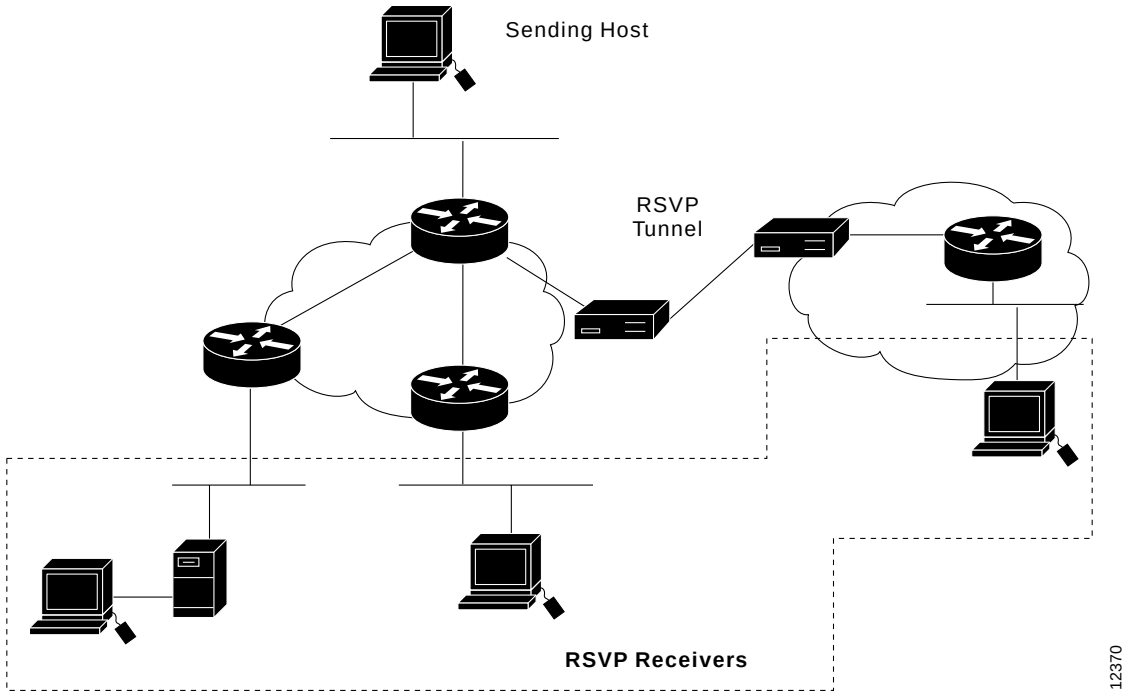
Resource Reservation Protocol (RSVP)

Background

The Resource Reservation Protocol (RSVP) is a network-control protocol that enables Internet applications to obtain special qualities of service (QoSs) for their data flows. RSVP is not a routing protocol; instead, it works in conjunction with routing protocols and installs the equivalent of dynamic access lists along the routes that routing protocols calculate. RSVP occupies the place of a transport protocol in the OSI model seven-layer protocol stack. RSVP originally was conceived by researchers at the University of Southern California (USC) Information Sciences Institute (ISI) and Xerox Palo Alto Research Center. The Internet Engineering Task Force (IETF) is now working toward standardization through an RSVP working group. RSVP operational topics discussed in this chapter include data flows, quality of service, session startup, reservation style, and soft state implementation. Figure 43-1 illustrates an RSVP environment.

Figure 43-1 In RSVP, host information is delivered to receivers over data flows.



RSVP Data Flows

In RSVP, a data flow is a sequence of messages that have the same source, destination (one or more), and quality of service. QoS requirements are communicated through a network via a *flow specification*, which is a data structure used by internetwork hosts to request special services from the internetwork. A flow specification often guarantees how the internetwork will handle some of its host traffic.

RSVP supports three traffic types: *best-effort*, *rate-sensitive*, and the *delay-sensitive*. The type of data-flow service used to support these traffic types depends on QoS implemented. The following sections address these traffic types and associated services. For information regarding QoS, refer to the appropriate section later in this chapter.

Best-effort traffic is traditional IP traffic. Applications include file transfer, such as mail transmissions, disk mounts, interactive logins, and transaction traffic. The service supporting best-effort traffic is called *best-effort service*.

Rate-sensitive traffic is willing to give up timeliness for guaranteed rate. Rate-sensitive traffic, for example, might request 100 kbps of bandwidth. If it actually sends 200 kbps for an extended period, a router can delay traffic. Rate-sensitive traffic is not intended to be run over a circuit-switched network; however, it usually is associated with an application that has been ported from a circuit-switched network (such as ISDN) and is running on a datagram network (IP).

An example of such an application is H.323 videoconferencing, which is designed to run on ISDN (H.320) or ATM (H.310) but is found on the Internet. H.323 encoding is a constant rate or nearly constant rate, and it requires a constant transport rate. The RSVP service supporting rate-sensitive traffic is called *guaranteed bit-rate service*.

Delay-sensitive traffic is traffic that requires timeliness of delivery and varies its rate accordingly. MPEG-II video, for example, averages about 3 to 7 Mbps depending on the amount of change in the picture. As an example, 3 Mbps might be a picture of a painted wall, although 7 Mbps would be required for a picture of waves on the ocean. MPEG-II video sources send key and delta frames. Typically, one or two key frames per second describe the whole picture, and 13 or 28 frames describe the change from the key frame. Delta frames are usually substantially smaller than key frames. As a result, rates vary quite a bit from frame to frame. A single frame, however, requires delivery within a frame time or the CODEC is unable to do its job. A specific priority must be negotiated for delta-frame traffic. RSVP services supporting delay-sensitive traffic are referred to as *controlled-delay service* (non-real time service) and *predictive service* (real-time service).

RSVP Data Flows Process

RSVP data flows are generally characterized by *sessions*, over which data packets flow. A session is a set of data flows with the same unicast or multicast destination, and RSVP treats each session independently. RSVP supports both unicast and multicast sessions (where a session is some number of senders talking to some number of receivers), whereas a flow always originates with a single sender. Data packets in a particular session are directed to the same IP destination address or a generalized destination port. The IP destination address can be the group address for multicast delivery or the unicast address of a single receiver. A generalized destination port can be defined by a UDP/TCP destination port field, an equivalent field in another transport protocol, or some application-specific information.

RSVP data distribution is handled via either multicasts or unicasts. Multicast traffic involves a copy of each data packet forwarded from a single sender toward multiple destinations. Unicast traffic features a session involving a single receiver. Even if the destination address is unicast, there might

be multiple receivers, distinguished by a generalized port. Multiple senders also might exist for a unicast destination, in which case, RSVP can set up reservations for multipoint-to-point transmission.

Each RSVP sender and receiver can correspond to a unique Internet host. A single host, however, can contain multiple logical senders and receivers, distinguished by generalized ports.

RSVP Quality of Service (QoS)

In the context of RSVP, quality of service (QoS) is an attribute specified in flow specifications that is used to determine the way in which data interchanges are handled by participating entities (routers, receivers, and senders). RSVP is used to specify the QoS by both hosts and routers. Hosts use RSVP to request a QoS level from the network on behalf of an application data stream. Routers use RSVP to deliver QoS requests to other routers along the path(s) of the data stream. In doing so, RSVP maintains the router and host state to provide the requested service.

RSVP Session Start-up

To initiate an RSVP multicast session, a receiver first joins the multicast group specified by an IP destination address by using the Internet Group-Membership Protocol (IGMP). In the case of a unicast session, unicast routing serves the function that IGMP, coupled with Protocol-Independent Multicast (PIM), serves in the multicast case. After the receiver joins a group, a potential sender starts sending RSVP path messages to the IP destination address. The receiver application receives a path message and starts sending appropriate reservation-request messages specifying the desired flow descriptors using RSVP. After the sender application receives a reservation-request message, the sender starts sending data packets.

RSVP Reservation Style

Reservation style refers to a set of control options that specify a number of supported parameters. RSVP supports two major classes of reservation: *distinct reservations* and *shared reservations.* Distinct reservations install a flow for each relevant sender in each session. A shared reservation is used by a set of senders that are known not to interfere with each other. Figure 43-2 illustrates distinct and shared RSVP reservation-style types in the context of their scope. Each supported reservation style/scope combination is described following the illustration.

Wildcard-Filter (WF) Style

The *wildcard-filter (WF) style* specifies a shared reservation with a wildcard scope. With a WF-style reservation, a single reservation is created into which flows from all upstream senders are mixed. Reservations can be thought of as a shared pipe whose size is the largest of the resource requests for that link from all receivers, independent of the number of senders. The reservation is propagated upstream toward all sender hosts and is automatically extended to new senders as they appear.

Figure 43-2 RSVP supports both distinct reservations and shared reservations.

Scope	Reservations	
	Distinct	Shared
Explicit	Fixed-Filter (FF) Style	Shared-Explicit (SE) Style
Wildcard	None Defined	Wildcard-Filter (WF) Style

ith4102

Fixed-Filter (FF) Style

The *fixed-filter (FF) style* specifies a distinct reservation with an explicit scope. With an FF-style reservation, a distinct reservation request is created for data packets from a particular sender. The reservation scope is determined by an explicit list of senders. The total reservation on a link for a given session is the total of the FF reservations for all requested senders. FF reservations that are requested by different receivers but select the same sender, however, must be merged to share a single reservation in a given node.

Shared-Explicit (SE) Style

The *shared-explicit (SE) style* reservation specifies a shared reservation environment with an explicit reservation scope. The SE style creates a single reservation into which flows from all upstream senders are mixed. As in the case of an FF reservation, the set of senders (and therefore the scope) is specified explicitly by the receiver making the reservation.

RSVP Reservation Style Implications

WF and SE are both shared reservations that are appropriate for multicast applications in which application-specific constraints make it unlikely that multiple data sources will transmit simultaneously. An example might be audio-conferencing, where a limited number of people talk at once. Each receiver might issue a WF or SE reservation request twice for one audio channel (to allow some over-speaking). The FF style creates independent reservations for the flows from different senders. The FF style is more appropriate for video signals. Unfortunately, it is not possible to merge shared reservations with distinct reservations.

RSVP Soft State Implementation

In the context of an RSVP, a *soft state* refers to a state in routers and end nodes that can be updated by certain RSVP messages. The soft state characteristic permits an RSVP network to support dynamic group membership changes and adapt to changes in routing. In general, the soft state is maintained by an RSVP-based network to enable the network to change states without consultation with end points. This contrasts with a circuit-switch architecture in which an end point places a call and, in the event of a failure, places a new call.

RSVP protocol mechanisms provide a general facility for creating and maintaining a distributed reservation state across a mesh of multicast and unicast delivery paths.

To maintain a reservation state, RSVP tracks a soft state in router and host nodes. The RSVP soft state is created and periodically refreshed by path and reservation-request messages. The state is deleted if no matching refresh messages arrive before the expiration of a cleanup timeout interval. The soft state also can be deleted as the result of an explicit teardown message. RSVP periodically scans the soft state to build and forward path and reservation-request refresh messages to succeeding hops.

When a route changes, the next path message initializes the path state on the new route. Future reservation-request messages establish a reservation state. The state on the now-unused segment is timed out. (The RSVP specification requires initiation of new reservations through the network two seconds after a topology change.)

When state changes occur, RSVP propagates those changes from end to end within an RSVP network without delay. If the received state differs from the stored state, the stored state is updated. If the result modifies the refresh messages to be generated, refresh messages are generated and forwarded immediately.

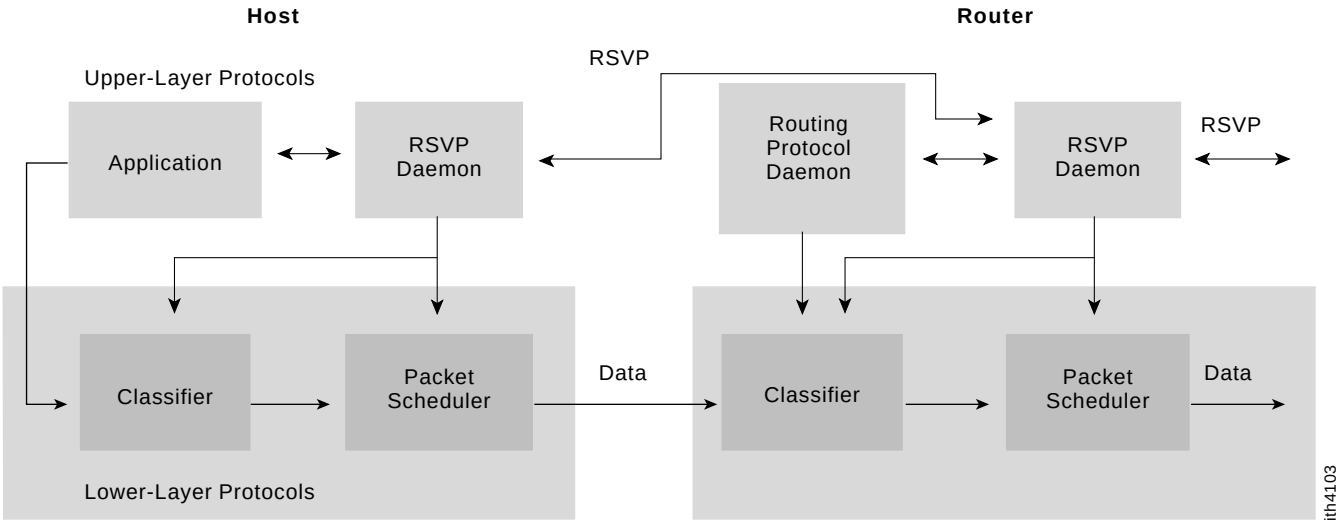
RSVP Operational Model

Under RSVP, resources are reserved for simple data streams (that is, unidirectional data flows). Each sender is logically distinct from a receiver, but any application can act as a sender and receiver. Receivers are responsible for requesting resource reservations. Figure 43-3 illustrates this general operational environment, while the subsequent section provides an outline of the specific sequence of events.

General RSVP Protocol Operation

The RSVP resource-reservation process initiation begins when an RSVP daemon consults the local routing protocol(s) to obtain routes. A host sends IGMP messages to join a multicast group and RSVP messages to reserve resources along the delivery path(s) from that group. Each router that is capable of participating in resource reservation passes incoming data packets to a packet classifier and then queues them as necessary in a packet scheduler. The RSVP packet classifier determines the route and QoS class for each packet. The RSVP scheduler allocates resources for transmission on the particular data link layer medium used by each interface. If the data link layer medium has its own QoS management capability, the packet scheduler is responsible for negotiation with the data-link layer to obtain the QoS requested by RSVP.

Figure 43-3 The RSVP operational environment reserves resources for unidirectional data flows.



The scheduler itself allocates packet-transmission capacity on a QoS-passive medium, such as a leased line, and also can allocate other system resources, such as CPU time or buffers. A QoS request, typically originating in a receiver host application, is passed to the local RSVP implementation as an RSVP daemon.

The RSVP protocol then is used to pass the request to all the nodes (routers and hosts) along the reverse data path(s) to the data source(s). At each node, the RSVP program applies a local decision procedure called admission control to determine whether it can supply the requested QoS. If admission control succeeds, the RSVP program sets the parameters of the packet classifier and scheduler to obtain the desired QoS. If admission control fails at any node, the RSVP program returns an error indication to the application that originated the request.

RSVP Tunneling

It is impossible to deploy RSVP or any new protocol at the same moment throughout the entire Internet. Indeed, RSVP might never be deployed everywhere. RSVP therefore must provide correct protocol operation even when two RSVP-capable routers are joined by an arbitrary cloud of non-RSVP routers. An intermediate cloud that does not support RSVP is unable to perform resource reservation, so service guarantees cannot be made. If, however, such a cloud has sufficient excess capacity, it can provide acceptable and useful real-time service.

To support connection of RSVP networks through non-RSVP networks, RSVP supports tunneling, which occurs automatically through non-RSVP clouds. Tunneling requires RSVP and non-RSVP routers to forward path messages toward the destination address by using a local routing table. When a path message traverses a non-RSVP cloud, the path-message copies carry the IP address of the last RSVP-capable router. Reservation-request messages are forwarded to the next upstream RSVP-capable router.

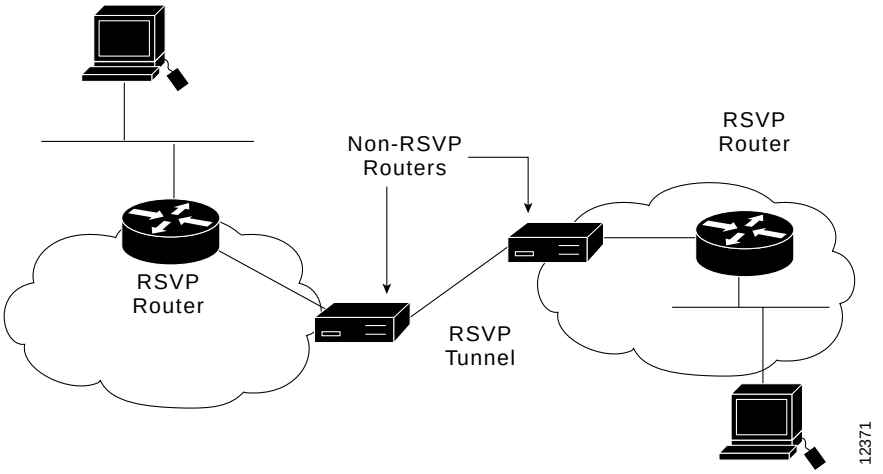
Two arguments have been offered in defense of implementing tunneling in an RSVP environment. First, RSVP will be deployed sporadically rather than universally. Second, by implementing congestion control in situations known to be highly congested, tunneling can be made more effective.

Sporadic, or piecemeal, deployment means that some parts of the network will actively implement RSVP before others parts. If RSVP is required end to end, no benefit is achievable without nearly universal deployment, which is unlikely unless early deployment shows substantial benefits.

Weighted Fair-Queuing Solution

Having the technology to enforce effective resource reservation (such as Cisco's weighted fair-queuing scheme) in a location that presents a bottleneck can have real positive effects. Tunneling presents a risk only when the bottleneck is within a non-RSVP domain and the bottleneck cannot be avoided. Figure 43-4 illustrates an RSVP environment featuring a tunnel between RSVP-based networks.

Figure 43-4 An RSVP environment can feature a tunnel between RSVP-based networks.



RSVP Messages

RSVP supports four basic message types: *reservation-request messages*, *path messages*, *error* and *confirmation messages*, and *teardown messages*. Each of these is described briefly in the sections that follow.

Reservation-Request Messages

A reservation-request message is sent by each receiver host toward the senders. This message follows in reverse the routes that the data packets use, all the way to the sender hosts. A reservation-request message must be delivered to the sender hosts so that the hosts can set up appropriate traffic-control parameters for the first hop. RSVP does not send any positive acknowledgment messages.

Path Messages

An RSVP path message is sent by each sender forward along the unicast or multicast routes provided by the routing protocol(s). A path message is used to store the path state in each node. The path state is used to route reservation-request messages in the reverse direction.

Error and Confirmation Messages

Three error and confirmation message forms exist: *path-error* messages, *reservation-request error* messages, and *reservation-request acknowledgment* messages.

Path-error messages result from path messages and travel toward senders. Path-error messages are routed hop-by-hop using the path state. At each hop, the IP destination address is the unicast address of the previous hop.

Reservation-request error messages result from reservation-request messages and travel toward the receiver. Reservation-request error messages are routed hop-by-hop using the reservation state. At each hop, the IP destination address is the unicast address of the next-hop node. Information carried in error messages can include the following:

- Admission failure
- Bandwidth unavailable
- Service not supported
- Bad flow specification
- Ambiguous path

Reservation-request acknowledgment messages are sent as the result of the appearance of a reservation-confirmation object in a reservation-request message. This acknowledgment message contains a copy of the reservation confirmation. An acknowledgment message is sent to the unicast address of a receiver host, and the address is obtained from the reservation-confirmation object. A reservation-request acknowledgment message is forwarded to the receiver hop-by-hop (to accommodate the hop-by-hop integrity-check mechanism).

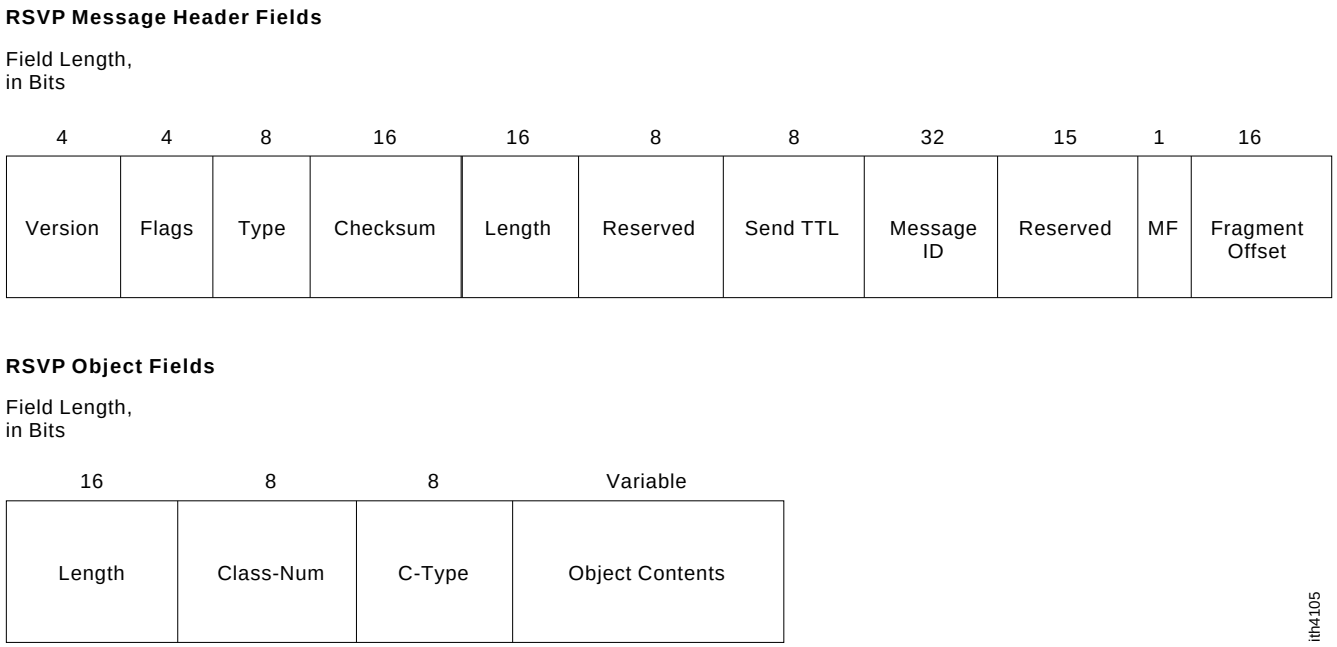
Teardown Messages

RSVP teardown messages remove the path and reservation state without waiting for the cleanup timeout period. Teardown messages can be initiated by an application in an end system (sender or receiver) or a router as the result of state timeout. RSVP supports two types of teardown messages: *path-teardown messages* and *reservation-request teardown* messages. Path-teardown messages delete the path state (which deletes the reservation state), travel toward all receivers downstream from the point of initiation, and are routed like path messages. Reservation-request teardown messages delete the reservation state, travel toward all matching senders upstream from the point of teardown initiation, and are routed like corresponding reservation-request messages.

RSVP Packet Format

Figure 43-5 illustrates the RSVP packet format. The summaries that follow outline the header and object fields illustrated in Figure 43-5.

Figure 43-5 An RSVP packet format consists of message headers and object fields.



RSVP Message Header Fields

RSVP message-header fields are comprised of the following:

- *Version*—4-bit field indicating the protocol version number (currently version 1).
- *Flags*—4-bit field with no flags currently defined.
- *Type*—8-bit field with 6 possible (integer) values, as shown in table 43-1.

Table 43-1 RSVP Message Type Field Values

Value	Message Type
1	Path
2	Reservation-request
3	Path-error
4	Reservation-request error
5	Path-teardown
6	Reservation-teardown
7	Reservation-request acknowledgment

- *Checksum*—16-bit field representing a standard TCP/UDP checksum over the contents of the RSVP message, with the checksum field replaced by zero.
- *Length*—16-bit field representing the length of this RSVP packet in bytes, including the common header and the variable-length objects that follow. If the More Fragment (MF) flag is set or the fragment offset field is non-zero, this is the length of the current fragment of a larger message.

- *Send TTL*—8-bit field indicating the IP time-to-live (TTL) value with which the message was sent.
- *Message ID*—32-bit field providing a label shared by all fragments of one message from a given next/previous RSVP hop.
- *More Fragments (MF) Flag*—Low-order bit of a 1-byte word with the other 7 high-order bits specified as reserved. MF is set on for all but the last fragment of a message.
- *Fragment Offset*—24-bit field representing the byte offset of the fragment in the message

RSVP Object Fields

RSVP object fields are comprised of the following:

- *Length*—16-bit field containing the total object length in bytes (must always be a multiple of 4 and be at least 4).
- *Class-Num*—Identifies the object class. Each object class has a name. An RSVP implementation must recognize the classes listed in Table 43-2.

The high-order bit of the Class-Num determines what action a node should take if it does not recognize the Class-Num of an object.

- *C-Type*—Object type, unique within Class-Num. The maximum object content length is 65528 bytes. Class-Num and C-Type fields (together with the flag bit) can be used together as a 16-bit number to define a unique type for each object.
- *Object Contents*—The Length, Class-Num, and C-Type fields specify the form of the object content. Refer to Table 43-2 for definitions of the classes of objects that can be included in the object contents.

Table 43-2 RSVP Object Classes

Object Class	Description
Null	Contains a Class-Num of zero, and its C-Type is ignored. Its length must be at least 4 but can be any multiple of 4. A null object can appear anywhere in a sequence of objects, and its contents will be ignored by the receiver.
Session	Contains the IP destination address and possibly a generalized destination port to define a specific session for the other objects that follow (required in every RSVP message).
RSVP Hop	Carries the IP address of the RSVP-capable node that sent this message.
Time Values	If present, contains values for the refresh period and the state TTL to override the default values.
Style	Defines the reservation style plus style-specific information that is not a flow-specification or filter-specification object (included in a reservation-request message).
Flow Specification	Defines a desired QoS (included in a reservation-request message).
Filter Specification	Defines a subset of session-data packets that should receive the desired QoS (specified by a flow-specification object within a reservation-request message).
Sender Template	Contains a sender IP address and perhaps some additional demultiplexing information to identify a sender (included in a path message).
Sender TSPEC	Defines the traffic characteristics of a sender's data stream (included in a path message).
Adspec	Carries advertising data in a path message.
Error Specification	Specifies an error (included in a path-error or reservation-request error message).
Policy Data	Carries information that will enable a local policy module to decide whether an associated reservation is administratively permitted (included in a path or reservation-request message).
Integrity	Contains cryptographic data to authenticate the originating node and perhaps to verify the contents of this reservation-request message.
Scope	An explicit specification of the scope for forwarding a reservation-request message.
Reservation Confirmation	Carries the IP address of a receiver that requested a confirmation. Can appear in either a reservation-request or reservation-request acknowledgment.