IMPLEMENTATION OF A SERVICE BROKER FOR MANAGEMENT OF QOS ENABLED VPNS

Ibrahim Khalil, *Torsten Braun*, *Manuel Günter*
Institute of Computer Science and Applied Mathematics (IAM)
University of Berne
Neubrückstrasse 10, CH-3012 Bern, Switzerland
ibrahim,braun,mguenter@iam.unibe.ch

Abstract

The ability of VPNs to emulate a private wide area network using IP facilities has recently generated tremendous interest in it's wide spread deployment to replace the expensive dedicated private leased lines. Since private networks built on using dedicated lines offer bandwidth and latency guarantees, there is a growing demand for similar guarantees in IP based VPNs. Although with the advent of Differentiated Services IP backbones can now provide various levels of quality of service (QoS) to VPN traffic, as today's network infrastructure continues to grow, the ability to manage increasing network complexity is a crucial factor for QoS enabled VPN solutions. In this paper, we describe the implementation of a Service Broker managing QoS enabled VPN for customers that have Service Level Agreements (SLAs) with their ISPs and allows one such user to specify demands through a WWW interface to establish a VPN with certain QoS between two endpoints. This paper not only shows the implementation details of various components of the Service Broker and connection admission and termination process, but also the pricing mechanism of such outsourced VPNs.

Keywords- Virtual Private Network (VPN), Differentiated Services (DiffServ), Quality of Service (QoS), Bandwidth Broker, Service Broker, Service Level Agreement (SLA), Pricing, Billing.

1 INTRODUCTION

Virtual Private Networks (VPNs) [7], [16], [6] enable secured private communications of distinct closed networks, for example, corporate networks, over a common shared network infrastructure. Rather than using expensive dedicated leased lines, VPNs use worldwide IP network services, including the internet cloud and service provider's IP backbones to connect multiple geographically dispersed sites to each other into a private network. The ability of VPNs to emulate a private wide area network using IP facilities has recently generated tremendous interest in it's wide spread deployment and replace the expensive dedicated private leased lines.

As extensions of the enterprise network a VPN solution must guarantee reliability and Quality of Service by enabling users to define enterprise-wide traffic management policies that actively allocate bandwidth for in-bound and out-bound traffic based on relative merit or importance to all other managed traffic. In fact, there is a growing demand that since private networks built on using dedicated lines offer bandwidth and latency guarantees, similar guarantees be provided in IP based VPNs. While the internet has not been designed to deliver performance guarantees, with the advent of differentiated services [2], [1], IP backbones can now provide various levels of quality of service. The Expedited Forwarding (EF)[9] Per Hop Behaviour (PHB) is the recommended method to build such a Virtual Leased Line Line (VLL) type point-to-point connection for VPN. This is absolutely critical to ensure that the VPN can deliver the myriad number of benefits of this rapidly growing technology.

However, the complexities introduced by VPNs and the requirement to provide QoS have made the job of the ISPs and systems administrators extremely difficult, and as today's network infrastructure continues to grow, the ability to manage increasing complexity is a crucial factor for VPN solutions. But, at the same time, this also opens the possibility for ISPs to sell VPN services to mostly corporate end users. For example, in a typical VPN-DiffServ deployment scenario (Figure 1), an ISP might offer to establish QoS enabled VPN between stub network A and B for a corporate end user who owns those networks and has regional offices there. Outsourcing VPN services not only would ease the job of the corporate customers, but also seems to be the only solution to dynamically construct end-to-end dedicated services over public IP infrastructure since customers have no control over resources in the transit network.

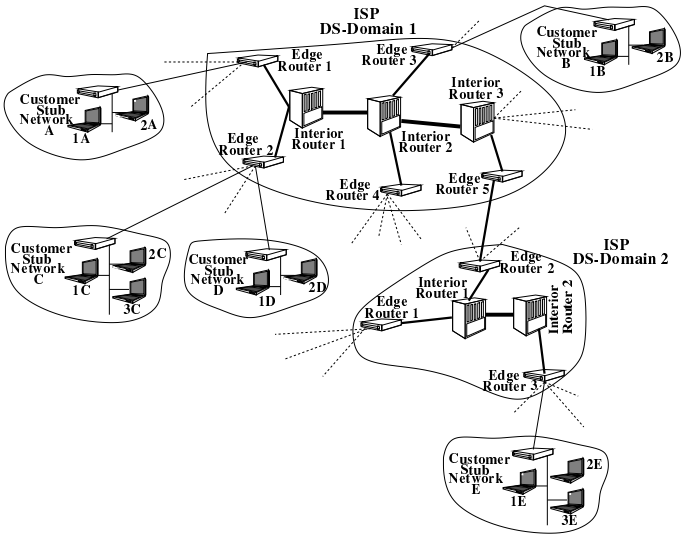


Figure 1: VPN-Diffserv Deployment Scenario

To offer such services, the ISPs will, however, need a management system not only to enable the users to construct services dynamically on demand, but also to provide new charging and billing facilities for the VPN services. In this paper, we describe the implementation of such a management system called Service Broker (SB). As the World Wide Web (WWW) is widely available we provide web based interfaces as front ends where registered users can login, verify themselves and initiate a VPN based on their predefined SLA. This would obviate the need of invoking help from system administrator or ISP and at anytime they can disconnect the VPN service or check their current bills.

The Rest of the paper is organized as follows: section 2 gives a brief architectural overview of the SB needed to construct end-to-end service, section 3 highlights the main security and QoS features necessary for a QoS enabled VPN tunnel, section 4 describes the details of functional components that constitute the SB. Then in section 5, we explain the system flow during VPN connection establishment or termination process and in section 6, we explain the resource admission control at edge devices. Section 7 describes differential tunnel pricing mechanism and section 8 shows the experimental network setup with some examples and performance results. Finally, in section 9 we conclude our paper.

2 A BROKER HIERARCHY FOR CONFIGURABLE SERVICES

Differentiated Services and VPN security can be seen as value-added network services. Such services allow the service providers to produce new revenues beyond pure connectivity services. However, the new services increase the complexity of the network management significantly. [18], [24] proposed so-called bandwidth brokers to address the dynamic configuration and management of DiffServ. The idea can be extended to network services in general, including VPN services and service bundles such as QoS-VPNs. This leads to a framework called the broker hierarchy.

The hierarchy is structured in functional components. We made the distinction between intra-domain and inter-domain service tasks. Furthermore we distinguish between stand-alone (orthogonal) services and composed services. Finally we added a layer to hide the diversity of network equipments. This results in a four-layered broker hierarchy as depicted in Figure 2. The configuration daemons (CD) hide the heterogeneity of the underlying network equipment. The internal service broker (ISB) manages the provider controlled network resources for a service, and coordinates them. For example, referring to Figure 1, if an user who has a SLA with the ISP maintaining DS-Domain 1 wants to establish a tunnel between host 1A in stub network A and host 1B in stub network B, ISB of that domain can handle that request to establish the desired connection. The external service broker (ESB), however, handles the negotiation between brokers of peer ISPs across the trust border. The necessity of ESB is obvious when the customer in the previous example wants to setup a tunnel between host 1A and host 1E in stub network E. Since Stub network E resides by the perimeter of another domain (DS-Domain 2), ESB of domain 1 needs to negotiate with the ESB of domain 2. The result of a negotiation is an inter-ISP service level agreement (SLA), which describes the collaboration between the two provider networks . A broker signalling protocol is necessary to set up an unbroken chain of SLAs between all involved ISPs in order to set up an Internet wide service.

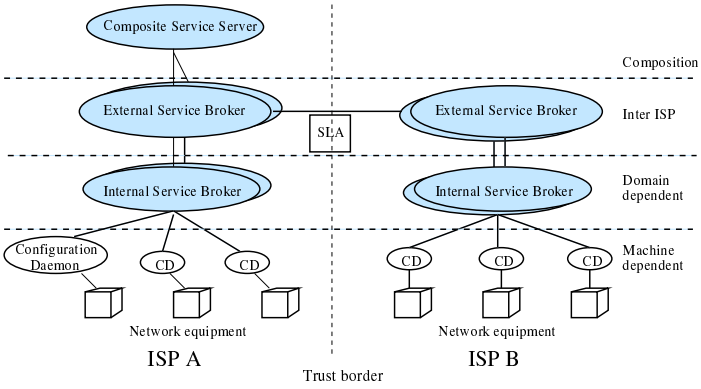


Figure 2: The broker hierarchy.

The broker hierarchy and its rationale is described in more detail in [8]. The work presented in this paper is an instance of this broker hierarchy framework. We focus on the implementation of a DiffServ VPN service using the IPSec [10] protocol. The synergy between DiffServ and IPSec [21] and a single administrative domain scenario lead to a stand-alone implementation using only the ISB and the CD layer. In this paper whenever we refer to Service Broker (SB), we essentially mean ISB.

3 REQUIREMENTS FOR A QOS ENABLED VPN

3.1 Data Transport Security

The most important feature of a Virtual Private Network service is its privacy ensuring mechanism, that protects VPN traffic from the underlying public network. For Internet VPNs the most powerful mechanisms are tunneling and encryption. Tunnelling (also called encapsulation) is a method of wrapping a packet in a new one thus providing it with a new header. The whole original packet becomes the payload of the new one as shown in Figure 3 and 4. When encryption is applied to a packet, it conceals the content of that packet. Tunnelling requires intermediate processing of the original packet on its route. The destination specified in the outer header retrieves the original packet and sends it to the ultimate destination. Although the encapsulation and encryption may degrade the performance to some extent, the processing overhead is compensated by extra security. In general, the following features are provided by a VPN tunnel.

- *Data confidentiality* - The sender can conceal cleartext by encrypting them before transmitting across a network.
- *Data integrity* - The receiver can verify that the data has not been altered during transmission, either deliberately or due to random errors.
- *Data Origin Authentication* - The receiver can authenticate that the data was originated from the sender. This service, however, is dependent upon the data integrity service.

In the Internet the security requirements of one user might vary from others under various circumstances depending on his needs.

Therefore, to facilitate these features we have used various tunneling options of IPSec [10] described below and presented them as user selectable options (Figure 6) so as to allow a customer to choose the one that suits him best.

- **IPSec AH in Tunnel Mode**: The Authentication Header (AH) is used to provide integrity and authentication to IP datagrams. When packets go through an AH type tunnel an AH is embedded (Figure 3(b)) in the data to be protected (a full IP datagram). This mode of operation greatly reduces the chances of successful denial of service attacks, which aim to block the communication of a host or gateway by flooding it with bogus packets. The AH protocol allows for the use of various authentication algorithms, most notably, MD5 (a hash message authentication code - HMAC variant) and SHA (HMAC variant). Both MD5 and SHA are widely used. HMAC is a keyed hash variant used to authenticate data.

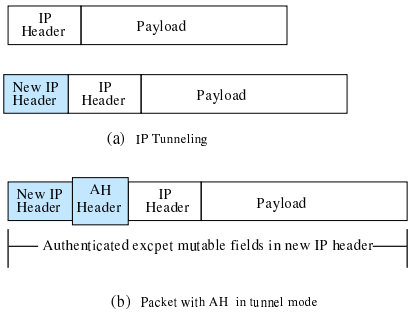


Figure 3: IP Tunneling: (a) basic IP tunneling, (b) AH tunnelled packet, (c) ESP tunnelled packet, (d) combined tunnel mode

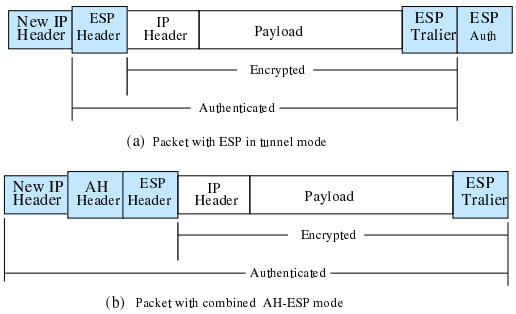


Figure 4: IP Tunneling: (a) ESP tunnelled packet, (b) combined tunnel mode

- **IPSec ESP in Tunnel Mode**: The Encapsulation Security Payload (ESP) is used to provide integrity check, authentication and encryption to IP datagrams (Figure 4(a)). Although both authentication and encryption are optional, at least one of them is always selected. If both of them are selected, then the receiver first authenticates the packet and only if this step was successful proceeds with decryption. This mode of operation reduces the vulnerability to denial of service attacks.

- **Combined Tunnel Mode**: Even though most tunnel gateways are required to support only an AH tunnel or ESP tunnel, sometimes it is desirable to have tunnels between gateways that combine both IPSec protocols. The result is that we have an outer IP header followed by the IPSec headers in the order set by the tunnel policy, then the original IP packet, as it is shown in Figure 4(b).

3.2 Guaranteed QoS

It is generally believed [7] that most VPN users would demand Virtual Leased Line (VLL) type low loss point-to-point connection. This is achieved by a simple model [2], [3] where VPN traffic entering a network is classified as EF, possibly conditioned at the boundaries of the network, and assigned to different behaviour aggregates. Each behaviour aggregate is identified by a single DS codepoint. In the interior of the network, with the help of DS codepoint- PHB mapping [17], [3], this quantitative VPN traffic can be allocated a certain amount of node resources. For example, referring to Figure 1 if an user wants to establish a 1 Mbps connection for a source 2A in stub network A and remote host 2B in stub network B, we will need to classify packets at the edge router 1 (traffic flow direction is from edge 1 towards edge 3), police at inbound and possibly shape at outbound of this router and also allocate 1 Mbps capacity in the interior router 1 and 2's outbound interfaces with scheduling mechanisms like Class Based Queueing (CBQ) [5] or it's variants [14], [22], [20].

4 STRUCTURE OF SB: COMPONENTS OF THE SYSTEM

The SB, which is the heart of our VPN management system, takes the role of a QoS manager to optimally configure network resources and adaptively decides based on user preferences and resource availability. These decisions could take place with minimum user intervention with respect to specifying the user's requirements. As the underlying network may provide different classes of services to satisfy various VPN customers, by identifying the generic functionality provided by any resource, we present our SB (Figure 5) with a standard interface (Figure 6) to the network resource.

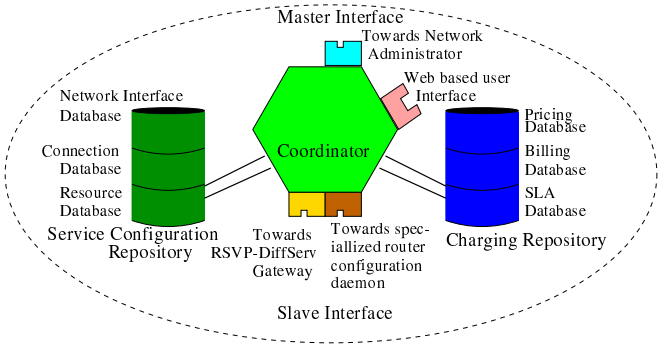


Figure 5: VPN components

Our SB interacts with specialized configuration daemons (CD) when a certain user request arrives to setup a tunnel and the SB has to decide whether it can allocate enough resources to meet the demand of that tunnel. The basic operation of our system is as follows: based on request parameters provided by the user, the SB first contacts a SLA database to check the validity of the user and it's request parameters. It then checks with the connection database whether a similar requested connection already exists or

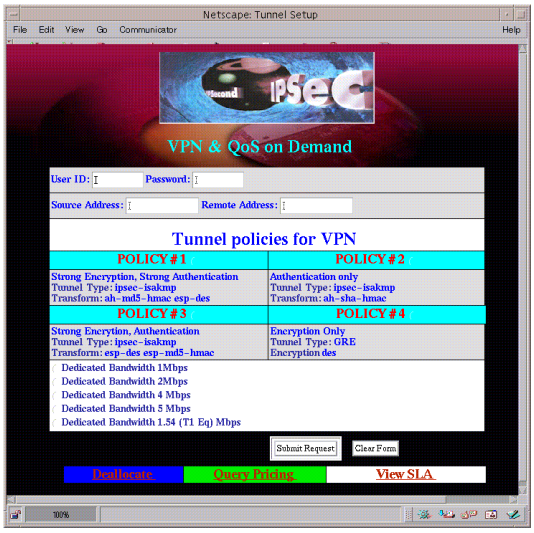


Figure 6: VPN Web interface

not. If this is not the case, the SB looks at its resource database to identify if the tunnel can be established. A positive answer would then lead to a tunnel establishment by the CD. When a user disconnects the VPN tunnel, the SB releases resources and invokes the pricing database to calculate the pricing for that tunnel. In the next few subsections we will briefly describe these components and their role before we move to the detailed description of system flows in section 5.

4.1 SLA Database

The SLA database does not only contain the user's identification but also specifies the maximum amount and type of traffic he/she can send and/or receive for a tunnel. As we are concerned about closed user groups, a SLA also contains the boundary of a valid VPN area. Referring to Figure 1 where stub networks A, B and C might belong to the same organization located at different remote locations, one can easily see that they form a mesh environment and any site may want to establish with the other under the same contract. Therefore, this boundary defines the perimeter of the VPN area and are entered in this database as source and remote stub addressess. User authentication process prohibits malicious users to setup unauthorized tunnel and access network resources illegally. The SLA, however, allows users to add new VPN areas to his old contracted list of valid VPN areas. It contains the following tuple:

<User ID, Password, Maximum BW in Mbps, Source Stub Address, Remote Stub Address>

4.2 Resource Database

The resource database contains resources available between any two edge routers. This means that this database has resource information of all the routers in a certain domain. In our implementation we keep records of pre-computed tunnels with Tunnel IDs. For each tunnel, however, we also need to know it's ingress router's ip address, tunnel source address (which might be the same as ingress router's ip address), egress router's ip address, tunnel destination address (this might as well be same as egress router's ip address) and the capacity of the tunnel in Mbps. Also, we need to keep track of the status of the tunnel in terms of availability. Therefore, the tuples are:

<tunnel id, ingress router, tunnel source addr, egress router, tunnel destination addr, bandwidth, status>

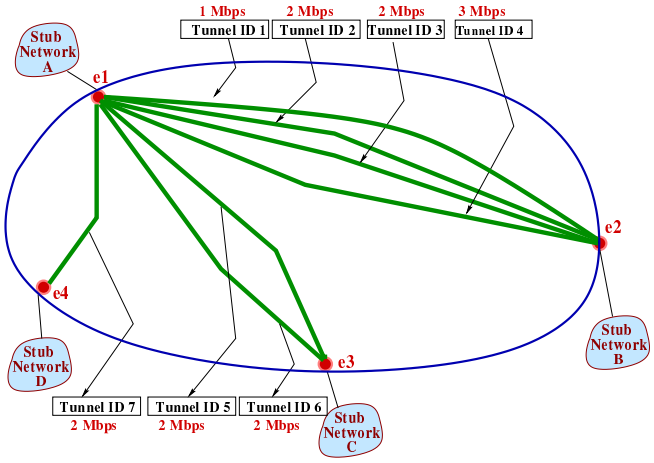


Figure 7: Mapping of resources to various tunnels

However, it should be clarified that a tunnel might originate from an ingress router to several possible egress routers. Referring to Figure 7, users residing in the stub network A might want to establish tunnels between router e1 and e2, or between e1 and e3, or between e1 and e4 to communicate with other stub networks. Assume that ISP has decided to allocate a maximum 10 Mbps capacity to traffic stemming from e1 and destined towards other edge points. The ISP might, however, allow one 1 Mbps, two 2 Mbps tunnels, one 3 Mbps tunnel to be created between e1 and e2 and also allow two 2 Mbps tunnels from e1 to e3 and only one 2 Mbps tunnel from e1 to e4. This would result in a map as shown in table 1. In the table while ingress and egress router addresses are necessary for identifying the edge routers, the tunnel source and destination addresses are needed to create tunnels. Ingress (or egress) router and it's corresponding tunnel source address (or dest. addr for egress) might be same if the same address is used in both cases.

Tunnel ID	Ingress Router	Tunnel Source Address	Egress Router	Tunnel Dest. Address	Bandwidth in Mbps	Status
1	e1	e1	e2	e2	1	1
2	e1	e1	e2	e2	2	1
3	e1	e1	e2	e2	2	1
4	e1	e1	e2	e2	3	1
5	e1	e1	e3	e3	2	1
6	e1	e1	e3	e3	2	1
7	e1	e1	e4	e4	2	1

Table 1: Resource Table for the Network of Figure 7

It is clear from this mapping that if all the tunnels as mapped in the table 1 are active simultaneously, then router e1 would need to support 14 Mbps. Since we have only 10 Mbps for this router we need to keep track of each router's capacity and perform an admission control as explained in section 6.

4.3 Connection Database

The connection database contains a list of currently active VPN connections. Here a connection always mean a VPN tunnel. It has various functions: (i) when a new request arrives for connection

or termination, the SB can check if that connection already exists or not and then make it's decision, (ii) it indicates how much resources have been consumed by VPN users, (iii) and provides a record to pricing mechanism. It's tuples are:

<user id, source address, destination address, bandwidth, tunnel id, activation time>

4.4 Interface Database

The interface database contains necessary records of edge routers that are used as tunnel end-points for the outsourced VPN model. In such a model since some customer stub networks are connected to the ISP edge router we need to specify which stub networks are connected to a particular edge router. Also, an edge router might have one or more inbound and outbound interfaces which also need to specified for each stub network that is actually connected to a particular inbound interface of a router. This is important becasue normally at the inbound interface tunnels are policed on individual basis and at the outbound they are shaped on aggregated basis. At the same time, outbound interfaces are also used as the tunnel endpoints. Finally, a tunnel map to which all defined tunnels are attached is also part of the record in this database that is activated at the outbound interface of the router. The tuples are :

< stub network, edge router, generic router name, inbound interface, outbound interface, tunnel map name>

4.5 Pricing and Billing Database

The pricing database contains pricing information of various tunnels. It's only interaction with the SB is at the time when a connection (tunnel) is terminated and the SB needs to know the price of that by making a query to it. The billing database contains details of terminated connections and their computed price. For details see section 7.

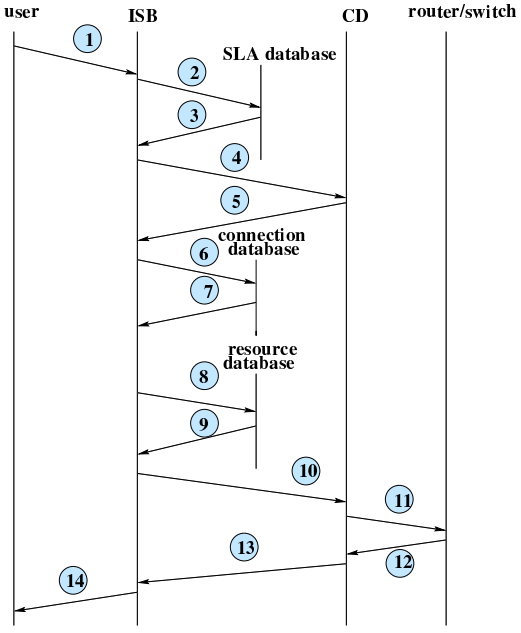
5 DESCRIPTION OF COMPLETE SYSTEM FLOWS

In this section we will describe how a connection is established or torn down, how various components interact with the SB, and under which circumstances a new connection request or tear down request get refused.

5.1 The Successful Connection Establishment

Figure 8 shows all the communications involved in setting up a VPN connection between two stub networks or simply between an originating host and the remote host. We will describe the operational details by referring to the communication marked on Figure 8. Considering each communication in turn :

- 1) A user sends a connection request message to the SB for a new connection request from the WEB or via other signalling mechanism such as RSVP. The SB is in charge for determining whether the connection should be allowed or refused. It achieves this by communicating with each of the components in turn. The request contains user id, user password, source and remote stub addressess, amount of bandwidth and encryption/encapsulation method.
- 2,3) The SB contacts the SLA database that is responsible for validating the user and his request. If the user is identified correctly, his source and remote address conforms the contract, and also the bandwidth requested is less than or equal to the agreed traffic contract, it sends a positive response.
- 4,5) The SB contacts the configuration daemon to check it's status. The status can be busy, available, or down. Only in the case of availability the user request can be processed further.
- 6,7) The SB contacts the connection database to check the existence of an exactly similar tunnel. This is because for a source and destination pair only one tunnel can remain active.
- 8,9) The SB asks the resource database to allocate a tunnel of a certain bandwidth. The resource database responds to the SB and either allocates the resource or denies based on resource availability.
- 10) The SB allocates the requested resource and tells the configuration daemon to create appropriate configuration scripts. In the meantime the resource and the connection database update their records. The new connection request data is appended to the connection database and the tunnel that has just been allocated from the resource database is marked as used.
- 11,12) The CD puts a busy signal on itself and creates the configuration scripts. It then sends configuration scripts to the routers. The routers send signals to the CD.
- 13,14) The CD removes the busy signal from itself and sends an acknowledgement to the SB which sends a notification to the user.



(a)

Figure 8: Successful Connection Setup

5.2 Successful Connection Termination

Now referring to Figure 9 for successful connection termination :

- 1,2,3,4,5,6,7) These steps are similar to the steps mentioned for connection setup. However, in step 2 only user id and password are checked. We also need to see if the daemon is busy or not, and also that the requested connection exists in the connection database. In summary, once a termination request arrives the system needs to make sure that the tunnel exists and was created by the same user who has sent the termination request.
- 8) If the connection is found in the connection database the SB talks to the CD to create an appropriate configuration script. In the meantime the connection record is deleted from the connection database and the resource database updates its records by making the same tunnel available which has just been deleted.
- 9,10,11) CD creates and sends the configuration script to the router. The router sends a signal to the CD which then confirms the SB about the configuration.
- 12,13,14) Before SB finally forwards this positive acknowledgement to the user it invokes the pricing database and calculates the appropriate pricing using the method described in section 7 and stores the necessary information of the terminated connection and computed price in the billing database. Once that is done user receives the acknowledgement via the web interface.

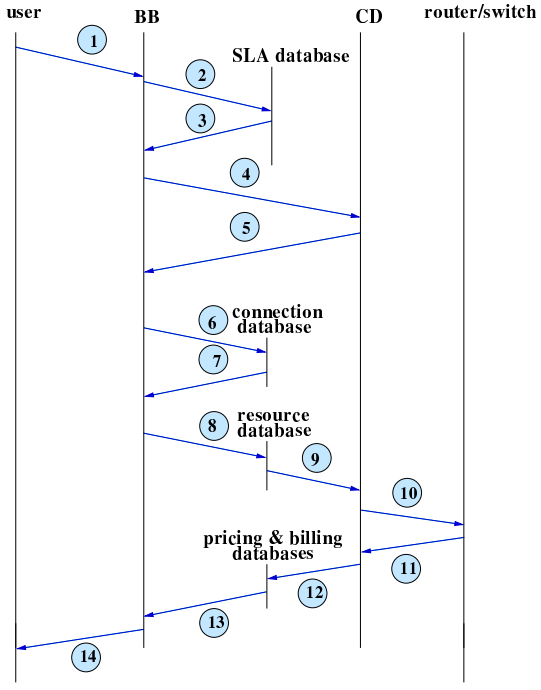


Figure 9: Successful Connection Termination

5.3 The Connection Rejection and Failure in Termination

A connection request is rejected if (i) the SLA profile doesn't match (case D-1) , (ii) the daemon is found busy (case D-2), (iii) the connection already exists (case D-3) or (iv) not enough resources are available (case D-4). The various stages where a connection creation process might get refused are shown in Figure 10. We will briefly describe them in the following:

- case (D-1): user id, password might be wrong, VPN areas for which user wants to establish tunnel might be invalid, or the bandwidth requested might be higher than the agreed one, and in such a case the user will not be granted a connection.
- case (D-2): even if the request parameters are valid the CD might be found busy and hence, the connection will not be possible.The system can, however, be tailored for automatic retry as desired by an user.
- case (D-3): if the request passes the above two stages successfully, it might be found that a connection already exists in the connection database. In such a case the request will be refused.

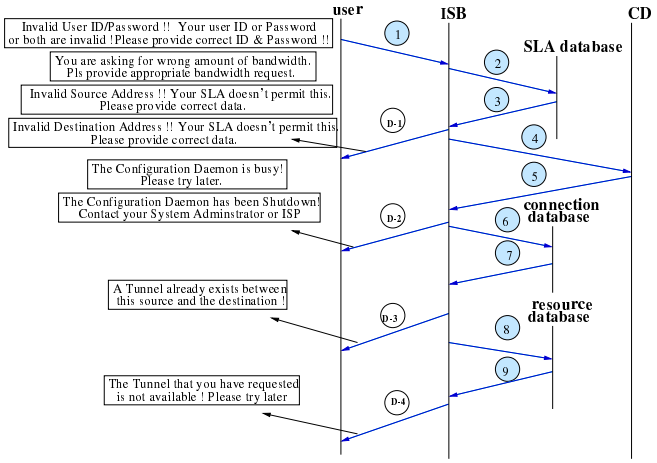


Figure 10: Denial of Connection Setup

- case (D-4): If the above cases don't happen during a tunnel creation process then the SB asks the resource database to grant the request user has asked for. If the resource database cannot meet that demand it sends a signal to the SB about resource unavailability, and the SB forwards that message to the user.

Other than case (D-1) an (D-2) whose actions are quite obvious, the connection termination request (Figure11) might be rejected (case D-3) if no connection entry is found in the connection database for deletion of a request or if the tunnel belongs to the someone else other than the requester.

6 RESOURCE ADMISSION CONTROL

We have earlier discussed the mapping of an interface's resources towards various possible destinations. Since a 10 Mbps capacity of an interface might be mapped to several destinations, the summation of those mappings may naturally be higher than 10 Mbps. However, one should not over allocate resources and restrict the number of tunnels originating from an interface as users might

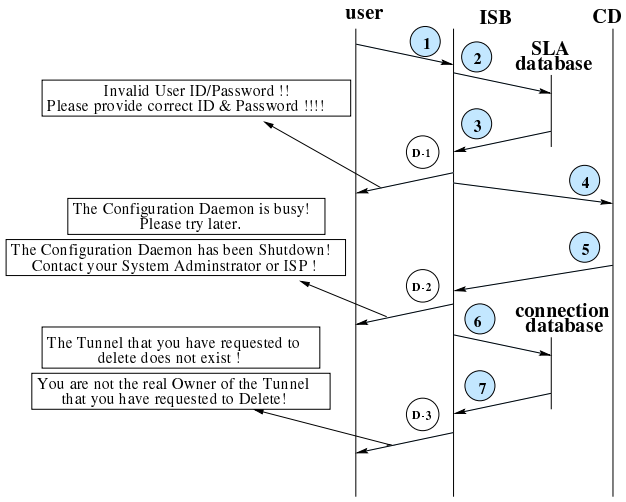


Figure 11: Failure in Connection Termination

observe poor performance otherwise. Therefore, for any ingress interface if C_{TOTAL} is the total capacity reserved for VPN traffic, $C_{allocated}$ is the bandwidth that is already allocated to existing tunnels, and $C_{request}$ is the requested capacity of the new incoming connection, then C_{TOTAL} should be more than or equal to $(C_{allocated} + C_{request})$.

7 VPN PRICING

Although the current flat rate pricing with uniform best effort data transport service is simple and attractive, it has many defects. It provides a single level of service quality, and doesn't allow users to select what is best for their needs. To many, this leads to misallocation of resources. To deal with this, there are proposals to regulate the usage by imposing fees based on the amount of data actually sent. This, however, is fundamentally flawed as usage based fees would impose usage costs on the user whether the network is congested or not and might even collapse the whole revenue model [4].

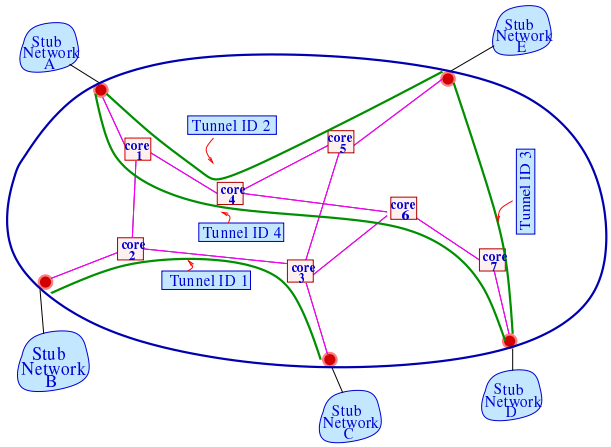
7.1 VPN Pricing Model

With our VPN Service model defined in earlier sections, Internet Service Providers are going to provide a variety of services through multiple service classes (e.g. 1 Mbps, 2 Mbps or 3 Mbps dedicated bandwidth) where each service class will provide a different performance. The objective is to allow users to select among a choice of services, so that users who wish to use more resources can pay accordingly.

In our implementation we have provided a method to compute the price of a VPN which considers the reserved bandwidth of the tunnel and the duration it was used. However, that price might change over the duration of an active period, e.g. if we have special tariffs for the day and the night. This actually reflects that price changes as the load changes, i.e. we consider price to be a function of resource and load. A 2 Mbps tunnel that is charged 4 cents per minute during peak period would not be charged the same during off load period. Also a 2 Mbps tunnel that is spanned over several core routers might be priced higher than a tunnel of same capacity but spanning over few routers. In reality, most of the costs will depend on the transmission lines and some of the expensive lines might be consisting of fewer routers than the less expensive lines of same capacity. In such cases tunnel price can be set accordingly. Based on this idea we propose that pricing for QoS enabled VPN tunnel should be calculated based on it's network resource reservation and the load during the time tunnel is active. Therefore,

$$Price = f(Resource, Load)$$

This model works much like the telephone system when one pays more for long distance call than a local call and price changes at different times. As understanding of pricing by the users is crucial for the success of managed VPN services, we believe this model would be attractive not only to the users but also will simplify the billing process for ISPs.



Price Per Minute in cents								
Tunnel ID	00:00 - 01:00	01:00 - 02:00	02:00 - 03:00	06:00 - 07:00	07:00 - 08:00	08:00 - 09:00	09:00 - 10:00	23:00 - 24:00
1	1	1	1	1	1	2	4	1
2	1	1	1	1	2	4	6	1
3	1	1	1	1	1	2	4	1
4	1	1	1	1	5	7	9	1

Figure 12: Differential Tunnel Pricing of an Example Network

7.2 Differential Tunnel Pricing

As in our system resource is a single quantitative item usually expressed as 1 Mbps, 2 Mbps etc. it well understood how one can price a pipe of various hop distances. For example, in Figure 12 both tunnel 1 and 2 are of 1 Mbps but tunnel 2 is priced higher at certain times since it spans over 3 core routers, and therefore, consumes more resources than tunnel 1. For the same reason tunnel 4 might be priced higher than tunnel 3 although both are of 2 Mbps. If tunnel 3 is highly congested line (or highly demanded) or it's installation cost was higher than normal cases (e.g. for intercontinental links), the ISP might well set the price 3 higher than tunnel 4 even thought it spans over less routers. Therefore, setting up the pricing matrix depends entirely on ISP's experience and business practice. However, having said that, we still need to incorporate load in the pricing formula. Normally, load changes with time and we can, therefore, define a differential tunnel pricing matrix as shown in Figure 12 where price changes with time. If $P_i(t)$ denotes the price of tunnel i at time t and T_{in} and T_{out} denotes the tunnel creation time and termination time respectively, then P_{TOT}, the total computed price of the tunnel for the duration $(T_{out} - T_{in})$, can be expressed as:

$$P_{TOT} = \sum_{T_{in}}^{T_{out}} P_i(t)$$

In our implementation we have defined 24 time zones, i.e. the price for a tunnel changes every hour. It is, however, possible to have more or less than 24 zones. All that is needed to be done in such a case is to define a similar pricing matrix based on the number of desired time zones. Again, how ISP should decide about the number of time zones depends ISP's experience with load fluctuations from past and also on it's policy. For example, if the load is frequently changing then higher number of time zones might be appropriate to reflect the high fluctuations of load on the price.

Based on our pricing model we will now show how one can calculate the price of a tunnel for certain duration of time. Let us assume that T_{in} and T_{out} have formats like $H1 : m1$ and $H2 : m2$ respectively where $H1$ and $H2$ represent hour portions of time while $m1$ and $m2$ represent the minute portions. If $p_i(h)$ denotes the price of tunnel $i = 1, 2, 3, 4...N$ (if we have N tunnels) for time zone $h = 0, 1, 2, 3, ...23$ etc, then the total price of a VPN tunnel can be defined as:

$$P_{TOT} = \begin{cases} p_i(h).[m2 - m1] & \text{if } M = 0 \\ p_i(h)[60 - m1] + p_i(h+1).m2 & \text{if } M = 1 \\ p_i(h)[60 - m1] + \sum_{h=h+1}^{h+M-1} p_i(h).60 & \\ \quad + p_i(h+M).m2 & \text{if } M \geq 2 \end{cases}$$

where $M = H2 - H1$. We will now provide a few examples using the formulas presented above and differential pricing matrix shown in Figure 12. If a certain user has used a tunnel of 1 Mbps from 6:10 a.m. to 6:20 a.m. between stub network B and C and that tunnel happens to have the tunnel ID 1, then the price can be calculated as: $p_1(6)$ *(20-10)=1*10= 10 cents. If the same tunnel is active from 6:10 to 7:20 then the price would be $p_1(6)$ *(60-10) + $p_1(7)$* 20 =1*50 + 1*20 =70 cents. Again, if we setup a tunnel having ID 4 between stub network A and D from 6:30 to 9:20, then we can calculate the price as $p_4(6)$ *(60-10) + $p_4(7)$* 60 + $p_4(8)$* 60 + $p_4(9)$* 20 = 1* 50 + 5 *60 + 7* 60 + 9*20 =950 cents.

7.3 Billing

In section 5.2, referring to Figure 9, we explained that when a VPN connection is terminated, the SB invokes the pricing database to compute the price and send the record to the customer's billing database. In brief, the complete charging system works as follows: if a new VPN request is accepted then that connection along with the login time is recorded in the connection database. After a certain period when the user disconnects his VPN tunnel that entry is deleted from the connection database. The SB then looks up the rate for the deleted tunnel from the pricing database, computes it's price using the formulas in above section 7.2 for the duration of activation and then adds deleted connection record along with logout time stamp and the computed price to the billing database. The user can also, at any time, ask the system to query the most recent billing.

In section 8.2 we will see that user catispp establishes a 1 Mbps connection between source 172.18.0.100 and destination 172.17.0.103 at time 6:17:20. Now, if the connection is terminated at 9:23:14 then SB deletes the connection entry from connection database as shown in table 6 and invokes the pricing database of table 4 and calculates the price using the method that we have earlier described. Once that is done, the computed total price, which is 606.5 cents, termination time 6:17:20, and deleted connection entry are added to customer's billing database as shown in table 7.

8 EXPERIMENTAL SETUP: EXAMPLES AND RESULTS

To test our implementation of the broker system and it's capabilities to setup VPN tunnels and allocate QoS to the established tunnels we ran some experiments in our campus network between two private subnets and also over the public SWITCH [23] network between Bern and Geneva. The topology we used is shown in Figure 13.

We have three private networks with closed user group properties. The machine 172.18.0.100 is a Webserver containing some pre-recorded MPEG-I streams and is located in a Private network 172.18.0.0 at the University of Geneva. Both 172.20.0.100 and 172.17.0.103 (also a webserver) are in two different networks at the University of Berne. All the machines are connected to routers having public IP addresses. Since machines with private addresses are not able to talk to each other over public network unless they communicate over tunnels, this setup is useful to demonstrate that tunnels are really created when a registered user sends a requests to the Broker via the web interface. The Broker runs on a machine 130.92.66.22 and communicates with the routers interfaces that have public IP addresses.

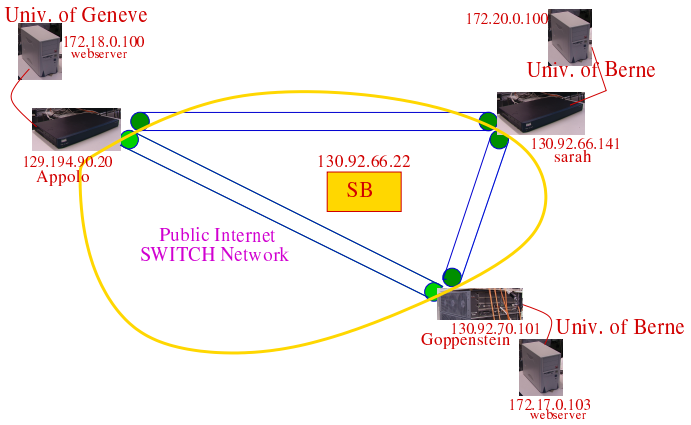


Figure 13: Experimental Setup of VPN

The routers 129.194.90.20 and 130.92.66.141 are both Cisco 26xx while 130.92.70.101 is a 7206 router. All the routers are IPsec and QoS capable. The hop distances between 129.194.90.20 and 130.92.70.101 is 10, between 129.194.90.20 and 130.92.66.141 is 9 and between 130.92.66.141 and 130.92.70.101 is 2. Ideally, to create Virtual Leased Line (VLL) type service the interior routers between the edges should also be DiffServ capable and be able to protect the traffic that are marked as EF at the edges by using CBQ or WFQ. However, since we didn't have control over all these routers, we restricted the experiments to tunnel creation and policing/shaping at the edge routers over which we have complete control.

Tunnel ID	Ingress Router	Tunnel Source Address	Egress Router	Tunnel Destination Address	Bandwidth in Mbps	Status
140	130.92.70.101	130.92.70.101	129.194.90.20	129.194.90.20	1	1
141	130.92.70.101	130.92.70.101	129.194.90.20	129.194.90.20	2	1
142	130.92.70.101	130.92.70.101	130.92.66.141	130.92.66.141	1	1
143	130.92.70.101	130.92.70.101	130.92.66.141	130.92.66.141	2	2
144	130.92.66.141	130.92.66.141	129.194.90.20	129.194.90.20	1	2
145	130.92.66.141	130.92.66.141	129.194.90.20	129.194.90.20	2	2

Table 2: Resource Table for the test Network

Stub Network	Edge Router	Generic Name of Router	Inbound Interface	Outbound Interface	Tunnel Map Name
172.17.0.0	130.92.70.101	Goppenstein	FastEthernet 1/0	FastEthernet0/0	cati-tunnel
172.20.0.0	130.92.66.141	sarah	FastEthernet0/1	FastEthernet0/0	cati-tunnel
172.18.0.0	129.194.90.20	Appolo	FastEthernet0/1	FastEthernet0/0	genbern

Table 3: Interface Database for the test Network

Although we had around 1.5 Mbps between the router in Geneva and routers in Bern during the daytime, we had ample of capacity between the routers in Bern.

8.1 Setting up Databases for the Test Network

To give a clear idea how the Service Broker works we will show how we setup the various databases that we discussed in section 4. Initially, we only need to setup the interface, resource, SLA and pricing databases which are invoked during establishment or termination of a QoS VPN tunnel. Tables 2, 3, 5, 4 show the partial database entries for the test network.

8.2 Examples of Connection Setup

Consider an example when user catispp wants to establish a 1Mbps a tunnel between Host 172.18.0.100 and 172.17.0.103 which are in two different private networks. Once he submits his request via the WWW interface the SB checks SLA validity, daemon status and connection database. While checking the SLA database (Table 5) it finds that catispp is a valid user (and password is correct), source and remote stub addresses are valid, and requested rate (1 Mbps) is less than the maximum contracted rate (4 Mbps). Assume that configuration daemon is on (i.e status is 1).

As no such connection (i.e. between source 172.18.0.100 and destination 172.17.0.103) record exists in the connection database prior to this request arrival, the SB now searches the resource database for the availability of a 1 Mbps tunnel. Since the private networks 172.18.0.0 and 172.17.0.0 are connected to routers Appolo (ip address 129.194.90.20) and Goppenstein (ip address 130.92.70.101) respectively, the Broker actually looks for a 1 Mbps tunnel between these two routers. It turns out that tunnel 140 is the exact match and is also available. Therefore, appropriate configuration scripts are created and loaded to the router to establish the reqested tunnel. For details see [11].

Tunnel ID	00:00 - 1:00	1:00- 2:00	. . .	6:00 - 7:00	7:00 - 8:00	8:00- 9:00	9:00 - 10:00	. . .	23:00 - 24:00
140	1	1	. . .	1.5	3.5	4	4	. . .	1
141	1	1	. . .	2	4.5	5.5	5.5	. . .	1
142	1	1	. . .	1.5	2	2.5	2.5	. . .	1
143	1	1	. . .	2	3	3.5	3.5	. . .	1
144	1	1	. . .	1.5	3.5	4	4	. . .	1
145	1	1	. . .	2	4.5	5.5	5.5	. . .	1

Table 4: Tunnel Pricing Matrix for Example Network

User ID	Password	Maximum BW in Mbps	Source Stub Address	Remote Stub Addresses
catispp	********	4	172.18.0.100 172.17.0.103	172.20.0.103 172.17.0.103

Table 5: SLA Database for some users

User ID	Source Address	Destination Address	Bandwidth in Mbps	Tunnel ID	Activation Time
catispp	172.18.0.100	172.17.0.103	1	140	6:17:20

Table 6: Connection Database

8.3 Performance Results

This section describes the performance of various QoS enabled tunnels that are setup by our Service Broker. The main intention here is not to find out the influence of buffer size or burst length on shaping or policing algorithm to illustrate performance changes, but rather to show the readers that such QoS mechanisms work with VPN tunnels established through our managed system.

For the demonstration of performance, we played some MPEG-I streams over various VPN tunnels. We selected three public domain bitstreams that we believe constitute a reasonable data set. Table 8 presents the characteristics of the bitstreams. Here, the three bit streams have different bit rate requirements.We believe the best metric to judge the performance of the QoS VPN tunnels is to measure to what extent the required bit rate is achieved while playing over those tunnels in real time.

Figure 14(a), 15(a), 16(a) show the bit rate distributions of the MPEG streams of *northamerica*, *heuris* and *sayit*. These rate consider the video frames only although the overall rates are higher. We first established a 1.25 Mbps VPN tunnel between routers Goppenstein and sarah for webserver (source) 172.17.0.103 and station 172.20.0.100 running MpegTV [15] player and played stream *northamerica* over that tunnel. As we can see from Table 8 that *northamerica* requires a rate 1.2184 Mbps, the trace of the recevied traffic as shown in Figure 14(b) demonstrates that capacity allocation was adequate to transmit this stream smoothly. In another set of experiments we again created 1.5Mbps and 2.5 Mbps VPN tunnel and ran *heuris* and *sayit* respectively. Both of these streams require 1.411 Mbps and 2.457 Mbps respectively. Output traces as shown in Figure 15(b) and 16 (b) again prove that allocation was sufficient.

Again, *sayit* was run from webserver 172.18.0.100 to station 172.20.0.100 over a tunnel between routers Appolo and sarah without any QoS enabled to it . Figure 17 shows the corresponding output trace. Since *sayit* requires 2.457 Mbps and no capacity was specifically allocated to the tunnel the stream ran at a slow frame rate. When at around 175th second some udp traffic was sent over the tunnel to fill up the pipe, this aggressive udp traffic completely stopped the transmission of the MPEG-I stream. When the udp transmission was stopped the stream again started running but still at a much slower rate than actually required for it. This further demonstrate the need for QoS in VPN tunnels and also show that such QoS mechanisms can work with various tun-

User ID	Source Address	Destination Address	Bandwidth in Mbps	Tunnel ID	Activation Time	Termination Time	Price in Cents
catispp	172.18.0.100	172.17.0.103	1	140	6:17:20	9:23:14	606.5

Table 7: Billing Database for user catispp

Stream	Stream Size (bytes)	Frame Size	Frames/ Second	Bit Rate (Per Sec.) only video	Mux Rate (Per Sec) with audio	I:P:B Ratio
northamerica	31096832	352x240	29.97	1.008	1.2184	18:31:51
heuris	20595900	352x240	29.97	1.152	1.4112	15:44:41
sayit	78021332	352x240	29.97	1.856	2.4576	15:41:42

Table 8: Sample MPEG-I Bitstreams

neling methods.

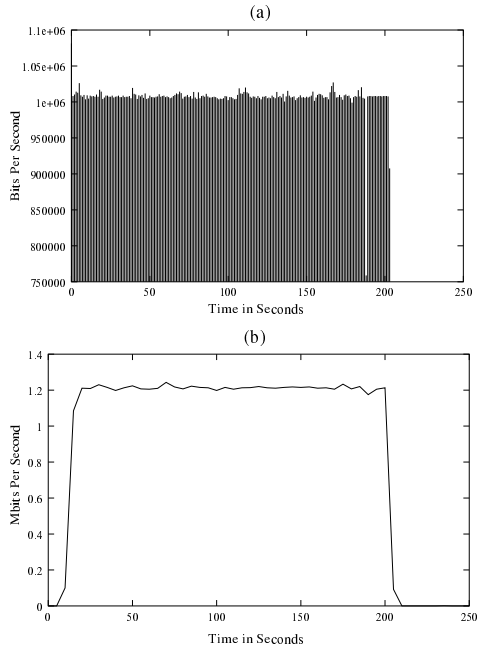


Figure 14: (a) Bit Rate Distribution of Video Framely Only for *northamerica*, (b) Output Traces over a 1.25 Mbps Tunnel

9 SUMMARY AND CONCLUSION

In this paper we have described the implementation of a Service Broker that allows registered users to establish and terminate QoS enabled VPN tunnels dynamically. As today's network infrastructure continues to grow, the ability to manage increasing network complexity is a crucial factor for QoS enabled VPN solutions. It is estimated that almost 40 percent of the total VPN budget in an organization is spent for deploying and management of VPN and network analysts advocate outsourcing the VPN services to the ISPs [25]. Based on these industry needs we have developed our VPN management system to be deployed by ISPs that would not only alleviate the pain of corporate administrators who often need human resources and huge amount of time in such complex implementations, but also benefit the service providers form the economic point of view.

The current version of our implementation supports a single ISP domain with simple SLA mechanism. Advanced SLA and edge configuration mechanisms have been addressed in [13]. Once we have better understanding of the dynamics of internal behaviour of such a managed system we will implement the ESB to provide end-to-end service across multiple domains. We need a robust signalling protocol for end-to-end resource allocation and our work [12] in this area is ongoing.

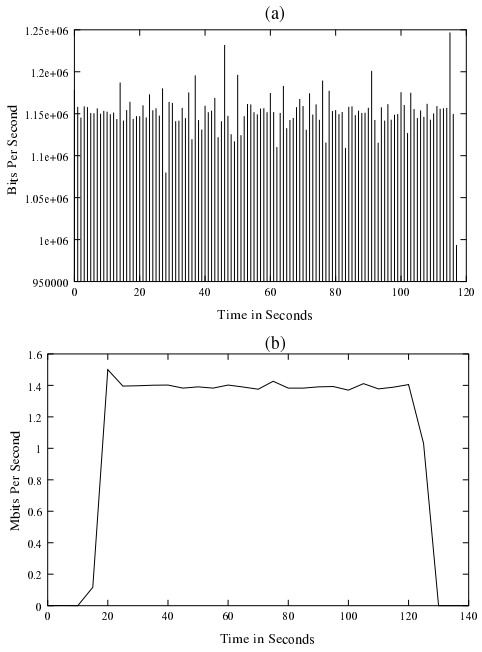


Figure 15: (a) Bit Rate Distribution of Video Framely Only for *heuris*, (b) Output Traces over a 1.50 Mbps Tunnel

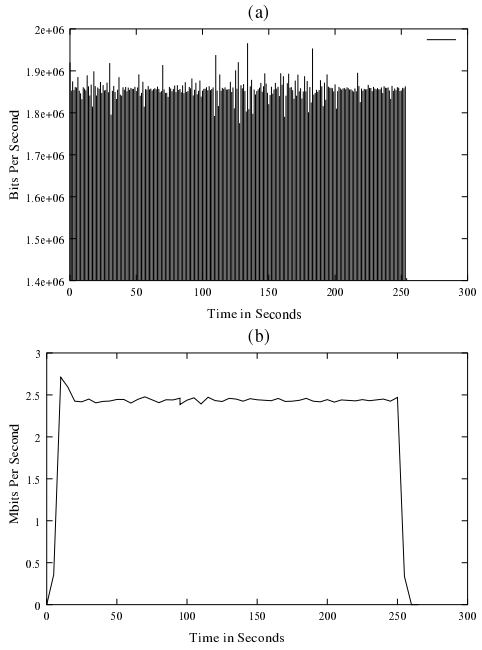


Figure 16: (a) Bit Rate Distribution of Video Framely Only for *sayit* , (b) Output Traces over a 2.5 Mbps Tunnel

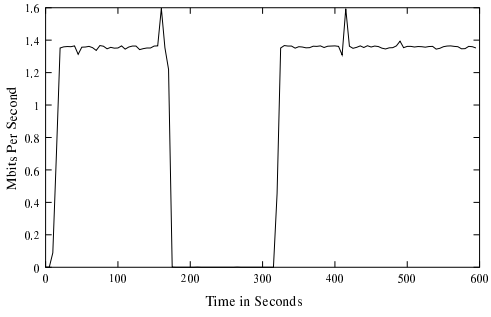


Figure 17: Transimission of *sayit* during a Period of Congestion

10 ACKNOWLEDGEMENTS

The work described in this paper is part of the work done in the project Charging and Accounting Technologies for the Internet (CATI) [19] funded by the Swiss National Science Foundation (Project no. 5003-054559/1 and 5003-054560/1). The implementation platform has been funded by the SNF R' Equip project no. 2160-053299.98/1 and the foundation Stiftung zur Förderung der wissenschaftlichen Forschung an der Universität Bern. The implementation platform was also partly supported by NEC Europe Ltd. The authors would also like to thank David Billard and Noria Foukia of University of Geneva for their generous help in the experimental setup.

References

[1] Yoram Bernet, James Binder, Mark Carlson, Brian E. Carpenter, Srinivasan Keshav, Elwyn Davies, Borje Ohlman, Dinesh Verma, Zheng Wang, and Walter Weiss. A Framework for Differentiated Services. Internet Draft `draft-ietf-diffserv-framework-02.txt`, February 1999. work in progress.

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

[3] S. Brim, B. Carpenter, and F. Le Faucheur. Per Hop Behavior Identification Codes. Internet Draft `draft-ietf-diffserv-phbid-00.txt`, October 1999. work in progress.

[4] David D. Clark. A Model for Cost Allocation and Pricing in the Internet. *Workshop on Internet Service Quality Economics, MIT*, Dec 2-3 1999.

[5] Sally Floyd and Van Jacobson. Link-Sharing and Resource Management Models for Packet Networks. *IEEE/ACM Transactions on Networking*, 3(4), August 1995.

[6] B. Fox and B. Gleeson. Virtual Private Networks Identifier, September 1999. RFC 2685.

[7] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis. A Framework for IP Based Virtual Private Networks. Internet Draft `draft-gleeson-vpn-framework-03.txt`, 1999. work in progress.

[8] M. Günter, T. Braun, and I. Khalil. An Architecture for Managing QoS-enabled VPNs over the Internet. In *Proceedings of the 24th Conference on Local Computer Networks LCN'99*, pages p.122–131. IEEE Computer Society, October 1999.

[9] V. Jacobson, K. Nichols, and K. Poduri. An Expedited Forwarding phb, June 1999. RFC 2598.

[10] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol, November 1998. RFC 2401.

[11] Ibrahim Khalil and T. Braun. Dynamic End-to-End Resource Reservation in Outsourced Virtual Private Networks. Technical Report IAM-00-005, Institut für Informatik, Universität Bern, Switzerland, Janurary 2000.

[12] Ibrahim Khalil and T. Braun. Implementation of a Bandwidth Broker for Dynamic End-to-End Capacity Reservation over Multiple Diffserv Domains. Technical Report IAM-00-006, Institut für Informatik, Universität Bern, Switzerland, April 2000.

[13] Ibrahim Khalil and Torsten Braun. Edge Provisioning and Fairness in DiffServ-VPNs. *IEEE International Conference on Computer Communication and Network (I3CN)*, Oct 16-18 2000.

[14] P E McKenny. Stochastic Fairness Queueing. *INFOCOM'90 Conference*, June 1990.

[15] MPEGTV. Mpeg tv home page. http://www.mpegtv.com/.

[16] Karthik Muthukrishnan and Andrew Malis. Core MPLS IP VPN Architecture. Internet Draft `raft-muthukrishnan-mpls-corevpn-arch-00.txt`, 2000. work in progress.

[17] K. Nichols, S. Blake., F. Baker, and D. Black. Definition of the differentiated services field (ds field) in the ipv4 and ipv6 headers, December 1998. RFC 2474.

[18] K. Nichols, Van Jacobson, and L. Zhang. A Two-bit Differentiated Services Architecture for the Internet, July 1999. RFC 2638.

[19] CATI Web Page. Charging and Accounting Technologies for the Internet (cati), Last update Tue Jul 7 09:42:02 MET DST 1998. http://www.tik.ee.ethz.ch/~cati/.

[20] D. Stiliadis and A. Varma. A general methodology for designing efficient traffic scheduling and shaping algorithms. *Proc. IEEE INFOCOM'97*, 1997.

[21] B. Stiller, T. Braun, M. Günter, and B. Plattner. Charging and accounting technology for the Internet. In *4th European Conference on Multimedia Applications, Services, and Techniques ECMAST'99*, LNCS 1629, pages 281–296. Springer-Verlag, May 1999.

[22] Ion Stoica, Hui Zhang, and T S E Ng. A Hierarchical Fair Service Curve Algorithm for Link-Sharing, Real Time and Priority Queues. *SIGCOMM'97 Conference*, September 1997.

[23] SWITCH. The switchlan backbone. http://www.switch.ch/lan/national.html.

[24] Andreas Terzis, Lan Wang, Jun Ogawa, and Lixia Zhang. A Two-Tier Resource Management Model for the Internet. In *IEEE Global Internet'99*, December 1999.

[25] VPNcon. Vpncon spring 2000. http://www.vpncon.com/spring/springvpn.htm.