CATI
Charging and Accounting Technology for the Internet
SNF SPP Projects 5003-054559/1 and 5003-054560/1

Virtual Private Network Architecture

T. Braun, M. Günter, M. Kasumi, I. Khalil
Institut für Informatik und Angewandte Mathematik (IAM), Uni Bern
Neubrückstr. 10, CH-3012 Bern
Tel. +41 31 631 89 57 Fax +41 31 631 39 65
http://www.iam.unibe.ch/~rvs/

Workpackage and Task:	**WP2T1**
Deliverable Identifier:	**CATI-IAM-DE-I-000-1.1**
Status:	**Internal**
Version:	**1.1**
Revision:	**2**
Reviewed by:	**D. Billard, G. Fankhauser**
Due Date:	**11.1.1999**
Delivery Date:	**08.1.1999**

Abstract

This document describes an architecture how QoS-enabled virtual private networks over the Internet can be built and managed. The basic technologies for secure VPNs and for QoS support are introduced in the first chapter. The second chapter describes our vision of a QoS-enabled VPN service over the Internet. It also discusses in detail the required components and their interactions of an appropriate architecture. Based on this architecture, a demonstrator will be implemented. Chapter 3 presents the simplified implementation scenario and some implementation details in order to achieve secure and QoS-enabled VPNs.

1. Introduction of Basic Concepts

This chapter introduces the basic terms and concepts used in the rest of the report. We will define what kind of virtual private networks (VPN) exist and focus on VPNs for the Internet. We will motivate VPNs in the context of e-business and identify the disadvantages of todays VPN solutions, namely the lack of QoS support and the additional management burden.

1.1 IP-VPNs

Ferguson and Huston came up with a somewhat formal characterization of the VPN term:

A VPN is a communications environment in which access is controlled to permit peer connections only within a defined community of interest, and is constructed through some form of partitioning of a common underlying communications medium, where this underlying communications medium provides services to the network on a non-exclusive basis.

Ferguson and Huston also provided a simpler, more approximate, and much less formal description:

A VPN is a private network constructed within a public network infrastructure, such as the global Internet.

In the CATI working package 2 we want to focus on the case where the public network is the Internet. For this case the above definitions are missing one key concept namely tunneling. VPN solutions, more often than not, set up *tunnels* to treat the Internet as one hop between two friendly parties. The endpoints of a tunnel encapsulate a data packet into another one (possibly using a different protocol). Tunneling is a powerful technique used in many different areas, e.g. to route packets of one protocol through a network using another one or for mobile IP. Here is a compilation of the properties of the VPNs we want to focus on:

- The VPN uses the Internet as a public communications infrastructure.
- The VPN ensures privacy at the network layer. This means privacy is ensured per packet. The technical way to do so is by encapsulating IP packets into other IP packets (tunneling) and using cryptographic mechanisms to authenticate and encrypt the contents.
- VPNs partition the Internet by allowing the internal use of private addressing schemes (e.g. for Intranets).
- In contrast to current VPN implementations the VPN solution proposed by the CATI project will support quality-of-service (e.g. assured bandwidth), thus eliminating the only real disadvantage of VPNs compared to real private networks using leased lines. (see also 1.5)

1.2 VPN Business Scenarios

Most vendors of VPN solutions (e.g. IBM, Cisco, or Aventail) identified three usage scenarios that are all placed around a corporate Intranet that is securely connected to friendly 'entities' over the Internet. The scenarios differ in the entities connected: remote users, branch office networks or partners/suppliers. Note that only network layer VPNs are general enough to handle all three scenarios

- **Remote access network.** A remote user at home or on the road needs access to his/her company's electronic resources. An ideal VPN enables the remote user to work as if (s)he was at a workstation in the office. Authentication, transparency and ease of use for the remote user are crucial factors for this scenario.

- **Branch office connection network.** Here, two or more trusted Intranets are connected. Usually, the Intranets are protected by firewalls which are the ideal location to deploy VPN software. Thus, the client workstations do not have to worry about the VPN and the network manager can be sure that all Internet traffic exchange between the two Intranets is secured. Although this is the most simple scenario, problems arise from managing unregistered (private) IP addresses.
- **Business partner/supplier networks.** This scenario (also called Extranet) represents the most recent trend for VPN usage and also the least mature field. Companies can grant their partners temporal and limited access to their Intranet. The wide availability of the Internet and its relatively small costs together with mature IP-VPN technology shall allow fully functional e-business applications including initial contact of customer, sales negotiation, order fulfillment and on-going support. Furthermore, such a solution allows to automate the supply chain and facilitate collaborative projects with partners.

1.3 A VPN Enabling Technology: IPSec

IPSec [1] evolved from the IPv6 development and is short of being finalized by the IETF. It is an open architecture for IP-packet encryption and authentication, thus it is located in the network layer. IPSec adds additional headers/trailers to an IP packet and can encapsulate (tunnel) IP packets in new ones. There are three main functionalities of IPSec separated in three protocols. One is the authentication through an Authentication Header (AH) the other is the encryption through an Encapsulating Security Payload (ESP) and finally automated key management through the Internet Key Exchange (IKE) protocol. We will refer to these three mechanisms as IPSec protocols.

IKE (formerly called ISAKMP/Oakley) is the most complex IPSec protocol and is still under discussion. Nevertheless, IPSec provides an architecture for key management, encryption, authentication and tunneling. Therefore all of the previously defined VPN business scenarios can be implemented with IPSec.

1.4 Integrated and Differentiated Services for QoS Support

1.4.1 Integrated Services and RSVP

The RSVP (Resource Reservation Setup Protocol) [2] protocol is a main component of the Integrated Services (IntServ) architecture which is used to request QOS levels such as Controlled-Load (CL) or Guaranteed Service (GS) for individual flows. RSVP operates on top of IP e.g. in the transport layer, it is a control protocol comparable to ICMP (Internet Control Message Protocol) or IGMP (Internet Gateway Message Protocol). RSVP alone is not sufficient to provide QoS. RSVP only sets up reservations for network resources, but enforcement of the reservation needs to be done by other components of the Intserv mechanisms. RSVP provides the following features

- Receiver orientation
- Simplex reservations
- Soft state
- Tunnels of non-RSVP clouds
- Unicast and Multicast support
- RSVP relies on underlying routing mechanisms

Although RSVP has many valuable features, the protocol has also some problems that are partly being solved within the CATI project.

- Scalability

  Using RSVP per-flow states must be stored at each router along the path. This results in a large amount of information to be stored within routers. The information must be stored and processed in the intra-domain backbones. This could degrade the main function of routers and have harmful impacts for them.

- Bad support of short-lived flows

  Many flows occurring in the Internet are very short-lived, e.g. HTTP connections. Usually, it is not worth to reserve resources in advance of a flow since the time required for reservation exceeds the time needed for data transfer.

- Security

  Another concern are security related issues. It is possible to spoof a reservation request which will lead to theft of services by unauthorized users.

- Policy Control

  Policy control is also an area of weakness in the RSVP protocol. Policy control includes QOS access control, user authentication and accounting information. This problem is addressed by other subprojects of CATI and by the RSVP admission policy group (RAP) in the IETF.

1.4.2 Differentiated Services

The Differentiated Services (DiffServ, DS) [3] approach tries to provide a solution for QoS support with better scalability than IntServ. Differentiated services can provide two or more QOS level without maintaining per-flow state at every router. The idea of DiffServ approach is to use the DS field in the IP header to designate the appropriate DiffServ level that the packet should receive. DiffServ can provide scalability by aggregating the flows into a small number of DiffServ classes and by implementing traffic conditioning at the boundary routers of a network or an administrative domain. Here is a non conclusive list of differentiated service proposals:

Assured Service: The Assured Service does not provide absolute bandwidth guarantee but offers soft guarantee with high probability that traffic marked with high priority tagging will be transmitted with high probability. By the current Assured Forwarding PHB[1] group a provider can provide four independent AF classes where each class can have one of three drop precedence values. In future we might have more classes. It is, therefore, expected that Weighted Fair Queueing (WFQ) or Class based Queueing (CBQ) scheduler can provide the class based requirements.

Premium Service: The Premium Service is extremely low delay leased line like service where service level are specified as peak bit rate for a user or group of users (a flow or aggregation of flows). For time critical applications which require explicit bandwidth and minimum delay guarantee, premium service is the ideal solution. Various solutions exist to realize this service:

- A priority queue mechanism where each priority queue is regulated.
- Using Weighted Round Robin (WRR) in Weighted Fair Queueing (WFQ) where bandwidth allocated to the queue serving the premium packets is equal to the configured rate.
- Class based Queueing (CBQ) Scheduler

1. Per-Hop-Behavior (PHB) is the externally observable forwarding behavior applied at a DS-compliant node to a DS behavior aggregate. DS behavior aggregate is a collection of packets with the same DS codepoint crossing a link in a particular direction [RFC2745].

User-Share Differentiation Service (USD): With this service link shares are divided into various categories of traffic proportionally. Introduction of this kind of service in routers has several advantages.For example, unused bandwidth by one class can be shared among others using their alloted ratio. Implementation of USD service in routers would not require the users to specify traffic profile. To support these services routers would need a scheduler like CBQ, or WFQ type scheduler.

Addressing the scalability problem, DiffServ had gained much importance for its deployment in the IP-based enterprise networks. However, DiffServ is not able to provide QoS guarantees on an application level. In our view, it is therefore advisable to combine both QoS-enabling technologies in order to support scalable end-to-end QoS.

1.4.3 Interoperability between the Intserv and DiffServ Approach

Integrated Services and Differentiated Services are approaches complementary to each other. Therefore, it is advisable to combine both of them. According to [4], the three alternatives for interoperability between Intserv and DiffServ are:

1. Intserv over DiffServ
2. Aggregated Intserv states
   - RSVP aggregation by using tunnels
   - RSVP aggregation by using DiffServ-like mechanisms
3. Parallel Operation

In our view, the first option has the most potential for early deployment and for future developments. Therefore, we will outline only the Intserv over DiffServ model in more detail.

1.4.3.1 Intserv over DiffServ Scenario

Basic model. In the basic model DiffServ is used to allocate the network bandwidth in support of the Intserv networks which use RSVP end-to-end signaling. Figure 1 shows a network configuration where IntServ networks (stub networks) serve as customers of a DiffServ network (transit networks). The entire network consists of two Intserv capable stub networks which are interconnected by one large DiffServ capable transit network. In this model, end systems in the stub network use RSVP as a signaling protocol to request a specific QoS, while DiffServ is used in the transit network to support QoS with better scalability.

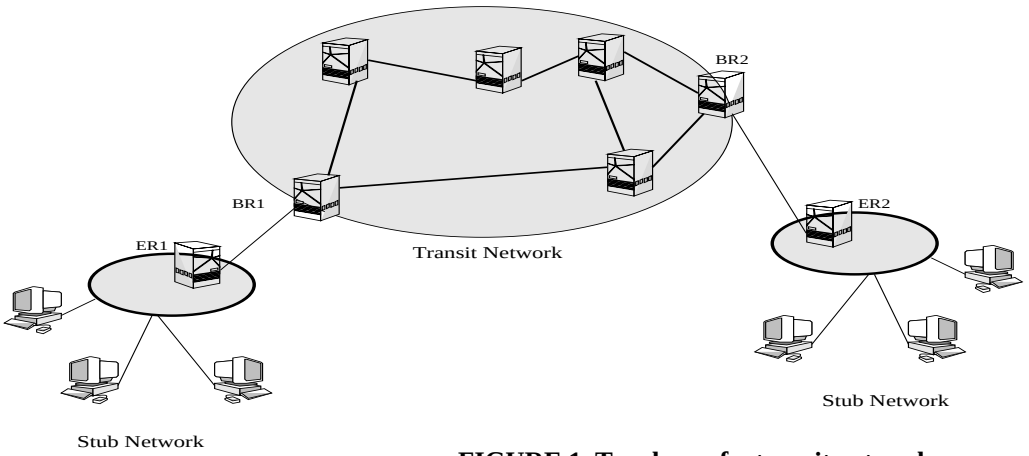

FIGURE 1. Topology of a transit network

Architecture. The IntServ over DiffServ architecture consists of the following elements:

- Hosts

  The hosts use RSVP signaling to request a specific QOS level. Some hosts are also able to provide traffic control functions e.g. DS marking of the packets and traffic shaping.

- Edge routers

  The edge routers are located at the boundary between the RSVP/Intserv network and the DiffServ network. These routers consist of two parts, one part is RSVP capable and interacts with the Intserv network and the other is DiffServ capable and interacts with the DiffServ admission control component to provide admission control feedback to the hosts generated RSVP signaling.

- Boundary Routers

  Boundary routers are located at the ingress and egress points of the DiffServ network. These routers provide traffic conditioning functions to ensure that the traffic conforms to the SLA (Service Level Agreement) negotiated between IntServ (customer) and DiffServ (ISP) networks. To implement traffic conditioning and to manage the resource allocation on the transit network, the boundary routers communicate with so-called bandwidth brokers (BB).

- Stub networks

  The stub networks are the sender's or the receiver's local access network. These stub networks contain Intserv capable hosts and a mesh of leaf routers which are not explicitly required to be Intserv capable. Leaf routers which are not Intserv capable act as a non-RSVP clouds. The stub networks may also use DiffServ mechanisms such as BBs for providing QoS to the end users.

- Transit network

  The transit network can provide different QoS levels by applying appropriate per-hop-behaviors (PHBs). The transit network is not able for RSVP signaling, but it is able to carry the RSVP messages transparently. The transit network could consist of several autonomous administrative domains.

Mapping between Intserv and DiffServ. In an Intserv network the traffic is classified based on the flow specs (e.g IP sources and destination addresses, port number), while in an DiffServ network the classification is performed based on the contents of the DS byte. Two approaches could be used for the mapping at the Intserv/DiffServ boundaries:

- Default mapping
- Customer-specific mapping

According to the first approach, the Differentiated Services classes are standardized (e.g., Premium, Assured and Best-effort). Thus, the easiest mapping function would be as follows:

- Controlled Load - Assured
- Guaranteed - Premium

Service Level Agreements. A service level agreement is a contract negotiated between an ISP and its customers. One ISP can be a customer of another ISP. The SLA specifies a traffic profile, network behavior and payment/billing scenario. Note that the SLA can refer to aggregated flow identifiers such as address prefixes. The SLA can be negotiated in a number of ways, for example via a phone call, e-mail or preferably using BBs. To deliver the user data and to provide QoS through the entire network such as the Internet that consists of multiple administrative domains, a SLA would be required not only between the customer and the ISP, but also between ISPs. Thus, each domain negotiates with its adjacent domain a bilateral SLA specifying the volume and the type of traffic. The agreements can be pre-negotiated and static or they can be established dynamically. The static agreements are defined with the initiation of the service and they change quite infrequently. The negotiation of static agreements is done by human interaction. In this model the network resources can be provisioned based on the expected negotiated traffic. For example, an ISP network can be provi-

sioned by allocating 20% of its bandwidth on its border links with an stub network for the Premium traffic. The dynamic agreements may change more frequently based on the current users requirements. This allows a user to react on current requests from its users. Over reservation of DiffServ capacities could be reduced. The agreements can be negotiated in case of having the traffic which is more conducive to a „pay-per-usage“ model. One example of this traffic is IP telephony, where users can pay on „per-call“ basis. Negotiation of dynamic agreements requires an automated protocol between the BBs.

1.4.3.2 Bandwidth Brokers

Bandwidth Brokers (BBs) [5] are agents that are responsible for resource allocation and traffic control of an administrative domain as well as for maintaining bilateral agreements between neighbor domains. BBs have their own policy databases that specify which users can use the resources and how much they are allowed to use. Figure 2 illustrates an end-to-end communication using BBs. Between each network an bilateral agreement is established specifying the traffic profile each network can send/receive. The traffic profile includes service class, a rate, maximum burst and the time period when the service is required.

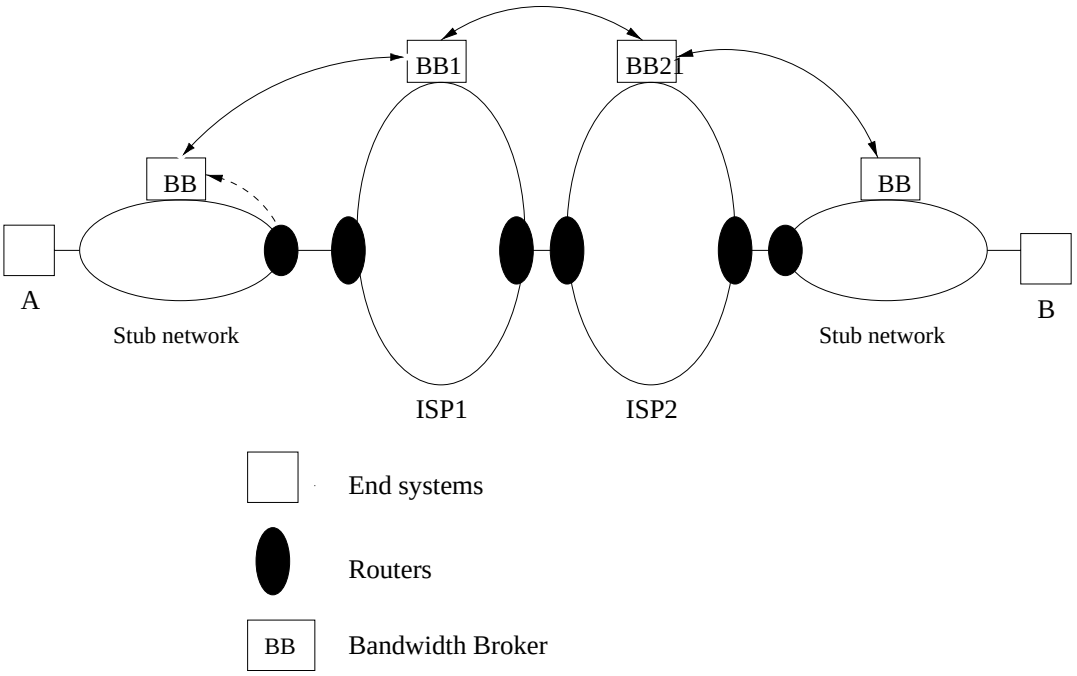

FIGURE 2. Networks with Bandwidth Brokers

The edge router in the stub network processes the individual RSVP messages generated by the hosts and forwards the information to the local BB. In turn, the local BB sends the message with aggregate flow requirements to BB1. When BB1 receives the request, it first authenticates the requester and then verifies if the available resources to the next domain (e.g. ISP2) are sufficient to serve the request. If the request is approved, it will set up an appropriate traffic profile in the border routers. In turn, BB2 and the local BB of the stub network will exchange the messages in order to reserve the resources for the aggregate flow between the domains. An inter-domain message exchanged between BBs could contain the following aggregate flow information [6]:

- ingress_address

  the address of ingress point where the aggregate flow enters the domain

- ingress_profile

  profile of aggregate flow (e.g rate, peak rate, burst size and PHB) entering the ingress point
- destination_address

  represents an egress point for some portion of the aggregated flow
- egress_profile

  contains information such as peak rate, burst size, rate for the portion of aggregate flow that will go to the destination_AS, because not all aggregated flows will be destined for the same AS.
- Number_of_destination_addresses

  provides the number of egress profiles for the specific ingress_profile.

In our view the BB approach has the greatest potential to solve several DiffServ problems such as dynamic SLA negotiation, and providing high service probabilities. However, the development of a BB protocol is required. Such a BB protocol could be based on extensions of existing protocols such as COPS, and RSVP. However, a full implementation, test and evaluation of such a BB protocol can hardly be achieved within the CATI project.

1.5 A New Valuable Service: QoS Enabled VPN Mechanism

A major competitor of IP-VPNs are private leased lines (Frame Relay, ISDN). While the leased lines are more expensive because the user has to pay even when not using the line, they usually come with guaranteed QoS. Enhancing todays VPN solutions with QoS will eliminate the VPNs only real disadvantage compared to the leased line solutions. External DiffServ seems to be the technology that is simple enough to enhance VPNs without restricting the global leachability of the Internet.

Another problem of VPNs is the relative high expertise necessary for managing a VPN. Mismanagement leads either to loss of privacy or loss of connectivity.

An Internet service provider (ISP) has control over a part of the Internet. Using emerging technologies it can offer QoS guarantees. It has the expertise and the financial resources to build up a VPN service, as well. Furthermore, in order to serve its customers and to lower costs the ISP must try to automate as much of its services as possible. Currently, the most promising technologies for such a service bundle are DiffServ and IPSec. In order for these technologies to work together it must be assured, that the tunnel endpoints and the ISP ingress nodes are located in the same machines. Else, the tunneling and encryption of the VPN service may hide information necessary for DiffServ. But given that an ISP provides the VPN and DiffServ together, it can place the tunnel endpoints accordingly. Although DiffServ and VPNs are two different services, they have similar concepts and can enhance each other:

- DiffServ can provide QoS commitments for a VPN as a whole or it can be used to differentiate the treatment of traffic classes within a VPN.
- VPNs are traffic aggregations with known traffic destination points. DiffServ also operates on traffic aggregations. The known destination points can furthermore ease the specification of necessary service level agreements.

- DiffServ and VPNs both need enhanced functionality of border routers of the ISP but not of intermediate routers. Both share some similar functionality in the border routers, e.g. the traffic classification.
- The simplicity and the coarse grained traffic classification make DiffServ a scalable technology. DiffServ is therefore suitable for the QoS support between different ISPs. On the other hand, a VPN tunnel that crosses intermediate ISPs is transparent to them and therefore does not allow fine grained QoS support.

The next chapter will raise the focus to a higher level of abstraction. VPNs and QoS support are presented as one of many services an Internet service provider (ISP) could offer. In order to do so effectively, the ISPs needs to collaborate with each other and automate their interaction processes. Here we present an architecture for doing so.

2. Vision of an Internet with Configurable Services

Todays Internet is made of the interconnection of multiple autonomous networks (domains). Some of these networks are driven by business companies which we will call Internet Service Providers (ISPs). The ISPs are eager to not only sell best effort transport services, but also to sell new and value added services. Such services can include QoS guarantees or privacy guarantees like for VPNs. The planing, accounting and the management of these services are further 'meta' services an ISP can provide. We can assume that the future Internet's ISPs have the following features:

- A network which is fully remotely controllable (configurable) by the ISP and whose hardware is able to support the new service.
- Intra domain resource management able to provide service guarantees. This can be a fine grained mechanism (e.g. per flow) if the domain is small.
- Application level interfaces for service requests by customers.

In order to produce the complete revenue of these investments, the ISPs need to collaborate. Today the collaboration is done by human network administrators communicating with each other by phone or fax. However, with automatically configurable networks and appropriate communication protocols, an automated approach is much more favorable. We envision the following requirements:

- Between each adjacent ISP there are electronic service level agreements (SLAs).
- Also, there is an inter domain resource management procedure which allows the new services to span multiple ISPs. In order to be scalable to large backbone networks, this management must handle aggregations of the intra domain resource management entities; it must be coarse grained.

In the rest of this chapter we describe an architecture that allows the automatic provision of new services spanning multiple ISPs based on the mentioned assumptions.

2.1 Goals for the Configurable Service Architecture

- The new services we will focus on are the support of QoS with differentiated services (DiffServ) and virtual private networks (VPNs). However, the architecture shall be service independent.
- The architecture shall not limit the ISP to a given business model.

- The architecture shall consist of generic components that are specialized for the given purpose.
- The architecture shall be open and interoperable in the sense that component interfaces are declared.
- The components interact with each other. The interactions that will be described in this document can be implemented using different existing/new protocols.
- The architecture shall allow for different implementations.
- The architecture shall focus on security issues (robustness and prevention of abuse).

2.2 Basic Components: Service Brokers

A service broker accepts service specifications for the specific service it sells. It can query the cost of the service and negotiate its price with a customer. Upon agreement, the broker can setup the service. The broker keeps the knowledge about the services provided in the form of service level agreements (SLA). It can thus accept requests to change the specifications of a service for a given customer. Note that the service broker may synchronize and coordinate multiple service requests. Furthermore, a service broker is an autonomous entity. It can proactively change an SLA. For such decisions, it needs access to various policy databases.

2.2.1 Broker classifications

Services can be classified as follows:

(1) (a) The service can be provided by ISP alone or
(b) it needs collaboration with other ISPs.
(2) (a) The service can be implemented by hardware configuration orthogonally to other services or
(b) must be coordinated with configuration of other services.

For the case of (2a) (orthogonal) we propose two kind of brokers. The Internal Service Broker (ISB) can be used to manage an ISP's service that is solely supported local to the ISP. The ISB can configure the ISP's network via secured connections to configuration daemons of each relevant network equipment (e.g. the border routers). The ISB manifests the fine grained resource management mentioned before. If the (orthogonal) service needs the collaboration of other ISPs we propose the External Service Broker component (ESB). The ESB has knowledge of the adjacent ISPs and can negotiate the necessary services of the adjacent ISP through its peer ESB broker. Therefore an ESB can control the corresponding ISBs and thus the network configurations. The ESB manifests the coarse grained resource management mentioned before.

Example: The bandwidth broker (BB) for differentiated services. Clearly, this is an external service broker. It is often neglected how the BB interacts with its network. In our case the BB would not only interact with other domains BBs but also with an internal service broker that would for example match DiffServ to ATM configurations.

Note that ESB and ISB represent a two class hierarchy which allows for a scalable solution. The separation matches the topology found in todays Internet. Figure 3 depicts the situation.

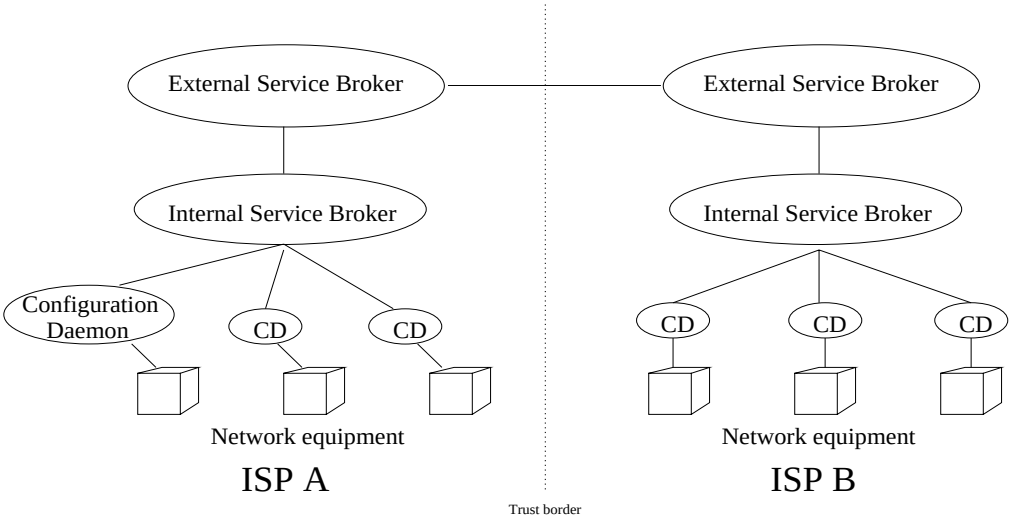

FIGURE 3. Broker Hierarchy

Class (2b) of services (non-orthogonal) must be handled by servers that are specially designed to offer a service combination. Such composite service servers (CoSS) can use and coordinate several ISBs and ESBs (of the different services influenced by this special service). Note that the management of such services is very complex. In general such a service can only be automated if the different 'sub'-services only interfere in very few specific areas.

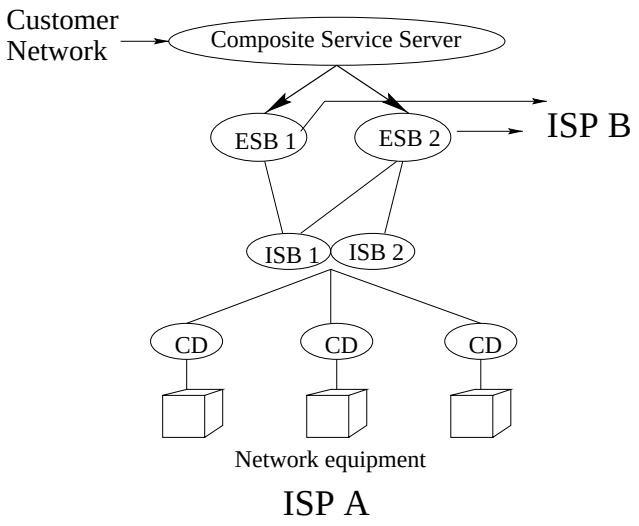

FIGURE 4. Combined Services

An example of such a service combination is the provision of VPNs with QoS guarantees with Diff-Serv. An example of such a service can be situated in the branch office scenario described before. The headquarter of a company is connected to its trusted ISP, the branch office to another ISP. The headquarter requests a VPN with QoS from its ISP using the QoS -VPN composite service server of the ISP. Figure 5 will depict the situation. The QoS-VPN server negotiates with the local ESBs that in their term negotiate with the next domain and so forth. Finally, the ISBs on the route between the

headquarter and the branch office will configure their network to support the requested QoS, and the ISBs of the border ISPs will configure their border routers for the VPN tunnel.

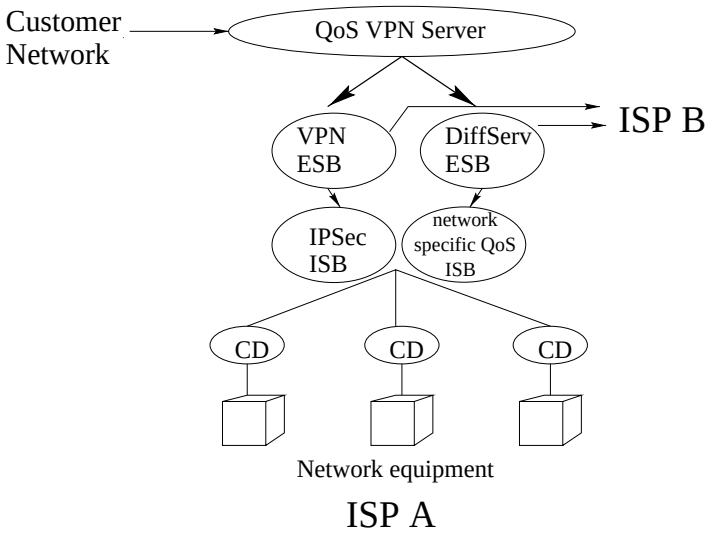

FIGURE 5. A QoS enhanced VPN service.

2.2.2 Customer Service Server

Conceptionally, there are only external service brokers interacting between two ISPs. At least in the case of a customer network it is desirable, that also a human can interact with a foreign service broker. We propose a simple GUI supporting service server to translate between a human and an ESB. We will explain this in more detail in the next section.

2.3 Interaction of Components

The interaction of components is implemented with communication protocols. In this architectural outline we will not limit the choice to a particular set of existing protocols. This is an implementation issue. Note however, that there is no complete protocol suite for this service broker architecture available today. We will describe the purpose of the different interactions between components by examples.

2.3.1 Configuration Daemon - Network Equipment Interaction

The configuration daemon (CD) is a configuration server specialized for a type of network equipment (e.g. a Cisco 7xxx Router). It accepts a machine independent, abstract configuration request and then starts to interact with the network equipment through a secured channel. It uses whatever authorization and configuration mechanism the network equipment requires to satisfy the configuration request. The CD notifies when its network equipment has installed the new configuration. It can keep a log of configurations done or even a complete backup configuration.

2.3.2 Configuration Daemon - Internal Service Broker Interaction

The Internal Service Broker (ISB) accepts requests concerning a service provided by the ISP's home network. It uses an abstract configuration language to send configuration requests to CDs and waits

for the correct establishment of the service at different places in the network. The configuration requests must be authenticated by the ISB. The ISB is responsible to guarantee that no two service requests that the ISB is handling interfere with each other. This can be complicated if an ISB is implemented in a distributed manner.

2.3.3 Internal Service Broker - External Service Broker Interaction

In order to handle external requests and agreeing on SLAs, an ESB must interact with corresponding ISBs. It must be able to query the state of current service configurations in the ISP's network. Furthermore the ISP can signal changes to these configurations. The ESB must wait for ISBs to react, and coordinate their effort to provide given services before it can complete its negotiation with peer ESBs of adjacent ISPs.

Note, that the brokers for each new service may use different protocols to interact. For example for bandwidth brokers RSVP, COPS and Diameter have been proposed.

2.3.4 ESB - ESB Interaction

This is the most complex of the interactions that we are going to discuss in this document. Like in the previously described interactions we need an authenticated and possibly private connection between the ESB peers. This is more complicated to establish since the peers are not in the same domain. Therefore, the peers do not trust each other. In the interaction between two brokers, one broker plays the role of a customer. The ESBs store and manipulate SLAs. This happens upon a service level negotiation between two ESBs. Along with the negotiation procedure the ESB-ESB protocol needs to include ways for payment. Note that the ESB also communicates with other entities such as a network administrator and various policy databases (e.g. for pricing, authentication, action triggering thresholds etc.).

2.3.5 Composite Service Server Interactions

The composite service servers (CoSS) can be seen as an extension of ESBs. Their structure and interaction patterns very much depend on what services they combine.

2.3.6 Customer - Customer Service Server

The customer service server (CSS) must be accessible by a well known protocol such as http. A customer contacting the CSS can chose a stub ESB for the desired service. With this graphically enhanced stub ESB the customer can negotiate a service level agreement. Note that the stub ESB has only reduced ESB functionalities, it cannot signal local ISBs or accepting another ESB as its customer.

2.3.7 An Interaction Example for Provisioning a New Service

A user learns about a service provided by an ISP through the WWW. She loads a GUI enhanced stub ESB (e.g. a Java Applet) which can invoke the appropriate ESB. The stub ESB lets the user negotiate with the ESB and visualizes the results[1]. Here is a generic sketch of the interaction taking place between the stub ESB (customer, controlled by a human) and the ISP's ESB (broker, automated):

1. These include e.g. a printout of the SLA negotiated, with service parameters and price lists.

1. Customer: Authenticated service request.
2. Broker: Answers a list of service forms. This list may include exemplars and partially prefilled forms.
3. Customer: chooses and requests a form.
4. Broker: sends form.
5. Customer: sends back the form now containing the parameters of the service requested.
6. Broker: internally checks the cost of the service (may also turn the request down rightaway). Broker answers the cost of the request. Along with the message the broker may suggest a payment method.
7. Customer: accepts or rejects. May also start renegotiation. Here, a variety of negotiation schemes can be implemented.
8. Customer: triggers the payment method and signals acknowledgment.
9. Broker: provides the service and acknowledges the customer upon establishment.

Note that the forms contain a unique identifier. This identifier must be kept by the customer in order to cancel the service contract later. The cancellation can be done with a single authenticated message to the broker.

This description of the interaction between components allows us to derive a more detailed picture of the components of the architecture.

2.4 Refinement of Components

2.4.1 ESB

The External Service Broker (see Figure 6) component bundles the most functionalities of all presented components. It is controllable by the network administrator via a master interface and it controls ISBs via a slave interface. The peer interface must implement the ESB-ESB protocol including form exchange, negotiations and electronic payment. The current state of the service collaborations between adjacent ISPs is stored in an SLA repository. The ESB can be equipped with a quite complex autonomous behavior because it should be able to automatically detect necessary updates to SLAs. Furthermore it should be able to react to changes in the other ISPs and it can vary its behavior depending on the network state and the e.g. the daytime. A central module of the ESB must coordinate its own activities, since there may be a lot of requests being processed at a given time.

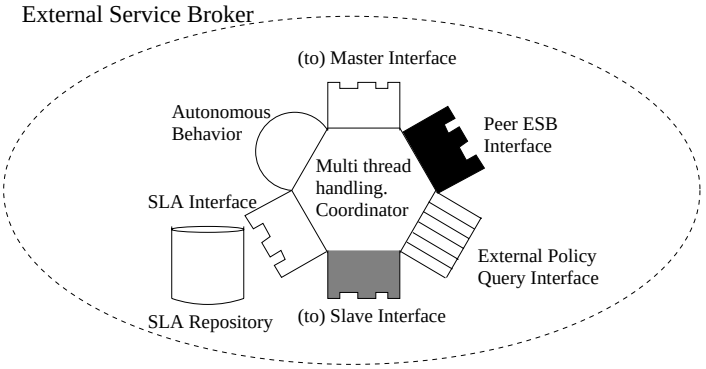

FIGURE 6. External Service Broker

2.4.2 ISB

The internal service broker's architecture is a simplified version of the one of the ESB (see Figure 7).This time, the master interface includes an interface for the network administrator as well as for local ESBs. The ISB coordinates the configuration of a service across its network. Therefore it needs an interface to control configuration daemons and a repository of the current configurations. An autonomous behavior module can be attached to an ISB that e.g. monitors a service relevant subset of the network equipment and triggers actions under certain conditions (e.g. alarms). The ISB needs coordination facilities as well.

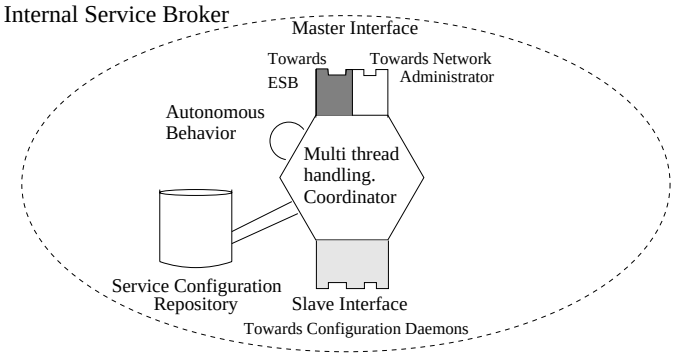

FIGURE 7. Internal Service Broker

2.4.3 Configuration Daemon

The configuration daemon (see Figure 8) is controlled by its master interface (usually by an ISB). It can log its configuration actions and may use this log for roll-backs of configurations. Only a simple coordination module is necessary unless the underlying network equipment is hard to configure online (which is not the case for e.g. Cisco routers). The CD can support a limited autonomous behavior for monitoring and notification of the network equipments state. The CD includes several modules for the support of different services (e.g. a tunnel establishment module or a DiffServ classifier configuration module). The CD uses a machine dependent interface to configure the network equip-

ment. If security would not be an issue here, the interface could use e.g. Telnet. Another possibility is to use the serial console port.

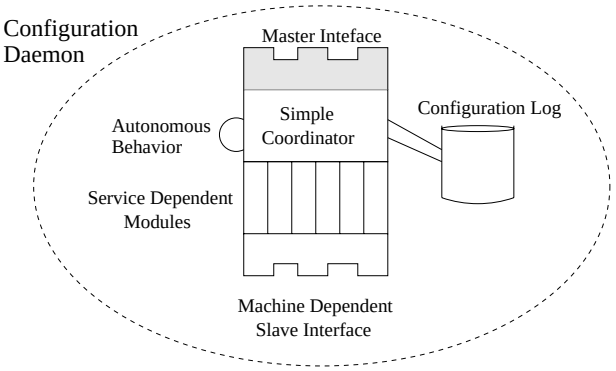

FIGURE 8. Configuration Daemon

2.4.4 Customer Stub ESB

As mentioned before the customer stub ESB (see Figure 9) is delivered to a human customer that wants to remotely configure a network service such as a VPN. Therefore, the stub has to be reasonably small. However, it needs to include an ESB peer interface. Note that this will be a significant piece of code since this interface implements the customer part of the ESB-ESB negotiation protocol. However policy and autonomous behavior modules of a normal ESB are not necessary since these functions are covered by the user. Additionally, the stub needs an interface towards a GUI. If the stub ESB is a Java applet, the GUI code will be provided by the web browser in use.

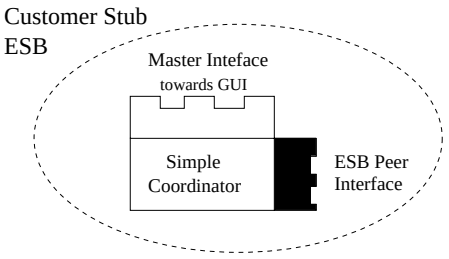

FIGURE 9. Customer Stub External Service Broker

2.5 Refinement of the Interactions

In this section we will present how the components exchange messages to accommodate service requests. The exact format of the messages is not specified in detail here. The messages are exchanged top down (ESB -> ISB -> CD) and bottom up (CD -> ISB -> ESB). There are six message types: queries, requests, commitments and cancellations go top down, replies and signals go bottom up.

2.5.1 Message Types

- **Queries**: A broker requests information from a peer or subordinate component about the service that the query target provides.
- **Request**: A broker announces that it will order a given service. Upon a request, resources for the service will be reserved, but the service will not yet be established. This is the first part of a two phase commitment. A request number will be assigned with which the announced service request can be identified.

- **Commitment**: The broker requests the establishment of the service under the given conditions. Commitments are only accepted after a successful request. The commitment may carry the payment for the service. The commitment carries the request number.
- **Cancellation**: A broker cancels an established service or a service request.
- **Reply**: This message type carries the reply of a peer or subordinate component, e.g. the results of a query or the acknowledgment of a request.
- **Signal**: A subordinate component sends information updates that were triggered by a third party (e.g. a network congestion).

The following two subsections will present the message exchanges between ISB and CD and an ESB - ESB message exchange. Note that the ESB-ISB message exchange is not presented, because in one possible implementation, the ESB and ISB of a service is running in the same process of a secured machine. In that case the interaction reduces to procedure calls.

2.5.2 ISB - CD Message Exchange

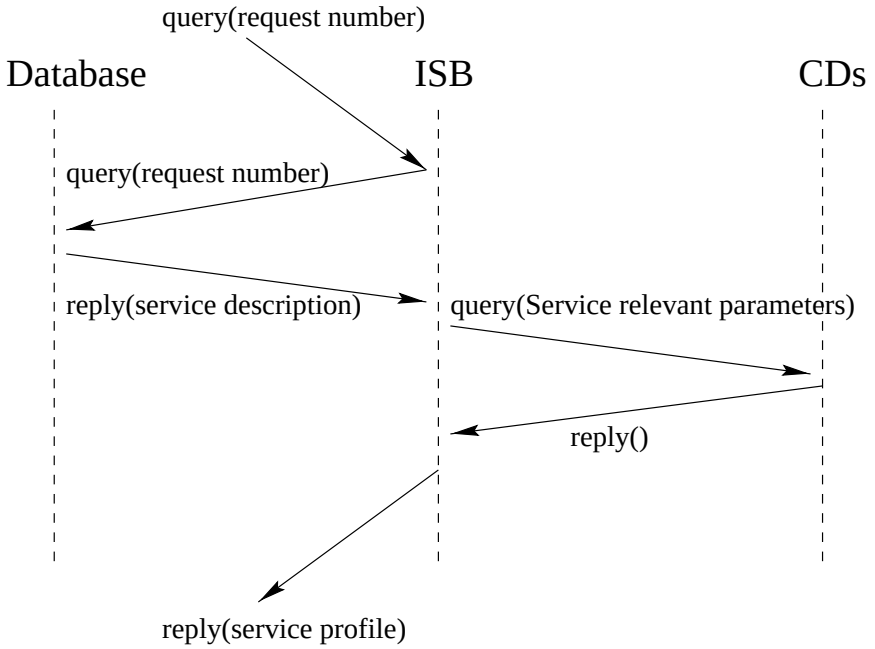

FIGURE 10. A service query

At any time, the ISB can be queried through its master interface by an administrator or an ESB. The configuration of a service consists of two phases. In the request phase, the ISB takes precaution that the service to be established is consistent with the already installed services. When the request phase terminates successfully, the service can be established by a commitment message. The two phases are depicted in the next two figures (Figure 11 and Figure 12).

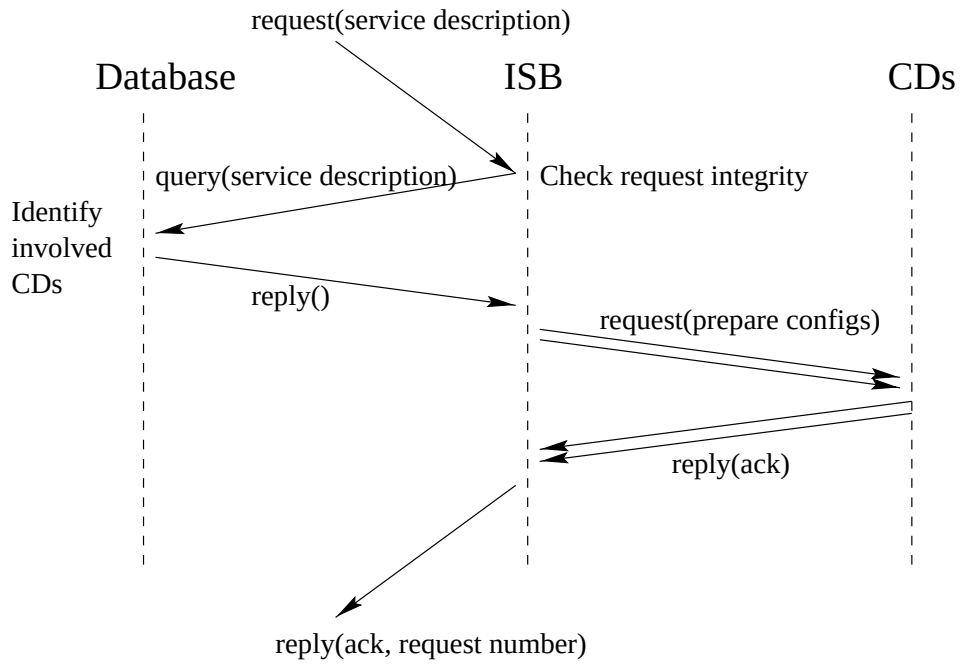

FIGURE 11. The successful request phase

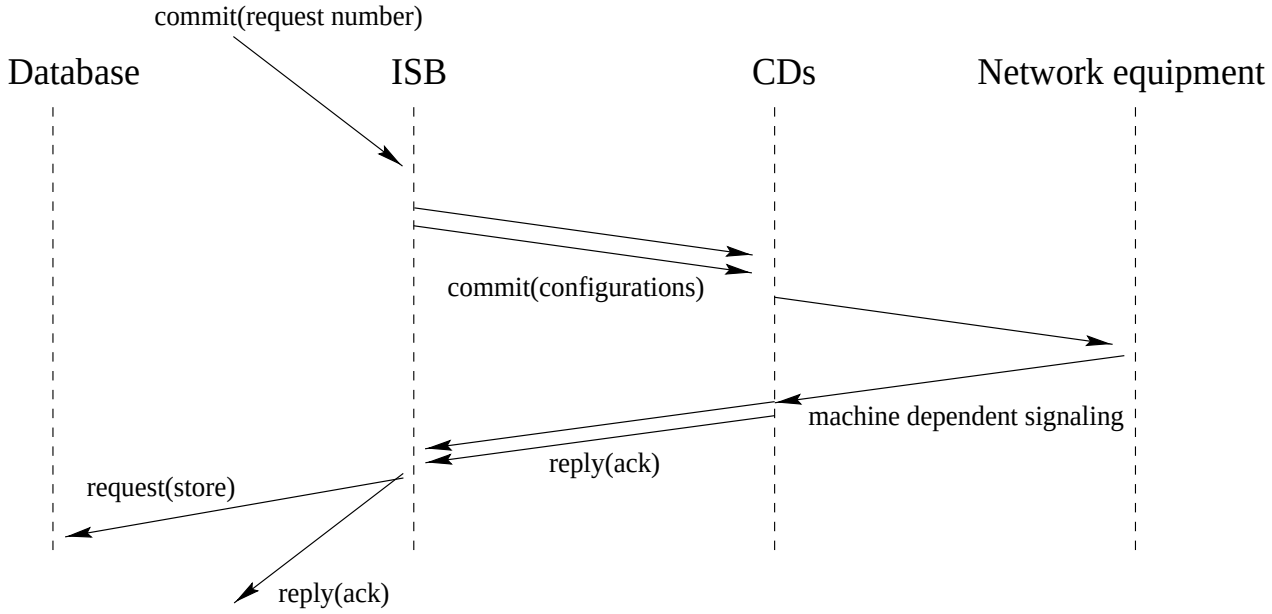

FIGURE 12. The successful commitment phase

2.5.3 ESB- ESB Message Exchange

The ESB-ESB message exchange is structured like the ISB - CD seen before. It consists of queries and request - commitment pairs. Here, each such message is relayed to the local ISB as well as to other ESBs if other ISPs must help to provide the service. Furthermore, reply messages can carry cost information, commitment messages can carry payment. The next figures shows the simplified

flow of messages for a query message and a request and a commitment message. Note that a cancellation message follows the same pattern as the request and the commitment message do.

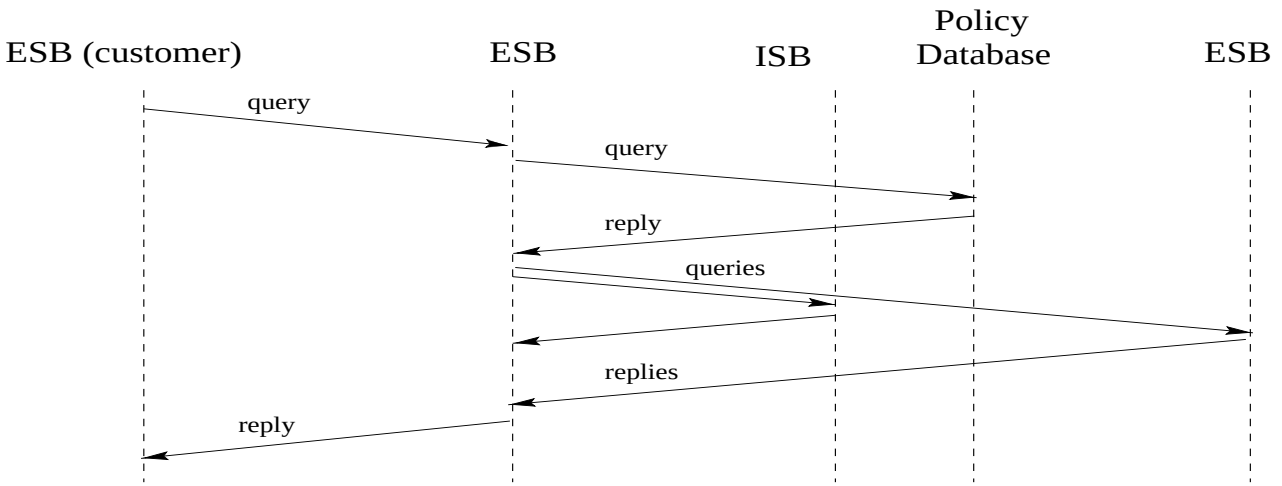

FIGURE 13. The successful ESB-ESB query

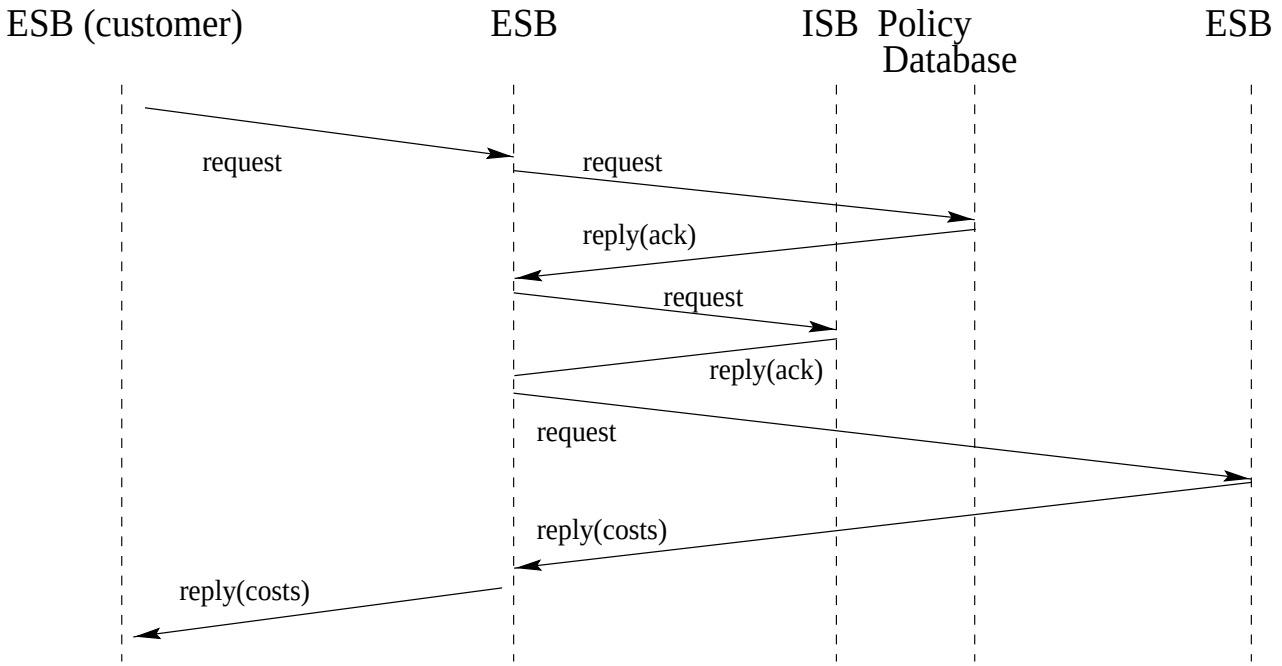

FIGURE 14. A successful ESB-ESB request

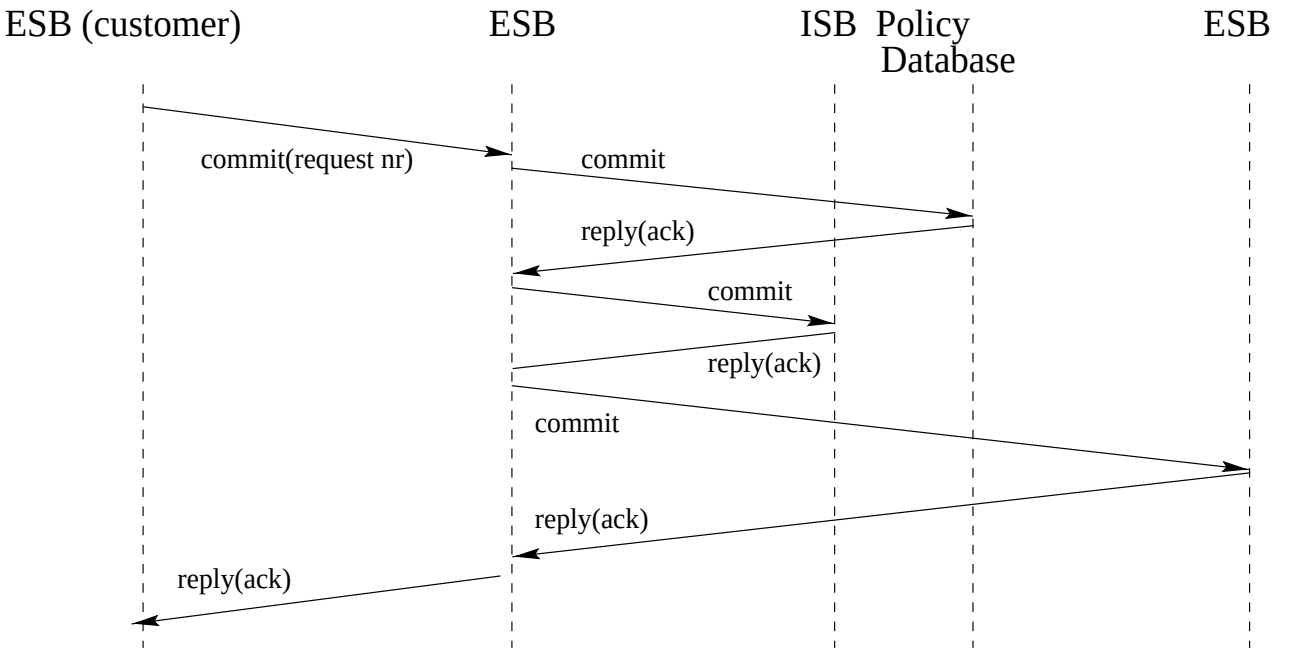

FIGURE 15. A successful ESB-ESB service establishment

As mentioned before, the composite service server (CoSS) is an extension of an ESB. The CoSS does not only communicate with adjacent ISP's ESBs but also with (at least two) local brokers. The interaction diagram for CoSS stays basically the same as for ESBs with the following difference: The Composite service server preprocesses (divides) queries, requests and commitments and relays them to the appropriate local brokers.

2.6 Charging and Accounting of Configurable Services

Our architecture does not restrict the choice of a payment method or pricing model to a given solution. In this section we discuss the access points for ISP individual charging and accounting modules. From a high-level view revenue is generated as follows. A customer (administrator of a stub network) needs a configurable service (e.g. wants to set up a branch office VPN). The customer is willing to pay for that service. This will be the money source. The customer contacts his ISP on-line via the ESB. It negotiates the service and the payment (price specs including payment method). In general, the deployment of the service will cost the customer a one time fee. In order to propose a price for the service, the ESB needs to check if other ISPs are involved and ask their ESBs what they would charge. This leads to a query chain that involves all participating ISPs and results in the distribution of the revenue. Finally the customer agrees on a SLA. From now on, the customer pays a flat- or usage based fee as negotiated in the SLA. The revenue is shared by the involved ISPs as described implicitly in the SLAs between the ISPs. The following subsections describe the different charging aspects.

2.6.1 One Time Charging

As mentioned before, the ESB - ESB signaling may not be free of charge. The query messages can be served for free or for a small charge. The request and the commitment messages must contain a payment or a promise for a payment. Cancellation of services may be subject to charging if this is legal.

An ISP in the role of a service seller must take into account, that it may have to use other ISPs services and pay for them, when it prices its service. Beside of that, it is free to define its pricing policy for ESB-ESB signaling. For example it might even reimburse costs to its customer when the customer decides to commit to the service. The ISP would then generate its revenues by continuous charging.

2.6.2 Continuous Charging

The price continuously charged for the provided configurable service is specified in the SLA. Thus the SLA record structure must be flexible enough to store a vast variety of payment schemes. It contains a usage and time of usage based price specification. Attributes can be bandwidth, peak traffic, total traffic, daytime and many others. The payment method must also be specified.

When an ISP classifies incoming traffic (e.g. for tunneling or DiffServ marking) it can also account it. Later, it can accumulate the continuous costs for a customer by using this accounting information and the appropriate SLA price specification.

2.6.3 Cost Calculations

In the case of a customer in the stub (access) network, a human will probably decide whether a SLA is acceptable. On the ISP side however the ESB must be able to automatically handle most of the negotiations. It should contact a human administrator only in few special cases. This is one of the big challenges for ESBs. The ISP is supposed to have its pricing models and policies available for the ESB. Upon a service request the ESB will contact other ESBs and also check with the local ISBs how much work and cost the request generates. Based on that it should be able to calculate a competitive price specification for the service. Such artificially intelligent behavior is out of the scope of the CATI project and subject of future research.

2.7 Security Issues

The higher we climb in the component hierarchy the more critical is the security of a component. A corrupt CD can only directly impact one machine. Once the corruption is detected, the reestablishment of correct service can be done relatively quickly. A corrupt ISB will affect the whole ISB network. A corrupt ESB will also directly affect neighbor ESBs. This case represents an extreme high threat potential, since the ESBs have the authority to handle payments automatically. The more ‘intelligence’ a component has the higher is the possible damage it can do in case of corruption. Malfunction of ESBs may only be detected in collaboration with other ISPs. It is difficult to reestablish correct service handling in that case.

While the interactions between components of the same ISP can be secured in a more statical manner with local key management, the ESB-ESB interaction needs to be secured using a trusted third party and/or public key cryptography. From what we said above we can draw the following conclusions:

- Not all components need the same level of protection. ESB must be protected using the highest level of security available. The interactions could be secured with an encryption algorithm allowing for long keys (>128bits). The key material must be refreshed automatically and in short terms.

- ESB - ISP interactions need also strong protection mechanisms but shorter keys with frequent key refreshment can suffice.
- ISP - CD interactions need protection but shorter keys and a less frequent key refreshment can suffice.
- CD - network equipment interactions may be protected using a dedicated cable or application layer security such as secure shells (ssh).

2.7.1 Security of Payment and Pricing Models

The trust model used here is the same as described in [7]. The ISPs do not trust each other. Therefore, as mentioned before, the proposed inter-component message exchange protocol must assure the integrity and authenticity of the messages exchanged. Then, customers can be held liable for the services they use.

We did not specify a payment method, thus the security of the chosen payment method must be evaluated separately. The one time charge for the request messages between ESBs is a special case. This message must be charged (a small to medium amount). If the request message was for free, the architecture would allow a malicious customer to launch a denial-of-service attack. The attacker could send many requests and cancel them again. This would cause much overhead in the service providers networks since the networks would be prepared for the provision of services that are then not deployed and not payed for.

2.8 VPN Service Configuration in the CATI Project

The configurable service architecture described here will be used as a framework for the VPN service configuration architecture and implementation for the CATI project. A full scale implementation of the architecture is beyond the available resources of the CATI project. Full scale implementation of frameworks with similar intend are currently under development of large companies (e.g. Cisco's customer network management framework [8]). Here are the components we are interested in for the CATI project:

- An internal service broker for VPNs.
- An internal service broker for QoS.
- A limited Composite Service Server that combines the services of the two previous components. The component is limited in the areas of autonomous behavior, and interaction capabilities with other ESBs.
- A limited stub ESB to talk to the CoSS.
- A customer service server for VPN services with QoS (delivering the stub ESB).

The next chapter will explain the CATI subset of the configurable Internet service model we presented in this chapter.

3. Implementation

This chapter describes the implementation scenario derived from the general scenario described in the previous chapter. It discusses which configuration steps a configuration daemon (CD) has to trigger for the demonstrator of a QoS enabled VPN service for the CATI project. We provide examples for the establishment of VPN tunnels and for hardware based QoS support.

3.1 Tunnel Establishment between two peers

The tunnel establishment module of CD establishes secure tunnels between two peers (e.g. routers). Any tunneling mechanism (IPSec or L2TP) can be used in our architecture, but we will assume that IPSec will be used in most cases since this is the recommended mechanism by IETF.

Using Cisco's IPsec implementation, traffic between two peers is protected by configuring access lists and enabling the access lists to interfaces by crypto map sets. Traffic selection can be based on source and destination addresses.

Crypto access lists are used to define which IP traffic will be protected by crypto and which traffic will not be protected by crypto. For example, access lists can be created to protect all IP traffic between subnet A and subnet Y or Telnet traffic between Host A and Host B. Referring to Figure 16, by issuing the command:

```
access-list 101 permit ip host 10.0.0.1 host 20.0.0.2
```

traffic from sender to receiver is protected. This is an example of access list.

Crypto map sets are accumulation of some crypto map entries in a group. Example #1 is an example of crypto map set and the entries in that example are crypto map entries. For example,

```
crypto transform-set someset1 ah-md5-hmac esp-des
```

defines how traffic encrypted and authenticated. This is an example of crypto map entry. When crypto map sets are applied to interfaces then all IP traffic passing through the interface is evaluated against the applied crypto map set. If a static crypto map entry sees outbound IP traffic that should be protected and the crypto map specifies the use of IKE, a security association is negotiated with the remote peer according to the parameters included in the crypto map entry.

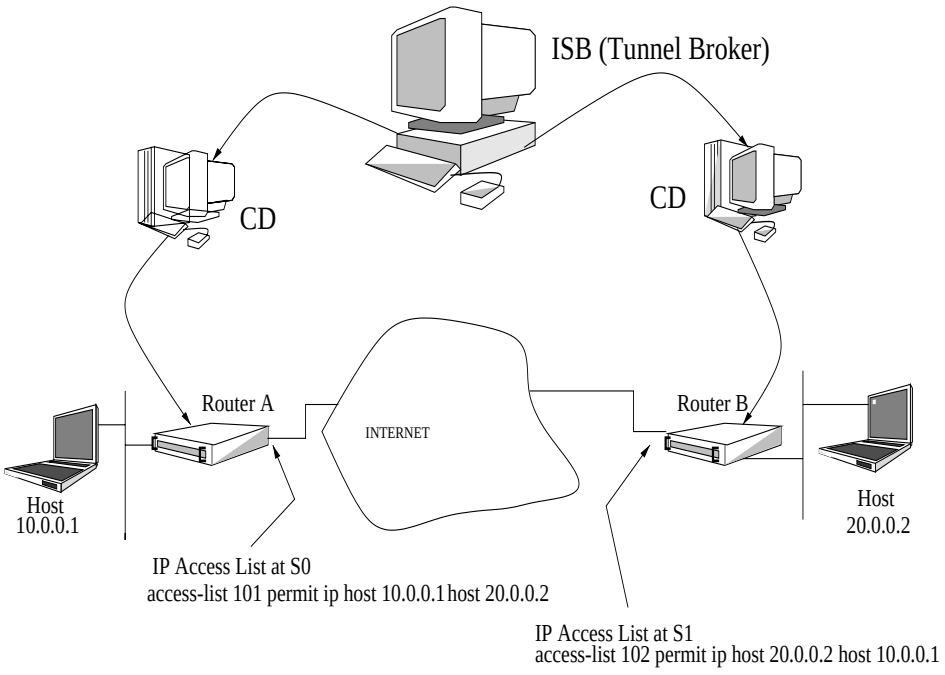

FIGURE 16. Tunnel Setup Scenario

In Figure 16, IPSec protection is applied to traffic between Host 10.0.0.1 and Host 20.0.0.2 as the data exits Router A's S0 interface enroute to Host 20.0.0.2. When ISB receives a request to establish a secure tunnel between two peers it sends that request to corresponding CDs which generate the required script to configure the connected peer. The example script that we have explained here, therefore, would be generated by the CD and then posted to the router to take necessary actions.

For traffic from Host 10.0.0.1 to Host 20.0.0.2, the access list entry on Router A is evaluated as follows:

source = host 10.0.0.1
dest = host 20.0.0.2

For traffic from Host 20.0.0.2 to Host 10.0.0.1, that same access list entry on Router A is evaluated as follows:

source = host 20.0.0.2
dest = host 10.0.0.1

After specifying which traffic should be protected a crypto map is also generated, for example, as follows

Example #1:

```
crypto transform-set someset1 ah-md5-hmac esp-des
crypto map mymap1 10 ipsec-manual
match address 102
set transform-set someset1
set peer 10.0.0.1
set session-key inbound ah 256 98765432109876549876543210987654
set session-key outbound ah 256 fedcbafedcbafedcfedcbafedcbafedc
set session-key inbound esp 256 cipher 0123456789012345
set session-key outbound esp 256 cipher abcdefabcdefabcd
```

Finally, crypto map set are applied to corresponding interface.

Example #2:

```
interface Serial0
crypto map mymap1
```

The above examples (#1and #2) are for only one peer. For IPSec to succeed between two IPSec peers, both peers' crypto map entries must contain compatible configuration statements. When two peers try to establish a security association, they must each have at least one crypto map entry that is compatible with one of the other peer's crypto map entries

For secure communication, we can use any encryption and hash algorithms available. For authentication, RSA is considered as preferred Public key technology. We can manually specify RSA public keys of two other Ipsec peers.

Example #3:

```
myrouter(config)# crypto key pubkey-chain rsa

myrouter(config-pubkey-chain)# addressed-key 10.0.0.1

myrouter(config-pubkey-key)# key-string

myrouter(config-pubkey)# 00302017 4A7D385B 1234EF29 335FC973
```

```
myrouter(config-pubkey)# 2DD50A37 C4F4B0FD 9DADE748 429618D5
myrouter(config-pubkey)# 18242BA3 2EDFBDD3 4296142A DDF7D3D8
myrouter(config-pubkey)# 08407685 2F2190A0 0B43F1BD 9A8A26DB
myrouter(config-pubkey)# 07953829 791FCDE9 A98420F0 6A82045B
myrouter(config-pubkey)# 90288A26 DBC64468 7789F76E EE21
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# addressed-key 20.0.0.2
myrouter(config-pubkey-key)# key-string
myrouter(config-pubkey)# 0738BC7A 2BC3E9F0 679B00FE 53987BCC
myrouter(config-pubkey)# 01030201 42DD06AF E228D24C 458AD228
myrouter(config-pubkey)# 58BB5DDD F4836401 2A2D7163 219F882E
myrouter(config-pubkey)# 64CE69D4 B583748A 241BED0F 6E7F2F16
myrouter(config-pubkey)# 0DE0986E DF02031F 4B0B0912 F68200C4
myrouter(config-pubkey)# C625C389 0BFF3321 A2598935 C1B1
myrouter(config-pubkey)# quit
myrouter(config-pubkey-key)# exit
myrouter(config-pubkey-chain)# exit
myrouter(config)#
```

In terms of the customizable service framework described in the previous chapter this example leads to the following interaction scenario: The CD and IPSec ISB is installed on workstations. Preliminary work has to be done for example a manual key exchange like described in example #3. When the broker and the daemons are running the broker can initiate the tunnel setup with a tunnel request. At this stage the CD can for example create the crypto map entries like in example #1. Upon the final ‘commit’ message (see previous chapter) the tunnel is configured like shown in example #2. Note, that Figure 16 and Figure 17 show the topology of the demonstrator intended to be implemented for CATI.

3.2 Providing Qos to the Tunnels

A VPN solution must guarantee QoS 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 tunnels. This ensures the performance of mission-critical and other high-priority applications without “starving out” lower priority applications.

In our architecture, various kinds of VPN tunnels can set up based on Qos requirements. For example, based on Intserv and DiffServ concept we can set up guaranteed tunnels, premium tunnels, assured tunnels etc. using various readily available QoS mechanisms like Weighted Fair Queueing (WFQ), RSVP, Random Early discard (RED) etc. It should be noted that the mechanisms should be enabled on the same tunnel interfaces in order to provide QoS to those tunnels (Figure 17). In the fol-

lowing, we briefly discuss these techniques:

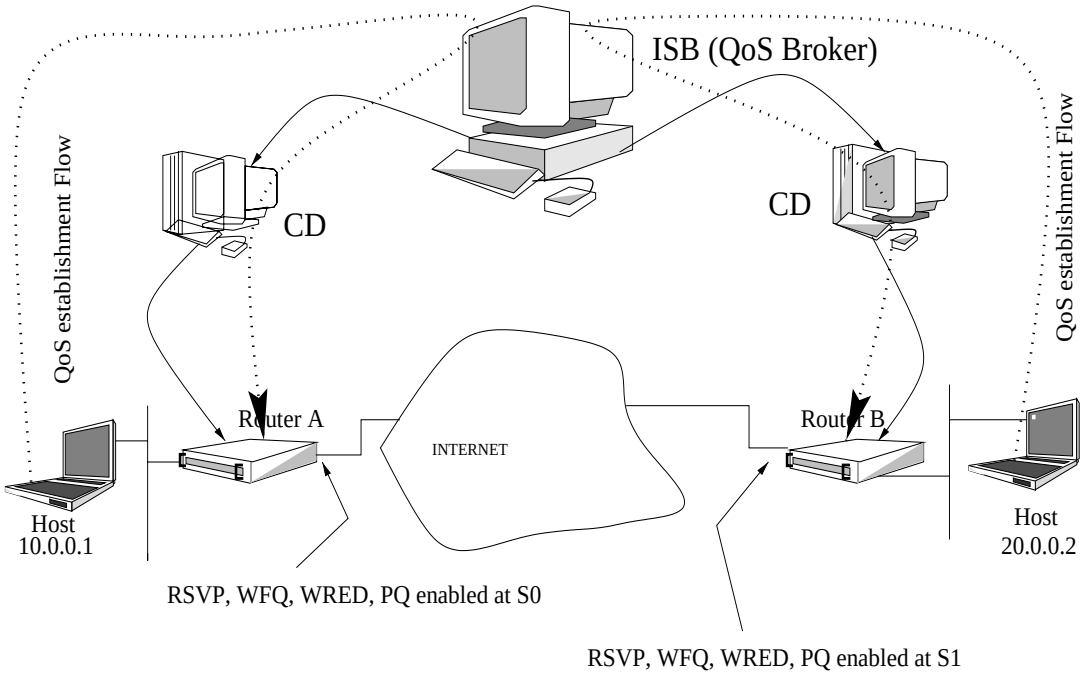

FIGURE 17. QoS setup scenario

3.2.1 *Absolute Guaranteed Service Tunnel based on RSVP*

Some tunnels would require absolute guaranteed bandwidth between any two (more peers). RSVP stands as the right candidate for that.

The following example shows a T1 (1536 kbps) link configured to permit RSVP reservation of up to 1158 kbps, but no more than 100 kbps for any given flow on Ethernet 0 and serial 0 interfaces. Fair queuing is configured with 15 reservable queues to support those reserved flows, should they be required.

Example #4:

```
interface Ethernet 0
ip rsvp bandwidth 1158 100
interface serial 0
fair-queue 64 256 15
```

3.2.2 Premium Service Tunnel

Premium service tunnels can be established by enabling priority queueing on the same Ipsec tunnel interface. In priority queueing, packets are first classified according to network protocol (ip, ipx, decnet, appletalk, vines, xns, x25 etc) or interfaces and then queued on one of the four queues - High, Medium, Normal and Low. The following example sets any packet type entering on Ethernet interface 0 to a high priority

Example #5:

```
priority-list 3 interface ethernet 0 high
```

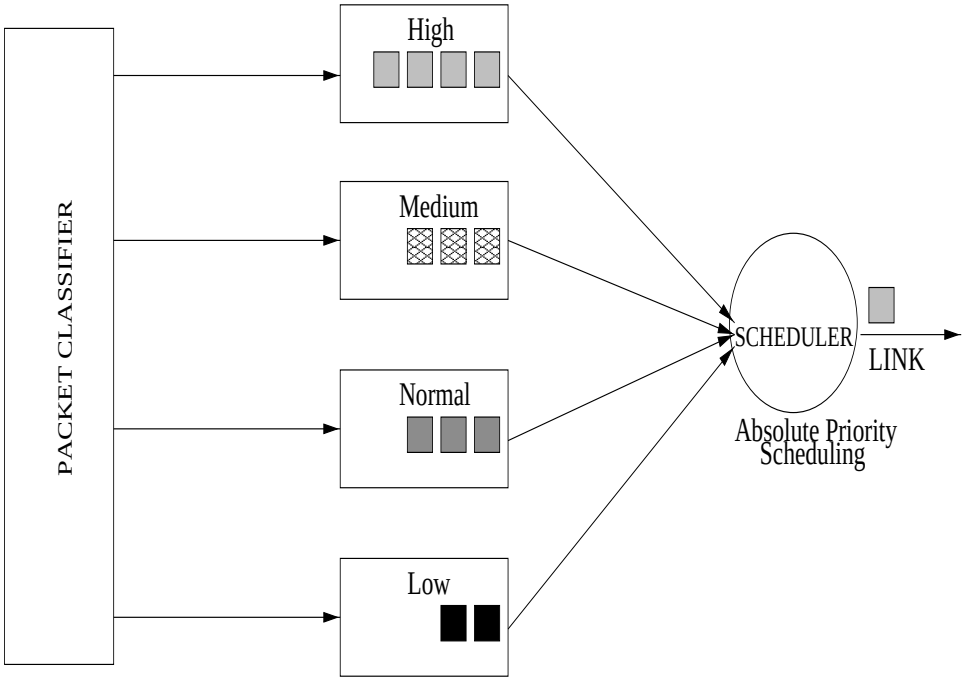

FIGURE 18. Priority Queueing

3.2.3 *Assured Service Tunnel*

Assured Service tunnel can be created by activating Class-based WFQ or Weighted RED or combination of both on the tunnel interface. If traffic originating from a tunnel (between two subnets) have various traffic classes where each traffic class is grouped based on ToS, then providing this kind assured service to tunneled traffic can be quite useful.

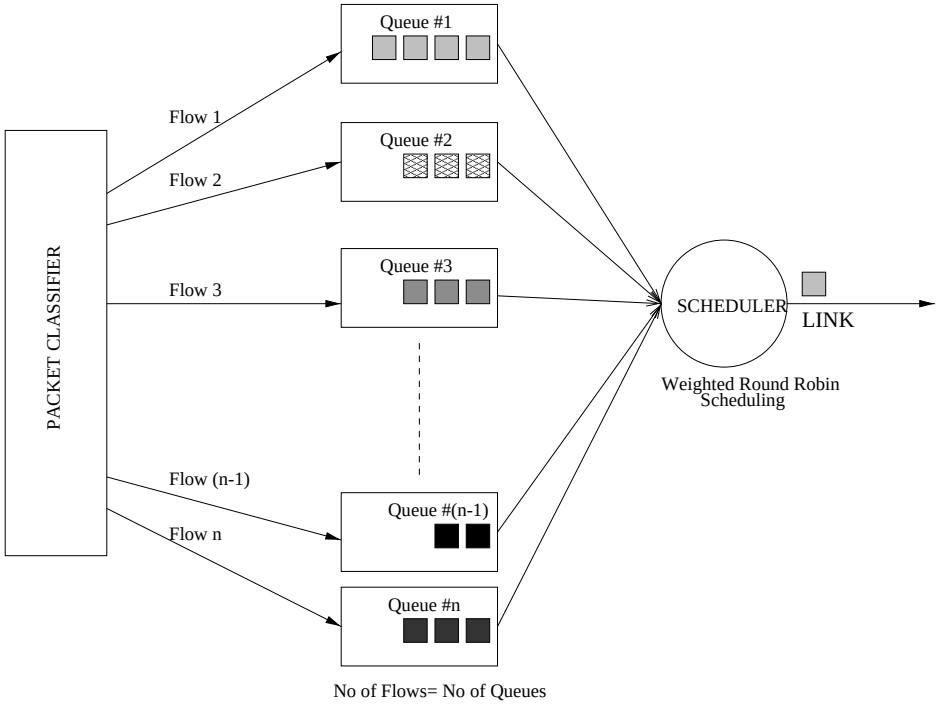

FIGURE 19. Weighted Fair Queueing

With class-based WFQ, packets are assigned to different queues based on their QoS group (an internal classification of packets used by the router to determine how packets are treated by certain QoS features) or the IP precedence bits in the DS (Type of Service, ToS) byte. Each class is assigned a weight and when congestion occurs, a percentage of the output bandwidth is allocated equal to the weight of that class. By using the following command for each ToS class (or QoS group), certain percentage of bandwidth can be allocated:

The following example configures Class (ToS) based WFQ where ToS class 1 is allocated 25% of the link bandwidth, ToS class 2 is allocated 35% of the bandwidth and it's queue depth is et to 27 packets, and so on. The remaining 5% bandwidth is allocated to ToS class 0. When the interface is not congested, queues can use any available bandwidth. If traffic originating from a tunnel have various traffic classes where each traffic class is grouped based on ToS, then providing this kind assured service to tunneled traffic can be quite useful.

Example #6:

```
Router # configure terminal

Router(config)# interface Hssi0/0/0

Router(config-if)# fair-queue tos 1 weight 25

Router(config-if)# fair-queue tos 1 limit 20

Router(config-if)# fair-queue tos 1 weight 35

Router(config-if)# ffair-queue tos 1 limit 27

Router(config-if)# fair-queue tos 1 weight 35
```

```
Router(config-if)# fair-queue tos 1 limit 20
```

3.2.4 USD Service Tunnel

The need for USD tunnels would arise when it will be required to allocate a certain portion of bandwidth to a group of tunnels. This kind of tunnels can also be set up by using the same WFQ method as described above or any hierarchical resource sharing method.

4. Conclusion

This report described an architecture that allows to implement QoS-enabled VPNs. The architecture is based on an generalization of the bandwidth broker concept introduced in the DiffServ environment. The architectural framework includes a service broker hierarchy that allows for automated service configuration. An instantiation of the framework allows a user to set up, change, and modify VPNs including parameters such as security and QoS related parameters. This architecture forms the basis of the implementation of a demonstrator scenario to be realized in the next phase of the CATI project. Since the presented general architecture is too complex to be implemented within the CATI project, a simplified demonstration scenario has been proposed that allows to demonstrate the principal concepts of the architecture.

5. References

[1] St. Kent, R. Atkinson: *Security Architecture for the Internet Protocol;* IETF Draft, July, 1998.

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

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

[4] 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.

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

[6] 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.

[7] N. Weiler, G. Joller, G. Fankhauser, B. Stiller: *CATI Trust Model;* Draft of Deliverable D 1.6.1, December, 1998.

[8] Cisco Systems: *Cisco Service Management System;* Whitepaper, http://www.cisco.com/warp/public/779/servpro/solutions/csm/csm_wp.htm, August 1998.