The Integrated Services in the Internet: State of the Art

PAUL P. WHITE AND JON CROWCROFT, SENIOR MEMBER, IEEE

Invited Paper

This paper is about the evolution of the Internet from a simple data network into a true multiservice network that can support the emerging multimedia applications and their protocols with appropriate performance and costs. The real-time delivery and specific bandwidth requirements of these multimedia applications have created a need for an integrated services Internet in which traditional best effort datagram delivery can coexist with additional enhanced quality of service delivery classes. The integrated services Internet will be able to commit to meet bandwidth, packet loss, and delay specifications for individual data flows by using the resource reservation protocol, together with appropriate packet forward scheduling policies.

Keywords—Controlled-load service, data flow, guaranteed service, integrated services, multicast, one pass with advertising, quality of service, random early detection, real-time protocol, reservation setup protocol.

I. INTRODUCTION

The Internet has evolved from a classical, closed-community data network into the infrastructure for the global information society. Users are coming to trust the Internet for business, including not only the burgeoning use of the World Wide Web and traditional applications such as e-mail and file transfer but increasingly also "real-time" multimedia such as audio, video, and shared applications (such as a whiteboard or meeting planner or shared document editor).

The original design goal of the Internet was to serve as a highly fault-tolerant data network for the defense community. To this end, the amount of state shared between the network and end systems is minimized, being merely sufficient information to calculate a set of routes and, for each packet sent by an end system, to determine the best current route. In its original implementation, a great deal of attention was paid to this distributed, dynamic route calculation, and not too much to the performance aspects of packet forwarding. It was not anticipated that the typical military users would share the network with other users or let it become overloaded. Because of this, early router implementations used “first in, first out” (FIFO) queues for all traffic, whose behavior under overload conditions was simply to drop the latest arriving packets. The only reason for this was simplicity.

Over time, it became apparent that traffic management for different applications, for different users, and for overloaded networks would be essential. A number of stages in the evolution of the protocols and implementations of the Internet have followed.

1) The first service differentiation was by application, and the idea of type of service (ToS) forwarding was proposed, albeit only ever deployed in a few research networks [18]. Conceptually, routers distinguish traffic either by examining special bits in the Internet protocol (IP) packet header or by application protocol (deduced by spying at high-level protocol fields, specifically, the transport protocol port numbers. This is often feasible since routers also commonly implement filters for security reasons based on these same fields). ToS routing is a win-win enhancement for some mixes of applications. For example, bulk transfer, versus interactive terminal traffic, can see a mutual benefit by allowing higher throughput for large file transfers but lower delay for light traffic flows, such as telnet typically offers. Although some commercial routers have a ToS forwarding feature, few applications currently take advantage of it.

2) Telnet, file transfer protocol (FTP), and the World Wide Web (WWW)'s application protocol, hypertext transfer protocol, all operate over the transmission control protocol (TCP), which enhances the inherently unreliable IP, to provide ordered, reliable delivery. In 1988, TCP was reengineered, a decade after

1The user datagram protocol (UDP) was experimentally used for carrying voice traffic even in the early Advanced Research Project Agency Network days, but the network was typically not overloaded, so that packets encountered no significant queues and there was no need for any traffic control. We discuss this more later.
its original design, to provide end-to-end congestion control. The net effect of ToS routing and TCP congestion control is to offer a network that provides a fair share to the set of concurrent users. (In fact, it is largely only an approximation to a fair share, but more recent work on slightly different queuing strategies may improve this—we discuss this later.) Because all TCP users achieve only a fair share of the capacity available at any moment, it has been common not to collect time-based usage charges, since the time to transfer any item of data depends on the number of other active users. An access leasing system is operated by most commercial Internet service providers, and this leads to very efficient cost recovery.

3) Group communication is very useful for applications that involve multiple simultaneous senders and receivers. The multicast backbone (MBone) is a virtual overlay on the current Internet that provides multicast [14], which results in a massive reduction of load on the network for such applications. Initially, it was expected that this might be used for replicated transactions, but in the absence of transport (end-to-end) protocols to support these, what was deployed more quickly was a family of applications based on the UDP, namely, the audio and video conferencing programs.

4) Typical applications based on UDP offer an approximately fixed rate of packets to the network. Fundamentally, to support arbitrary numbers of users of these applications and not suffer overload, the Internet had to be enhanced in some way. A number of possibilities exist [19]:
   a) overengineering;
   b) resource reservation;
   c) usage-based charging;

The first approach may work for a large class of applications, but many researchers and practitioners believe that there will always be some application that can dominate the network capacity, so b) and perhaps c) as well are needed.

The second of these approaches has received the most attention in the protocol standards development arena, and we discuss this at length in the rest of this paper. The third approach is perceived as complex in the Internet community and possibly very hard to deploy—typically, commercial Internet providers find this difficult to install in the absence of a set of global agreements on tariffs.

II. APPLICATION REQUIREMENTS

While computers have infinite patience, humans have limited tolerance for delay. This is reflected mostly in the throughput requirements for traditional applications in the Internet, such as FTP, e-mail, and WWW. With multimedia applications, it is more important that the quality of communication is maintained. This means that latency must be bounded and jitter must be controlled, but also that both loss and a minimum throughput are tolerable. With sufficient engineering [16], applications may be highly loss tolerant, but there are still limits.

It is possible that either simple congestion or pricing alone will cause people to cease to use applications when a network is overloaded. It depends very much, however, on the mix of end-to-end protocol behavior in an FIFO queuing system with “tail drop,” just exactly what mix of applications are running at the stable operating point of a full network. Provided that all applications use the same scheme to adapt to a bottleneck, there is no problem in achieving reasonable fairness. Multicast UDP-based applications cannot use the feedback scheme used in TCP, however, and as yet, such congestion control is not well understood.³

One alternative approach is to alter the network to keep some kind of minimal state by observation of current traffic and to penalize so-called badly behaved sources. One extremely promising scheme is RED [17]. As the name suggests, random early detection is a mechanism to give feedback at the onset of congestion rather than after congestion has occurred. Essentially, the mechanism uses the persistent presence of a number of packets in a bottleneck queue as an indication of uncooperative behavior and penalizes such sources with a higher loss rate. Under normal conditions, adaptive sources adapt sooner, so that the queues are shorter than without RED operating. This leads to more buffer’s being available for legitimate short-term bursts. A bias against persistent long-term bursts must still be retained to prevent hogging.

It must be assumed, however, that the “gene pool” of congestion control, or adaptive applications, may increase to more than one (namely, TCP’s) and that at such a point, the service differentiation task will become more complex. In any case, RED only alleviates congestion. It cannot guarantee a minimum packet rate that will be forwarded. In such cases, it is believed that dynamic specification of a source (or receiver) requirement is preferable. To this end, the resource reservation protocol (RSVP) can be used, for both unicast and multicast, to request “special treatment” for a traffic flow.

There are two main classes of multimedia applications that might make such a request:

1) adaptive applications, with users that may benefit from some minimum capacity’s being guaranteed at some reasonable probability;

²While there are some adaptive multimedia applications, most notably S. McCanne’s work on Vic, I. Wakeman’s work on Ivs and the INRIA Free Phone, and UCL’s robust audio tool [15], these are currently a minority of approaches. In any case, an audio or video application does have a limited acceptable range over which it can adapt.

³“Drop tail” is the term sometimes used to describe the behavior of an FIFO queuing system when it runs out of buffer. Packets arriving at the full buffer are dropped. This is in contrast to random early detection (RED) and related schemes, where packets are dropped from other (perhaps random) places in the overloaded queue to reflect more fairly the different contribution of different sources to the overload circumstance.
2) legacy, constant-rate applications with a requirement for strong guarantees of service.\(^4\)

In the next three sections, we describe the emerging system for supporting these services in detail.

III. INTERNET ENGINEERING TASK FORCE INTEGRATED SERVICES

In response to the growing demand for an integrated services (IS) Internet, the Internet Engineering Task Force (IETF)\(^5\) set up an Integrated Services Working Group,\(^6\) which has defined several service classes that, if supported by the routers traversed by a data flow,\(^7\) can provide the data flow with certain QoS commitments. By contrast, best effort traffic entering a router will receive no such service commitment and will have to make do with whatever resources are available. The level of QoS provided by these enhanced QoS classes is programmable on a per-flow basis according to requests from the end applications. These requests can be passed to the routers by network-management procedures or, more commonly, by using a reservation protocol such as RSVP, which is described in Section IV. The requests dictate the level of resources (e.g., bandwidth, buffer space) that must be reserved, along with the transmission scheduling behavior that must be installed in the routers to provide the desired end-to-end QoS commitment for the data flow.

In determining the resource allocations necessary to satisfy a request, the router needs to take account of the QoS support provided by the link layer in the data-forwarding path. Furthermore, in the case of a QoS-active link layer, such as ATM or certain types of local-area networks (LAN’s), the router is responsible for negotiations with the link layer to ensure that the link layer installs appropriate QoS support should the request be accepted. This mapping to link-layer QoS is medium dependent, and the mechanisms for doing so are currently being defined by the Integrated Services over Specific Lower Layers Working Group of the IETF.\(^8\) In the case of a QoS-passive link layer such as a leased line, the mapping to the link-layer QoS is trivial since transmission capacity is handled entirely by the router’s packet scheduler.

Each router must apply admission control to requests to ensure that they are only accepted if sufficient local resources are available. In making this check, admission control must consider information supplied by end applications regarding the traffic envelope\(^9\) that their data flow will fall within. One of the parameters in the traffic envelope that must be supplied is the maximum datagram size of the data flow. Should this be greater than the maximum transmission unit (MTU) of the link, then admission control will reject the request since the integrated services models rely on the assumption that datagrams receiving an enhanced QoS class are never fragmented.

Once an appropriate reservation has been installed in each router along the path, the data flow can expect to receive an end-to-end QoS commitment, provided no path changes or router failures occur during the lifetime of the flow\(^10\) and that the data flow conforms to the traffic envelope supplied in the request. Service-specific policing and traffic shaping actions, as described in Sections III-A and III-B, will be employed within the network to ensure that nonconforming data flows do not affect the QoS commitments for behaving data flows. The IETF has considered various QoS classes such as [1], [6], [10], and [12], although to date, only two of these—guaranteed service [10] and controlled-load service [12]—have been formally specified for use with RSVP [13]. First, we will discuss the simpler of these services, namely, controlled load.

\(\text{A. Controlled-Load Service}\)

Controlled-load service [12] provides approximately the same QoS under heavy loads as under light loads. A description of the traffic characteristics (the tspec, described in Section III-B) for the flow desiring controlled-load service must be submitted to the router as for the case of guaranteed service, although it is not necessary to include the peak-rate parameter. If the flow is accepted for controlled-load service, then the router makes a commitment to offer the flow a service equivalent to that seen by a best effort flow on a lightly loaded network. The important difference from the traditional Internet best effort service is that the controlled-load flow does not noticeably deteriorate as the network load increases. This will be true regardless of the level of load increase. By contrast, a best effort flow would experience progressively worse service (higher delay and loss) as the network load increased. The controlled-load service is intended for those classes of applications that can tolerate a certain amount of loss or delay, provided it is kept to a reasonable level. Examples of applications in this category include adaptive real-time applications.

Controlled load has some fairly simple implementations in terms of the queuing systems in routers. It also functions adequately for the existing MBone applications, which can adapt to the modest (small) scale end-to-end delay, variations, and jitter that it may introduce, through the

\(^4\)In the asynchronous transfer mode (ATM) standards, there are a variety of network services targeted at a range of applications that are quite similar (although extending to other application models, too). The specification of the application behavior and the network service are also separated in a way that is quite similar to the integrated service Internet, as we shall see below.

\(^5\)See http://www.ietf.cnri.reston.va.us.


\(^7\)A data flow identifies the set of packets to receive special quality of service (QoS). It is defined by a “session” comprising the IP address, transport layer protocol type, and port number of the destination, along with a list of specific senders to that session that are entitled to receive the special QoS. Each sender is identified by source address and port number, while its protocol type must be the same as for the session.

\(^8\)http://www.ietf.org/html.charters/issl-charter.html.

\(^9\)Think of the packet arrival process as having a pattern like a waveform. The envelope is its typical shape.

\(^10\)Due to refresh timers used by RSVP, a flow may recover its QoS commitment without taking any special action. With the advent of QoS-aware routing, it may be that there is not even a gap in the perceived provision of the service contract.
use of adaptive play-out buffering [16]. It is not suited to applications that require very low latency (e.g., distributed virtual reality systems and so forth). Next, we discuss the service provided where the user requires some commitment to a delay guarantee, namely, guaranteed service.

1) Example of Controlled-Load Usage/Requirement: One possible example of the use of the controlled-load service is that of MBone applications over a private so-called intranet. Here, traffic conditions and global policies can be managed such that a statistical throughput guarantee is enough and propagation delays will be low enough that for most users, interactive software-based multimedia conferencing tools will perform adequately. A more interesting example might be the provision of systems network architecture (SNA) or Digital Equipment Corporation (DEC) local-area terminal (LAT) tunnelling across a public Internet service provider’s backbone network. SNA and DEC LAT are both somewhat delay sensitive due to their detailed protocol operations, although not as much as some real-time systems. However, using the same Internet path to carry them with arbitrary interference from other applications’ flows would not work well (or at all).

B. Guaranteed Service

Guaranteed service [10] provides an assured level of bandwidth, a firm end-to-end delay bound, and no queuing loss for conforming packets of a data flow. It is intended for applications with stringent real-time delivery requirements such as certain audio and video applications that have fixed “play-out” buffers and are intolerant of any datagram arriving after their playback time. Guaranteed service really addresses the support of “legacy” applications that expect a delivery model similar to traditional telecommunications circuits.

Each router characterizes the guaranteed service for a specific flow by allocating a bandwidth $R$ and buffer space $B$ that the flow may consume. This is done by approximating the “fluid model” of service [8], [9] so that the flow effectively sees a dedicated wire of bandwidth $R$ between the source and receiver. In a perfect fluid model, a flow conforming to a token bucket of rate $r$ and depth $b$ will have its delay bounded by $b/R$, provided $R \geq r$. To allow for deviations from this perfect fluid model in the router’s approximation,11 two error terms, $C$ and $D$, are introduced. These errors arise from the finite packet sizes that are being dealt with. For example, any packet may experience an excess delay as it is forwarded due to the size of the packets in the same queue and due to inaccuracies in scheduling from packets (of a possibly different size) in other queues bound for the output link. These terms are derived for weighted fair queueing schedulers in Parekh’s seminal work [8], [9]. Consequently, the delay bound now becomes $b/R + C/R + D$. With guaranteed service, however, a limit is imposed on the peak rate $p$ of the flow, which results in a reduction of the delay bound. In addition, the packetization effect of the flow needs to be taken into account by considering the maximum packet size $M$.12 These additional factors result in a more precise bound on the end-to-end queuing delay as follows:

$$Q_{\text{delay\_end\_end}} = \frac{(b - M)(p - r)}{R(p - r)} + \frac{(M + C_{\text{tot}})}{R} + D_{\text{tot}},$$

$$Q_{\text{delay\_end\_end}} = \frac{(M + C_{\text{tot}})}{R} + D_{\text{tot}}, \quad \text{(case } p > R \geq r\text{)} \quad (1)$$

$$Q_{\text{delay\_end\_end}} = \frac{(M + C_{\text{tot}})}{R} + D_{\text{tot}}, \quad \text{(case } R \geq p \geq r\text{).} \quad (2)$$

The composed terms $C_{\text{tot}}$ and $D_{\text{tot}}$ represent the summation of the $C$ and $D$ error terms, respectively, for each router along the end-to-end data path. In (1), there are three delay terms, made from the contributions of the burst of packets $b$ (the bucket depth) sent at the peak rate $p$ and serviced at the output link rate $R$, plus the sum over all hops of errors introduced at each hop due to a packet size worth of fluid flow approximation, plus a third term arising from cross-traffic scheduling approximation contributions. In (2), the first term is absent, since the link rate is greater than the peak, so there are no packets queued from this flow itself.

For a router to invoke guaranteed service for a specific data flow, it needs to be informed of the traffic characteristics of the flow (tspec) along with the reservation characteristics (rspec). Furthermore, to enable the router to calculate sufficient local resources to guarantee a lossless service requires the terms $C_{\text{sum}}$ and $D_{\text{sum}}$, which represent the summation of the $C$ and $D$ error terms, respectively, for each router along the path since the last reshaping point (see below). The tspec parameters include:

- $p$ peak rate of flow (bytes/second);
- $b$ bucket depth (bytes);
- $r$ token bucket rate (bytes/second);
- $m$ minimum policed unit (bytes)13;
- $M$ maximum datagram size (bytes).

The rspec parameters include:

- $R$ bandwidth, i.e., service rate (bytes/second);
- $S$ slack term (ms) (see Section IV-F).

Guaranteed service traffic must be policed at the network access points to ensure conformance to the tspec so that traffic does not interfere with other flows and cause them to miss their contract. We discuss this more below.

In addition to the policing of data flows at the edge of the network, guaranteed service also requires the reshaping of traffic to the token bucket of the reserved tspec at certain points on the distribution tree. Any packets failing the reshaping are treated as best effort and marked accordingly.

---

11 Among other things, the router’s approximation must take account of the medium-dependent behavior of the link layer of the data forwarding path.

12 While the IP in principle permits a wide range of packet sizes, in practice, the range supported makes stating this upper limit practical and realistic.

13 Policing will treat any IP datagram less than size $m$ as being of size $m$. 
if such a facility is available. Reshaping\textsuperscript{14} must be applied at any point where it is possible for a data flow to exceed the reserved tspec, even when all senders associated with the data flow conform to their individual tspecs. Such an occurrence is possible in the following two cases.

1) At branch points in the distribution tree where the reserved tspecs of the outgoing branches are not the same. In this case, the reserved tspec of the incoming branch is given by the “maximum”\textsuperscript{15} of the reserved tspecs on each of the outgoing branches. Consequently, some of the outgoing branches will have a reserved tspec that is less than the reserved tspec of the incoming branch. So it is possible that in the absence of reshaping, traffic that conforms to the tspec of the incoming branch might not conform when routed through to an outgoing branch with a smaller reserved tspec. As a result, reshaping must be performed at each such outgoing branch to ensure that the traffic is within this smaller reserved tspec.

2) At merge points in the distribution tree for sources sharing the same reservation, since in these cases, the sum of the tspecs relating to the incoming branches will be greater than the tspec reserved on the outgoing branch. Consequently, when multiple incoming branches are each simultaneously active with traffic conforming to their respective tspecs, it is possible that when this traffic is merged onto the outgoing branch, it will violate the reserved tspec of the outgoing branch. Hence, reshaping to the reserved tspec of the outgoing branch is necessary.

This reshaping will necessarily incur an additional delay (essentially, smoothing a collection of peaks over some troughs of traffic flow must entail slowing down early packets, since one obviously cannot speed up later packets).

1) Killer Reservation Problems: When merging heterogeneous reservation requests from receivers onto the tree flowing from the same source, there is an additional problem known as killer reservation, which manifests itself in two ways.

1) A large reservation made subsequent to an existing smaller reservation may fail. If this is the case, a naive implementation will cause the entire reservation to fail. The solution to this is to introduce extra states into the reservation protocol such that subsequent failures do not break existing reservations.

2) A receiver may continually attempt to make a large reservation, retrying quickly after every failure. This may continually block a smaller reservation request that might otherwise succeed. Again, a merge point might keep state concerning recent failed reservations and favor new ones that are more likely to succeed over retries for ones that have recently failed.

2) Example of Use/Requirement for Guaranteed Service: There are a number of applications in the military and commercial worlds that have hard delay-bound requirements. For example, the distributed interactive simulation program has a scenario with 100,000 participants in an on-line war game simulation, where the applications send messages to each other that represent events between objects in the real world. Participants should see progress (missiles hitting tanks) in an ordered and timely fashion. A somewhat difference scenario, but with remarkably similar network guarantee requirements, is that of share-dealer networks. Here, share-price advertisements are multicast to dealer terminals with hard delay bounds on delivery delays (and rates) and delay bounds on the response times, since the price in the real world is varying, possibly very rapidly. The requirement is that a bid to buy at a price does not encounter an offer to sell that is significantly out of date.

C. Policing and Conformance of Controlled-Load and Guaranteed Services

Routers implementing controlled-load and guaranteed services must check for conformance of data flows to their appropriate reserved tspecs. This is known as policing. Any nonconforming data flows must not be allowed to affect the QoS offered to conforming data flows or to unfairly affect the handling of best effort traffic. Within these constraints, the router should attempt to forward as many of the packets of the nonconforming data flow as possible. This might be done by dividing the packets into conforming and nonconforming groups and forwarding the nonconforming group on a best effort basis. Alternatively, the router may choose to degrade the QoS of all packets of a nonconforming data flow equally.

The usual enforcement policy is to forward nonconforming packets as best effort datagrams.\textsuperscript{16}

An additional requirement for policing over that of meeting the global traffic contract is that there are possible consequences for pricing if excess traffic is not seen to receive a lesser service guarantee.

D. Instantiating Integrated Services on Given Link Technology

Assuming that a backbone network is implemented as a set of routers connected together by point-to-point circuits, integrated services must be implemented by putting in place the appropriate queueing strategies in the routers. Typically, theory tells us that weighted fair queueing (or hierarchical round-robin service) will provide at least a base line for

\textsuperscript{14}Traffic conforming to a leaky-bucket specification still has some degree of freedom to take different shapes within the envelope. Shaping can improve the way such traffic mixes and can improve its buffering requirements.

\textsuperscript{15}Maximum according to rules defined in [3].

\textsuperscript{16}Action with regard to nonconforming datagrams should be configurable to allow for situations such as traffic sharing, where the preferred action might be to discard nonconforming datagrams. This configuration requirement also applies to reshaping. If and when a marking facility (e.g., a bit in the IPv6 header to indicate that a packet has exceeded its flow parameters) becomes available, these nonconforming datagrams should be marked to ensure that they are treated as best effort datagrams at all subsequent routers.
implementation. In fact, for controlled load, simple priority queueing schemes may suffice. There are a number of other service disciplines being researched.

Where routers are interconnected by other types of links, particularly shared media (LAN’s, satellite channels, etc.), or switches, then the interconnect technology must be controlled as well. In the standards work, specifications are emerging for mapping the guaranteed service and controlled load to run over Token Ring, switched multimegabit data service, frame relay, and a variety of transfer capabilities and QoS classes on switched ATM networks.

For nondeterministic (but popular) subnetworks such as Ethernet, the technology must be enhanced somehow. This is a matter for current research.

IV. RESOURCE RESERVATION PROTOCOL

The RSVP [3] was designed to enable the senders, receivers, and routers of communication sessions (either multicast or unicast) to communicate with each other in order to set up the necessary router state to support the services described in Sections III-A and III-B. It is worth noting that RSVP is not the only IP reservation protocol that has been designed for this purpose. Others include ST-II [11] and ST-II+ [5], which incidentally contain some interesting architectural differences to RSVP, such as the use of hard-state and sender-initiated reservations rather than soft-state and receiver-initiated reservations, as in RSVP. For the rest of this paper, however, the only reservation protocol we consider is RSVP since currently, this has the most industry support. For further discussion on the mentioned alternatives, the interested reader can refer to [7].

RSVP is a novel signaling protocol in at least three ways.

1) It accommodates multicast, not just point-to-multipoint (one-to-many) reservations. To this end, the receiver-driven request model permits heterogeneity in principle, and the filter mechanism allows for calls that reserve resources efficiently for the aggregate traffic flow (e.g., for audio conferencing).

2) It uses soft state, which means that it is tolerant of temporary loss of function without entailing fate sharing between the end systems and the network nodes. This means that QoS routing can be deployed separately (in more than one way).

3) RSVP is quite straightforward in packet format and operations and so is relatively low cost in terms of implementation in end systems and routers. One thing that RSVP is not is a routing protocol. RSVP does not support QoS-dependent routing itself (in other words, such routing is independent of RSVP and could precede or follow reservations).

RSVP identifies a communication session by the combination of destination address, transport layer protocol type, and destination port number. It is important to note that each RSVP operation only applies to packets of a particular session, and as such, every RSVP message must include details of the session to which it applies. For the remainder of this tutorial, it will be assumed that any discussion is for a single session only. In addition, although RSVP is applicable to both unicast and multicast sessions, we concentrate on the more complicated multicast case. We do not discuss RSVP policy issues such as authorization, authentication, and questions relating to charging for reservations.

RSVP is not a routing protocol but a signaling protocol; it is merely used to reserve resources along the existing route set up by whichever underlying routing protocol is in place. Fig. 1 shows an example of RSVP for a multicast session involving one sender (S1) and three receivers (RCV1–RCV3). The primary messages used by RSVP are the Path message, which originates from the traffic sender, and the Resv message, which originates from the traffic receivers. The primary roles of the Path message are first, to install reverse routing state in each router along the path, and second, to provide receivers with information about the characteristics of the sender traffic and end-to-end path so
that they can make appropriate reservation requests. The primary role of the Resv message is to carry reservation requests to the routers along the distribution tree between receivers and senders. Returning now to Fig. 1, as soon as S1 has data to send, it begins periodically forwarding RSVP Path messages to the next hop, R1, and down the distribution tree. RSVP messages can be transported “raw” within IP datagrams using protocol number 46, although hosts without this raw input–output capability may first encapsulate the RSVP messages within a UDP header.

A. Reservation Styles and Merging

Associated with each reservation made at a router’s interface is a filterspec describing the packets to which the reservation applies, along with an effective flowspec. Both the filterspec and effective flowspec are obtained from a merging process applied to selected Resv messages arriving on the router’s interface. The rules for merging are dependent upon the reservation style of each Resv message, as described below. In addition, the router calculates the filterspec and flowspec of Resv messages to be sent to the previous hop(s) upstream by applying style-dependent merging of stored reservation state. Any changes to stored reservation state that result in changes to the Resv messages to be sent upstream will cause an updated Resv message to be sent upstream immediately. Otherwise, Resv messages are created based on stored reservation state and sent upstream periodically. As for Path state, all reservation state is stored in routers using soft state and consequently relies on periodic refreshes via Resv messages to prevent state timeout. In addition, just as a PathTear message exists explicitly to tear down Path state, a ResvTear message exists explicitly to tear down reservation state. Currently, three reservation styles are permissible, as described below and illustrated in Figs. 2–6, where the convention style(filterspec{flowspec}) is used to summarize the requests made by the Resv messages. It should be noted that the merging processes described below apply only to packets of the same session (this is true of any RSVP process). Also, merging can only occur between messages with the same reservation style. Details of the reservation styles are as follows, where it is assumed that each interface I in Figs. 2–4 is routable to each of the router’s other interfaces.

1) Fixed Filter (FF) (Distinct Reservation and Explicit Sender Selection): The filterspec of each FF reservation installed at an interface consists of a single sender only. The effective flowspec of the reservation installed is the
maximum of all FF reservation requests received on that interface for that particular sender. The flowspec of the FF Resv message unicast to the previous hop of a particular sender is given by the maximum flowspec of all reservations installed in the router for that particular sender.

2) Wildcard Filter (WF) (Shared Reservation and Wildcard Sender Selection): The filterspec of each WF reservation installed at an interface is a wildcard and matches on any sender from upstream. The effective flowspec installed is the maximum from all WF reservation requests received on that particular interface. The flowspec of each WF Resv message unicast to a previous hop upstream is given by the maximum flowspec of all WF reservations installed in the router.

3) Shared Explicit (SE) (Shared Reservation and Explicit Sender Selection): The filterspec of each SE reservation installed at an interface contains a specific set of senders from upstream and is obtained by taking the union of the individual filterspecs from each SE reservation request received on that interface. The effective flowspec installed is the maximum from all SE reservation requests received on that particular interface. The filterspec of an SE Resv message unicast out of an interface to a previous hop upstream is the union of all senders whose previous hop is via that interface and who are contained in the filterspec of at least one SE reservation in the router. Likewise, the flowspec of this SE Resv message is given by the maximum flowspec of all SE reservations whose filterspec contain at least one sender whose previous hop is via that interface.

SE and WF styles are useful for conferencing applications where only one sender is likely to be active at once, in which case reservation requests for, say, twice the sender bandwidth could be reserved in order to allow an amount of overspeaking.

Although RSVP is unaware of to which service (controlled-load or guaranteed) reservations refer, RSVP is able to identify those points in the distribution tree that require reshaping in the event that the reservations are for guaranteed service, as described in Section III-B. Consequently, at all such points, RSVP informs the traffic-control mechanisms within the appropriate router accordingly, although such action will only result in reshaping if the reservation is actually for guaranteed service.

B. Path Messages

Each Path message includes the following information:

- **phop**, the address of the last RSVP-capable node to forward this Path message; this address is updated at every RSVP-capable router along the path;
- **sender template**, a filter specification identifying the sender, which contains the IP address of the sender and optionally the sender port (in the case of IPv6, a flow label may be used in place of the sender port);
- **sender tspec**, defining the sender traffic characteristics;
- an optional **adspec** containing “one pass with advertising” (OPWA) information (see Sections IV-D and E), which is updated at every RSVP-capable router along the path to attain end-to-end significance before being presented to receivers to enable them to calculate the level of resources that must be reserved to obtain a given end-to-end QoS.

C. Processing and Propagation of Path Messages by Network Routers

Each intermediate RSVP-capable router along the distribution tree intercepts Path messages and checks them for validity. If an error is detected, then the router will drop the Path message and send a PathErr message upstream to inform the sender, who can then take appropriate action.

Assuming that the Path message is valid, the router does the following:

- Update the Path state entry for the sender identified by the sender template. If no Path state exists, then it is created. Path state includes the sender tspec, the address, the phop of the previous hop upstream router, and optionally an adspec. The phop address needs to be updated at every RSVP-capable router along the path to attain end-to-end significance before being presented to receivers to enable them to calculate the level of resources that must be reserved to obtain a given end-to-end QoS.
Fig. 5. \( R = 2.5 \text{ Mb/s}, S_1 = 0 \). Reservation request denied.

Fig. 6. \( R_1 = 3 \text{ Mb/s}, S_1 > 0, R_2 = 2 \text{ Mb/s}, S_2 < S_1 \). Reservation accepted.

to be stored in order to route Resv messages in the reverse direction up the tree. The sender tspec provides a ceiling to clip any inadvertently overspecified tspecs subsequently received in Resv messages.

- Set cleanup timer equal to cleanup timeout interval and restart timer.

- Associated with each Path state entry is a cleanup timer, the expiration of which triggers deletion of the Path state. Expiration of the timer will be prevented if a Path message for the entry is received at least once every cleanup timeout interval. This is the so-called RSVP soft-state mechanism and ensures that state automatically times out if routing changes while subsequent Path messages install state along the new routing path. In this way, the use of soft rather than hard state helps to maintain much of the robustness of the initial Internet design concepts, whereby all flow-related state was restricted to the end systems [4].

The router is also responsible for generating Path messages based on the stored Path state and forwarding them down the routing tree, making sure that for each outgoing interface, the adspec (see Section IV-D) and phop objects are updated accordingly. Path messages will be generated and forwarded whenever RSVP detects any changes to stored Path state or is informed by the underlying routing protocol of a change in the set of outgoing interfaces in the data-forwarding path. Otherwise, a Path message for each specific Path state entry is created and forwarded every refresh period timeout interval in order to refresh downstream Path state.

The refresh period timeout interval is several times smaller than the cleanup timeout interval, so that occasional lost Path messages can be tolerated without triggering unnecessary deletion of Path state. It is still recommended, however, that a minimum network bandwidth be configured for RSVP messages to protect them from congestion losses.

Although all Path state would eventually time out in the absence of any refreshes via Path messages, RSVP includes an additional message, PathTear, to expedite the process. PathTear messages travel across the same path as Path messages and are used to explicitly tear down Path state. PathTear messages are generated whenever a Path state entry is deleted, and so a PathTear message generated by a sender will result in the deletion of all downstream Path state for that sender. It is recommended that senders do this as soon as they leave the communications session. Also, the deletion of any Path state entry triggers the deletion of any dependent reservation state (see Section IV-A).

D. Adspec

The adspec is an optional object that the sender may include in its generated Path messages in order to advertise to receivers the characteristics of the end-to-end communications path. This information can be used by receivers to determine the level of reservation required in order to achieve their desired end-to-end QoS. The adspec consists of a message header, a default general parameters fragment, and either guaranteed service fragment or controlled-load service fragment. Omission of either the guaranteed or controlled-load service fragment is an indication to receivers that the omitted service is not available. This feature can be used in a multicast session to force all receivers to select the same service. At present, RSVP does not accommodate heterogeneity of services between receivers within a given multicast session, although, within the same
service model, the parameters may differ for receivers in the same session, so the core objective of supporting heterogeneity is mainly met.

The default general parameters fragment includes the following fields, which are updated at each RSVP-capable router along the path in order to present end-to-end values to the receivers.

- **Minimum path latency** (summation of individual link latencies): This parameter represents the end-to-end latency in the absence of any queuing delay. In the case of guaranteed service, receivers can add this value to the bounded end-to-end queuing delay to obtain the overall bounded end-to-end delay.

- **Path bandwidth**: The minimum of individual link bandwidths along the path.

- **Global break bit**: This bit is cleared when the adspec is created by the sender. Encountering any routers that do not support RSVP will result in this bit’s being set to one in order to inform the receiver that the adspec may be invalid.

- **IS hop count**: This count is incremented by one at every RSVP/IS-capable router along the path.

- **PathMTU**: The path maximum transmission unit (minimum of MTU’s of individual links along the path).

The correct functioning of IETF IS requires that packets of a data flow are never fragmented. This also means that the value of $M$ in the tspec of a reservation request must never exceed the MTU of any link to which the reservation request applies. A receiver can ensure that this requirement is met by setting the value of $M$ in the tspec of its reservation request to the minimum of the PathMTU values received in “relevant” Path messages. The value of $M$ in each generated reservation request may be further reduced on the way to each sender if merging of Resv messages occurs (see Section IV-B). The minimum value of $M$ from the tspec of each Resv message received by the sender should then be used by the sending application as the upper limit on the size of packets to receive special QoS. In this way, fragmentation of these packets will never occur. It is worth noting that [13] recommends that the value of $M$ in the sender tspec, which has played no part in the above MTU negotiation process, should be set equal to the maximum packet size that the sender is capable of generating rather than what it is currently sending.

The guaranteed service fragment of the adspec includes the following fields, which are updated at each RSVP-capable router along the path to present end-to-end values to the receivers.

- $C_{tot}$: end-to-end composed value for $C$.
- $D_{tot}$: end-to-end composed value for $D$.
- $C_{sum}$: composed value for $C$ since last reshaping point.
- $D_{sum}$: composed value for $D$, since last reshaping point ($C_{sum}$ and $D_{sum}$ values are used by reshaping processes at certain points along the distribution tree).
- **Guaranteed service break bit**: cleared when the adspec is created by the sender; encountering any routers that do support RSVP/IS but do not support guaranteed service will result in this bit’s being set to one in order to inform the receiver that the adspec may be invalid and the service cannot be guaranteed.

- **Guaranteed service general parameters headers/values**: these are optional, but if any are included, then each one overrides the corresponding value given in the default general parameters fragment as far as a receiver wishing to make a guaranteed service reservation is concerned. These override parameters could, for example, be added by routers along the path that have certain service-specific requirements. For example, a router may have been configured by network management so that guaranteed service reservations can only take up a certain amount $B_{gs}$ of the outgoing link bandwidth. Consequently, if the default path bandwidth value in the adspec to be sent out of this interface is greater than $B_{gs}$, then a guaranteed service-specific path bandwidth header and value equal to $B_{gs}$ may be included in the adspec. As for default general parameters, any service-specific general parameters must be updated at each RSVP hop.

The controlled-load service fragment of the adspec includes the following fields, which are updated at each RSVP-capable router along the path in order to present end-to-end values to the receivers.

- **Controlled-load service break bit**: This bit is cleared when the adspec is created by the sender. Encountering any routers that do support RSVP/IS but do not support controlled load will result in this bit’s being set to one in order to inform the receiver that the adspec may be invalid and the service cannot be guaranteed.

- **Controlled-load service general parameters headers/values**: As for the guaranteed service fragment, override service-specific general parameters may be added to the controlled-load service fragment.

E. Making a Reservation Using OPWA

OPWA refers to the reservation model for the case where the sender includes an adspec in its Path messages to enable the receiver to determine the end-to-end service that will result from a given reservation request. If the sender omits the adspec from its Path messages, then the reservation model is referred to simply as one pass, in which case there

---

22 It might be possible to devise a scheme to support QoS for fragmented traffic, but the key problem of how loss of fragment results in loss of overall datagram is hard to work around.

23 MTU discovery can be employed to avoid this.

24 In cases where the last hop to a sender is a shared medium LAN, the sender may receive Resv messages across the same interface from multiple next hop routers.

25 Composed, as explained above, from the sum of each router/hop’s estimate of the error.
is no easy way for the receiver to determine the resulting end-to-end service. The objective of this aspect of the RSVP reservation model, both for one pass and one pass with advertising, is to minimize the latency (in terms of number of handshakes between senders and recipients) before a reservation is in place. Here, we consider the OPWA case.

Let us assume that the sender omits the controlled-load service data fragment from the adspec, thereby restricting each receiver to reservation of guaranteed service only. Upon receiving Path messages, the receiver extracts the following parameters from the sender tspec contained therein: \( r, b, p, \) and \( \bar{m} \). In addition, the following are extracted from the adspec: minimum path latency, \( C_{\text{path}} \), \( D_{\text{path}} \), PathMTU, and Path bandwidth.

In a way similar to incremental route calculation, OPWA permits incremental accumulation of the delay for a reservation. The required bound on end-to-end queuing delay, \( \text{qdelreq} \), is now calculated by subtracting the minimum path latency from the value of end-to-end delay required by the receiver’s application. Typically, the receiver would then perform an initial check by evaluating (2) for \( R \) equal to the peak rate \( p \). If the resultant delay was greater than or equal to qdelreq, then (2) would be used for calculation of the minimum value of \( R \) necessary to satisfy qdelreq. Otherwise, (1) would be used for this purpose. This minimum value of \( R \) is then obtained by inserting qdelreq into either (1) or (2) along with \( M \) (given by PathMTU), \( C_{\text{path}}, D_{\text{path}}, r, b, \) and \( p \), as appropriate. If the obtained value of \( R \) exceeds the Path bandwidth value as obtained from the adspec of the received path message, then it must be reduced accordingly. The receiver can now create a reservation specification, rspec, comprising 1) the calculated value \( R \) of bandwidth to be reserved in each router and 2) a slack term that is initialized to zero.\(^{26}\) The rspec can now be used in the creation of a Resv message, which also includes the following:

- an indication of the reservation style, which can be FF, SE, or WF (see Section IV-A);
- a filter specification, filterspec (omitted for the case of WF reservation style), used to identify that the sender(s) and the format is identical to that of the sender template in a Path message;
- a flow specification, flowspec, comprising the rspec and a traffic specification, tspec. Tspec is usually set equal to the sender tspec, except \( M \) will be given by the PathMTU obtained from the received adspec;
- optionally, a reservation confirm object, ResvConf, containing the IP address of the receiver. If present, this object indicates that the node accepting this reservation request at which propagation of the message up the distribution tree finishes should return a ResvConf message to the receiver to indicate that there is a high

\(^{26}\)In some cases, even with \( R \) set to the minimum permissible value of \( r \), the resultant end-to-end queuing delay as given by (1) and (2) will still be less than qdelreq, in which case the difference can be represented in a nonzero slack term. In addition, there are other scenarios explained in Section IV-F in which the slack term may not be initialized to zero.

probability\(^{27}\) that the end-to-end reservation has been successfully installed.

The Resv message is now sent to the previous hop upstream as obtained from the stored Path state. Upon reaching the next upstream router, the Resv messages can be merged with other Resv messages arriving on the same interface according to certain rules (as described in Section IV-A) to obtain an effective flowspec and filterspec. The following action is then taken:

- The effective flowspec is passed to the traffic-control module within the router, which applies both admission control and policy control to determine whether the reservation can be accepted. Admission control is concerned solely about whether enough capacity exists to satisfy the request, while policy control also takes into account any additional factors that need to be considered (e.g., certain policies may limit a user’s reserved bandwidth even if spare bandwidth exists).
- If the reservation attempt is denied, then any existing reservations are left unaltered and the router must send a ResvErr message downstream.
- If the reservation request is accepted, then reservation state is set up in accordance with the effective flowspec and filterspec, as described in Section IV-A. In accepting the request, it may be permissible to alter the rspec associated with the reservation from \( (R_{\text{res}}, S_{\text{res}}) \) to \( (R_{\text{res}}, S_{\text{conf}}) \) in accordance with the rules described in Section IV-F. The resultant reservation may then be merged with other reservations in accordance with the rules in Section IV-A to obtain a new Resv message that is sent to the next router upstream, the address of which is obtained from the stored Path state.

\( F. \text{ Slack Term} \)

When a receiver generates an rspec for a Resv message to be sent for a guaranteed service reservation request, it must include a slack term \( S(m_S) \), as well as the amount of bandwidth \( R \) to be installed in each router along the path. \( S \) represents the amount by which the end-to-end delay bound will be below the end-to-end delay required by the application, assuming each router along the path reserves \( R \) bandwidth according to the guaranteed service fluid approximation. Inclusion of a nonzero slack term offers the individual routers greater flexibility in making their local reservations. In certain circumstances, this greater flexibility could increase the chance of an end-to-end reservation’s being successful. Some routers have deadline-based schedulers that decouple rate and delay guarantees. Such a scheduler may sometimes be unable to meet its deadline requirement for guaranteed service, in which case it might still be able to accept the reservation, provided the slack term is at least as large as the excess delay. The excess delay would then be subtracted from the slack term before unicasting the Resv message to the previous hop upstream.

\(^{27}\)In practice, there are certain scenarios in which a ResvConf message might be received by a receiver, only for the request to be rejected shortly afterwards.
Similarly, a rate-based scheduler might be able to admit a reservation request by reserving less than the requested bandwidth and unicasting the reduced reservation request to a previous hop upstream, provided it could extract enough slack. Any router using available slack to reduce its reservation must conform to the rules in (3) to ensure that the end-to-end delay bound remains satisfied

\[ S_{out} + \frac{b}{R_{out}} + C_{tot} \leq S_{in} + \frac{b}{R_{in}} + C_{tot}, \]

\[ r \leq R_{out} \leq R_{in}, \]

where \( C_{tot} \) is the cumulative sum of the error terms \( C \) for all the routers that are upstream of and including the current element \( i \), \((R_{in}, S_{in})\) is the reservation request received by router \( i \), and \((R_{out}, S_{out})\) is the modified reservation request unicasted to the previous hop router upstream.

An example of how intelligent use of the slack term can increase the probability of an end-to-end reservation request’s being accepted is illustrated in Figs. 5 and 6. Suppose the token bucket rate (as defined earlier) of the data to be sent is 1.5 Mb/s, and the receiver has calculated from the tspec and adspec parameters in received Path messages that the desired end-to-end delay can be achieved by a reservation of \((R = 2.5 \text{ Mb/s}, S = 0)\), which is then requested in Fig. 5. Because R3 has only 2 Mb/s of unused bandwidth and there is no slack available, however, the reservation is denied. In Fig. 6, the reservation is increased to \( R = 3 \text{ Mb/s} \), and the amount by which such a reservation would be within the required delay bound is put in the slack term \((S > 0)\). R5 and R6 reserve the requested 3 Mb/s. R3 can only reserve a value of 2 Mb/s, which, if used as the new reservation value in the propagated Resv message, will cause an increase in the end-to-end delay bound. R3 can calculate this increase \( d_i \), and if it is less than the value of the slack term \( S1 \) in the received Resv message, then the request can be accepted and a reservation of 2 Mb/s installed in R3. R3 will then set the tspec in the Resv message to \((R = 2 \text{ Mb/s}, S2 = S1 - d_i)\) before unicasting it to the next hop upstream, which results in R2 and R1’s also reserving 2 Mb/s. The end-to-end delay bound of the reserved path is now no greater than for a reservation of 2.5 Mb/s in every router if that were possible.

Without RSVP, a fall-back service of best effort is still available from the unused capacity. In the near future, some research needs to be carried out in a number of areas.

1) Accounting and billing needs to be integrated into the model in a scalable way.

2) Aggregation of nonspecifically related reservations (or flows) would be useful. In the same way that ATM provides virtual paths as well as virtual circuits, we might like to build virtual private Internets using, for example, the address aggregation mechanism of classless interdomain routing, to be used within a reservation (the extension has been proposed in the RSVP working group to allow the generalized port to be accompanied by masks, in the same way that routing protocols distribute updates with masks).

3) Authentication of users of RSVP is clearly essential if we are to incur bills when we use it.

4) The usage accounting model must accommodate mirror servers in some way.

5) Some scheme to permit settlements or something akin to them will need to be evolved to allow deployment of RSVP and integrated services across paths that entail more than a single intranet or commercial Internet service provider.

6) Last, simple experience of using a multiservice network is needed to see which pieces of this complex system are really used frequently, since it is not at all clear that the entire edifice is either necessary or sufficient.

The current status of RSVP in the IETF is that it is undergoing standardization as of the writing of this paper.

ACKNOWLEDGMENT

The authors wish to acknowledge the generous support of British Telecommunications and the feedback from the anonymous reviewers, as well as the many helpful comments and suggestions from Guest Editor B. Goode.

REFERENCES


Paul P. White received the B.Eng. degree (first class honors) in electronic and electrical engineering from the University of Birmingham, England, in 1989 and the M.Sc. degree (with distinction) in data telecommunications and networks from the University of Salford, England, in 1995. He currently is working toward the Ph.D. degree at University College London in the field of integrated services in the Internet.

He has more than five years of work experience in various fields of hardware/software technology, including telecommunications.

Mr. White is a member of the Institute of Electrical Engineers.

Jon Crowcroft (Senior Member, IEEE) graduated in physics from Trinity College, Cambridge University, England, in 1979. He received the M.Sc. degree in computing in 1981 and the Ph.D. degree in 1993 from London University, England.

He is a Professor of networked systems in the Department of Computer Science, University College London, where he is responsible for a number of European- and U.S.-funded research projects in multimedia communications. He has been working in these areas for more than 15 years. He is the coauthor (with M. Handley) of WWW: Beneath the Surf (London: UCL Press, 1995) and the author of Open Distributed Systems (Norwood, MA: Artech House, 1996).

Dr. Crowcroft is a member of the Association of Computing Machinery (ACM), the British Computer Society, and the Institute of Electrical Engineers. He is a member of the Internet Architecture Board and General Chair for the ACM Special Interest Group on Data Communication. He also is on the Editorial Board of IEEE/ACM TRANSACTIONS ON NETWORKING.