Smart Box (SBoX) Architecture for Guaranteed QoS in IP Networks

Bülent Yener * Gong Su † Eran Gabber ‡

Extended Abstract

The explosive growth of the Internet is accompanied by two trends. First, the IP-based networking is expected to be the dominant infrastructure. Second, QoS-sensitive applications such as IP telephony, Virtual Private Networking, Real-time transactions, IP HDTV are expected to be carried on the IP-based networks. These two trends are translated to following challenge: how to provide *deterministic* Quality of Service (QoS) in the IP-based Internet. Fundamentally the IP-based networking is designed for delivering data traffic with best-effort (i.e., no guarantee) service, thus it is not capable of providing end-to-end QoS.

Several architectures have been proposed for providing QoS in the Internet: The *integrated services* (Intserv) model is based on reservations and can provide QoS, however; it is not scalable. The *differentiated services* (Diffserv) approach is scalable but falls short of ensuring deterministic guarantees —in particular for the services that belong to the same class. Finally, the *multi protocol label switching* (MPLS) architecture provides mechanisms for QoS-based routing but does not have the necessary resource management and scheduling support to ensure it.

This work proposes a hybrid solution which combines the best of these technologies. First, at the network boundary Diffserv like Service Level Agreements (SLA) are provided to users by the intelligent edge routers called the **SBoX servers**. Each SLA is a *macro-flow* with a specific bandwidth and established between two hosts. A macro-flow is shared by all the application-level micro-flows between these hosts. All the macro-flows in the SLA are aggregated to a *commodity-flow*. The packets of the same commodity-flow are marked by an MPLS label, which is globally unique within an Autonomous System (AS). The SLAs are enforced and packed into a commodity-flow by Class Based Queuing (CBQ) packet scheduling. Thus, the SBoX server is an intelligent edge device which combines the Diffserv and MPLS architectures at a network access point.

Second, inside the network, a subset of the best-effort IP routers are enhanced with a programmable, add-on Label Switching Router (LSR) called the **SBoX router** (SBoX). An SBoX router enforces deterministic service guarantees on some of the links of a best-effort IP router. The SBoX router label switches the commodity-flows using their global labels and protects them from each other and from best-effort traffic using CBQ. SBoX add-on routers address two current problems: (1) the lack of end-to-end deployment of LSRs (i.e., many routers are not MPLS capable), and (2) the lack of programmability of the legacy routers. An add-on approach provides service guarantees in a network that contains legacy routers without replacing or changing them. Thus, the SBoX router performs MPLS with CBQ over IP, in a totally transparent way to current IP-based network infrastructure and protocols.

Third, new algorithms are proposed for provisioning, allocation of the network resources to CBQs and admission control. As the core of provisioning and resource allocation problem, we consider where to place the SBoX routers and which links to enforce QoS guarantees. The SBoX routers and the links that they manage induce an virtual premium network called the **reserved-net**. QoS based routing is confined to the reserved-net.

Given the pairwise statistical demands for QoS traffic over a fixed interval, we formulate the problem designing an optimal reserved-net as a multi-commodity problem with integer variables. The objective function is to *pack* the commodity-flows into a minimal feasible subgraph while avoiding bandwidth fragmentation on the links. An SBoX router is associated (activated) with every node that appears in the solution of this optimization problem. In other words, the SBoX router controls the links of such nodes, so that it switches labeled traffic directly, and passes other traffic (best effort) to the legacy router.

The solution to the optimization problem also gives the amount of bandwidth allocation necessary for each commodity. Bandwidth allocation and protection is done by setting CBQ mechanisms in the involved SBoX routers. The parameters of the CBQ scheduling and the labels of macro-flows are signaled with a light weight protocol which aims to minimize the communication overhead.

This paper explains the above issues of the SBoX architecture in depth and reports experimental results obtained on an experimental network consisting of clients, SBoX servers, and a legacy router (Cisco) controlled by an add-on SBoX router.

*Bell Laboratories, Lucent Technologies, 700 Mountain Ave., Murray Hill, NJ 07974 E-mail: yener@research.bell-labs.com, Tel: (908) 582 7087

†Dept. of Computer Science, Columbia University, NY, NY 10027 E-mail: gongsu@cs.columbia.edu

‡Bell Laboratories, Lucent Technologies, 700 Mountain Ave., Murray Hill, NJ 07974 E-mail: eran@research.bell-labs.com

1 Introduction

As the Internet gets commercialized, the need for providing QoS becomes imminent. Current infrastructure of the Internet cannot support QoS since it is designed for best-effort service model. Its routers operate with FIFO scheduling without any guarantees. Increasing the network capacity by adding more bandwidth and routers is not always feasible or efficient. The network must have mechanisms to distinguish QoS requirements of different applications that share the same infrastructure and process them accordingly.

There are two approaches to the QoS problem in the Internet: (i) *Integrated Services* (Intserv), and (ii) *differential services* (Diffserv). The Intserv approach [SPG97, Wro97, SW97] requires high-end routers to maintain per-micro-flow [1] state information and perform complex link scheduling algorithms. As the number of flows increase and/or change frequently, the overhead of this approach does not scale.

The Diffserv approach aims to [Cla97, CW97, NJZ97, SZ98, ea98] to reduce the per-flow complexity by providing an aggregated treatment of user traffic that belongs to the same service class. However, the Diffserv approach can provide only qualitative QoS. Packets in the Diffserv model are marked, at the network entry points, to indicate whether or not the source follows its Service Level Agreements (SLA). The packets of different SLAs are classified and marked by corresponding Diffserv Code Points (DSCP). The DSCP specifies, at each transit node, the Per Hop Behavior (PHB) which is suppose to determine the packet dropping probability and scheduling treatment [ea98]. Due to the lack of support for sophisticated scheduling algorithms (e.g., Weighted Round Robin, hierarchical FIFO, WFQ, CBQ) on the existing infrastructure, the immediate PHB is based on packet dropping which is based on modification of Random Early Detection (RED) protocol and called RIO. The packets that violate their SLA (i.e, OUT packets) are dropped with a higher probability than the packets obey their SLA (i.e, IN packets). Several performance studies [BW99, IN98] of Diffserv approach show that (1) it cannot offer a quantifiable service to TCP traffic, (2) there is strong dependency between IN and OUT packets (due to shared queue) and consequently IN packets may be dropped, and (3) mixing IN and OUT packets in a single TCP connection reduces the connection's performance. Thus, once a packet enters into the network, there is no *deterministic* guarantee for QoS. Non-deterministic and non-quantifiable QoS cannot be acceptable for certain applications, such as IP telephony, real-time interactive transactions, IP high definition video (e.g., HDTV), and Virtual Private Networking (VPN) for which pricing and QoS are strongly coupled.

Recently, Multi Protocol Label Switching (MPLS) [RVC98] has become an attractive technology. The main contributions of MPLS are twofold. First, it decouples routing from forwarding. Thus, *explicit routing*, based on metrics different from the ones used by the traditional routing algorithms can be deployed. Second, it aggregated the flows to Forwarding Equivalent Classes

[1] A micro-flow is a application level flow defined by its source, destination address, port numbers and protocol identifier.

using virtual circuit switching. The former enables traffic engineering and QoS based constrained routing while the latter provides a scalable solution. Although it contributes significantly, the MPLS is not enough for providing QoS since it does not address resource allocation, scheduling and admission control problems.

Finally, a novel approach called Dynamic Packet States (DPS) [ea99] is proposed to provide guaranteed QoS. The DPS approach requires checking and updating a state information associated with each packet; Thus it introduces extra processing complexity at each node. Furthermore it requires modification of existing routers.

We note that a new generation of routers with QoS enhancements are becoming available. These routers provide complex flow isolation, buffer management and link scheduling algorithms. The complexity of high-end routers necessitates sophisticated processing inside the network at the speeds close to ever increasing data rates in high speed links (e.g., OC192). Consequently, high-end routers are expensive and their deployment may be limited to the network core. To provide end-to-end QoS, the QoS capability provided at the core has to be supplied in the entire network all the way to the edge routers. Currently, there is no such technology available for deterministic QoS.

This work proposes a hybrid solution which combines the best of these technologies. First, at the network boundary Diffserv like Service Level Agreements (SLA) are provided to users by the intelligent edge routers called the **SBoX servers**. Each SLA is a macro-flow with a specific bandwidth and established between two hosts. A macro-flow is shared by all the application-level micro-flows between these hosts. All the macro-flows in the SLA are aggregated to a *commodity-flow*. The packets of the same commodity-flow are marked by an MPLS label which is globally unique within an Autonomous System (AS). The SLA's are enforced and packed into a commodity-flow by Class Based Queuing (CBQ) packet scheduling. Thus, the SBoX server is an intelligent edge device which combines the Diffserv and MPLS architectures at a network access point.

Second, inside the network, a subset of the best-effort IP routers are enhanced with a programmable, add-on Label Switching Router (LSR) called the **SBoX router** (SBoX). An SBoX router enforces deterministic service guarantees on some of the links of a best-effort IP router. The SBoX router label switches the commodity-flows using their global labels and protects them from each other and from best-effort traffic using CBQ. The SBoX routers address the following problems: (1) the lack of end-to-end deployment of LSRs (i.e., many routers are not MPLS capable), and (2) the lack of programmability of the legacy routers. An add-on approach provides service guarantees in a network that contains legacy routers without replacing or changing them. Thus, the SBoX router performs MPLS with CBQ over IP, in a totally transparent way to current IP-based network infrastructure and protocols.

Third, new algorithms are proposed for provisioning, allocation of the network resources to CBQs and admission control. The core of provisioning and resource allocation problem is the placement of SBoX routers and the selection of the links that enforce QoS guarantees. The SBoX routers and the links that they manage induce an virtual premium network called the **reserved-net**. QoS based routing is confined to the reserved-net.

Given the pairwise statistical demands for QoS traffic over a fixed interval, we formulate the problem designing an optimal reserved-net as a multi-commodity problem with integer variables. The objective function is to *pack* the commodity-flows into a minimal feasible subgraph while avoiding bandwidth fragmentation on the links. An SBoX router is associated (activated) with each node that appears in the solution of this optimization problem. In other words, the SBoX router controls the links of such nodes, so that it switches labeled traffic directly, and passes other traffic (best effort) to the legacy router.

The solution to the optimization problem also includes the amount of bandwidth allocation necessary for each commodity. Bandwidth allocation and protection is done by setting CBQ mechanisms in the involved SBoX routers. The parameters of the CBQ scheduling and the labels of macro-flows are signaled with a light weight protocol which aims to minimize the communication overhead.

This paper is organized as follows. In Section 2 we address provisioning and configuration issues. Section 3 presents the components of the SBoX architecture. In Section 4 we explain the prototype implementation and experimental results. Finally we conclude in Section 5.

2 Network Provisioning and Configuration

We consider the IP-based Internet as an interconnection of autonomous systems (AS) in which an ISP has control of all the resources (i.e., it is a single administrative domain). An AS can be accessed through some specific entry points called *Service Access Points* (SAPs). An ISP aggregates the user traffic at the SAPs. The network can be connected to the other networks via border routers (BRs) that are treated as SAPs in our model. SAPs are involved with admission control and perform policy based bandwidth management at network boundary. The discussion in this work is focused on a single AS, and each neighboring AS is considered as an SAP.

We distinguish between two types of traffic carried by the network: (1) reserved or committed QoS traffic, and (2) non-reserved or best-effort traffic. Network resources are provisioned and configured to accommodate to the QoS traffic. Thus, provisioning and configuration require estimating pairwise bandwidth demands for QoS traffic between all SAPs and BRs in an AS. There are several techniques for source modeling that can be adapted for this purpose [ML97]. Provisioning and configuration of QoS traffic in the SBoX architecture are achieved by creating a virtual premium network called the **reserved-net**. Figure 1 depicts the main steps for provisioning and configuration of QoS traffic.

The reserved-net is a virtual dedicated network used to (i) protect the reserved traffic from the best effort traffic, and (ii) forward it via virtual circuit switching. Thus, the physical network is partitioned into two subnets, one for the QoS traffic (the reserved-net) and the other containing the remaining network elements. This partition ensures that (1) each subnet connects all the SAPs and BRs, and (2) each subnet is node (or link) disjoint. The result of this partition is two virtual networks which may share common nodes. Routing on the reserved-net employs virtual circuit switching (MPLS) with CBQ scheduling and done by the SBoX routers. The best-effort traffic may be permitted to use the reserved-net in addition to the second subnet; however, the reserved traffic has preemptive priority.

We note that the problem of provisioning a reserved-net is a version of network optimization problem for which a rich literature exists (e.g., [Dov91a, Dov91b, CFZ94, ACL94, Ash95, For96, Lee95, MMR96]). However the the objective function of our problem is different. In particular, the cost measure is to *pack* the connection requests together so that (i) number of links used by the solution is minimal, and (ii) bandwidth fragmentation is minimized for each link included into the reserved-net. Bandwidth fragmentation is described in detail in the following section. It is a measure of the unused bandwidth in the reserved-net, which is not needed for QoS traffic. The underlying optimization problem can be formulated as an instance of 0-1 integer programming as we discuss next.

2.1 Designing an Optimum Reserved-Net

Network is represented by a graph $G = (V, E)$ where node set V is partitioned into two subsets. Set R contains the routers, set S contains the SAPs. End-points of a macro-flow belong

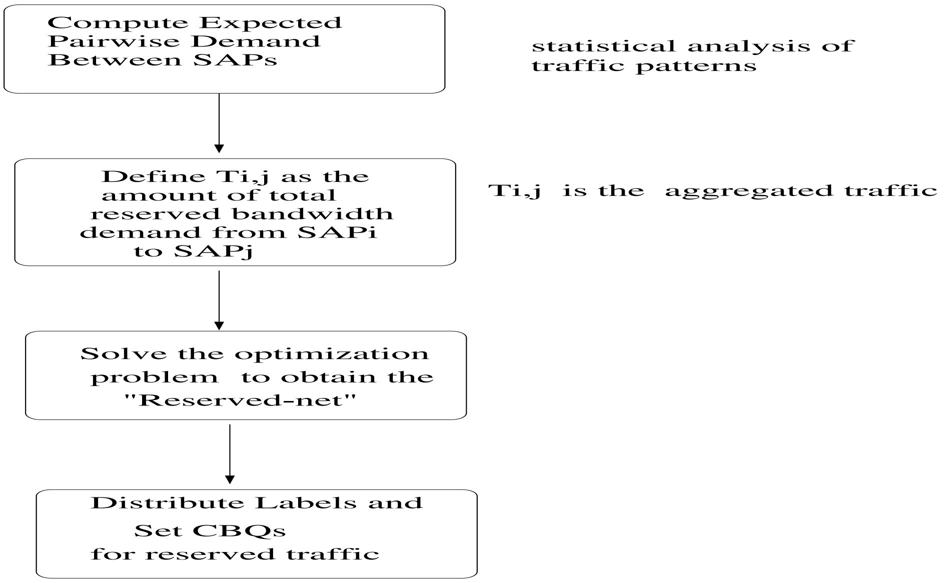

Figure 1: Steps for Provisioning and Configuration of QoS Traffic

to the set S. The set E contains the network links, each of which has capacity (bandwidth) $c_{(i,j)}$ (bits/sec) $\forall (i,j) \in E$. For simplicity we assume a symmetric model so that each direction of a link has the same capacity. The reserved-net is a subgraph $G_r \subset G$, represented as $G_r = (R', S, E')$. Nodes and edges or G_r are called **reserved**. Let T be a traffic matrix in which an entry $t_{i,j}$ indicates the aggregated bandwidth request for QoS traffic from i to j $\forall i,j \in S$. In other words $t_{i,j}$ is the sum of the bandwidth of macro-flows from SAP i to SAP j. We assume that T is based on the statistical information capturing a correlated source behavior (e.g., the traffic volume between 8-11 AM).

The optimization problem can be formulated as an instance of mixed-integer multi-commodity flow problem, where a commodity k is defined for a pair of nodes $k,l \in S$ such that $t_{k,l} > 0$ and $k \neq l$. In other words, a commodity k is provided from each SAP k to another SAP l, so that k has non-zero bandwidth request to l.

Our cost measure is to *pack* the *routes* of reserved traffic together so that (i) number of links used by the solution is minimum, and (ii) bandwidth fragmentation is minimized for each link included into the reserved-net.

Bandwidth fragmentation is defined as the amount of "unused" or "left-over" bandwidth of a reserved link which is smaller than any expected macro-flow bandwidth request. Thus, for each link we have a threshold value $\delta_{i,j}$. The main motivation behind the thresholds is to obtain maximum bandwidth utilization by the QoS traffic. If unused or left-over bandwidth is less than $\delta_{i,j}$ then it is considered to be wasted and the link bandwidth is said to be fragmented. In such case, the variable $y_{i,j}$, which is 1 only if link (i,j) is used, contributes a penalty term to the objective function with a coefficient α as shown in Figure 3. Note that α can be chosen to

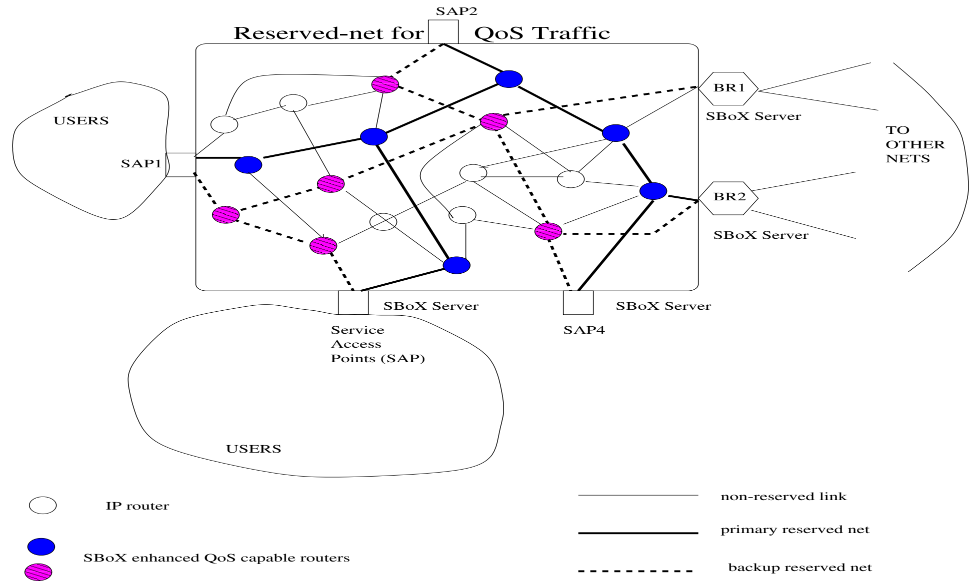

Figure 2: Reserved-Net -partitioning of network resources.

Minimize: C= $\sum_{i,j} x_{i,j} + \alpha(y_{i,j}) \quad \forall (i,j) \in E$ and $\forall k$

Constraints:

s.t.

(1) $\sum_{j \neq i} f^k_{(j,i)} - \sum_{j \neq i} f^k_{(i,j)} = \begin{cases} t_{k,i} & \forall k, i \;\; s.t. \;\; k \neq i \\ 0 & \forall i \in R \end{cases}$

(2) $\sum_k f^k_{i,j} \leq (c_{i,j} - \delta_{i,j}) x_{i,j} + \delta_{i,j} y_{i,j} \quad \forall (i,j) \in E$

(3) $y_{i,j} \leq x_{i,j}$

(4) $x_{i,j} \in \{0, 1\} \quad \forall (i,j) \in E$

(5) $f^k_{i,j} \geq 0$

Figure 3: Mixed-integer formulation of the problem

control the impact of bandwidth fragmentation on the cost function.

The variables and parameters of the optimization problem are as follows:

• Flow variable $f^k_{i,j}$ which takes a real value and indicates the amount of commodity k over the link (i,j).

• A 0-1 integer variable $x_{i,j}$ which indicates the link (i,j) is used by any commodity. It is 1 if used, 0 otherwise. In other words, $x_{i,j}$ denotes whether link (i,j) is included in the reserved-net.

• A 0-1 integer variable $y_{i,j}$ which indicates whether or not the *unused* bandwidth of the link (i,j) is below a given *fragmentation threshold* $\delta_{i,j}$.

Constraint (1) ensures the flow balance for each commodity. If the node is an intermediate node (i.e., $i \in R$ is a router), the in-flow should be equal to the out-flow. Otherwise, the node i is a macro-flow end-node and the flow into this node from commodity k is $t_{k,i}$ for each commodity k. The inequalities (2) and (3) ensure that total load on a link (i, j) does not exceed its capacity. Furthermore, the load is assigned such that the bandwidth fragmentation is minimized by the penalty term $y_{i,j}$. Inequality (3) ensures that the penalty term exists only for the links in the reserved-net. Inequalities (4) and (5) are self explanatory.

The solution to this optimization problem is a set of $x_{i,j}$ and $y_{i,j}$ values, that minimize the cost function while connecting the nodes for each VPN. The links with $x_{i,j} = 1$ are called **reserved-links** and the nodes incident to reserved-links are called **reserved-nodes**. The union of the reserved-links induces a connected graph which is our reserved-net as shown in Figure 2.

A path from SAP_k to SAP_l on the reserved-net is called a **reserved-path** and carries all or a part of the commodity-flow for commodity k. Note that the formulation does not enforce a single path between a pair of SAPs, however it minimizes the number of paths indirectly. It may be desirable to over allocate the capacity for providing fault-tolerance and dynamic admission control. A simple way to achieve this is to add slack variables to the $t_{i,j}$ and solve the above optimization problem.

The solution of the above optimization problem is computationally expensive; thus we may use heuristics to obtain suboptimal solutions. Since our focus is on the QoS aspects of the problem, we will not further elaborate on the optimization issues.

2.2 Heuristic Approach

In this section we propose a simple algorithmic solution to the construction of the reserved-net. The main idea is to build multiple (edge) disjoint trees of G to connect the SAPs. This problem can be treated as an instance of the Steiner Tree problem and several heuristics are known for finding approximate solutions for a steiner tree.

In particular, we consider building two steiner trees which have minimal number of steiner vertices (i.e, vertices other than the SAPs) in common. One tree is used as the primal routing tree, while the other tree is kept for backup. Of course, the minimum capacity on tree links must be at least the maximum bandwidth provisioned for the reserved traffic.

The algorithm has three phases. First, we eliminate all the links that do not have the capacity to handle aggregate bandwidth demand of the reserved traffic. Second, we construct 2 edge-disjoint spanning trees of the modified network graph. It is known that there are at least

$\lfloor \frac{c}{2} \rfloor$ edge-disjoint spanning trees in a graph with edge connectivity c. Third, the trees are refined to trim the branches that have no SAP nodes as their leaves.

Assuming that reserved-net is obtained from the above heuristic approach, there is a single reserved-path between any pair of SAPs. Thus, the QoS based explicit routing for MPLS is loop-free and obvious. Finally, we remark that there is a one-to-one correspondence between each commodity k and the reserved-path to carry k.

2.3 Resource Allocation and Label Distribution

The solution to the optimization problem determines reserved-links and the amount of flow that can be assigned to these links. In particular the $f_{i,j}^k$ value indicates the bandwidth request of commodity-flow on link (i,j) that resides in the reserved-path from SAP_k to SAP_l. In order to enforce service guarantees, $f_{i,j}^k$ must be reserved and protected on link (i,j).

In the proposed architecture, an SBoX router is associated with each reserved-node to enforce the service guarantees on each reserved-link. An SBoX router uses CBQ to protect the commodity-flows from each other and from the best effort traffic. In particular, two mechanisms are deployed within an SBoX router. First, an MPLS "label" is assigned to each commodity k. Second, a "class" with bandwidth $f_{i,j}^k$ is defined using CBQ discipline on each reserved-link (i,j). The filtering of the traffic in the CBQ is based on the MPLS labels. We will explain the details of CBQ settings in an SBoX router in Section 3.

Different policies may be adapted to adjust the interaction between the reserved and the best-effort traffic. We propose a **movable boundary** scheme, in which the best-effort can use the available reserved bandwidth, where reserved traffic has preemptive priority as shown in Figure 4. The position of the boundary can be dictated by the network economics and utilization. For example, an ISP may decide to reserve only 60% of link capacity to the aggregated reserved-traffic.

The policy for setting such a boundary can be enforced in the optimization problem by the threshold variable $\delta_{i,j}$ and the coefficient α. Our simulation study shows that when the average utilization of the reserved bandwidth is at least 85%, the best effort packets will not benefit from the movable boundary scheme.

Distribution of MPLS labels and setting the CBQs can be done using a signaling protocol such as RSVP or a Label Distribution Protocol (LDP). In the next section we present a protocol

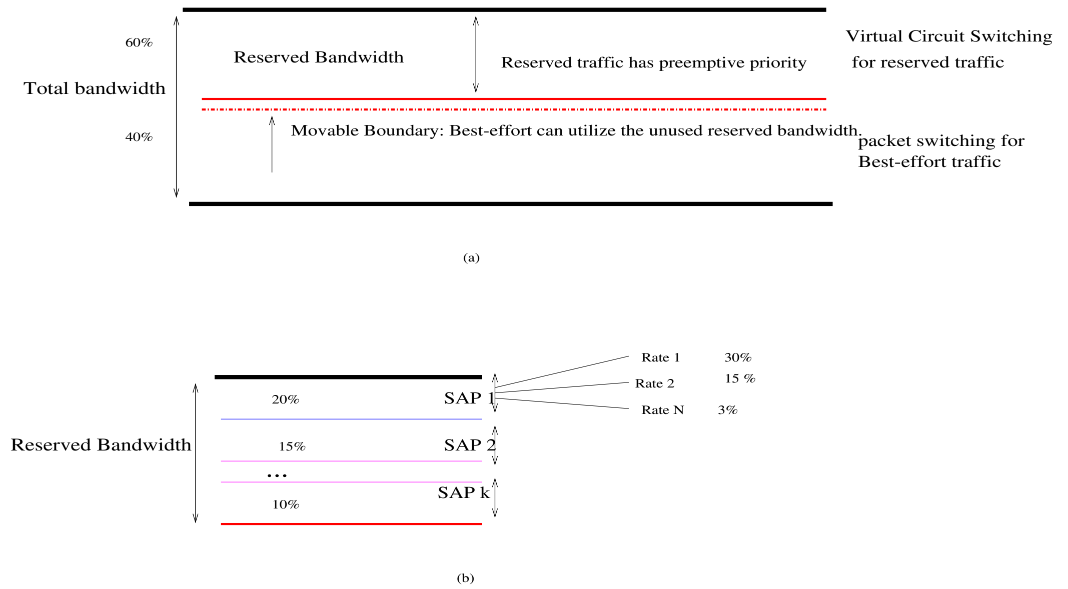

Figure 4: Best effort vs. reserved traffic bandwidth configuration

based on 3-way handshake that can be an extension to RSVP.

2.4 Light Weight Signaling Protocol for Reservations

The signaling protocol has the following steps:

1. SAP_x makes a request of R bps to a SAP_y for the corresponding commodity-flow by

1.1 sending $REQ(Label, R, seq\#)$ message.

1.2 starts a timer for reply.

2. each SBoX router on the reserved-path

2.1 marks the forwarding table for rate R and $Label$ and

2.2 appends its ID to the REQ message.

2.3 (marking is not a commitment but a tentative state. It is associated with a timer and if no reply (REP) message is received, then there will be timeout and the tentative commitment will be reset).

3. upon receiving the REQ message, the SBoX server at SAP_y

3.1 it sends $REP(Label, R, seq\#)$ message back by reversing the reserved-path recorded into the REQ message.

3.1 sets a timer for confirmation (CONF) message

4. each SBoX router on the path

4.1 identifies the REQ/REP using the Label,

4.2 *commits* for the rate R that REP message carries,

4.3 unmarks the REQested rate

4.4 starts a timer for confirmation

5. upon receiving the REP message, the SAP_x confirms the reserved-path by

5.1 sending a $CONF(Label, R)$ message to SAP_y or

5.2 starts transmitting data.

6. if the timer in SAP_x times out for the REQ message then it resends a REQ message with an incremented seq #.

7. upon committing to a REQ message if an SBoX router receives no CONF message or data then time out and undo the commitment.

Having a reserved-net ensures that the routing problem is simplified, and for any reserved-link (i, j), $\sum_k f_{i,j}^k \leq c_{i,j}$. Thus, the protocol is not concerned with the feasibility, but simply consistency.

3 SBoX Architecture

The SBoX architecture comprises two components: (1) SBoX servers, and (2) SBoX routers. The SBoX servers involve with admission control and management at the network edge. SBoX routers are the interior network nodes that perform only label switching of macro flows for QoS based routing. We will describe the SBoX servers and routers in detail in this section.

3.1 Intelligent Edge Routers: SBoX Servers

The SBoX architecture deals with the complexity of QoS support by delegating it to the SBoX servers. An SBoX server is an **intelligent edge-router** that can reside at an SAP and in charge of managing the resources and requests from SAPs. A SBoX server is a programmable device

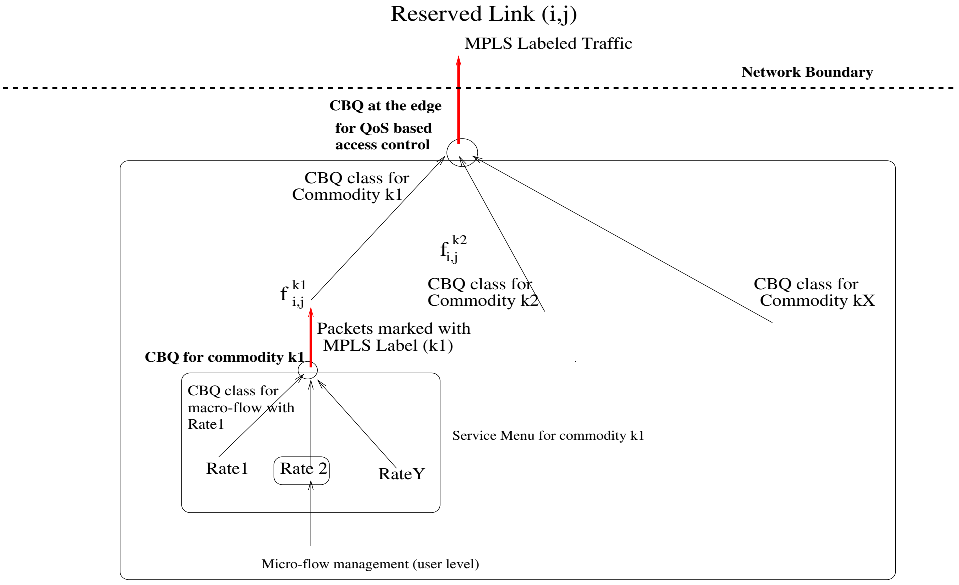

Figure 5: CBQ hierarchy and labeling maintained by an SBoX Server

with an open architecture. It can separate QoS traffic from best-effort. It labels the QoS traffic for MPLS routing inside the network. An SBoX server uses hierarchical traffic management mechanisms which can be configured dynamically to adapt to the changing policies of an ISP. We note that since the adjacent AS are treated as SAPs, their resource management is also handled by SBoX servers.

3.1.1 Traffic Management and Labeling

SBoX server is a configurable device which has hierarchical traffic management capability. Figure 5 depicts the hierarchical management of a reserved-link (i, j) by an SBoX server. At the top of the hierarchy, the bandwidth of (i, j) is shared by multiple commodity-flows whose bandwidth allocation is determined by the optimization problem presented in section 2.1. Link (i, j) needs to be managed to ensure: (1) protection of commodity-flows from best effort, and (2) protection of each commodity-flow from each other. The second level of the hierarchy concerns with the management of each commodity-flow. Each commodity-flow is divided into a set of macro-flows which are offered as service level agreements (SLAs) to users of a SAP for this commodity. Thus, each SLA provides a predefined service rate which are offered by a **service menu**. In the last level, macro-flows are managed in application level flow (i.e., micro-flow) granularity. We will explain the service menus and management of macro-flows in the next section.

SBoX servers performs IP-MPLS mapping at the commodity-flow level. Each commodity flow is associated with an MPLS label which can be global within the AS [2]. Packets of each commodity-flow are marked by the MPLS label. As shown in Figure 5, the SBoX server maintains a CBQ scheduling mechanism for each commodity-flow that uses edge (i, j). Each commodity is considered as a class and classes are strictly isolated from each other (i.e., fire-walled). The resource sharing between best-effort and QoS traffic is based on the movable boundary scheme as explained above. Packets of each commodity are filtered and scheduled based on their MPLS labels.

An SBoX server also maintains billing and accounting information at different levels of granularity. It can provide call description record (CDR) type of information at the commodity-flow or at the macro-flow level. We will explain billing and charging issues at the next section further.

3.1.2 Admission Control: Service Level Agreements

The committed traffic is accepted to the network through admission control, which ensures that bandwidth allocation satisfies the service descriptions of user requests. An important part of admission control is monitoring and managing the *available bandwidth* for each commodity.

The SBoX architecture provides various data rates called **service rates**, that are a multiple of a base rate (e.g., 1Mbps, 5Mbps, 10Mpbs, 20Mbps) in a service menu. Each data rate represents a different SLA. Users sign a contract with their provider by picking an SLA from a service menu.

As shown in Figure 5, each SLA corresponds to a macro-flow. It is left to the users/enterprises to manage the access to the pipe. There are several commercial products available to differentiate business critical traffic based on enterprise policies (e.g., Xedia's Accesspoint, Packeteer's Packetshaper, Checkpoint's Flood Gate, Allot's AC 200-330).

The challenge is how to decide which service rates to offer and how many requests to grant for each (macro-flow) rate. We believe that the answer should have low overhead and also address the network economics. For example, each service rate in the service menu can be assigned a number of **tokens** (permits) during provisioning. The weighted sum of the tokens is equal to the total amount of reserved traffic that the provider is willing to accept (where the weights represent

[2]Note that 20 bits from the 4-byte MPLS label can support 2^{20} commodities which corresponds to approximately 10^3 SAPs within an ASP. If the number of SAPs exceeds such number then labels must be swapped and cannot be global.

the data rates). Each SAP and BRs are given a set of tokens based on (i) the expected number of connections, and (ii) the bandwidth demands between them. Thus, each SAP/BR can make a local decision how much traffic to accept to the reserved-net using the token/permit pool it has. For example, lets refer to Figure 5. Suppose that the SAP is given x tokens of 512Kbps and y tokens of 1Mbps for commodity k_1. Then this SAP can accept at most $x + y$ requests for this commodity such that $x512 + y100 \leq f_{i,j}^{k_1}$ kbps.

Since users must pick an SLA from a service menu in advance, the policing, shaping and scheduling in the network can be set ahead of time, even before the SLAs are offered. Thus, there are two advantages of providing service menus of SLAs in the SBoX architecture. First, it reduces the traffic management overhead at the commodity-flow level. For example, if we replace the predetermined rates with a dynamic management scheme then requests for macro-flows must be accommodated on demand basis. This requires setting the traffic filters and shapers on demand as well. Thus, it may increase the processing time for requests and may delay responses to requests. Second, it provides efficient billing and accounting mechanism to an ISP. For example, each token can have a price based on the rate that it is associated with. Thus charging can be reduced to the token holding time multiple with the price of this token. The price of a token can be changed based on the demand for bandwidth. For example a discount price can be offered during off-peak hours while price can be increased during peak hours.

3.2 Dedicated Label Switching Routers: SBoX Routers

An SBoX router is a programmable, stand-alone, label switching router with a limited number of network interfaces. An SBoX router cannot perform IP-based routing and its routing capability is limited only to label switching with CBQ scheduling.

Provided that legacy routers are vertically integrated closed boxes, an SBoX router can be deployed as an add-on device [3]. An SBoX router controls the reserved links, identified by the reserved-net, incident to a legacy router, as shown in Figure 6.

The number of network interfaces and the line speeds that an SBoX supports are dictated by the network economics and considered to be outside the scope of this paper. However such constraints can be accommodated into the formulation of the optimization problem to limit the

[3] Certainly, it would be the best if an SBoX can be transplanted into a legacy router however current legacy routers do not provide such modular and open architectures.

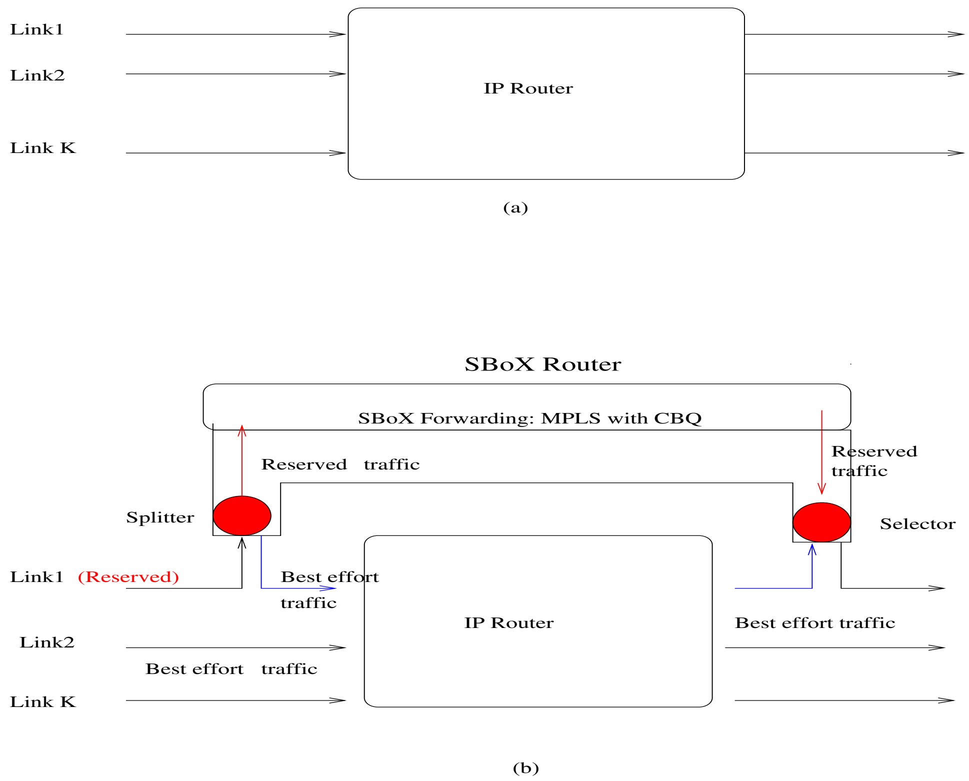

Figure 6: SBoX Operation: QoS traffic has precedence over best-effort traffic. Best-effort traffic uses either links that are not a part of the reserved-net or the idle portion of reserved link. Combination of SBoX and best-effort router provides functionality found in a *dual-switch* [Kum74, Zaf74, OY94, NBS95].

number of reserved links incident to a legacy router that can be in the solution. The main functions of the SBoX are as follows:

Fget: intercept the traffic into the *reserved node*;

• **Fchk:** check the header to determine its type;

• **FcuthruS:** IF the packet belongs to a macro flow then cut through to the SBoX;

• **FcuthruR:** ELSE forward the packet to the same input link incident to the router.

• **Fdrop:**(optional) if the packet is not a IP control packet.

• **Fnext:** check the label and the outgoing link;

• **Fvc:** perform MPLS compliant virtual circuit switching with per commodity-flow CBQ scheduling.

• **Fblock:** prevent the router to access the output link if the SBoX has a packet to be transmitted to it as well.

Some of these functions are performed by two network interface modules called the **splitter**, and the **selector**. The splitter intercepts all incoming packets, examines their header information, and decides where to forward the packet (outgoing link for MPLS cut-through, or an input link of the best-effort router. The splitter performs the above **Fget**, **Fchk**, **FcuthruS**, **FcuthruR**, and **Fdrop** operations. The splitter uses a buffer to store the initial portion of incoming packets until the label is processed. If the packet does not carry a label, then it is considered as a best-effort, and it is forwarded immediately to the best-effort router without waiting and storing the rest of the packet. The size of the buffer is a function of the link rate and the splitter processing speed.

Contention occurs if best-effort router is allowed to forward packets to a reserved link. Thus we need to control the access to the output link. The **selector** is an arbitrator circuit, which ensures that as long as there is data in the SBoX's output buffer, the reserved link cannot be used by the legacy router. The splitter is functionally a a 2x1 multiplexer. The inputs of the multiplexer are the SBoX and the best-effort router. The multiplexer selects the SBoX as long as it has data. Since the selector may not accept packets from the router at certain times, it may signal the router to block in order to avoid excessive packet loss. To make this activity transparent to the router, the selector emulates a point-to-point link. In particular, it may perform one of the following operations on the best-effort link: (1) accept the packet and drop it, (2) send a control signal (e.g., SEND-NO-PACKETS) back to the router asking for withhold transmissions, (3) deceive the router that link is down. The splitter and selector reside inside

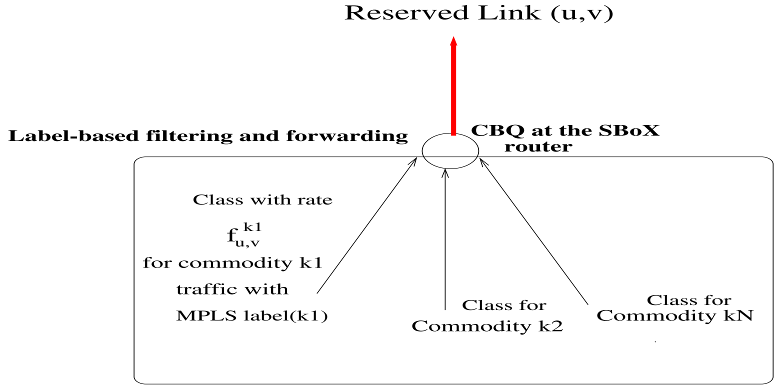

Figure 7: CBQ Class hierarchy in an SBoX router

the SBoX and their functions can be combined in an interface card with two ports and a limited processing capability.

3.2.1 Forwarding (MPLS) and Protecting (CBQ) the flows

While the best-effort router performs IP-based routing, packet forwarding in an SBoX is based on label switching using MPLS. Upon entering the network, QoS traffic is handled exclusively by the SBoX routers. Thus it bypasses all best effort traffic. Each SBoX router has a **forwarding table**, associated with each link. An entry in the forwarding table indicates the outgoing link and the reserved bandwidth for a commodity-flow identified by its MPLS label.

The protection and enforcement of service guarantees is done with CBQ as shown in Figure 7.

4 SBoX Testbed Prototype

The testbed for SBoX project consists of seven Intel Pentium IIIs and a Cisco 2621 router connected with the topology depicted in Figure 8. Four of the seven PIIIs are running Windows NT, a pair of which for the Lucent MPEG video streaming (*mpegsrvr* and *mpegclnt*) and the other pair of which for the Lucent Cineblitz video streaming (*cnbzsrvr* and *cnbzclnt*). Three of the remaining PIIIs are running Linux with MPLS and CBQ support compiled into the kernel, two of which are the SBox servers (*sboxsvr1* and *sboxsvr2*) and one of which is an SBox router (sboxrtr1) managing commodity-flows inside the network. The Cisco 2621 router (sboxrtr2) plays the role of a conventional IP router for best-effort traffic.

One MPEG video stream of approximately 4Mbps flows from *mpegsrvr* to *mpegclnt*. Multi-

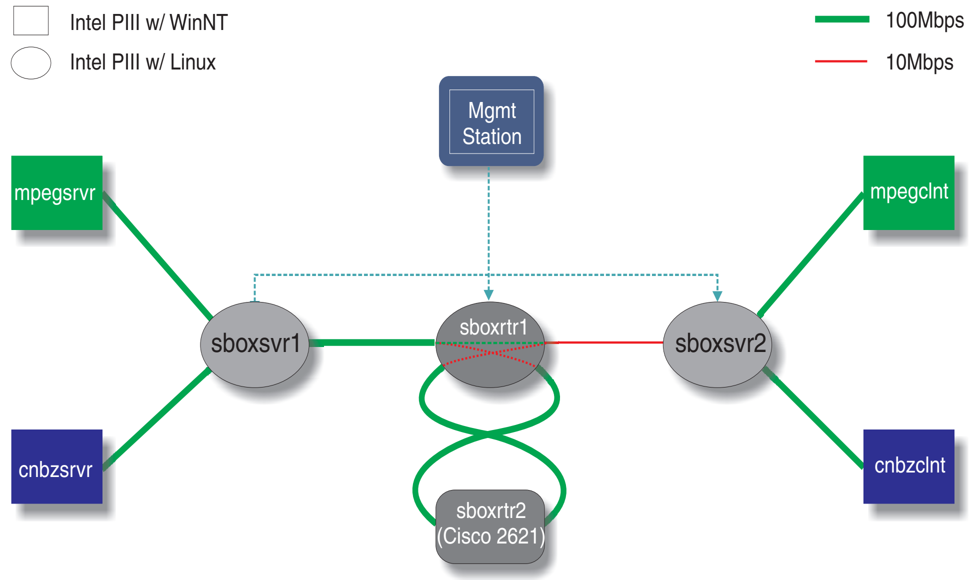

Figure 8: SBoX Prototype Testbed.

ple cineblitz video streams, each about 1.5Mbps, flow from *cnbzsrvr* to *cbnzclnt*. These video streams are the premium QoS traffic and can be selectively protected with guaranteed bandwidth. Premium traffic are label-switched across the network between *sboxsvr1* and *sboxsvr2*. A best-effort UDP traffic, with adjustable bandwidth consumption, flows from *sboxsvr1* to *sboxsvr2* as the background noise to create congestion. The noise traffic is not label-switched; it is routed through regular IP routing.

The noise traffic entering *sboxrtr1* is diverted to the Cisco router and then looped back to *sboxrtr1* before it is routed to its final destination. This is to simulate (1) the Split and Select operation, and (2) the ability of SBoX to inter-operate with existing IP routers without affecting the existing IP traffic. Note that all traffic goes through the bottleneck link, the 10Mbps link between *sboxrtr1* and *sboxsvr2*.

4.1 SBoX Software Components

4.1.1 SBoX Daemon and SBoX Management Client

The SBoX daemon is a user-space daemon that runs on all SBoX servers and routers. SBoX daemon listens on a well known port and responds to configuration, control, and management requests from a management station through the SBoX Management Protocol (SB-MP). SBMP allows the management station to,

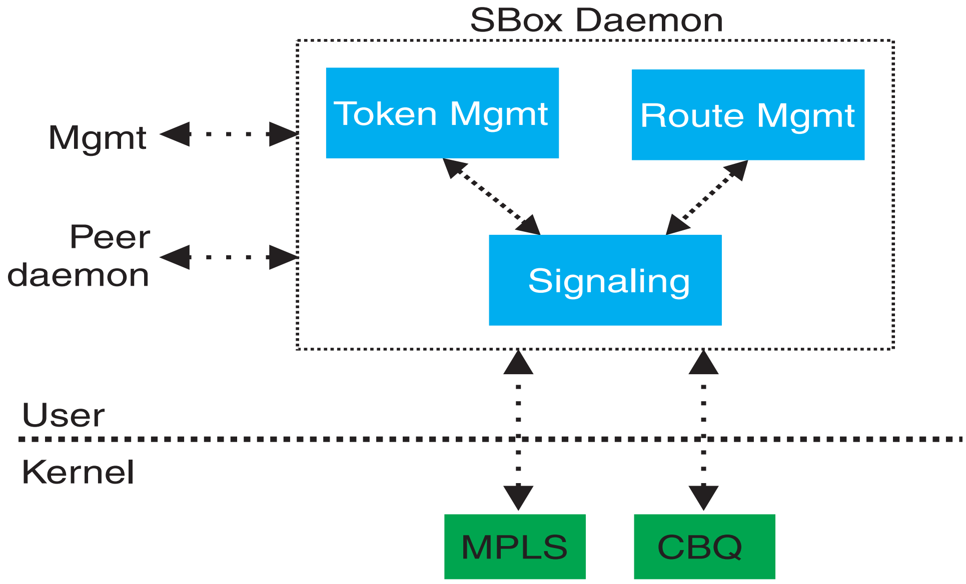

Figure 9: SBoX Software Components.

1. configure token tables on SBoX servers.

2. configure routing tables on SBoX servers and routers.

3. configure MPLS on SBoX servers and routers

4. configure CBQ on SBoX servers and routers

The SBoX daemon also implements the SBoX Reservation Protocol (SBRP) (see Section 2.4) for setting up an end-to-end QoS guaranteed path and distributing the labels. Figure 9 depicts the functional components of an SBoX daemon. The SBoX management client is a command-line driven tool that can be used by a management station to communicate with SBoX daemons. The tool is built on top of the SBox Management API which implements the SBox Management Protocol. The separation of the tool and the API allows more sophisticated management tools such as a GUI front-end to be built easily on top of the API. The following is a list of a few typical API calls.

1. *token_set, token_get, token_del.* On an SBoX server, assign, list, and remove tokens for links that comprise the path from the (source) SBoX server to another (destination) SBoX server.

2. *token_set, token_get, token_del.* On an SBoX server, assign, list, and remove tokens for links that comprise the path from the (source) SBoX server to another (destination) SBoX server.

3. *route_set, route_get, route_del.* On an SBoX server or an SBoX router, install, list, and remove routing information for traffic flowing between two end stations that will traverse the SBoX server or the SBoX router.

4. *resv_rqst.* On an SBoX server, initiates the SBox Reservation Protocol for reserving bandwidth for a macro-flow.

4.1.2 SBoX Client Reservation Web Interface

Clients on end stations such as *mpegclnt* and *cnbzclnt* use a simple web form served from their SBox servers to make bandwidth reservations for the video stream between the video server and the client (note that the reservation can be used for any type of traffic, not just videos). The web form executes the CGI program (written in C) on the SBoX server, which dynamically generates another web form with options of current available bandwidth for different video servers. When the client commits the reservation, the CGI program on the SBoX server initiates SBoX Reservation Protocol by calling the *resv_rqst* API call exported by the SBoX Management API.

4.2 MPLS and CBQ Settings

4.2.1 MPLS Setup

The MPLS support for Linux is version 0.011 of MPLS/LDP implementation by James R. Leu from the University of Wisconsin. Labeling and switching for the premium video traffic is currently statically configured via automated scripts using *mplsadm*. Implementation of dynamic MPLS configuration by the SBoX daemon is in progress. Figure 10 shows the label assignment for the video traffic.

A label-switched path corresponds to a commodity-flow. Note that although there can be different types of micro-flows, such as Telnet, Ftp, Http, etc., flowing between the two end stations that fall into the same macro-flow, i.e., they use the same label. This is the main mechanism of SBoX architecture to deal with scalability issues of flow management.

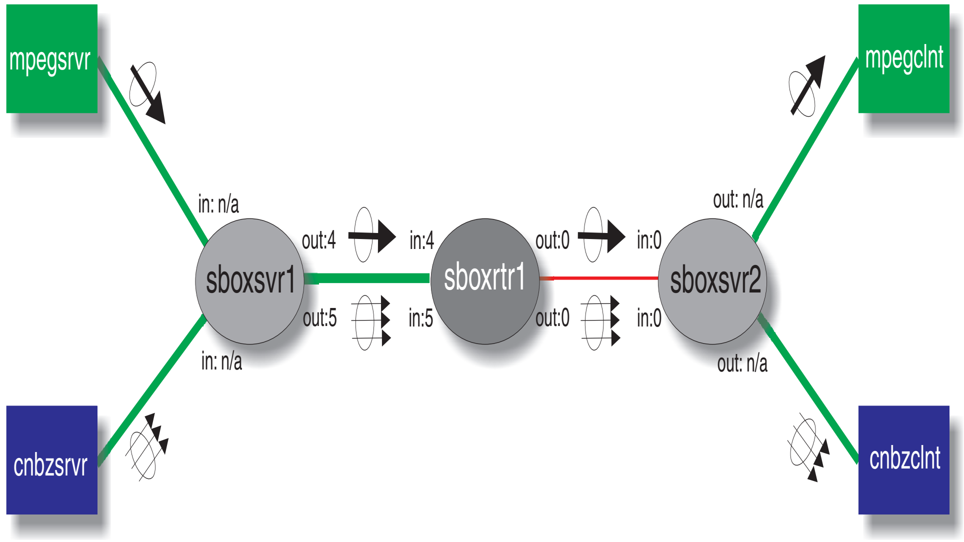

Figure 10: MPLS Setup.

4.2.2 CBQ Setup

The CBQ support for Linux is the kernel implementation on Linux version 2.3.3. Setting up CBQ consists of two parts. The static part that sets up the queuing discipline and root class is done automatically at system startup time via a script using *iproute* tool. The dynamic part that sets up the class and filter for each premium traffic is done by the SBoX daemon through SBoX Reservation Protocol, which also makes use of the *iproute* tool. The Lucent Cineblitz video streaming protocol requires certain amount of bandwidth in the opposite direction of the video stream, i.e., from *cnbzclnt* to *cnbzsrvr*. Since the 10Mbps link between sboxrtr1 and sboxsvr2 is a simplex cross-cable, we created a "fillup" CBQ class of 8Mbps for all the traffic going from *sboxrtr1* to *sboxsvr2*. This effectively makes the 10Mbps link appear to be an 8Mbps link for traffic going from *sboxrtr1* to *sboxsvr2* and leaves 2Mbps for the traffic going in the opposite direction. The CBQ class hierarchy on *sboxrtr1* is shown in Figure 11.

Note that this is the class hierarchy used by *sboxrtr1* inside the network, which shows only two types of traffic, i.e., the Lucent MPEG video stream and the Lucent Cineblitz video stream, regardless of the number of micro-flows within each type of traffic. On the SBox servers at the network boundary, additional classes and filtering rules (such as matching port numbers) are used to control and manage micro-flows within a macro-flow.

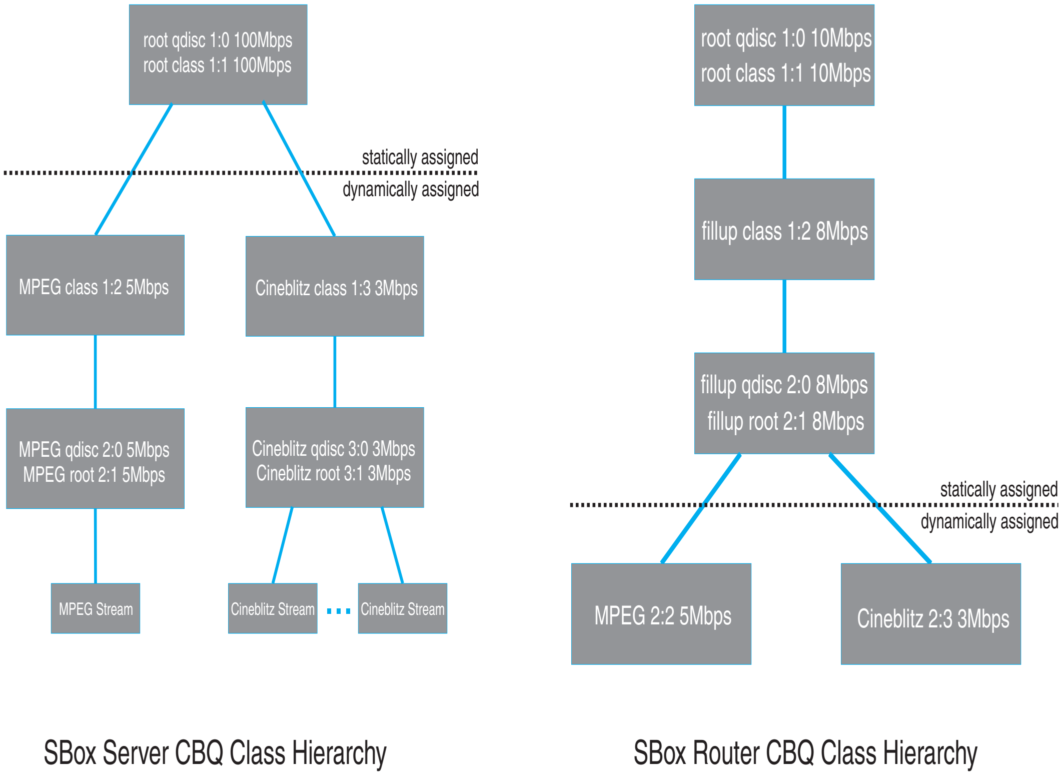

Figure 11: CBQ CLass Hierarchy.

4.3 Experimental Results

Using the testbed implementation we conducted experiments to examine the performance of the SBoX architecture. The SBoX router *sboxrtr1* manages two commodity-flows and it is connected to the *sboxsvr2* with a full-duplex bottleneck link with capacity 10 Mbps as mentioned above.

We observe in Figure 12 that MPLS compliant QoS traffic for both commodities are protected from each other. Furthermore, their total sending rate is bounded by 8Mbps which is the total reserved bandwidth on this bottleneck link. Figure 13 shows that the best-effort traffic takes all the unused reserved bandwidth thus maintaining a high link utilization. As the total reserved traffic amount increases, the best-effort rate reduces to the unreserved portion of the link bandwidth. This behavior verifies that the *movable boundary* scheme works efficiently for providing high utilization. Furthermore it also shows that QoS traffic is protected from the best-effort.

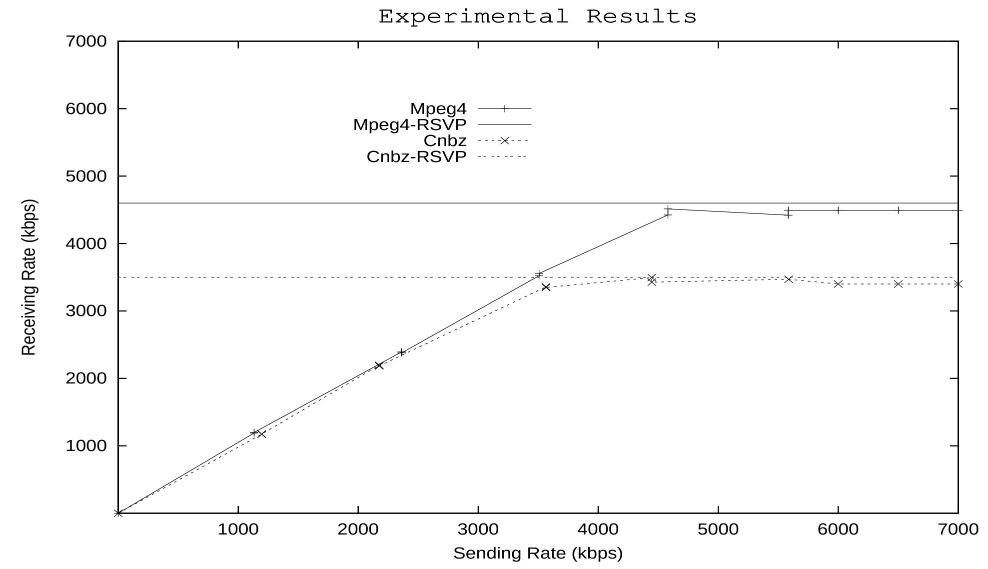

Figure 12: Behavior of reserved traffic

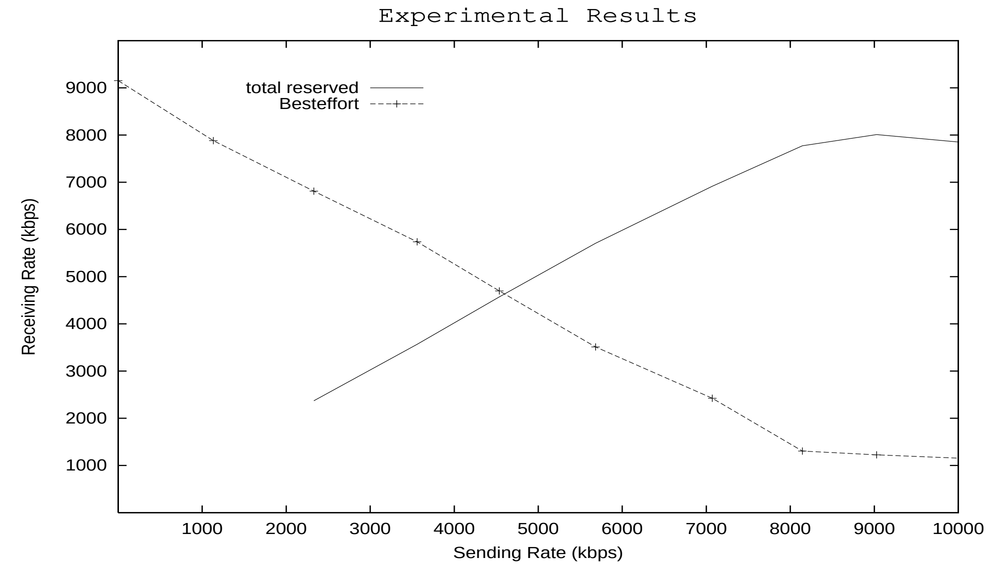

Figure 13: Behavior of best effort and reserved traffic

5 Conclusion

In this work we presented a hybrid architecture to provide deterministic QoS on the IP-based networks. The architecture combines the best features of Diffserv, Intserv, and MPLS proposals. At the core of the proposed architecture, precise provisioning and admission control algorithms are proposed. These algorithms determine a virtual premium network that is used to handle QoS traffic. The complexity of micro-flow management is done at the network boundary by intelligent edge routers called the SBoX servers. The service level agreements offered at the network boundary are supported by add-on label switching routers called the SBoX routers inside the network.

A prototype testbed is implemented and experiments indicate that the proposed architecture delivers its features.

References

[ACL94] G. R. Ash, K. K. Chen, and J-F. Labourdette. Analysis and design of fully shared networks. *Proc. of 14'th International Teletraffic Congress*, pages 1311–1320, 1994.

[Ash95] G. R. Ash. Dynamic network evolution, with examples from at&t's evolving dynamic network. *IEEE Communications Magazine*, 33(7):26–39, 1995.

[BW99] A. Basu and Z. Wang. Fair bandwidth allocation for differentiated services. *in Protocols for High Speed Networks VI, J. Touch and J. Sterbenz ed. Kluwer Academic Publishers*, 1999.

[CFZ94] I. Chlamtac, A. Farago, and T. Zhang. Optimizing the system of virtual paths. *IEEE Trans. on Networking*, 2(6):581–587, Dec. 1994.

[Cla97] D. Clark. Internet cost allocation and pricing. *Internet Economics, L. W. McKnight and J. P. Bailey (eds)*, pages 215–252, 1997.

[CW97] D. Clark and J. Wroclawski. An approach to service allocation in the internet. *Internet Draft, draft-clark-diff-svc-alloc-00.txt*, July 1997.

[Dov91a] R. D. Doverspike. Algorithms for multiplex bundling in a telecommunications network. *Operations Research*, 39:925–944, 1991.

[Dov91b] R. D. Doverspike. A multi-layered model for survivavility in intra-lata transport networks. *Proc. IEEE GLOBCOM'91*, pages 2025–2031, 1991.

[ea98] S. Blake et al. A framework for differentiated services. *draft-ietf-diffserv-framework-0.1.txt*, October 1998.

[ea99] I. Stoica et. al. Per hop behaviours based on dynamic packet states. *internet draft. http://www.cs.cmu.edu/ istoica/DPS/draft.txt*, February 1999.

[For96] ATM Forum. P-nni specification, version 1.0. March 1996.

[IN98] J. Ibanez and K. Nichols. *internet draft : draft-ibanez-diffserv-assured-eval-00.txt*, August 1998.

[Kum74] K. Kummerle. Multiplexer performance for integrated line and packet switched traffic. *Int. Conf. Comput. Commun. Record*, pages 507–515, August 1974.

[Lee95] W. C. Lee. Topology aggregation for hierarchical routing in atm networks. *Comput. Commun. Rev. (USA)*, 25(2):82–92, 1995.

[ML97] H. Michiel and K. Laevens. Teletraffic engineering in a broad-band era. *Proc. of IEEE*, 85(12):2007–2033, December 1997.

[MMR96] D. Mitra, J. A. Morrison, and K. G. Ramakrishnan. Atm network design and optimization: A multirate loss network framework. *Proc. IEEE INFOCOM'96*, pages 994–1003, 1996.

[NBS95] A. Nguyen, N. Bambos, and M.H. Sherif. Adaptive (t1, t2)-multiplexing transmission schemes for voice/data integrated networks. *IEEE Syp. Comp. and Communications*, pages 430–435, 1995.

[NJZ97] K. Nichols, V. Jacobson, and L. Zhang. A two bit differentiated services architecture for the internet. *Internet Draft, draft-nicolas-diff-svc-arch-00.txt*, November 1997.

[OY94] Y. Ofek and M. Yung. The integrated MetaNet architecture: A switch-based multimedia LAN for parallel computing and real-time traffic. *IEEE INFOCOM'94*, 1994.

[RVC98] E. C. Rosen, A. Viswanathan, and R. Callon. Multi protocol label switching. *draft-ietf-mpls-arch-02.txt*, July 1998.

[SPG97] S. Shenker, C. Partridge, and R. Guerin. Specification of guaranteed quality of service. *rfc2211*, September 1997.

[SW97] S. Shenker and J. Wroclawski. General characterization parameters for integrated services network elements. *rfc2215*, September 1997.

[SZ98] I. Stoica and H. Zhang. Lira: An approach for service differentiation in the internet. *Proc. of NOSSDAV'98, Cambridge, England*, 1998.

[Wro97] J. Wroclawski. Specificiation of the controlled-load network element service. *rfc2211*, September 1997.

[Zaf74] P. Zafiropolo. Flexible multiplexing for networks supporting line switched and packet switched data traffic. *Int. Conf. Comput. Commun. Record*, pages 517–523, August 1974.

Contents