A PROPOSAL FOR RSVP OVER DIFFERENTIATED SERVICES NETWORKS

Yiqun Shi

A thesis
in
The Department
of
Computer Science

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Computer Science
Concordia University
Montréal, Québec, Canada

September 2003

Your file Votre référence
ISBN: 0-612-83919-2
Our file Notre référence
ISBN: 0-612-83919-2

Abstract

A Proposal for RSVP over Differentiated Services Networks

Yiqun Shi

Classical IP routing provides only a "best effort" service, which makes routing simple, but which cannot meet the Quality of Service (QoS) requirements of streaming applications such as voice and video. End users negotiate their needs with the network using the Resource Reservation Protocol (RSVP). One model for the management of the network is Integrated Services (IntServ), which provides for per-flow negotiation, but does not scale to many connections. It was designed to work together with RSVP, and is well-suited to managing connections in access networks.

For backbone networks, where there may be hundreds or thousands of connections, Differentiated Services (DiffServ) was introduced. It assigns flows to classes, and uses only the classes to assign priority for packet handling in the routers. However, DiffServ routers do not understand RSVP messages. Independent DiffServ domains may use different IntServ-to-Diffserv mappings.

Within a DiffServ domain, path assignment and admission control are handled by a "Bandwidth Broker" (BB). However, the BB concept does not extend well to access networks. Inter-BB communication can be achieved using a proposed protocol, SIBBS.

By making each BB aware of RSVP messages, and extending SIBBS to carry RSVP messages between an ingress router and its associated BB, between adjacent BBs, and between the "last" BB on a path and the associated egress router, we develop an architecture that permits retaining the advantages of IntServ in the access networks, retaining the advantages of DiffServ in the backbone networks, and minimizing the changes that have to be made to existing DiffServ networks. The set of DiffServ domains appear to the end user as a series of RSVP nodes, and the routers inside the DiffServ domains need not be aware of RSVP messages at all.

In this way, the user's needs are met, without impacting the scalability of the backbone network.

Acknowledgments

Sincere thanks to my supervisor, Dr. J.W. Atwood, for his great help in supervising my thesis. His enthusiasm toward the technology and serious attitude to academic research deeply impressed me during the pursuit of my degree.

Also, I would like to thank my wife for her great support in taking care of my son and my life during the long time of research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

This thesis paper is motivated by the demands for the deployment of Quality of Service (QoS) over IP networks. The current QoS End-to-End (E2E) architecture is well constructed in the phase of prototype. Each prototype is feasible to specific circumstance respectively. However, it still lacks some overall mechanism to fulfill the E2E requests in order to provide Quality of Service assurance to the ISP and end user. The IETF has defined two types of QoS models on different purposes: Integrated Services (IntServ) and Differentiated Services (DiffServ). IntServ architecture works well in providing a guarantee to the per-application data flow. DiffServ architecture, on the other hand, focuses on the data classification from multi-applications. As a result, IntServ is suitable for the end user to provide QoS to the data from individual applications, whereas DiffServ is good in resource provision in the network backbone by means of differentiating data traffic according to the class assigned to each data packet. However, neither architecture is able to handle the QoS request all the way from end to end across a large network such as the Internet. The suggested idea from the IETF is to combine both architectures to provide E2E QoS. The related prototype of this idea has been announced in RFC 2998 [3].

One of the key questions in the prototype proposed in RFC 2998 is how to guarantee Integrated Service in a Differentiated Service region without bringing too much impact to the currently running IP network. As an attempt to address this problem, this thesis paper proposes a new E2E QoS architecture by taking advantage of the prevailing QoS protocols and mechanisms to meet the demand of QoS requests in large networks.

1.2 The QoS benefits

The Internet is increasingly relied upon for doing business, and the expectations for quality assurances are the same as for a private, controlled network. The Internet is being used to power both intra-nets within the enterprise and extra-nets that enable electronic commerce with business partners. As business is increasingly conducted over the web, it becomes more important that IT managers ensure that the networks deliver appropriate levels of quality. Quality of Service (QoS) technologies provide the tools for IT managers to deliver mission critical business over the public network [15].

Applications are getting more demanding. Mission-critical applications deployed over IP networks increasingly require quality, reliability, and timeliness assurances. In particular, applications that use voice, video streams, or multi-media must be carefully managed within an IP network to preserve their integrity [15].

Managing QoS becomes increasingly difficult because many applications deliver unpredictable bursts of traffic. For example, usage patterns for web, email, and file transfer applications are virtually impossible to predict, yet network managers need to be able to support mission-critical applications even during peak period [15].

QoS technologies allow IT managers and network managers to

- Manage jitter sensitive applications, such as audio and video playbacks
- Manage delay-sensitive traffic, such as real time voice
- Control loss in times of inevitable burst congestion.

Clearly, enterprises and corporations are becoming more attuned to the requirements of mission-critical business conducted over the public network. Increasingly, they are also outsourcing more and more network services to service providers, which allows them to focus more on internal business and reduce capital expenses. QoS technologies will allow service providers to offer more services, such as real-time traffic support, or specific bandwidth allocations, to build into a Service Level Agreement (SLA) portfolio. This creates more revenue generation for service providers, while offering more services to enterprises [15].

Chapter 2

Quality of Service Concepts and Principles

2.1 QoS concept

Simply to say, Quality of Service (QoS) is intended to provide consistent, predictable data delivery service. It requires the network elements, such as application, host and router, to have the ability to ensure its service requirement from top to bottom as well as end to end. QoS does not create extra bandwidth. It effectively manages the network traffic so as to guarantee the services requirement. Several characteristics qualify QoS, including the capability to minimize delivery delay, reduce delay variations, and provide consistent data throughput capacity. The basic idea on realizing the QoS is to break the equal treatment to each network traffic in terms of network resource, reserve network resource for specific traffic flow, or differentiate traffic into different classes.

Five parameters are used to describe a QoS request:

1. Availability
   The user has reliable network connectivity. It is the basic parameter of telecommunication.

2. Latency

   The time between a node of sending a packet and receipt of this packet in another node. In a router, it is the amount of time between receipt and sending of a packet.

3. Jitter

   In packet-switched network, it is a distortion of interval between two adjacent packets compared to their original sending interval. This distortion is particularly damaging to multimedia traffic.

4. Bandwidth

   The theoretical transmission capacity of the network. It is the data rate that the system can transfer. As the maximum theoretical bandwidth is approached, the transmission delay will probably be bigger.

5. Packet loss probability

   The rate of losing packets. It is important for mission critical applications.

2.2 Demand for QoS

The IP and its architecture are based on the simple concept that a datagram with source and destination addresses can traverse a network with IP router independently, without the help of sender and receiver. It does not provide other service except "Best Effort (BE)" to data packets. The advantage of BE is to keep the network simple. This is the reason that IP has achieved great success. The Best Effort service serves the data delivery pretty well in terms of traditional data transferring, such as email, file transfer and web application. When the requirement of real-time data delivery comes up, however, the best effort service inevitably gets into trouble in case of traffic burst even if the bandwidth is wide enough. Take the use of IP telephony as an example. If there are 50Kbps bandwidth available for 10 sessions of IP telephone competition, each of which occupies 10Kbps bandwidth, then the result of the competition for Best Effort service is that none of the users will be satisfied. Obviously, some services must be added to current IP architecture as complementary to distinguish the traffic with strict time requirement from those that can tolerate delay, jitter and loss. In the above IP telephony example, we may guarantee 5 sessions by implementing a certain discrimination policy. That is what QoS intends to do.

Chapter 3

Types of QoS and Its Technologies

Basically, QoS is the ability of a network element (e.g., an application, a host or a router) to provide some level of assurance for consistent network data delivery. Two types of QoS, Integrated Service (IntServ) and Differentiated Service (DiffServ), proposed by the IETF, are complementarily designed for use in combination for different network contexts. IntServ is applied to individual application "flows", whereas DiffServ focuses on flow aggregates. One of the major purposes of QoS is to meet the demand of the applications requiring services prior to Best-Effort from end to end system. There are many protocols and mechanisms involved in the end-to-end QoS architecture design. We only choose those typical protocols and mechanisms in the construction of the elements achieving end-to-end quality of service along end-to-end path. Those concerns other than achieving end-to-end QoS are out of the scope of this paper. Some of them can be looked up in the articles listed in the Bibliography.

3.1 Integrated Services/RSVP Architecture

The objective of IntServ is to create a framework of components to facilitate the transmitting of Best-Effort and real-time traffic flow over an IP network [4]. The IETF IntServ framework defines mechanisms and interfaces for providing network Quality of Service control useful for applications that require more predictable network services than traditional IP Best-Effort service. IntServ is proposed to create framework of components to facilitate the transmitting of better-than-Best-Effort and real-time traffic flow over IP network. IntServ defines the model for expressing service type, quantifying the service required and determining the availability of resource required (admission control). The IntServ traffic flows are supposed to have the same destination address and port number. It is a per flow QoS.

To support these capabilities, two things are required:

- **Services**

  Individual network elements (subnets and IP routers) along the path followed by an application's data packets must support mechanisms to control the quality of service delivered to those packets. In the Integrated Services framework, this function is provided by QoS control services, such as Controlled-Load [23] and Guaranteed Service [22].

  Guaranteed Service: This comes as close as possible to emulating a dedicated virtual circuit. It provides firm bounds on end-to-end queueing delays by combining the parameters from the various network elements in a path, in addition to ensuring bandwidth availability according to the TSPEC parameters [22].

  Controlled Load: This is equivalent to "Best-Effort service under unloaded conditions". Hence, it is better than Best-Effort, but can not provide the strictly bounded service that Guaranteed Service promises [23].

- **Convey**

  A way to communicate the application's requirements to network elements along the path and to convey QoS management information between network elements and the application must be provided. It may be provided in a number of ways, but is frequently implemented by a resource reservation signaling protocol, such as RSVP [20].

3.1.1 Guaranteed Service

Guaranteed Service comes as close as possible to emulating a dedicated virtual circuit. It provides firm bounds on E2E delays, and availability of bandwidth. It is the highest level of QoS.

Guaranteed service ensures that datagrams will arrive within the guaranteed delivery time and will not be discarded due to queue overflows, provided the flow's traffic stays within its specified traffic parameters. This service is intended for applications that need a firm guarantee that a datagram will arrive no later than a certain time after it was transmitted by its source. For example, some audio and video "play-back" applications are intolerant of any datagram arriving after their play-back time. Applications that have hard real-time requirements will also require guaranteed service.

This service does not attempt to minimize the jitter (the difference between the minimal and maximal datagram delays); it merely controls the maximal queuing delay. Because the guaranteed delay bound is a firm one, the delay has to be set large enough to cover extremely rare cases of long queuing delays. Several studies have shown that the actual delay for the vast majority of datagrams can be far lower than the guaranteed delay. Therefore, authors of playback applications should note that datagrams will often arrive far earlier than the delivery deadline and will have to be buffered at the receiving system until it is time for the application to process them.

This service represents one extreme end of delay control for networks. Most other services providing delay control provide much weaker assurances about the resulting delays. In order to provide this high level of assurance, guaranteed service is typically only useful if provided by every network element along the path (i.e., by both routers and the links that interconnect the routers). Moreover, effective provision and use of the service requires that the set-up protocol or other mechanism used to request service provides service characterizations to intermediate routers and to the endpoints.

3.1.2 Controlled Load

Controlled Load is similar to BE under unloaded condition. No strictly bounded service is provided. It is better than BE.

A flow receiving controlled-load service at a network element may expect to experience:

- Little or no average packet queuing delay over all timescales significantly larger than the "burst time". The burst time is defined as the time required for the flow's maximum size data burst to be transmitted at the flow's requested transmission rate, where the burst size and rate are given by the flow's TSPEC, as described below.

- Little or no congestion loss over all timescales significantly larger than the "burst time" defined above. In this context, congestion loss includes packet losses due to shortage of any required processing resource, such as buffer space or link bandwidth. Although occasional congestion losses may occur, any substantial sustained loss represents a failure of the admission control algorithm.

3.1.3 RSVP

Resource reSerVation Protocol(RSVP), as a signaling protocol, was originally designed to carry IntServ requests into a network. The application providing Integrated Service may signal the required services to the network elements along the data traffic path by the RSVP protocol. Every network element has a "soft state" set up by RSVP, which stores those Integrated Service requirements. By means of the "soft state", the network element can treat the traffic flows with the service required by application hop by hop. RSVP is the most complex of all the QoS technologies, for applications (hosts) and for network elements (routers and switches). As a result, it also represents the biggest departure from standard "Best-Effort" IP service and provides the highest level of QoS in terms of service guarantees, granularity of resource allocation and detail of feedback to QoS-enabled applications and users [4].

There are several RSVP message types specified. Two of them are fundamental: PATH and RESV [20].

- PATH
  Each RSVP sender host transmits RSVP "PATH" messages downstream along the unicast/multicast routes provided by the routing protocol(s), following the paths of the data. These PATH messages store "path state" in each node along the way. This path state includes at least the unicast IP address of the previous

hop node, which is used to route the RESV messages hop-by-hop in the reverse direction.

PATH messages are sent with the same source and destination addresses as the data, so that they will be routed correctly through non-RSVP clouds.

- RESV

  Each receiver host sends RSVP reservation request (RESV) messages upstream toward the senders. These messages must follow exactly the reverse of the path(s) the data packets will use, upstream to all the sender hosts included in the sender selection. They create and maintain "reservation state" in each node along the path(s). RESV messages must finally be delivered to the sender hosts themselves, so that the hosts can set up appropriate traffic control parameters for the first hop.

  RESV messages are sent hop-by-hop, each RSVP-speaking node forwards a RESV message to the unicast address of a previous RSVP hop.

There is another important concept in RSVP:

- Soft State

  RSVP takes a "soft state" approach to managing the reservation state in routers and hosts. RSVP soft state is created and periodically refreshed by PATH and RESV messages. The state is deleted if no matching refresh messages arrive before the expiration of a "cleanup timeout" interval. State may also be deleted by an explicit "teardown" message. At the expiration of each "refresh timeout" period and after a state change, RSVP scans its state to build and forward PATH and RESV refresh messages to succeeding hops.

  PATH and RESV messages are idempotent. When a route changes, the next PATH message will initialize the path state on the new route, and future RESV messages will establish reservation state there; the state on the now-unused segment of the route will time out. Thus, whether a message is "new" or a "refresh" is determined separately at each node, depending upon the existence of state at that node.

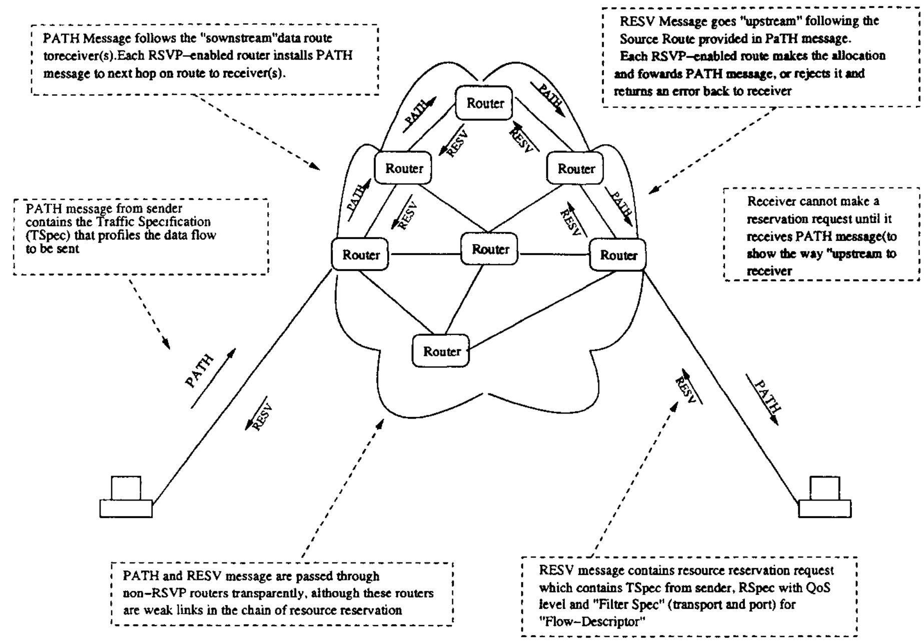

Figure 1: RSVP Signaling

Here is a simplified overview of how RSVP works, as illustrated in Figure 1 [4].

1. Senders characterize outgoing traffic in terms of the upper and lower bounds of bandwidth, delay, and jitter. RSVP sends a PATH message from the sender that contains this traffic specification (TSPEC) information to the (unicast or multicast receiver(s)) destination address.

2. Each RSVP-enabled router along the downstream route establishes a "path-state" that includes the previous source address of the PATH message (i.e., the next hop "upstream" toward the sender).

3. ADSPEC object is refreshed in each router to collect the information of available resources for this traffic flow.

4. To make a resource reservation, receivers send a RESV (reservation request) message "upstream". In addition to the TSPEC, the RESV message includes a request specification (RSPEC) that indicates the type of Integrated Services

required—either Controlled Load or Guaranteed—and a filter specification (filter spec) that characterizes the packets for which the reservation is being made (e.g., the transport protocol and port number). Together, the RSPEC and filter spec represent a flow-descriptor that routers use to identify each reservation (a "flow" or a "session").

5. When each RSVP router along the upstream path receives the RESV message, it uses the admission control process to authenticate the request and allocate the necessary resources. If the request cannot be satisfied (due to lack of resources or authorization failure), the router returns an error to the receiver. If accepted, the router sends the RESV upstream to the next router.

6. When the last router receives the RESV and accepts the request, it sends a confirmation message back to the receiver (note: the "last router" is either closest to the sender or at a reservation merge point for multicast flows).

7. There is an explicit tear-down process for a reservation when sender or receiver ends an RSVP session.

3.1.4 RSVP with IntServ

Since RSVP works closely with IntServ to provide QoS to an application, it is merged into the IntServ/RSVP model (see Figure 2). In IntServ/RSVP model, RSVP is considered as control plane, while IntServ is as traffic plane.

Because RSVP is designed to be used with a variety of QoS control services, and because the QoS control services are designed to be used with a variety of setup mechanisms, a logical separation exists between the two specifications. The RSVP specification does not define the internal format of those RSVP protocol fields, or objects, which are related to invoking QoS control services. Rather, RSVP treats these objects as opaque. The objects can carry different information to meet different application and QoS control service requirements [5].

Similarly, interfaces to the QoS control services are defined in a general format, so that the services can be used with a variety of setup mechanisms.

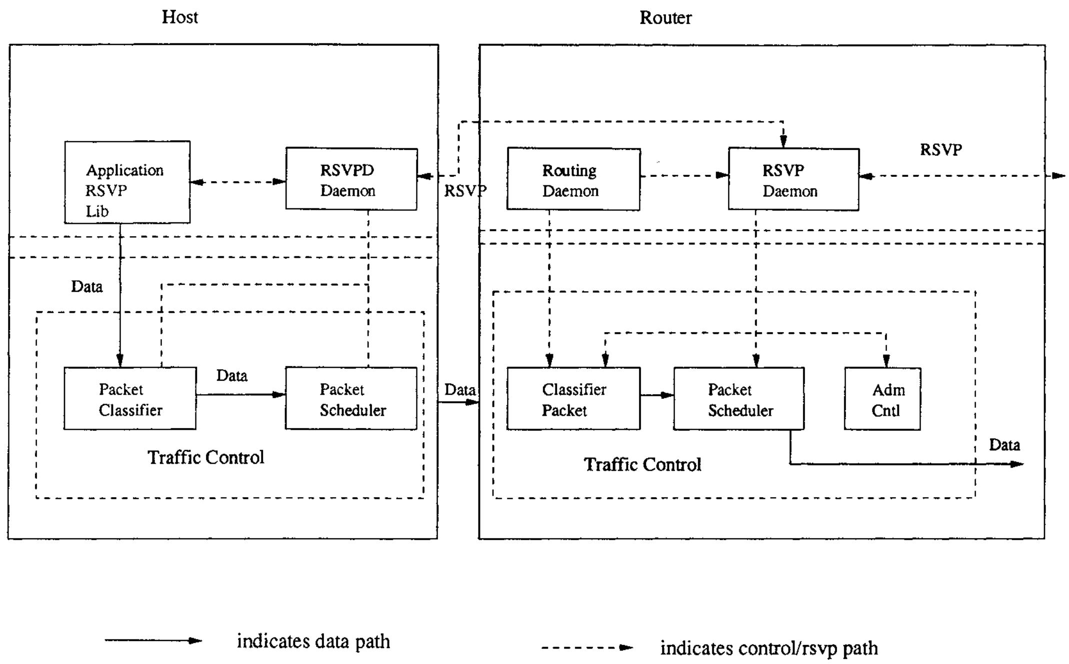

Figure 2: IntServ/RSVP Model

The mechanism of "Use RSVP with IntServ" provides the information required to use RSVP and the integrated service framework's QoS control services together. It defines the usage and contents of three RSVP protocol objects, the FLOWSPEC, ADSPEC, and SENDER_TSPEC, in an environment supporting the Controlled-Load and/or Guaranteed QoS control services.

IntServ/RSVP basic components

IntServ/RSVP basic components include:

1. Admission control
   Performs a local accept/reject decision of resource requirement based on the comparison of request resource and the available resource in the network, along with the additional admission control policy relevant to resource allocation at local router.

2. Packet classifier

   Determines the class of every incoming packet according to its multi fields (MF).

3. Packet scheduler

   Applies traffic management mechanisms, such as queuing (for example, Weighted Fair Queuing or WFQ) to ensure the services, bandwidth and time delay, applied to packet.

4. RSVP

   Convey IntServ request into network.

Data objects in RSVP

Several types of data must be transported between applications and network elements to correctly invoke QoS control services.

These data objects include:

- RSVP FLOWSPEC object

  Information generated by each receiver describing the QoS control service desired, a description of the traffic flow to which the resource reservation should apply, and whatever parameters are required to invoke the service. This information is carried from the receivers to intermediate network elements and the sender(s) by RSVP FLOWSPEC objects. The information being carried in a FLOWSPEC object may change at intermediate points in the network due to reservation merging and other factors.

- SENDER_TSPEC object

  Information generated at each sender describing the data traffic generated by that sender. This information is carried from the sender to intermediate network elements and the receiver(s) by RSVP, but is never modified by intermediate elements within the network. This information is carried in an RSVP SENDER_TSPEC object.

- ADSPEC object

  Information generated or modified within the network and used at the receivers to make reservation decisions. This information might include available services, delay and bandwidth estimates, and operating parameters used by specific QoS control services. This information is collected from network elements and carried toward receivers in RSVP ADSPEC objects. Rather than carrying information from each intermediate node separately to the receivers, the information in the ADSPEC represents a summary, computed as the ADSPEC passes each hop. The size of this summary remains (roughly) constant as the ADSPEC flows through the network, giving good scaling properties.

In addition to the data used to directly invoke QoS control services, RSVP carries authentication, accounting, and policy information needed to manage the use of these services.

From the point of view of RSVP objects, the breakdown is as follows:

- The RSVP SENDER_TSPEC object carries the traffic specification (SENDER TSPEC) generated by each data source within an RSVP session. It is transported unchanged through the network, and delivered to both intermediate nodes and receiving applications. It is conveyed by a PATH message.
- The RSVP ADSPEC object carries information that is generated at either data sources or intermediate network elements, is flowing downstream toward receivers, and may be used and updated inside the network before being delivered to receiving applications. This information includes both 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. It is conveyed by a PATH message.
- The RSVP FLOWSPEC object carries reservation request (Receiver TSPEC and RSPEC) information generated by data receivers. The information in the FLOWSPEC flows upstream toward data sources. It may be used or updated at intermediate network elements before arriving at the sending application. It is conveyed by an RESV message.

The contents of RSVP objects are listed in detail in the Appendix.

Procedure of RSVP signaling

The procedures using RSVP with IntServ are as follows [5]:

An application instance participating in an RSVP session as a data sender registers with RSVP. One piece of information provided by the application instance is the Sender TSPEC describing the traffic the application expects to generate. This information is used to construct an RSVP SENDER_TSPEC object, which is included in RSVP PATH messages generated for the application.

The sending application also constructs an initial RSVP ADSPEC object. This ADSPEC carries information about the QoS control capabilities and requirements of the sending application itself, and forms the starting point for the accumulation of path properties described below. The ADSPEC is added to the RSVP PATH message created at the sender.

The ADSPEC is modified by subsequent network elements as the RSVP PATH message moves from sender to receiver(s). At each network element, the ADSPEC is passed from RSVP to the traffic control module. The traffic control module updates the ADSPEC, which may contain data for several QoS control services, by identifying the services mentioned in the ADSPEC and calling each such service to update its portion of the ADSPEC. If the traffic control module discovers a QoS control service mentioned in the ADSPEC but not implemented by the network element, a flag is set to report this to the receiver. The updated ADSPEC is then returned to RSVP for delivery to the next hop along the path.

Upon arrival of the PATH message at an application receiver, the data in the SENDER_TSPEC and ADSPEC objects are passed across the RSVP API to the application. The application (perhaps with the help of a library of common resource-reservation functions) interprets the arriving data, and uses it to guide the selection of resource reservation parameters. Examples of this include use of the arriving "PATH_MTU" composed characterization parameter [7] to determine the maximum packet size parameter in the reservation request and use of the arriving Guaranteed service "C" and "D" parameters [22] to calculate a mathematical bound on delivered packet delay when using the Guaranteed service.

An application receiver wishing to make a resource reservation supplies its local RSVP with the necessary reservation parameters. Among these are the QoS control service desired (Guaranteed or Controlled-Load), the traffic specifier (TSPEC) describing the level of traffic for which resources should be reserved, and, if needed by the selected QoS control service, an RSPEC describing the level of service desired. These parameters are composed into an RSVP FLOWSPEC object and transmitted upstream by RSVP.

At each RSVP-aware point in the network, the SENDER_TSPECs arriving in PATH messages and the FLOWSPECs arriving in RESV messages are used to request an appropriate resource reservation from the desired QoS control service. State merging, message forwarding, and error handling proceed according to the rules of the RSVP protocol.

Finally, the merged FLOWSPEC object arriving at each of RSVP session's data senders is delivered to the application to inform each sender of the merged reservation request and properties of the data path.

3.1.5 Summary of IntServ/RSVP architecture

The integrated services architecture provides an extensional set of services to the traditional Best-Effort service of the Internet with the goal of allowing end-to-end QoS to be provided to applications. One of the key components of the architecture is a set of service definitions. The current set of services consists of the Controlled Load (CL) and Guaranteed Services (GS). Integrated Service is recommended by the IETF as the standard expression for application QoS request.

RSVP is an application-oriented signaling protocol that applications may use to request resources from the network. The network responds by explicitly admitting or rejecting RSVP requests. Applications that have quantifiable resource requirements may express these requirements using IntServ parameters as defined in the IntServ specifications.

RSVP works pretty well in the IntServ domains of end systems. However, when it

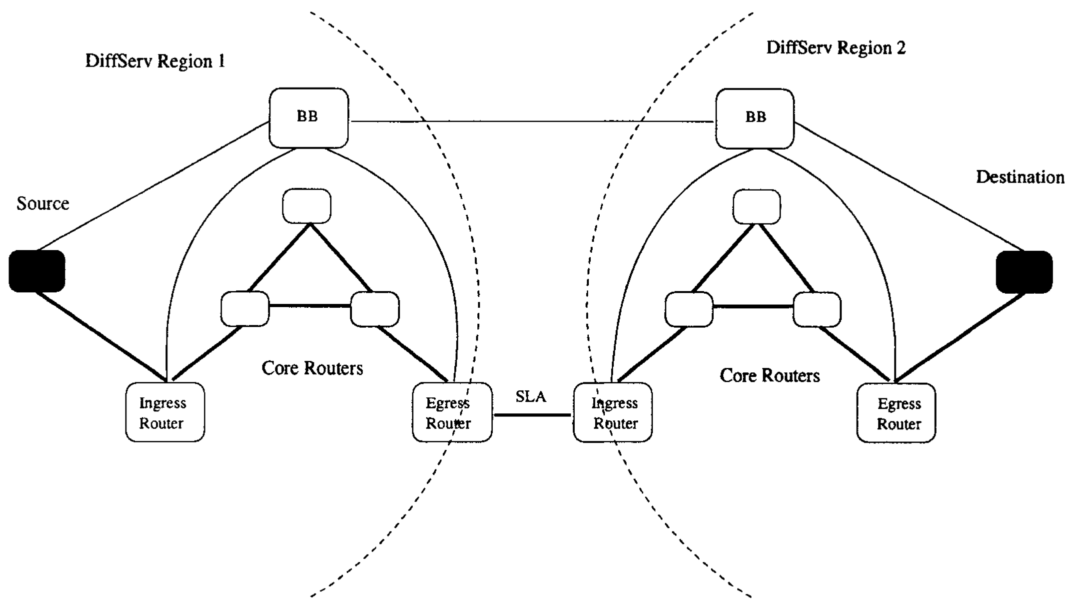

Figure 3: DiffServ Architecture

comes to large networks, the amount of per-flow state and per-flow processing raises scalability concerns. For this reason, RSVP is only usable in the end systems and access networks. Resource reservation in the network backbone must be done using aggregates.

The current prevailing idea of QoS implementation in end systems is based on a combined RSVP/IntServ architecture. In this architecture, RSVP signals per-flow resource requirements to network elements, using IntServ parameters. These network elements apply IntServ admission control to signaled requests.

The principle of "Push RSVP to the edge of the network, keep the network core simple" has been widely accepted in the deployment of QoS in the Internet.

3.2 Differentiated Services Architecture

The Differentiated Services (DiffServ) architecture is a standard from the IETF that is expected to be widely deployed across the Internet due to its fundamentally simple design and scalability. DiffServ contains a set of technologies by which network service providers can offer differing levels of network QoS to different customers and their traffic flows. DiffServ does not require explicit end user signaling of traffic characteristics and QoS requirements. Instead, the routers within DiffServ networks handle packets in different traffic flows by applying different per-hop behaviors (PHBs) to the packet forwarding. The PHB to be applied is indicated by a DiffServ 6-bit field code-point (DSCP) in the IP header of each packet. All routers within the DiffServ networks rely on this DSCP to provide forwarding treatment to a packet, which might be distinct from the treatment given to packets with a different DSCP.

The DSCP field is the 6 most significant bits from the 8-bit Type of Service (TOS) field in IPV4 or from the 8-bit Class Service (CS) field in IPV6. The DSCP value in the packet can be assigned by end hosts or a router along the path, based on policy or a detailed examination of the packet. Detailed examination of the packet, which is also called packet classification, is expected to happen closer to the edge of the network so that core routers are not overloaded. Routers along the path could use the DSCP field to limit the amount of resources allocated per traffic class.

The behavior of an individual device when handling traffic in the DiffServ architecture is called a Per Hop Behavior (PHB). PHBs only define an externally observable behavior such as low latency, low loss, etc. PHBs do not specify an implementation (for example, it might be possible to provide low latency to a particular class of traffic by using either Priority Queuing, or Weighted Fair Queuing). If all devices along a path provide a particular PHB, it is possible to construct an end-to-end service using such a PHB. The IETF proposes to standardize a set of PHBs and possibly services constructed out of them. Currently, two PHBs standardized by the IETF are Assured Forwarding (AF) and Expedited Forwarding (EF).

3.2.1 Assured Forwarding (AF)

Assured Forwarding (AF) PHB group is a means for a provider DiffServ domain to offer different levels of forwarding assurances for IP packets received from a customer DiffServ domain. Four AF classes are defined, where each AF class is in each DiffServ node allocated a certain amount of forwarding resources (buffer space and bandwidth).

IP packets that wish to use the services provided by the AF PHB group are assigned by the customer or the provider DiffServ domain into one or more of these AF classes according to the services that the customer has subscribed to. Further background about this capability and some ways to use it may be found in [11].

Within each AF class IP packets are marked (again by the customer or the provider DiffServ domain) with one of three possible drop precedence values. In case of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class. A congested DiffServ node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a higher drop precedence value [11].

In a DiffServ node, the level of forwarding assurance of an IP packet thus depends on

1. How much forwarding resource has been allocated to the AF class that the packet belongs to.
2. What is the current load of the AF class in case of congestion within the class.
3. What is the drop precedence of the packet.

For example, if traffic conditioning actions at the ingress of the provider DiffServ domain make sure that an AF class in the DiffServ nodes is only moderately loaded by packets with the lowest drop precedence value and is not overloaded by packets with the two lowest drop precedence values, then the AF class can offer a high level of forwarding assurance for packets that are within the subscribed profile (i.e., marked with the lowest drop precedence value) and offer up to two lower levels of forwarding assurance for the excess traffic [11].

3.2.2 Expedited Forwarding (EF)

The EF PHB is used to achieve a low loss, low latency, low jitter, assured bandwidth end-to-end service. This service appears to the endpoint as a virtual leased line

(VLL), also called premium service. The departure rate of the EF traffic from any DiffServ node must equal or exceed a configurable rate. The EF traffic should receive this rate independent of the intensity of any other traffic attempting to transit the node. It should have the average rate at least the configured rate when measured over any time interval equal to or longer than the time it takes to send an output link MTU sized packet at the configured rate [12].

The EF service is fundamentally different from the Internet's current Best-Effort service. But this service is not meant to replace Best-Effort, instead it is primarily to meet an emerging demand for a commercial service that can share the network with the Best-Effort traffic. It is expected that Premium traffic would be allocated a small percentage of the total network capacity, but its users will pay a much higher price than the users of Best-Effort traffic.

As illustrated in Figure 2-3, PHBs are applied by the conditioner to traffic at a network ingress point (network border entry) according to pre-determined policy criteria. The traffic may be marked at this point, and routed according to the marking, then unmarked at the network egress (network border exit). Originating hosts can also apply the DiffServ marking, and there are a number of advantages in doing so [13].

3.2.3 QoS related technologies in DiffServ

MPLS Label Switching

Proposed and developed by the IETF, Multi-protocol Label Switching (MPLS) is a network management protocol originally intended to integrate layer 2 information about network links (bandwidth, latency, utilization) into layer 3 (IP) elements within a particular system.

While traditional IP networks have no means of labeling, categorizing or monitoring the packets that traverse them, MPLS technology works to solve those IP shortcomings, placing labels on IP packets and providing that labeling function. And because MPLS is an overlay protocol it can operate with the IP protocol in the same

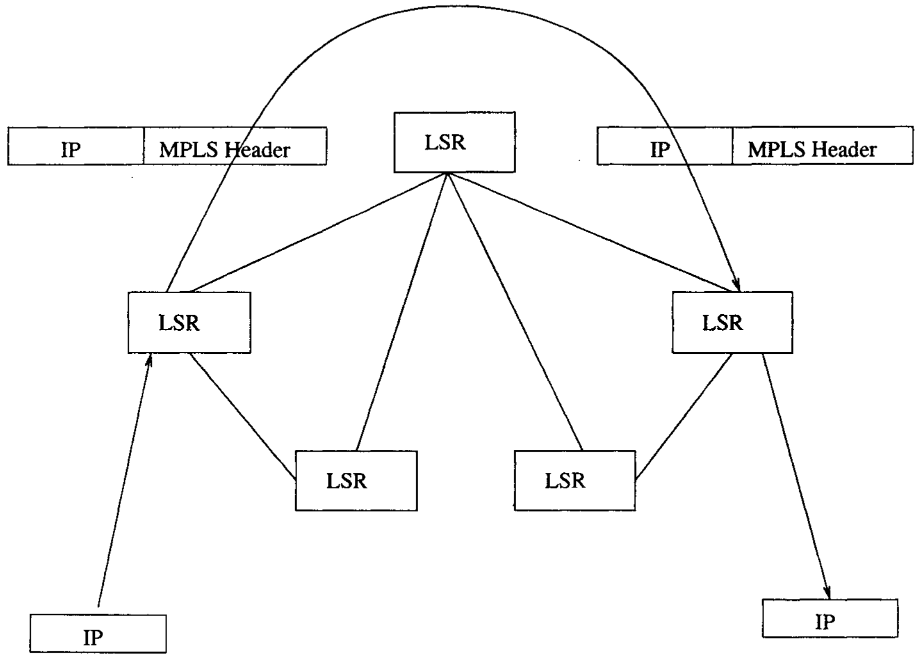

Figure 4: MPLS Network

network without interference.

MPLS-equipped networks use MPLS-aware devices known as label edge routers (LERs), positioned at the network's edges. These devices are designed to inspect IP packets entering the network and add MPLS headers, as well as removing the headers from packets leaving the MPLS network. Inside the boundaries of the MPLS network, devices known as label switch routers (LSRs) look for a MPLS label on each packet that passes through them, looking up and following the instructions contained in those labels, routing them based on a list of instructions.

MPLS allows administrators to define routes known as label switched paths (LSPs) from one LER to another, through a series of LSRS, across the MPLS network. These LSPs are pre-assigned and pre-engineered paths that packets with a certain label should follow.

Multi-Protocol Label Switching is similar to DiffServ in some respects, as it also marks traffic at ingress boundaries in a network, and un-marks at egress points. But unlike DiffServ, which uses the marking to determine priority within a router, MPLS markings (20-bit labels) are primarily designed to determine the next router hop. MPLS is not application controlled (no MPLS APIs exist), nor does it have an end-host protocol component. Unlike any of the other QoS protocols, MPLS resides only on routers. In addition, MPLS is protocol-independent (i.e., "multi-protocol"), so it can be used with network protocols other than IP (such as IPX, ATM, PPP or Frame-Relay) or directly over data-link layer as well [17] [16]. MPLS is more of a "traffic engineering" protocol than a QoS protocol. MPLS routing is used to establish "fixed bandwidth pipes" analogous to ATM or Frame Relay virtual circuits.

Since MPLS is not protocol-specific, it has been applied for RSVP and DiffServ.

- MPLS for RSVP
  As described in [19], there is a proposal to use an EXPLICIT_ROUTE object in RSVP to pre-determine paths taken by label-switched RSVP flows. These flows use virtual pipes established through MPLS-enabled routers (LSRs). Even without the EXPLICIT_ROUTE object in RSVP reservations, it is possible for MPLS to assign labels according to the RSVP FLOWSPECs. In either case, the effect is a significant simplification of RSVP support on the MPLS routers. By referencing MPLS labels, LSRs do not need to manage RSVP state [16].

- MPLS for DiffServ
  As might be expected, because DiffServ and MPLS are similar with respect to the qualitative QoS they enable (i.e., classification), mapping DiffServ traffic onto MPLS "pipes" (LSPs) is relatively simple. It is, but there are still DiffServ-specific considerations. To support DiffServ's per-hop model, an MPLS network operator needs to allocate a set of aggregate forwarding resources for each DiffServ forwarding class in each MPLS router (LSR) and assign labels. Additionally an LSR may need to associate the packet with a particular drop-precedence (which could be stored in the "experimental" (Exp) field of the MPLS header) [18].

The COPS protocol

The Common Open Policy Service (COPS) is a simple query and response protocol that can be used to exchange policy information between a policy server (Policy Decision Point or PDP) and its clients (Policy Enforcement Points or PEPs). One example of a policy client is an RSVP router that must exercise policy-based admission control over RSVP usage [20].

A chief objective of this policy control protocol is to begin with a simple but extensible design. The main characteristics of the COPS protocol include:

1. The protocol employs a client/server model where the PEP sends requests, updates, and deletes to the remote PDP and the PDP returns decisions to the PEP.

2. The protocol uses TCP as its transport protocol for reliable exchange of messages between policy clients and a server. Therefore, no additional mechanisms are necessary for reliable communication between a server and its clients.

3. The protocol is extensible in that it is designed to leverage off self-identifying objects and can support diverse client specific information without requiring modifications to the COPS protocol itself. The protocol was created for the general administration, configuration, and enforcement of policies.

4. COPS provides message level security for authentication, replay protection, and message integrity. COPS can also reuse existing protocols for security such as IPSEC or TLS to authenticate and secure the channel between the PEP and the PDP.

5. The protocol is stateful in that it allows the server to push configuration information to the client, and then allows the server to remove such state from the client when it is no longer applicable [21].

3.2.4 Summary of DiffServ architecture and its technologies

The Differentiated Service Architecture is a priority-oriented mechanism. The advantage of DiffServ is that many traffic streams can be aggregated to one of a small number of behavior aggregates (BA), which is invoked by the DSCP in the packet's IP header. The packets are each forwarded using the same PHB at the DiffServ router, thereby simplifying the processing and associated storage. In addition, there is no signaling, other than what is carried in the DSCP of each packet, and no other related processing that is required in the DiffServ network since QoS is invoked on a packet-by-packet basis.

DiffServ itself does not define the services that a DiffServ network can provide. Its successful deployment depends on the ability to use the technology to provide services. These services are reflected to customers at the edges of the DiffServ network in the form of a Service Level Specification (SLS) [3]. The ability to provide these services depends on the availability of cohesive management tools that can be used to provision and monitor a set of DiffServ routers in a coordinated manner. RSVP aggregates and Bandwidth Broker (see next section) are two of these tools.

DiffServ pushes the traffic classification and conditioning to the edge of the network, thus greatly reducing the overhead in the network.

3.3 RSVP in DiffServ

RSVP was originally designed to signal the Integrated Service requirement into the network. There have been attempts to apply it to Differentiated Service signaling ever since the DiffServ architecture was proposed. Therefore, two RSVP-related mechanisms have been proposed in the IETF to apply RSVP to DiffServ. They are RSVP over DiffServ [3] and RSVP aggregates [8].

3.3.1 Per flow RSVP over DiffServ region

In this approach, routers in the DiffServ network region respond to the standard per flow RSVP signaling originating from the IntServ-capable nodes outside the DiffServ region. These IntServ requests have to be properly interpreted by DiffServ routers so that the traffic with a IntServ request can get equivalent treatment in the DiffServ region. For example, the Controlled Load request of IntServ might be implemented using Assured Forwarding PHB in the DiffServ region. The mechanism interpreting IntServ to DiffServ is stated in [6]. This approach provides the benefits of the conventional RSVP approach (dynamic, topology aware admission control) without requiring aggregated RSVP support. Resources are also used more efficiently as a result of the per flow admission control. This approach only applies to those RSVP-aware routers in DiffServ region. When the routers are non RSVP-aware, other approaches must be implemented to achieve equivalent service in a DiffServ region. However, the demands on RSVP signaling resources within the DiffServ network region may be sufficiently high such that the routers in the DiffServ region are not capable of handling the huge amount of soft status for RSVP reservation. Therefore, per-flow RSVP over DiffServ is by no means feasible in the IP backbone.

3.3.2 RSVP aggregates

As described before, by classifying traffic flows, DiffServ and MPLS create what are effective "pipes" for these aggregates. For these pipes to provide any service quality better than standard Best-Effort, traffic on these virtual pipes must not exceed capacity. The problem is that neither DiffServ nor MPLS have specific protocol mechanics for detecting how much bandwidth they need and then allocating the necessary resources for dedicated usage. RSVP is designed to do that. Hence, although RSVP was originally designed to allocate bandwidth for individual application flows, it is very important for allocating bandwidth to accommodate the needs of traffic aggregates as well [19]. This need highlights the challenge, however, for network engineers using DiffServ or MPLS to know the bandwidth demands to anticipate, so they can make the appropriate resource reservation request. Additionally, senders and receivers at both ends of the virtual pipes must make these reservation requests so the appropriate

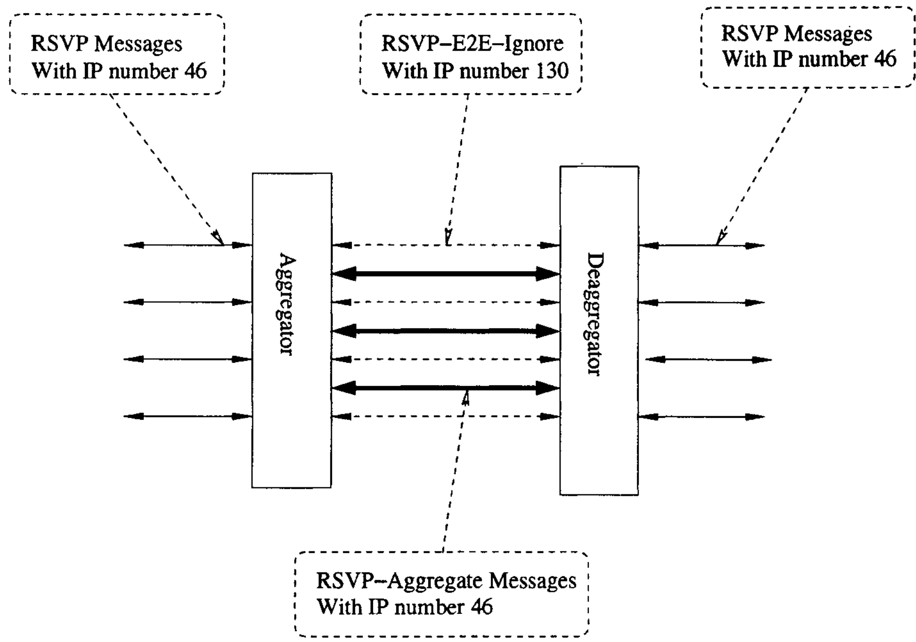

Figure 5: RSVP Aggregate Signaling

PATH and RESV messages can be sent from and to the appropriate unicast locations. "A key problem in the design of conventional RSVP version is, as noted in its applicability statement, that it lacks facilities for aggregation of individual reserved sessions into a common class. The use of such aggregation is required for scalability" [8]. So the consideration of using RSVP to provision for RSVP aggregates emerged.

RSVP aggregation solution involves the aggregation of several E2E reservations that cross an "aggregation region" and share common ingress and egress routers into one larger reservation from ingress to egress. An aggregation region is defined as a contiguous set of systems capable of performing RSVP aggregation along any possible route through this contiguous set.

Communication interfaces fall into two categories with respect to an aggregation region; they are "exterior" to an aggregation region, or they are "interior" to it. Routers that have at least one interface in the region fall into one of three categories with respect to a given RSVP session; they aggregate, they deaggregate, or they are between an aggregator and a deaggregator.

Aggregation depends on being able to hide E2E RSVP messages from RSVP-capable routers inside the aggregation region. To achieve this end, the IP Protocol Number in the E2E reservation's PATH, PATHTear, and ResvConf messages is changed from RSVP (46) to RSVP-E2E-IGNORE (134) upon entering the aggregation region, and restored to RSVP at the deaggregator point. These messages are ignored by each router within the aggregation region whenever they are forwarded to an interior interface. Since the deaggregating router perceives the previous RSVP hop on such messages to be the aggregating router, RESV and other messages do not require this modification; they are unicast from RSVP hop to RSVP hop anyway.

The token buckets (SENDER_TSPECs and FLOWSPECS) of E2E reservations are summed into the corresponding information elements in aggregate PATH and RESV messages. Aggregate PATH messages are sent from the aggregator to the deaggregator using RSVP's normal IP Protocol Number. Aggregate RESV messages are sent back from the deaggregator to the aggregator, thus establishing an aggregate reservation on behalf of the set of E2E flows that use this aggregator and deaggregator. The process is shown in Figure 6.

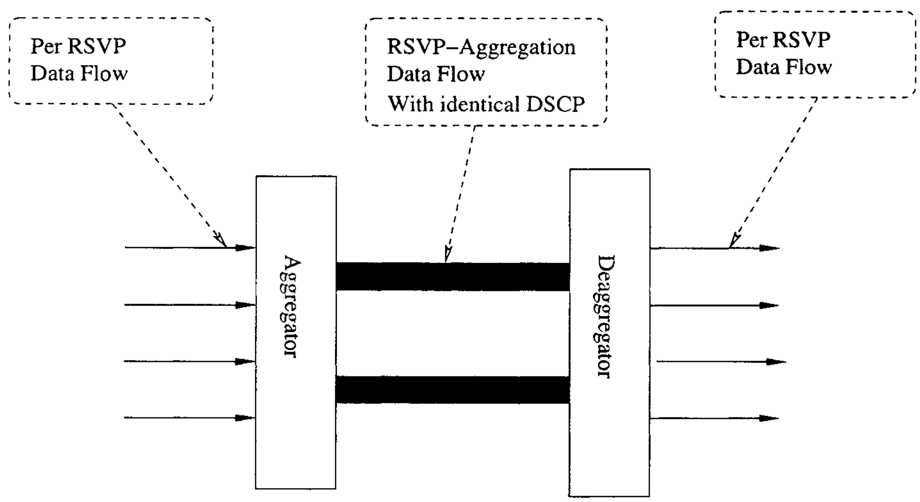

Figure 6: RSVP Aggregate Traffic Flow

Such establishment of a smaller number of aggregate reservations on behalf of a larger number of E2E reservations yields the corresponding reduction in the amount of state to be stored and number of signaling messages exchanged in the aggregation region.

By using Differentiated Services mechanisms for classification and scheduling of traffic supported by aggregate reservations (rather than performing per aggregate reservation classification and scheduling), the amount of classification and scheduling state in the aggregation region is even further reduced. It is not only independent of the number of E2E reservations, it is also independent of the number of aggregate reservations in the aggregation region. One or more DSCPs are used to identify traffic covered by aggregate reservations and one or more DiffServ PHBs are used to offer the required forwarding treatment to this traffic. There may be more than one aggregate reservation between the same pair of routers, each representing different classes of traffic and each using a different DSCP and a different PHB. The process is shown in Figure 6.

3.3.3 Summary of RSVP in DiffServ network

The traditional RSVP fails in the deployment in DiffServ network, because too many RSVP signals severely impact the DiffServ network. RSVP aggregates alternatively attempts to solve the problem by "hiding" RSVP messages from the DiffServ network, while guaranteeing the services requested by traditional RSVP in the manner of RSVP Aggregates in DiffServ network. Also, the number of RSVP states are greatly reduced in RSVP aggregation region (between ingress and egress of DiffServ network), which makes the deployment of RSVP in a DiffServ network possible.

However, the factor of the bandwidth efficiency comes up to consideration. How often is the bandwidth allocated to an aggregate reservation updated? Ideally, bandwidth should be updated after every change to the individual reservations of an aggregate. This would ensure that only the minimum possible amount would be allocated, but would most likely translate into a significant signaling load. Alternatively, bandwidth allocation could be updated less frequently to minimize signaling overhead. However, this could affect network efficiency by providing some aggregate reservations with more bandwidth than they really need, potentially preventing others from getting the bandwidth they require. Apparently there is no compromise to achieve both scalability and efficiency of bandwidth use in the same time.

In addition, the routers in the DiffServ region have to be RSVP-aware, although the amount of reservation state is greatly reduced in routers. Technically, it is not a problem at all. However, the requirement that network backbone has to be aware of RSVP does prevent the deployment of RSVP aggregates in the real world.

3.4 Bandwidth Broker

An "oracle" called Bandwidth Broker (BB) is an agent responsible for allocating preferred service to users as requested, and for configuring the network routers with the correct forwarding behavior for the defined service. A BB is associated with a particular trust region, one per domain. A BB has a database that keeps the information on who can do what, when and a method of using that database to authenticate requesters. Several interfaces are implemented in BB for either intra-domain communication, which is responsible for BB and its intra-domain routers, or inter-domain communication, which is responsible for the negotiation between adjacent BBs. Only a BB can configure the edge routers to deliver a particular service to flows. We discuss how a BB works in this section.

The BB model is as follows: In general, a bandwidth broker may receive a resource allocation request (RAR) from one of two sources: Either a request from an element in the domain that the bandwidth broker controls (or represents), or a request from a peer (adjacent) bandwidth broker. In either case, the bandwidth broker responds to this request with a confirmation of service or denial of service. This response is known as a Resource Allocation Answer (RAA). The mechanism for triggering the response is defined in the protocol specification [10].

3.4.1 Bandwidth Broker Specification

A Bandwidth Broker is a policy agent, which is responsible not only for bandwidth allocation, but also for some policy making and implementation.

The bandwidth broker consists of a few basic components shown in Figure 7. Their functions are defined in [9]:

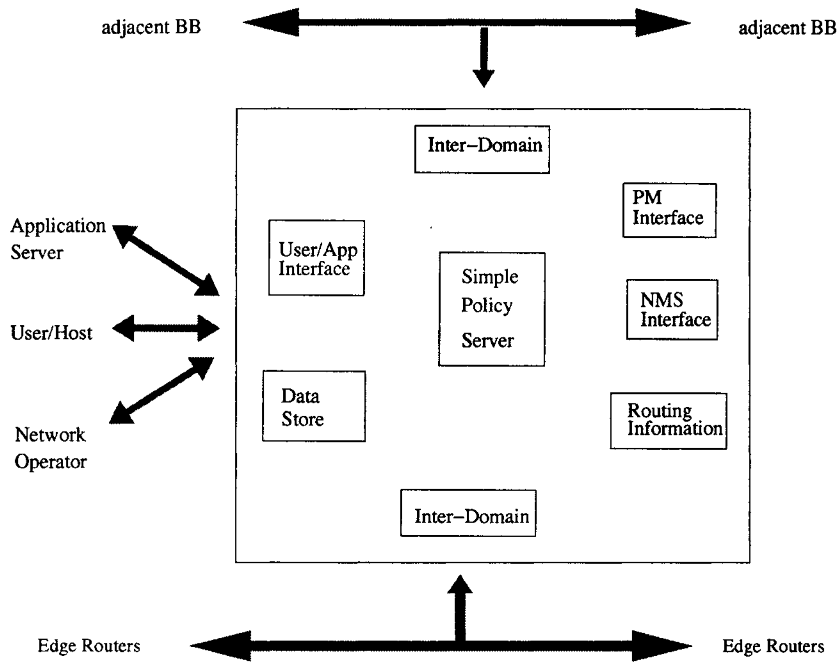

Figure 7: BB Architecture

1. User Interface: The user/application interface provides a means for the user to make resource requests directly, or to the network operator who will enter his requests. The interface also receives messages from setup protocols.

2. Inter-domain Interactions: Communicating between BBs in adjacent domains, negotiating SLA information between domains.

3. Intra-domain Interactions: Communicating between BB and intra-domain routers to provide quality of service.

4. Routing Table: A routing table is maintained to access inter-domain routing information so that BB can determine the edge routers and the downstream routers before allocating their resources. Also, additional routing paths can be maintained for different flows within the domain.

5. Database: A database is used to store information about all the BB parameters. The information that is stored within the repository includes SLAs, current reservations, configuration of routers, DSCP mapping, and policy information.

6. Policy Manager interface (alternatively, BB may provide policy management services) - utilize complex QoS/policy management functionality in policy manager - coordination of service level contracts and network resources - provide complete admission control processing.

7. Network Management interface (alternatively the BB may provide network management service) for coordination of network provisioning and monitoring.

Bandwidth Broker architecture basically consist two fundamental parts:

- BB messages, which represent the service type and level information. They are created and transfered among BBs and the routers involved in bandwidth resource management.

- BB protocols, which communicate between BB and routers within BB domain, as well as peer BBs.

BB messages and BB protocols are described as follows.

3.4.2 BB Messages

Several BB protocols work together to provide inter-domain and intra-domain communication, as well as resource management. The BB messages used by these BB protocols have two functions:

- Resource Allocation

  There are two messages for resource allocation, which are Resource Allocation Request (RAR) and Resource Allocation Answer (RAA).

  After Service Level Specification (SLS) is negotiated, the service is available. It however has not actually been allocated to any specific flows. The BB should not allow this available service to be used until it receives Resource Allocation Requests (RARs) for the service within a Domain and SLS between domains. The RAR requests within a Domain allow a user or network application to request resources from the Bandwidth broker. The RAR allow BB's in a single domain to keep track of the actual quantity of an available service that is allotted to traffic within a domain, and to make admission decisions based on this. The BB is able to process RARs from within a domain once the SSR has been confirmed.

A BB receives a Resource Allocation Request (RAR) message either from the Intra-domain routers or from Inter-domain BB. In either case a similar set of checks will be performed. As long as the requested resources are available, the Intra-domain RAR processed by BB X results in an RAR to the adjacent BB Y. If BB X received an RAR from BB Y, it will return a Resource Allocation Answer (RAA) confirming the requested availability of resources through domain Y.

- Resource Cancellation

  There are two messages for resource allocation cancellation, which are CANCEL and CANCEL ACK. The CANCEL message is simply a list of reservations that are no longer in force at a particular (set of) ingress or egress point(s) of the domain. The originating bandwidth broker, since it tracks these reservations, creates a CANCEL list consisting of the CANCEL originator's ID and authentication string, and a list of source prefix, destination prefix and reservation ID for each reservation being canceled. When the reserved resources are subject to failure by any reason, intermediate bandwidth brokers will unilaterally remove reservations. They do this by sending a CANCEL message to both upstream and downstream adjacent BB peers.

  The forwarding bandwidth broker also sends a CANCEL ACK to the peer bandwidth broker who forwarded the CANCEL to him. When the Cancellation Answer is sent the BB updates the status of all resources that were reserved for the specific service. When the adjacent domain receives a Cancellation Answer, the relevant conditioning parameters of the adjacent DS boundary routers are disabled.

3.4.3 BB Protocols

BB protocols are categorized as intra-domain and inter-domain protocols, according to their scope of functioning.

- Intra-domain protocol

  The purpose of this protocol is to communicate BB decisions to routers within the bandwidth broker's domain in the form of router configuration parameters for QoS operation and (possibly) communication with the policy enforcement agent within the router. Current bandwidth broker implementations have a number of different protocols for communicating with routers, including COPS, DIAMETER, SNMP, and vendor command line interface commands.

- Inter-domain protocol

  The purpose of this protocol is to provide a mechanism for peering BBs to ask for and answer with admission control decisions for aggregates and exchange traffic. There is already an inter-domain protocol defined by Internet2 as Simple Inter-domain Bandwidth Broker Signaling (SIBBS).

Simple Inter-domain Bandwidth Broker Signaling (SIBBS)

Simple Inter-domain Bandwidth Broker Signaling (SIBBS) is proposed by Internet2 community and adopted by Qbone architecture for communication between bandwidth brokers. It defines how peer BBs communicate in order to make a decision for each bandwidth reservation request. We describe here how the protocol works from end to end system [10].

We assume first, for purposes of description, that the bandwidth broker for a domain is a single entity and accessible to all end systems in the domain. Assume that the end systems have implemented the protocol to communicate with the bandwidth broker.

The Figure 8 gives an overview of the communication involved in this scenario. Its working sequence is summarized as follows.

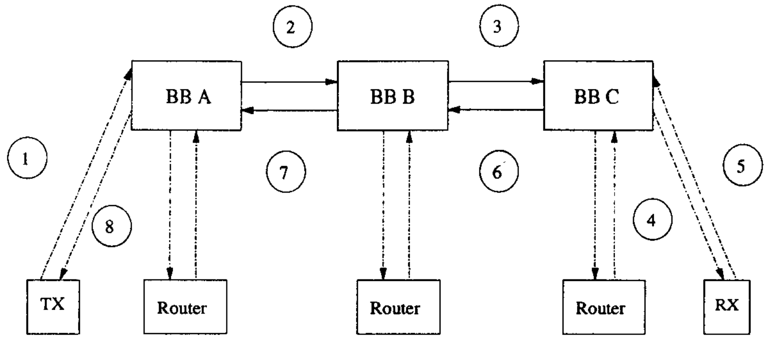

Figure 8: BB Signaling

1. The end system TX in the source domain sends a request (RAR) to its bandwidth broker A whose job it is to do resource allocation and make admission control decisions. Bandwidth broker A then checks whether the request fits into the SLSs that it currently has with adjacent domain(s) in the direction of domain C.

2. Assuming that the request can be handled, BB A sends an inter-domain RAR to BB B. BB A may aggregate this with other requests. It is not necessarily the case that each request received by a BB results in an inter-domain request, because BB fully keeps track of the resource reservation in its domain by checking the SLS.

3. BB B receives the inter-domain request from BB A. BB does not necessarily send an intra-domain RAR. BB B may again perform some level of aggregation and then sends the request further on to BB C.

4. BB C makes the determination, again based on existing SLSs, whether to admit the reservation and responds to BB B. This may involve further communication with the end system RX (4)(5) in the diagram. BB C then returns the decision to BB B.

5. BB B then sends an intra-domain request to reserve the resource upon the request from BB C. BB B notes the success or failure of the reservation and returns the information to BB A.

6. BB A notes the success or failure of the reservation and returns the results to the hosts.

7. The BBs in all domains may adjust the parameters in the edge routers in their domains as a result of the reservation.

In order to understand SIBBS further, we also offer the behavior of each BB while processing the resource allocation request as follows [10].

- Behavior of the bandwidth broker in the originating domain

  The end system sends an RAR to the bandwidth broker (1). This message includes a globally well-known service ID and an IP destination IP address, a source IP address, an authentication field, times for which the service is requested and the other parameters of the service. The bandwidth broker makes a number of decisions at this point, including the following:

  1. Whether the requester is authorized for this service.
  2. The egress router to which the flow may be assigned.
  3. The route through the domain to the egress router.
  4. Whether the flow fits in the SLS of the egress router with the net domain in the path to the destination.
  5. Whether the flow (possibly according to the policies of the domain) may be accepted for the specified service.

  If these decisions all have a positive outcome, the bandwidth broker will modify the RAR by including the ID for the domain and sign the request with its own signature (2).

  In case these decisions have negative outcomes, then the bandwidth broker returns a Resource Allocation Answer (RAA) to the end system (8). There can be additional information included, such as a reason code for the rejection and hints about what parameters might be acceptable at the moment that the answer is sent.

- Transit domain handling of the request

  In this case, the bandwidth broker receives an RAR from an adjacent bandwidth broker with a fully-specified destination address specification (2). The transit bandwidth broker must perform a number of functions:

  1. Authenticate that the request is indeed from a peer bandwidth broker.
  2. Determine egress router (interface) from its (inter-domain) routing tables.
  3. Check that the requested resources fall within the SLS with the sending domain connecting via one of the ingress routers of this domain.
  4. Check that the requested resources fall within the SLS connecting to a successor domain en route to the destination.
  5. Ensure that there are sufficient resources within the domain to support the flow from the ingress border router and (possibly) determine the intra-domain route. This determination may involve the domain's resource allocation strategy.
  6. Determine whether the flow may be accepted (possibly according to the policies of the domain).

  In the case where all these decisions have positive outcomes, the transit bandwidth broker modifies the RAR as appropriate (e.g., putting its own ID in the sender's ID field and authentication string in the message) and sends it to the bandwidth broker of the following domain en route to the destination IP address (3).

  In the case where these decisions have negative outcomes, the BB returns an RAA to the sending domain (7). Additional information, such as rejection reason code and hints about acceptable parameters may be returned along with the RAA.

- Behavior of the bandwidth broker in the destination domain

  In this scenario, the bandwidth broker of the destination domain knows the address of the end system that is to receive the flow. As in the behavior just described, on the reception of the RAR (3), it makes the following decisions:

1. Authenticate that the request is indeed from a peer bandwidth broker.
2. Determine the intra-domain route from the ingress router to the end system and decide whether the resources are available to support the flow.
3. Check that the requested resources fall within any possible SLS with the end system.
4. Determine whether the flow may be accepted (possibly according to the policies of the domain).

In the case where these decisions have negative outcomes, an RAA is sent back (6), possibly with a reason code and hints about acceptable parameters.

In the case where all these decisions have positive outcomes, the bandwidth broker sends the RAR to the end system with appropriate changes (4). In this case, the end system makes the determination whether it can receive the flow. This is signaled with an RAA to the bandwidth broker of the destination domain (5). The RAA contains authentication of the end system, and parameters for the flow which the end system is willing to accept (which may be different from those received). In case the flow is rejected, the RAA contains a reason code and possibly hints about the set of service parameters that would be acceptable.

Upon receiving the RAA from the end system (5), the bandwidth broker authenticates the answer and forwards the RAA, with appropriate changes, to the peer bandwidth broker that sent the RAR (6). At the same time, the bandwidth broker may configure traffic conditioners at the ingress router and possibly at other routers along the intra-domain path to the destination.

- Transit domain processing of the RAA

The RAA received from the peer bandwidth broker (6) is authenticated and the appropriate fields are modified and the RAA is sent to the next bandwidth broker in the chain back to the originating domain (7). Internally to the domain, the bandwidth broker may modify traffic conditioners and PHB parameters in the ingress and egress border routers in the path of the flow. In addition, resource allocation internal to the domain may be initiated by the bandwidth broker. This would consist of modifying PHB parameters and traffic conditioners in internal routers.

- Originating domain processing of the RAA

When the bandwidth broker of the originating domain receives the RAA (7) and authenticates it, the bandwidth broker completes any resource allocation actions within the domain, modifies PHB and traffic conditioner parameters at the egress router for the flow and forwards the RAA to the requesting end system (8). This may include setting the marking functions for the flow in the access router serving the requesting end system.

The end system receives the RAA and is able to send the flow. Note that there is nothing to prevent the end system from sending the flow earlier; however, the flow will not receive the requested service until the RAA is received and the DSCP of packets sent earlier than this will not be marked consistently with the service.

3.4.4 Summary of Bandwidth Broker and its technologies

Bandwidth Broker architecture implies the resource allocation is entirely handled by BBs all the way from end to end. In the network backbone (i.e., Internet), it works well for the fact that the topology of backbone is rather simple, and the number of network nodes in the backbone is rather small. The data stored in BB and the communication between BB and the routers in its domain are therefore quite small, hence BB is able to quickly process the resource allocation requests so as to meet the demand of speed in a large network scope.

However, the network topology of "last mile" is relatively dense compared with the network backbone where DiffServ is applied. The complexity and density of network where the end systems reside would definitely make the centralized network management scheme like BB overloaded. In addition, there is nothing so far mentioned in the BB specification [9] about how BB works with per-flow QoS reservation requests. It might make the management of BB too complicated if a BB has to handle the per-flow QoS requests for a large number of end systems.

3.5 The motivation for proposing the new architecture

RSVP provisions resources for individual application traffic flows, whereas DiffServ simply marks and prioritizes traffic. RSVP is more complex and demanding than DiffServ in terms of router requirements, so can negatively impact backbone routers. For this reason, the use of RSVP is strictly limited on the Internet backbone.

DiffServ is a perfect compliment to RSVP as the combination can enable end-to-end QoS. End hosts may use RSVP requests with high granularity. Border routers at backbone ingress points can then map those RSVP reservations to a class of service indicated by a DSCP. At the backbone egress point, the RSVP provisioning may be honored again, to the final destination. Ingress points essentially do traffic conditioning on a customer basis to assure that service level agreements (SLAs) are satisfied.

The DiffServ approach seems very promising, nevertheless it can not cover all the needs that the IntServ/RSVP model asks for. For example, an important difference between the IntServ/RSVP and DiffServ models is that the IntServ takes care of QoS request for individual applications, while DiffServ basically specifies local behaviors, which must be somehow composed to achieve per flow and end-to-end QoS that IntServ is supposed to provide. Having IntServ/RSVP request originated from end systems acquire the same level of service treatment in DiffServ network, without impacting the beauty of the simplicity of traditional IP network, is a big challenge in delivering end-to-end QoS across large network.

Currently, there are several approaches working in this purpose to have IntServ implemented over DiffServ network. The typical ones, which have been previously discussed in the this paper, are RSVP over DiffServ, RSVP aggregates, and Bandwidth Broker. Among those three typical QoS technologies in DiffServ, both per-flow

RSVP over DiffServ and RSVP aggregates impose an obligation on the network backbone, which is that the network must be provided with additional RSVP functions. This has been proved unacceptable by industry. Moreover, both RSVP over DiffServ and RSVP aggregates are not scalable, and therefore are not suitable for the large network. Bandwidth Broker has the most potential to provide good scalability and high efficiency of bandwidth use for service differentiation in the network backbone for the fact that its resource management scheme is quite flexible. However since it is not applicable in dealing with the huge number of QoS requests from end systems, it should be limited in the network backbone where the DiffServ applies.

At the current time, most of BB's ideas are on the conceptual stage. So it is interesting to make an applicable architecture, taking advantage of the gained research achievement of current QoS technologies discussed above, to realize end-to-end QoS. This idea will be developed in the chapters that follow.

Chapter 4

Proposed Architecture for RSVP over Network Backbone

In the previous chapters, we summarized the successful ideas of the three architectures of resource allocation. We now propose a new E2E QoS architecture. The basic principles of the proposal are listed as follows:

1. Apply IntServ/RSVP architecture at the end system to request IntServ QoS for an application.
2. Apply DiffServ at the network backbone to differentiate network traffic.
3. Apply BB architecture in a DiffServ region to implement resource reservation aggregates.
4. Treat all the BBs in DiffServ region as a series of RSVP nodes.
5. Hide RSVP messages in the DiffServ region to prevent the network backbone from being overwhelmed.

4.1 Overview of the Model

We assume in this proposal that the DiffServ region (network backbone) physically consists of several DiffServ domains. A DiffServ domain is a contiguous set of DiffServ nodes which operate with a common service provisioning policy and set of PHB groups implemented on each node. Each DiffServ domain is designated with one individual Bandwidth Broker. The BBs are both RSVP and DiffServ-aware nodes. The IntServ QoS requests from the end systems are mapped onto the equivalent DiffServ QoS requests in the BBs. The resource reservation can be done by means of intra-domain protocols, such as COPS, within each DiffServ domain. As to the resource reservation spanning the whole DiffServ region, there are three alternatives coming to the consideration:

1. Isolated BBs
2. Unique BB
3. Series of BBs

Each alternative acts differently in terms of communicating the resource reservation information among the DiffServ domains. The RSVP QoS requests from the end systems are handled appropriately in three alternatives so that the requested services are guaranteed across DiffServ region. In this way, the DiffServ region can be regarded as one single RSVP-aware node or a number of RSVP-aware nodes along the RSVP path from the point of view of end system. Therefore, Integrated Service can be deployed over the DiffServ region.

4.2 Alternatives of BB Responsibility in DiffServ Region

1. Isolated BBs

   In this alternative, each BB is only responsible for the QoS request within its own domain. The features of this alternative are:

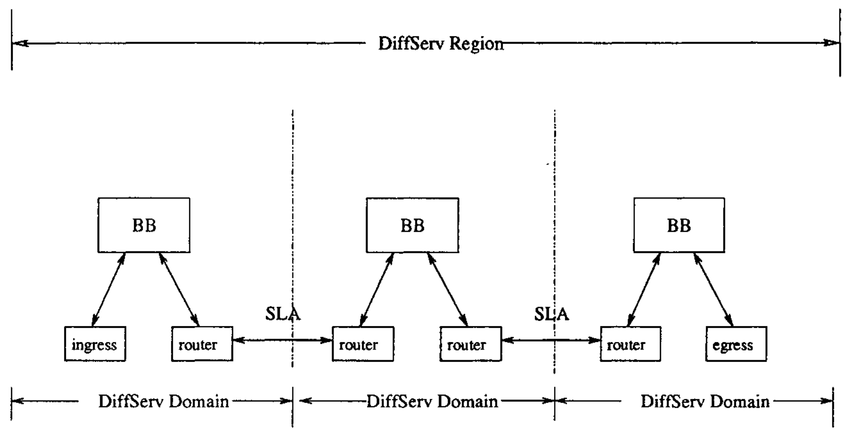

Figure 9: Isolated BBs in DiffServ Region

(a) No inter-domain communication protocol exists between BBs

(b) BBs allocate resource upon QoS request in their own domain respectively

(c) Service negotiation and guarantee are through SLAs

Because it lacks efficient communication between BBs, the scalability can not be attained.

2. Unique BB

   In this alternative, a single BB is designated to handle the QoS request across multiple DiffServ domains. The resource availability is communicated among BBs through inter-domain protocol. The advantage of this alternative is that the resource management is easier to be implemented because the resource management power is concentrated in the unique BB. The disadvantage however is also obvious. The unique BB has to be involved in the resource management of other DiffServ domains, which imposes certain restrictions on the freedom of IP network. Therefore, it is difficult to deploy.

3. Series of BBs

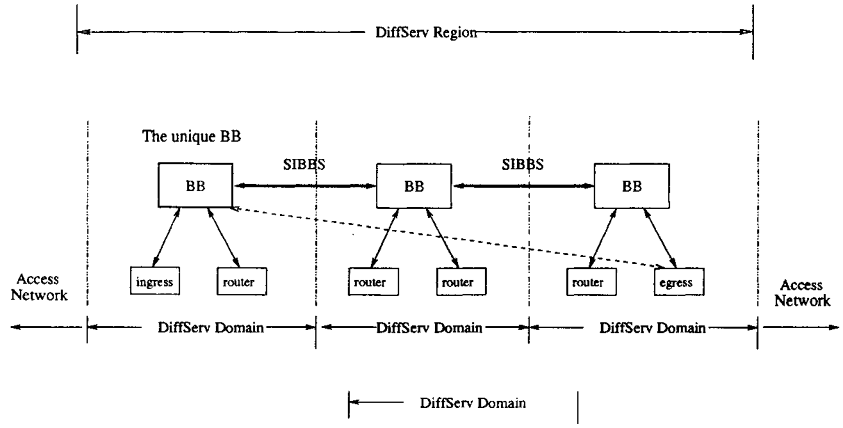

Figure 10: Unique BB

In this alternative, all BBs share the responsibility to allocate the resource for the service request from outside DiffServ region. They take the advantage of the Internet2 community proposed inter-domain protocol, SIBBS, to communicate the resource availability among them. Its advantage is that the load of allocating resources is shared by each BB. Little modification to the currently proposed DiffServ resource management mechanisms is introduced, therefore, it is scalable and easy to deploy

By comparing the above three alternatives, we adopt Series BBs as the mechanism to guarantee identical RSVP QoS across DiffServ region. The DiffServ region is therefore capable of supporting Integrated Services along the paths that span the domains within DiffServ region.

From the point of view of end systems, we may simply regard the DiffServ region as a sequence of RSVP-aware nodes that are responsible for resource management in the network backbone. The routers connecting end system domains to DiffServ region are deemed as the border routers for the entire DiffServ region as showed in Figure 11. The idea of making the network backbone as an integrated DiffServ region is supported in the IETF [2]:

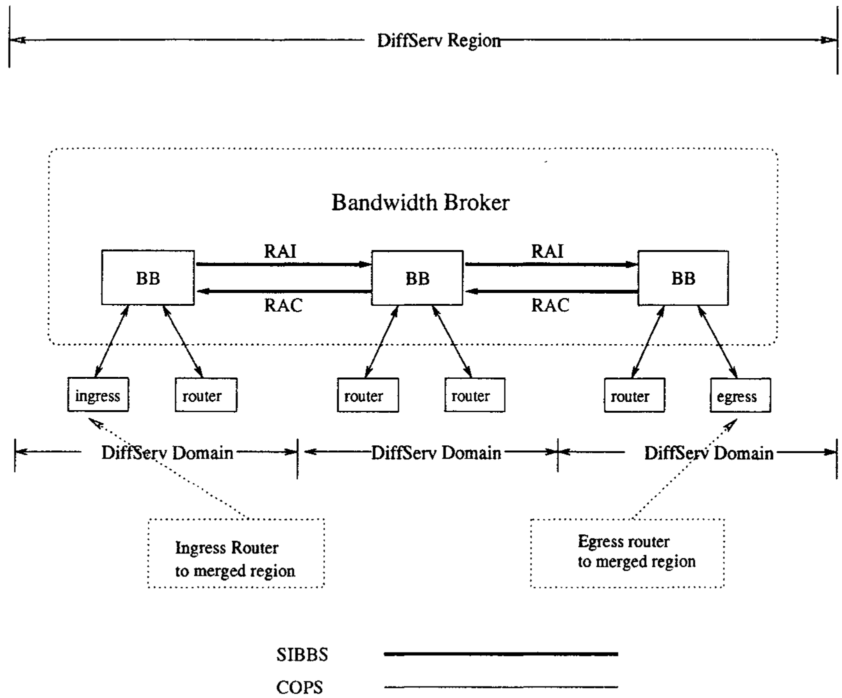

Figure 11: Series of BBs

"IntServ and DiffServ are the endpoints of what can be seen as a continuum of control models, where the fine-grained precision of the per application invocation reservation model can be aggregated into larger, more general and potentially more approximate aggregate reservation states, and the end-to-end element-by-element reservation control can be progressively approximated by treating a collection of subnetworks or an entire transit network as an aggregate service element."

By now, we may start to introduce the new architecture of treating DiffServ region as a series of RSVP Node (See Figure 12).

In the new architecture, we assume the two end systems do not reside in the same network domain. The packets originating from one end system have to travel across:

- the sender's access network
- the network backbone

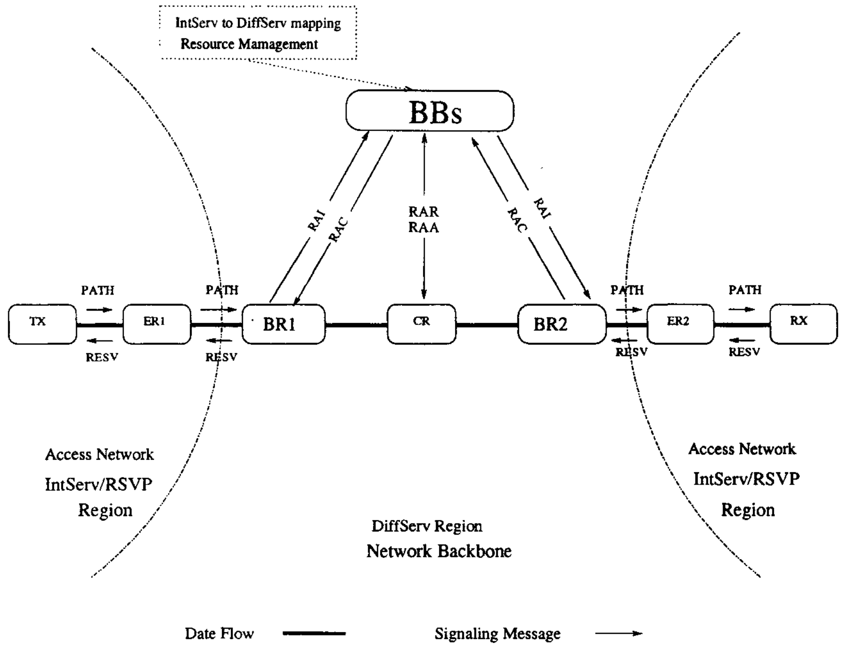

Figure 12: DiffServ Region as RSVP Nodes

- the receiver's access network

to reach the end system on the other side. The proposal is also based on the assumption that there is only one exclusive routing path between each pair of ingress/egress routers.

The access network

The sender and receiver networks are access networks that are supposed to provide the network connection to a large number of end users in the access network where the network topology is considerably dense. The QoS is mainly concerned about time-critical applications, which have to be run at the end systems. The per-application based QoS requests from those applications might be significantly large due to the large number of end users in access network. Bandwidth Broker in this case is apparently not applicable to manage the network resource for each individual QoS request because of the large number of QoS requests. In addition, the overwhelming could also be caused by the communication between BB and a large number of end users. Therefore, we do not apply BB in the access network for resource management.

On the contrary, Integrated Service has proven quite successful in describing QoS on the basis of per-application flows in the end system. So we apply Integrated Service to request QoS for end systems in the whole access network regions. The IntServ resource reservation requests are signaled by RSVP in the access network regions where the sender and receiver reside. The access network regions is therefore referred to IntServ/RSVP regions.

Network backbone

The network backbone is the core network (i.e., Internet) that connects the access networks. The topology of the network backbone is rather sparse compared to the access network. Differentiated Service is considered to be the feasible choice to carry on the QoS request in network backbone. So we call the network backbone DiffServ region. Bandwidth Broker is currently considered to be a solution to give the DiffServ region a manageable resource provision.

All the routers in DiffServ region are DiffServ-enabled. BB is the only RSVP aware node in each DiffServ domain, which is in full compliance with the RSVP process. The RSVP messages originating from the access network are subject to standard RSVP process only in BB, thus preventing the network backbone from being overwhelmed by RSVP soft state refreshing. BB is also the agent responsible for service mapping of IntServ to DiffServ. Since BB is both IntServ/RSVP and DiffServ aware agent in DiffServ region, the IntServ QoS request carried by RSVP messages is therefore able to be interpreted and implemented by DiffServ region. RSVP messages themselves are conveyed through DiffServ region in standard RSVP processing, which may guarantee the requested service identical after they traverse the DiffServ region.

BB is the policy server, which is called Policy Decision Point (PDP). It is responsible for service mapping and resource management. The routers in its domain are its clients, which are called Policy Enforcement Points (PEPs) [21]. Out of the regular routers, the border (ingress/egress) routers are committed with mission of admission control. BB communicates with its clients by means of the COPS protocol.

4.3 Requirement of Elements

4.3.1 Bandwidth Broker

Bandwidth Broker in this proposal is facilitated with the additional features of IntServ-DiffServ mapping and RSVP message process, as well as those specified in BB specification [9]. In order to meet the requirement of being an RSVP-aware node, several additional components are introduced into the BB architecture in this proposal:

- Two new SIBBS messages, Resource Allocation Inquiry (RAI) and Resource Allocation Confirmation (RAC). RAI conveys PATH message, while RAC conveys RESV message across a DiffServ region.
- Resource Reservation Table (RRT). It records the available resources for each pair of ingress/egress routers.
- Mapping Interface. It is responsible for IntServ-DiffServ mapping and service inquiry to RRT.

1. RAI

   Resource Allocation Inquiry. It is created at the border router (BR1) where PATH message firstl arrives in DiffServ region from upstream sender network. Its data format is almost the same as RAR of SIBBS, except that the Service Specification Object (SPO) field of RAR message is filled with the PATH message. It takes the source and destination addresses of the end systems that require QoS as its source and destination addresses. The PATH message is honored and subject to standard RSVP processing in each BB along the RSVP path in DiffServ region. RAI is signaled by SIBBS. To be concise, only the source address field, destination address field and SPO field are listed in Figure 13. The remaining fields of RAI should be the same as those of RAR, which are listed in the appendix.

2. RAC

Source Prefix (IP address of source end system)
Destination Prefix (IP address of destination end system)
SPO (filled with PATH message)

Figure 13: Resource Allocation Inquiry (RAI) message

Resource Allocation Confirmation. It is created at the border router (BR2) where RESV message first arrives in DiffServ region from downstream. Its data format is almost the same as RAA of SIBBS, except that the SPO field of RAC message is filled with the RESV message. It takes the source and destination addresses of the end systems that require QoS as its source and destination addresses. The RESV message is honored and subject to standard RSVP process in each BB along the RSVP path in DiffServ region. RAC is signaled by SIBBS. To be concise, only the source address field, destination address field and SPO field are listed in Figure 14. The remaining fields of RAC should be the same as those of RAA, which are listed in the appendix.

3. Resource Reservation Table (RRT)

   The RRT is saved in Data Store in BB. It records the available resource between each pair of DiffServ border routers. The available resource in RRT is obtained by picking up the smallest available resource along the routers between the pair. RRT is updated whenever Resource Allocation Request (RAA) from an interior router is approved.

   In Figure 15, BOmn is the Bandwidth Object recording the minimum available bandwidth along the path between border routers. It does not have to be tied to the traffic between BRm and BRn. For instance, if the path for another ingress/egress pair BRj-BRk happens to cross over the path of BRm-BRn, the

Source Prefix (IP address of source end system)
Destination Prefix (IP address of destination end system)
SPO (filled with RESV message)

Figure 14: Resource Allocation Confirmation (RAC) message

available bandwidth of the cross-point router could decline because of the increment of the resource consumption for BRj-BRk path. So RRT has to be checked every time before a new reservation request is accepted. Also, the RRT is refreshed every time an Resource Allocation Request (RAA) is accepted.

4.3.2 Sender and Receiver

TX is the sender of the application data with QoS request. RX is the receiver of the application data with QoS request. We assume that both sender and receiver use RSVP to communicate the quantitative QoS requirements of QoS-aware applications running on the host. In principle, other mechanisms may be used to establish resource reservations in IntServ-capable nodes, but RSVP is clearly the prevalent mechanism for this purpose.

4.3.3 Edge Router

ERs are the edge routers, residing adjacent to the DiffServ network region. Since it is the last station connecting the access network to network backbone, the SLA is stored in ER. It is wise to monitor the reservation in ER so that the traffic claiming

	BR1	BR2	BR3	. . .	BRn
BR1	NULL	BO12	BO13	. . .	BO1n
BR2	BO21	NULL	BO23	. . .	BO2n
⋮	⋮	⋮	⋮	⋮	⋮
BRn	BOn1	BOn2	BOn3	. . .	NULL

Figure 15: Resource Reservation Table(RRT)

the resource reservation is legitimate. ER will downgrade the illegitimate request to best effort service so that they can not "steal" bandwidth by chance. The resource requested by RSVP message is inspected in edge router. If it is beyond the SLA that has already been negotiated, the QoS request will be refused.

4.3.4 Border Router

BR1 and BR2 are the border routers of DiffServ region connecting access domains. BR1 and BR2 are named as ingress and egress routers respectively according to the data flowing direction. Aside from all the functions they have as regular DiffServ routers, they have two more special features for admission control and treatment of RSVP messages:

- They are capable to differentiate the RSVP signaling messages from other IP packets by checking the IP header of incoming messages, because RSVP packets are marked with RSVP as protocol number in their IP header. Alternatively, the recognition of RSVP message can be done by marking RSVP messages with unique DSCP value in the end system. The packets with DSCP of RSVP will be easily recognized and treated in DiffServ region. This method also has another advantage that the control plane, like RSVP message, might be routed in higher

Per–flow Session 1	Reserved resources
Per–flow Session 2	Reserved resources
. . .	. . .
Per–flow Session n	Reserved resources
Available Resources	

Figure 16: Bandwidth Broker Object (BO)

priority. The RSVP messages are then conveyed to the local BB by SIBBS.

- Border routers also act as admission control agents for the DiffServ network region. A new component, Admission Control Table (ADCT), is introduced to record the reserved resource for each access domain. ADCT is stored in ingress router. It provides the criteria for admission control to each access domain which has an SLA with the DiffServ region. Each time when a new reservation is approved or the reserved resource is released by BB, BB will update ADCT in the corresponding BR. This is to prevent the reserved bandwidth from being abused. The entry of ADCT is on per-domain basis (see Figure 17).

4.3.5 Core Router

CRs are the core routers which are pure DiffServ-enabled routers. They only implement aggregated traffic control according the value of DSCP.

Egress1	Session Object1
Egress2	Session Object2
⋮	⋮
Egressn	Session Object n
Available Resource	

Figure 17: Admission Control Table (ADCT)

4.4 E2E QoS Reservation Setup

RSVP messages are issued by both end systems (sender and receiver) in access network. They are passed toward the end system on the other side. They are subject to standard RSVP processing in the access network (IntServ/RSVP region), and finally enter into network backbone (DiffServ region) at the border routers (ingress/egress routers). The routing path of an RSVP message in the DiffServ region is the shortest path connection between ingress/egress routers and the BB. This path may also be determined by the local policy.

The BB, designated to each DiffServ domain, is the only element providing standard RSVP processing. Border (ingress/egress) routers may distinguish RSVP signaling messages from other traffic by looking at the protocol ID in its header (or checking the unique DSCP of RSVP, if applicable). When RSVP messages arrive at border routers of the DiffServ region, they trigger the SIBBS which forcibly "routes" them to the local BB and then the next BBs till another border router in the direction of the RSVP messages' destination. The BBs work together to provide the standard RSVP processing to the RSVP messages. From the perspective of the sender and the receiver, the DiffServ region behaves like a series of RSVP aware nodes in terms of resource reservation.

Another critical mission in the BB is service mapping. The BB is both RSVP-and-DiffServ-aware element in DiffServ region. The QoS request represented in RSVP messages must be interpreted into DiffServ service level specifications (SLSs), which are represented by various DSCP, and so does the service mapping of DiffServ to IntServ. The issue of service mapping is still under study by the IETF. An alternative proposal can be found in [6].

In this way, we "drive" RSVP signaling messages away from the data routers while treating the whole DiffServ region as a series of RSVP aware nodes.

The processing of RSVP signaling messages from end to end is described as follows:

1. Issue RSVP PATH message in sender domain
   The QoS process on the sender TX generates an RSVP PATH message that describes the traffic requested by the sending application. The PATH message is forwarded toward the receiver Rx. In sender's access network, standard RSVP/IntServ processing is applied to capable network elements.

2. Handle PATH message in ER1
   RSVP PATH message sent toward the receiver RX arrives in ER1, the edge router of sender domain. It is subject to standard RSVP processing. The PATH message is sent onward to DiffServ region, which arrives in the edge router BR1.

3. Handle PATH message in BR1

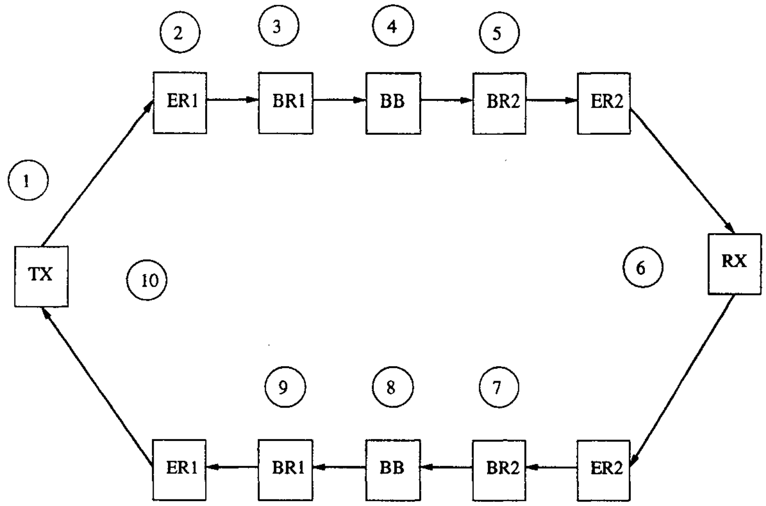

Figure 18: E2E QoS signaling

BR1 is the first element in DiffServ region that receives RSVP signaling messages from upstream IntServ/RSVP region. BR1 identifies PATH message by looking at the protocol ID (or DSCP value of RSVP) in the header of PATH message. Realizing that it is an E2E PATH message and, its destination address is toward interior DiffServ region, BR1 issues an RAI message towards the BB. Since the BB is the only RSVP-capable element in DiffServ region, hence the RSVP delegate of DiffServ region along E2E reservation path, RAI message encapsulates the E2E PATH message in the field of SPO and directly delivers it to the BB by SIBBS. The encapsulation of the E2E PATH message is intended to guarantee the integrity of E2E PATH message when it is to be used for setting up/updating soft state in the BBs. The source address of RAI is the IP address of sender Tx. The destination address of RAI message is the IP address of the receiver Rx.

4. Handle RAI message in BBs
   Upon receiving RAI message, the BBs unpack the PATH message from the RAI. The BBs update ADspec object in the PATH message based on the DiffServ_to_RSVP mapping rules [6], as well as local policies. The updated PATH message is again encapsulated into the RAI message and then sent to the next BB in the direction towards the receiver. RAI message is processed in the BBs

in the same way as mentioned above. The BB in the last DiffServ domain next to the receiver's domain sends RAI to the edge router ER2. As a result, the soft-state is set up in every BB and the PATH message is updated with the current resource availability across the DiffServ region.

5. Handle RAI message in BR2

   On receipt of the RAI message at the BR2, the BR2 retrieves the E2E PATH message from the RAI message and forwards it towards outside the receiver's domain. The BR2 also checks the requested reservation to see if it fits in the SLA.

6. Handle PATH and RESV message in receiver domain

   When the PATH message reaches the receiver Rx, the operating system generates an RSVP RESV message, indicating interest in offered traffic of a certain IntServ service type.

   The RESV message is carried back towards the DiffServ network region and the sender. Consistent with standard RSVP/IntServ processing, it may be rejected at any RSVP-capable node in the path if resources are deemed insufficient to carry the traffic requested.

7. Handle RESV message in BR2

   On receipt of E2E RESV message from downstream IntServ/RSVP region, BR2 issues an RAC message, encapsulating E2E RESV message in the SPO field, and conveys it towards BB by SIBBS.

8. Handle RAC message in the BBs

   The RESV message is unpacked from RAC at the BB and similarly subject to standard RSVP/IntServ processing. The IntServ QoS request presented by RESV is mapped to DiffServ request by the service mapping interface. The BB then consults its Resource Reservation Table (RRT). RESV is either accepted or rejected based on the available resource and local policy.

   If the BB approves the request, the BB then updates RRT to reflect the reduced capacity available at the admitted service level. After the RESV message is processed, it is again encapsulated into RAC message and forwarded to the next BB. RAC message is conveyed and processed in each BB all the way towards the most upstream DiffServ domain. Realizing it is the border domain connecting outside DiffServ world, the BB sends RAC to the border router BR1 by SIBBS. Internally, the BBs initiate resource reservation announcements to all the involved routers by inter-domain protocol (COPS, for instance). In each internal router, the requested resources are reserved and the available resources are consequently reduced. In addition, the ADCT in the ingress router is also updated with the change of the resource allocation.

Internally to the DiffServ domain, the BB may modify traffic conditioners and PHB parameters in the ingress and egress border routers to adjust the condition of admission control and DiffServ routing.

If RESV is rejected, it is not forwarded and the appropriate RSVP error messages are sent.

9. Handle RAC message in BR1

   When the RAC arrives in the BR1, indicating the approval of the resource reservation for the RSVP session, the BR1 then retrieves the RESV message from the RAC message. The corresponding information, such as the RSVP session ID and the amount of reserved resource, are also retrieved from the RESV message. The BR1 is then updated with these resource reservation information. The RESV is forwarded upstream to the sender's domain afterwards.

10. Handle the RESV message in sender domain

    The RESV message proceeds through the network region to which the sender is attached. Any RSVP node in this region may reject the reservation request due to inadequate resources or policy. If the request is not rejected, the RESV message will arrive at the sender TX.

    At TX, the QoS process receives the RESV message. It interprets receipt of the message as an indication that the specified traffic flow has been admitted for the specified IntServ service type (in the IntServ-capable nodes). It may also learn the appropriate DSCP marking to apply to packets for this flow from the information provided in the RESV.

    TX may mark the DSCP in the headers of packets that are transmitted on the admitted traffic flow. The DSCP may be the default value which maps to the

IntServ service type specified in the admitted RESV message, or it may be a value explicitly provided in the RESV.

In this manner, we obtain end-to-end QoS through a combination of networks that support RSVP/IntServ and networks that support DiffServ.

4.5 Routing of Data Flow with QoS Request

The data flows with IntServ QoS request are aggregated at the ingress router of DiffServ region, based on the DSCP. The admission to the RSVP data flows is implemented at the ingress router. The admitted data flows must not be beyond the reserved resource specified in ADCT. The data flows not being identified within the reserved resources are routed with Best-Effort service. In addition, the data flows exceeding the reserved bandwidth would be either shaped or dropped.

The admitted data flows with RSVP requests are routed in PHBs in DiffServ region.

4.6 E2E QoS Reservation Refresh and Tear Down

Reservation Refresh
If the RSVP message is new, then it follows the reservation setup procedure mentioned above. Otherwise, the BBs are refreshed by the constantly issued service-keeping PATH/RESV messages, as specified in RFC 2208 [20].

Service Tear Down
Each time when a new RESV message arrives at each BB from downstream, BB has to check the resource availability in its domain by means of intra-domain interface. If the available resource is inadequate, then tear down message is issued by BB.

When no RSVP message for the specific data flow refreshes BB, which leads the reserved session time out, then a tear down message is issued in both directions towards TX and RX.

4.7 Dynamic Provision of Resource in DiffServ Region

When the resource request is approved in DiffServ region, the DiffServ service level resources equivalent to the admitted RSVP request will be reduced from the available resources in each router. Similarly, this portion of resources will be released when this RSVP session ends up. The BBs periodically collect the updated available resources in routers by means of the intra-domain interface. Routers might also send messages to the BBs indicating traffic change on a specific service level. Corresponding to the alert message, BB either reduces the traffic reservation, or reconfigures the router to get more resources from other service levels. This determination is made based on the local policy. This makes the dynamic resource management a reality in the DiffServ region.

Considering that the number of end users in the access system could be large, it might not be feasible to keep track of the resource reservation status in BB for every application. It is, however, practical to have the per-application resource allocation requests managed in their own access domain. BB trusts the access domain that has the SLA with BB. QoS requests originating from the end system belong to the only entry representing its domain in RRT.

Chapter 5

Conclusion and open issues

5.1 Conclusion

The attempts offering the most promising QoS to the IP network is not universal adoption of a single architecture. A tailored approach should be used instead to aggregate service elements in the network backbone where scalability is a major design objective. Per-flow service is provided at the edge of the network (access networks in this paper) where accuracy of the service response is a sustainable outcome.

The per-flow QoS mechanism is adopted in the network edge (access network) to make the QoS request as precise as it could be. There is not much work to do at the edge of network since IntServ/RSVP architecture has proven successful.

In this proposed architecture, "hiding" RSVP messages within the DiffServ region successfully protects the DiffServ region from being overwhelmed by RSVP processing.

Instead of checkng five fields of RSVP data packets, which are source address, destination addresss, source port number, destination port number and IP protocol number in the IP header, the DiffServ routers check only the field of DSCP. This makes the packet forwarding much faster in DiffServ region than in IntServ region, while guaranteeing the Integrated Services over DiffServ region.

The network resource in network backbone is utilized dynamically, it is therefore scalable to address the traffic with various services. Different sorts of traffic are able to share the network resource scalably in the network backbone. It creats minimum impact on current network, because there is no other new load applied, except RSVP over BB, in network backbone.

This proposed architecture attempts to abstract the successful points of the prevailing QoS technologies, while strictly in compliance with the E2E QoS principle, "Push the complex to the network edge and keep the core simple". Therefore, it has a momentum for support.

5.2 The Open Issues

In order to deploy the E2E QoS, there are still several issues that need to be addressed in the future:

1. Active network resource management
   There is a more fundamental issue here concerning resource management and traffic engineering. The approach of single path selection with static load characteristics does not match a networked environment that contains a richer mesh of connectivity and dynamic load characteristics. In order to make an efficient use of a rich connectivity mesh, it is necessary to be able to direct traffic with a common ingress and egress point across a set of available network paths, spreading the load across a broader collection of network links. At its basic form this is essentially a traffic engineering problem. To support this function it is necessary to calculate per-path dynamic load metrics, and allow the network's ingress system the ability to distribute incoming traffic across these paths in accordance with some model of desired traffic balance. To apply this approach to a QoS architecture would imply that each path has some form of vector of quality attributes, and incoming traffic is balanced across a subset of available paths where the quality attribute of the traffic is matched with the quality vector of each available path [2].

2. Universal service level definition

   Each domain will have its local policy to manage the resource. The same Integrated Service level could be interpreted into various DSCP, hence the QoS request could be distorted after transmission over many DiffServ domains. The solution might be standardizing the DiffServ levels universally, or remarking DSCP in different DiffServ regions so as to keep the identical service level across the network.

3. Per-domain resource reservation in BB

   Considering the number of end users in access network could be significantly large, it might not be feasible to keep track of the resource reservation status in BB for every application. It is however practical to have the per-application resource allocation requests managed in their own access networks. BB trusts the access networks that have the SLAs with BB. BB grants QoS request from those trusted domains instead of per-application. This will make the deployment of this architecture more feasible.

Appendix A

RSVP message

A.1 PATH

A Path message contains the following information in addition to the previous hop address:

- Sender Template

  A Path message is required to carry a Sender Template, which describes the format of data packets that the sender will originate. This template is in the form of a filter spec that could be used to select this sender's packets from others in the same session on the same link.

  Sender Templates have exactly the same expressive power and format as filter specs that appear in Resv messages. Therefore a Sender Template may specify only the sender IP address and optionally the UDP/TCP sender port, and it assumes the protocol Id specified for the session.

- Sender Tspec

  A Path message is required to carry a Sender Tspec, which defines the traffic characteristics of the data flow that the sender will generate. This Tspec is used by traffic control to prevent over-reservation, and perhaps unnecessary Admission Control failures.

- Adspec

  A Path message may carry a package of OPWA advertising information, known as an "Adspec". 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.

A.2 RSVP Object Formats

This section specifies the detailed contents and wire format of RSVP SENDER_TSPEC, ADSPEC, and FLOWSPEC objects for use with the Guaranteed and Controlled-Load QoS control services. The object formats specified here are based on the general message construction rules given in Appendix of [5]

-RSVP SENDER_TSPEC Object

The RSVP SENDER_TSPEC object carries information about a data source's generated traffic. The required RSVP SENDER_TSPEC object contains a global Token_Bucket_TSpec parameter (service_number 1, parameter 127, as defined in [RFC 2215]). This TSpec carries traffic information usable by either the Guaranteed or Controlled-Load QoS control services.

<table>
<tr><td>o (a)</td><td colspan="3">reserved</td><td>7 (b)</td></tr>
<tr><td colspan="2">1 (c)</td><td>0</td><td>reserved</td><td>6 (d)</td></tr>
<tr><td colspan="2">127 (e)</td><td colspan="2">0 (f)</td><td>5 (g)</td></tr>
<tr><td colspan="5">Token Bucket Rate [r] D(32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Token Bucket Size [b] (32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Peak Data Rate [p] (32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Minimum Policed Unit [m] (32-bit integer)</td></tr>
<tr><td colspan="5">Maximum Packet Size [M] (32-bit integer)</td></tr>
</table>

(a) - Message format version number (0)

(b) - Overall length (7 words not including header)

(c) - Service header, service number 1 (default/global information)

```
(d) - Length of service 1 data, 6 words not including header
(e) - Parameter ID, parameter 127 (Token_Bucket_TSpec)
(f) - Parameter 127 flags (none set)
(g) - Parameter 127 length, 5 words not including header
```

In this TSpec, the parameters [r] and [b] are set to reflect the sender's view of its generated traffic. The peak rate parameter [p] may be set to the sender's peak traffic generation rate (if known and controlled), the physical interface line rate (if known), or positive infinity (if no better value is available). Positive infinity is represented as an IEEE single-precision floating-point number with an exponent of all ones (255) and a sign and mantissa of all zeros. The format of IEEE floating-point numbers is further summarized in [RFC 1832].

The minimum policed unit parameter [m] should generally be set equal to the size of the smallest packet generated by the application. This packet size includes the application data and all protocol headers at or above the IP level (IP, TCP, UDP, RTP, etc.). The size given does not include any link-level headers, because these headers will change as the packet crosses different portions of the internetwork.

The [m] parameter is used by nodes within the network to compute the maximum bandwidth overhead needed to carry a flow's packets over the particular link-level technology, based on the ratio of [m] to the link-level header size. This allows the correct amount of bandwidth to be allocated to the flow at each point in the net. Note that smaller values of this parameter lead to increased overhead estimates, and thus increased likelyhood of a reservation request being rejected by the node. In some cases, an application transmitting a low percentage of very small packets may therefore choose to set the value of [m] larger than the actual minimum transmitted packet size. This will increase the likelyhood of the reservation succeeding, at the expense of policing packets of size less than [m] as if they were of size [m].

Note that the an [m] value of zero is illegal. A value of zero would indicate that no data or IP headers are present, and would give an infinite amount of link-level overhead.

The maximum packet size parameter [M] should be set to the size of the largest packet the application might wish to generate. This value must, by definition, be equal to or larger than the value of [m].

- RSVP FLOWSPEC Object

The RSVP FLOWSPEC object carries information necessary to make reservation requests from the receiver(s) into the network. This includes an indication of which QoS control service is being requested, and the parameters needed for that service.

The QoS control service requested is indicated by the service_number in the FLOWSPEC's per-service header.

FLOWSPEC object when requesting Controlled-Load service

The format of an RSVP FLOWSPEC object originating at a receiver requesting Controlled-Load service is shown below. Each of the TSpec fields is represented using the preferred concrete representation specified in the 'Invocation Information' section of [23]. The value of 5 in the per-service header (field (c), below) indicates that Controlled-Load service is being requested.

<table>
<tr><td>o (a)</td><td colspan="3">reserved</td><td>7 (b)</td></tr>
<tr><td colspan="2">5 (c)</td><td>0</td><td>reserved</td><td>6 (d)</td></tr>
<tr><td colspan="2">127 (e)</td><td colspan="2">0 (f)</td><td>5 (g)</td></tr>
<tr><td colspan="5">Token Bucket Rate [r] D(32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Token Bucket Size [b] (32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Peak Data Rate [p] (32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Minimum Policed Unit [m] (32-bit integer)</td></tr>
<tr><td colspan="5">Maximum Packet Size [M] (32-bit integer)</td></tr>
</table>

(a) - Message format version number (0)

(b) - Overall length (7 words not including header)

(c) - Service header, service number 5 (Controlled-Load)

(d) - Length of controlled-load data, 6 words not including per-service header

(e) - Parameter ID, parameter 127 (Token Bucket TSpec)

(f) - Parameter 127 flags (none set)

(g) - Parameter 127 length, 5 words not including per-service header

In this object, the TSpec parameters [r], [b], and [p] are set to reflect the traffic parameters of the receiver's desired reservation (the Reservation TSpec). The meaning of these fields is discussed fully in [23]. Note that it is unlikely to make sense for the [p] term to be smaller than the [r] term.

The maximum packet size parameter [M] should be set to the value of the smallest path MTU, which the receiver learns from information in arriving RSVP ADSPEC objects. Alternatively, if the receiving application has built-in knowledge of the maximum packet size in use within the RSVP session, and this value is smaller than the smallest path MTU, [M] may be set to this value. Note that requesting a value of [M] larger than the service modules along the data path can support will cause the reservation to fail.

The value of [m] can be chosen in several ways. Recall that when a resource reservation is installed at each intermediate node, the value used for [m] is the smaller of the receiver's request and the values in each sender's SENDER_TSPEC.

If the application has a fixed, known minimum packet size, than that value should be used for [m]. This is the most desirable case.

For a shared reservation style, the receiver may choose between two options, or pick some intermediate point between them.

- if the receiver chooses a large value for [m], then the reservation will allocate less overhead for link-level headers. However, if a new sender with a smaller SENDER_TSPEC [m] joins the session later, an already-installed reservation may fail at that time.

- if the receiver chooses a value of [m] equal to the smallest value which might be used by any sender, then the reservation will be forced to allocate more overhead for link-level headers. However it will not fail later if a new sender with a smaller SENDER_TSPEC [m] joins the session.

For a FF reservation style, if no application-specific value is known the receiver should simply use the value of [m] arriving in each sender's SENDER_TSPEC for its reservation request to that sender.

FLOWSPEC Object when Requesting Guaranteed Service

The format of an RSVP FLOWSPEC object originating at a receiver requesting Guaranteed service is shown below. The flowspec object used to request guaranteed service carries a TSpec and RSpec specifying the traffic parameters of the flow desired by the receiver.

Each of the TSpec and RSpec fields is represented using the preferred concrete representation specified in the "Invocation Information" section of [22]. The value of 2 for the service header identifier (field (c) in the picture below) indicates that Guaranteed service is being requested.

<table>
<tr><td>o (a)</td><td colspan="3">Unused</td><td>10 (b)</td></tr>
<tr><td colspan="2">2 (c)</td><td>0</td><td>reserved</td><td>9 (d)</td></tr>
<tr><td colspan="2">127 (e)</td><td colspan="2">0 (f)</td><td>5 (g)</td></tr>
<tr><td colspan="5">Token Bucket Rate [r] D(32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Token Bucket Size [b] (32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Peak Data Rate [p] (32-bit IEEE floating point number)</td></tr>
<tr><td colspan="5">Minimum Policed Unit [m] (32-bit integer)</td></tr>
<tr><td colspan="5">Maximum Packet Size [M] (32-bit integer)</td></tr>
</table>

(a) - Message format version number (0)

(b) - Overall length (9 words not including header)

(c) - Service header, service number 2 (Guaranteed)

(d) - Length of per-service data, 9 words not including per-service header

(e) - Parameter ID, parameter 127 (Token Bucket TSpec)

(f) - Parameter 127 flags (none set)

(g) - Parameter 127 length, 5 words not including parameter header

(h) - Parameter ID, parameter 130 (Guaranteed Service RSpec)

(i) - Parameter 130 flags (none set)

```
(j) - Parameter 130 length, 2 words not including parameter header
```

In this object, the TSpec parameters [r], [b], and [p] are set to reflect the traffic parameters of the receiver's desired reservation (the Reservation TSpec). The meaning of these fields is discussed fully in [22]. Note that it is unlikely to make sense for the [p] term to be smaller than the [r] term.

The RSpec terms [R] and [S] are selected to obtain the desired bandwidth and delay guarantees. This selection is described in [22].

The [m] and [M] parameters are set identically to those for the Controlled-Load service FLOWSPEC, described in the previous section.

- RSVP ADSPEC Object

An RSVP ADSPEC object is constructed from data fragments contributed by each service which might be used by the application. The ADSPEC begins with an overall message header, followed by a fragment for the default general parameters, followed by fragments for every QoS control service which may be selected by application receivers. The size of the ADSPEC varies depending on the number and size of per-service data fragments present and the presence of non-default general parameters.

The complete absence of a data fragment for a particular service means that the application sender does not know or care about that service, and is a signal to intermediate nodes not to add or update information about that service to the ADSPEC. It is also a signal to application receivers that they should not select that service when making reservations.

Each fragment present is identified by a per-service data header. Each header contains a field identifying the service, a break bit, and a length field.

The length field allows the ADSPEC information for a service to be skipped over by a network elements which does not recognize or implement the service. When an element does this, it sets the break bit, indicating that the service's ADSPEC data was not updated at at least one hop. Note that a service's break bit can be set without otherwise supporting the service in any way. In all cases, a network element encountering a per-service data header it does not understand simply sets bit 23 to report that the service is not supported, then skips over the rest of the fragment.

Data fragments must always appear in an ADSPEC in service_number order. In particular, the default general parameters fragment (service_number 1) always comes first.

Within a data fragment, the service-specific data must alway come first, followed by any non-default general parameters which may be present, ordered by parameter_number. The size and structure of the service-specific data is fixed by the service definition, and does not require run-time parsing. The remainder of the fragment, which carries non-default general parameters, varies in size and structure depending on which, if any, of these parameters are present. This part of the fragment must be parsed by examining the per-parameter headers.

Since the overall size of each data fragment is variable, it is always necessary to examine the length field to find the end of the fragment, rather than assuming a fixed-size structure.

- RSVP ADSPEC format

The basic ADSPEC format is shown below. The message header and the default general parameters fragment are always present. The fragments for Guaranteed or Controlled-Load service may be omitted if the service is not to be used by the RSVP session. Additional data fragments will be added if new services are defined.

<table>
<tr><td>o (a)</td><td>reserved</td><td>Msg length - 1 (b)</td></tr>
<tr><td colspan="3">Default General Parameters fragment (Service 1) (c)
(Always Present)</td></tr>
</table>

```
(a) - Message format version number (0)
(b) - Overall message length not including header word
(c, d, e) - Data fragments
```

- Default General Characterization Parameters ADSPEC data fragment

All RSVP ADSPECs carry the general characterization parameters defined in [RFC 2215]. Values for global or default general parameters (values which apply to the all services or the path itself) are carried in the per-service data fragment for service number 1, as shown in the picture above. This fragment is always present, and always first in the message.

1 (c)	x	reserved	8 (d)
4 (e)	(f)		1 (g)
IS hop cnt (32-bit unsigned integer)			
6 (h)	(i)		1 (j)
Path b/w estimate (32-bit IEEE floating point number)			
8 (k)	(l)		1 (m)
Minimum path latency (32-bit integer)			
10 (n)	(o)		1 (p)
Composed MTU (32-bit unsigned integer)			

(c) - Per-Service header, service number 1 (Default General Parameters)

(d) - Global Break bit ([RFC 2215], Parameter 2) (marked x) and length of General Parameters data block.

(e) - Parameter ID, parameter 4 (Number-of-IS-hops param from [RFC 2215])

(f) - Parameter 4 flag byte

(g) - Parameter 4 length, 1 word not including header

(h) - Parameter ID, parameter 6 (Path-BW param from [RFC 2215])

(i) - Parameter 6 flag byte

(j) - Parameter 6 length, 1 word not including header

(k) - Parameter ID, parameter 8 (minimum path latency from [RFC 2215])

(l) - Parameter 8 flag byte

(m) - Parameter 8 length, 1 word not including header

(n) - Parameter ID, parameter 10 (composed path MTU from [RFC 2215])

(o) - Parameter 10 flag byte

(p) - Parameter 10 length, 1 word not including header

Rules for composing general parameters appear in [RFC 2215].

In the above fragment, the global break bit (bit 23 of word 1, marked with (x) in the picture) is used to indicate the existence of a network element not supporting QoS control services somewhere in the data path. This bit is cleared when the ADSPEC is created, and set to one if a network element which does not support RSVP or integrated

services is encountered. An ADSPEC arriving at a receiver with this bit set indicates that all other parameters in the ADSPEC may be invalid, since not all network elements along the path support updating of the ADSPEC.

The general parameters are updated at every network node which supports RSVP:

- When a PATH message ADSPEC encounters a network element implementing integrated services, the portion of the ADSPEC associated with service number 1 is passed to the module implementing general parameters. This module updates the global general parameters.

- When a PATH message ADSPEC encounters a network element that does not support RSVP or implement integrated services, the break bit in the general parameters service header must be set. In practice, this bit will usually be set by another network element which supports RSVP, but has been made aware of the gap in integrated services coverage.

- In either case, the ADSPEC is passed back to RSVP for delivery to the next hop along the path.

- Guaranteed Service ADSPEC data fragment

The Guaranteed service uses the RSVP ADSPEC to carry data needed to compute the C and D terms passed from the network to the application. The minimum size of a non-empty guaranteed service data fragment is 8 32-bit words. The ADSPEC fragment for Guaranteed service has the following format:

2 (a)	x	reserved	N-1 (b)
133 (c)	0 (d)		1 (e)
End-to-end composed value for C [Ctot] (32-bit integer)			
134 (f)	(g)		1 (h)
End-to-end composed value for D [Dtot] (32-bit integer)			
135 (i)	(j)		1 (k)
Since-last-reshaping point composed C [Csum] (32-bit integer)			
136 (l)	(m)		1 (n)
Since-last-reshaping point composed D [Dsum] (32-bit integer)			
Service-specific general parameter headers/values, if present			

(a) - Per-Service header, service number 2 (Guaranteed)

(b) - Break bit and Length of per-service data in 32-bit words not including header word.

(c) - Parameter ID, parameter 133 (Composed Ctot)

(d) - Parameter 133 flag byte

(e) - Parameter 133 length, 1 word not including header

(f) - Parameter ID, parameter 134 (Composed Dtot)

(g) - Parameter 134 flag byte

(h) - Parameter 134 length, 1 word not including header

(i) - Parameter ID, parameter 135 (Composed Csum).

(j) - Parameter 135 flag byte

(k) - Parameter 135 length, 1 word not including header

(l) - Parameter ID, parameter 136 (Composed Dsum).

(m) - Parameter 136 flag byte

(n) - Parameter 136 length, 1 word not including header

When a node which actually implements guaranteed service creates the guaranteed service adspec fragment, the parameter values are set to the local values for each parameter. When an application or network

element which does not itself implement guaranteed service creates a guaranteed service adspec fragment, it should set the values of each parameter to zero, and set the break bit to indicate that the service is not actually implemented at the node.

An application or host RSVP which is creating a guaranteed service adspec fragment but does not itself implement the guaranteed service may create a truncated "empty" guaranteed adspec fragment consisting of only a header word:

2 (a)	1	(b)	0 (c)

```
(a) - Per-Service header, service number 2 (Guaranteed)
(b) - Break bit (set, service not implemented)
(c) - Length of per-service data in 32-bit words not
      including header word.
```

This might occur if the sending application or host does not do resource reservation iself, but still wants the network to do so. Note that in this case the break bit will always be set, since the creator of the adspec fragment does not itself implement guaranteed service.

When a PATH message ADSPEC containing a per-service header for Guaranteed service encounters a network element implementing Guaranteed service, the guaranteed service data fragment is updated:

- If the data block in the ADSPEC is an empty (header-only) block the header-only fragment must first be expanded into the complete data fragment described above, with initial values of Ctot, Dtot, Csum, and Dsum set to zero. An empty fragment can be recognized quickly by checking for a size field of zero. The value of the break bit in the header is preserved when the additional Guaranteed service data is added. The overall message length and the guaranteed-service data fragment size (field (b) in the pictures above) are changed to reflect the increased message length.

The values of Ctot, Csum, Dtot, and Dsum in the ADSPEC data fragment are then composed with the local values exported by the network element according to the composition functions defined in [22].

- When a PATH message ADSPEC with a Guaranteed service header encounters a network element that supports RSVP but does not implement Guaranteed service, the network element sets the break bit in the Guaranteed service header.

- The new values are placed in the correct fields of the ADSPEC, and the ADSPEC is passed back to RSVP for delivery to the next hop along the path.

When a PATH message ADSPEC containing a Guaranteed service data fragment encounters a network element that supports RSVP but does not implement Guaranteed service, the network element sets the break bit in the Guaranteed service header.

When a PATH message ADSPEC without a Guaranteed service header encounters a network element implementing Guaranteed service, the Guaranteed service module of the network element leaves the ADSPEC unchanged. The absence of a Guaranteed service per-service header in the ADSPEC indicates that the application does not care about Guaranteed service.

- Controlled-Load Service ADSPEC data fragment

Unlike the Guaranteed service, the Controlled-Load service does not require extra ADSPEC data to function correctly. The only ADSPEC data specific to the Controlled-Load service is the Controlled-Load break bit. Therefore the usual Controlled-Load service data block contains no extra information. The minimum size of the controlled-load service data fragment is 1 32-bit word.

5 (a)	x	(b)	N-1 (c)
Service-specific general parameter headers/values, if present			

```
(a) - Per-Service header, service number 5 (Controlled-Load)
(b) - Break bit
(c) - Length of per-service data in 32 bit words not including
      header word.
```

The Controlled-Load portion of the ADSPEC is processed according to the following rules:

- When a PATH message ADSPEC with a Controlled-Load service header encounters a network element implementing Controlled-Load service, the network element makes no changes to the service header.

- When a PATH message ADSPEC with a Controlled-Load service header encounters a network element that supports RSVP but does not implement Controlled-Load service, the network element sets the break bit in the Controlled-Load service header.

- In either case, the ADSPEC is passed back to RSVP for delivery to the next hop along the path.

A.3 BB messages

A.3.1 RAR

The RAR message format is outlined in Table 1. Note that not all of the fields are used in an RAR sent between end systems and bandwidth brokers (i.e., intra-domain).

Field	Explanation.
Version	Bandwidth broker protocol version ID (current version is 1)
RAR ID	Unique RAR ID (perhaps IP address + sequence number) generated by initial RAR sender and propagated forward; may be used for bookkeeping purposes by any intermediate BB; must be returned in matching RAA message.
Sender ID	Identifier of the DS domain that sent the RAR; rewritten by intermediate domains; used to authenticate the RAR.For RARs sent to or from end systems, this field is not used.
Sender Signature	Each RAR message should be signed with the public key of the sending DS domain; this field in conjunction with the Sender ID allows the RAR receiver to authenticate that the RAR is from a peer DS domain and to reference internal state on the SLS in place with that domain
Source Prefix	IP address prefix for source terminus of the service request
Destination Prefix	IP address prefix for destination terminus of service request

Ingress Router ID	IP address of the interface between two domains for which the sending domain is requesting service. This field is replaced in the message by each sending bandwidth broker. When sent by an end-system, this field contains the IP address of the access router interface through which the flow will pass (for example, the default router) en route to the destination. When sent from a bandwidth broker to an end system, it contains the IP address of the access router interface over which the flow will be forwarded.
Start Time	now — specific future time
Stop Time	indefinite — as long as possible — specific future time
Flags	The following flags are defined: receiver pays (collect call) probe (determine parameters/acceptance but do not commit resources) establish Core Tunnel renegotiation delta/absolute values for Service Parameterization Object (SPO) increment/decrement values for SPO
GSID	Globally well-known service ID
SPO	Service specification parameters dependent on the particular GWS indicated by the GSID.
Additional TLVs	Core Tunnel Voucher

Table 1: RAR message format

A.3.2 RAA

Corresponding to each RAR generated, is an RAA message, each having the format shown in Table 2:

Field	Explanation
Version	Bandwidth broker protocol version ID (current version is 1)

RAR ID	Unique RAR ID (perhaps IP address + sequence number) generated by initial RAR sender and propagated forward; may be used for bookkeeping purposes by any intermediate BB; must be returned in matching RAA message
Sender ID	Identifier of the DS domain that sent the RAR; rewritten by intermediate domains; used to authenticate the RAR.For RARs sent to or from end systems, this field is not used.
Sender Signature	Each RAR message should be signed with the public key of the sending DS domain; this field in conjunction with the Sender ID allows the RAR receiver to authenticate that the RAR is from a peer DS domain and to reference internal state on the SLS in place with that domain
Source Prefix	Copied from RAR
Destination Prefix	Copied from RAR
Ingress Router ID	Copied from RAR as received by this bandwidth broker
Start Time	Copied from RAR
Stop Time	Copied from RAR. If 'as long as possible' was specified in the RAR, then this may be set to a specific future time.
Flags	The following flags are defined: RAR Accepted. If this bit is set (on) then the RAR has been accepted and the learned service parameters may be found in the SPO; if this bit is off, the RAR was rejected and the SPO may optionally be rewritten to reflect the nearest match reservation that would have been accepted.Additionally, a reason code TLV may be included following the SPO. Core tunnel set up. This bit indicates that a core tunnel was set up as a result of the associated RAR and that there is a 'voucher' TLV contained in this message.
GSID	Copied from the RAR

SPO	Service specification parameters dependent on the particular GWS indicated by the GSID; parameters that were left blank in the RAR may be completed in the RAA or rewritten to reflect a renegotiation hint as described in the Flags field above.
Additional TLVs	Reason Code TLV Core Tunnel Voucher TLV

Table 2: RAA message format

A.3.3 CANCEL

Table 3 gives the important fields of the CANCEL message.

Field	Explanation
Version	Bandwidth broker protocol version ID (current version is 1)
CANCEL ID	Unique CANCEL ID (perhaps IP address + timestamp) generated by initial CANCEL sender and propagated forward; may be used for bookkeeping purposes by any intermediate BB; must be returned in matching CANCEL ACK message
Sender ID	Identifier of the DS domain that sent the CANCEL; rewritten by intermediate domains; used to authenticate the CANCEL.
Sender Signature	Each CANCEL message should be signed with the public key of the sending DS domain; this field in conjunction with the Sender ID allows the CANCEL receiver to authenticate that the CANCEL is from a peer DS domain.
Flags	Upstream or Downstream, indicates the direction that the CANCEL is flowing.
CANCEL List	List of reservations to be cancelled.

Table 3: CANCEL message fields

A.3.4 CANCEL ACK

Table 4 gives the important fields of the CANCEL ACK message.

Field	Explanation
Version	Bandwidth broker protocol version ID (current version is 1)
CANCEL ID	Unique CANCEL ID (perhaps IP address + timestamp) generated by initial CANCEL sender and identifies the CANCEL to which this ACK refers.
Sender ID	Identifier of the DS domain that sent the CANCEL ACK; rewritten by intermediate domains; used to authenticate the CANCEL ACK.
Sender Signature	Each CANCEL ACK message should be signed with the public key of the sending DS domain; this field in co njunction with the Sender ID allows the CANCEL ACK receiver to authenticate that the CANCEL ACK is from a peer DS domain.

Table 4: CANCEL ACK message fields

!b

Version	Reserved
Sender ID	
Sender Signature	
PDU length	
SIBBS message	

Table 5: SIBBS PDU Header

A.3.5 SIBBS PDU message

SIBBS PDU header

```
Version (8 bit)
One octet unsigned integer containing the version number of the protocol.
        00000001        SIBBS version 1 PDU.
        All other settings are reserved for future use.

Reserved (24 bit)
        Reserved for future use

Sender ID (64 bit)
        ASCII string identifying the DS domain that sent the PDU.

Sender Signature
        TBD octet string based on the certificates exchanged during
the start of operations.

PDU length (32 bit)
        32 bit unsigned integer specifying the total length of this
PDU in 32 bit words, excluding the Version, Sender ID, and
PDU length fields. The maximum PDU Length is 64 kbytes.
```

U	Message Type	Message Length
Mandatory Parameters		
Optional Parameters		
PDU length		
SIBBS message		

Table 6: Inter domain bandwidth broker message format

SIBBS message

Integer multiple of 32 bits field containing the SIBBS message.

The SIBBS PDU message format is shown in Table 6:

U bit

Unknown message bit. Upon receipt of an unknown message, if U is clear (=0), a notification is returned to the message originator; if U is set (=1), the unknown message is silently ignored. The sections following that define messages specify a value for the U- bit.

Message Type

Identifies the type of message

Message Length

Specifies the cumulative length in 32 bit words of the Message ID, Mandatory Parameters, and Optional Parameters. The maximum message length is 64 kbytes.

Mandatory Parameters

Variable length set of required message parameters. Some messages have no required parameters.

For messages that contain required parameters, the required parameters MUST appear in the order specified by the individual message specifications in the section Messages.

Optional Parameters

Variable length set of optional message parameters. Many messages have no optional parameters. For messages that have optional parameters, the optional parameters may appear in any order.

Parameters in SIBBS PDU message

- The version is SIBBS version 1
- Sender ID is set to the domain name of the sending BB's domain
- The Sender Signature is created based on the message contents and on the security model to be created by Andy Adamson and Volker Sander.
- The SIBBS PDU contains a RAR message, consisting of mandatory parameters only (i.e. no optional parameters included).
- The first parameter of the RAR message is the RAR ID:
  - Type is normally default: IPv4
  - IPv4 address of the BB creating this message
  - The sequence number is created from stable storage by adding 1 to the value there
  - Since this is the first RAR associated with the reservation, the extension number is 0
  - The Source Address prefix is the originating host (It could also be the longest prefix of its IPv4 address that the domain needs to make available for the application of policy).
  - The destination address prefix in this case is the full IPv4 address of the destination host.

- The next parameter is the Ingress Router ID:

  – The ingress router ID is derived from the pinned egress router for the originating domain. In this case, it is the IP address of the ingress interface of the domain to which the RAR is being forwarded. This gives the receiving BB the exact link and router that will be used as the ingress to its domain.

  – The next parameter of the RAR is the SPO:

  – The SPO service type is set to default: 0x0001

  – The P-Bit is set to b'1' indicating a globally defined service.

  – The service-specific parameters of the SPO are set as follows:

  – PHB code is set to default: the EF DSCP code (RFC 2598)

  – The DSCP is set to default: EF (b'101110')

  – The Peak rate is set to the number of octets per second desired by the reservation

  – We don't define any experimental values for release 1

  – The start time vector is omitted (operational default).

  – The end time vector is omitted. (operational default - Implies that the reservation will be explicitly taken down by the originating BB).

  – The RAR Flags TLV is omitted (operational default).

Bibliography

[1] Chris Metz, General-Purpose Signaling for IP, Cisco System, 1999

[2] G. Huston, Next Steps for the IP QoS Architecture, RFC 2990, November 2000

[3] Y. Bernet, P. Ford, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Bradeni, B. Daviei, J. Wroclawski, E. Felstaine, A Framework for Integrated Services Operation over Diffserv Networks, RFC 2998, November 2000

[4] QoS protocol and architecture, Q0S Forum, 1999

[5] J. Wroclawski, The Use of RSVP with IETF Integrated Services, RFC 2210, September 1997

[6] J. Wroclawski, A. Charny, Integrated Service Mappings for Differentiated Services Networks, Internet Draft draft-ietf-issll-ds-map-01.txt, February, 2001

[7] S. Shenker and J. Wroclawski, General Characterization Parameters for Integrated Service Network Elements, RFC 2215, September 1997.

[8] F. Baker, C. Iturralde, F. Le Faucheur, B. Davi, Aggregation of RSVP for IPv4 and IPv6 Reservations, RFC 3175, September 2001

[9] Rob Neilson, Jeff Wheeler, Francis Reichmeyer, Susan Hares, A Discussion of Bandwidth Broker Requirements for Internets Qbone Deployment, Internet2 Qbone BB Advisory Council, August, 1999

[10] QBone Bandwidth Broker Architecture, Internet2 Qbone, June 2002

[11] J. Heinanen, Telia Finlandi, F. Baker, W. Weiss, J. Wroclawski, Assured Forwarding PHB Group, RFC 2597, June 1999

[12] B. Davie, A. Charny, J.C.R Bennett, K. Benson, J.Y.Le Boudec, W. Courtney, S. Davari, V. Firoiu, D.Stiliadis, An Expedited Forwarding PHB (Per-Hop-Behavior), RFC 3246, March 2002.

[13] S. Blake, D. Black,M. Carlson, E. Davies, Z. Wang, W. Weiss, An Architecture for Differentiated Services, RFC 2475, December 1998

[14] A. Mankin, et al, Resource ReSerVation Protocol (RSVP) Version 1 Applicability Statement - Some Guidelines on Deployment, RFC 2208, September 1997

[15] The IP QoS FAQ, The Quality of Services Forum, 1999

[16] E. Rosen, A. Viswanathan, R. Callon, Multiprotocol Label Switching Architecture, RFC 3031, January 2001

[17] R. Callon, N. Freedman, A. Fredette, G. Swallow, A. Viswanathan, A Framework for Multiprotocol Label Switching, RFC 3353, August 2002

[18] J. Heinanen, Differentiated Services in MPLS Networks, draft-heinenen-diffserv-mpls-00.txt, June 1999

[19] D. Awduche, L. Berger, D. Gan, T. Li, G. Swallow, V. Srinivasan, Extensions to RSVP for LSP Tunnels, Internet Draft draft-ietf-mpls-rsvplsp -tunnel-08.txt, February 2001

[20] Braden, R., Zhang, L., Berson, S., Herzog, S. and S. Jamin, Resource ReSerVation Protocol (RSVP) Version 1 - Functional Specification, RFC 2205, September 1997.

[21] D. Durham, Ed, J. Boyle, R. Cohen, S. Herzog, R. Rajan, A. Sastry, The COPS (Common Open Policy Service) Protocol, RFC 2748, January 2000

[22] S. Shenker and C. Partridge and R. Guerin, Specification of Guaranteed Quality of Service, RFC 2212, September 1997

[23] J. Wroclawski, Specification of the Controlled-Load Network Element Service, RFC 2211, September 1997

[24] K. Nichols and S. Blake and F. Baker and D. Black, Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, RFC 2474, December 1998.

[25] QBone Signaling Design Team Final Report, Internet2 Qbone, 2002