CONTENT-AWARE QoS MANAGEMENT FOR MULTIMEDIA APPLICATIONS IN A DIFFERENTIATED SERVICES ENVIRONMENT

by

MANISH K. MAHAJAN

A thesis submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

written under the direction of

Professor. Manish Parashar

and approved by

____

____

____

____

New Brunswick, New Jersey

May, 2002

ABSTRACT OF THE THESIS

Content-Aware QoS Management for Multimedia Applications in a Differentiated Services Environment

By MANISH MAHAJAN

Thesis Director:
Professor Manish Parashar

The explosive growth of the Internet coupled with bandwidth hungry, timing-sensitive and mission-critical network applications presents a formidable challenge to provide reliable Quality of Service (QoS) guarantees in an environment of finite resources. Networked multimedia applications generate traffic at varying rates and have a varying ability to tolerate delays and jitter. These applications require time bound processing and by their very nature ignore congestion related feedback from the network. Internet Protocol (IP) does not provide any QoS guarantees as there are no mechanisms in IP for policing and controlling unresponsive and high bandwidth flows that can cause congestion. Consequently, QoS management for networked multimedia applications over IP is a significant and immediate challenge.

This thesis presents design, implementation and evaluation of a content-aware bandwidth broker (CABB) for the differentiated services (diffserv) environment. The bandwidth broker provisions resources and controls different multimedia flows between diffserv domains using tiered service offerings, bandwidth on demand, usage-based billing, service policies defined based on client requirement, and tolerant adaptability of the client's application. CABB builds on the observation that multimedia applications are flexible with respect to network parameters such as packet loss, delay and jitter. CABB exploits this flexibility of multimedia flows to network level parameters to adapt the flows based on the state of network resources to maintain some level of QoS despite of unfavorable network conditions. For example, when the application's demand for resources exceeds availability, rather than refuse allocation to the application, CABB may admit and maintain the flow at a reduced QoS until the required resources become available. This can be significant for time critical applications. Furthermore, in case of network congestion, CABB and adapt to the network state and reduce QoS rather than completely disrupting the flow. CABB also prevent non-confirming (or rogue) flows from affecting the performance for conforming flows by constantly monitoring and gradually degrading the level of service for the rogue flows. Thus it provides the incentive in support of end-to-end congestion control for best effort traffic.

CABB is implemented and evaluated using the NS-2 simulator toolkit. Our implementation builds on the diffserv model provided by Nortel Networks and provides intra- and inter-domain brokering for simulated streaming multimedia applications using. Results show that by exploiting flexible nature of multimedia flows, CABB improves network resource allocations. The results also show that multimedia flows are better managed and controlled, thereby improving perceived QoS and avoiding possible congestion. Flow throughputs also increase as CABB enables more flows to be admitted.

Acknowledgement and/or Dedication

I thank my advisor, Dr. Manish Parashar, for his guidance, encouragement and support to conduct this research. I am grateful for his continuous support to understanding the principles of my specific research area. Also, I am grateful for Dr. Parashar's invaluable help in improving my technical writing and presentation skills.

I also thank all my committee members. I acknowledge their suggestions and guidance in developing my technical understanding for this interesting research topic. In particular I am grateful for the internship opportunity at Sun Microsystems which significantly enriched my research skills and my understanding of the research community.

In addition, I'd like to thank my friend Sonal Dhingra for her permanent willingness to decode my English and Diwakar Kedlaya for his helpful hints on my presentations. Also, I thank my friends at Buell and TASSL lab for their superb company and support.

Finally, I thank my family for their love and constant encouragement throughout my life.

Table of Contents

List of illustrations

Chapter 1

Introduction

1.1 Objective

The objective of this thesis is to develop a content aware bandwidth broker that enables multimedia applications to perform satisfactorily under constrained system and network resource availability. The broker should be able to provision resources and control different flows (multimedia traffic) between diffserv domains using tiered service offerings, bandwidth on demand, usage-based billing, service policies defined based on client requirement and the client applications' tolerant adaptability to network level parameters of packet delay, loss and jitter. It should manage the network so as to give guaranteed/assured service to application flows with a gradual degradation of service with increasing traffic in light of client's service level agreement (SLA).

1.2 Background

The tremendous popularity of the Internet has come with increasing diversity and heterogeneity in terms of client device capability, network bandwidth, and user preferences. This has led to emergence of a new generation of networked applications with widely varying characteristics and requirements. Video Conferencing, Video-on-Demand and IP telephony are a few of these successful commercial networked applications. Those that have timeliness constraints are called real-time applications. Among real-time applications there are those that are tolerant or intolerant depending on whether they can tolerate occasional loss. Such real-time applications are competing with traditional Internet applications – email, file transfer for network level resources such as bandwidth and queue buffers. They demand high bandwidth and assurance of timeliness of data delivery from the underlying network

IP provides best effort data delivery service, which allows the complexity to stay in the end-hosts, so the core network can remain simple. It depends on higher layers of the protocol stack to satisfy other application specific data transfer constraints such as reliability, latency and consistency of data throughput. This has proved to be a robust and scalable solution as evidenced by the ability of the Internet to support more networks and hosts for traditional Internet applications such as email, file transfer and other web applications.

As more hosts are connected, network service demands eventually exceed capacity, thereby denying service to hosts or applications. The increase in distributed multimedia applications such as VoIP (voice over IP), poses a significant challenge for network engineering for integration of such applications with an array of complex data applications, each with different service requirements. These applications typically operate in heterogeneous environment where the network resources and end-host processing capabilities vary significantly. Since the states of the network and end host system are dynamic, distributed multimedia applications have to contend with unpredictable resource availability.

1.3 Problem Description

The overall quality of network connections (e.g. link capacity, available end-to-end bandwidth, congestion, etc) has a significant impact on the performance of networked applications. These applications generate traffic at varying rates and have a varying ability to tolerate delays and jitter in the network. Many networked multimedia applications are delay-sensitive, and require services with guarantees of resource availability and timeliness. Multimedia applications have time bounded processing and communication requirements, primarily due to the coding and compression techniques involved, impose temporal dependencies on media. Playback procedures typically involve reproduction of multiple media in a tightly synchronized manner. This requires temporal and spatial guarantees from the underlying network. They present a significant challenge, as they require quality guarantees from the network.

QoS guarantees are needed at multiple layers in an end-to-end protocol architecture, which means delivering end-to-end QoS requires architecture for resource management at the system end-points (e.g., computer workstation hosts), as well as in the underlying network. Different networks have varying frame sizes requiring additional conversion overheads at the gateways. Furthermore, they have different scheduling policies and their interconnectivity results in multiplexing and demultiplexing of traffic. The best effort Internet Protocol (IP) does not provide any Quality of Service (QoS) guarantees [4] – that is, there are no mechanisms in IP for policing or controlling unresponsive and high bandwidth flows that can cause congestion in the network. As a result, all QoS management is left to the application [9]. Multimedia applications have very limited feedback control to stop them from causing congestion in the network. Consequently, QoS management for networked multimedia applications over IP is a significant and immediate challenge.

However, these applications exhibit a common characteristic that they operate satisfactorily in less than ideal situations by allowing for a tradeoff between certain service requirements. These applications are flexible with respect to network resource requirements and are willing to sacrifice the performance of some quality parameters in order to preserve the quality of critical parameters. For example, one approach taken to leverage this property is rate adaptiveness where video-coding algorithms can trade off bit rate versus quality for a highly loaded network. Content-aware brokering enables operation of these multimedia applications with acceptable performance despite insufficient network and end-system resources and provides a useful tool for flow allocation and control.

1.4 Internet QoS

Quality of Service (QoS) can be broadly defined as the degree of user satisfaction. Network QoS refers to the ability of the network to handle traffic such that it meets the service needs of certain applications. This requires fundamental traffic handling mechanisms in the network, the ability to identify traffic that is entitled to these mechanisms and the ability to control these mechanisms. Any QoS assurance is only as good as the weakest link in the "chain" between sender and receiver. So QoS is fundamentally an end-to-end issue implying that QoS assurances have to be configurable, predictable and maintainable from source to destination. This means that it should be relevant over all architectural layers from source media devices down the protocol stack across the network element and up the receiver protocol stack to playback devices. Consequently, the issue of QoS can be addressed at different levels of the network protocol stack including:

- User level by specifying (qualitatively or quantitatively) user perceivable service parameters.
- Application level by ensuring that the application adapts according to the network and system resource availability.
- Network level by defining traffic models, classification of service disciplines, and resource reservation on a per-flow or flow aggregate basis to ensure that the applications' resource requirements are met.

Our work builds on the considerable research effort directed in handling QoS at the network level. Some of the promising approaches are discussed in the chapter on related work. Different applications have different requirements regarding handling of their network traffic. These requirements are expressed using QoS related parameters such as bandwidth, latency, jitter and loss. Content aware brokering utilizes an applications' adaptability to these parameters to allocate and maintain optimum QoS under existing resource constraints. The broker is sensitive to the needs of the application and possesses knowledge about the network and system resources for maintaining end-to-end QoS for the application.

1.5 Overview of Thesis

This thesis presents a content aware bandwidth broker (CABB) that provides content adaptive brokering to allocate network resources among responsive (TCP) and unresponsive (UDP) flows. It builds on the observation that multimedia applications are adaptive (flexible) in terms of network parameters such as packet loss, delay and jitter. For example, a multimedia application flow may be tolerant or intolerant to packet loss [5]. CABB understands nature of information being transmitted, the flow's requirements and flexibility to network level parameters including packet loss tolerance. It uses the content information to adapt multimedia flows to maintain some level of QoS for these flows rather than refusing them allocations or disrupt their flow when a rogue flow causes congestion. This is done such that when the demand for resources exceeds availability the flow is allowed and maintained by reducing level of quality. The goal of the allocation is to ensure fair resource utilization for all flows, give them guaranteed/assured QoS by delivering them (especially multimedia) in a timely manner in the event of high congestion in one or more links along the path. It prevents non-confirming (or rogue) flows from affecting the performance for conforming flows by constantly monitoring network condition with a gradual degradation of service for rogue flows. Thus it provides the incentive in support of end-to-end congestion control for best effort traffic.

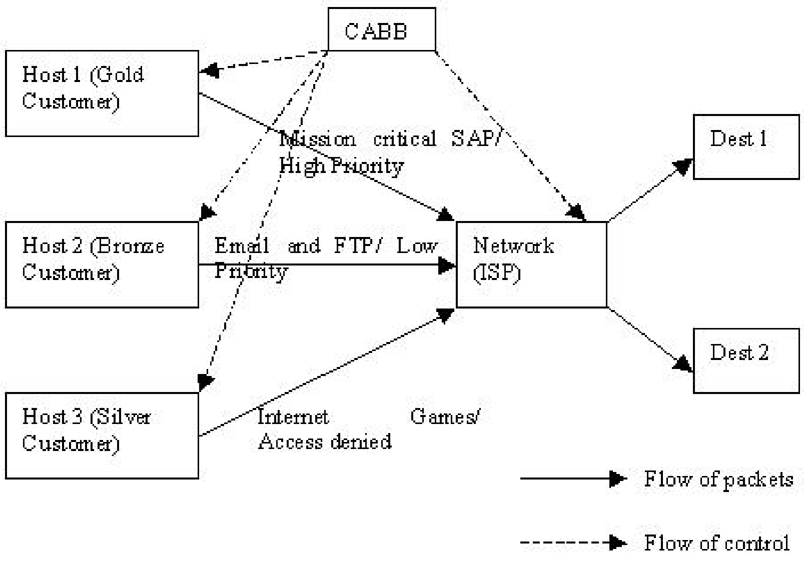


Figure 1.1 Generic view of CABB

Fig 1.1 shows a generic view of CABB along with flow allocation results. Results show that the content aware broker uses information about the flexible nature of multimedia flows to improve resource allocation. The flow throughput increases as more flows can contend for resources that were previously over allocated. It also shows that the amount of such flows arising in the network can be better controlled thus avoiding possible congestion.

1.6 Contributions

The main contributions of this thesis are:

- Design and implementation of a content-aware bandwidth broker (CABB) to provide content adaptive brokering for a better QoS to end users of multimedia applications in heterogeneous environments. Mechanisms are provided for inter-broker communication to reserve resources till destination.
- Policy algorithms that allocate and reserve resources based on the flexibility of the application to network level parameters such as delay, loss and jitter.
- Experimental study of flexible (loss based) adaptations for streaming applications

1.7 Outline of Thesis

Chapter 2 presents related and background work that discusses various approaches adopted by the research community, and identifies different issues being addressed by various research groups to provide Quality of Service for distributed multimedia applications over heterogeneous networks.

Chapter 3 outlines the design and implementation of the content aware bandwidth broker. A modular approach is followed to facilitate adding functionality to the mechanism in a phased manner. Responsibilities of the different modules and their interactions are specified.

Chapter 4 describes the experimentation setup and a simulation based evaluation of the content aware broker. The assumptions made for the simulation setup are noted and the results are plotted. The deductions from results obtained are discussed.

Chapter 5 summarizes the research work and discusses directions for future work.

Chapter 2

Background and Related work

2.1 Network Protocols with QoS support

Different approaches have been proposed to provide service guarantees to multimedia applications. The approaches can be broadly divided into network level protocols, reservation based schemes and adaptation based schemes. Reservation based schemes reserve network/system resources based on the application's requirements and are typically accompanied by admission control schemes to match application's request with existing resource availability. Adaptation based schemes utilize the adaptive behavior of applications that do not require hard service guarantees, to provide them with a better than best effort service by performing application aware active resource management with runtime adaptations. Network level protocols provide QoS support by interpreting the application's requirements in terms of network parameters and enhancing the network switches to service application flows or flow aggregates according to the assigned service levels. A number of network level QoS protocols have evolved to satisfy the variety of application needs. A brief description of these protocols is given below. These protocols can be applied to individual application "flows" or to flow aggregates.

- Per Flow: A "flow" is defined as an individual, uni-directional, data stream between applications (sender and receiver), uniquely identified by a 5-tuple (transport protocol, source address, source port number, destination address, and destination port number).
- Per Aggregate: An aggregate is simply two or more flows. Typically the flows in an aggregate has something in common (e.g. any one or more of the 5-tuple parameters or some authentication information).

2.1.1 Integrated Services

The Integrated Services, a QoS architecture developed in the IETF and often associated with RSVP (Resource Reservation Protocol) provides fine-grained QoS approach. It provides QoS to individual applications or flows. Network resources are apportioned as per an application's QoS request, and subject to bandwidth management policy [5]. This reserving mechanism as implemented in RSVP [11], provides hard service guarantees, high granularity of resource allocation and a detailed feedback mechanism. It provides different service classes for intolerant and tolerant, adaptive applications referred to as guaranteed service and controlled load respectively. Guaranteed service comes as close as possible to emulating a dedicated virtual lease line and provides firm (mathematically provable) [45] bounds on end-to-end queuing delay and other parameters such as delay and latency from the various network elements in a path. The aim of controlled load is to emulate a lightly loaded network for those applications that request the service, even though the network as a whole may in fact be heavily loaded. The trick is to use a queuing mechanism such as WFQ (Weighted fair queuing) to isolate the controlled load traffic from the other traffic, and some form of admission control to limit the total amount of controlled load traffic on a link such that the load is kept reasonably low. These two service classes are a subset of all the classes that might be provided.

The key mechanisms in Integrated Services, that provide these services to applications are flowspecs, admission control, resource reservation protocol and packet scheduling. Flowspecs describes the flow's traffic characteristics and the service requested from the network. Admission control is very dependent on the type of requested service and on the queuing discipline employed in the routers. Admission control looks at the flowspecs of the flow and tries to decide if the desired service can be provided to that amount of traffic, given the currently available resources, without causing any previously admitted flow to receive worse service than it had requested. It is closely related to the important issue of policy. The reservation protocol can allot, increase or decrease resource allocation provided to a receiver. It maintains a soft state in routers by periodically sending refresh messages and hence it is easy to send a new reservation that asks for a new level of resources. Packet classifying and scheduling tune the routers to deliver the required performance. Classification associates each packet with the appropriate reservation so that it can be handled correctly. Packet scheduling manages packets in queues so that they receive the service that has been requested.

RSVP provides an explicit reservation and teardown phase. In the setup phase, a "PATH" message, carrying traffic specification, is sent by sender to receiver and is used to establish a "path-state" at intermediate RSVP enabled routers. A reciprocal "RESV" message, carrying flow descriptor (request specification and filter specification), is sent by receiver to sender and is used by routers to reserve resources. RSVP uses a token-bucket model for traffic shaping to characterize its input/output queuing algorithm. Soft reservations requiring periodic refresh are made in each router. Reservations are receiver-based to handle heterogeneous multicast receiver groups. In a multicast scenario, reservations are merged at traffic replication points. This mechanism is highly complex involving elaborate signaling mechanism, and imposed considerable overheads on applications and network elements. In principle this is a significant deviation from the highly successful and scalable best effort IP. Furthermore, non-RSVP routers in traffic path can be weak links degrading QoS provided. Consequently this mechanism is ill suited for some applications, specially those that provide scope for adaptability in the resource requirements, and can operate more efficiently with mechanisms providing as simpler and less fine-tuned QoS support.

2.1.2 Differentiated Services

Differentiated Services (DS) is a set of technologies that are used to provide QoS in a world of best effort service provisions [2]. In DS, all the complexities are pushed out to the edge routers and the core routers are maintained as simple as possible. The DS architecture is based on a simple model where the traffic entering a network is classified and possibly conditioned at the boundaries of the network, and then assigned to different behavior aggregates. In the approach taken by DS, individual micro flows are classified at the edge routers in the network, into one of the many classes. Per –class service is applied in the core of the network and network resources allocated based on management policy as seen in fig 2.1. In the DS domain, Service Level Agreements (SLA) are setup between adjacent networks, SLA establishes policy criteria, and defines the traffic profile to be adhered by independently managed domains. Bandwidth brokers

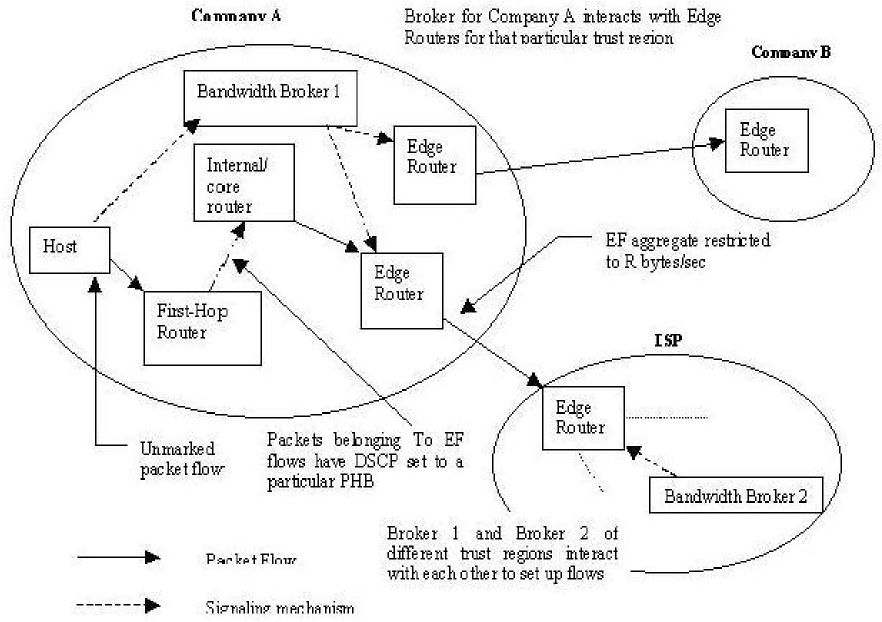

Company A
Broker for Company A interacts with Edge Routers for that particular trust region
Company B
Bandwidth Broker 1
Edge Router
Edge Router
Internal/ core router
Host
Edge Router
First-Hop Router
EF aggregate restricted to R bytes/sec
ISP
Edge Router
Unmarked packet flow
Packets belonging To EF flows have DSCP set to a particular PHB
Bandwidth Broker 2
Broker 1 and Broker 2 of different trust regions interact with each other to set up flows
Packet Flow
Signaling mechanism

Figure 2.1: Generic model of bandwidth broker in a diffserv domain

(BB) are identified in each diffserv domain to manage and negotiate network resources based on SLAs.

As shown in fig 2.2, the classification is done at the ingress router, based on one or more bits in the packet. Then the packet is marked, using code points, as belonging to one of the many classes and injected into the network. The core routers that forward the packet examine this marking and use it to decide how the packet should be treated. Most of the work in this scheme is done at the edge routers. These routers are responsible for classifying, using a multifield classifier and a traffic meter, policer and decide the next action to be taken on the packet. The traffic policer is used to ensure that the packet conforms to the traffic profile previously agreed upon by the network provider and the customer. The packets are then marked with Diffserv Codepoint (DSCP). DS uses six bits of the IPV4 or IPV6 header to convey the DSCP, which selects a per hop behavior (PHB) i.e. the treatment given to the flow at the router on the way to destination.

All packets with the same code point are grouped together and are known as a behavior aggregate (BA). There are two defined PHBs: expedited forwarding (EF), and assured forwarding (AF) [15]. EF PHB supports low loss, low delay, and low jitter giving a sense of virtual leased line as compared to the guaranteed service in RSVP. AF PHB defines four relative classes of service

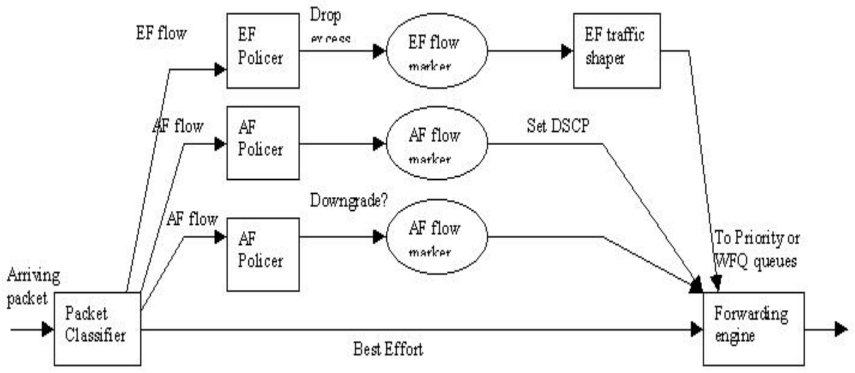


Figure 2.2: Input functionality at edge router

with each service supporting three levels of drop precedence (so a total of twelve code points). Excess AF traffic is not delivered with as high a probability as the traffic “within profile” (conforming to SLA) which means that it may be demoted or downgraded (which defines slightly reduced bandwidth requirements) but not necessarily dropped. Thus Diffserv provides simple and coarse mechanism for QoS support. It exhibits greater flexibility and is able to allocate resources efficiently while still providing service guarantees. Consequently this approach is well suited for providing network level adaptive QoS support to distributed applications operating in heterogeneous networks.

2.1.3 Multi Protocol Label Switching

Mutli Protocol Label Switching (MPLS) [12] is a traffic engineering protocol that provides resource management for flow aggregates via network routing control according to fixed length 'labels' in packet headers. Like Differentiated Services it marks traffic at the ingress network boundary, and un-marks it at egress points. The MPLS-enabled router, Label Switching Router (LSR), routes efficiently, using the fixed length label to determine the next hop. Distribution and management of labels among MPLS routers is done using a complex algorithm, Label Distribution Protocol (LDP), to ensure the various labels have a uniform meaning. Packets are classified and routed at the ingress LSRs of an MPLS-capable domain. MPLS headers are then inserted. When a LSR receives a labeled packet, it will use the label as the index to look up the forwarding table. This is faster than the process of parsing the routing table in search of the longest match done in IP routing. The packet is processed as specified by the forwarding table entry. The outgoing label replaces the incoming label and the packet is switched to the next LSR. This label-switching process is similar to ATM's VCI/VPI processing. Inside a MPLS domain, packet forwarding, classification and QoS service are determined by the labels and the Class of Service (COS) fields. This makes core LSRs simple. Before a packet leaves a MPLS domain, its MPLS label is removed. MPLS is a protocol independent mechanism resident in network level switches with no application control. Hence higher layer QoS protocols such as Differentiated services can readily leverage on the management support provided by MPLS.

2.2 Various QoS and Optimization schemes

Several QoS and optimization schemes have been proposed to enhance the Internet infrastructure for better QoS support. Such schemes involve monitoring tools for getting the network and system resource state in real time, integrating media encoding and transport for better adaptability support, considering QoS state information for routing decisions etc.

In [19] K. Nahrstedt et al. propose a QoS broker that orchestrates resources at the end-points, coordinating resource management across layer boundaries. As an intermediary, it hides implementation details from applications and per-layer resource managers. The broker to properly configure the system to application needs uses services such as translation, admission and negotiation.

A multi-resource reservation algorithm [14] utilizes the resource broker model for an integrated approach to reserving and scheduling the resources with low resource contention. It adopts a component based approach with Resource brokers, QoSProxies and service components as the main entities. The end-to-end QoS provided to the client is determined by the service quality achieved by each individual service component. Input and output qualities of each service component are represented as vectors of multiple QoS parameters. A dependency graph is generated with service components as its nodes and their inter-dependencies as its edges. The algorithm computes a resource reservation plan to reserve a minimum amount of bottleneck resources, i.e. resources with maximum conflicting requests, while deciding appropriate levels of input and output quality for each service component. Simulations with uniform and varying average request arrival rates indicated that the proposed algorithm works better, in terms of reservation success rate than a random reservation path selection algorithm.

In [27], Zhang et al. present an end-to-end transport architecture for multimedia streaming over the Internet. They propose a new multimedia streaming TCP-friendly protocol (MSTFP), which combines forward estimation of network conditions with information feedback control to optimally track the network conditions. They dynamically allocate resources according to network status and media characteristics to improve end-to-end QoS.

[17] answers some questions regarding the conditions on the network load that allow a best-effort network like Internet to support connections of given duration that require a certain QoS. The key idea in their approach is to consider the amount of bandwidth that a new connection will receive over its duration which depends on the transient behavior of the network.

The basic problem solved in this approach is to provide ways to calculate, for a given time window and a state of congestion (number of active connections), the percentage of the time during which the bandwidth the best-effort network allocates to a new active connection will be above a minimul level. This methodology develops a tractable way to compute the admission control policy that takes into consideration such broader QoS definitions that include the time duration over which the quality must be guaranteed, the delay in call setup (e.g. VoIP), and the fraction of time the bandwidth requirements must hold.

[21] explores the possibility of providing a simple, robust and pricing-free QoS solution by practicing “differentiated fairness” : different classes have equivalent performance according to their specific needs. No admission control is required, no absolute guarantee is provided. This “soft” model builds on a scheduler that sets up the router queues and can balance resource sharing so that “differentiated fairness” is obtained.

[35] addresses the issue of adapting the compression of video / audio applications without requiring the video-servers to re-encode the data, and fitting the resulting stream into the rapidly varying available bandwidth. They present a mechanism for using layered video in the context of unicast congestion control. This quality adaptation mechanism adds and drops layers of the video stream to perform long-term coarse-grain adaptation, while using a TCP-friendly congestion control mechanism to react to congestion on very short timescales. The mismatches between the two timescales are absorbed using buffering at the receiver. This scheme allows the server to trade short-term improvement for long-term smoothing of quality.

[8] presents the concept of service brokers which applications and service providers use to identify the network resources needed to meet QoS and cost objectives. Service brokers can incorporate a detailed understanding of an application domain, allowing them to make intelligent tradeoffs and to interact with applications and service providers at a high level. This is possible by building applications around value-added services that encapsulate a variety of simpler resources. This enables both the specification of QoS in terms meaningful to applications, and

global optimization of resource allocation across multiple streams and data types. Service brokers also provide the ability to deal with heterogeneous networks and hierarchical resource management.

[23] addresses the issues in designing metrics that are important in evaluating the QoS of video transmission. Based on real video workload environments and user behavioral patterns, this paper defines parameters of resource-cost (storage and network bandwidth etc.) and user satisfaction (jitter, synchronization skew) and derive analytical interrelatinships among the metric parameters. It also draws on economic relationship between the user-satisfaction and resource consumption factors to solve metric optimization relations.

[24] builds on a pricing framework to provide QoS in diffserv networks. In this model, users are given the freedom to choose the priorities of their traffic, but are charged accordingly. A game theoretic framework is considered to study the case where users choose an allocation of priorities to packets in order to optimize their net benefit. For the case where users with bursty traffic access a single link, it shows that there always exists equilibrium for the corresponding non-cooperative game.

2.3 Qbone Bandwidth Broker architecture

Bandwidth Broker (BB) is an agent that provides a centralized mechanism to control the resources within a DS domain. All agreements between the customer and the service provider that pertain to the type of service required are known as service level agreements (SLA). The BB manages a domain's resources using service policies defined based on the clients' requirements. These SLAs are used to define the relation between policies and the PHBs, while a service provisioning policy (SPP) indicates how traffic conditioners are configured at the edge of the domain and how the traffic streams are mapped to the DS behavior aggregates. The BB requires both the SLAs and the SPPs to achieve a range of services, which are provided to the user. Based on the SLAs the broker decides whether it can provide the allocation, and configures the edge router to mark and classify the packets as decided in the SLA [2, 16].

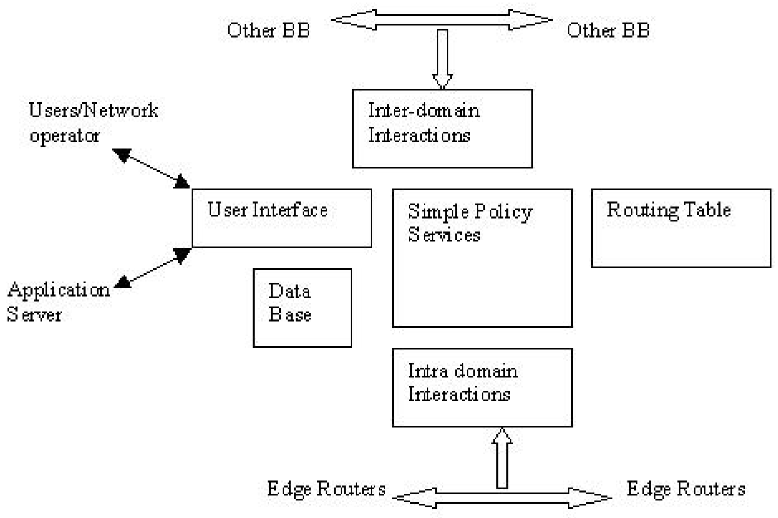


Figure 2.3: Functional Decomposition of Bandwidth Broker

The bandwidth broker consists of a few basic components shown in fig 2.3 [16]

- **User Interface**: The user/application interface provides a means for the user to make resource requests directly, or to the network operator who enters the users' requests. The interface also receives messages from setup protocols (for example RSVP messages)
- **Inter-domain Interactions**: The interactions provide a method of allowing peer BBs to make requests for resources and take admission control decisions to enable flow of traffic.
- **Intra-domain Interactions**: The interactions provide a method for the BB to configure the edge routers within the domain so as to provide quality of service.
- **Routing Table**: A routing table is maintained at the BB to access inter-domain routing information so that the BB can determine the edge routers and the downstream routers before allocating their resources. Further, additional routing paths may be maintained in the routing table for different flows within the domain.

- **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, router configurations, DSCP mapping, and policy information.

The bandwidth broker has been designed to add intelligence to the DiffServ, to help optimize the existing resources. An advisory committee has been initiated to define the protocols implemented by the broker [16]. In the diffserv architecture, flows are allocated resources without any understanding of the nature of information being transmitted. As a result, the broker statically overallocates resources so as to meet guarantees made to the client. This over allocation wastes resources and causes future flow requests to be rejected. Furthermore, the broker does not consider the nature of the flow and may allocate resources for rogue flows that can exceed their allocations and hog resources causing congestion, and affect the QoS of guaranteed flows. Thus, while diffserv does provide a sense of resource allocation and QoS, it does not guarantee QoS or eliminate the possibility of congestion.

Chapter 3

CABB: Architecture, Policies, Operations and Implementation

3.1 Architecture

A CABB may receive a resource allocation request from one of two sources: a request from an element in the domain that the broker controls (or represents), or a request from a peer (adjacent) bandwidth broker. The bandwidth broker either confirms or denies the request and responds accordingly. It might generate additional request messages for downstream resources. As explained in the previous section the clients' traffic rate, time for which the service is required, delay and jitter are some of the parameters for consideration while defining a policy. The broker maintains a database of parameters pertaining to the various flows. The policy table contains various parameters such as service level agreements, service mappings/DSCP mappings, policy information, and management information. The broker also maintains another database to store parameters such as current reservations/allocations, edge router configurations. Using these parameters the broker agent makes a reservation for the client and assigns a DSCP for that service. Since each client gets to define its requirements and these get translated into SLA's, this helps the broker decide on the resource allocation. The normal broker would only take the current resources into account before allocating resources to a flow request. But this does not take into account the adaptability of applications and hence cannot ensure that a rogue flow in a particular service class will not affect flows belonging to it or other service classes. In case a flow cannot be allocated resources as specified in SLA, then that flow is rejected and if a flow does not confirm to profile then it is dropped.

CABB builds on the observation that multimedia applications are adaptive (flexible) in terms of network parameters such as packet loss, delay and jitter. For example, a multimedia application flow may be tolerant or intolerant to packet loss [5]. CABB understands nature of information being transmitted, the flow's requirements and flexibility to network level parameters including packet loss tolerance. It uses the content information to adapt multimedia flows to maintain some level of QoS for these flows rather than refusing them allocations or disrupt their flow when a rogue flow causes congestion. This is done such that when the demand for resources exceeds availability the flow is allowed and maintained by reducing level of quality. The goal of the allocation is to ensure fair resource utilization for all flows, give them guaranteed/assured QoS by delivering them (especially multimedia) in a timely manner in the event of high congestion in one or more links along the path. It prevents non-confirming (or rogue) flows from affecting the performance for conforming flows by constantly monitoring network condition with a gradual degradation of service for rogue flows. Thus it provides the incentive in support of end-to-end congestion control for best effort traffic. CABB decides to allocate flows to a particular service class depending on SLAs, the adaptability of application to network level parameters (flexibility) and current available resources by assigning a particular code point (DSCP) and then updating the policy table for this particular flow. It gives an application another chance to use the network even if resources as specified in SLA are not available. It tries to allot lesser resources in a particular service class to allow the application to go through. This is done such that the end user perceives a quantifiable QoS although full resources were not allocated.

The resources available to the CABB are the various types of queues and available bandwidth for those queues for all the edge routers in its domain. Depending of whether it can satisfy an application's request, the demanded and/or allocated service class is mapped to already existing DSCP for that service class inside the broker's policy table. Diffserv-enabled routers use the DSCP marking to map packets or flows to the particular queue to provide the requisite service (e.g. Priority queue for EF flows) or PHB. The DSCP mappings may be unique to each router but the PHB is the same for all diffserv-enabled routers. We have three different service classes in our implementation (EF, AF and best effort). To allow for all these service classes we implemented a scheduler that allowed us to implement and service various types of queues for the various service classes. To aid the broker in it's DSCP mapping, we implemented a priority queue for EF PHB (also called EF code points) and weighted round robin queues for AF and Best effort PHB (also called AF and Best effort code points respectively). The main idea here is that EF flows should receive guaranteed service, which means EF flows don't face any queuing delay while traveling towards the destination. A priority queue starves off other traffic while it's being served. The diffserv capable edge router also takes care of shaping and policing the EF flows which ensures that the flow downstream confirms to the profile. The weighted round robin queues are serviced according to their weights. We put higher weights for AF flows and low weights for Best Effort flows. The CABB is aware of the routers' queue implementation and available resources because it stores that as a table for every edge router in that domain. This is important because while allocating any flow to any queue in an edge router, this agent needs to know the available resources (bandwidth) at that router. The CABB's policy table has only one entry for a particular source-destination pair. Any change in that entry is then reflected in all the edge-routers of that domain.

For interbroker communication, the CABB looks up its database and routing table to figure out the downstream edge router and peering broker. It looks up this broker from a table of brokers managed by a central entity and communicates with it giving the appropriate flow parameters. The broker down the link, then uses these parameters along with it's idea of the link's latest characteristics and resources to decide whether to allow the flow to continue or not. If this is not successful then it informs the broker uplink and the flow is given another chance to retry before being rejected else the 2nd broker will invoke a 3rd broker down the link, and continue this way till the destination is reached. This is a one-way communication from host to destination and does not imply that the destination has resources reserved at the same time as the host if it has to communicate to host. CABB gives us the advantage of setting up a flow using RSVP type signaling without the overheads of keep and refreshing up-to date network state information.

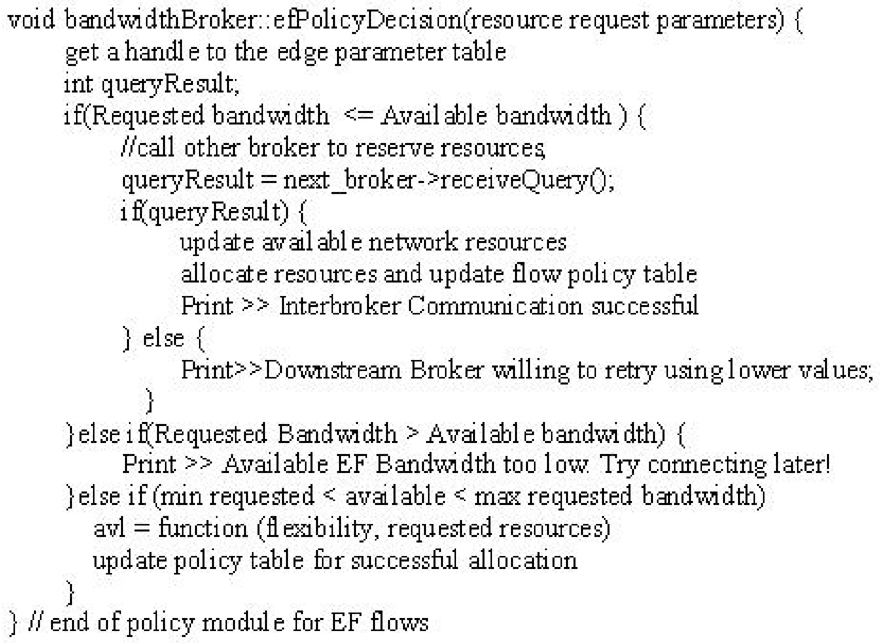
Networked multimedia flows are usually bursty and it's not easy to define the characteristics of such a flow [4]. The CABB uses the minimum information given by such a flow (such as average bit rate and peak bit rate), the ranked importance of an application, the importance of the user, the current available resources, the result of inter-broker communication and the tolerant adaptability of the application to network level parameters to come to a decision as to allow the flow to continue or not. If the flow is allowed to continue then the broker set's up its policy table else it tries to give the application another chance for a retry (with a different set of input parameters) before rejecting the flow's request for either EF or AF service. This way the broker tries to set up a flow taking into account a highly congested network down the link. A highly congested network may stop best effort but can allow the high priority traffic to go through. Hence even though it's congested the users of such a network don't see the congestion because they've paid for a certain service (either premium or assured forwarding or both). Hence, multimedia can still go through the network through some form or other with the broker doing coarse tuning of bandwidth requirements and application level adaptive QoS doing the fine-tuning of the applications' response/sensitivity to network changes.

Some of the preset conditions of allocation for the broker are that there should be a limit on the number of flow reservations allowed per class (the total bandwidth allocated to that service class), and a fixed amount of bandwidth must be reserved for best effort services [2]. This is to make sure that a particular set of traffic doesn't starve the rest. By limiting the number of reservations of each type of service class we ensure a good utilization of the bandwidth.

3.2 Operations and Policies

Flexible networked applications can accept and tolerate resource scarcity to a certain minimum bound, and can improve its performance if given a larger share of resources. Furthermore, they are willing to sacrifice the performance of some quality parameters in order to preserve the quality of critical parameters [1]. In the context of variations in resource availability, it is thus desirable to trade off less critical quality parameters for preserving quality assurance for critical parameters. The CABB translates the flexibility in terms of particular network level parameters to decide on QoS range (QoS_{min}, QoS_{max}) to be allocated.

When an application requests the CABB for network resources, it passes application specific QoS parameters. These parameters consist of media quality descriptions for the specific media characteristics (e.g., height, width, and color specification in a video stream), the media sample rate, priority/criticality and transmission characteristics requirements for end-to-end delivery (e.g., end-to-end delay bounds). On receiving a flow request along with application QoS parameters, CABB translates them to network resource requirements such as usage profiles for rate (chief and peak information rate), burst, delay, jitter, flexibility to delay, jitter, loss and a time for which the profile is to be active. CABB tracks current allocation of marked traffic interpreting new requests in light of the policies and available resources. It then looks up a policy table for the existing policy that governs the host and contains parameters such as service level agreements (SLA), service mappings/DSCP mappings, policy information, management information, current reservations/allocations, and edge router configurations. These parameters along with the application's flexibility [1, 6] are passed to different policy engines as defined in SLA. These simple policies further translate the given parameters to specific network actions including bandwidth management (allocated transmission rate), queuing (per hop behavior), buffer space for queuing, network monitoring and accounting. They can sense the users' bandwidth requirements (after translating flexibility to map to minimum resource requirement [1]) and allocate resources accordingly in a succinct and organized fashion treating all flows fairly thereby making them a very effective tool. The resource allocation is done irrespective of client's demand for higher and more than required network resources thereby conserving and allocating them for flow requests later. This is possible because CABB understands applications requirements thoroughly, effectively translating them between various service levels [8]. By implementing intelligent policy engines depending on the type of service desired it performs better admission control. A flow is rejected if there are insufficient resources inspite of reduced requirements or history of its rogue nature (e.g. continuously bursty after promising constant rate). A log for all flows is one convenient way to check the flow’s history. For successful admission, CABB’s resource allocator makes a resource reservation for the client (usable bandwidth, buffers) and assigns a DSCP for that service. It then schedules the flow to a particular queue manager (Priority



```
void bandwidthBroker::efPolicyDecision(resource request parameters) {
      get a handle to the edge parameter table
      int queryResult;
      if(Requested bandwidth  <= Available bandwidth ) {
            //call other broker to reserve resources,
            queryResult = next_broker->receiveQuery();
            if(queryResult) {
                  update available network resources
                  allocate resources and update flow policy table
                  Print >> Interbroker Communication successful
            } else {
                  Print>>Downstream Broker willing to retry using lower values;
             }
      }else if(Requested Bandwidth > Available bandwidth) {
            Print >> Available EF Bandwidth too low. Try connecting later!
      }else if (min requested < available < max requested bandwidth)
         avl = function (flexibility, requested resources)
         update policy table for successful allocation
      }
} // end of policy module for EF flows
```

Figure 3.1: Sample code for EF policy engine

or Weighted Fair Queue). Diffserv’s internal policing mechanism then forces the flows to adhere to agreed policy. Furthermore, with the help of broker manager, CABB may generate additional request messages for downstream resources [16, 17, 19]. Each downstream CABB takes into account inter-domain traffic SLA before allocating resources. Fig 3.1 shows sample code and explains the algorithm behind the EF policy decision.

3.3 Illustrative Example

The CABB functioning can be explained with the help of a test network as shown in figure 3.2. The test network includes two DS domains, two sources in domain 1 and a destination in domain 2. The two DS domains are required to show the inter-domain interaction between the broker agents to provide end-to-end resource allocation for a source-destination pair. We assume static routes in this example.

When a source 1 in diffserv domain 1 (DS1) requests service, it contacts the CABB agent 1(BB1) in DS1 enroute to the destination 2 giving requirements such as it's average and

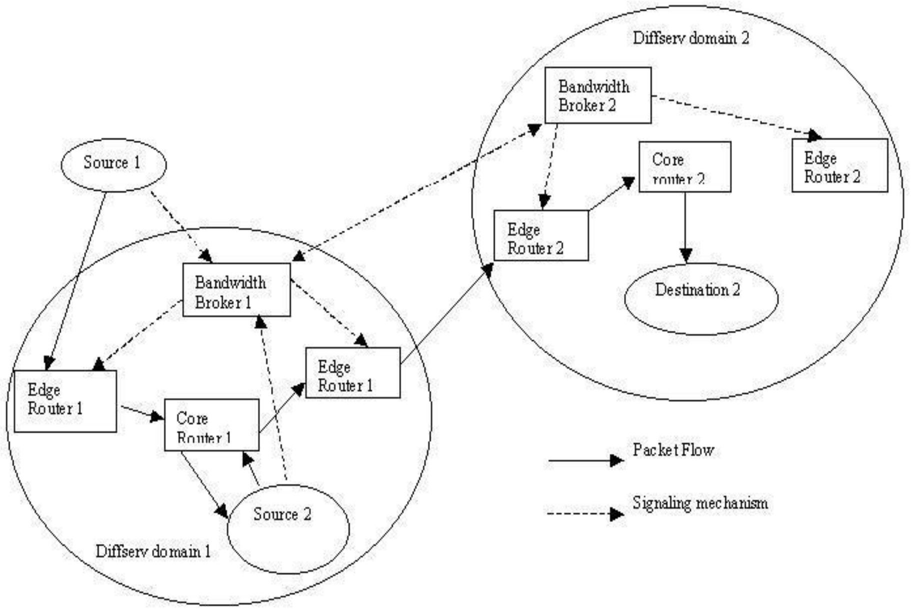


Figure 3.2: Test Network with different scenarios

peak input or transmission rates, delay and jitter along with its type; whether it's a multimedia (audio, video, video conferencing etc) flow or not. The BB1 looks into its database, to decide upon the best available bandwidth, jitter and delay parameters taking into account the agreed upon SLA, the current available resources, and a particular flexibility number for that application. This flexibility number it gives an indication on how much the application can tolerate loss, delay or jitter and at the same time give a certain quantifiable or perceived QoS to the end user. It is fixed for various multimedia applications and is arrived at after considering the tolerance of the application to the above network level parameters. BB1 passes the flow's parameters (same or reduced) to downstream broker (BB2) in diffserv domain 2 (DS2) to request for resources till the destination. After it receives a positive acknowledgement from the downstream CABB (BB2), BB1 then assigns a DSCP for the traffic flow between this source-destination pair. If present resources are insufficient or BB2 returns a negative acknowledgement, then BB1 retries to set up of the flow with a different set of input parameters or assign a set of lower DSCPs, which define slightly lesser bandwidth requirements and informs the application of this intent. Although these parameter values are lesser than the previous values, the agent has better idea of the applications tolerance to loss and other characteristics such as delay and jitter requirements on account of the flexibility number, irrespective of what the user requests. Based on this the broker guarantees/assures that the information/content sent to destination 2 has enough useful information which will be understood by user on destination 2 giving the impression of a certain level of QoS and at the same time will prevent unnecessary traffic from hogging the network which would only cause congestion and later packet drops at edge as well as core routers.

For e.g. source1, an adaptive multimedia audio application with less flexibility due to the tight bounds on its loss, delay and jitter requirements (value of 2 in a range from 1 – 20) has asked for premium service (EF PHB) till destination2. The diffserv router to police EF flows uses two parameters: chief information rate (CIR) and peak information rate (PIR). The application will generate data at an average rate of CIR and can go to peak rates of PIR. Hence effectively it is reserving PIR as its peak bandwidth although it may not use PIR all the time during its duration. If the flow exceeds PIR at any time, it goes out-of-profile and will be downgraded to a lower DSCP and eventually dropped. The CABB invokes its EF service policy manager to decide the result of the given request. It first checks for available resources in the EF queue for that particular edge router enroute to destination2 i.e. edge router1. If the resources aren't sufficient to meet the request as agreed in the SLA i.e. the available bandwidth is less than CIR or between CIR and PIR, BB1 informs the application that it is willing to retry setting up the flow with reduced resource requirement. It now uses the flexibility number as a tool to decide the resources to allocate (bandwidth between CIR and PIR) for this application for it to be meaningful to the end-user under the current resource crunch. After ensuring sufficient resources, i.e. the available bandwidth is greater than PIR; BB1 contacts the downstream BB2 with the current set of parameters and so on till it reaches the destination. If at all points in various domains the flow receives a guarantee by the respective CABBs according to the inter-domain SLAs, BB1 sets up it's policy table with the source-destination pair, fills in the required parameters and updates the EF service policy manager on available resources. If for some reason this second retry is not successful the application has to keep on retrying till it gets the required resources.

Source2 an adaptive multimedia video application has asked for AF PHB till destination2. This application is more flexible compared to multimedia audio and its flexibility number is greater than that for audio (value of 10 in a range from 1 – 20) [5]. It can do with AF PHB till destination2 on account of fewer restrictions on loss and a certain delay bound. The diffserv router to police AF flows uses the following two parameters: chief information rate (CIR) and committed bucket size (CBS). Hence effectively it is reserving CIR as its peak bandwidth. If the flow exceeds CIR at any time, it will be downgraded to a lower DSCP and eventually dropped. The CABB decides to invoke AF service policy manager to decide the result of this request. It first checks for available resources in the AF queue for that particular edge router enroute to destination2 i.e. edge router1. If the resources aren't sufficient to meet the request as agreed in the SLA i.e. the available bandwidth is less than CIR, BB1 informs the application that it is willing to retry setting up the flow with reduced resource requirements given by the flexibility number. Using this flexibility number it calculates a lesser CIR than that requested by the application such that it allows the flow to use the network and at the same time be meaningful the end user. If there are sufficient resources then BB1 contacts the downstream BB2 with the current set of parameters and so on till it reaches the destination. If at all points in various domains the flow is given an assurance by the respective CABBs according to the inter-domain SLAs, BB1 then sets up it's policy table with the source-destination pair, fills in the required parameters and updates AF policy manager on available resources. If for some reason this second retry is not successful, the application has to keep on retrying till it gets the required resources. The CABB will write to a log about the state of flow allocation that can be verified later.

Hence instead of waiting for congestion to happen and then take preventive action, we have effectively ordered the flows in such a manner that they confirm to the traffic profile as agreed in the original SLA or the reduced profile arrived at using the flexibility number. Enough flows at a time are allowed such that the network is not unduly loaded. Since controlling multimedia traffic is difficult because there are no inherent feedback control mechanisms in such applications, which use UDP for data transmission, we have managed to control such traffic at the edge router. As such all complexity is kept at edge router level and core is kept simple. The broker's intelligent decisions also ensure that the bandwidth pipe is full and all applications get a fair share of bandwidth according to their SLA. At times of congestion, if the broker allows a multimedia application to go through, it may do so at the expense of reduced parameters but since it is content aware and knows the flexibility of network level loss tolerance of such applications and their adaptability to such losses, it takes this into account while allocating for a certain reduced flow rate. It is up to the application later whether to go through with such a connection or not. The application can later retry to get richer set of services, which satisfies its parameters. In case of premium service with EF flows, the broker guarantees that this flow will not face any queuing delay along the way and will be delivered within the constraints of the given parameters giving the impression of a virtual leased line. Such flows will rarely experience any packet drops (early or late) at the routers. From the results, we see that, queue management and diffserv policing work such that for any non-conforming flow only that particular queue will be penalized to which the flow belongs. The other flow aggregates are not affected. If for any reason the broker is not able to allow a flow after retrials, it will log this report indicating the flow's requirements, source, destination and time the flow was requested.

3.4 Implementation

We have implemented the bandwidth broker on the Network Simulator-2 (NS-2) toolkit. The NS-2 toolkit has substantial functionality for simulating different network topologies and traffic models. NS-2 also has an open architecture that allows users to add new functionalities which proves very useful to us. Using the diffserv patch provided by Nortel Networks [32] and extending it by our scheduler which helps set up and serve various FIFO queues such as non-preemptive priority queues, weighted round-robin queues and best effort queues for the various edge routers and our CABB we can generate diffserv domains and create suitable test networks as shown in the experimental evaluation.

The diffserv implementation has three modules to it. Two of them are with regards to the edge router and core routers, and the third module is the policy and resource manager. The policy class handles the creation, manipulation and enforcement of edge router policies. A policy defines the treatment the packets will receive at an edge router. Policies are set using Tcl scripts [32]. The policy class uses a policy table to store the parameter values. The table is in the form of

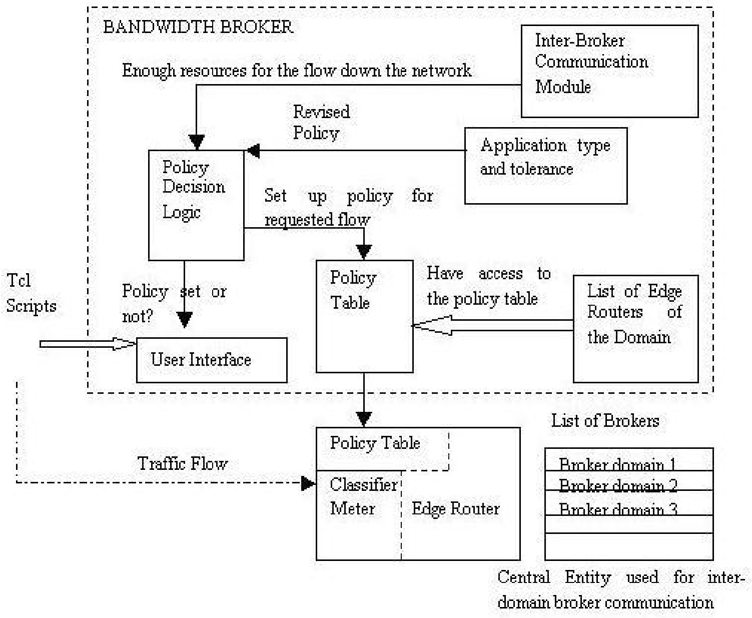

BANDWIDTH BROKER
Inter-Broker Communication Module
Enough resources for the flow down the network
Revised Policy
Application type and tolerance
Policy Decision Logic
Set up policy for requested flow
Policy Table
Have access to the policy table
List of Edge Routers of the Domain
Tcl Scripts
Policy set or not?
User Interface
List of Brokers
Broker domain 1
Broker domain 2
Broker domain 3
Traffic Flow
Policy Table
Classifier
Meter
Edge Router
Central Entity used for inter-domain broker communication

Figure 3.3: CABB internal block diagram and interactions

an array that has various fields such as SLA, source-destination, current reservation, router configuration policies, and DSCP mappings. The packet that arrives at the edge router is classified to decide as to which traffic aggregate it belongs to, and a specified meter is used to check the average traffic rate of that client to make sure it corresponds to the current sending rate, else it gets downgraded to a lower DSCP.

As shown in figure 3.3, the bandwidth broker is used to configure the policy module of the diffserv. Our broker implementation consists of four modules viz. user interface module through which the user/network operator can allocate resources, a database module that stores all the parameters required to make the reservation decisions, a service policy manager module and a central entity called the broker manager. The broker manager stores handles to all the brokers in the topology inside a table that is useful for inter-broker communication. The policy manager module helps the broker to create a particular policy module for EF, AF or best effort flows and passes a handle to this policy module to all the other edge routers in its domain. Any changes that are made in this module are reflected in the diffserv module for all edge routers of that domain. A timing diagram in figure 3.4 shows the steps involved in setting up a connection when a flow requests transmission.

The broker makes the provisioning based on the SLA's as agreed upon with the client/user (through the user interface module) using Tcl scripts and in correspondence with other parameters in the database module such as the current reservations and the router configurations which are also set using Tcl scripts. The configuration changes are made to the policy module, and these changes are reflected in the policy module of the diffserv edge routers of that domain. Within the policy module we associate every source-destination flow with a policy type, meter type, current rate of traffic (the rate agreed upon with the client) and other policer specific parameters. We associate a set of DSCPs with this flow. Each DSCP corresponds to a different traffic rate and PHB and is internally implemented as the queue that will serve the packet. Using our scheduler, we can implement a mixture of different types of queues for any diffserv-enabled router.

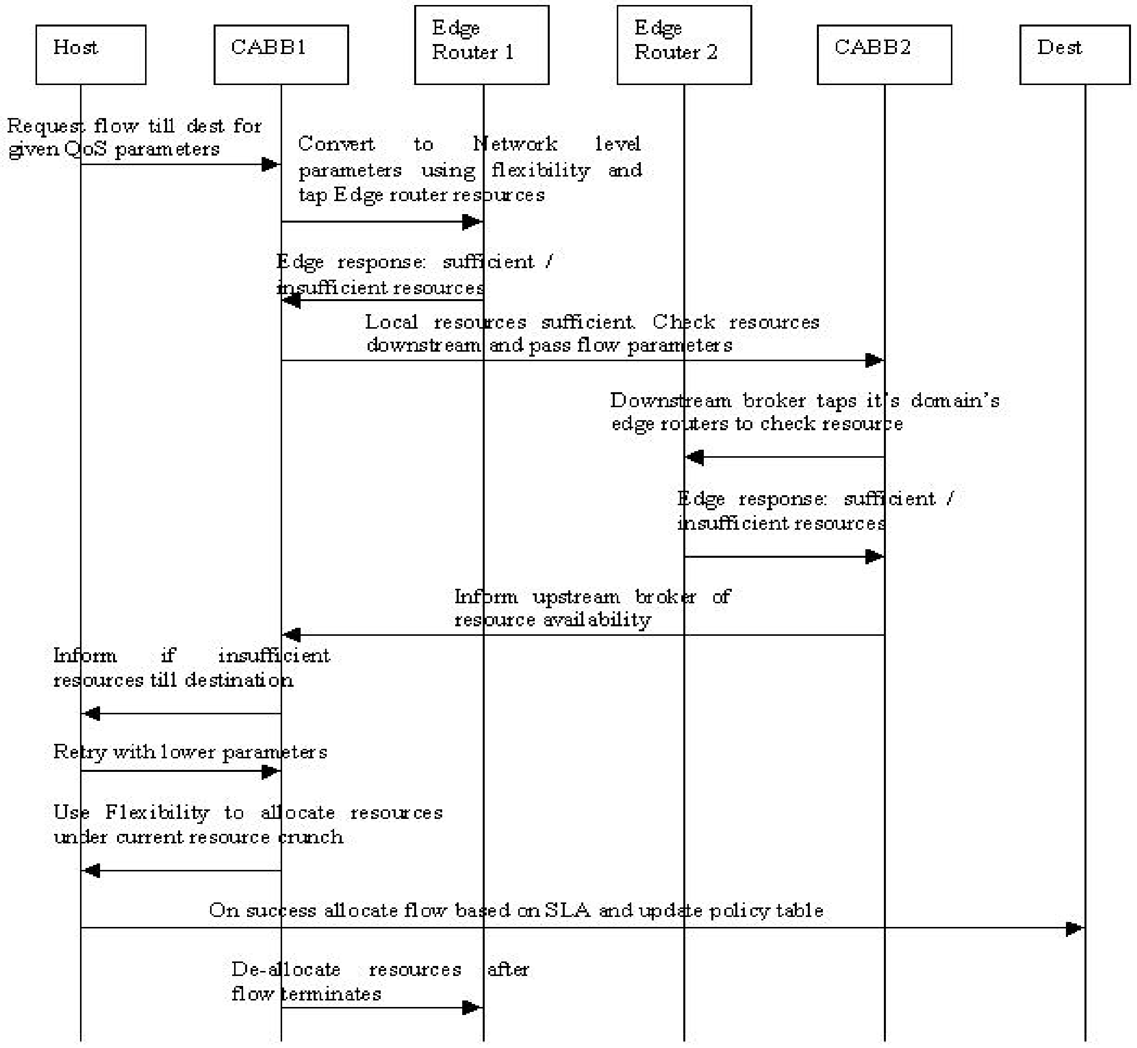

Host
CABB1
Edge Router 1
Edge Router 2
CABB2
Dest
Request flow till dest for given QoS parameters
Convert to Network level parameters using flexibility and tap Edge router resources
Edge response: sufficient / insufficient resources
Local resources sufficient. Check resources downstream and pass flow parameters
Downstream broker taps it's domain's edge routers to check resource
Edge response: sufficient / insufficient resources
Inform upstream broker of resource availability
Inform if insufficient resources till destination
Retry with lower parameters
Use Flexibility to allocate resources under current resource crunch
On success allocate flow based on SLA and update policy table
De-allocate resources after flow terminates

Figure 3.4: CABB timing sequence after host initiates transmission request till destination dest

Chapter 4

Experimental Evaluation

In order to study the performance of CABB, simulations were conducted using various topologies. These simulations are designed to stress the system and bring out a comprehensive evaluation of the broker by checking the packet throughput for the defined DSCPs at various points in time, by using congested links and out-of-profile clients, by introducing the worst-case situation of all the clients using their allotted bandwidth, by using particular queues for particular multimedia applications, by testing the retrial strategy of broker and instances of failure when unable to allocate enough resources. We concentrate on multimedia flows and explain the results based on experimenting on congested links, under-allocation of resources (in case of retry) and out of profile clients. The topologies show different applications (for e.g. audio, video, video conferencing etc) running on the source nodes. There is only one destination node, which acts as a sink. These experiments show how multimedia flows (both EF and AF) are analyzed, given resources and regulated before being allowed to use the network.

In the experiments that follow, we define the following parameters: Peak Information Rate (PIR Mbps- to monitor EF), Committed Information Rate (CIR Mbps- to monitor EF, AF), Committed Burst Size (CBS MBps- to monitor AF) and packet transmission rate (RATE Mbps), Available Bandwidth (ABW), and Usable Bandwidth (UBW- actually allocated bandwidth). Early drops (edrops) at routers follow RED algorithm [5], while late drops (ldrops) occur when packet arrival exceeds buffer size. We allocate 1/3rd bandwidth to EF flows, 5/12th bandwidth to AF flows and the rest (1/4th) to best effort in all the topologies given below [5]. Two simple policy engines are shown in table 4.1

EF policy Manager	AF policy Manager
If ABW > PIR, UBW = f (PIR) if CIR < ABW < PIR, UBW = f (flexibility, CIR, PIR), if ABW < CIR || UBW retry with reduced parameters	If ABW > CIR, UBW = f (CIR, CBS) If ABW < CIR, UBW = f (flexibility, CIR, CBS). If ABW < UBW, retry with reduced parameters

Table 4.1: EF and AF policy decision engines

The CABB uses the above policy rules. These simple but effective rules give all flows equal chance of utilizing the network and at the same time provide CABB a very powerful control for flow allocation. It maintains a log regarding the flow's content, timings and parameters. We define code point 10 for EF PHB and Code points 21, 23 for AF PHB. For non-conforming flows code point 10 (EF code point) is downgraded (which defines slightly lesser bandwidth requirements) to code point 11 and code point 21 and 23 (AF code point) are downgraded to code point 22 and 24 respectively.

The CABB considers requests in the order asked (first come first serve), informs the respective node/application of the result, allocates resources to these flows based on their content, availability of resources, SLA and interbroker communication or else requests for different parameters before rejecting the applications' request for a particular service class. After setting up the parameters, it assigns a DSCP to that flow which is used by the diffserv-enabled router for classification and monitoring. For the experiments below, graph 1 shows the entire statistics including TotPkts- Total Packets, TxPkts – Transmitted Packets, edrops – Early drops and ldrops – Late drops at the core router (Core). Graphs 2 and 3 show the EF profile (CP 10 and 11). Graph 4 and 5 show the AF profile (CP21 and 22). The table shows the simulation configuration parameters.

4.1 Topology One: Single EF and Two AF flows

The topology in Fig 4.1 has a mixture of one EF and two AF flows. There are three sources (S0, S1, S2), destination host (dest), two edge routers (E1, E2) and core router (Core). S0 (audio – EF PHB), S1 (video –AF PHB), and S2 (video conferencing- AF PHB) perform unidirectional data transmission across the bottleneck link (Core-E2) to dest. We analyze (un)responsive AF and EF flows in light of CABB resource allocation. The objective here is to see how the network elements react when these unresponsive flows pump packets at a higher than agreed upon rates in light of CABB's allocation to these flows. We will also analyze the effect of

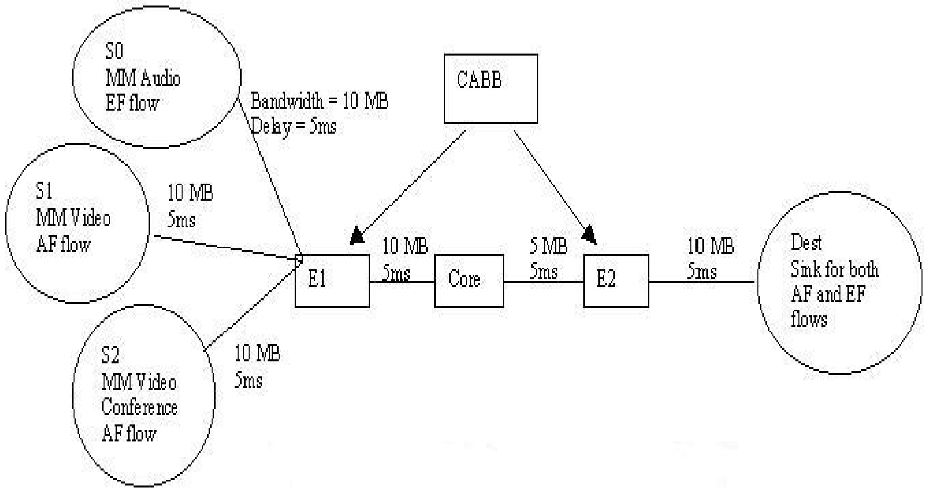


Figure 4.1: Topology (one EF and two AF flows)

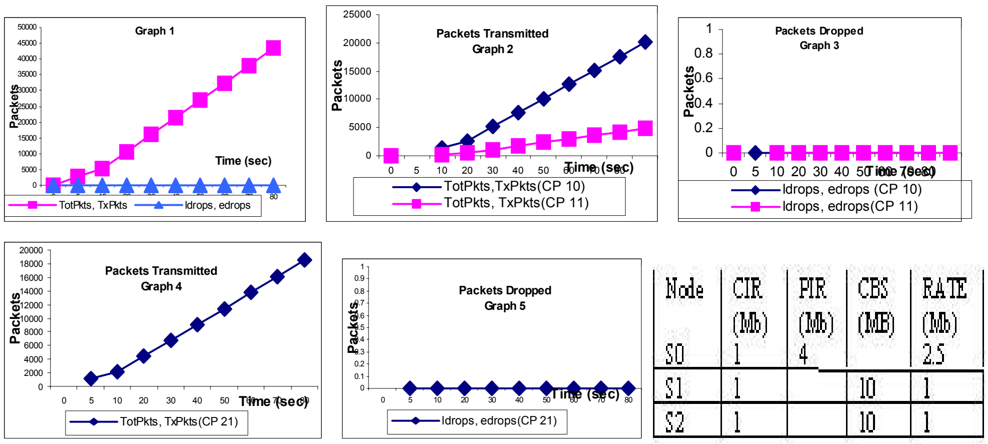
one flow aggregate on other flow aggregates as well as allocation and control of various flows.

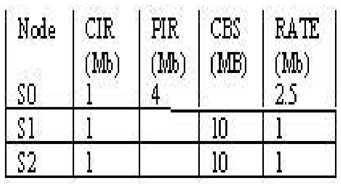
<u>Experiment 1</u> (Out-of-profile EF and in-profile AF): The objective is to show how each flow is regulated according to it's own service level agreement and the effects of a rogue flow in a particular class do not affect the flows belonging to other classes of aggregate data. S0 transmits data at a greater rate going out-of-profile. The other sources maintain in-profile state throughout.

Rate $_{S0}$ = 4 Mbps; UBW $_{S0}$ = function (2,1,4) = 2 Mbps.- accepted.

Rate $_{S1}$ = 1 Mbps; UBW $_{S1}$ = function (1, 10) = 1 Mbps.-accepted.

Rate $_{S2}$ = 1 Mbps; UBW $_{S2}$ = function (1, 10) = 1 Mbps.-accepted





Node	CIR (Mb)	PIR (Mb)	CBS (MB)	RATE (Mb)
S0	1	4		2.5
S1	1		10	1
S2	1		10	1

Figure 4.2: Out-of-profile S0 gets downgraded not affecting in-profile S1 and S2

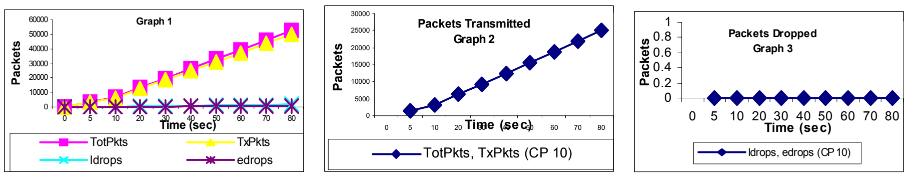
Router Response: In this experiment, S0, S1 and S2 demand 4mb (PIR), 1mb (CIR), and 1mb (CIR) and transmit at 2.5mb, 1mb, and 1mb respectively. The results from this experiment are plotted in Figure 3. As can be seen from these plots (graphs 2 and 3), S0 is allocated reduced bandwidth (2 MB) after retrial with reduced resources and its packets are downgraded to CP 11 since $\text{Rate}_{S0} > \text{UBW}_{S0}$. S1 and S2 are allocated their requested resources and are never downgraded as there are in-profile (see graphs 3 and 4). This shows that CABB does not allow the out-of-profile EF flow to downgrade or drop in-profile AF flow from S1 and S2.

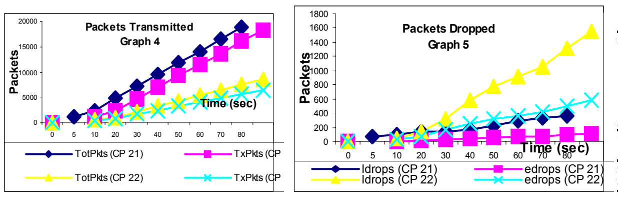
Experiment 2 (An out-of profile AF source): This experiment stresses out the AF class and we observe its effect on the EF and AF aggregate. S0 and S2 maintain in-profile flow but S1 goes out-of-profile and the AF aggregate packets should be downgraded and eventually dropped without affecting EF flow.

$\text{Rate}_{S0} = 3$ Mbps; $\text{UBW}_{S0} = \text{function}(3) = 3$ Mbps.- accepted.

$\text{Rate}_{S1} = 1$ Mbps; $\text{UBW}_{S1} = \text{function}(1, 10) = 1$ Mbps.-accepted.

$\text{Rate}_{S2} = 1$ Mbps; $\text{UBW}_{S2} = \text{function}(1, 10) = 1$ Mbps.-accepted





Node	CIR (Mb)	PIR (Mb)	CBS (MB)	RATE (Mb)
S0	1	3		2.5
S1	1		10	2
S2	1		10	1

Figure 4.3: S1 goes out-of-profile causing AF aggregate to downgrade

Router Response: In this experiment, S0, S1, S2 demand 3mb, 1mb, and 1mb and transmit at 2.5mb, 2mb, and 1mb respectively. The results from this experiment are plotted in Figure 4. As can be seen from these plots, S0, S1 and S2 are allocated full resources. Graphs 2 and 3 show that S0 is in-profile and is never downgraded. S1 goes out of profile and the AF aggregate packets are downgraded to CP22 and are eventually dropped (see in graphs 4 and 5). As seen in graph 5, edrops are less than corresponding ldrops. Finally, we observe that the out of profile AF flow does not affect the in-profile EF aggregate. We see here how rogue flows in one service class do not affect flows in other service classes.

4.2 Topology Two: Multi EF flows

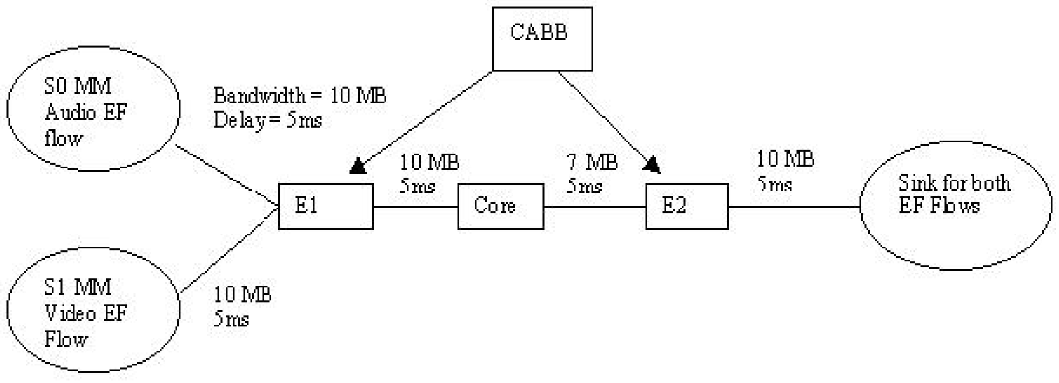


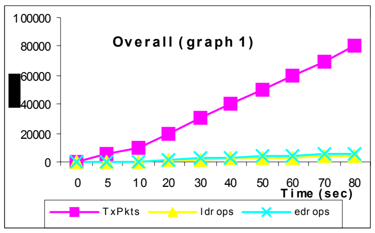
Figure 4.4: Topology 2 (Multi EF flows)

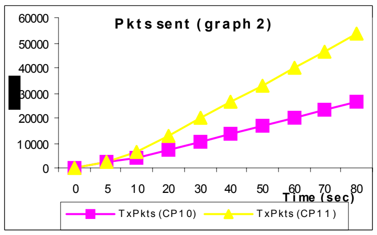
The topology in Fig 4.4 has two EF flows. There are two sources (S0, S1), destination host (dest), two edge routers (E1, E2) and core router (Core). S0 (audio – EF PHB), S1 (video – EF PHB) perform unidirectional data transmission across the bottleneck link (Core-E2) to dest. CABB allocates resources between E1, E2. We analyze CABB decisions regarding amount of flow allocations and flow control for unresponsive EF flows.

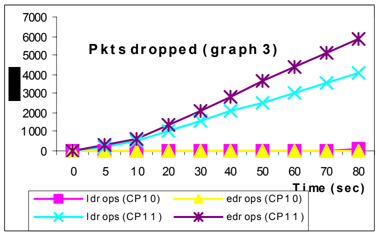
<u>Experiment 1</u> (Out-of-profile EF): The objective is to show how each flow is regulated according to it's own service level agreement and the effects of a rogue flow in a particular class do not affect the flows belonging to other classes of aggregate data. S0 transmits data at a greater rate going out-of-profile. The other sources maintain in-profile state throughout.

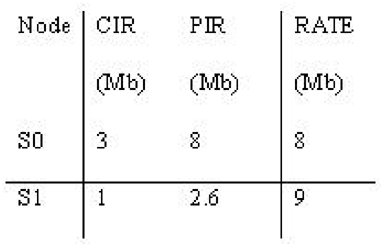
Rate $_{S0}$ = 8 Mbps; UBW $_{S0}$ = function (1,3,8) - rejected.

Rate $_{S1}$ = 9 Mbps; UBW $_{S1}$ = function (5, 1, 2.6) = 2.6 Mbps.-accepted.









Node	CIR (Mb)	PIR (Mb)	RATE (Mb)
S0	3	8	8
S1	1	2.6	9

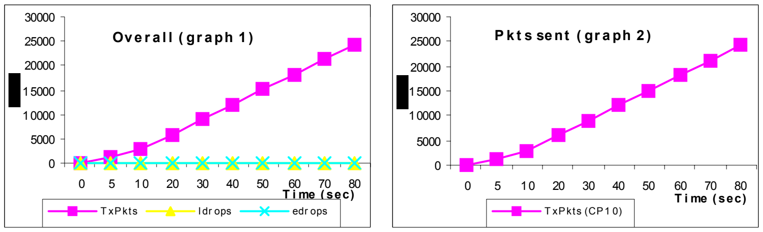
Figure 4.5: S0 is rejected and S1 goes out-of-profile

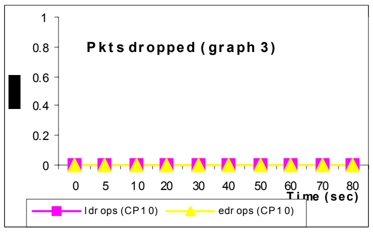
<u>Router Response:</u> S0 was rejected after interbroker communication failed (congested link) for less flexible audio application. S1 was allocated full resources (PIR) but Rate$_{S1}$ >> UBW$_{S1.}$ Thus one flow is rejected despite retrial and another allocated full resources but policed and forced to adhere to SLA.

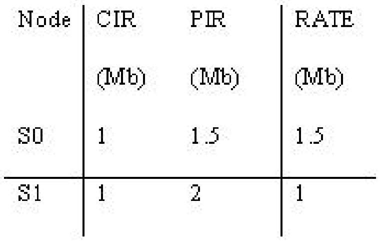
<u>Experiment 2</u> (Use of flexibility for allocation of resource): The objective is to show how flow allocations are increased by allocating reduced resources for particular flows arrived at after considering flexibility number and increasing flow throughput. Also, we see high bandwidth flows are rejected if there isn't enough resource.

Rate $_{S0}$ = 1.5 Mbps; UBW $_{S0}$ = function (1.5) =1.5 Mbps-accepted.

Rate $_{S1}$ = 9 Mbps; UBW $_{S1}$ = function (5, 1, 2.6) = 2.6 Mbps.-accepted.







Node	CIR (Mb)	PIR (Mb)	RATE (Mb)
S0	1	1.5	1.5
S1	1	2	1

Figure 4.6: S1 allocated lesser resources after considering flexibility of video

Router Response: S0 was allocated full resources. This reduced the total resources availability. S1 was however allocated reduced resources rather than being denied service on account of higher video flexibility than audio. We observed that CABB increased flow allocations with some flows getting reduced resources arrived at from their flexibility number

4.3 Topology Three: Allocation and inter-effects of UDP and TCP flows

The topology in Fig 4.7 has a mixture of one EF and two AF flows. There are three

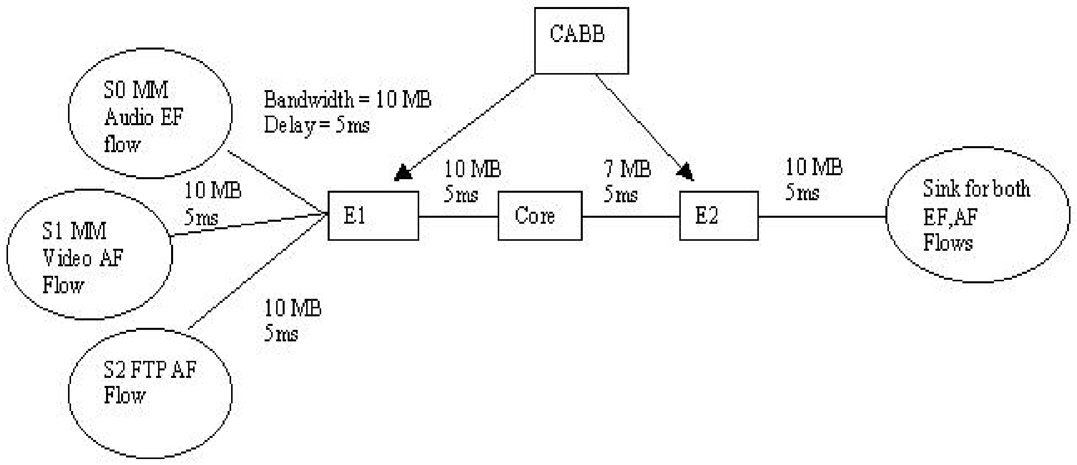


Figure 4.7: Topology 3 -- Allocation and inter-effects of UDP and TCP flows

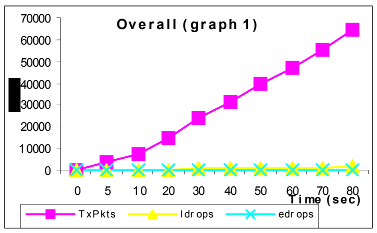
sources (S0, S1, S2), destination host (dest), two edge routers (E1, E2) and core router (Core). S0 (audio – EF PHB), S1 (video –AF PHB), and S2 (FTP- AF PHB) perform unidirectional data transmission across the bottleneck link (Core-E2) to dest. We analyze (un)responsive AF and EF flows, number of allocated flows, effects of one aggregate over the other specifically the effects of unresponsive UDP over responsive TCP.

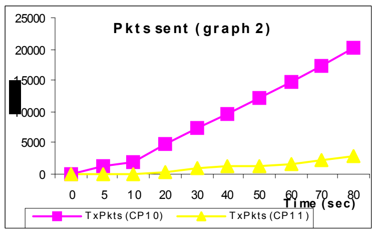
Experiment 1: In this experiment we observe that all flows are allocated resources given their needs and effect of one flow aggregate do not affect the others.

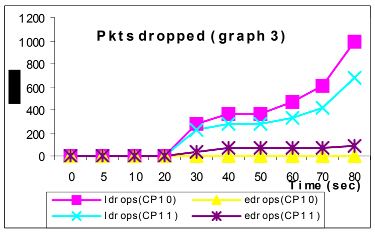
Rate $_{S0}$ = 5 Mbps; UBW $_{S0}$ = function (2.6) = 2.6 Mbps.- accepted.

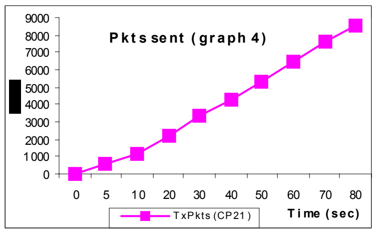
Rate $_{S1}$ = 1 Mbps; UBW $_{S1}$ = function (1) = 1 Mbps.-accepted.

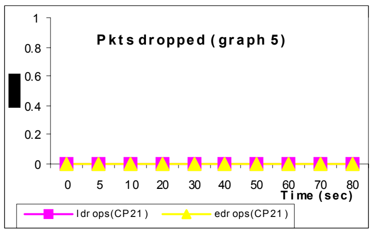
Rate $_{S2}$ = 3 Mbps; UBW $_{S2}$ = function (1, 15) = 1.5 Mbps.-reduced and accepted

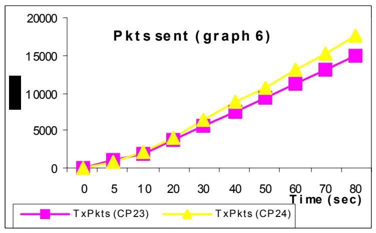


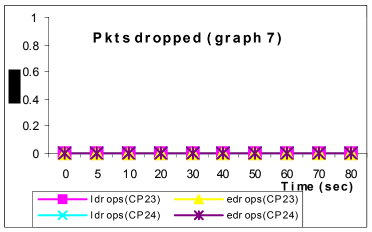












Node	CIR (Mb)	PIR (Mb)	CBS (KB)	RATE (Mb)
S0	1	2.6		5
S1	1		5	1
S2	3		5	

Figure 4.8: EF Audio dropped and AF FTP downgraded

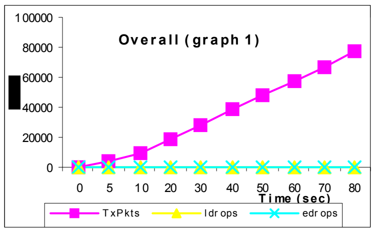
Router Response: From fig 4.8, S0, S1 are allocated full resources while S2 is allocated less. Out-of-profile S0 incurs packet drops while S2 is downgraded. In-profile S1 is unaffected by rogue S0. S2 which runs FTP is not starved on account of S0 going out-of-profile, rather it is itself downgraded.

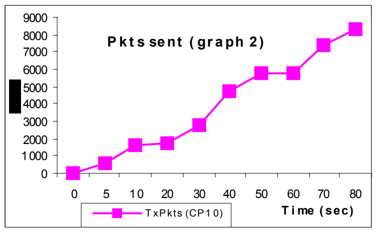
Experiment 2: In this experiment we observe that all flows are allocated resources given their needs and effect of one flow aggregate do not affect the others specifically the effect of TCP going out-of-profile on UDP.

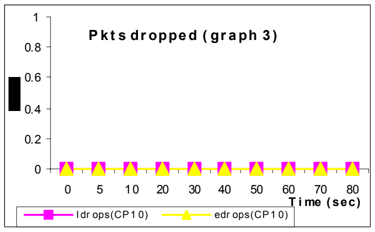
Rate $_{S0}$ = 2.6 Mbps; UBW $_{S0}$ = function (2.6) = 2.6 Mbps.- accepted.

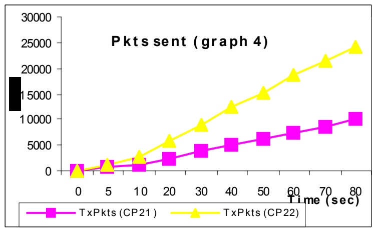
Rate $_{S1}$ = 4 Mbps; UBW $_{S1}$ = function (1) = 1 Mbps.-accepted.

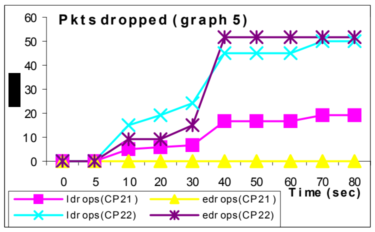
Rate $_{S2}$ = 1 Mbps; UBW $_{S2}$ = function (1) = 1 Mbps.- accepted

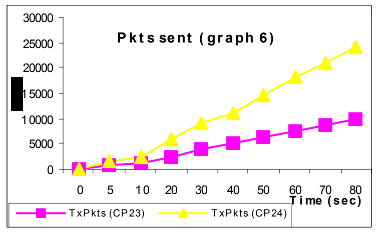


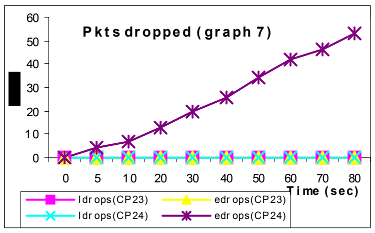












Node	CIR (Mb)	PIR (Mb)	CBS (KB)	RATE (Mb)
S0	1	2.6		2.6
S1	1		5	4
S2	1		5	

Figure 4.9: Effects of AF-FTP, Video on EF Audio

Router Response: S0, S1 and S2 are allocated full resources. From graph 4,5 in fig 4.9, we see that S1 is downgraded and dropped. Graph 6,7 in fig 4.9 show that FTP application flow experiences downgrade of code point but as seen in graph 2,3 in fig 4.9, the EF flow isn't affected by these flow aggregates. Out-of-profile S1, S2 are downgraded while in-profile S1 is unaffected. We see that TCP flow is not starved but is penalized for not conforming to SLA. Furthermore it does not affect in-profile audio (UDP).

Chapter 5

Conclusions and Future Work

5.1 Summary of Results

Multimedia flows are analyzed, allocated resources and regulated before being allowed to use the network. CABB effectively ordered flows so that they confirm to the traffic profile as agreed in the original SLA or the reduced profile arrived at using the flexibility number. We see that queue management and diffserv policing work such that when a flow goes out-of-profile its packets are downgraded and eventually dropped thus regulating a flow. EF flows serviced by priority queues are allowed to go through with minimal drop. Among flows, the downgraded DSCP faces harsher penalty compared to the initially allocated one. The out-of-profile AF flows does not affect the throughput of EF flows and vice versa. Multimedia applications, due to their flow requirements are usually assigned to EF flows but those that can tolerate losses are also assigned to AF flows depending on the application's or broker's decision. Hence, they can go through a congested network through some form or other with CABB doing coarse tuning of bandwidth requirements (avoiding overallocation) and application level adaptive QoS doing the fine-tuning of the applications' response/sensitivity to network changes [9]. This is done such that the end user perceives a quantifiable QoS even with lesser-allocated resources. CABB can ensure higher flow throughput by identifying and controlling rogue flows. Also, CABB eliminates the need for all applications ever having to deal directly with the diffserv router for resources [7]. Both TCP and UDP flows get a fair share of the network. TCP flows are not starved by rogue UDP flows and at the same time are regulated to confirm to SLA.

The normal broker does not account for application's adaptability leading to overallocation of resources. It allocates flows based on available resources and hence cannot ensure that a rogue flow will not affect flows belonging to same or other service classes. We see that compared to a normal broker the CABB efficiently utilizes the network by allowing only those flows that do not congest it.

5.2 Conclusions

A reservation-based environment requires end-to-end multi-resource reservation plans. In this paper we discussed the CABB architecture in a diffserv environment. The CABB decides a policy for a particular flow based on end user service level agreement, the flow's characteristics (flexibility), network resource availability and interbroker communication to provide certain application adaptive level of end-to-end QoS under the constraint of current resource availability. CABB's intelligent policy decisions regarding multimedia flows prevented congestion (congestion avoidance as against congestion control) in the downstream network. These policy decisions were simple, unbiased and effective to allow enough flows at a time such that the network is not unduly loaded thereby controlling traffic at the edge router keeping the core simple. This design is easily scaleable since no state is maintained in routers.

Multimedia applications or those that use UDP for data transmission are now coarsely controlled by the broker's policy decision. The broker thoroughly considers the applications demands before allocating resources ensuring that the flow will confirm to the profile else be downgraded. Thus it intelligently allows flows to utilize resources and maximize the throughput without congestion.

5.3 Contributions

- Design and implementation of a content-aware bandwidth broker (CABB) to provide content adaptive brokering for a better QoS to end users of multimedia applications in heterogeneous environments. Mechanisms are provided for inter-broker communication to reserve resources till destination.

- Policy algorithms that allocate and reserve resources based on the flexibility of the application to network level parameters such as delay, loss and jitter.
- Experimental study of flexible (loss based) adaptations for streaming applications

5.4 Future Work

This work can be extended further. Our next step is to enhance the CABB's intelligence with a better understanding of the distribution profile of the usage of multimedia applications or flows and/or users and also to make the policy decisions more dynamic. Using active resource management ideas we can build on a more dynamic scheme to increase available resources to help the CABB in allocating more flows. These ideas can be carried over from the wired network topology (our current emphasis) to a wireless network for total QoS solutions involving wired/wireless transmissions. Our work at this stage was simulated and analytical and we aim to implement these ideas on a physical test network.

References

[1] B. Li, D. Xu, K. Nahrstedt, J. W. S. Liu, "End-to-End QoS support for Adaptive Applications Over the Internet," SPIE International Symposium on Voice, Video and Data Communications, pp 147 – 161, November 1998.

[2] Nichols K., Jacobson V., Zhang L., "A Two-bit Differentiated Services Architecture for the Internet," RFC 2638, July 1999, see http://www.cs.wisc.edu/~cs640-1/papers/jacobson.qos.ps

[3] Bonald T., Proutière A., Roberts J. W., "Statistical Performance Guarantees for Streaming Flows using Expedited Forwarding," INFOCOM 2001, 26-28, Tallinn, Estonia, April, 2001.

[4] Xiao X., Ni L. M., "Internet QoS: The Big Picture," IEEE Network Magazine, March/April, pp. 8-18, 1999.

[5] Peterson L., Davie B. S., "Computer Networks - A Systems Approach," pp. 446-514, 2nd edition, 2000, Morgan Kaufmann Publishers.

[6] D.D.Clark, S. Shenker, and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanisms," Proc. ACM Sigcomm 92, pp. 14-26, ACM Press, New York, Aug. 1992.

[7] Jitae Shin, Jong-Won Kim, and C.C Jay Kuo, "Content-Based Packet Video Forwarding Mechanism in Differentiated Services Networks," Proceedings of the Packet Video Workshop'2000, May 2000

[8] Prashant Chandra, Allan L. Fisher, Correy Kosak and Peter Steenkiste, "Network Support for Application-oriented QoS," pp. 187-195, Sixth International Workshop on Quality of Service (IWQoS' 98).

[9] Narendra Shaha, Ashish Desai, Manish Parashar, "Multimedia Content Adaptation for QoS Management over Heterogeneous Networks," to appear in International Conference on Internet Computing 2001, Las Vegas, USA

[10] L. W. Ilias Andrikopoulos and G. Pavlou, "A fair traffic conditioner for the assured service in a differentiated service internet," in *Proceedings of IEEE International Conference on Communications (ICC 2000*), New Orleans, LA, June 2000.

[11] Resource ReSerVation Protocol (RSVP), RFC 2205, Sep 1997.

[12] Multiprotocol Label Switching Architecture (MPLS), RFC 3031, Jan 2001

[13] K Kim, K. Nahrstedt, "QoS Translation and Admission Control for MPEG Video", In A. Campbell and K. Nahrstedt, editors, *Proc. of 5th. International Workshop on Quality of Service IWQoS'97)*, New York,May 1997.

[14] Xu D., Nahrstedt K., Viswanathan A., Wichadakul D., "QoS and Contention-Aware Multi-Resource Reservation," In Proceedings of the 9th IEEE International Symposium on High Performance Distributed Computing (HPDC-9), Pittsburgh, PA, August 2000.

[15] IETF "Differentiated Services" Working Group, see http://www.ietf.org/ids.by.wg/diffserv.html

[16] QBone, Simple Inter-domain Bandwidth Broker Signaling, Internet 2, see http://qbone.internet2.edu/bb/

[17] C. Courcoubetis, A. Dimakis, "Providing bandwidth guarantees over a best-effort network: call-admission and pricing", submitted to Infocom 2001.

[18] T. K. Lee, M. Zukerman, R. G. Addie, "Admission Control Schemes for Bursty Multimedia Traffic," to appear in Infocom 2001, April 22-26, 2001

[19] K. Nahrstedt, J. M. Smith, "The QoS Broker," IEEE Multimedia, Vol 2, No 1, pp. 53-67 (1995)

[20] J. C. Bolot, "End-to-End Packet Delay and Loss Behavior in the Internet," Journal of High Speed Networks, Vol 2, pp.305-323, September 1993.

[21] B. Gaidioz, P. Primet, B. Tourancheau, "Differentiated fairness: Service model and implementation," In *Proceedings of High Performance Switching and Routing (HPSR) 2001*, pages 260-264, Dallas, USA, May 2001. IEEE
[22] A. Hu, "Video-on-Demand Broadcasting Protocols: A Comprehensive Study," In Proc. IEEE Infocom'01 (Anchorage), April 2001.
[23] N. Venkatasubramanian, K. Nahrstedt, "An Integrated Metric for Video QoS," Proceedings of ACM Multimedia '97. Also to be published as a UIUC technical report.
[24] P. Marbach, "Pricing Differentiated Services Networks: Bursty Traffic," In proceedings of Infocom 2001.
[25] Ramanathan A., Parashar M., "Active Resource Management for The Differentiated Services Environment," Third Annual International Workshop on Active Middleware Services, Network Services Session, San Francisco, CA, August, 2001
[26] S. Wang, D. Xuan, R. Bettati, W. Zhao, "Providing Absolute Differentiated Services for Real-Time Applications in Static-Priority Scheduling Networks," submitted to IEEE/ACM Transactions on Networking.
[27] Q.Zhang, W. Zhu, Y. Zhang, "Resource Allocation for Multimedia Streaming over the Internet," Multimedia, Vol 3, No. 3, September 2001 339
[28] J. Liebeherr, "Multimedia networks: Issues and challenges," April 1995 (Vol. 28, No. 4). pp. 68-69 Multimedia networks
[29] The Network Simulator - NS-2, see http://ww.isi.edu/nsnam/ns/
[30] Information and Telecommunication Technology Center (ITTC) in IP QoS research, see http://qos.ittc.ukans.edu.
[31] V. Marques, R. Cadime, A. de Sousa, A. M. O. Duarte, "DMIF based QoS Management for MPEG-4 Multimedia Streaming: ATM and RSVP/IP Case Studies," In proceedings of ConfTele 2001
[32] I. F. Akyildiz, J. McNair, "Medium Access Control Protocols for Multimedia Traffic in Wireless Networks," IEEE Network-July/August, 1999
[33] P. Pieda, N. Seddigh, B. Nandy, "The Dynamics of TCP and UDP Interaction in IP-QoS Differentiated Services Networks," Presented at the 3rd Canadian Conference on Broadband Research, November 1999.
[34] M. Baines, B. Nandy, P. Pieda, N. Seddigh, "Using TCP Models To Understand Bandwidth Assurance in a Differentiated Services Network"
[35] R. Rejaie, M. Handley, D. Estrin, "Quality Adaptation for Congestion Controlled Video Playback over the Internet," In Proceedings of ACM SIGCOMM '99, Cambridge, MA, September 1999.
[36] P. P. Mishra, H. Saran, "Capacity Management and Routing Policies for Voice over IP Traffic," Presented at Comsoc March 2002
[37] S. Jo, P. E. Cantrell, "A Dynamic QoS Control Scheme for Videoconferencing in a Heterogeneous Internet," 1999, INET 99, San Jose
[38] R. Keller, S. Choi, M. Dasen, D. Decasper, G. Fankhauser, B. Plattner, "An Active Router Architecture for Multicast Video Distribution," Infocom 2000, Tel Aviv, March 2000.
[39] C. Gbaguidi, H. J. Einsiedler, P. Hurley, W. Almesberger, J Hubaux, "A Survey of Differentiated Services Proposals for the Internet," Tech. Report SSC/1998/020, http://sscwww.epfl.ch, May 1998.
[40] Implementation of the Two Tier Differentiated Services Architecture, Computer Science, University of California, Los Angeles, see http://irl.cs.ucla.edu/twotier/
[41] R. L. Carter, M. E. Crovella, "Measuring Bottleneck Link Speed in Packet-Switched Networks, " Tech. Rep. 1996, Department of Computer Science, Boston University 1996.
[42] QoS Protocols and Architectures, white paper. See: http://www.stardust.com/qos/whitepapers/protocols.htm.
[43] Differentiated Services (diffserv), RFC 2475, Dec 1998

[44] Common Open Policy Service (COPS), RFC 2748, Jan 2000
[45] Manish Mahajan, Manish Parashar, "Content Aware Bandwidth Broker," Technical Report, Department of Electrical And Computer Engineering, Rutgers University, December 2001 (Available at http://www.caip.rutgers.edu/~manishm/bandwidthbroker/CABB.PDF).