Management of Quality of Service Enabled VPNs

Torsten Braun, Manuel Guenter, and Ibrahim Khalil, University of Bern, Switzerland

ABSTRACT

New emerging IP services based on differentiated services and the IP security architecture offer the level of communication support that corporate Internet applications need nowadays. However, these services add an additional degree of complexity to IP networks which will require sophisticated management support. The management of enhanced IP services for their customers is thus an emerging important task for Internet service providers. This article describes a potential management architecture service providers will need for that task, considering problems such as multiprovider services and service automation. We will focus on a quality-enhanced virtual private network service which is particularly useful for corporate internetworking.

INTRODUCTION

The Internet's global presence makes it attractive as a universal communications infrastructure for businesses. With distance-independent rates and flat fees, the costs of corporate Internet communications become predictable and tend to get cheaper. However, some Internet design principles discourage the use of the Internet as a universal communication platform. First, all Internet traffic shares the available resources and is forwarded in a best-effort manner. Such resource sharing with all other Internet users makes it impossible for Internet service providers (ISPs)to offer the service guarantees that some corporate communication applications (e.g. IP telephony, videoconferencing, stock information services) need. In order to alleviate this problem, the Internet Engineering Task Force (IETF) defined quality of service (QoS) support mechanisms for the Internet (discussed later). The second problem with the Internet is its lack of built-in security support. IP traffic is easy to eavesdrop on, and IP packets can easily be manipulated (e.g., to appear to come from an arbitrary source address). Furthermore, the best-effort nature of the Internet invites denial-of-service attacks based on excessive generation of traffic. Such security threats fuel the trend to protect corporate network traffic on the Internet.

A widely used approach is a virtual private network (VPN). A VPN is a private network on a public network infrastructure (Internet). The VPN traffic is logically separated from other traffic by tunneling mechanisms (discussed later). Furthermore, the traffic content is often protected by cryptographic means. A particularly fruitful solution for corporate communications is a QoS-enabled Internet VPN. Such a solution takes advantage of the cheap and ubiquitous Internet, but provides quality and security guarantees at the same time. However, QoS and VPN techniques introduce new challenges. They need extensive configurations in the routers. The local configurations have to be consistent across the network. Many companies may not have the knowledge and resources to deploy and manage enhanced Internet services by themselves. Rather, they will outsource the service management to their Internet service provider. This is a win-win scenario because the provider can profit from the economy of scale by sharing of technical and human resources. Furthermore, ISPs see management as a new product with greater potential service differentiation than a pure connectivity service. However, management of the provider network will get much more complex. As of today, no complete integrated solution exists for a provider that wants to offer and manage enhanced IP services. Difficulties occur if the customer is able to order a service online (service automation) and the service requires the collaboration of several providers.

In this article we outline a generic architecture for the management and outsourcing of QoS-enhanced Internet VPNs. The application scenario is shown in Fig. 1. The service provider operates an IP network with VPN-enabled edge routers and a core network that support resource reservation mechanisms (e.g., multiprotocol label switching, MPLS). The customers operate stub networks and shall be enabled to order VPN connectivity with guaranteed service quality between those subnets.

We discuss IPSec, differentiated services (DiffServ), and other enabling technologies for QoS VPNs. We describe network management techniques, then propose a service management

0163-6804/01/$10.00 © 2001 IEEE

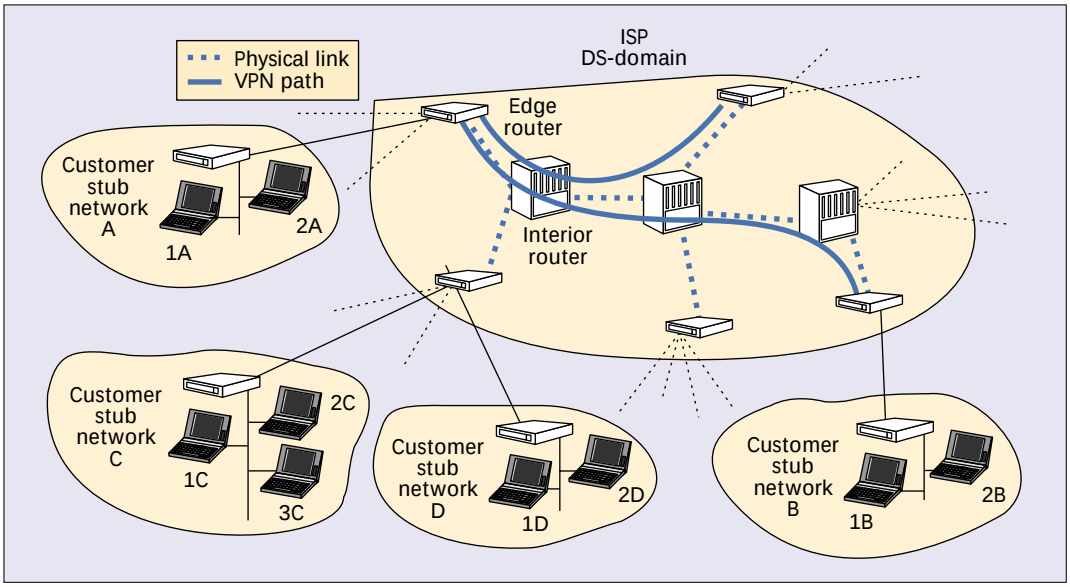

Figure 1. *A VPN deployment scenario.*

Tunneling is a method of wrapping a packet in a new one by prepending a new header. The whole original packet becomes the payload of the new one. At the tunnel starting point the new header (containing the tunnel endpoint address) is added.

architecture. We present a prototype implementation of that management architecture and conclude the article.

VPN Enabling Technologies

Tunneling (also called packet encapsulation) is a method of wrapping a packet in a new one by prepending a new header. The whole original packet becomes the payload of the new one. At the tunnel starting point the new header (containing the tunnel endpoint address) is added. When the new packet arrives at the tunnel endpoint, the header is removed and the original packet forwarded. Tunneling is often used to transparently transport packets of one network protocol through a network running another protocol. GRE (IETF RFC 1701), for example, is a multiprotocol carrier protocol for tunnels through IP networks. VPNs need tunneling to forward the encapsulated (private) packet (with private addresses) through the public network. Often, the private packet is also encrypted. The Security Architecture for IP (IPSec; IETF RFC 2401) provides protocols that support this style of tunneling.

IPSec is an open and widely supported architecture for IP packet encryption and authentication, that works on a per-packet basis. 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); another is the encryption through an encapsulating security payload (ESP); and finally, key management is automated through the Internet key exchange (IKE) protocol. IPSec can use any encryption algorithm to protect data and any message digest algorithm to authenticate data. However, for interoperability there is a standard set of cryptographic operations. Both AH and ESP support a tunneling mode. The tunneling mode also allows nesting of IPSec protocols. A common usage is, for example, to protect an IP packet with ESP and encapsulate/tunnel the result within the AH (in tunnel mode).

With IPSec-enabled access routers, a provider can set up VPN tunnels for customers. These tunnels can provide integrity, authenticity, and confidentiality to the traffic. However, to do so the provider has to perform a significant amount of configuration work at the tunnel endpoints (also called security gateways). Security associations (SAs) need to be established to define which traffic will go through the tunnels, and which encryption algorithms and secret keys will be used. The IKE protocol helps to establish SAs automatically. However, IKE depends on a security policy database which defines the guidelines for SA establishment. These guidelines may also regulate the interaction between the security gateways and a public key infrastructure such as a X.509 certificate server. The security policy database is not specified in IPSec. Furthermore, the discovery of security gateways is beyond the scope of IPSec. A provider that wants to offer VPN services thus cannot only rely on IPSec built-in management mechanisms, but must integrate IPSec into a service management framework.

QoS Support for VPNs

Differentiated Services

DiffServ (IETF RFC 2475) provides QoS in IP networks by traffic aggregation based on DiffServ code points (DSCPs). Packets are classified and processed by traffic conditioners at the edge of a DiffServ domain. DiffServ domains are typically equivalent to administrative domains, that is, a customer network or the network of an ISP. Service level specifications (SLSs) must be established among the various DiffServ domains. These SLSs form the basis for traffic conditioning actions such as shaping, policing, and remarking at the edge routers. DiffServ can provide QoS to Internet VPNs. Both technologies fit well with each other because of several commonalities:

Management of a DiffServ domain can be done using so-called bandwidth brokers. A bandwidth broker maps SLSs to concrete configurations of DiffServ routers, in particular to edge routers of a DiffServ domain.

- Both architectures logically separate service traffic from public traffic.
- Both classify IP packets.
- Both are enforced in edge routers.
- Both aggregate traffic flows.

For providing QoS guarantees similar to those customers are used to from leased line services, the expedited forwarding (EF) per-hop behavior (PHB) seems to be the appropriate choice (IETF RFC 2598). The EF PHB can be used to build a low-latency assured-bandwidth end-to-end service through DiffServ domains. Such a service appears to the endpoints like a point-to-point connection or virtual leased line. A typical SLS for such a service might include the ingress and egress points of the DiffServ domain that shall provide the service and a peak rate which can be guaranteed to the traffic stream.

The assured forwarding PHB group (IETF RFC 2597) is a means for a DiffServ provider to offer different levels of forwarding assurances for IP packets received from a customer DiffServ domain. Four AF classes are defined with three drop precedences each. A typical SLS includes rates for low and medium drop priority packets, and might also specify ingress and egress points. Although AF is considered more complex to configure in DiffServ domains, we also see great potential in AF for VPNs. In particular, a customer might prefer to specify a bandwidth range [Y,Z] rather than a single peak rate for a VPN tunnel. To support bandwidth ranges, an SLS can be configured with Y as the rate for low drop precedence and Z-Y as the rate for medium drop precedence.

When providing DiffServ-to-VPN tunnels, special attention must be given to DSCP mapping at the tunnel starting point. In outsourced VPNs the ingress router of an ISP might perform DiffServ classification and traffic conditioning as well as tunnel encapsulation. If encapsulation is performed first, the ingress router can select the appropriate DSCP for the corresponding tunnel; if DiffServ processing is done first, the DSCP of the inner IP header must be copied to the DSCP field of the outer IP header.

Management of a DiffServ domain can be done using so-called bandwidth brokers (discussed later). A bandwidth broker maps SLSs to concrete configurations of DiffServ routers, in particular to edge routers of a DiffServ domain.

Traffic Engineering

An important issue for providing QoS in VPNs is traffic engineering. Traffic engineering is the process of controlling how traffic flows through a network in order to optimize resource utilization and network performance. The basic motivation for traffic engineering arises from the fact that shortest-path-based routing leads to congestion on certain links while others remain relatively unused. This leads to inefficient network resource usage and increases the probability that a VPN tunnel cannot be established due to lacking resources along the shortest path between tunnel ingress and egress points. Traffic engineering in the past has been based on overlay models, running IP over an underlying connection-oriented network technology such as asynchronous transfer mode (ATM) or frame relay. In this case, meshes of connections have been established to interconnect IP routers of particular VPNs. In this case the overlay model served two purposes: providing a tunneling infrastructure to build VPNs and for traffic engineering, including QoS support. However, there are several problems with the overlay model. Most important is the management complexity with handling two kinds of network technology — IP and the underlying connection-oriented network (ATM or frame relay). Furthermore, mesh-like networks do not scale for large VPNs.

Multiprotocol Label Switching (MPLS)

MPLS (IETF RFC 3031) overcomes several limitations of the overlay model concerning scalability and efficiency. Mainly, two MPLS features are responsible for that: MPLS allows tunnels to be set up by appending a MPLS header in front of the IP header. This 32-bit MPLS header avoids the large overhead of another IP header required with IP-in-IP tunneling. The MPLS header can therefore be used several times (i.e., labels can be stacked).

Label stacking is in particular being used when building MPLS/Border Gateway Protocol (BGP)-based VPNs. In such a case, a packet is classified at an ingress router of an ISP based on the interface belonging to a particular VPN. The ingress router has learned via BGP to which VPN it belongs, to which egress router the packet must be sent, and via which egress interface the destination is reachable. The ingress router appends two labels to a packet belonging to a VPN. The inner label specifies the egress port at the ISPs egress router (i.e., the link toward the destination subnetwork of the VPN). The outer label is being used to forward the packet toward the egress router and can be learned by MPLS signaling protocols such as (CR)-LDP or TE-RSVP. Both labels are popped by the egress router. Note that MPLS makes the private VPN addresses of a customer transparent to the routers of the ISP (tunneling).

Another issue with MPLS is QoS support. Again, DiffServ fits very nicely since, in both concepts, DiffServ and MPLS additional intelligence (packet classification, MPLS labeling, DSCP marking, etc.) is required in edge routers, while interior routers just perform packet processing based on MPLS labels and DiffServ DSCPs. Although MPLS is a scalable technique for VPN tunneling, provides traffic engineering capabilities by its connection-oriented nature, and supports DiffServ QoS, it does not have built-in security mechanisms.

Traditional IP Network Management

Today, traditional IP network management consists mostly of manual network monitoring and configuration. The network administrator has to access each device, browse log files, and manually install configuration parameters. In order to provide QoS guarantees the administrator needs to possess a significant amount of information

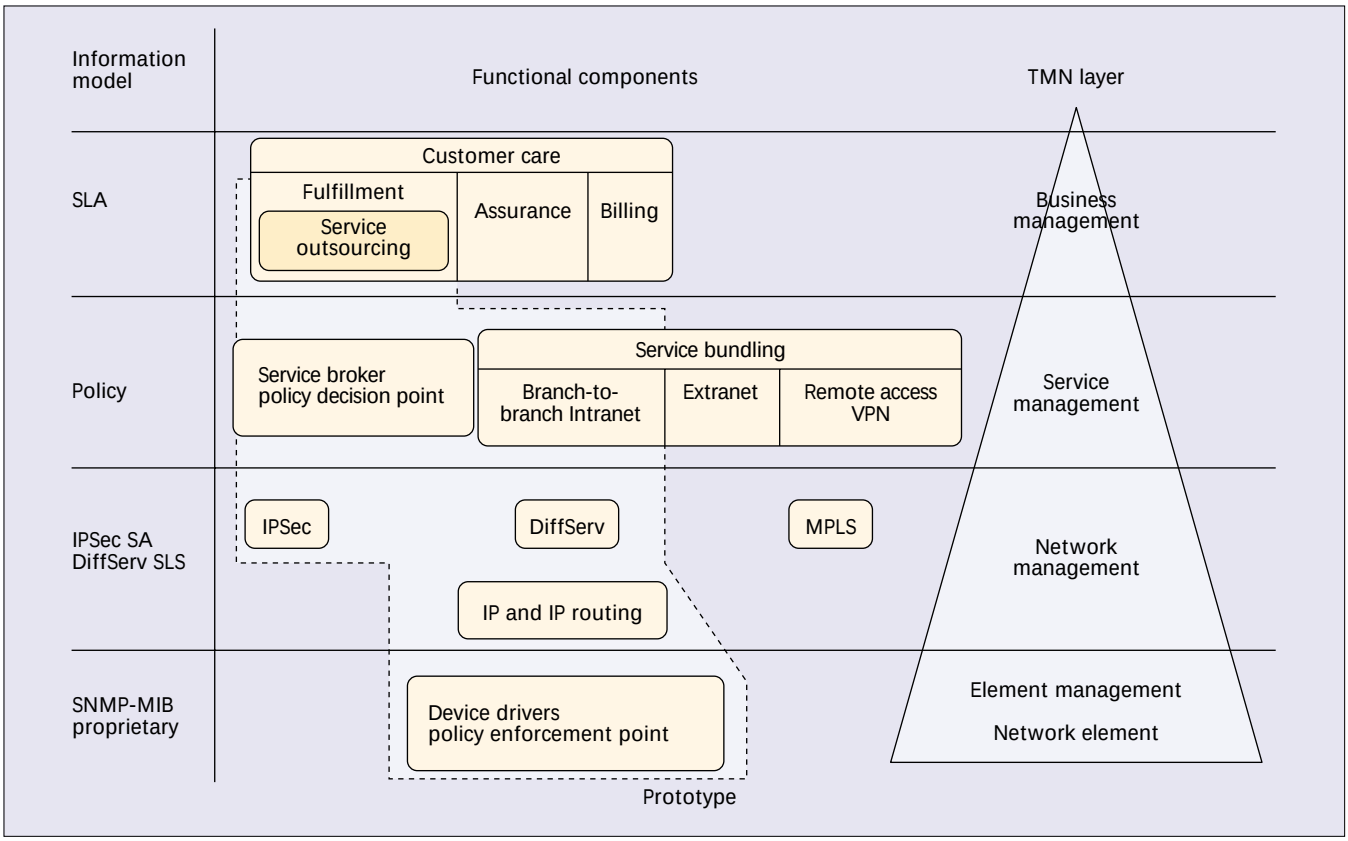

Figure 2. *QoS enabled VPN operation building blocks.*

about the network's devices and various applications, since not all devices support the same queuing and congestion control mechanisms. QoS configuration requirements are dynamic. Different locations (e.g., edge vs. interior routers) in the network require different QoS mechanisms to be implemented at different times. Manual configuration is time-consuming and prone to errors. VPN setup and management introduces additional management challenges in the area of security. Management security becomes even more important here.

For VPN management (and network management in general) a variety of commercial products exist. However, those products usually include proprietary technology and focus on a single task/service (e.g., QoS with corresponding performance monitoring). In addition, such management systems often demand a homogeneous hardware platform. To support network management in general, the IETF standardized the Simple Network Management Protocol (SNMP — IETF RFC 1157). SNMP allows monitoring of network elements and pushing of configuration information into all kinds of networking devices. The main advantage of SNMP is that it is open and widely adopted. However, SNMP does not support service management directly. The services have to be broken into device-specific networking functions which are outlined in a management information base (MIB). The time between service specification and deployment of MIB support in the devices is long (several years). MIB implementation for DiffServ and IPSec do not yet exist.

Policy-based network management [1] is another recent approach to solve scalability and consistency problems in network management. The Common Open Policy Service (COPS) protocol (IETF RFC 2748) outsources policy decisions to policy servers, also called policy decision points (PDPs). Various PDPs can access a centralized policy repository by means of SNMP or LDAP. The system administrator manages this repository. The policy servers provide the appropriate configuration information to the policy enforcement points (the network devices) on demand or by pushing. Policy-based network management was originally intended for QoS management, but was recently proposed to manage IPSec VPNs [2] and MPLS networks [3]. However, policy-based network management does not deal with customer care issues.

For DiffServ management *bandwidth brokers* have been proposed [4]. A bandwidth broker incorporates policy server functions, but also deals with customer contact and network resource allocation. For the outsourced QoS-VPN service, we propose to combine these open management technologies into a generic and hierarchical architecture with a high degree of automation.

A Generic QoS VPN Management Architecture

The telecommunications industry, which is the largest player in the IP network provisioning business, has standardized the telecommunications management network (TMN) model [5].

At the network layer several functional components are needed in order to offer QoS VPN service. A basic component to manage IP networks is required. Furthermore, network layer QoS support and IP security management components are needed.

The model provides a way to think logically about how the business of a service provider is managed. The model consists of five layers, starting from the network element layer followed by four management layers: element management, network management, service management, and business management (Fig. 2, right side). Each layer provides capabilities to the upper layers and each layer imposes requirements on the lower layers. The components of the lower layers are more distributed and technical. The higher the layer, the more information is concentrated into high-level abstractions. Higher layers can thus be more centralized, which eases preserving consistency. The business layer contains processes that deal with the provider's corporate strategy and customer relations. The service layer deals with the products offered to customers, namely the network services. The network management layer incorporates the management processes necessary for the provider's overall network infrastructure. The element management deals with processes concerning single devices in the network (servers, routers, switches, etc.). Finally, the element layer represents the heterogeneous devices that constitute a network.

Traditionally, the processes at all layers go through a more or less similar life cycle consisting of four phases. After planning (e.g., which equipment to buy or service to offer) and deployment there is a third phase, consisting of operation, maintenance, and monitoring. This phase is supposed to generate revenues for the provider. The cycle ends with an evolution/upgrade phase, which may lead to a new planning phase or withdrawal. The operation phase is ideally the longest phase and is not directly related to strategic decisions. It includes many repetitive tasks (monitoring, accounting, etc.). Therefore, it offers the greatest potential in automation. The rest of this article focuses on the management automation of the operation phase.

AUTOMATED QOS VPN SERVICE OPERATIONS

Today, the automation of network service management has reached the network layer of the TMN model. Our goal is to extend the automation so that functionalities in the service and business layers are covered, too. For automated QoS VPN operations management we propose a generic architecture. The functional components of the architecture are shown in Fig. 2. The architecture brings together the TMN model, SNMP, the DiffServ management architecture, and policy based network management.

The management automation tools for single devices vary with the devices' hardware and operating systems. Nevertheless, the devices should have a uniform interface toward the upper management layers. The *device driver* is a functional component that hides the vendor-specific configuration procedures of a network device. A widely accepted standard for representing device management information is the SNMP MIB. However, for some devices proprietary information models must be supported.

At the network layer several functional components are needed in order to offer a QoS VPN service. A basic component to manage IP networks (e.g., routing) is required. Furthermore, network layer QoS support and IP security management components are needed. These components manage the building blocks (security gateways, border routers) of a QoS VPN service according to requests of the service management layer. The information representation depends on the enabling technology. Examples for IPSec are SAs and for DiffServ the service level specifications (SLSs).

The service layer includes a *service bundling* component to compose several network layer mechanisms into end-to-end services for the customers. IPSec tunnels or MPLS paths are logically bundled into customer VPNs. The choice of which network services are bundled is a matter of the service planing phase and is not discussed here. Nevertheless, the automated management system must know which networking mechanisms are involved, and how they interact or interfere with each other. This is reflected in the service bundling component, which can support three of the most popular VPN models: remote access by individual users, branch-office-to-branch-office intranet, and multiparty extranets with buyers and suppliers. Consistency among the network configuration and running services is enforced through a centralized and intelligent software agent, also referred to as a *service broker*. The broker treats requests according to service policies, which is a typical function of a policy server (PDP). It also provides additional functionality such as capacity allocation for DiffServ (discussed later). The predominant information models at the service layer are policies (e.g., IPSec security policies or DiffServ policies).

The business layer comprises the *customer care* functionalities, which can be divided into three subgroups [6]. The fulfillment component represents, for example, the process of sales negotiations, including the establishment of SLAs. The Web is the dominant medium to access this component. The assurance component provides service status information to the customer, since customers of a QoS VPN service may want to make sure they get the service they have paid for (e.g., security) [7]. The billing component collects and sends the invoices. The predominant information representation at the business layer is the SLA. In an automated management architecture an SLA is a full-fledged electronic contract with all its legal implications. An SLA is established between the customer and the provider. In a multiprovider scenario a provider needs to access the management of peer providers. Here, one provider plays the role of an (outsourcing) customer. This can be expressed by setting up an SLA between providers. The service outsourcing business management component incorporates this functionality.

Information Flows: Two main information flows go through the presented architecture during operation: configuration requests and measurements. A stream of configuration commands descend from the top layer down to the devices. High-level instructions are decomposed into commands to the next lower layer, until

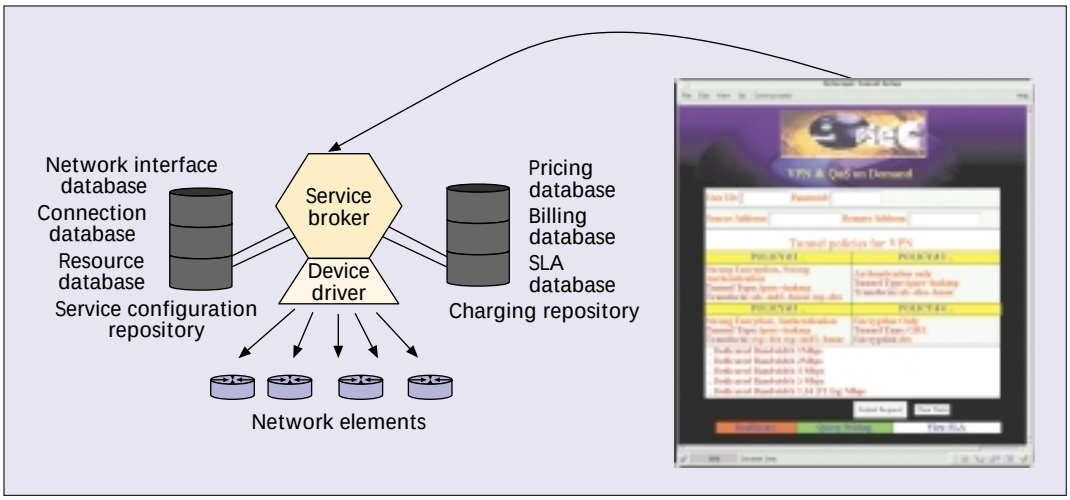

Figure 3. *Service broker components and the broker's Web interface.*

Today, the automation of network service management has reached the network layer of the TMN model. Our goal is to extend the automation so that functionalities in the service and business layers are covered, too.

finally device-specific configurations are activated. The root of the bottom-up information flow consists of measurements in network devices. This measurement data is compressed into network status and usage information at the network layer. The service layer uses the data to ensure correct service operation and to generate accounting records that are finally processed (among other information, service bundling in the SLAs) by the billing component. The following QoS VPN outsourcing example illustrates cooperation between functional components. The successful setup of a QoS VPN goes through three stages: request, admission control, and service activation.

Request — A customer wishes leased-line-like IP connectivity between two subnets. The privacy of the IP traffic shall be protected, and the customer desires throughput guarantees. A provider offers a QoS VPN service with selectable dedicated bandwidth and security features. Therefore, the customer requests an SLA from the outsourcing component of this provider. The outsourcing component contacts the service broker.

Admission Control — The service broker learns the semantics of the QoS VPN service through interaction with the service bundle component. It therefore knows how different network service functionalities must be combined to provide the requested end-to-end service. In this example the request involves the DiffServ and IPSec components. The broker combines information provided by the IP routing component, the service bundle component, and the customer request to calculate the network resources needed to support the service. The service broker contacts the appropriate network management components to query if the required resources are available given the selected service parameters. For example, it queries the IPSec management component if the access routers are not overburdened with the additional work imposed by IPSec given the selected cryptographic mechanisms. The service broker also verifies that the network can support the additional traffic load with the desired PHB. It interacts with DiffServ management to do so. If the requested QoS VPN also involves peering providers, the service broker contacts the service outsourcing component of those providers.

Service Activation — After the service broker admits the customer request, it reserves the resources. The broker then triggers service activation. The activation is performed by the IPSec and DiffServ management components. These components delegate the configuration of the network devices to the device drivers. The broker notifies the outsourcing component, which then commits to the SLA, thereby informing the customer that the VPN is ready.

The prototype described in the next section implements the functionalities framed in Fig. 2.

A Prototype Implementation of a QoS VPN Management System

Within the Charging and Accounting Technology for the Internet (CATI) [8] project we have developed a prototype system [9, 10] to demonstrate dynamic establishment of QoS-enabled VPN tunnels that is also capable of generating user billing information. The central component, the *service broker* (SB), plays the role of the policy server. It is the heart of our VPN management system that acts as a QoS manager to optimally allocate network resources by performing admission control at edge devices. The admission decisions could take place with minimum user intervention with respect to specifying the user's requirements. Since the underlying network may provide different classes of service to satisfy various VPN customers by identifying the generic functionality provided by any resource, we present our SB with a standard interface (Fig. 3) to the network resources. The interface provides user-selectable policy options to make it easier for end users to create VPNs dynamically, almost in point-and-click fashion. The user can select a VPN service using authentication only, encryption only, encryption with

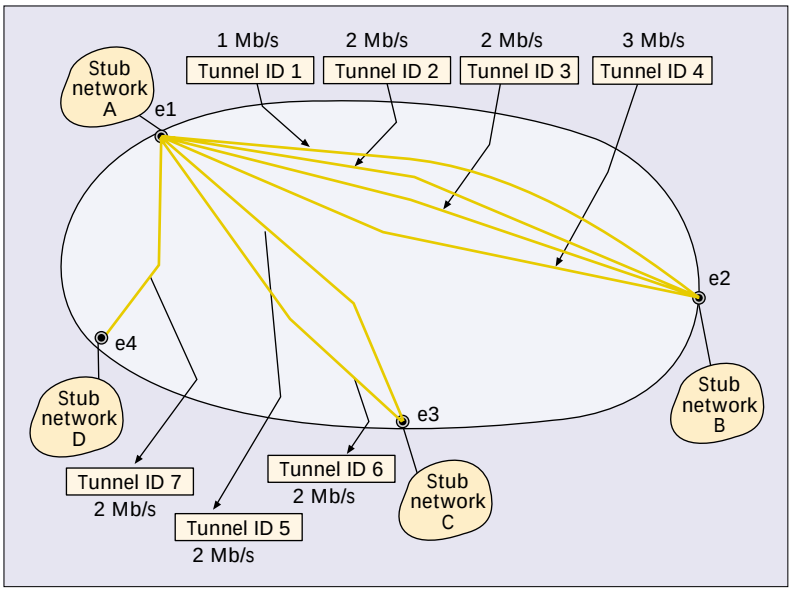

Figure 4. *Mapping of resources to various tunnels.*

partial authentication, or encryption with full authentication. Furthermore, a set of dedicated bandwidth classes are offered.

SB Structure and System Components

Our SB interacts with a specialized intelligent device driver when a certain user request arrives to set up a tunnel and the SB has to decide whether it can allocate enough resources to meet the demand of that tunnel. Device drivers are intelligent provisioning agents that are able to translate user requests and SB-generated pseudo rules into device-specific rules to configure the routers/switches since we might have several of those devices from various vendors that need to be configured in different ways. The SB uses a service configuration and a charging repository (Fig. 3). These are collections of various databases needed for dynamic service activation.

SLA Database — The SLA database not only contains the user's identification, but also specifies the maximum amount and type of traffic he/she can send and/or receive for a tunnel. The SLA also contains the boundary of the valid VPN area. Referring to Fig. 1. where stub networks A, B, and C might belong to the same organization and be located at different remote locations, one can easily see that they form a mesh environment, and any site may want to establish a connection with another under the same contract. Therefore, this boundary defines the perimeter of the VPN area and is entered in this database as source and remote stub addresses. The user authentication process prohibits malicious users from setting up unauthorized tunnel and access network resources illegally. The SLA, however, allows users to add new VPN areas to their current contracted list of valid VPN areas. It contains the following tuple:

<User ID, Password, Maximum BW in Mb/s, Source Stub Address, Remote Stub Address>

Resource Database — The resource database contains resources available between any two edge routers. This means that this database has resource information for all the routers in a certain domain. In our implementation, dynamically established tunnels traverse through pinned paths (e.g., MPLS label switched paths, LSPs) that can be established statically. We assume that those paths are able to protect the traffic marked as EF at VPN edges by using appropriate queuing mechanisms. We allocate tunnel IDs to the tunnels in order to keep track of resource usage. For each tunnel, however, we also need to know its ingress router's IP address, ingress outbound, egress router's IP address, egress outbound (this might as well be the same as the egress router's IP address), and the capacity of the tunnel in megabits per second. While ingress and egress router addresses are necessary for identifying the edge routers, the outbound addresses are needed to define tunnel endpoints, and in fact, for a single router there might be several outbound interfaces. Also, in the database we need to keep track of the status of the tunnel in terms of availability. Therefore, the tuples are:

<tunnel id, ingress router, ingress outbound, egress router, egress outbound, bandwidth, status>

However, it should be clarified that a tunnel might originate from an ingress router to several possible egress routers. Referring to Fig. 4, users residing in stub network A might want to establish tunnels between routers e1 and e2, or e1 and e3, or e1 and e4 to communicate with other stub networks. Assume that an ISP has decided to allocate a maximum 10 Mb/s capacity to traffic stemming from e1 and destined toward other edge points. The ISP might, however, allow 1 Mb/s tunnel, two 2 Mb/s tunnels, and one 3 Mb/s tunnel to be created between e1 and e2, and also allow two 2 Mb/s tunnels from e1 to e3 and only one 2 Mb/s tunnel from e1 to e4.

It is clear that if all the tunnels were active simultaneously, router e1 would need to support 14 Mb/s. The edge admission control of the service broker ensures that the allocated resources always stay below the maximum capacity (10 Mb/s in this example).

Connection Database — The connection database contains a list of currently active VPN connections. Here a connection always means a VPN tunnel and is identified by source-destination pairs. It has various functions:

- When a new request arrives for connection activation or termination, the SB can check whether or not that connection already exists and then make its decision.
- It indicates how many resources have been consumed by VPN users.
- It provides records to the pricing mechanism.

Its tuples are:

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

Interface Database — The interface database contains necessary records of edge routers that

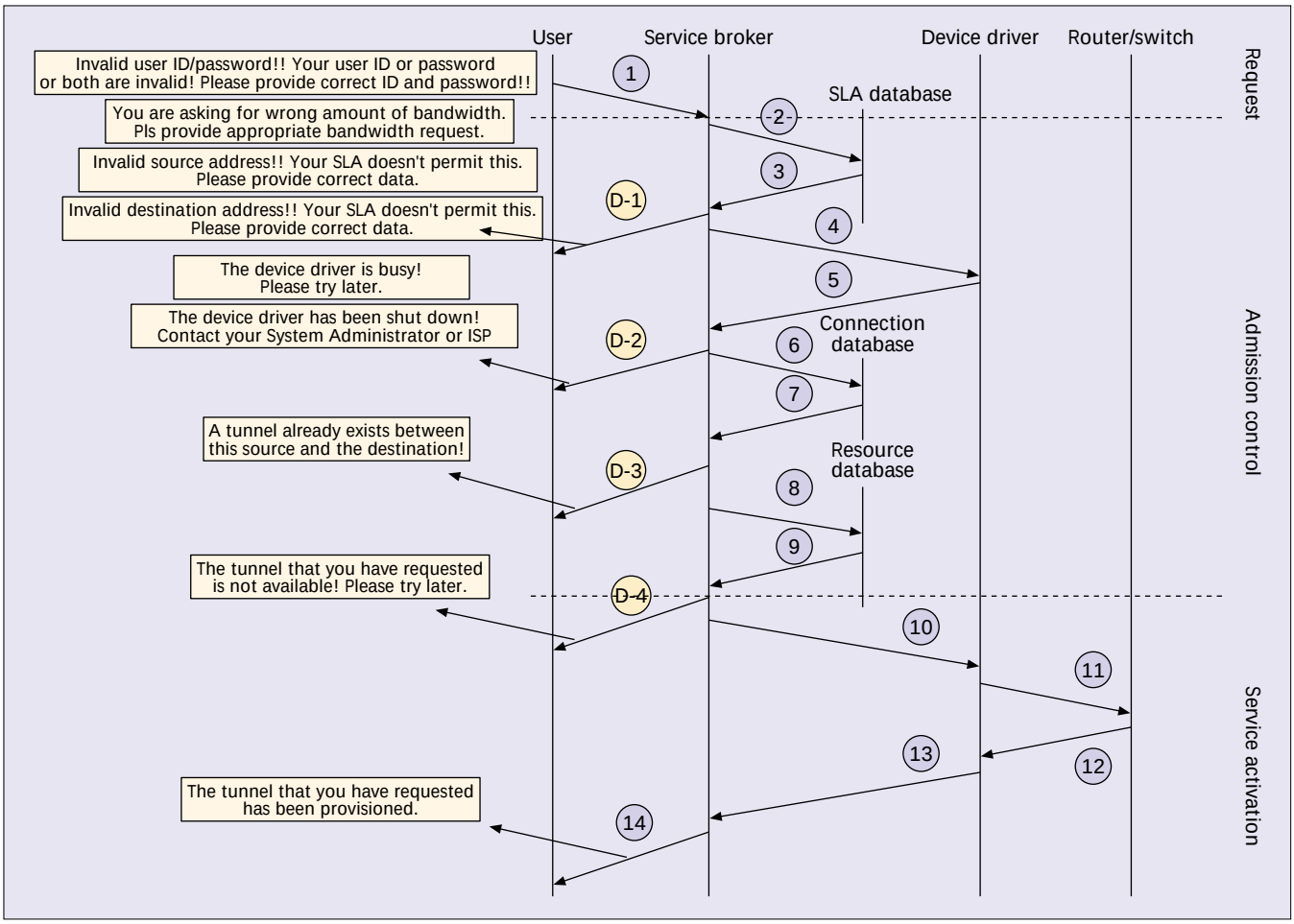

Figure 5. *The VPN setup procedure or possible rejection (D-x: denial).*

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

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

Pricing and Billing Database — The pricing database contains pricing information on various tunnels. Its only interaction with the SB is when a connection (tunnel) is terminated and the SB needs to know the price of it by making a query to it. The billing database contains details of terminated connections and their computed prices.

Operational Details — Figure 5 shows all the communications involved in setting up a VPN connection between two stub networks or simply between an originating host and a remote host. We will describe the operational details by referring to the communications marked on Fig. 5, considering each communication in turn.

A user sends a VPN connection request containing user ID, user password, source and remote address for the tunnel, amount of bandwidth, and encryption/tunneling method (for example, policy #1 on the webpage of Fig. 3). The SB contacts the SLA database that is responsible for validating the user and his/her request. If the user has been identified correctly, his/her source and remote address conforms to the contract, and also the bandwidth requested is less than or equal to the agreed traffic contract, it sends a positive response (steps 2 and 3). After this validation the SB sends a signal to the device driver to check its status. The status can be busy, available, or down. Only in the case of availability can the user request be processed further (steps 4 and 5). The SB then contacts the connection database to check the existence of a similar tunnel. This is because between a source and destination, only one tunnel can remain active (steps 6 and 7). Before edge admission control the SB finds which edge routers should be configured by checking the interface database. During the admission con-

The implementation presented here supports Web access by customers and enables them to set up IPSec VPN tunnels with QoS guarantees. The system is a step toward solving the problem of automated and integrated management of QoS VPNs.

trol process the resource database responds to the SB and either allocates the resource or denies it based on resource availability (steps 8 and 9). A positive admission control decision prompts the SB to signal the device driver to create appropriate configuration scripts (step 10). In the meantime, the resource and connection databases 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. In the remaining steps the routers are configured, and the SB sends notification to the user.

A connection request is rejected if:

- The SLA profile doesn't match (case D-1).
- The driver is found busy (case D-2).
- The connection already exists (case D-3), or not enough resources are available (case D-4).

The various stages where a connection creation process might get refused are shown in Fig. 5. A connection termination process releases reserved resources by updating records in the resource database. It also invokes pricing and billing databases to generate user billing at the end of tunnel termination.

CONCLUSION

Internet-based QoS VPNs have become a feasible and economically interesting solution for deploying wide area corporate networks. However, the QoS and VPN enabling technologies increase network management complexity significantly. In this article we propose a TMN-based architecture that enables service providers to operate QoS VPNs for their customers. The architecture is compatible with emerging IP network management techniques. We then present an implementation instance of the architecture. The implementation supports Web access by customers and enables them to set up IPSec VPN tunnels with QoS guarantees. The system is a step toward solving the problem of automated and integrated management of QoS VPNs.

ACKNOWLEDGMENTS

The work described in this article is part of the work done in the CATI project (nos. 5003-054559/1 and 5003-054560/1) and its continuation, Advanced Network and Agent Infrastructure for the Support of Federations of Workflow Trading Systems (ANAISOFT) (SPP ICS ElCom Project 5003-057753), both funded by the Swiss National Science Foundation (SNF). The implementation platform has been funded by SNF R'Equip project no. 2160-053299.98/1 and Foerderung der wissenschaftlichen Forschung an der Universitaet Bern. The authors want to thank the anonymous reviewers, Dr. J. Schneider, and Silvia Bechter for their helpful comments.

REFERENCES

[1] R. Rajan *et al.*, "A Policy Framework for Integrated and Differentiated Services in the Internet," *IEEE Network*, vol. 13, no. 5, Sept./Oct. 1999, pp. 36–41.
[2] S. J. Baek, J. W. Baek, and J. T. Park, "Policy-based Management Architecture for Internet VPN Using DEN," *IEEE IPOM*, Sept. 2000.
[3] P. Aukia *et al.*, "RATES: A Server for MPLS Traffic Engineering," *IEEE Network*, vol. 14, no. 2, Mar./Apr. 2000.
[4] A. Terzis *et al.*, "A Two-tier Resource Management Model for the Internet," *IEEE Global Internet '99*, Dec. 1999.
[5] ITU-T Rec. M-3400, "TMN Management Functions."
[6] TM Forum, Telecom operations map, http://www.tmforum.org,Mar. 2000.
[7] M. Guenter and T. Braun, "Internet Service Delivery Control with Mobile Code," H. R. van As, Ed., *Telecommunication Network Intelligence*, IFIP, Kluwer, Sept. 2000, pp. 3–19.
[8] B. Stiller *et al.*, "Charging and Accounting Technology for the Internet," *ECMAST '99*, LNCS 1629, Springer-Verlag, May 1998, pp. 281–96.
[9] I. Khalil, T. Braun, and M. Guenter, "Implementation of a Service Broker for Management of QoS Enabled VPNs, " *IEEE IPOM 2000*, Sept. 2000.
[10] I. Khalil and T. Braun, "Implementation of a Bandwidth Broker for Dynamic End-to-End Resource Reservation in Outsourced Virtual Private Networks," *25th Annual IEEE LCN*, Nov. 9–10, 2000.

BIOGRAPHIES

TORSTEN BRAUN (braun@iam.unibe.ch) got his degree and Ph.D. from the University of Karlsruhe, Germany, in 1990 and 1993, respectively. From 1994 to 1995 he was a guest scientist at INRIA Sophia-Antipolis,France. From 1995 to 1997 he worked at the IBM European Networking Center, Heidelberg, Germany, at the end as a project leader and senior consultant. Since 1998 he has been a full professor of computer science at the Institute of Computer Science and Applied Mathematics at the University of Bern, Switzerland, where he is heading the Computer Networks and Distributed Systems research group. He was elected a member of the Swiss Academic Research Network (SWITCH) committee in 2000.

MANUEL GUENTER (mguenter@iam.unibe.ch) got his degree from the University of Bern, Switzerland, in 1998. He is now a research assistant and Ph.D. student in the group of Prof. T. Braun. His research interests are new IP services, interdomain collaboration, network and service monitoring, and mobile agents.

IBRAHIM KHALIL (ikhalil@ponte.com) has an M.S. degree in computer engineering from University Putra Malaysia and a B.S. in electrical engineering from BIT Rajshahi, Bangladesh. He was a graduate research assistant with the ATM research group in electronics and computer engineering, University Putra Malaysia between 1993 and 1996. Prior to joining the group of Prof. Braun in 1998 as research assistant and Ph.D. student, he briefly worked with the LTS and C3i groups of EPFL, Switzerland. His research interests are dynamic network provisioning, QoS issues of VPN, MPLS, and network pricing. He is currently working with Mountain View, California based Ponte Communications, USA.