Technologies: The Next Decade.
Ed. Saba Zamir
Boca Raton: CRC Press LLC, 2000

1 Resource ReSerVation Protocol (RSVP)

Sonia Fahmy and Raj Jain

CONTENTS

1.1 INTRODUCTION

In 1993, the Internet Engineering Task Force (IETF) chartered the Resource ReSerVation Protocol (RSVP) working group to specify the signaling protocol to set up resource reservations for the new (real-time) services to be provided in the Internet. The RSVP is a state-establishment protocol. RSVP will enable the Internet to support real-time and multimedia applications, such as teleconferencing and videoconferencing applications. These applications require reservations to be made in the Internet routers, and RSVP is the protocol to set up these reservations.

The Internet protocol (IP) currently supports a best effort service, where no delay or loss guarantees are provided. This service is adequate for nontime-critical applications, or time-critical applications under light load conditions. Under highly overloaded conditions, however, buffer overflows and queuing delays cause the real-time communication quality to quickly degrade.

To support real-time applications, a new service model was designed for the Internet. In this model, both real-time and nonreal-time applications share the same infrastructure, thus benefiting from statistical multiplexing gains. Applications specify their traffic characteristics and their quality of service requirements. Admission control is employed to determine whether those requirements can be met, and reservations are made. Packet scheduling services different applications with different priorities to ensure that the quality of service requirements are met.

RSVP can also transport other messages in addition to reservation messages, such as policy control and traffic control messages. The key features of RSVP include flexibility, robustness, scalability through receiver control of reservations, sharing of reservations, and use of IP multicast for data distribution.

This chapter gives an overview of RSVP and its use to support integrated services in the Internet. We first discuss RSVP goals, features, messages, and interfaces. Then we describe RSVP management and security. We also discuss how RSVP can be used with the integrated services, and how it can be used on specific link layers, such as ATM and IEEE 802 networks. Finally, we discuss the interoperability of RSVP with non RSVP routers and examine a number of issues currently under investigation, such as aggregation and differentiated services, policy control, label switching, routing, and diagnostics.

1.2 WHAT IS RSVP?

RSVP is the means by which applications communicate their requirements to the network in an efficient and robust manner. RSVP does not provide any network service; it simply communicates any end-system requirement to the network. Thus

RSVP can be viewed as a 'switch state establishment protocol,' rather than just a resource reservation protocol.

RSVP is developed to support traffic requiring a guaranteed quality of service over both IP unicast and multicast. Figure 1.1 shows the IP multicast model. The Internet Group Management Protocol (IGMP) is used to handle group membership requests. IGMP is the protocol through which hosts indicate to their local routers their interest in joining a group. The Internet multicast routing protocols are employed in Internet routers to set up the multicast trees used for forwarding the data packets to the appropriate group members.

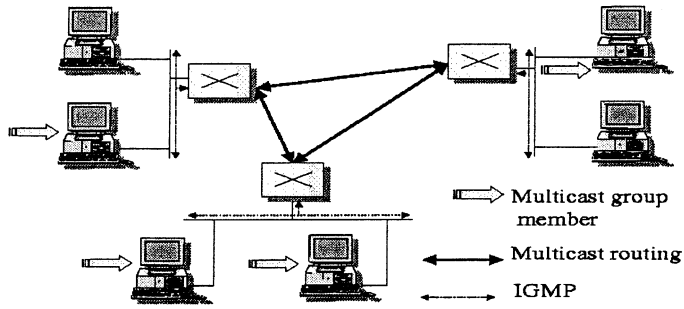

Figure 1.1 Multicasting in the Internet

Through RSVP, applications that receive real-time traffic inform networks of their needs, while applications that send real-time traffic inform the receivers about their traffic characteristics. RSVP is the signaling protocol that installs and maintains reservation state information at each router along the path of a stream. RSVP transfers reservation data as opaque data; it can also transport policy control and traffic control messages. RSVP operates on top of IP (both version 4 and version 6), and it is concerned only with the quality of service (QoS) of the packets forwarded according to routing.

The term *session* will be used throughout the chapter, since RSVP operates on a per-session basis. In the context of RSVP in the Internet, an RSVP session is a simplex data stream from a sending application to a set of receiving applications, usually defined by the triple (DestAddress, ProtocolId [, DstPort]). DestAddress is the IP destination address of the data packets and may be a unicast or multicast address. ProtocolId is the IP protocol identifier. The optional DstPort parameter could be defined by a UDP/TCP destination port field, by an equivalent field in another transport protocol, or by some application-specific information.

1.2.1 Components of an RSVP-Capable Router

Each router in the new Internet model must contain several components, as illustrated in Figure 1.2. These components interact through explicit interfaces to improve the

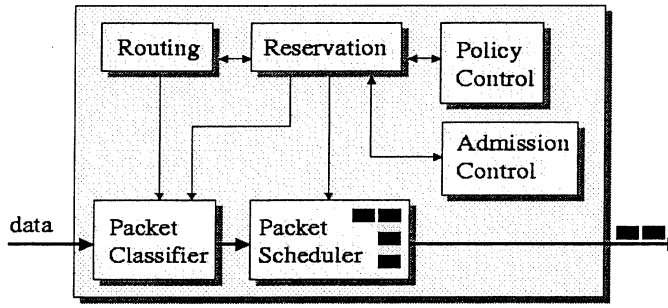

Figure 1.2 RSVP-capable routers

modularity and independence of the scheme. In addition to the routing mechanism and the flow QoS specification scheme, the router must contain an admission control process, to determine if sufficient resources are available to make the reservation, and a policy control process, to determine if the user has permission to make the reservation. If the RSVP process gets an acceptance indication from both the admission control and policy control processes, it sends the appropriate parameter values to the packet classifier and packet scheduler. The packet classifier determines the QoS class of packets according to the requirements, and the packet scheduler manages various queues to guarantee the required QoS. To guarantee the bandwidth and delay characteristics reserved by RSVP, a fair packet-scheduling scheme, such as weighted fair queuing, can be employed. Fair scheduling isolates data streams and gives each stream a percentage of the bandwidth on a link. This percentage can be varied by applying weights derived from RSVP's reservations. The admission control process, packet classifier, and packet scheduler are collectively called traffic control.

1.2.2 RSVP Design Goals

We explain the RSVP design goals by giving the problem and RSVP solution associated with each goal[18] in Table 1.1.

1.3 RSVP FEATURES

Figure1.3 shows the router model employed by RSVP. Data arrives on the incoming interfaces from the previous hops and is routed to one of the next hops through the outgoing interfaces. An RSVP sender uses the PATH message to communicate with receivers informing them of flow characteristics. RSVP provides receiver-initiated reservation of resources, using different reservation styles to fit a variety of applications. RSVP receivers periodically alert networks to their interest in a data flow, using RESV messages that contain the source IP address of the requester and the destination IP address, usually coupled with flow details. The network allocates the needed bandwidth and defines priorities. RSVP decouples the packet classification and scheduling from the reservation operation, transporting the messages from the

TABLE 1.1
RSVP Goals

Goal	Problem	RSVP Solution
Accommodate heterogeneous receivers	Receivers in the same (multicast) session, and paths to these receivers, can have different capacities and require different QoS	RSVP allows receivers to make different reservations
Adapt to changing multicast group membership	New members can join a multicast group at any time, and existing members can leave at any time	RSVP gracefully handles group member changes to scale to large groups
Adapt to route changes	Routing protocols adapt to changes in topology and load by establishing new routes	RSVP handles route changes by automatically reestablishing resource reservations along new paths if adequate resources are available
Control protocol overhead	Refreshing reservations over the multicast routing tree can create a high overhead	RSVP overhead does not grow proportional to the group size, and parameters are used to control the overhead
Use network resources efficiently	Sometimes resources are wasted if each sender in the same session makes a separate reservation along its multicast tree	RSVP allows users to specify their needs so that the aggregate resources reserved for the group reflect the resources actually needed. Receivers can specify the specific senders that can use the reserved resources
Accommodate heterogeneous underlying technologies	The protocol design should interoperate and coordinate with different routing algorithms and other components	RSVP design is relatively independent of the flow specification, routing, admission control and packet scheduling functions

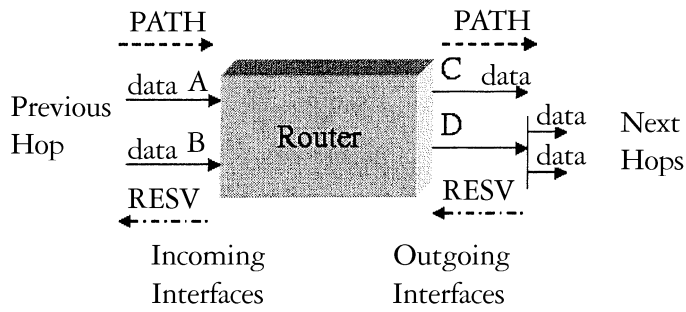

Figure 1.3 RSVP router model

source and destination as opaque data. Periodic renewal of state allows networks to be self correcting despite routing changes and loss of service. This enables routers to understand their current topologies and interfaces, as well as the amount of network bandwidth currently supported. These features are discussed below.

1.3.1 Receiver-initiated Setup

The RSVP protocol provides receiver-initiated setup of resource reservations for both unicast and multicast data flows. For multicast data flows, reservation requests merge when they progress up the multicast tree. The reservation for a single receiver travels only until it reaches a reserved branch of the tree. Receiver-initiated reservation works better than sender-initiated reservation because it is the receivers that know their possibly different (in this case we call them heterogeneous receivers) and possibly changing requirements and limitations. Hence, receiver-controlled setup is more scalable. RSVP reserves resources in only one direction, so the sender is logically separate from the receiver.

1.3.2 Packet Classification and Scheduling

RSVP does not determine which packets can use the resources; it specifies only the amount of resources reserved for each flow. RSVP interacts with the packet classifier and the packet scheduler to determine the classes (and perhaps routes) and achieve the required QoS. Thus, RSVP transports reservation data as opaque data. An RSVP reservation request consists of a FlowSpec, specifying the desired QoS, as well as a FilterSpec, defining the flow to receive the desired QoS. The FlowSpec is used to set parameters in the packet scheduler, while the FilterSpec is used in the packet classifier.

The FlowSpec in a reservation request will generally include a service class and two sets of numeric parameters: (1) an RSpec (R for reserve) that defines the desired QoS, and (2) a TSpec (T for traffic) that describes the data flow. The basic FilterSpec format defined in the present RSVP specification has a very restricted form: sender IP address and, optionally, the UDP/TCP port number SrcPort.

1.3.3 Soft State

The soft state feature of RSVP increases the robustness of the protocol. RSVP adapts to changing group memberships and changing routes throughout the application lifetime in a manner that is transparent to applications. This is accomplished through *soft state*, which means that state maintained at network switches expires after a certain period of time (called *cleanup timeout interval*), unless it gets reinstated. Reinstatement is performed through periodic "refresh" messages sent by the end users (both senders and receivers) every "refresh timeout" period, to automatically maintain state in the switches along the reserved paths. In the absence of such refresh messages, reservation state in the routers times out. This is also one way of releasing resources in reservations that are shared by multiple receivers, as explained next.

1.3.4 RSVP Reservation Styles

The RSVP protocol supports several reservation styles to fit a variety of application requirements. A reservation style allows the applications to specify how reservations for the same session are aggregated at the intermediate switches. The basic distinction among different reservation styles is whether a separate reservation should be established for each upstream sender in the same session, or if a single reservation can be shared among the packets of selected senders. The selection of senders in such a case can be done through an explicit list of senders or through a wildcard that selects all the senders in the session (see Table 1.2). Reservation styles supported by RSVP include wildcard filters, fixed filters, and shared explicit filters.

TABLE 1.2
Reservation Styles

	Reservations	
Sender selection	Distinct	Shared
Explicit	Fixed filter	Shared explicit
Wildcard	None	Wildcard filter

The fixed filter creates a distinct reservation per specified sender (without installing separate reservations for each receiver to the same sender). The total reservation on a link for a given session is the sum of the flow specifications for all requested senders. An example of applications that can use the fixed filter is video conferencing applications for which enough resources must be reserved for the number of video streams a receiver wishes to watch simultaneously.

The wildcard filter shares the same reservation among all upstream senders, reserving resources to satisfy the largest resource request (regardless of the number of senders). A wildcard reservation automatically extends to new senders in the session as they appear. An example of applications that can use the wildcard filter is audio conferencing applications where all the senders can share the same set of reserved resources, since multiple senders are unlikely to transmit at the same time.

The last type of reservation style is the shared explicit filter, where a single reservation is created but can be shared only by selected upstream senders. An example of applications that use the shared explicit filter is audio conferencing applications where the receivers want to block traffic from specific senders. Note that the shared explicit filter incurs more overhead than the wildcard filter. In addition, reservations of different styles cannot be merged.

Example:
Figure 1.4 illustrates a router with two incoming interfaces, A and B, and two outgoing interfaces, C and D. There are three upstream senders and three downstream receivers. Outgoing interface D is connected to a broadcast LAN. Thus, R2 and R3 are reached

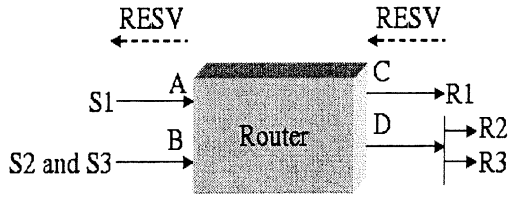

Figure 1.4 Router configuration

via different next-hop routers (not shown). Data packets from each sender shown in Figure 1.4 are routed to both outgoing interfaces.

The FlowSpec is given as a multiple of some base resource quantity R. In Tables 1.3–1.5, the Receives column shows the RSVP reservation requests received over outgoing interfaces C and D (for interface D, the requests received from R2 and R3 are separated by the word *and*). The Reserves column shows the resulting reservation state for each interface. The Sends column shows the reservation requests that are sent upstream to previous hops A and B.

Table 1.3 shows an example of merging wildcard filter style reservations. Merging is required twice in this example. First, each of the two next hops on interface D requests a reservation, and these two requests must be merged into the FlowSpec 2R used to make the reservation on interface D. Second, the reservations on the interfaces C and D must be merged in order to forward the reservation requests upstream. The larger FlowSpec 3R is forwarded upstream to each previous hop.

TABLE 1.3
Wildcard Filter Example

Receives		Reserves		Sends	
Interface	**(Sender, Rate)**	**Interface**	**(Sender, Rate)**	**Interface**	**(Sender, Rate)**
C	(*,3R)	C	(*,3R)	A	(*,3R)
D	(*,2R) and (*,R)	D	(*,2R)	B	(*,3R)

Table 1.4 shows fixed filter style reservations. For each outgoing interface, there is a separate reservation for each source that has been requested, but this reservation will be shared among all the receivers that made the request. The flow descriptors for senders S2 and S3, received through outgoing interfaces C and D, are placed into the request forwarded to previous hop B. The three different flow descriptors (from R1, R2, and R3) specifying sender S1 are merged into the single request (S1,4R) sent to previous hop A.

Table 1.5 shows an example of shared explicit style reservations. When such reservations are merged, the resulting FilterSpec is the union of the original FilterSpecs, and the resulting FlowSpec is the largest FlowSpec.

TABLE 1.4
Fixed Filter Example

Receives		Reserves		Sends	
Interface	**(Sender, Rate)**	**Interface**	**(Sender, Rate)**	**Interface**	**(Sender, Rate)**
C	(S1,4R),(S2,5R)	C	(S1,4R),(S2,5R)	A	(S1,4R)
D	(S1,3R),(S3,R) and (S1,R)	D	(S1,3R),(S3,R)	B	(S2,5R),(S3,R)

TABLE 1.5
Shared Explicit Example

Receives		Reserves		Sends	
Interface	**(Sender, Rate)**	**Interface**	**(Sender, Rate)**	**Interface**	**(Sender, Rate)**
C	((S1,S2),R)	C	((S1,S2),R)	A	(S1,4R)
D	((S1,S3),4R) and (S2,2R)	D	((S1,S2,S3),4R)	B	((S2,S3),4R)

1.4 RSVP MESSAGES

This section explains RSVP message formats and message processing, routing, and merging. As previously mentioned, RSVP receivers use the reserve (RESV) message to periodically advertise to the network their interest in a flow, specifying the flow and filter specifications. RSVP senders, on the other hand, use the PATH message to indicate that they are senders and give information such as the previous hop IP address, the multicast group address, templates for identifying traffic from that sender, and sender traffic specifications. The message is sent to all receivers in the multicast tree using the forwarding table maintained by the multicast routing protocol. The RESV message is forwarded back to the sources by reversing the paths of PATH messages (using the previous hop IP address stored from PATH messages). RSVP supports one pass with advertising (OPWA): the PATH messages contain a field (AdSpec) to gather information that may be used to predict the end-to-end QoS. The results are delivered by RSVP to the receivers which construct, or dynamically adjust, an appropriate reservation request.

1.4.1 Message Formats and Message Processing

An RSVP message consists of a common header, followed by a body consisting of a variable number of variable length, typed *objects*. The main RSVP messages are PATH and RESV messages. In addition, there is a reservation confirmation message (ResvConf), two types of error messages (PathErr and ResvErr), and two types of teardown messages (PathTear and ResvTear). These messages are shown in Figure 1.5 and briefly explained in Table 1.6. This section examines them in more detail.

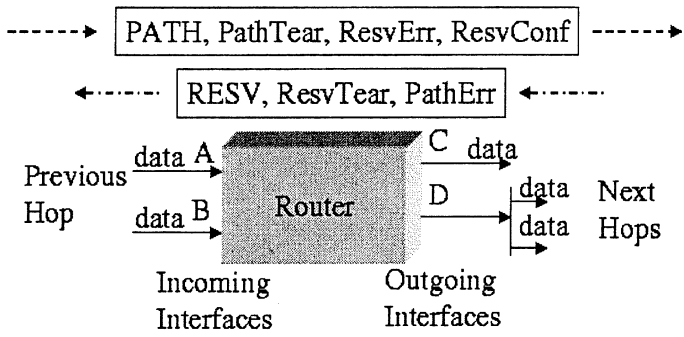

Figure 1.5 RSVP messages

TABLE 1.6
RSVP Messages

Message	Meaning	Purpose
PATH	Path establishment	Used by senders to specify their traffic characteristics
RESV	Reservation request	Used by receivers to advertise to the network their interest in a flow
ResvConf	Reservation confirmation	Indicates to the receiver successful installation of a reservation at an upstream node
PathErr	Path error	Indicates to the sender an error in the path message
ResvErr	Reservation error	Indicates to the receivers that a reservation request has failed or an active reservation has been preempted
PathTear	Path teardown	Deletes path state and dependent reservation state
ResvTear	Reservation teardown	Deletes reservation state

1.4.1.1 PATH Message

The PATH message contains the following:

- *Session identifier* and *timeout values* to control refresh frequencies.
- *Previous hop address.* This is maintained at every node to route RESV messages in the reverse path taken by PATH messages.
- *Sender Template.* The sender template describes the data packets that the sender will originate. This is given as a FilterSpec to distinguish the sender packets from others in the same session on the same link. The sender template may specify only the sender IP address and optionally the sender port, assuming the same protocol identifier specified for the session.

- *Sender TSpec.* This defines the traffic characteristics of the data flow that the sender will generate. The TSpec is used by traffic control to prevent over-reservation and unnecessary admission control failures.
- *AdSpec.* The AdSpec (optional) includes parameters describing the properties of the data path (including the availability of specific QoS control services) and parameters required by specific QoS control services to operate correctly. An AdSpec received in a PATH message is passed to the local traffic control, which returns an updated AdSpec. The updated version is then forwarded in PATH messages sent downstream.

The PATH message may also contain integrity objects and policy data objects.

Each RSVP-capable node along the path captures the PATH message and processes it to create path state for the sender defined by the sender template and session objects (the next section defines the state information maintained at each node). The sender TSpec and objects such as policy data and AdSpec are also stored in the path state. The RSVP process forwards PATH messages and replicates them as required by multicast sessions (modifying the previous hop and AdSpec fields). This operation uses routing information RSVP obtains from the appropriate routing process (see Figure 1.6). Periodically, the RSVP process scans the path state to create new PATH messages to forward towards the receivers.

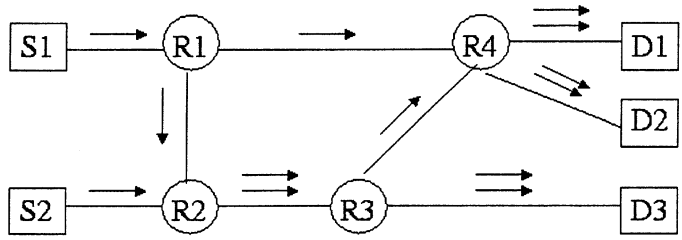

Figure 1.6 RSVP PATH messages

1.4.1.2 RESV Message

The RESV message contains the following:

- *Session identifier* and *timeout values* to control refresh frequencies.
- *Hop address* which contains the address of the interface through which the message was sent and the interface on which reservation is required.
- *Confirmation required or not* and the *receiver address* to which to send the confirmation.
- *Reservation style, FilterSpec,* and *FlowSpec.* In case of wildcard filters, only a FlowSpec needs to be given. For a fixed filter, the FilterSpec and FlowSpec for each sender should be given. For shared explicit reservations, one FlowSpec and a set of FilterSpecs must be given.
- *Set of senders* to which to forward the RESV message (scope).

The RESV message may also contain integrity and policy data objects.

The RSVP process at each intermediate node first passes the reservation request to admission control and policy control. If either test fails, the reservation is rejected and the RSVP process returns an error message to the appropriate receivers. If both succeed, the node sets the packet classifier to select the data packets defined by the FilterSpec. RSVP then interacts with the appropriate link layer to obtain the desired QoS defined by the FlowSpec. The action to control QoS occurs at the upstream end of the link, although the RSVP reservation request originates from receivers downstream. Once the reservation is made at the node, the reservation request is propagated upstream towards the appropriate senders. The FlowSpec in the RESV message may have been modified by the traffic control mechanism, and reservations from different downstream branches of the multicast tree for the same sender (or set of senders) are merged as reservations travel upstream. The RESV messages will be propagated immediately to the next node *only if there will be a net change after merging.* Otherwise, the messages are refreshed periodically. RESV messages must finally be delivered to the sender hosts themselves, so the hosts can set up appropriate traffic control parameters for the first hop.

1.4.1.3 Confirmation Messages

When a receiver originates a reservation request, it can also request a confirmation message by including in the RESV message a confirmation request object containing its IP address. The reservation confirmation (ResvConf) message is sent by an upstream node to indicate that the request was probably, but not necessarily due to merging, installed in the network. If the reservation request is merged with a larger one at an intermediate node, the intermediate node sends the confirmation message, because the reservation request is not propagated upstream in this case. This reservation might then fail if the merged request fails.

1.4.1.4 Error Messages

There are two RSVP error messages: ResvErr and PathErr.

PathErr messages are sent upstream to the sender that created the error, and they do not change path state in nodes through which they pass.

As for ResvErr messages, there are many ways for a syntactically valid reservation request to fail at a node along the path. Nodes may also preempt an established reservation. Because the failed request may be a combination of a number of requests, a ResvErr message must be sent to all of the appropriate receivers. In addition, merging heterogeneous requests creates a potential problem (called the *killer reservation* problem), in which a request could deny service to another.

The problem is simple when a reservation R1 is already in place. If another receiver makes a larger reservation R2, the result of merging R1 and R2 may be rejected by admission control in an upstream node. The service to R1 will not be denied, however, because when admission control fails for a reservation request existing reservations are left in place.

When the receiver making a reservation R2 is persistent even though admission control is failing for R2 at a certain node, another receiver should be able to establish a smaller reservation R1 that would succeed if not merged with R2. To enable this, a ResvErr message establishes an additional state, called *blockade state*, in each node through which it passes. Blockade state in a node modifies the merging procedure to omit the offending FlowSpec from the merge, allowing a smaller request to be established. A reservation request that fails admission control creates blockade state but is left in place in nodes downstream of the failure point.

1.4.1.5 Teardown Messages

RSVP teardown messages remove path or reservation state immediately. There are two types of RSVP teardown messages: PathTear and ResvTear. A PathTear message travels towards all receivers downstream from its point of initiation and deletes path state, as well as all dependent reservation state. A ResvTear message travels towards all senders upstream from its point of initiation and deletes reservation state. It is possible to tear down any subset of the established state; for path state, the granularity for teardown is a single sender, while for reservation state the granularity is an individual FilterSpec.

Teardown requests can be initiated by end systems or by routers as a result of state timeout or service preemption. The state deletion is immediately propagated to the next node only if there will be a net change after merging (same as with the reservation state). Hence, a ResvTear message will prune the reservation state back as far as possible.

1.4.2 State Data

The following data structures are maintained by the RSVP protocol at each node[7]:

- *Path state block* stores state from the PATH message for each session and sender template.
- *Reservation state block* holds a reservation request that arrived in a particular RESV message, corresponding to the triple: session, next hop, and FilterSpec list.
- *Traffic control state block* holds the reservation specification that has been handed to traffic control for a specific outgoing interface. In general, this information is derived from the reservation state block for the same outgoing interface. The traffic control state block defines a single reservation for the triple: session, outgoing interface, and FilterSpec list.
- *Blockade state block* contains an element of blockade state (see Section 1.4.1). Depending upon the reservation style in use, this information may be per session and sender template pair, or per session and previous hop pair.

1.4.3 Message Routing

PATH messages are sent with the same source and destination addresses as the data so they will be routed correctly through non-RSVP clouds. On the other hand, RESV messages are sent hop-by-hop; each RSVP-capable node forwards a RESV message to the unicast address of a previous RSVP hop.

RSVP acquires routing entries by sending route queries to the routing protocol. As previously mentioned, RSVP needs to send a different copy of the PATH message on each outgoing interface. Hence, RSVP simulates its own multicast forwarding so it can specify a single interface to send a multicast packet without any loop back.

RSVP may ask routing to notify it when a particular route changes. Route change notification enables RSVP to quickly adapt its reservations to changes in the route between a source and destination. For multicast destinations, a route change consists of any local change in the multicast tree for a source-group pair (including prunes and grafts), as well as routing changes due to failed or recovered links. RSVP adapts to route changes by resending PATH or RESV messages where needed. If routing cannot support route change notification, RSVP must poll routing for route entries in order to adapt to route changes.

1.4.4 Message Merging

RSVP merges RESV messages in the network for the *same* reservation style (see Figure 1.7). The RESV messages are forwarded only until the point where they

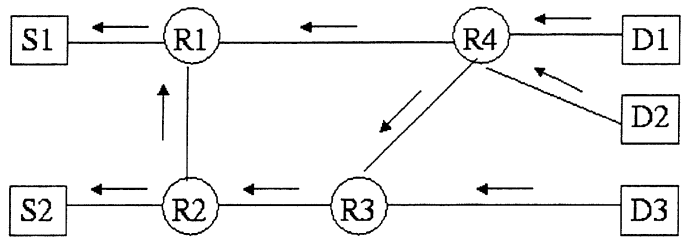

Figure 1.7 RSVP RESV messages

merge with a larger request. A RESV message forwarded to a previous hop carries a FlowSpec that is the largest of the FlowSpecs requested by the next hops to which the data flow will be sent. Since FlowSpecs are opaque to RSVP, the rules for comparing FlowSpecs are defined in the integrated services specifications. RSVP must call service-specific routines to perform FlowSpec comparison and merging.

FlowSpecs are generally multidimensional vectors, and each of their TSpec and RSpec components may itself be multidimensional. It may not be possible to strictly order two FlowSpecs. For example, if one request specifies a lower bandwidth than the other, but the other specifies a looser delay bound, neither is larger than the other. In this case, instead of taking the larger, the service-specific merging routines

must be able to return a third FlowSpec that is at least as large as each of them. This is the least upper bound (LUB) of the flows. In some cases, a FlowSpec at least as small is needed, and this is the greatest lower bound (GLB).

1.5 RSVP INTERFACES

RSVP interacts with other components in the router through well-defined interfaces. The RSVP/policy control interface will be discussed in Section 1.11 because it is still being specified. The remaining interfaces are briefly described below.

- *Application/RSVP Interface.* The application/RSVP interface should allow the application to register a session, define a sender, reserve resources, and release resources. It should also inform the application of the receipt of all types of RSVP messages.
- *RSVP/Traffic Control Interface.* This interface enables establishing a reservation, modifying a reservation, deleting a FlowSpec, deleting or adding a FilterSpec, updating the AdSpec, and preempting a reservation.
- *RSVP/Routing Interface.* This interface supports route querying, route change notification, and interface list discovery.
- *RSVP/Packet I/O Interface.* RSVP must be able to use the promiscuous receive mode for RSVP messages, force a packet to be sent to a specific interface, specify the source address and time-to-live in PATH messages, and send messages with the router alert option.
- *Service-Dependent Manipulations.* RSVP must be able to compare FlowSpecs, compute their LUBs and GLBs, and compare and sum TSpecs (as explained in the last section of this chapter).

1.6 RSVP MANAGEMENT

The simple network management protocol (SNMP) version 2 defines an architecture for network management for the Internet protocol suite and a framework for accessing, describing, and naming objects to be managed. Managed objects are accessed via a virtual information store, which is called the management information base (MIB). Each object type is named by an administratively assigned name, called the object identifier.

The RSVP MIB is composed of the following sections defined in RFC 2206[2]:

- General objects
- Session statistics table
- Session sender table
- Reservation requests received table
- Reservation requests forwarded table
- RSVP interface attributes table
- RSVP neighbor table

1.7 RSVP SECURITY

In 1995, an architecture for providing security in IP versions 4 and 6 (IPSEC) was developed. Two methods for IP security were defined. The first method introduces an authentication header (AH) in IP packets after the IP header, but before the information being authenticated. The authentication header can provide authentication, integrity, and possibly non-repudiation, depending on the cryptographic algorithm employed. The second method is the IP encapsulating security payload (ESP). ESP can provide confidentiality and integrity, and possibly authentication to IP packets.

RSVP as specified in Braden et al.[8] can support the two above mentioned IP security protocols, but only on a per-address, per-protocol basis, not on a per-flow basis. This is because RSVP relies on transport protocol port numbers (e.g., TCP or UDP ports). For flows without such port numbers, such as IPSEC packet flows where such information is encrypted, flow definition is solely dependent on the IP address and protocol.

In Berger and O'Malley,[3] RSVP is extended to permit per-flow use of the AH and ESP techniques. This is accomplished through using the IPSEC security parameter index (SPI) instead of the transport protocol port numbers. This, however, necessitates that the FilterSpec object contain the SPI. The session object will also require the addition of a virtual destination port to be able to demultiplex sessions beyond the IP destination address. Therefore, the processing of the RESV and PATH messages is modified. One limitation of this method, however, is that when the wildcard filter is used, all flows to the same IP destination address and with the same IP protocol identifier will share the same reservation.

Hop-by-hop integrity and authentication of RSVP messages and sessions can be provided through an integrity object. This is especially important in order to protect the integrity of the admission control mechanism against corruption and spoofing. A scheme is proposed in Baker[1] to transmit the result of applying a cryptographic algorithm to a one-way function or digest of the message together with a secret authentication key.

1.8 USE OF RSVP WITH INTEGRATED SERVICES

We first give an overview of the integrated services model then describe how RSVP can be used to set up reservations for integrated services.

1.8.1 Integrated Services

The integrated services model was based on the premise that applications can be either inelastic (real-time) which requires end-to-end delay bounds, or elastic, which can wait for data to arrive. Real-time applications can be further subdivided into those that are intolerant to delay and those that are more tolerant, called delay adaptive.[6] These three application types were directly mapped onto three service categories to be provided to IP traffic: the guaranteed service for delay-intolerant applications, the controlled load service for delay-adaptive applications, and the currently available best-effort service for elastic applications. The guaranteed service

gives firm bounds on the throughput and delay, while the controlled load service tries to approximate the performance of an unloaded packet network. In this section, we will provide a brief overview of these two services. Each service is specified by a TSpec and an RSpec, as previously discussed.

1.8.1.1 Guaranteed Quality of Service

The guaranteed service gives firm end-to-end delay bounds as well as bandwidth guarantees. If the traffic of the flow obeys the TSpec, the packets are guaranteed to be delivered within the requested delay bound. The service does not give any guarantees on the delay variation (jitter). The TSpec of the flow is given in the form of a token bucket (bucket rate and bucket depth), a peak rate, a minimum policed unit, and a maximum packet size. The RSpec is described using a rate and a slack term.

Given the token bucket parameters and the data rate given to the flow, it is possible to compute a bound on the maximum queuing delay (thus the maximum delay) experienced by packets. This is because the network elements are required to approximate the fluid model of service. The network element must also export two error-characterization terms which represent how the network element implementation deviates from the fluid model.

At the edge of the network, arriving traffic is compared against the TSpec and policed. In addition, traffic is reshaped at heterogeneous branch points (when TSpec for all branches in a multicast tree is not the same) and at merge points. Reshaping means that the traffic is reconstructed to conform to the TSpec. Reshaping needs to be done only if the TSpec on the outgoing link is less than the TSpec reserved on the immediate upstream link.[13]

1.8.1.2 Controlled Load Service

The controlled load service approximates the behavior of best-effort service with underload conditions. It uses admission control to ensure that adequate resources are available to provide the requested level of quality with overload conditions. Applications can assume that (1) a high percentage of the transmitted packets are successfully delivered to the destinations, and (2) the transit delay experienced by a high percentage of the delivered packets will not highly exceed the minimum transit delay. The controlled load service does not give specific delay or loss guarantees (thus there is no RSpec). Over all timescales significantly larger than the burst time, a controlled load service flow should experience little or no average packet queuing delay and little or no congestion loss.[14]

The controlled load service can borrow bandwidth needed to clear bursts from the network, using an explicit borrowing scheme within the traffic scheduler or an implicit scheme based on statistical multiplexing and measurement-based admission control. Information from measurement of the aggregate traffic flow or specific knowledge of traffic statistics can be used by the admission control algorithm for a multiplexing gain.

As with the guaranteed service, the TSpec for the controlled load service is given by a token bucket, a peak rate, a minimum policed unit, and a maximum packet size. Over all time periods *T*, the length of the burst should never exceed $rT+b$, where *r* is the token bucket rate and *b* is the bucket depth. Nonconformant controlled load traffic is forwarded on a best-effort basis only under overload conditions.

1.8.2 Using RSVP to Set Up Reservations for Integrated Services

As previously mentioned, the RSVP specification does not define the internal format of the RSVP objects related to invoking QoS control services. Interfaces to the QoS control services are also defined in a general format. RFC 2210[15] defines the usage and contents of three RSVP protocol objects:

- FlowSpec: includes the QoS service desired by the receivers, a description of the traffic flow to which the resource reservation should apply, and the parameters required to invoke the service
- AdSpec: includes parameters describing the properties of the data path (including the availability of specific QoS services), and parameters required by the QoS services to operate correctly
- Sender TSpec: includes a description of the data traffic generated by the sender

RFC 2210 also specifies a procedure for applications using RSVP facilities to compute the minimum MTU (maximum transmission unit) over a multicast tree and return the result to the senders to avoid fragmentation.

1.9 SUPPORT OF RSVP BY LINK LAYERS

IP can operate over a number of link layers, including ATM, 802 technologies such as Ethernet, and point to point links such as PPP. This section discusses the support of RSVP by ATM and IEEE 802.

1.9.1 ATM Networks

Table 1.7 illustrates the different principles that underlie the design of IP and ATM signaling, especially multicast. Different receivers in an IP multicast group can specify different QoS requirements through RSVP. In addition, receivers are allowed to dynamically change their QoS requirements throughout the connection lifetime (since reservations are periodically refreshed). Group membership also changes throughout connection lifetime.

Since ATM networks provide QoS guarantees, it is natural to map RSVP QoS specifications to ATM QoS specifications and establish the appropriate ATM switched virtual connections (VCs) to support the RSVP requirements. The issue, however, is complicated by the factors that were previously mentioned: RSVP allows heterogeneous receivers and reservation parameter renegotiation, while ATM does

TABLE 1.7
IP/RSVP Multicast versus ATM Multipoint

Category	IP/RSVP	ATM UNI 3.1
Connection type	Connectionless	Virtual connections
Cell ordering	Not guaranteed	Guaranteed
QoS	New services are being added to best effort	CBR, rt-VBR, nrt-VBR, ABR, UBR and more (GFR)
QoS setup time	Separate from route establishment	Concurrent with route establishment
Renegotiation	Allowed	Not allowed
Heterogeneity	Receiver heterogeneity	Uniform QoS to all receivers
Tree Orientation	Receiver-based	Sender-based (UNI 4.0 adds leaf-initiated join)
State	Soft (periodic renewal)	Hard
Directionality	Unidirectional	Unidirectional point-to-multipoint VCs
Tree construction	Different algorithms	Multicast servers or meshes

not. The solution for providing RSVP over ATM must tackle these problems, ensuring scalability. It must also support both UNI 3.1 and UNI 4.0, which support only point-to-multipoint connections.

The problem of supporting RSVP over ATM consists of two main subproblems: first, mapping the IP integrated services to ATM services and, second, using ATM VCs as part of the integrated services Internet. The IP guaranteed service is mapped to constant bit rate (CBR) or real-time variable bit rate (VBR-rt); the controlled load service is mapped to non real-time VBR (VBR-nrt) or available bit rate (ABR) with a minimum cell rate; and the best-effort service is mapped to unspecified bit rate (UBR) or ABR. The second subproblem, managing ATM VCs with QoS as part of the integrated services Internet, entails computing the number of VCs needed and designating the traffic flows that are routed over each VC. Two types of VCs are required: data VCs that handle the actual data traffic, and control VCs which handle the RSVP signaling traffic. The control messages can be carried on the data VCs or on separate VCs.

The best scheme for VC management should use a minimal number of VCs, waste minimal bandwidth due to duplicate packets, and handle heterogeneity and renegotiation in a flexible manner. Proposals that significantly alter RSVP should be avoided. Furthermore, using special servers might introduce additional delays, so cut-through forwarding approaches are preferred. The problem of mapping RSVP to ATM is simplified by the fact that while RSVP reservation requests are generated at the receiver, actual allocation of resources occurs at the sub-net sender. Thus senders establish all QoS VCs, and receivers must be able to accept incoming QoS VCs. The key issues to tackle are data distribution, receiver transitions, end-point identification, and heterogeneity. Several heterogeneity models are defined by Crawley et al.[9] that provide different capabilities to handle the heterogeneity problem. The dynamic QoS problem can be solved by establishing new VCs with minimal signaling, but a timer should guarantee that the rate at which VCs are established is not excessively high.

1.9.2 IEEE 802 Networks

The IETF is currently working on supporting integrated services and RSVP over IEEE 802 networks. In particular, the subnet bandwidth manager (SBM) protocol is being defined as a signaling protocol for RSVP-based admission control over IEEE 802-style networks. SBM defines the operation of RSVP-enabled hosts/routers and link layer devices (switches, bridges) to support reservation of LAN resources for RSVP-enabled data flows. In the absence of any link-layer traffic control or priority queuing mechanisms in the underlying LAN (such as a shared LAN segment), the SBM-based admission control mechanism limits only the total amount of traffic load imposed by RSVP-enabled flows. A protocol entity called designated SBM (DSBM) exists for each managed segment and is responsible for admission control over the resource reservation requests originating from the DSBM clients in that segment. DSBM obtains information such as the limits on fraction of available resources that can be reserved on each managed segment under its control. Then, for each interface attached, a DSBM client determines whether a DSBM exists on the interface. When a DSBM client sends or forwards an RSVP PATH message over an interface attached to a managed segment, it sends the PATH message to the segment DSBM instead of sending it to the RSVP session destination address (as is done in conventional RSVP processing). The DSBM processes the RSVP RESV message based on the bandwidth available and returns a ResvErr message to the requester if the request cannot be granted. If sufficient resources are available and the reservation request is granted, the DSBM forwards the RESV message towards the previous hop, based on its local PATH state for the session. The addition of a DSBM for admission control over managed segments results in some additions to the RSVP message-processing rules at a DSBM client.[16]

1.10 RSVP INTEROPERABILITY

RSVP must operate correctly even when two RSVP-capable routers are joined by an arbitrary “cloud” of non-RSVP routers. This is because RSVP will be deployed gradually in the Internet and might never be implemented in some parts. An intermediate cloud that does not support RSVP will be unable to perform resource reservation. If that cloud has sufficient capacity, however, it may still provide useful real-time service.

Both RSVP and non-RSVP routers forward PATH messages towards the destination address using their local routing table. The PATH message carries to each RSVP-capable node the IP address of the last RSVP-capable router. RESV messages are thus forwarded directly to the next RSVP-capable router on the path(s) back towards the source, as shown in Figure 1.8. Non-RSVP-capable nodes can affect the QoS provided to receivers. Thus, RSVP passes a non-RSVP flag bit (also called global break bit) to the local traffic control mechanism when there are non-RSVP-capable hops in the path. Traffic control forwards such information on the service capability to receivers using the AdSpecs.

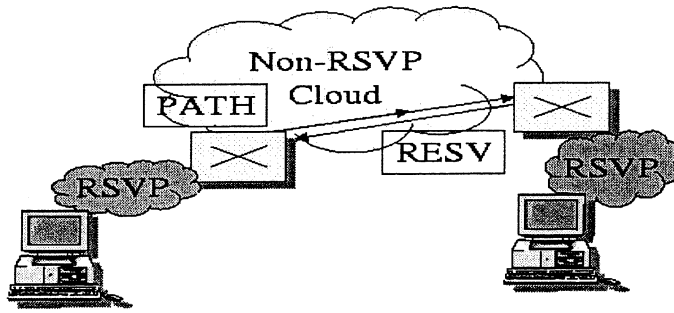

Figure 1.8 Interoperation with non-RSVP capable routers

Some topologies of RSVP and non-RSVP routers can cause RESV messages to arrive at the wrong RSVP-capable node or to arrive at the wrong interface of the correct node. To handle the wrong interface case, a Logical Interface Handle (LIH) is used. In addition to the address of the previous node, the PATH message includes a LIH defining the logical outgoing interface, and this is stored in the path state block. A RESV message arriving at the addressed node carries both the IP address and the LIH of the correct outgoing interface.

1.11 OPEN ISSUES AND CURRENT WORK

RSVP is recommended to be deployed gradually and with caution, first in intranets, then limited Internet Service Provider environments, before being used on a large scale in the Internet. This is because RSVP is still immature and may be changed if problems are found. In addition, some issues pertaining to RSVP remain unresolved. For example, scalability of RSVP is a major concern, since resource requirements for running RSVP on a router increase proportionally to the number of RSVP reservations. To overcome this problem, it is foreseen that the "edge" of the backbone will aggregate groups of streams requiring special service, as is discussed below.

Security considerations are also a subject of current study. It is essential to protect against modified reservation requests used to obtain service to unauthorized parties or lock up network resources. Hop-by-hop checksums and encryption were proposed to detect reservation requests that are modified between RSVP neighbor routers. Key management and distribution for such encryption is not yet in place. Policies for making or limiting reservations are also still being studied. Caution is warranted because of limited experience with setting and controlling such policies.[12]

Much work is currently being done on RSVP security and support by specific link layers, as well as on policy control, aggregation of flows, routing interface, label switching, and diagnostics. Some of these issues, such as security and support by specific link layers, have been previously discussed. The remaining issues will be discussed in this section.

1.11.1 Aggregation and Differentiated Services

An extremely large number of flows travels on network backbone links. Many researchers have expressed concerns about how well RSVP will scale to such situations, since RSVP exhibits overhead in terms of state, bandwidth, and computation required for each flow. One of the solutions proposed to this problem is the aggregation of flows. Aggregation, however, must be provided without affecting the end-to-end QoS guarantees of individual flows.

QoS aggregation in RSVP has two major components. The first is the extension of RSVP to support aggregate QoS requests made by a set of flows rather than individual flows. Aggregate requests are not currently supported by RSVP and require the definition of new filter specifications. The second component is the aggregation of a large number of individual RSVP requests without precluding support for individual QoS guarantees where feasible. This serves to ensure individual end-to-end QoS guarantees, without requiring the awareness of individual flows on every segment of their path.

Routers at the edge of a region doing aggregation keep detailed state, while in the interior of the region routers keep a greatly reduced amount of state. Tunneling of RSVP requests and the use of the router alert option have been proposed to reduce RSVP overhead in the backbones. Packets can be tagged at the edge with scheduling information that will be used in place of the detailed state.[5] One way of doing that is through the use of the type of service (TOS) field in the IP header. This idea has been adopted by the Differentiated Services Working Group, which first met in March 1998.

The aim of the group is to find an alternative to per-flow processing and per-flow state to enable the deployment of differentiated services in large carrier networks. A number of proposals have emerged which suggest that RSVP can be used with the differentiated services model to meet the needs of large Internet service providers. Most of these proposals envision the use of the differentiated services model in large core networks and the use of RSVP in peripheral stub networks (see Figure 1.9). This model enables the scalability of the differentiated services model to be combined with the fine granularity, minimal management requirements, admission control, and policy support of the integrated services and RSVP model.[4]

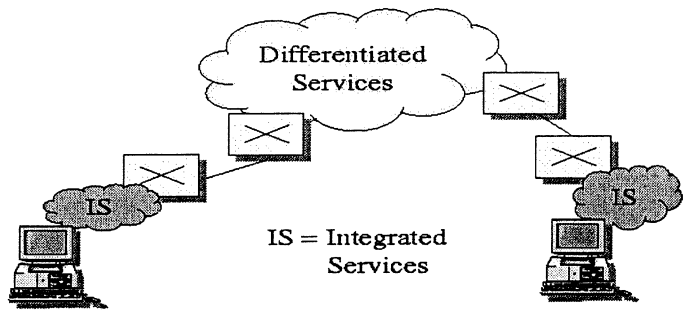

Figure 1.9 Interoperation of integrated services and differentiated services

1.11.2 Policy Control

Since RSVP allows users to reserve network resources, it is important to have firm control over which users are allowed to reserve resources, and how much resources they can reserve. Network managers and service providers must be able to monitor, control, and enforce use of network resources and services based on policies derived from criteria such as the identity of users and applications, traffic or bandwidth requirements, security considerations, and time-of-day or week.

A policy is defined to be the combination of rules and services, where rules define the criteria for resource access and usage. Policy control determines if a user has the permission to make a reservation. This can be as simple as access approval or as complex as sophisticated accounting and debiting mechanisms. The RSVP specification in RFC 2205[8] contains a place holder for policy support in the form of policy data objects that can be carried in RSVP messages. A policy object contains policy-related information, such as policy elements, which are units of information necessary for the evaluation of policy rules, such as the identity of the user. The mechanisms and message formats for policy enforcement are not yet specified. The interface between RSVP and the local policy modules (LPMs) at the network nodes is also not specified. LPMs are responsible for receiving, processing, and forwarding policy data objects. LPMs may also rewrite and modify the objects as they pass through policy nodes.

The policy enforcement point (PEP) is the point where the policy decisions are actually enforced. This usually resides in a network node (for example, a router). The policy decision point (PDP) is the point where policy decisions are made. This may or may not be local to the network node, and if not local it may or may not be located in a policy server. In some cases, local policy decisions need to be made. Such preliminary policy decisions are made by a local decision point (LDP). The final decision is still made by the PDP.[17] Figure 1.10 illustrates these components.

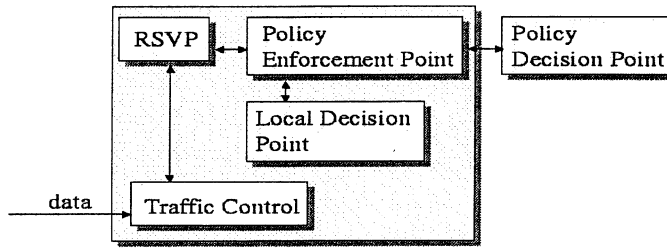

Figure 1.10 Policy control components

When the PEP receives a message requiring a policy decision, it first consults the LDP, if available. Then the PEP sends a policy decision request to the PDP. That request may contain one or more policy objects, in addition to the admission control information in the original message that triggered the policy decision request. It will also contain the LDP decision, if any. The PDP returns the policy decision, and the PEP enforces the policy decision by appropriately accepting or denying the request.

The PDP may also return additional information to the PEP. The interaction between the PEP and PDP can use a protocol such as the Common Open Policy Service (COPS) protocol.

1.11.3 Routing and Label Switching

Routing

A number of proposals for the RSVP interface to the routing protocol are currently being discussed. Some proposals suggest that the interface should not only allow RSVP to request information and services from routing, but also to pass relevant information to the routing protocol. For example, it may be desirable to use different routes for flows belonging to different applications, having different QoS requirements, or different specified explicit routes, even if the packets of these two flows have the same source and destination addresses. If RSVP is to support this type of routing, its interface to routing must allow it to pass information such as port numbers, QoS requirements, and explicit routes, in addition to the addresses. Thus, the RSVP module should pass to the routing module all relevant information that may be useful in making a routing decision.[11]

Label Switching

Multiprotocol Label Switching (MPLS) allows labels to be bound to various granularities of forwarding information, including application flows. Labels can be allocated and bound to RSVP flows, and RSVP messages can be used to distribute the appropriate binding information. Hosts and routers that support both label switching and RSVP can associate labels with RSVP flows. This enables label-switching routers to identify the appropriate reservation state for a packet based on its label value. Two new objects are defined for this purpose: *RSVP label* carries a label in an RSVP message, and *hop count* enables time-to-live processing for RSVP flows that pass through ATM label-switching routers. There are several alternatives to mapping RSVP flows to labels, one of which specifies a model in which, on a given link, each sender to a single RSVP session is associated with one label.[10]

1.11.4 Diagnostics

Diagnostic messages for RSVP are useful for collecting information about the RSVP state along the path. Such information includes information on different hops when a path or reservation request has failed, as well as feedback regarding the details of a reservation that has been made, such as whether, where, or how, the reservation request was merged with those of others. This information can be useful for debugging purposes and for network resource management. Diagnostic messages are independent from any other RSVP control messages and do not change RSVP state. This diagnostic tool can be invoked by a client from any host that may or may not be a participant of the RSVP session to be diagnosed.

Two types of RSVP diagnostic packets are defined: diagnostic request (DREQ) and reply (DREP). A client invokes RSVP diagnostic functions by generating a

DREQ packet and sending it along the RSVP path to be diagnosed. This DREQ packet specifies the RSVP session and a sender host to that session. The DREQ packet starts collecting information at the last node and proceeds backwards towards the sender. Each RSVP-capable router receiving the DREQ packet adds to the packet a response data object containing the router RSVP state for the specified RSVP session, then it forwards the request via unicast to the router that it believes to be the previous hop for the given sender. When the DREQ packet reaches the sender, the sender changes the packet type to DREP and sends the completed response to the original requester.[19]

GLOSSARY

Admission Control Process determines if sufficient resources are available to make the reservation.

AdSpec includes parameters describing the properties of the data path and parameters required by specific QoS control services to operate correctly. The AdSpec is generated by data sources or intermediate network elements and may be used and updated inside the network before being delivered to receiving applications. The receivers can use the AdSpec to predict the end-to-end service.

FilterSpec defines the flow to receive the desired QoS and is contained in an RSVP reservation request. The basic FilterSpec format defined in the present RSVP specification has a very restricted form: sender IP address and optionally the UDP/TCP port number SrcPort.

Fixed Filter creates a distinct reservation per specified sender (without installing separate reservations for each receiver to the same sender).

FlowSpec specifies the QoS desired by the receivers and is contained in an RSVP reservation request. The FlowSpec will generally include a service class and two sets of numeric parameters: an RSpec that defines the desired QoS, and a TSpec that describes the data flow.

Packet Classifier determines the quality of service class of packets according to the requirements.

Packet Scheduler manages the various queues to guarantee the required quality of service.

Policy Control Process determines if the user has permission to make the reservation.

Reservation Protocol (RSVP) is the means by which applications communicate their requirements to the network in an efficient and robust manner. RSVP does not provide any network service; it simply communicates any end-system requirement to the network. Thus RSVP can be viewed as a switch state establishment protocol, rather than just a resource reservation protocol.

Reservation Styles refers to the method by which reservation requests from various receivers in the same session are aggregated inside the network: whether a separate reservation should be established for each upstream sender in the same session, or if a single reservation can be shared among the packets of selected senders (or all senders in that session).

RSpec (R for reserve) gives the quality of service requested from the network.

Session refers to a simplex data flow with a particular (unicast or multicast) destination, transport-layer protocol, and an optional (generalized) destination port.

Shared Explicit Filter is a filter where a single reservation is created, but can only be shared by selected upstream senders.

Soft State is the state maintained at network switches which is periodically and automatically refreshed by RSVP.

Traffic Control refers to the admission control process, packet classifier and packet scheduler.
TSpec (T for traffic) describes the flow traffic pattern to the network.
Wildcard Filter shares the same reservation among all upstream senders, reserving resources to satisfy the largest resource request (regardless of the number of senders).

REFERENCES

1. F. Baker. RSVP cryptographic authentication. Work in progress, draft-ietf-rsvp-md5-05.txt, August 1997.
2. F. Baker, J. Krawczyk, and A. Sastry. RSVP management information base using SMIv2. RFC 2206, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2206.txt.
3. L. Berger and T. O'Malley. RSVP extensions for IPSEC data flows. RFC 2207, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2207.txt.

.4. Y. Bernet, R. Yavatkar, P. Ford, F. Baker, and L. Zhang. A framework for end-to-end QoS combining RSVP/Intserv and differentiated services. Work in progress, draft-bernet-intdiff-00.tx, March 1998.

5. S. Berson and S. Vincent. Aggregation of internet integrated services state. Work in progress, draft-berson-classy-approach-01.txt, November 1997.
6. R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview. RFC 1633, June 1994. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc1633.txt.
7. R. Braden and L. Zhang. Resource reservation protocol (RSVP) - version 1 message processing rules. RFC 2209, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2209.txt.
8. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP). RFC 2205, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2205.txt.
9. E. Crawley, L. Berger, S. Berson, F. Baker, M. Borden, and J. Krawczyk. A framework for integrated services and RSVP over ATM. Work in progress, draft-ietf-issll-atm-framework-03.txt, April 1998.
10. B. Davie, Y. Rekhter, E. Rosen, A. Viswanathan, V. Srinivasan, and S. Blake. Use of label switching with RSVP. Work in progress, draft-dietf-mpls-rsvp-00.txt, March 1998.
11. R. Guerin, S. Kamat, and E. Rosen. Extended RSVP-routing interface. Work in progress, draft-guerin-ext-rsvp-routing-intf-00.txt, July 1997.
12. A. Mankin, F. Baker, R. Braden, S. Bradner, M. O'Dell, A. Romanow, A. Weinrib, and L. Zhang. Resource reservation protocol (RSVP) - version 1 applicability statement: Some guidelines on deployment. RFC 2208, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2208.txt.
13. S. Shenker, C. Partridge, and R. Guerin. Specification of guaranteed quality of service. RFC 2212, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2212.txt.
14. J. Wroclawski. Specification of the controlled-load network element service. RFC 2211, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2211.txt.
15. J. Wroclawski. The use of RSVP with IETF integrated services. RFC 2210, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2210.txt.
16. R. Yavatkar, D. Hoffman, Y. Bernet, and F. Baker. SBM (subnet bandwidth manager): Protocol for RSVP-based admission control over IEEE 802-style networks. Work in progress, draft-ietf-issll-is802-bm-05.txt, November 1997.

17. R. Yavatkar, D. Pendarakis, and R. Guerin. A framework for policy-based admission control. Work in progress, draft-ietf-rap-framework-00.txt, November 1997.
18. L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new Resource ReSerVation Protocol. *IEEE Network Magazine*, September 1993. ftp://parcftp.xerox.com/pub/net-research/rsvp.ps.Z.
19. L. Zhang and A. Terzis. RSVP diagnostic messages. Work in progress, draft-ietf-rsvp-diagnostic-msgs-03.txt, November 1997.