Virtual Private Network and Quality of Service Management Implementation

R. Balmer, F. Baumgartner, T. Braun, M. Günter, I. Khalil

IAM-99-03

July1999

Virtual Private Network and Quality of Service Management Implementation

R. Balmer, F. Baumgartner, T. Braun, M. Günter, I. Khali

Abstract

This document describes the implementation of a quality-of-service (QoS)-enabled, Internet based, virtual private network(VPN) management system. For QoS mechanisms, the system focuses on scalable Differentiated Services, but the document also describes an implementation that allows the mapping of fine grained Integrated Services QoS mechanisms to Differentiated Services. The security mechanisms of the VPN management system are based on the Internet Protocol security architecture (IPSec).

CR Categories and Subject Descriptors: C.2.1 [Computer-Communication Networks] Network Architecture and Design; C.2.3 [Computer-Communication Networks] Network Operation; C.2.6 [Computer-Communication Networks] Internetworking

General Terms: Design, Management, Security

Keywords: Internet, Virtual Private Networks, Quality-of-Service, Differentiated Services, IPSec, Bandwidth Broker

1. Introduction

Today's Internet services are insecure and restricted to best-effort data packet transport. With new technology and protocols emerging, the service providers are given the chance to enhance their service beyond these restrictions. The resulting value-added network services call for more elaborate charging mechanisms than the flat-fee based charging of data transport services seen today.
In the work presented here, the focus is on providing a secure network service and to improve the competitiveness of such a service by adding Quality-of-service (QoS) enabling mechanisms. QoS in the network can be expressed as statistical statements about e.g. delay, jitter and bandwidth availability that the traffic experiences. Security of the network can be expressed in terms of the robustness of the involved protocols (including their built-in or attached cryptographic mechanisms) against accidental or intentional misuse.
The communication model for security that we use is that of a Virtual Private Network (VPN)[1]. VPNs allow its participants to communicate over a public infrastructure in a *transparently secure* manner. Here, we focus on IP-VPNs, which are VPNs using the Internet. In order to implement a VPN, IP traffic is usually tunneled and encrypted. Such techniques allow to authenticate all VPN traffic. This means nobody's data packets can pretend to be coming from a VPN participant other than the sender. Furthermore, the integrity and the confidentiality of all VPN traffic is ensured. Thus, nobody except the sender can change the traffic packets unnoticed and nobody except the receiver and the sender can read the contents of the packets. Protocols for IP-VPNs were recently specified. However, they always introduce complexity and management overhead to a network. Especially the operation of different network equipment and the concurrent operation of unprotected Internet access can add unbearable management load to a small or medium sized company's network administrator. This problem opens a new market for Internet service providers (ISP), namely a *VPN service* market. However, the ISPs need a management system, that allows its customers to customize their VPN service to their needs, as well as new charging and accounting facilities for the VPN service. An implementation of such a VPN service management system is described in this document. The implementation is an instance of a generic architecture, which allows for the Internet-wide composition of such configurable services [2]. A key issue for the success of a VPN solution is its QoS. While VPN solutions using the Internet accumulate less transport cost than their alternatives (leased lines, frame relay), they offer little or no QoS support. We propose to combine a VPN service with a QoS service in order to get a highly competitive product. Our management system supports the customizing and management of such a QoS VPN service. We use existing technologies such as the IPSec protocol [3] for tunneling and encryption of the VPN traffic and Differentiated Service (DiffServ, DS) for the QoS support. The protocols are both proposed by the IETF and are designed to interoperate.

1.1 Differentiated Services

In the Integrated Service (IntServ) architecture [4], network resources are allocated per flow at each reservation capable router between the traffic source and the destination. However, this leads to an unbearable burden for the routers of the core network. If IntServ would be deployed in the backbone networks, each of their routers would have to keep track of millions of flows simultaneously. Therefore, the IETF came up with the Differentiated Services concept [5]. DiffServ is a light-weight and scalable QoS mechanism. A single byte (DS Code Point, formerly called TOS) in the IP header is used to code different per-hop behaviors (PHB) that an IP packet can experience. Inside of a network, all IP traffic using the same code point is called 'DiffServ behavior aggregate' and is treated the same way. Since there is only a handful of PHBs, the DiffServ architecture scales also to large core networks.
Up to now, the IETF proposed two PHBs. One is called Expedited Forwarding (EF) [9] and the other

is called Assured Forwarding (AF)[10]. The EF service allows packets to be forwarded at a constant bit-rate. As long as a traffic source sends its traffic in-profile, the traffic is guaranteed to be delivered within a well-defined delay. Traffic is in-profile if the packet rate does not exceed a rate that was previously agreed upon with the service provider. For its constant delivery rate, and its predictable delay characteristics, the EF service is also referred to as 'virtual leased line' service.
The Assured Forwarding (AF) service is more flexible. It allows for the composition of a variety of different services. AF allows for a service which can handle bursts by using buffering mechanisms. It contains up to four (but at least two) service classes. Each service class is logically separated from each other, which must be ensured by buffering and scheduling mechanisms. The classes are ordered by priority; a different amount of resources is allocated for each class. That way, the AF service supersedes the previously suggested 'Olympic Services' with 'gold', 'silver' and 'bronze' services. AF traffic always stays within the same class, so that the packets of a flow using one AF class are not accidentally delivered out of order. The reclassification of AF traffic in case of congestion is done by the use of three different drop precedences. Packets with the lowest drop-precedence are discarded first. Out-of-profile packets from lower drop-precedences are reclassified to higher drop-precedence. New PHBs can be defined by following the guidelines provided by the IETF in [11].

1.2 Structure of the Document

We have now introduced the context of the document and the basic terminology and technologies involved. In section 2. we place the QoS VPN management system in the context of the generic architecture we developed. Section 3. is the core of this document. It presents the implementation of the management system. This includes the system's structure and information flow involved, various security and QoS mechanisms used, the charging mechanism, the interfaces and configuration examples. Section 4. presents ways to use the fine-grained IntServ resource reservation mechanisms on top of a DiffServ backbone network. Section 5. concludes this report.

2. QoS VPN Implementation as an Instance of a Generic Architecture

VPNs and DiffServ are just two of many services that providers could offer using their control over their networks. However, the different providers have to interoperate to set up a service that exploits the global connectivity of the Internet. They need an infrastructure for the following purposes:

- Charging and accounting between their customers and between their peer providers.
- Automatic service negotiation, establishment and maintenance mechanisms.
- An automatic service configuration interface for the customers.

To address these challenges, we took the idea of the Bandwidth Brokers as described in [8]. We generalized the idea. Instead of bandwidth brokers, which are software agents that manage bandwidth in the DiffServ architecture, we propose a service broker hierarchy (in our VPN architecture paper [2]). The next two sections present the architecture and its partial realization.

2.1 A Broker Hierarchy for Configurable Network Services

The hierarchy is structured into functional components. We made the distinction between domain-local 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 1. 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. The external service broker (ESB) handles the negotiation between brokers of peer ISPs across the trust border. The result of a negotiation is a service level agreement (SLA), which describes the collaboration between the networks of two providers. A broker signaling protocol is necessary to set up a unbroken chain of SLAs between all involved ISPs in order to set up an Internet wide service.

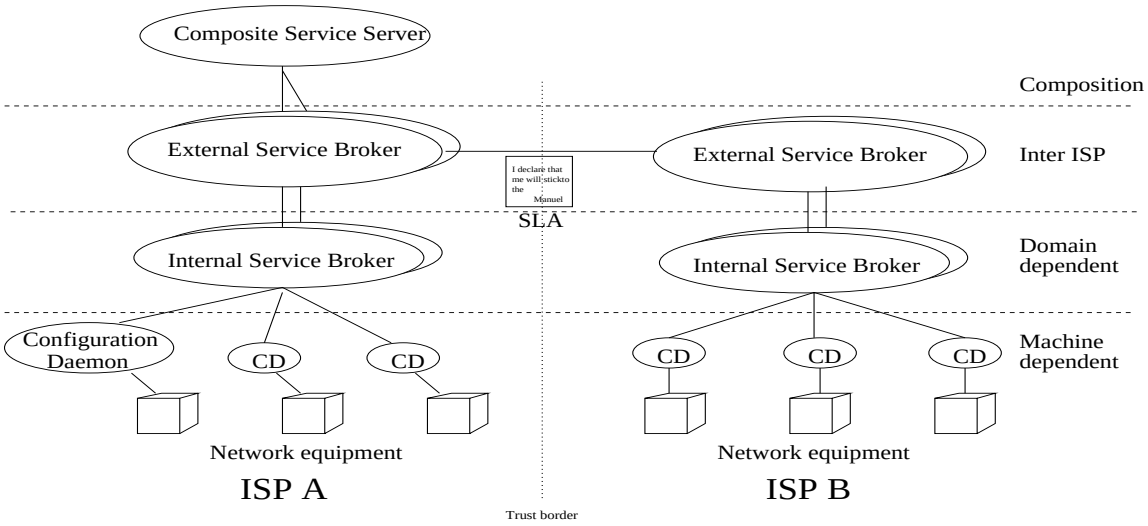


FIGURE 1. A broker hierarchy for configurable services.

In the work described here, we focus on a particular instance of this hierarchy, where the services are QoS support and VPN support. The technologies used are a proprietary IPSec implementation, proprietary queuing mechanisms (both included in Ciscos Internet Operating System (IOS)) and ATM. Unfortunately, the IPSec and the DiffServ services are not orthogonal. The encryption used within IPSec can interfere with QoS policies. Therefore, as shown in Figure 2, the implementation of a QoS VPN service uses the composition layer of the broker hierarchy.

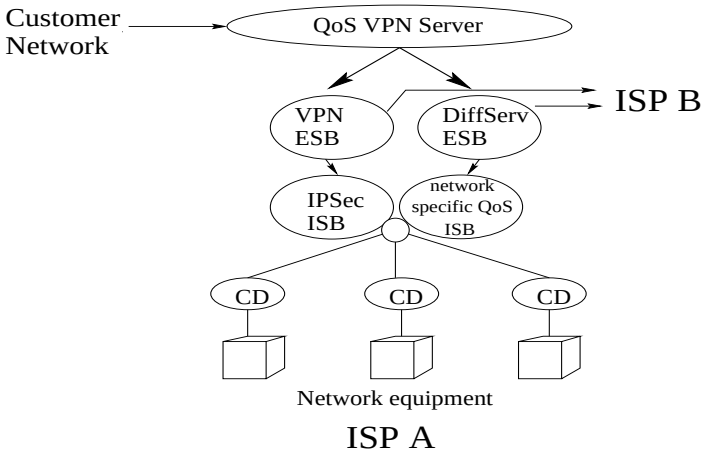


FIGURE 2. A QoS VPN service in the broker hierary.

2.2 Implementation of the Architecture

The architecture provides a framework for global deployment of customizable Internet network services. The implementation of the whole framework is way beyond of the goals of our project. Nevertheless, important aspects of the architecture are about to being implemented. This documents the implementation of a QoS VPN instance of the architecture. Here, we start with implementing config-

uration daemons, specialized to control the commercial routers that we use. Then we add internal service brokers for QoS and IPSec configurations customized to our test network. The implementation as described in this document furthermore contains a light weight charging and accounting mechanism taking the role of the external service broker. Since the interaction between peer ESBs involving SLAs is not yet specified to all detail, the implementation described here does not include a full-fledge SLA and inter-domain signaling mechanism. However, another kind of service is implemented and described here. The automatic mapping of integrated services, a fine-grained, but not scalable resource reservation mechanism, to the DiffServ mechanisms. The mapping is implemented via configuration daemons and brokers, too, and it is described in Section 4.
But first, we proceed with the description of the bottom-up implementation of a QoS VPN service management system, which is our first proposed instance of the broker architecture.

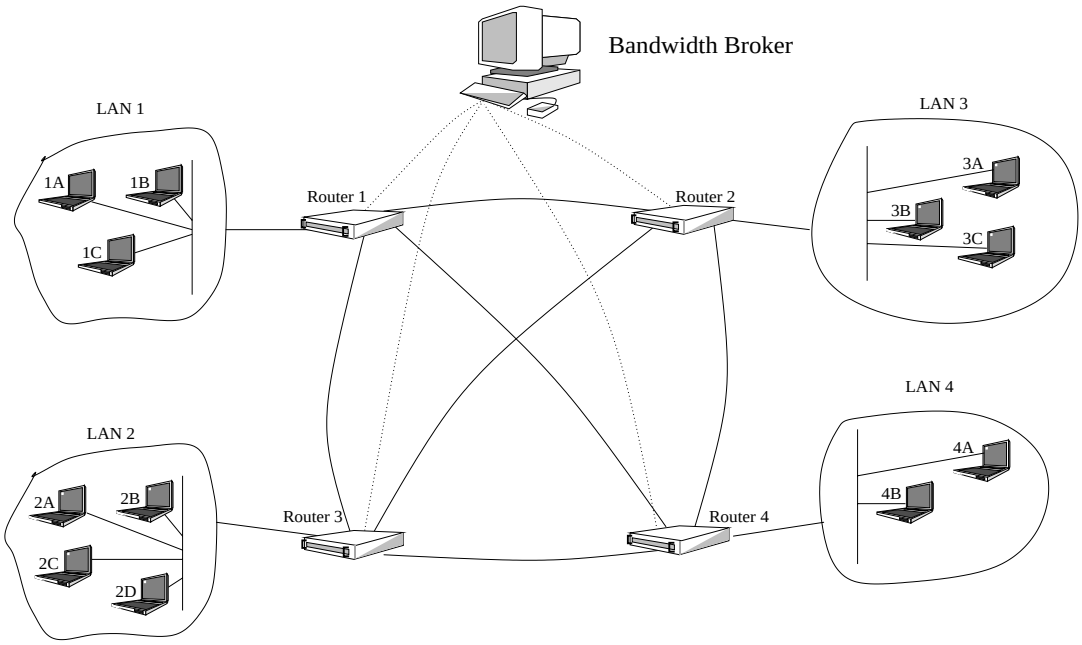
3. The DiffServ VPN Service Management

As today's network infrastructure continues to grow, the ability to manage increasing complexity is a crucial factor for VPN solutions. As extensions of the enterprise network, a VPN naturally increases network traffic as well as the risk that network performance may be affected. 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. This ensures that the performance of mission-critical and other high priority applications without starving out lower priority applications. This is absolutely critical to ensure that the VPN can deliver the myriad number of benefits of this rapidly growing technology.

It is widely accepted that the economic objective of Virtual Private Network is to emulate the private networks using widely available and less expensive public resources while retaining the latter's characteristics. These characteristics are:

- Private line networks has closed user group properties where only the group of entities that are connected to the private line can communicate with each other.
- Data in private networks is absolutely secure in the sense that it remains free of hackers access.
- Since data in private networks don't get mixed up with traffic originating from outside sources both bandwidth and delay can be readily guaranteed.

Figure 3 depicts a situation where VPN constructed over public internet has resemblance with private networks and inherits the properties outlined above. Different closed user groups or a specific user have various distinct bandwidth and security needs. For example, source 1A needs a 2Mbps line to destination 3A while it needs a T1 equivalent line while connecting to 3B. In our implementation we facilitate all these features of a private network and present them as user configurable network services. As World Wide Web (WWW) is widely available we provide web based interfaces 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. In the following sections we will describe details of various components and system flows of such a VPN management system.



Source	Destination	Bandwidth Reqd
1A	3A	2Mbps Dedicated
1A	3B	T1 Dedicated
1A	3C	1 Mbps Dedicated
..		
..		
::		
..		
LAN 1	LAN 4	1 Mbps Dedicated
LAN 2	LAN 3	1 Mbps Dedicated
LAN 3	4A	2 Mbps Dedicated
..		
..		
::		
..		

FIGURE 3. VPN in a closed user group: a single ISP layout.

3.1 Structure

The ISB, which is the heart of our VPN management system, takes the role of a QoS manager to optimally configure network resources and takes adaptive decisions 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 ISB with a standard interface to the network resource.

Our ISB interacts with specialized configuration daemons when a certain user request arrives to setup a tunnel and the ISB has to decide whether it can allocate enough resources to meet the demand of that tunnel. Referring to Figure 3 this request might come from 1A to establish an encrypted 2Mbps dedicated tunnel with 3A. The basic operation of our system is as follows: based on request parameters provided by the user, the ISB 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 not. If this is not the case, the ISB 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 ISB invokes pricing database to calculate the pricing for that tunnel.

In the next section we will briefly describe these components and their role before we move to

detailed description of system flows in section 3.3. In section 3.4 we describe the call acceptance and rejection procedure with an example.

3.2 Components of the System

All the components mentioned above play an important role in ISB's decision making prices to accept or reject a VPN setup request.

The **SLA database** contains not only 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 3 where LAN 1, 2 and 3 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. This also prohibits malicious users to setup unauthorized tunnel for others to avoid basic setup fee. 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, service type, maximum bandwidth, range of valid VPN areas>

The **connection database** contains a list of currently active VPNs. It has various functions:
(1) when a new request arrives for connection or tear down ISB can check if that connection already exists or not and then make it's decision,
(2) it indicates how much resources have been consumed by VPN users,
(3) provides a record to pricing mechanism. It's tuples are:

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

The **resource database** contains resources available between any two adjacent routers. This means that this database has resource information of all the routers in a certain domain. In our implementation we keep records of precomputed tunnels with all the features it has. This includes tunnel's source address, destination address, the tunnel routers' administrative address, bandwidth, type indicating what kind of traffic it can carry (e.g. CBR, GS) etc. It also includes the availability of the tunnel.

<tunnel id, tunnel source addr, source router, tunnel destination addr, desination router, bw, type, availability>

The **pricing database** contains pricing of various tunnels. It's only interaction with the ISB is at the time when a connection (tunnel) is deleted and the ISB needs to know the price of that by making a query to it. More is described about it in section 3.6.

3.2.1 Realization of Components of the Architecture

As mentioned in Section 2.2 the implementation presented here is an instance of a generic architecture. The architecture distinguishes between external and internal brokers for VPN and for QoS. The implementation realizes those for components in one service broker. The functionality of this service broker is limited due to ongoing work. The main limitation concerns the ESB to ESB communication that is necessary in a multi provider environment. It is subject to current and future research [12]. Thus the ESB to ESB interfaces do not exist as implementations at the moment. This is the main reason we call the implemented broker still an internal service broker, even though it handles charging between the provider and its users. Combining some functionalities of external and internal brokers, the resulting internal broker looks as depicted in Figure 4.

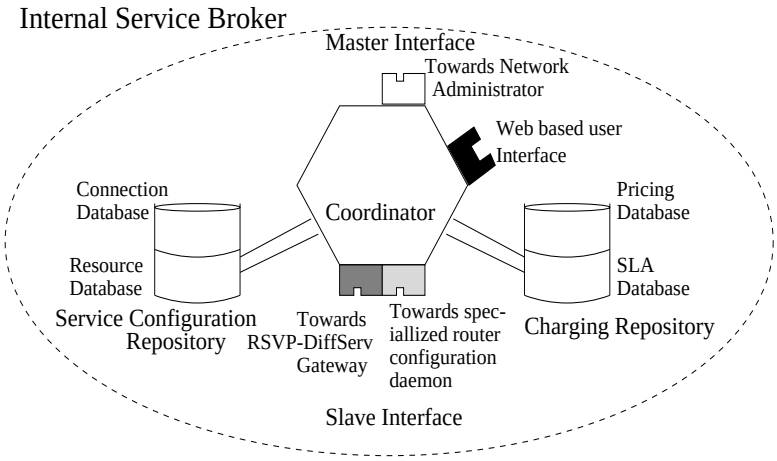


FIGURE 4. Realized ISB.

3.3 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 ISB, under which circumstances a new connection request or tear down request is refused.

3.3.1 The Successful Connection Establishment (Figure 5)

1) A user sends a connection request message to the ISB for a new connection request from the WEB or via other signalling mechanism such as RSVP. The ISB 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 address for the tunnel, amount of bandwidth and encryption method.

2,3) The ISB 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 ISB 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 ISB contacts the connection database to check the existence of an exactly similar tunnel. This is because between a source and destination only one tunnel can remain active.

8,9) The ISB asks the resource database to allocate an encrypted tunnel with a certain amount of resources. The resource database responds to the ISB and either allocates the resource or denies based on resource availability.

10) The ISB finds the correct resource type and it 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 routing 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 acknowledgment to ISB which sends a notification to the user.

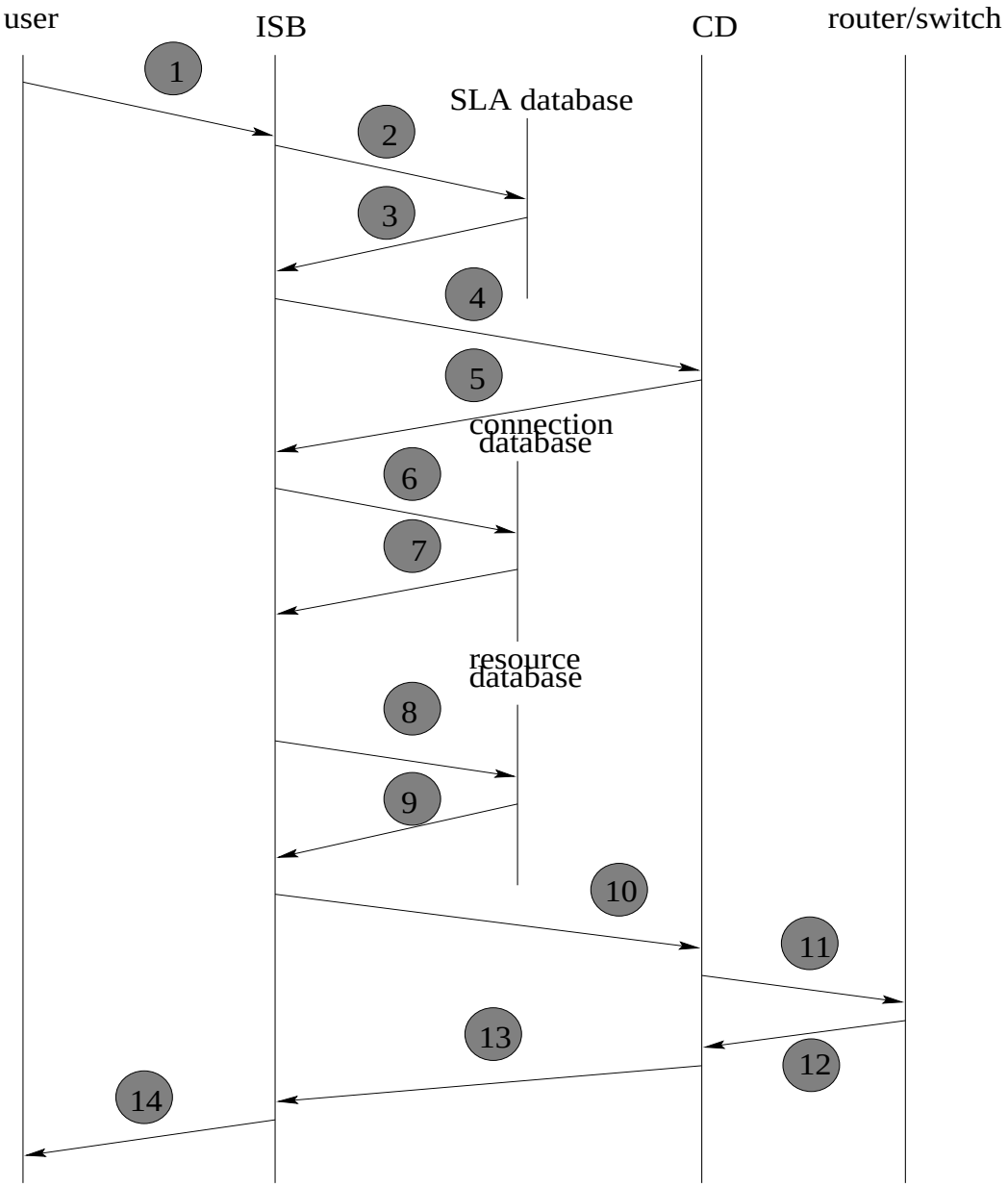


FIGURE 5. Successful setup.

3.3.2 The Connection Rejection (Figure 6, Figure 7)

A connection is rejected if
(i) the SLA profile doesn't match (Figure 6(a)) or
(ii) the daemon is found busy (Figure 6(b)) or
(iii) the connection already exists (Figure 6(c)) or

(iv) not enough resources are available.

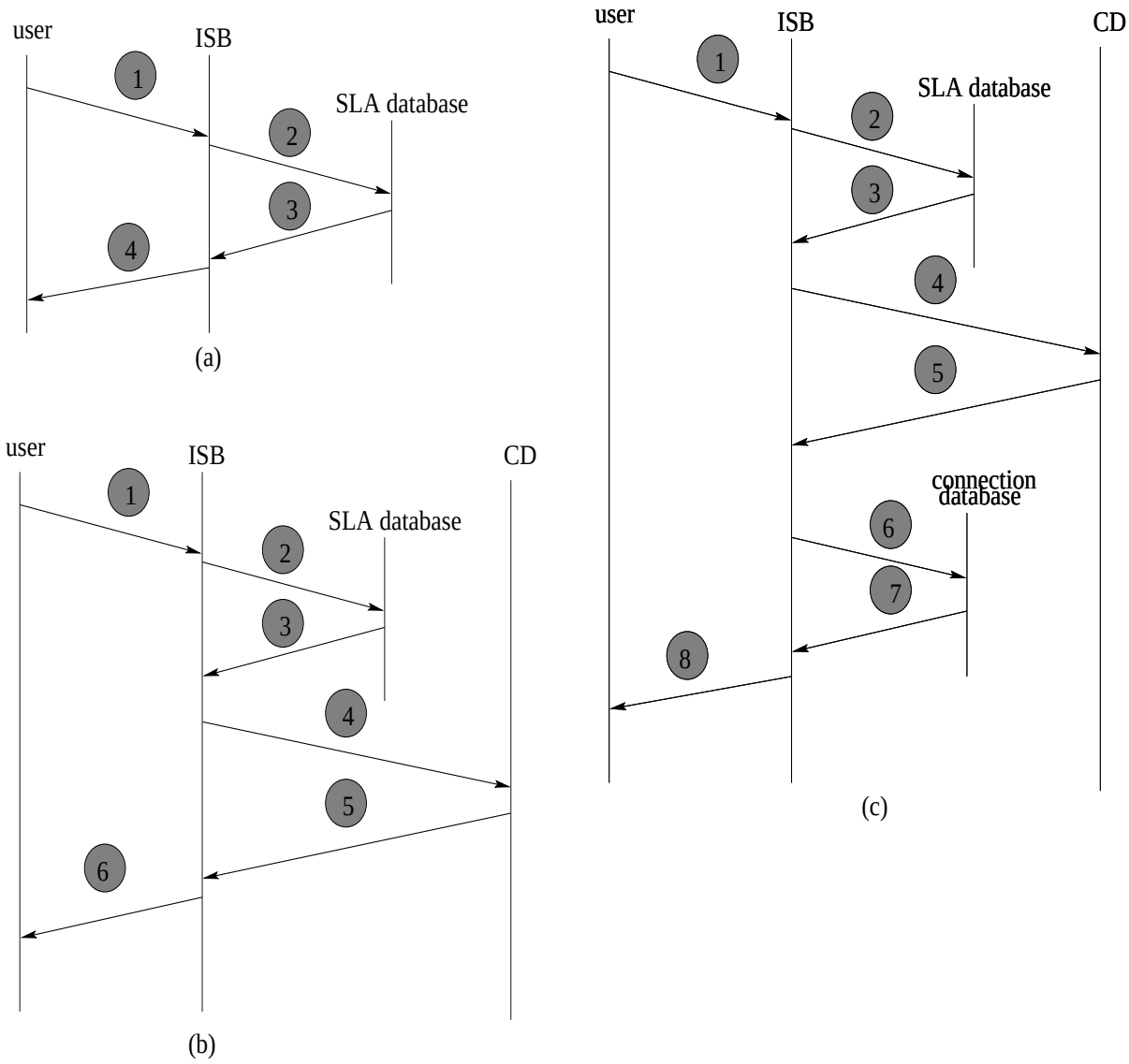


FIGURE 6. Denial of setup.

Regarding case (i), user id, password might be wrong, VPN areas for which a user wants to establish a tunnel might be invalid, or the bandwidth requested might be higher than the agreed one, and in such a case a connection will not be granted.

In the next case, even if the request parameters are valid the CD might be found busy and hence, the connection will not be possible.

In case (iii), 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.

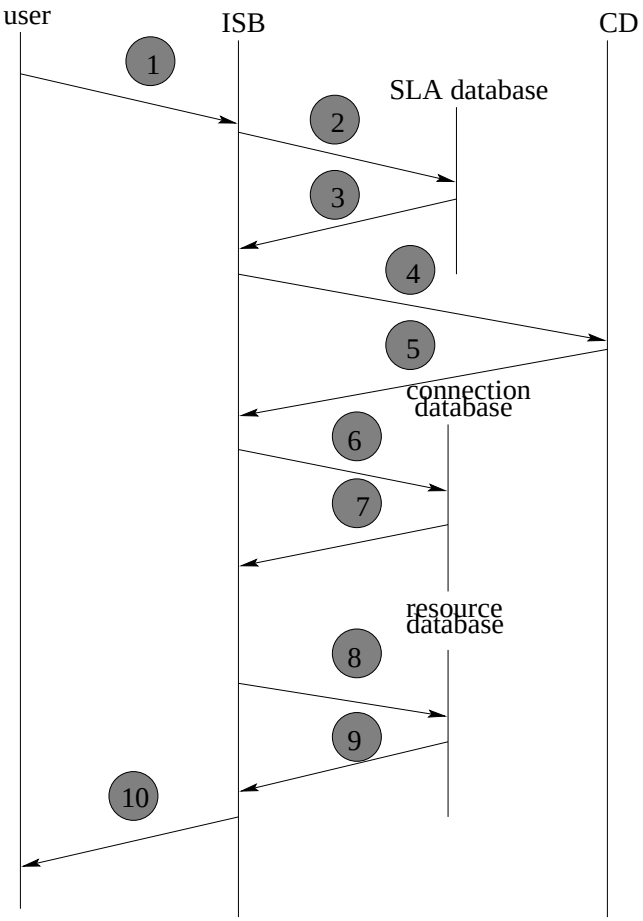


FIGURE 7. Denial because of resource unavailability.

Case (iv) happens if the requested resource type is not available (Figure 7). From step (1) to (7) the ISB receives positive acknowledgments. Then, it asks the resource database to grant the request type user has asked for (step 8). If the resource database cannot meet that demand it sends the signal (step 9) to the ISB about resource unavailability, and the ISB forwards that message to the user.

3.3.3 Successful Connection Deletion (Figure 8)

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.

8) If the connection is found in the connection database the ISB talks to the CD to create an appropriate routing 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.

9,10,11,12) CD creates and sends the configuration script to the router. The router sends signal to the CD which then confirms the ISB about the configuration. The ISB finally forwards this positive acknowledgment to the user.

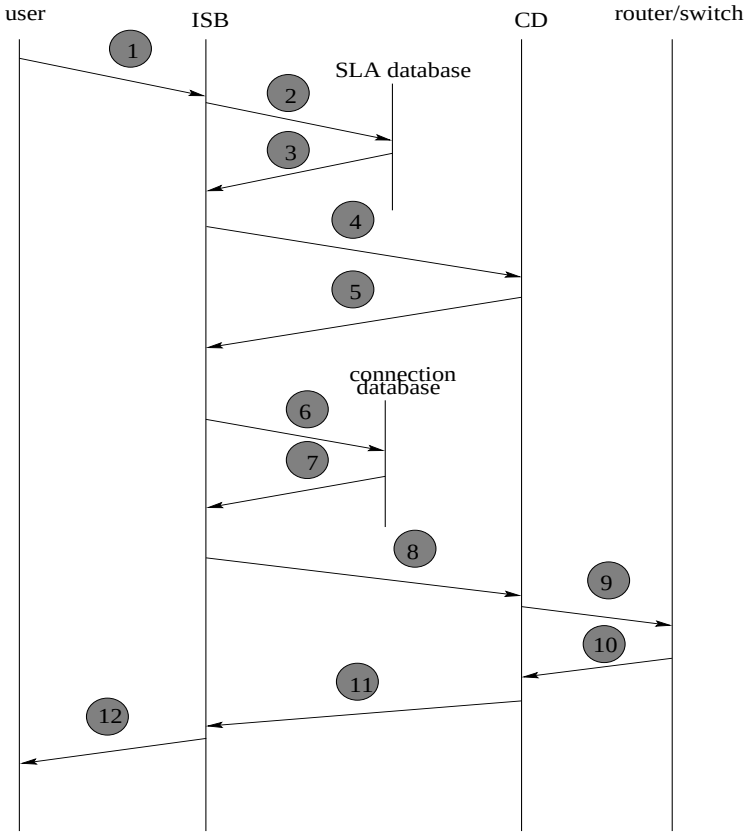


FIGURE 8. Successful discard of a service.

3.3.4 Failure in Connection Tear-Down (Figure 9)

1,2,3,4,5,6) These steps are similar to the steps as discussed for the connection acceptance procedure.

7) If no connection is found in the connection database for deletion of a request or if the tunnel doesn't belong to the owner then the ISB is informed about that.

8) The ISB sends an error message to the user.

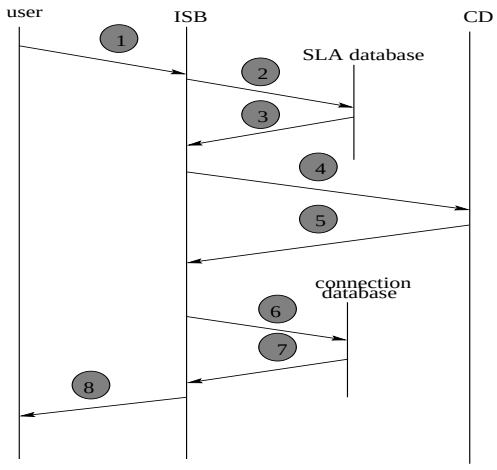
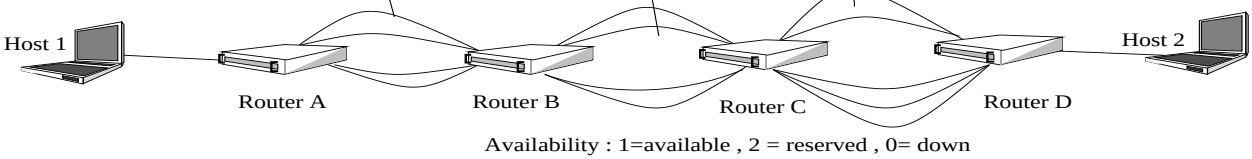


FIGURE 9. Failure in connection teardown.

3.3.5 Example of Call acceptance (Figure 10)

First, consider an example where a user wants to establish a 1Mbps IP/IP encrypted tunnel between Host 1 and Host2 via the WWW interface. He talks to the ISB which checks SLA validity, daemon status and connection database. If everything goes well, the ISB will check the availability of an 1 Mbps tunnel between A and B, between B and C, and between C and D. It finds that between A and B tunnel 4 is free, between B and C tunnel 6 is free and between C and D tunnel 11 is free. Therefore, the request is accepted. In our implementation by using policy routing we force the traffic to follow a certain path to meet the QoS required by them. One might ask how do we find the route A-B-C-D required for Host1's traffic to travel to Host2. We have created a small static routing table where each router knows it's next hop router. If Host1 and Host2 are directly connected to A and D respectively their address can be found by various methods, for example by traceroute. Since from our static routing table A knows B's address as the next hop router, similarly B can also find it's next hop router, and this search continues until D is found. This procedure greatly helps the ISB to dynamically determine how and which routers should be configured.

Peer Router	Peer Router	Tunnel ID	BW (Mbps)	Servioce Type	Availability
A	B	1	2	GS	2
A	B	2	1	GS	1
A	B	3	1	GS	1
A	B	4	1	GS	1
B	C	5	2	GS	2
B	C	6	1	GS	1
B	C	7	1	GS	1
B	C	8	1	GS	1
C	D	10	2	GS	2
C	D	11	1	GS	1
C	D	12	1	GS	1
C	D	13	1	GS	1
C	D	14	4	GS	1



Availability : 1=available , 2 = reserved , 0= down

FIGURE 10. Example of a Call acceptance.

3.4 Security Configuration

The most basic 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.

Tunneling (also called encapsulation) is a method of wrapping a packet in a new one thus providing it with a new header. The whole old packet becomes the payload of the new one as shown in Figure 11. If the encapsulated packet is encrypted, a hacker cannot figure out, for example, the destination address of that packet and contents of it as well. Tunneling 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.

Any tunneling protocol (IPSec, IP over IP, L2TP) can be used in our architecture, but in the current implementation IPSec and IP over IP (or GRE) are used. As in tunneling the encapsulated header is not processed by the internet routers and only the endpoints of the tunnel (the gateways) need to have globally assigned addresses. The hosts in the private LANs behind them can have private addresses. However, in special cases private addresses can also be used for tunnel endpoints if we use policy

based routing when packets passing through an encrypted tunnel are forced to choose the next hop router to meet the QoS requirements.
In the Internet the security requirements of one user might vary from others under various circumstances depending on his needs. Our implementation provides multiple options and allows a user to choose any of them to deployed in his/her own tunnel. In general, the followings are needed 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 data was originated from the sender. This service, however, is dependent upon the data integrity service.

To facilitate these features we have provided various user selectable options described in the following subsections.

3.4.1 IPSec AH in Tunnel Mode

The Authentication Header (AH) (RFC 2402) is used to provide integrity and authentication to IP datagrams. When packets go through an AH type tunnel an AH is embedded in the data to be protected (a full IP datagram). AH authenticates as much of the IP datagram as possible. Some mutable fields like TOS, Flags, Fragment Offset, TTL, Header Checksum in the new IP header aren't protected by AH. Packets that fail authentication are discarded and never delivered to upper layers. 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. This mode is enabled in a router by issuing the following command

```
crypto ipsec transform-set ah-md5-hmac ah-md5-hmac
```

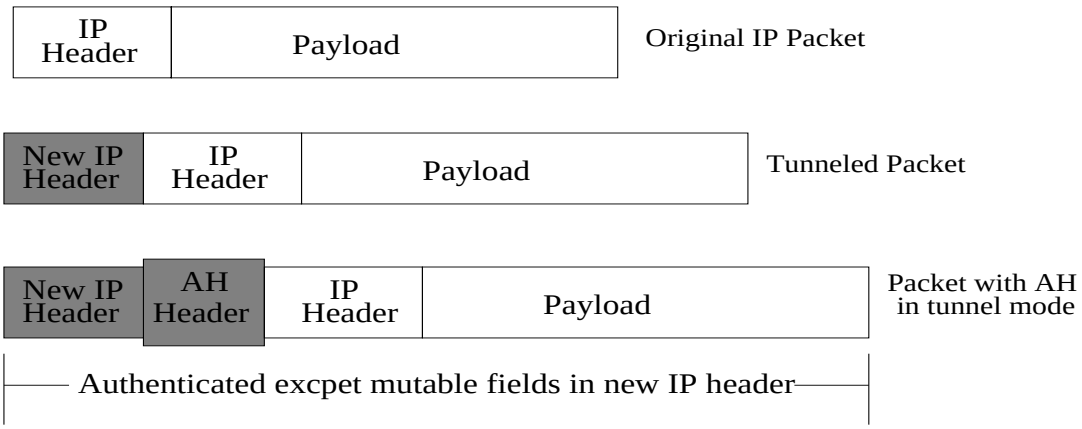


FIGURE 11. AH tunnelled packets.

The AH protocol allows for the use of various authentication algorithms. For point-to-point communication, suitable authentication algorithms include keyed Message Authentication Codes (MACs) based on symmetric encryption algorithms (e.g., DES) or on one-way hash functions (HMAC). In our implementation we focus on HMACs. The HMAC algorithm (RFC 2403) provides a framework for inserting various hashing algorithms such as MD5 to authenticate packets. Our implementation

uses the two mandatory hash protocols MD5 (Message Digest 5) and SHA-1 (Secure Hash Algorithm 1).

3.4.2 IPSec ESP in Tunnel Mode

The Encapsulation Security Payload (ESP) is used to provide integrity check, authentication and encryption to IP datagrams. 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.

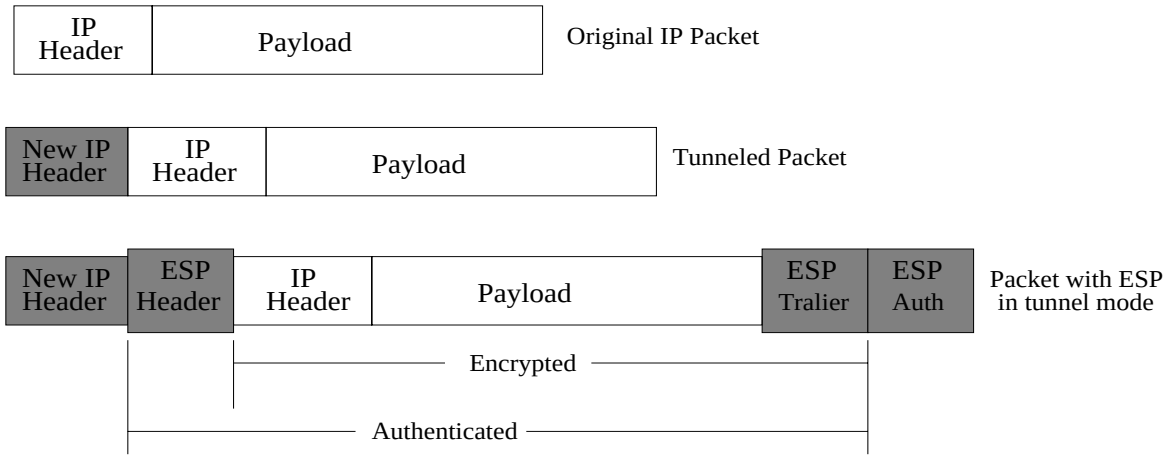


FIGURE 12. ESP tunnelled packets.

A new packet is constructed with a new IP header and then the ESP header is inserted right after the IP header as illustrated in Figure 12. Since the original datagram becomes the payload data for the new ESP packet, its protection is total if both encryption and authentication are selected. However, the new IP header is still not protected. This mode is invoked in the router by issuing the following command:

```
crypto ipsec transform-set esp-encryption esp-des esp-md5-hmac
```

3.4.3 Combined Tunnel Mode

Even though most of tunnel gateways are required to support only an AH tunnel or ESP tunnel, it is sometimes desirable to have tunnels between gateways that combine both IPSec protocols (Figure 13). 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 the Figure below. In our implementation this mode can also be enabled as follows:

```
crypto ipsec transform-set ah-md5-hmacANDesp-des ah-md5-hmac esp-des
```

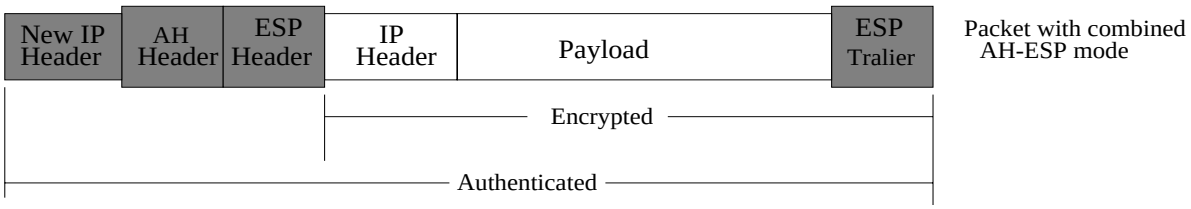


FIGURE 13. Combined tunnel mode.

3.5 Quality-of-Service Configuration

With Cisco's custom output queueing (see Figure 14), a weighted fair queueing strategy is implemented for the processing of interface output queues. It would then be possible to control the percentage of an interface's available bandwidth that is used by a particular kind of traffic. When custom queueing is enabled on an interface, the system maintains 17 output queues for that interface that can be used to modify queueing behavior. One can specify queues 1 through 16.

For queue numbers 1 through 16, the system cycles through the queues sequentially, delivering packets in the current queue before moving on to the next. Associated with each output queue is a configurable byte count, which specifies how many bytes of data the system should deliver from the current queue before it moves on to the next queue. When a particular queue is being processed, packets are sent until the number of bytes sent exceed the queue byte count or the queue is empty. Bandwidth used by a particular queue can only be indirectly specified in terms of byte count and queue length.

Queue number 0 is a system queue; it is emptied before any of the queues numbered 1 through 16 are processed. The system queues high-priority packets, such as keep-alive packets, to this queue. Other traffic cannot be configured to use this queue.

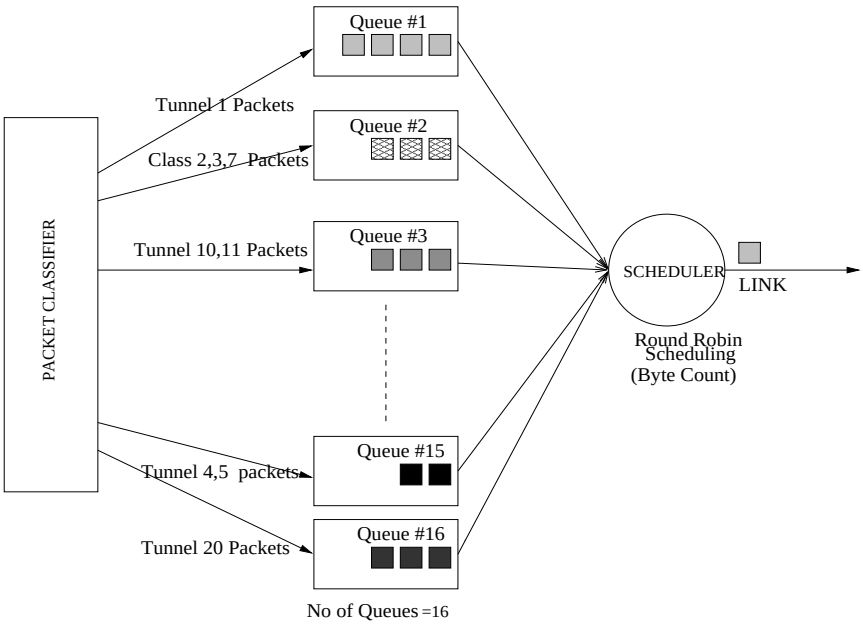


FIGURE 14. Custom queuing.

In our implementation we allocate bandwidth to the tunnel in advance as it was the case in encryption. A new request to establish a tunnel between a source and destination the is simply mapped into the appropriate tunnel which can satisfy the bandwidth requirement. Here is an example how we can create precomputed tunnels of 5 Mbps and 2.5 Mbps with the queuing mechanism described in this section. Assume that the total bandwidth of the physical interface is 10 Mbps.

```
access-list 121 permit ip host 2.2.2.2 host 1.1.1.1

access-list 122 permit ip host 2.2.3.1 host 1.1.3.1

queue-list 1 protocol ip 1 list 121

queue-list 1 protocol ip 2 list 122

queue-list 1 default 3

queue-list 1 queue 1 byte-count 3000

queue-list 1 queue 2 byte-count 1500

traffic-shape group 121 5000000 120000 10000 1000

traffic-shape group 122 2500000 120000 10000 1000
```

In the first two lines tunnels are first classified and in line 3, 4 and 5 tunnel (2.2.2.2 -> 1.1.1.1) is asked to send its traffic to queue 1 and tunnel (2.2.3.1->1.1.3.1) is told to pass through queue 2 while all the best effort traffic is told to pass through queue 3. In line 6 and 7 we are defining that in a round robin cycle queue 1 will first send 3000 bytes followed by queue 2 and 3 which are each allowed to send 1500 bytes. Therefore, queue 1 gets 3000/ (3000 + 1500 +1500) * 100 = 50% of the bandwidth while by a similar calculation both queue 2 and 3 gets 25% each. Since the link bandwidth is 10 Mbps tunnel (2.2.2.2 -> 1.1.1.1) gets 5 Mbps and tunnel (2.2.3.1->1.1.3.1) is allocated 2.5 Mbps. Only 2.5 Mbps is left for rest of the traffic. As we know that traffic does not always behave well, we also shape the traffic in case it violates contract and send more traffic than agreed previously. The last two lines enforce the agreed contract. Note that scheduling mechanism is round robin and there is no priority assigned to any queue. Hence no delay can be guaranteed. Note, that the addresses of tunnel end-points mentioned in the example are loopback addresses.

3.6 QoS for VPN with Diffserv-ATM mapping

All along we have been assuming that end to end tunnels are created in a homogenous Internet environment. However, today's Internet might consist of high speed networks like ATM. ATM's use has become a norm not only in the backbone, but also in campus network gateways where aggregated traffic might be quite substantial. In Figure 15 we try to depict such a scenario where most of the network consists of diffserv clouds and an ATM cloud sits in between them. LAN 1 from site 1 may want to establish 2 Mbps GS tunnel with LAN 4 in site 2 and we assume that this is provided in segment A-B and segment C-D. How about segment B-C? We need to provide something similar which would be equivalent 2 Mbps Guaranteed Service (GS). Before we discuss the mechanism to map from diffserv to ATM network we give a short description of what kind of services are supported in our mapping mechanism.

In the context of ATM networks there are constant bit rate (CBR), real time variable bit rate (rtVBR), non-real-time variable bit rate (nrt-VBR), unspecified bit rate (UBR), and available bit rate (ABR). In the context of Intserv/ Diffserv Internet there are guaranteed services (GS) /Premium service, Controlled load (CL)/ Assured service, and best effort service.

The behavior of the rtVBR/CBR and GS/Premium service categories are essentially the same and are designed to accommodate real-time voice and video and other applications requiring strict bounds on delay and delay variation and very low packet loss. To provide this type of service, fixed bandwidth allocation is almost always the only viable option.

The UBR and Best Effort service categories were envisioned for information retrieval, bulk transfer, and other applications where delay is not critical. No QoS guarantees of any sort are associated with UBR. However, even if QoS guarantees are provided, a major issue with UBR VPCs is the lack of per-VC fairness and isolation of the congestion control schemes in the core ATM switch.

Available bit rate service was meant to give QoS guarantees that can possibly change over the duration of the connection. A connection giving a controlled service contract should appear as a lightly loaded queue. Both ABR and controlled load can tolerate light packet loss and both have delay requirements as strict as rtVBR/CBR and GS/Premium service.

Based on above comparison we have adopted the following table in our implementation:

Internet Type of Service	ATM Type of Service
GS /Premium	CBR/ rtVBR
Best Effort	UBR
CL/Assured	ABR

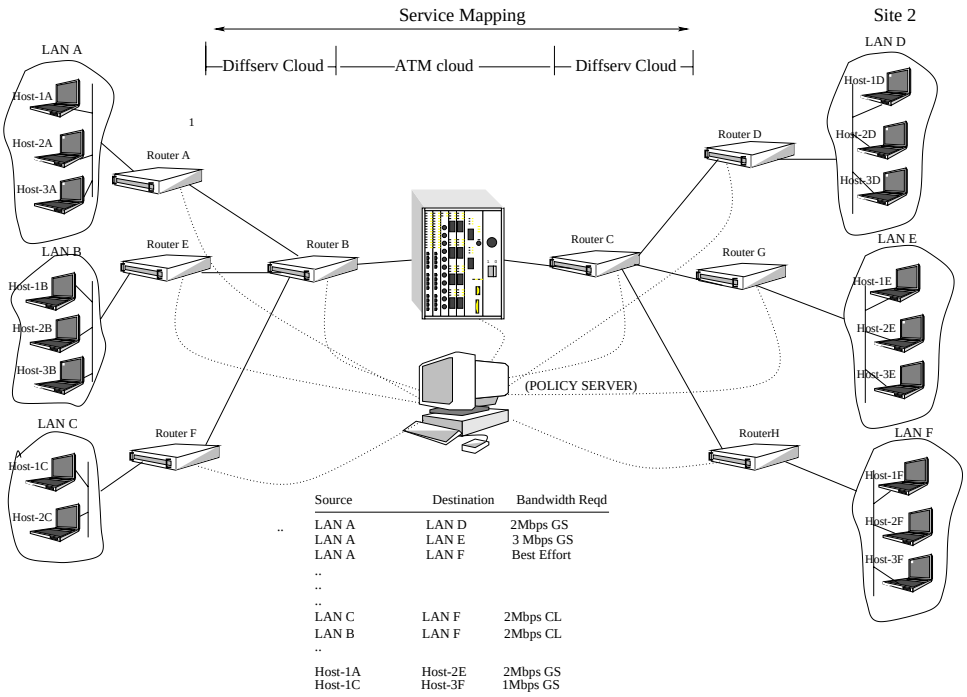


FIGURE 15. An ATM backbone network.

To explain how we provide this appropriate mapping mechanism in our implementation lets look at a similar example (Figure 10) we described earlier. The only difference is that in this case there is an ATM Switch between router B and C which provides various kinds of PVC. The resource database is to be slightly modified to reflect the new ATM resources as shown in Figure 16.

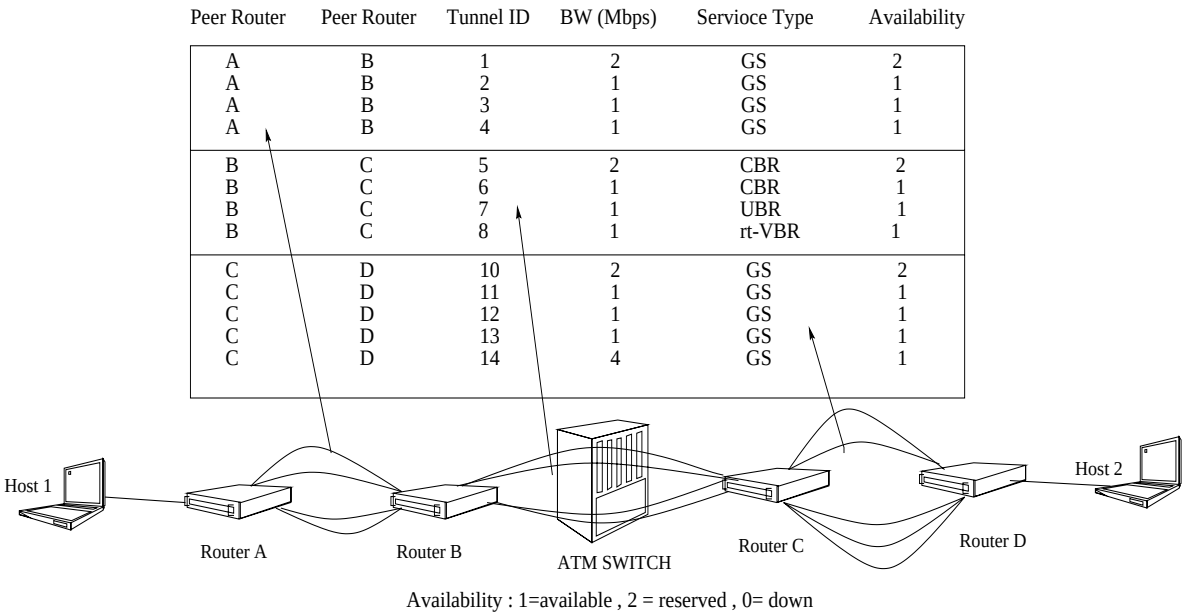


FIGURE 16. Diffserv to ATM Service Mapping.

3.7 Charging

Internet charging is a complicated research issue and VPN pricing is not different from that. In our implementation we have provided a method to compute the price of a VPN which considers the bandwidth of the tunnel being used 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 bandwidth 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.

	Price Per Minute in cents							
Tunnel ID	00:00 - 00:59	01:00 - 01:59	02:00 - 02:59.....	06:00 - 06:59	07:00 - 07:59	08:00 - 08:59	09:00 - 09:59.....	23:00 - 23:59
4	1	1	1	1	1	2	4	1
5	1	1	1	1	1	2	3	1
6	1	1	1	1	1	2	4	1
7	1	1	1	1	1	3	4	1

: : : : :

Our implementation requires that a service provider creates a pricing matrix similar to the one shown in above table. If a certain user has used a tunnel of 2 Mbps from 6:10 a.m. to 9:50 a.m. and that tun-

nel happens to have the tunnel ID 6, then using the matrix given above the price can be calculated as: (1 * 10) + (1 * 60) + (2 * 60) + (4 * 50)= 3.90 SFr.

In brief, the complete charging system works as follows: if a new VPN request is accepted, 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 and added to the billing database with logout time stamp added to that entry. In fact, before being added to billing database the system invokes the pricing database to compute the price of a specific tunnel.Figure 17 clarifies this process. The user can, at any time, ask the system to query the most recent billing. This is shown in the next section.

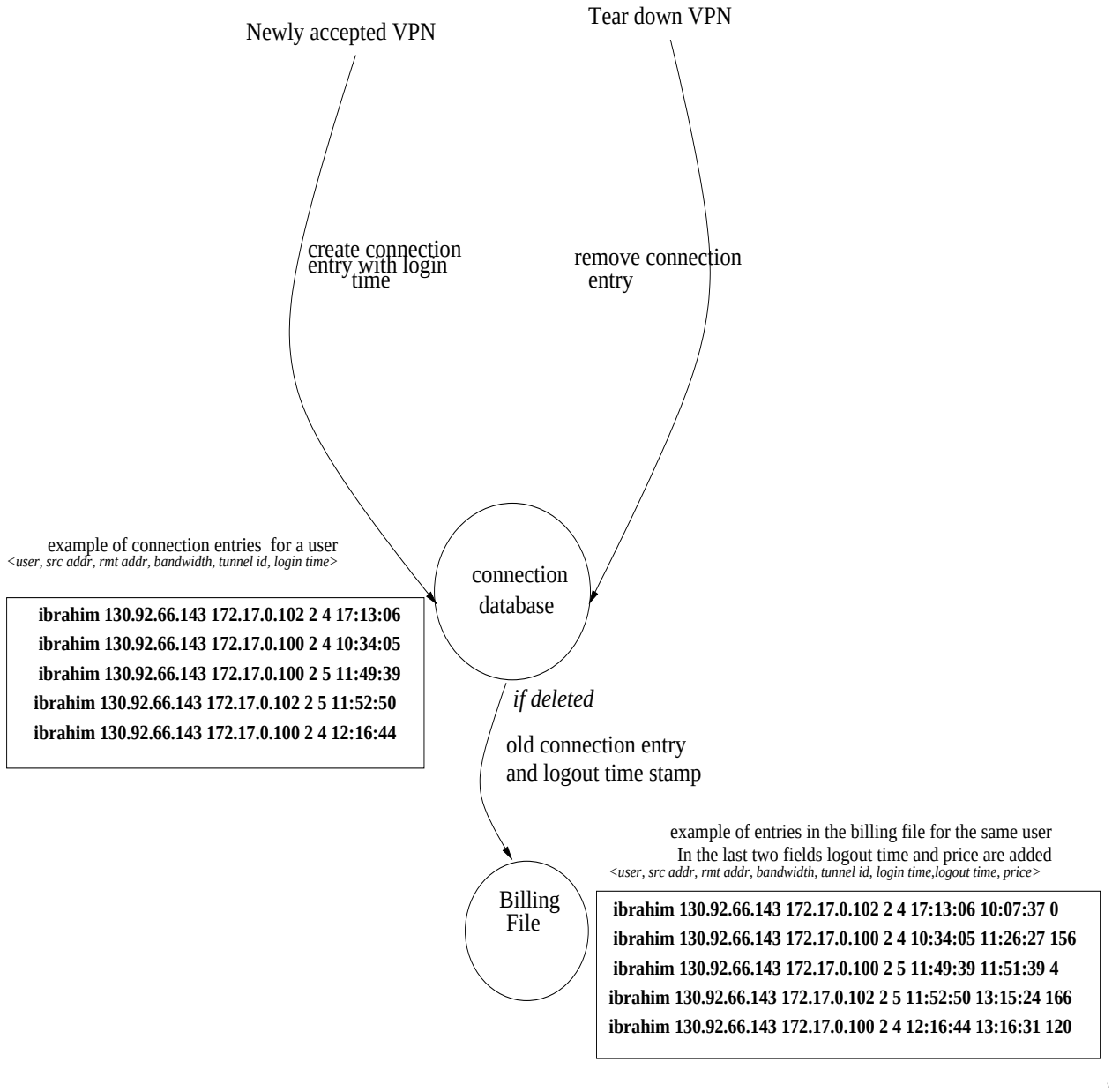


FIGURE 17. VPN cost calculation.

3.8 User Interfaces

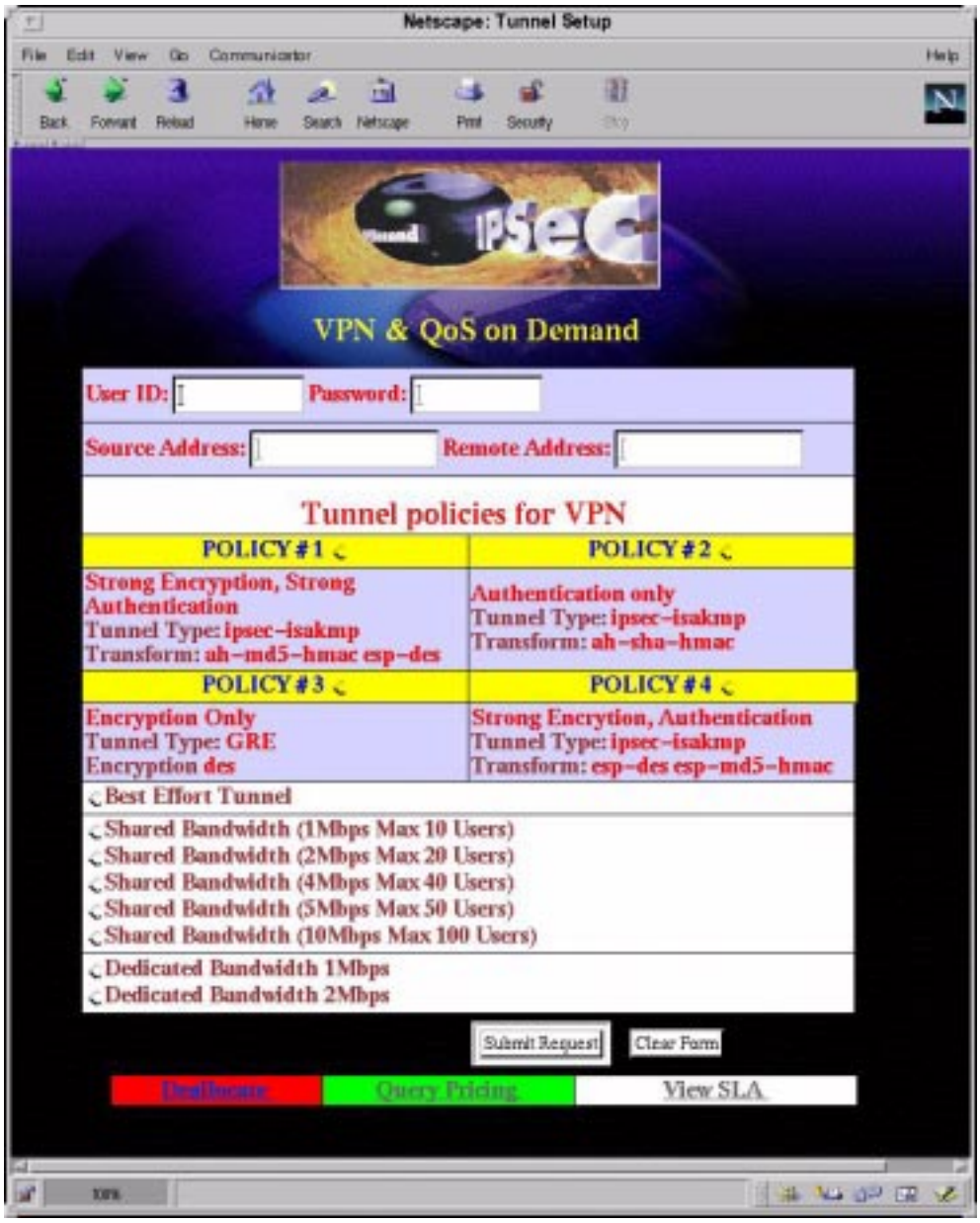


FIGURE 18. QoS VPN tunnels interface.

Our system provides several interfaces for enterprise and residential users and tries to keep them simple for the latter without compromising technical details needed by the others. This makes the system usable and meaningful to wider groups of users. Figure 18 shows the main VPN user interface which allows a user to identify himself, select any of the displayed security policies and available bandwidth options. In case of an error, e.g. a mistyped user id or password, the ISB sends a

feedback message to the user to make the system user friendly and interactive (Figure 19). However, it is not friendly to those groups of users who try to gain access to resources without having the right SLA, prohibiting not only intruders but also valid users trying to violate any previously established service contract.The system also lets the user query the most recent billing Figure 21.

FIGURE 19. Configuration error message.

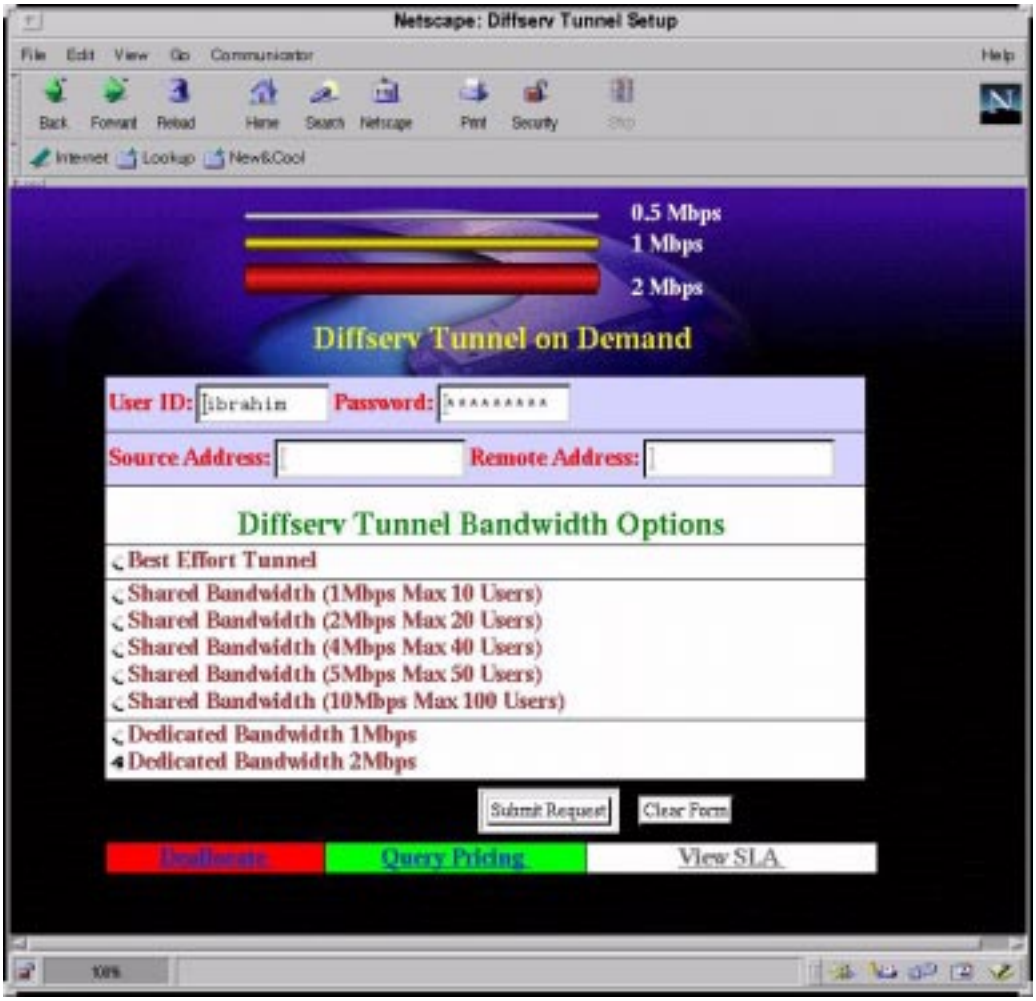


FIGURE 20. DS Tunnel on demand interface.

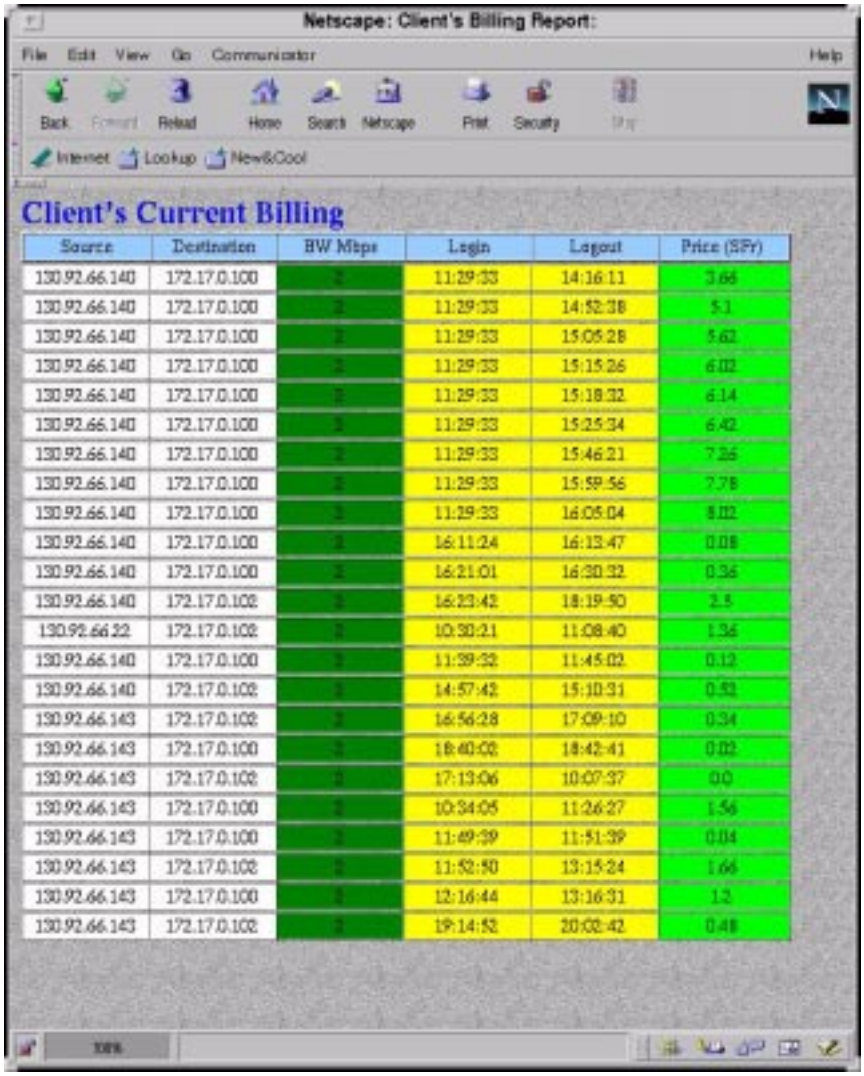


FIGURE 21. Result of a pricing query.

4. Enable IntServ over DiffServ

The deployment of the fine grained resource reservation mechanisms of IntServ and namely the Resource Reservation Setup Protocol RSVP [4] is steadily growing in the host networks. Unfortunately, as mentioned before, the IntServ mechanisms are not deployed in the core networks. There, the coarse grained DiffServ approach is more promising. The broker architecture we propose automates the establishment of DiffServ service level agreements. Therefore, the IntServ resource reservation signaling can be automatically mapped to the broker signaling for DiffServ. However, because of the different granularity of the reservation mechanisms, we need to *aggregate* several IntServ flows into one DiffServ reservation and class. Note, that VPNs represent traffic aggregations themselves, since they also classify several traffic flows into one indistinguishable class (an encrypted tunnel).

The IETF also identified the need and the possible benefit to combine IntServ and DiffServ. Two alternatives for interoperability between Intserv and Diffserv are mentioned in the resulting Internet draft[6]:

- Parallel Operation

- IntServ over DiffServ

The first option assumes to run IntServ and DiffServ independently from each other. Some flows such as real-time flows might get an IntServ service while others are supported by DiffServ mechanisms. This operation is simple but limits the use of RSVP /IntServ to a lower number of flows. In this mode, each node within the differentiated service network may also be an RSVP capable node.
The second approach assumes a model in which peripheral customer premises networks (CPN) are RSVP and Intserv aware. These are interconnected by differentiated service networks that appear as a single network to the RSVP nodes. Hosts attached to the peripheral Intserv networks signal to each other for per-flow resource requests across the differentiated service networks. Standard RSVP processing is applied within the Intserv peripheral networks. RSVP signaling messages are carried transparently through the differentiated service networks. Devices at the boundaries between the Intserv networks and the differentiated service networks (edge routers) process the RSVP messages and provide admission control based on the availability of appropriate resources within the differentiated service network [6]. This model is based on the availability of services within the differentiated service network. Multiple integrated services micro-flows which exist in peripheral networks are aggregated into the a behavior aggregate at the boundary of the diffserv network. When an RSVP request for an integrated service arrives at the boundary of a differentiated service network, RSVP style admission control is applied based on the amount of resources requested in the Intserv FlowSpec and the availability of DiffServ at the corresponding service level. If admission control succeeds, the originating host or the aggregating router marks traffic on the signaled microflow, for the appropriate differentiated service level. RSVP/Intserv over DiffServ is especially suitable for providing quantitative end-to-end services. The use of RSVP signaling provides admission control to the DiffServ network, based on resource availability and policy decisions.

4.1 Basic Scenario

Figure 22 shows the standard scenario with an ISP cloud in the middle and two access networks A and B being connected via an ingress and an egress router (edge routers). In the ingress router the RSVP flows have to be mapped to DiffServ Service classes and vice versa in the egress router.

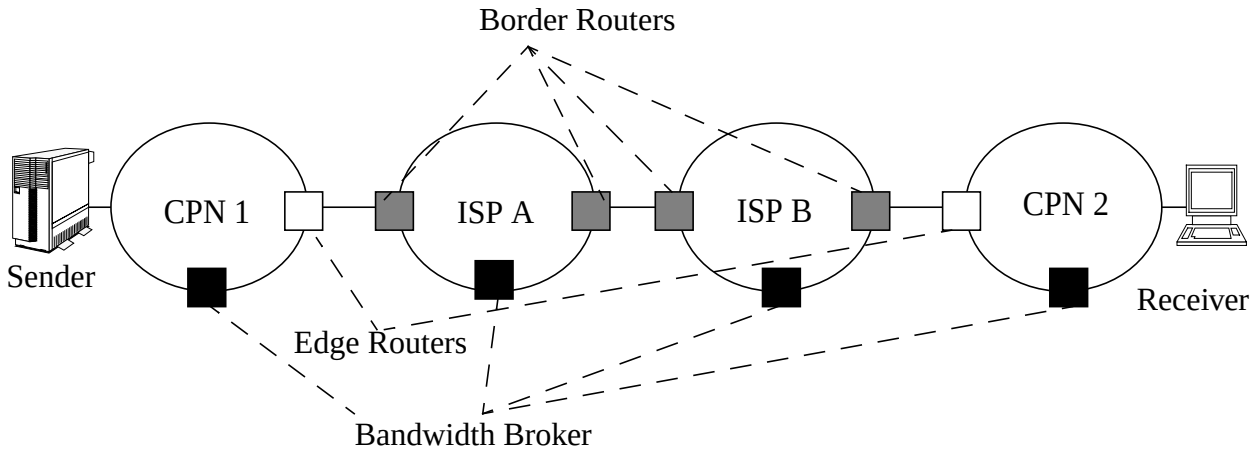


FIGURE 22. IntServ/DiffServ scenario.

This task of mapping can be split into two parts. The first one is the RSVP signaling, which is of course used for the resource reservation in the access networks and also for triggering resource reservation in the ISP.
The second one is the technique of aggregating flows and reserving bandwidth inside the ISP's network. We propose a central instance in the ISP a so called Bandwidth Broker (BB), which can be queried if there is bandwidth available and which supervises the ISP's resource management.
This document focuses on RSVP signalling and the communication between the RSVP components and the BB. Figure 23 shows the equipment and topology of the test and demonstration network.

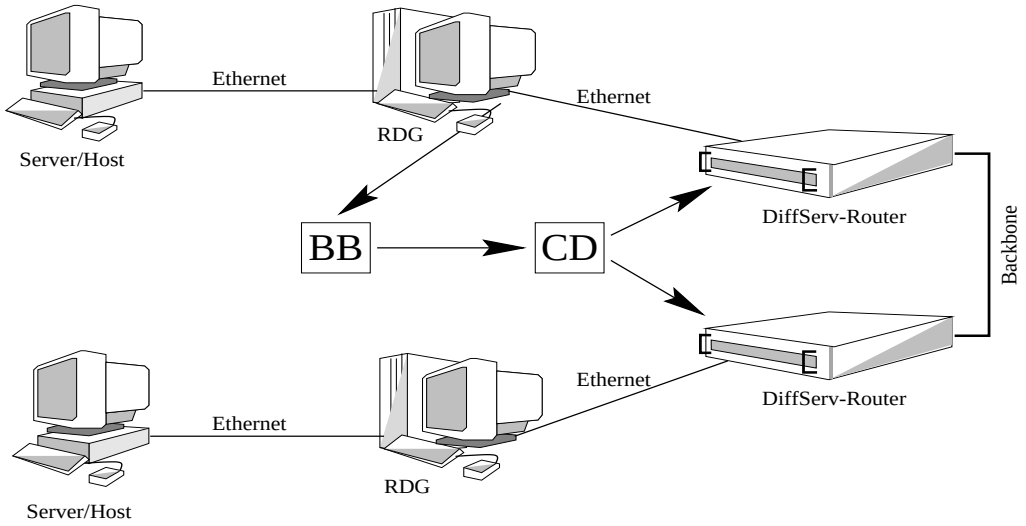


FIGURE 23. Test and Demonstration network with two DSR and two RDG routers.

The two DiffServ routers (DSR) are the ISP's border routers. Note that we assume in some scenarios that the DSR routers also run an RSVP daemon.[1] Because the DSR are capable to reserve bandwidth for specific flows and to aggregate several flows to DiffServ of VPN tunnels they are used for the resource reservation inside the ISP. The two RSVP DiffServ Gateway (RDG) (in our case Linux PC routers) at the edge fo the DiffServ domain have to keep track of RSVP signalling, the reservation of local resources and the interaction with the BB, which configures/monitors the DSR routers. The configuration daemons (CD) between the BB and DSRs shall be used as a adaption layer. So the BB can use some "platform independent router configuration language" to configure the DSRs. This shall allow the easy exchange of the router platform.
The RSVP DiffServ Gateway (RDG) directly connects to its BB. In reality the RDG may connect the BB of his local network, which then will negotiate with the ISPs BBs if necessary. Alternatively, the ISPs can also run their own RDGs. Then, it's these RDGs that contact the ISP's BB.

4.2 RSVP to DiffServ Concepts

In this section a short introduction to the Resource Reservation Protocol will be given. After that different concepts for the realization of the DiffServ mapping shall be presented and compared.
RSVP is used to negotiate and set up a resource reservation for a specific flow. So in every RSVP

1. Our commercial router (Cisco 7206) supports like many others RSVP and DiffServ to some extent, but cannot do the mapping between them.

capable router information about the flows have to be stored, leading to the above mentioned scalability problems. Figure 24 shows the setup of a resource reservation.

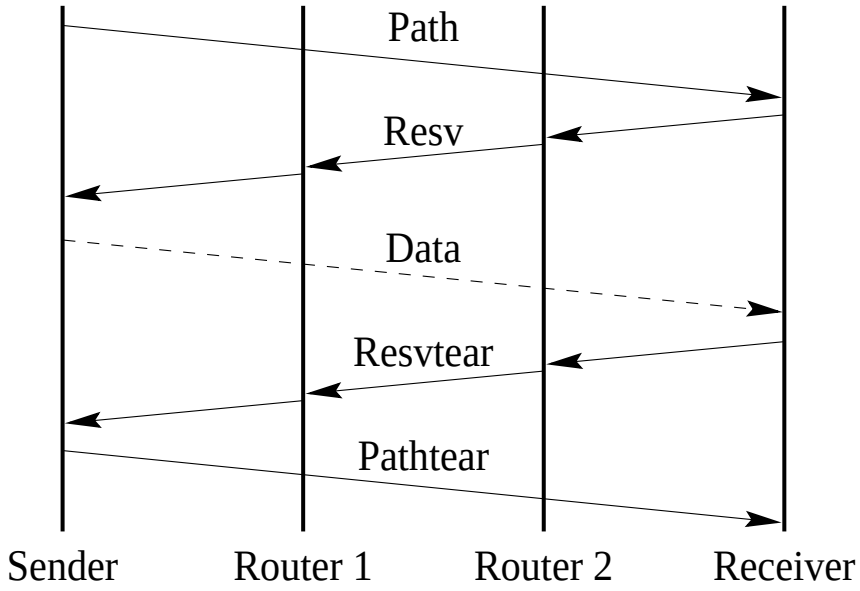


FIGURE 24. RSVP Resource Reservation.

First of all a PATH message is sent from the sender to the receiver. This message is primarily for the determination of the path and to check out in which router which amount of bandwidth or other resources have to be allocated and to initialize some flow state information in each router. After processing the PATH message, every router knows his neighbors for this flow. In a second step, the receiver sends an RESV message to the adjacent upstream router. If this is capable to meet the requirements it will forward the message to the next router and so on. If a router is not able to reserve the desired bandwidth it sends back an RESV_ERROR message. If an reservation shall be shut down, an RESV_TEAR is sent. Finally to delete all path states in every router on the path, a PATH_TEAR message is sent.
The ultimate goal is to avoid any RSVP resource reservation between the ISP's border routers. Our commercial routers (DSR) have some basic functionality to map RSVP to DiffServ tunnels, but they are missing any functionality to decide which requests are allowed and which methods of flow aggregation shall be applied. The information about the permissions of each flow are stored in the BB's SLA database. For each request the BB has to decide whether a request can be served or not. So there is a need for interaction between the RDG and the BB resp. the CD. In the following, four concepts of realizing this interaction are presented and compared.

4.3 Bandwidth Allocation Using the RSVP Path Message (Option A)

In the RSVP signaling (see Figure 24) the PATH message is the first step of the reservation. Since the PATH message contains information about the flow requirements it can be used to query the BB. The BB then decides whether this allocation can be done or not and will configure the DSRs accordingly. The aggregation of flows hereby is done by the RSVP to DiffServ capabilities of the DSR routers. Figure 25 shows the signalling.

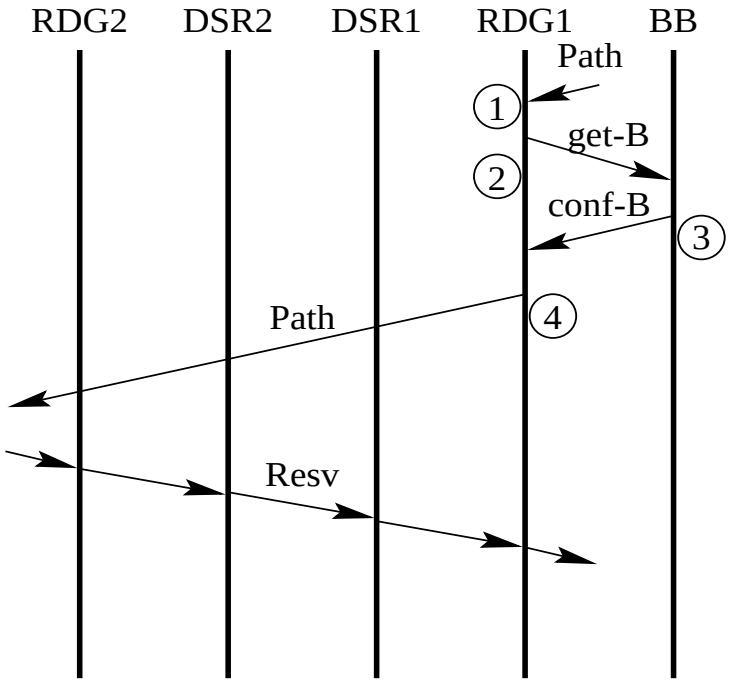


FIGURE 25. RDG Signaling using the PATH message.

1. The RDG1 receives the PATH message. This message contains information about the required resources.
2. RDG1 queries the BB, if there is bandwidth available (get-bandwidth message)
3. If the bandwidth can be allocated, the BB reconfigures the DSRs and sends an according confirm bandwidth message to the RDG. If not the DSR will answer the later RSVP-RESV message by an RESV-Error.
4. As soon as RDG1 receives the confirm message he forwards the stored PATH message.

Problems.

- The PATH message is forwarded using the Router Alert option. So a delay of the message is complicated and will cause timing problems.
- The amount of bandwidth specified in the PATH message, is not the amount of bandwidth the receiver wants to have (and is willing to pay for) but the amount of bandwidth the sender suggests.
- No control over RSVP to Diffserv mapping is done inside the DSR.
- Admission control is done by our DSR's RSVP daemon. So a detailed configuration or handling of specific flows is not possible.

4.4 DSR-Router Uses RSVP without Admission Control (Option B)

This mechanism doesn't use the PATH, but the RESV message for resource allocation inside the ISP's network. Figure 26 shows the signaling. Our DSR's RSVP daemons are configured to switch off their admission control. This may be done by allocating at the start-up of the DSR as much bandwidth to the RSVP daemons, as the DSR is able to transport. So the RDG router has to do admission control for the DSR. If an RESV message is received, the RDG router will ask the BB if this band-

width request is allowed. The BB has only to take care of not exceeding the maximum amount of configured RSVP traffic of the DSRs. Figure 26 shows the signalling.

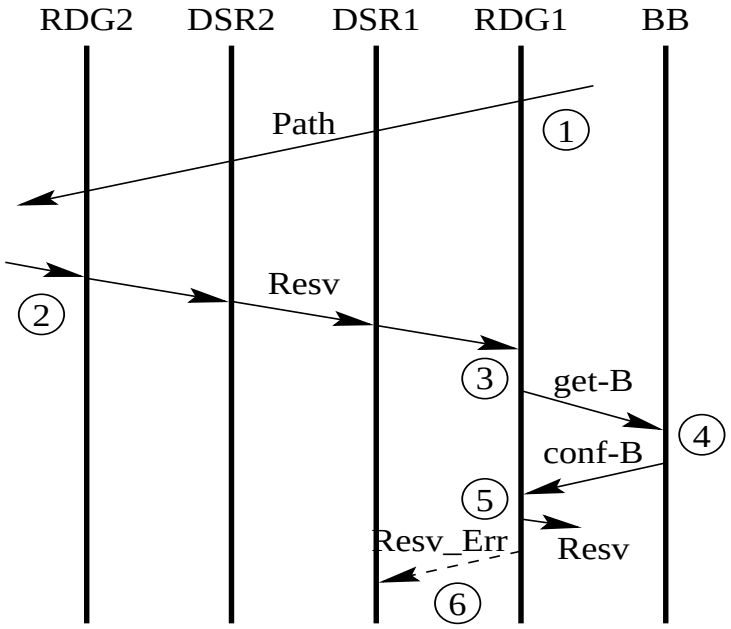


FIGURE 26. RSVP using our DSRs with RSVP, but without admission control.

1. The PATH message is forwarded through all routers. This is like standard RSVP.
2. The RESV message reaches RDG2 and is processed normally. The DSRs do the reservation as long as the maximum allowed amount of bandwidth is not exceeded.
3. RDG2 receives the RESV message and sends a `get-bandwidth`-message to the BB.
4. The BB checks whether this flow is allowed and if there is sufficient bandwidth available. He has not to contact the DSR to decide this. The BB sends back a positive or negative `confirm-bandwidth`-message.
5. If the result is positive the local reservation is made and the RESV message is forwarded normally.
6. If negative the reservation is denied and an RESV_ERROR message is returned.

Problems.

- The DSR-Router are configured statically. There is no synchronization between DSR and BB
- The RESV message is delayed.
- There is no control over RSVP to DiffServ mapping inside the DSR.

4.5 Usage of the RESV-Error Message to Adjust DSR Configuration (Option C)

The RSVP aggregation is done by the two DSR border routers. The (re-) configuration of the DSRs is triggered by an RESV_ERROR message. The idea behind this concept is, that the BB and the RDG only get active, when a bandwidth request of the DSRs fails. A DSR generates a RESV_ERROR message, because of it cannot reserve the desired bandwidth. The RESV-Error message is grabbed by a RDG router. This router then asks the BB, which --depending on SLA and bandwidth-- reconfigures the DSR. The main problem with this approach is, that after a denied

request a RSVP router is blocked for reservation equal or bigger than the one which causes the blocking. So after the RESV_ERROR message the path states in all router between have to be deleted requiring some new RSVP messages. So the RDG has to know the entry point to the ISPs network to delete the path states. Figure 27 shows the signalling.

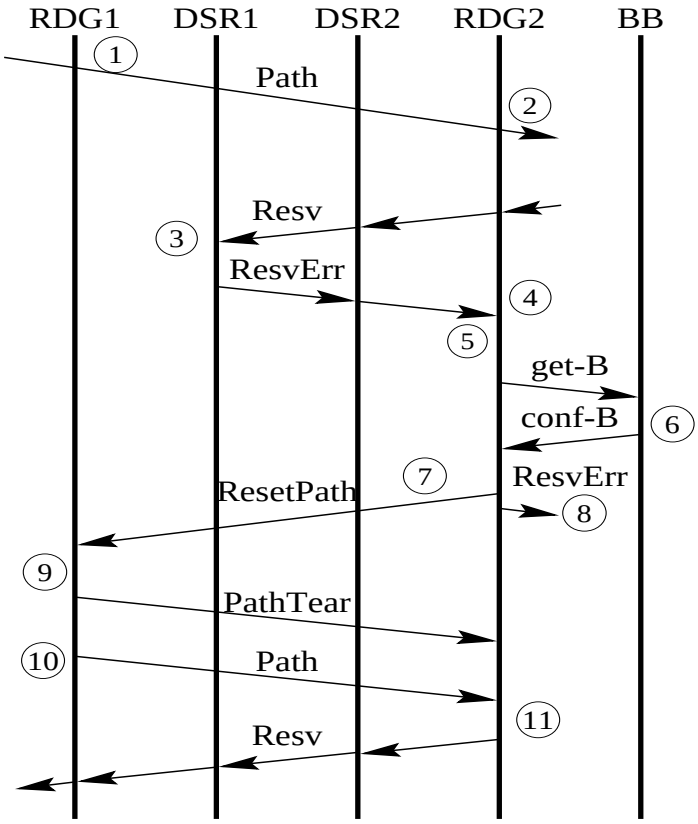


FIGURE 27. RSVP signaling using the RESV-Error message.

1. The Path message reaches the RDG router RDG1 and is forwarded.
2. The Path messages reaches RDG2 and is forwarded
3. The reservation fails at DSR1, because the desired bandwidth is not available. The DSR generates an RESV_ERROR message.
4. The RDG-Router (RDG2) receives the RESV_ERROR message
5. The RDG-Router (RDG2) blocks the RESV_ERROR message and starts up BB interaction to allocate -- if possible -- more bandwidth in the ISP.
6. The BB answers the request and reconfigures the DSR on an positive request.
7. If the conf-Bandwidth-Message was positive, RDG2 has to contact RDG1 to reset the path states.
8. The conf-Bandwidth-Message was negative. There is no reservation possible. The RESV-ERROR message is forwarded.
9. RDG1 receives the Reset-Path message to initialize the Path-States and creates a PATH_TEAR message to delete the path blocking states in the DSR routers. RDG1 adds a PATH erase Object to the PATH tear object.
10. A new PATH message is generated.
11. When the new PATH message is received at RDG2 the new RESV message is generated.

Problems.

- It is complicated!
- The deletion of the blocking state requires that a RDG router knows the RDG router on the other side of the ISP. So a new *CONFIG* object has to specified.
- To delete the path between RDG1 and RDG2 only an *ERASE* object is needed (limitation of the Path-Tear message).
- New fields in the path-state block to store the additional information (address of RDG routers, DSRs etc.)
- New message *Reset-Path*, to re-initialize the PATH message

4.6 Aggregation via RDG Routers and Bandwidth Broker (Option D)

The difference of this approach to all the others is, that RSVP is completely switched off inside the DSRs. The complete admission control is done by the RDG boxes. When a RDG Router receives an Resv-message it will query it's local queuing system, whether this reservation is possible and will also query the BB, whether the ISP's network is capable to do the reservation. The BB is also responsible for the configuration of the DSRs. The DSR router may aggregate the different RSVP traffic types to different tunnels. It may also be possible to allocate own tunnels to special flows or apply some encryption technology. Figure 28 shows the signaling.

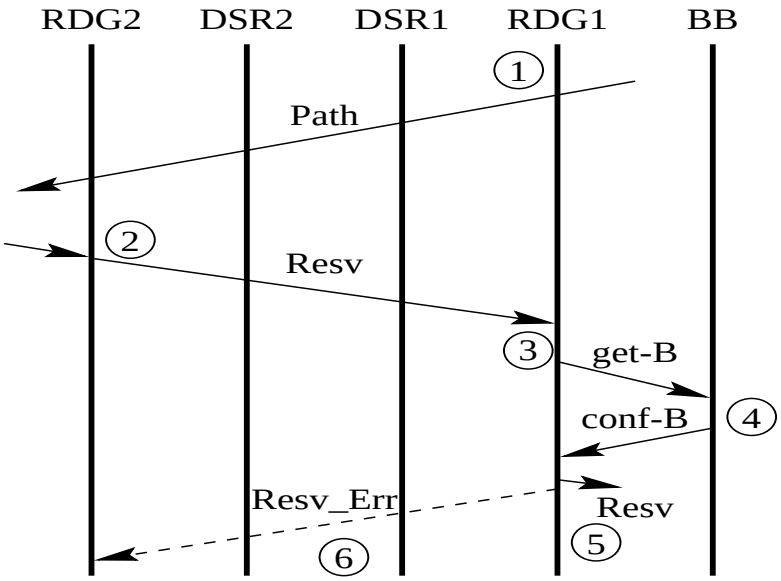


FIGURE 28. Signaling without RSVP in the DSRs.

1. Since RSVP is disabled, the PATH message is forwarded normally. The DSR routers will not react on the PATH message.
2. The Resv-message reaches RDG2 and is forwarded normally. (no RSVP in the DSRs)
3. The Resv-message reaches RDG1. RDG1 queries the BB if the reservation is allowed/possible.
4. The BB decides upon the reservation, configures the DSRs via the CD and sends an positive or negative confirm to RDG2.
5. If positive the reservation is accepted

6. If negative the RDG2 will send an RESV-Error message

Problems.

- Policing and flow control in the RDG routers required.

4.7 Comparison of the Different Concepts (Option A-D)

In the sections above four different concepts were presented. The following table gives a short overview over the different methods and their advantages respectively disadvantages.

TABLE 1. Comparison of Options.

	A	**B**	**C**	**D**
New RSVP Objects	no	no	yes	no
Full control over flow aggregation	no	no	no	yes
RSVP behavior changes	yes	no	no	no
Signaling overhead	small	small	big	small
Implementation Complexity	medium	small	big	small

4.8 Implementation

For the implementation the concept D presented in Section 4.6 was chosen. The advantage of this concept is, that the DiffServ cloud can be treated like a huge extension of RDG1's local queueing system. In the RSVP software we used as a basis for out implementation there is a nice interface between the queueing system (based on the programs/libs tc and ip [13]) and the daemon itself. So it is quite easy to put a third layer for the BB interaction between the daemon and the queueing system. So every time local resources are reserved, freed or modified also some routines are called, querying the Bandwidth Broker. By this, a reservation is only successful, when the local traffic control system and the BB agreed. Note, that we use the broker (QoS-VPN ISB) presented in Section 2.2 and described Section 3. as the bandwidth broker. A special interface allows the RDG to connect the broker and to use the reservation mechanisms. Thus, another advantage is the full transparency of the extension, so no RSVP user outside the ISB will have to change anything or even consider a RSVP to DiffServ mapping occurred.

5. Conclusion

The document presented two particular implementations of our broker based architectural framework for configurable network services. The focus is on the implementation of a quality-of-service enabled virtual private network management system. The system shows that the challenges of VPN service provisioning can be tackled. The VPN service is set up using the IPSec protocol, and QoS is described in terms of Differentiated Services. Other technologies such as GRE for VPN tunneling, a proprietary encryption scheme and ATM are used as well, which validates the generality of the architecture. The core of the implementation is a service broker providing the following functionality:

- The broker handles web based requests for QoS enabled connections and/or VPN tunnels with QoS.
- It configures its managed network equipment on-the-fly to ensure the QoS (by classification, scheduling, shaping) and the security support (by tunneling, encryption).
- The broker charges and accounts for the service. It does so by keeping an SLA database.

- The broker also offers an expert interface for the network administrator of the ISP.

Furthermore we presented design options of an implementation of a RSVP to DiffServ gateway (RDG). By using the gateway, fine grained IntServ resource reservations (RSVP) is translated transparently to the coarse grained service broker reservation mechanism (DiffServ). The RDG autonomously contacts the service broker upon specific reservations. Thus, end users using IntServ applications do not even have to contact the service broker any more to get appropriate QoS.

6. Acknowledgments

The work described in this document is part of a deliverable to the „Charging and Accounting Technologies for the Internet“ (CATI) [14] project funded by the Swiss National Science Foundation. We like to thank all colleagues from the other CATI project partners for discussions about the presented architecture. This work has been funded by the Swiss National Science Foundation (SNF), 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 "Förderung der wissenschaftlichen Forschung an der Universität Bern".

7. References

[1] T. Braun, and M. Günter: *Virtuell aber Real - Virtuelle Private Netze und deren Basistechnologien;* NET - Zeitschrift für Kommunikationsmanagement, Hütig Fachverlage, April, 1999.

[2] M. Günter, T. Braun, I. Khalil: *An Architecture for Managing QoS-enabled VPNs over the Internet;* to be published in LNC'99, 1999.

[3] St. Kent, R. Atkinson: *Security Architecture for the Internet Protocol;* RFC 2401, November, 1998.

[4] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin: *Resource Reservation Protocol (RSVP);* RFC 2205, September, 1997.

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

[6] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, K. Nichols and M. Speer: *A Framework for Use of RSVP with DiffServ Networks;* IETF Draft, November, 1998.

[7] K. Nichols, V. Jacobson and L.Zhang: *A Two-bit Differentiated Services Architecture for the Internet;* IETF Draft, November, 1998.

[8] F. Reichmeyer, L. Ong, A. Terzis, L. Zhang and R. Yavatka: A Two-Tier Resource Management Model for Differentiated Services Networks*;* IETF Draft, November, 1998.

[9] V. Jacobson, K. Nichols, and K. Poduri: *An Expedited Forwarding PHB;* RFC 2598t, June, 1999.

[10] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski: *Assured Forwarding PHB Group;* RFC 2597, June, 1999.

[11] K. Nichols, S. Blake, F. Baker, and D. Black: *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers;* RFC 2474, December, 1999.

[12] M. Günter, and T. Braun: *Evaluation of Bandwidth Broker Signaling;* Tech. Rep. IAM-99-002, May, 1999.

[13] W. Almensberger, J. H. Salim, A. Kuznetsov: *Differentiated Services on Linux;* IETF Internet Draft, June, 1999.

[14] B. Stiller, T. Braun, M. Günter, and B. Plattner: *The CATI Project: Charging and Accounting Technology for the Internet;* ECMAST'99, LNCS 1629, May 1999.