Chapter 5

Defining a VPN

Solutions in this chapter:

- **What IS a VPN?**
- **Public Key Cryptography**
- **IPSec**
- **SSL VPNs**
- **SSH Tunnels**
- **Others**

- ☑ **Summary**
- ☑ **Solutions Fast Track**
- ☑ **Frequently Asked Questions**

Introduction

In this chapter, we cover concepts of virtual private networks (VPNs), how they operate, and the different types of VPNs in use today. Before we dive into the details, you may be thinking, "What is a VPN, and why would I need to use one?" There are several good reasons to implement VPN technology in your infrastructure, starting with security. A VPN is a means of creating secure communications over a public network infrastructure. VPNs use encryption and authentication to ensure information is kept private and confidential. This means you can share data and resources among several locations without the worry of data integrity being compromised. Alone, the ability to make use of a public network to transmit data is also an advantage of VPN technology. Without using the Internet as a transport mechanism, you would have to purchase point-to-point T1s or some other form of leased line to connect multiple locations, or use frame relay service. Leased lines are traditionally expensive to operate, especially if the two points being connected are across a large geographic region. Using VPNs instead reduces the operating cost for your company.

VPNs are also cost effective for traveling users. Without VPNs, a traveling salesperson working outside the office might have to dial in to a modem bank at the office and incur long distance charges for the call. A dialup VPN is much more cost effective, allowing the salesperson to connect to a local ISP (Internet service provider) and then access the corporate network via a VPN.

Suppose your company's corporate office has a database-driven intranet site it wants your branch offices to be able to access, but does not want the rest of the world to have access to this site. Sure, you could just stick the application on an Internet-facing server and give each user a password-protected account, but the information will still be transmitted unencrypted to the user. By creating a VPN between the two sites, the branch office can access the intranet site and share resources with the corporate office, increasing productivity and maintaining a higher level of security all at the same time.

What Is a VPN?

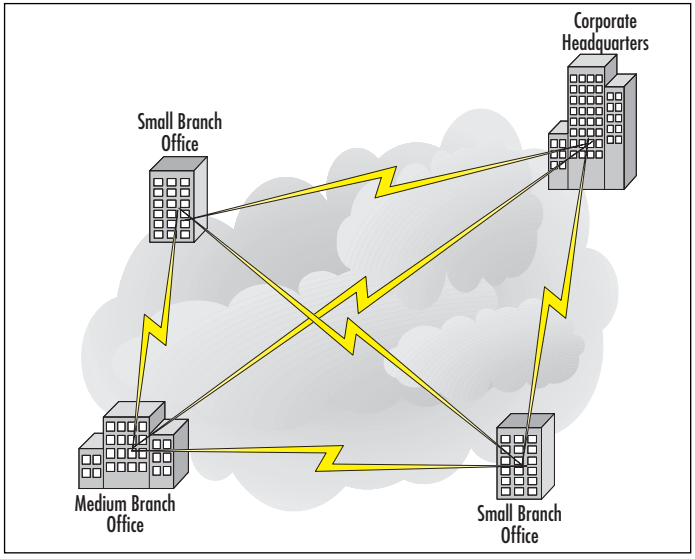
A traditional corporate WAN is shown in Figure 5.1. Branch office networks are connected either through a circuit-switched data path such as ISDN, providing low-end, broadband connection, or through packet-switched technologies such as frame relay or leased lines (T1, DS3, etc.). The cost of such a WAN topology increases significantly as the number of sites and interconnections between the sites increases. For a fully meshed topology with four endpoints, six frame relay or serial connections

are required. In general, a full meshed network with *n* nodes requires (*n** (*n* – 1)) / 2 links. This system quickly becomes quite expensive as the number of nodes increases.

VPNs provide dramatic flexibility in network design and a reduced total cost of ownership in the WAN. A VPN can be best described as an encrypted tunnel between two computers over an insecure network such as the Internet. VPNs provide secure encrypted channel to secure communication, and cost savings in the ranges of 30 to 80 percent depending on the leased line and the destination.

There are various ways to implement VPN services, including at the enterprise-edge router, the firewall, or a dedicated VPN appliance. Additionally, MPLS can be provided by the ISP for site-to-site VPN traffic. Another possibility is the virtual private dialup network (VPDN). Primarily used for remote-access connection to an enterprise campus network, this type of VPN combines the traditional dialup network through the PSTN with either Layer 2 Forwarding (L2F) or L2TP. All of these various technologies are available in today's marketplace, but the most popular VPN technology, by far, is the IPSec VPN.

Figure 5.1 Fully Meshed Enterprise WAN Connectivity



VPN Deployment Models

One of the first decisions you must make when deploying a VPN is choosing a device to serve as the termination point for the VPN tunnel. This decision is pri-

marily driven by the placement of the VPN tunnel endpoint, and the capabilities of the device that will serve as the tunnel endpoint. IPSec VPNs require devices capable of handling the traffic traversing outside the VPN and the VPN traffic and the encryption of the data across the VPN. Insufficient computing power results in a slow connection over the VPN and poor performance overall. Many vendors address this problem by offering VPN accelerator modules (VAMs) onboard processors designed to provide encryption services for the VPNs.

Deployment of VPNs in the enterprise DMZ is primarily done through the three models listed here and shown in Figures 5.2 through 5.4:

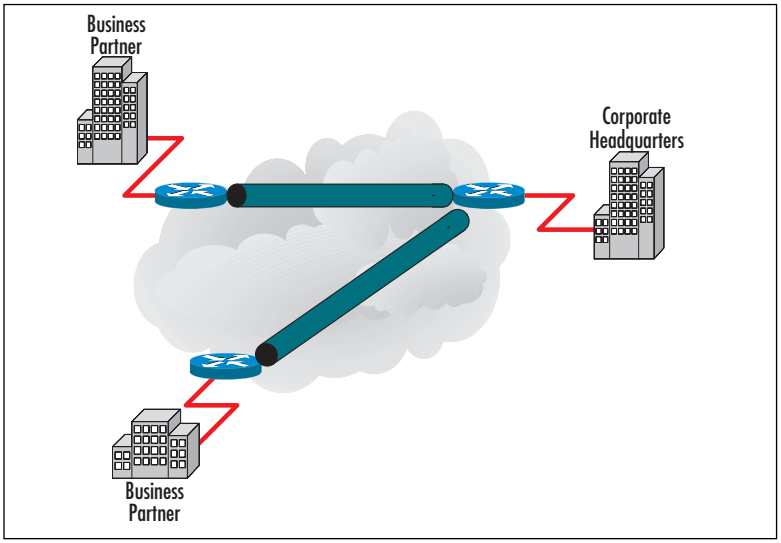
- VPN termination at the edge router
- VPN termination at the corporate firewall
- VPN termination at a dedicated appliance

Each of these deployment models presents its own difficulties that must be addressed for the VPN topology to be successful. One concern that must be addressed is the use of Network Address Translation (NAT). Due to its design, IPSec is not capable of traversing NAT devices. The problem comes when the NAT device changes information in the IP header of the IPSec packet. The changes will result in an incorrect IPSec checksum that is calculated over parts of the IP header. There are vendor workarounds for this problem, where the IPSec packet is encapsulated in a UDP or TCP packet and then transmitted to the other side. Currently, this solution is an Internet draft and has not reached request for comment (RFC) status. The ports to use for such communication are negotiated during tunnel setup.

VPN Termination at the Edge Router

Termination of the VPN at the edge router has the benefit of ensuring that all VPN traffic must conform to external firewall policies to reach the internal network. This topology (shown in Figure 5.2) is best deployed for extranet connections where the business partners do not require access to the internal network but do require access to servers in the DMZ itself that might not necessarily be exposed to normal Internet traffic. As the number of business partners connecting through VPNs increases, the load on the routers due to the encryption and decryption of packets entering and exiting the VPN tunnels also increases. This situation requires the use of VAMs to offload the encryption/decryption process from the router CPU.

Figure 5.2 VPN Termination at Edge Routers



VPN Termination at the Corporate Firewall

Termination of the VPN at the corporate firewall allows for direct access from branch networks to the internal corporate core network. Remote users can then access all internal services without having to authenticate a second time. This particular topology (shown in Figure 5.3) is best reserved for LAN-to-LAN connections such as branch-office-to-corporate-enterprise networks, but can also be used for WAN connections if there is a router in front of the firewall to direct traffic over the Internet. The drawback to this topology is that as more branch offices are connected to the corporate office, the load on the firewall increases due to the increased amount of encryption each VPN requires. When the load on the firewall reaches a point at which there is an overall impact on network connectivity, it is best to either add a VAM to the firewall or offload the VPN services to a dedicated device.

VPN Termination at a Dedicated VPN Appliance

Dedicated VPN appliances are designed to provide VPN tunnel services for LAN-to-LAN connections. Termination of the VPN at the corporate firewall allows for direct access from branch networks to the internal corporate core network. Remote users can then access all the internal services provided without having to authenticate a second time. This particular topology (shown in Figure 5.4) is best reserved for LAN-to-LAN connections such as branch-office-to-corporate-enterprise networks. The drawback to this topology is that as more branch offices are connected to

the corporate office, the load on the firewall increases due to the increased amount of encryption each VPN requires. When the load on the firewall reaches a point at which there is an overall impact to network connectivity, it is best to either add a VAM to the firewall or offload the VPN services to a dedicated device.

Figure 5.3 VPN Termination at the Firewall

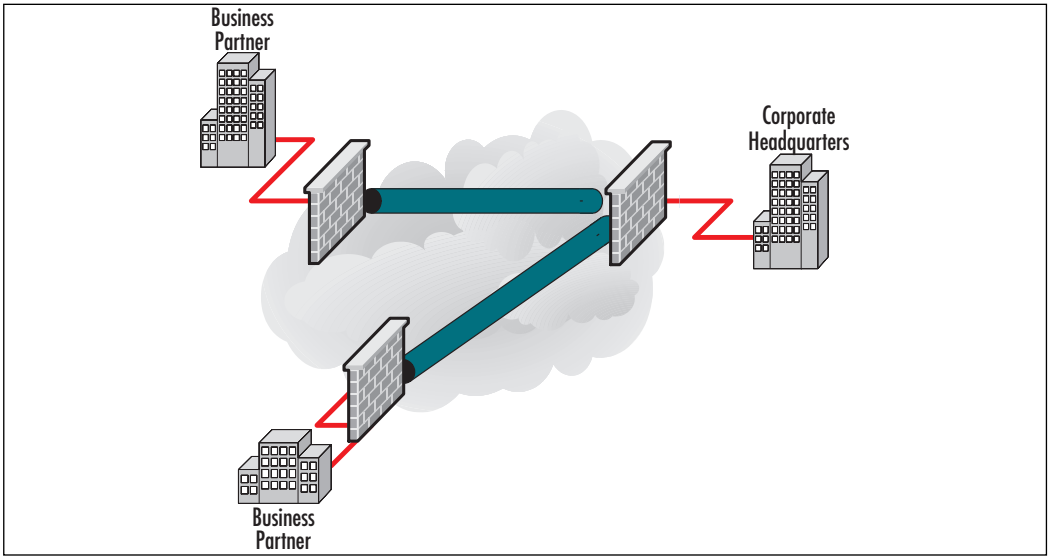
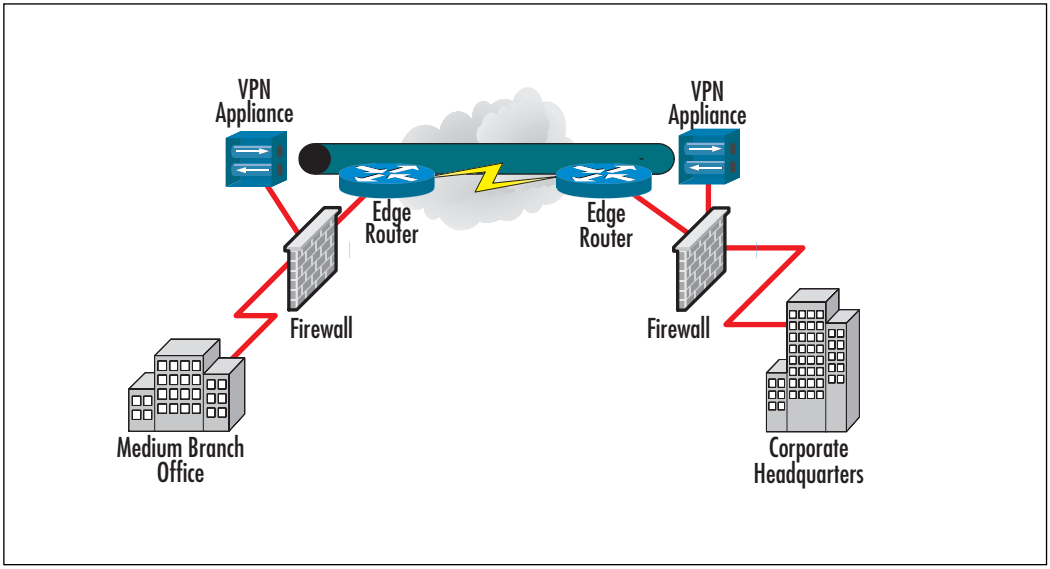


Figure 5.4 VPN Terminations at a Dedicated VPN Appliance



A further benefit to this deployment model is the ability to use the VPN appliances in conjunction with wireless networks.

Topology Models

The deployment models we've discussed represent how VPNs can be implemented to provide access either to the DMZ or to the internal corporate network. This section focuses on the various topologies in which these models can be deployed. There are four general topologies to consider:

- Meshed (both fully and partially meshed)
- Star
- Hub and spoke
- Remote access

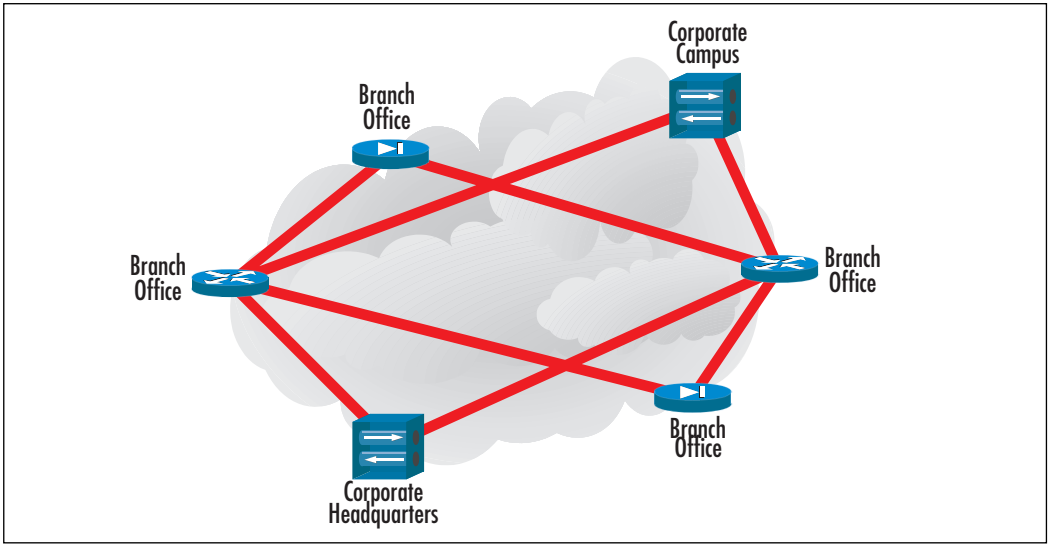
Each of these topologies is considered in greater detail in this section.

Meshed Topology

Like their traditional WAN counterparts, meshed VPN topologies can be implemented in a fully or partially meshed configuration. Fully meshed configurations have a large number of alternate paths to any given destination. In addition, fully meshed configurations have exceptional redundancy because every VPN device provides connections to every other VPN device. This topology was illustrated in Figure 5.1. A simpler compromise is the partial-mesh topology, in which all the links are connected in a more limited fashion to other links. A partial-mesh topology is shown in Figure 5.5.

Mesh topology provides an inherent advantage that there is no single point of failure. Overall performance of the setup is independent of a single node or a single system. Sites that are geographically close can communicate with each other. Its main drawback is maintenance and key maintenance. For a fully meshed network, whenever a new node is added, all the other nodes will have to be updated. Even with the replacement of traditional WAN services such as frame relay or leased lines, fully meshed topologies can be expensive to implement due to the requirement to purchase a VPN device for every link in the mesh.

Figure 5.5 Partial-Mesh VPN Topology



NOTE

Another issue you should be aware of with full versus partial-mesh topology is the number of tunnels you need to configure and manage. If you have 100 sites and add one router, think of all the connections you must make to rebuild a full mesh! In essence, the partial mesh is the way you want to go, but you might see an extra hop in the route from place to place because you will no longer have a single hop to any single destination. There is always give and take. Think about what method suits your design needs, and implement that method accordingly.

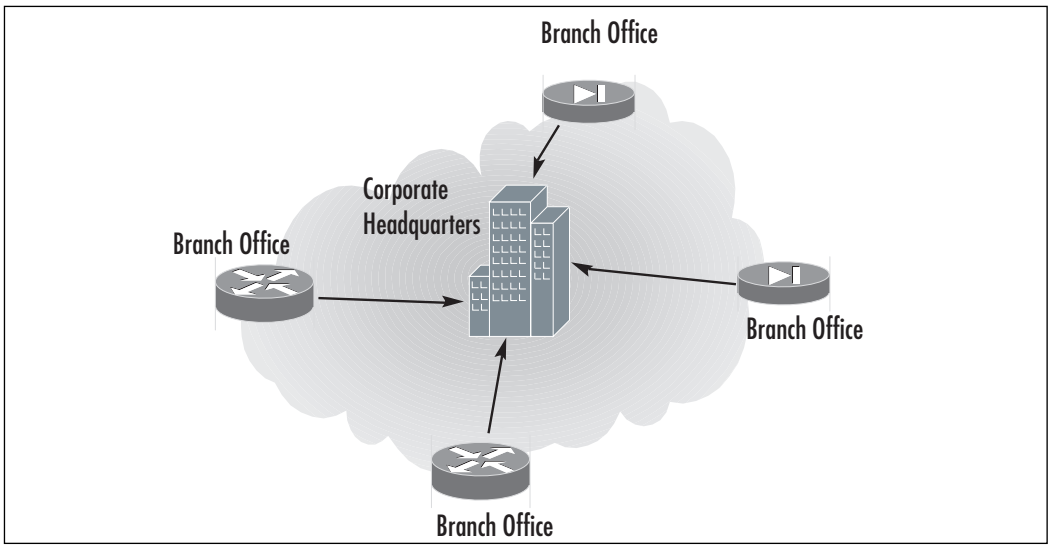
Star Topology

In a star topology configuration, the remote branches can communicate securely with the corporate headquarters or central site. However, intercommunication between the branches is not permitted. Such a configuration could be deployed in a bank network so that compromise of one branch will not immediately lead to the compromise of a second branch without being detected. To gain access to a second branch, the attacker would have to first compromise the central network that would hopefully be able to detect such an attack. A star topology configuration is shown in

Figure 5.6. Star topologies provide an inherent advantage that a new site can be added with ease; only the central site will have to be updated.

In star topology, the central site plays an important role; if it fails, all the connections will go down. Performance of the central hub dictates the performance of the connection. For a star topology, it may happen that two nodes might be closed to each other; however, they will have to communicate via central node.

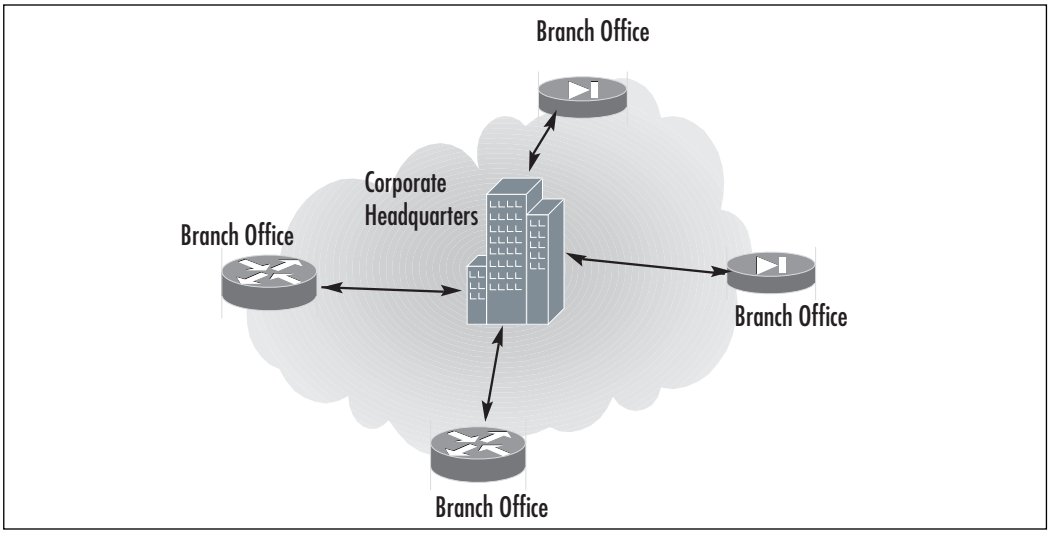
Figure 5.6 Star VPN Deployment Topology



Hub-and-Spoke Topology

A hub-and-spoke topology by design looks very similar to the star topology (Figure 5.7). However, there is one significant difference: Unlike the star topology, all branch or stub networks in a hub-and-spoke topology are able to access other branch or stub networks. The central, corporate network works as a simple transit point for all traffic from one end of the network to another. As traffic transits through the central corporate network, the data is decrypted, inspected, and re-encrypted for transmission to the final destination. This topology has more risk inherent in the design than the star topology because an attacker who is able to compromise one branch network might then be able to attack another branch network through the VPN without being required to attack the central, corporate network.

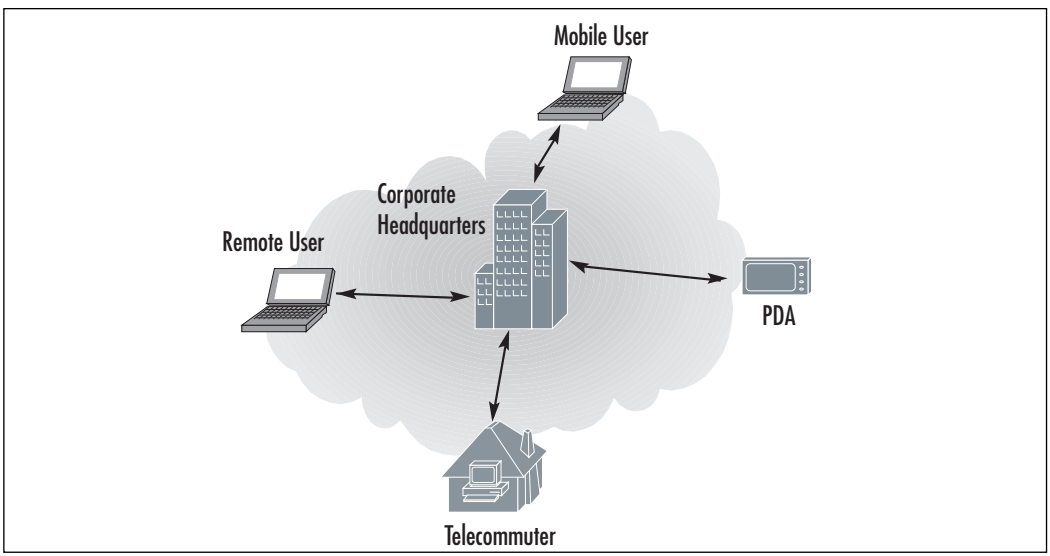
Figure 5.7 Hub-and-Spoke VPN Topology



Remote Access Topology

A final topology to consider is the remote access topology. Built on the hub-and-spoke VPN topology, this design focuses more on providing connectivity for remote users such as telecommuters, mobile workers, and other users who need access from no static IP addresses (Figure 5.8).

Figure 5.8 Remote Access VPN Topology



The next section presents advantages and disadvantages of VPNs.

Pros of VPN

By using a VPN, organizations are not dependent on the expensive leased or frame relay lines. The employees are able to connect remote users to their corporate networks via a local Internet service provider (ISP) instead of via expensive 800-number or long distance calls to *resource-consuming modem banks*. This in turn will result in low cost to the companies. In addition, a VPN provides the best security using encryption and authentication protocols. By using a VPN, corporate can also use the remote access infrastructure within ISP. Corporations can add virtually unlimited amount of capacity without adding significant infrastructure. Mobile workers can access the corporate networks on their high-speed broadband connectivity. This in turn will yield flexibility and efficiency.

Cons of VPN

Although a VPN provides security and flexibility, it does not offer quality of service. In a VPN, end-to-end throughput is not guaranteed, and there can be packet losses, delivered out of order, and fragmented. In a VPN, it is tough to achieve bandwidth reservation. Bandwidth reservation refers to the ability to reserve the bandwidth for traffic. It might be possible to reserve on out-bound traffic. For inbound traffic if you wish to have a bandwidth reservation, the VPN carrier will have to configure the VPN. For dialup users, a VPN tunnel can provide overhead in terms of additional protocol header, increase in authentication latency, and poor PPP and IP compression as compared to the direct link. This in turn will increase the latency for reconnection time. Encrypted data is not compressible, so when we use VPN to encrypt data, compression of data cannot be achieved. This in turn also means that hardware compression over the modem connection cannot be achieved.

The next section discusses public key cryptography, which will in turn help you to understand the concepts of IPSec.

Public Key Cryptography

Public key cryptography, introduced in the 1970s, is the modern cryptographic method of communicating securely without having a previously agreed upon secret key. Public key cryptography typically uses a pair of keys to secure communications—a private key that is kept secret, and a public key that can be widely distributed. You should not be able to find one key of a pair simply by having the other. Public key cryptography is a form of asymmetric-key cryptography, since not

all parties hold the same key. Some examples of public key cryptography algorithms include RSA, Diffie-Hellman, and ElGamal.

Notes from the Underground…

What Is Diffie-Hellman?

The Diffie-Hellman (DH) key exchange protocol, invented in 1976 by Whitfield Diffie and Martin Hellman is a protocol allowing two parties to generate shared secrets and exchange communications over an insecure medium without having any prior shared secrets. The Diffie-Hellman protocol consists of five groups of varying strength modulus. Most VPN gateways support DH Groups 1 and 2. The Diffie-Hellman protocol alone is susceptible to man-in-the-middle attacks, however. Although the risk of an attack is low, it is recommended that you enable *Perfect Forward Secrecy* (PFS discussed later in the chapter) as added security when defining VPN tunnels. For more information on the Diffie-Hellman protocol, see www.rsasecurity.com/rsalabs/node.asp?id=2248 and RFC 2631 at ftp://ftp.rfc-editor.org/in-notes/rfc2631.txt.

So, how does public key encryption work? Suppose John would like to exchange a message securely with Chris. Prior to doing so, Chris would provide John with his public key. John would then take the message he wishes to share with Chris and encrypt the message using Chris' public key. When Chris receives the message, he takes his private key and decrypts the message. Chris is then able to read the message John had intended to share with him. However, what if someone intercepts the message and has possession of Chris' public key? Absolutely nothing happens. When messages are encrypted using Chris' public key, they can only be decrypted using the private key associated with that public key.

PKI

PKI is the meshing of encryption technologies, services, and software together to form a solution that enables businesses to secure their communications over the Internet. PKI involves the integration of digital certificates, certificate authorities (CAs), and public key cryptography. PKI offers several enhancements to the security of your enterprise.

PKI gives you the ability to easily verify and authenticate the identity of a person or organization. By using digital certificates, it is easy to verify the identity of

parties involved in a transaction. The ease of verification of identity is also beneficial to access control. Digital certificates can replace passwords for access control, which are sometimes lost or easily cracked by experienced crackers.

Certificates

Digital certificates are nothing more than a way to verify your identity through a CA using public key cryptography. There are certain steps you must take before you can use a certificate to validate your identity. First, you must generate a certificate request from within the VPN appliance. Then, the VPN appliance generates a public/private key pair. You then send a request with the public key to your CA. A response, which incorporates the public key, will be forwarded to you that will have to be loaded into the VPN appliance. This response generally includes three parts:

- The CA's certificate, which contains the CA's public key
- The local certificate identifying your VPN device
- In some cases, a certificate revocation list (CRL), which lists any certificates revoked by the CA

You can load the reply into the VPN device either through the Web UI or via TFTP (Thin File Transport Protocol) through the CLI (command line interface), whichever you prefer. Loading the certificate information into the VPN gives the following:

- Your identity can be verified using the local certificate.
- The CA's certificate can be used to verify the identity of other users.
- The CRL list can be used to identify invalid certificates.

CRLs

A *certificate revocation list*, or CRL, is used to ensure that a digital certificate has not become invalid. VPN appliances use CRLs to check for invalid certificates before connecting VPN tunnels. When using digital certificates with VPNs, the certificate is validated during phase 1 negotiations. In the event no CRL has been loaded into the VPN, the appliance tries to retrieve a CRL via LDAP (Lightweight Directory Access Protocol) or HTTP (Hypertext Transfer Protocol), which is defined inside the CA certificate. Many VPN appliances also allow you to specify an address to refer to for the CRL. If you do not define an address, the default address within the CA's certificate is used.

IPSec

IPSec's main design goals are to provide:

- **Data confidentiality** Encrypt data before transmission.
- **Data integrity** Each peer can determine if a received packet was changed during transit.
- **Data origin authentication** Receiver can validate the identity of a packet's sender.
- **Anti-replay** The receiver can detect and reject replayed packets, protecting it from spoofing and man-in-the-middle attacks.

IPSec Core Layer 3 Protocols: ESP and AH

IPSec provides confidentiality and integrity protection for transmitted information, authentication source and destinations, and anti-replay protection. Two main network protocols, Encapsulating Security Payload (ESP) and Authentication Header (AH), are used to achieve this goal. All other parts of the IPSec standard merely implement these protocols and configure the required technical parameters. Applying AH or ESP to an IP packet may modify the data payload (not always) and may insert an AH or ESP header between the IP header and the packet contents. See Figures 5.9 and 5.10 for illustrations of how these transformations are performed.

Figure 5.9 AH Encapsulation

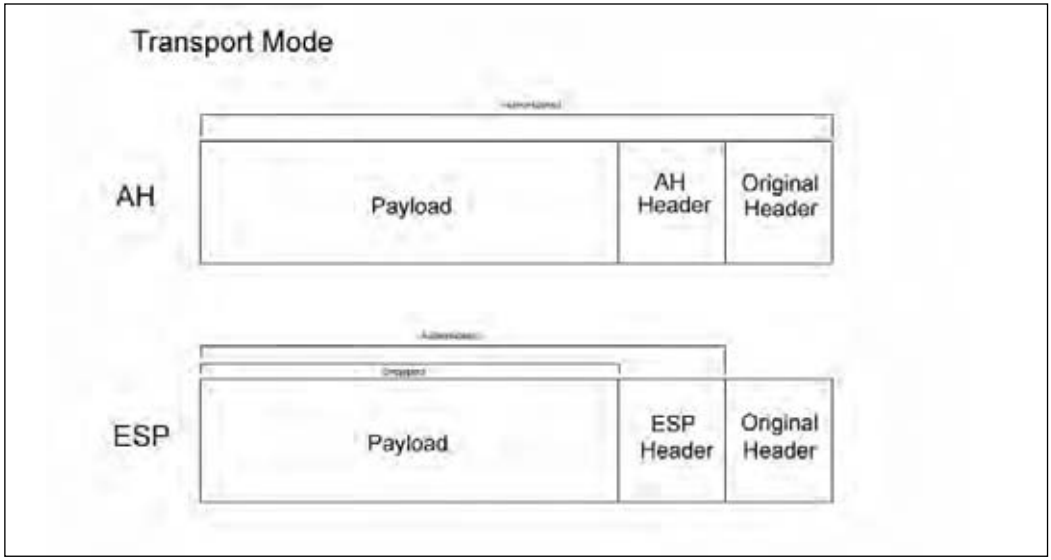
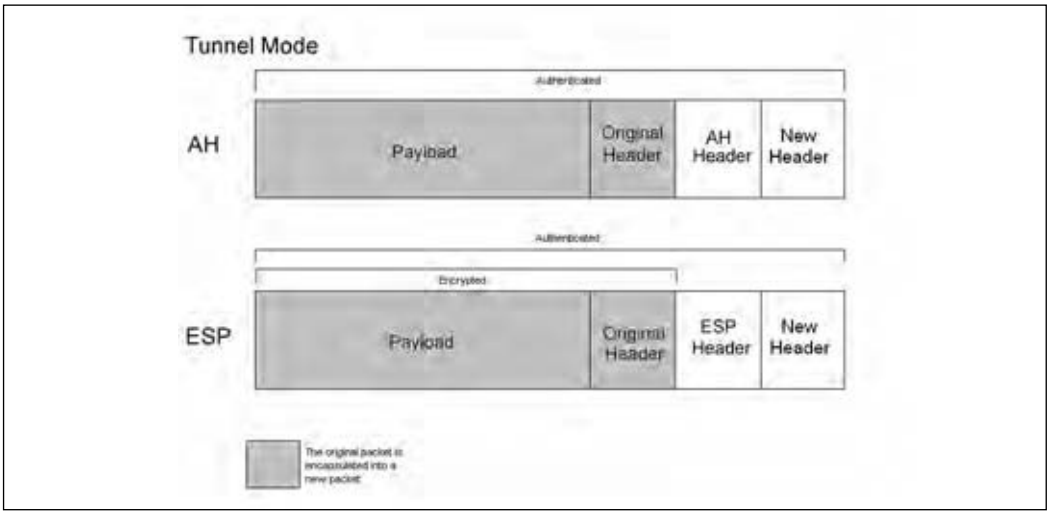


Figure 5.10 ESP Encapsulation



Authentication Header

The AH, which is defined as IP protocol 51, ensures:

- **Data integrity** Calculates a hash of the entire IP packet, including the original IP header (but not variable fields such as the TTL), data payload, and the authentication header (excluding the field that will contain the calculated hash value). This hash, an integrity check value (ICV), can be either Message Authentication Code (MAC) or a digital signature. MAC hashes are more common than digital signatures. Hashing algorithms include MD5 and SHA-1. Both are known as *keyed hashes*, meaning that they use an extra value to calculate the hash, which is known only to the participating parties. When the packet is received, its content, excluding some fields, is hashed by the receiver and the result is compared with the ICV. If they are the same, the packet is declared authentic.
- **Data origin authentication** AH provides source IP authentication. Since the source IP is included in the data used to calculate the hash, its integrity is guaranteed.
- **Replay protection** AH also includes an IPSec sequence number, which provides protection against replay attacks because this number is also included in authenticated data and can be checked by the receiving party.

AH provides no confidentiality because no encryption is used.

NOTE

Pure AH is always broken by NAT. For example, when an authenticated packet goes through an address-translation device, the IP address in its header changes and the Message Authentication Code (MAC) hash calculated by the receiver on a new packet will be incorrect, so the packet will be rejected. It is not possible for a translating gateway to recalculate the new MAC hash and insert it into the packet, because only the endpoints of a transmission know the hashing keys. This was a common problem with IPSec—trying to use AH when NAT is occurring somewhere in the path. It will simply not work.

Encapsulating Security Payload

ESP, which is defined as IP protocol 50, provides:

- Packet padding to prevent traffic analysis, and encrypts the results using ciphers such as DES, 3DES, AES, or Blowfish.
- Optional authentication using the same algorithms as the AH protocol. IP header information is not included in the authenticated data, which allows ESP-protected packets to pass through NAT devices. When a packet is created, authentication data is calculated after encryption. This allows the receiver to check the packet's authenticity before starting the computationally intensive task of decryption.
- Optional anti-replay features.

The original ESP definition did not include authentication or anti-replay, as it was assumed the sender and receiver would use ESP and AH together to get confidentiality *and* authentication. Since ESP can also perform most of the AH functions, there is no reason to use AH. Because ESP works on encapsulation principles, it has a different format: All data is encrypted and then placed between a header and a trailer. This differentiates it from AH, where only a header is created.

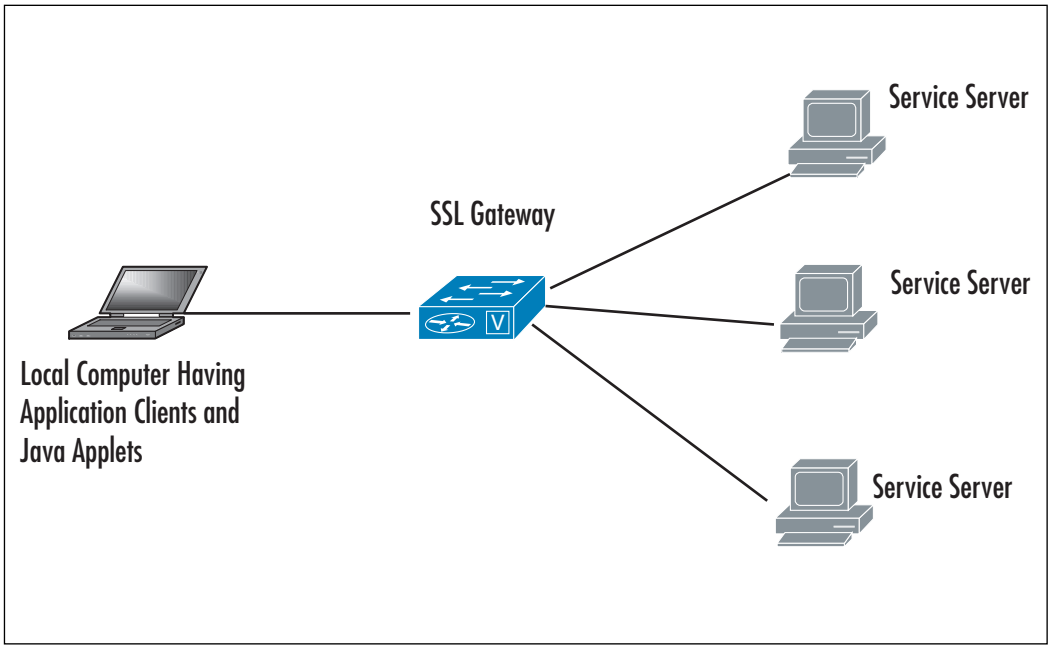
IPSec Communication Modes: Tunnel and Transport

Both AH and ESP can operate in either transport or tunnel mode. In transport mode, only the data portion of an IP packet is affected; the original IP header is not changed. Transport mode is used when both the receiver and the sender are end-

points of the communication—for example, two hosts communicating directly to each other. Tunnel mode encapsulates the entire original packet as the data portion of a new packet and creates a new external IP header. (AH and/or ESP headers are created in both modes.) Tunnel mode is more convenient for site-to-site VPNs because it allows tunneling of traffic through the channel established between two gateways.

In transport mode, the IP packet contains an AH or ESP header right after the original IP header, and before upper layer data such as a TCP header and application data. If ESP is applied to the packet, only this upper layer data is encrypted. If optional ESP authentication is used, only upper layer data, not the IP header, is authenticated. If AH is applied to the packet, both the original IP header and upper layer data are authenticated. Figure 5.11 shows what happens to the packet when IPSec is applied in transport mode.

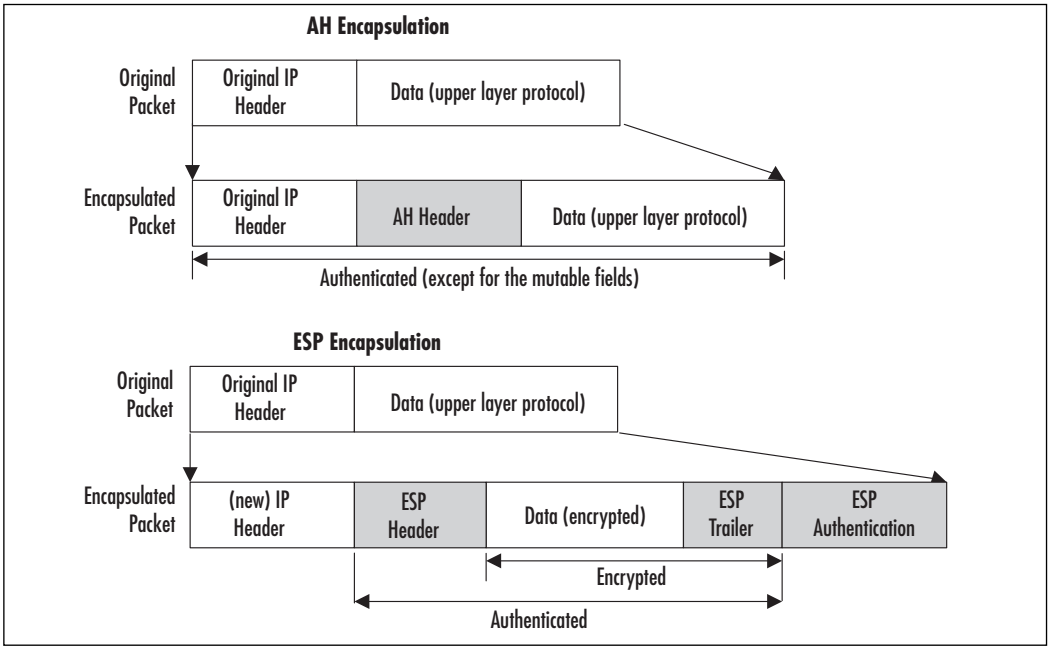
Figure 5.11 Packet Structure in Transport Mode



Tunnel mode is typically used to establish an encrypted and authenticated IP tunnel between two sites. The original packet is encrypted and/or authenticated and encapsulated by a sending gateway into the data part of a new IP packet, and then the new IP header is added to it with the destination address of the receiving gateway. The ESP and/or AH header is inserted between this new header and the

data portion. The receiving gateway performs decryption and authentication of the packet, extracts the original IP packet (including the original source/destination IPs), and forwards it to the destination network. Figure 5.12 demonstrates the encapsulation performed in tunnel mode.

Figure 5.12 Packet Structure in Tunnel Mode



If AH is used, both the original IP header and the new IP header are protected (authenticated), but if ESP is used, even with the authentication option, only the original IP address, not the sending gateway's IP address, is protected. ESP is more than adequate since it is very difficult to spoof an IPSec packet without knowing many technical details. The exclusion of the new IP header from authenticated data also allows tunnels to pass through devices that perform NAT. When the new header is created, most of the options from the original IP header are mapped onto the new one—for example, the Type of Service (ToS) field.

Internet Key Exchange

IPSec protocols use cryptographic algorithms to encrypt and authenticate, and requires encryption/authentication keys. It is possible to configure these keys manually, but there are disadvantages to this approach. First, it is very difficult to scale; second, it is not possible to renegotiate Security Associations (SAs) because they are

fixed until manually changed. Thus, there is a strong need for tools for managing keys and SAs. Key management includes generation, distribution, storage, and deletion of the keys. The initial authentication of the systems to each other and protecting the key exchange is critical. After keys are exchanged, the channel is protected with these keys and is used to set up other parameters, including SAs.

The protocol the IETF adopted for performing these functions is Internet Security Association and Key Management Protocol (ISAKMP), defined in RFC 2408. RFC 2408 describes authenticated key exchange methods. ISAKMP has an IANA-assigned UDP port number of 500. ISAKMP is a generic protocol and is not tied to IPSec or any other key-using protocol.

ISAKMP can be implemented directly over IP or any transport layer protocol. When used with other key management protocols such as Oakley (RFC 2412) and Secure Key Exchange Mechanism (SKEME), we end up with a protocol called the Internet Key Exchange (IKE), which is defined in RFC 2409. Although not strictly correct, the abbreviations IKE and ISAKMP are often used interchangeably.

IKE has two exchange phases, and each can operate in one or two modes. IKE Phase 1 starts when two peers need to establish a secure channel—that is, they do not have IPSec SAs needed for communication over IPSec. This phase includes authentication of systems by each other, agreement on encryption and authentication algorithms used from then on to protect IKE traffic, performing a Diffie-Hellman (DH) key exchange, and finally, establishing an IKE Security Association (IKE SA). IKE SAs are bidirectional; each IKE connection between peers has only one IKE SA associated with it.

IKE Phase 2 negotiates one or more IPSec SAs, which will be used for the IPSec tunnel between these peers. It uses key material from IKE Phase 1 to derive keys for IPSec. One peer tells the other which traffic it wants to protect and which encryption/authentication algorithms are supported. The second peer then agrees on a single protection set for this traffic and establishes the keys.

While implementing different phases adds processing overhead, there are advantages to this approach:

- Trust between peers is established in the first phase and used in the second phase.
- Key material established in the first phase can be used in the second phase.
- Renegotiations of the first phase can be assisted by the second-phase data.

Phase 1 has two modes: main and aggressive. Main mode uses three exchanges between peers; each exchange consists of two messages, a request, and a reply:

- The first exchange in main mode negotiates parameters to protect the IKE connection. The initiating side sends a proposal to its counterpart, and includes parameters it supports. These parameters include one encryption algorithm (DES, 3DES, etc.) and one of three authentication algorithms: preshared secret, RSA public key encryption with Diffie-Hellman exchange group 1 and 2, or public key RSA signature (this includes use of certificates). The other peer then selects and accepts a single pair from the offered set. If there is no match or agreement, the IKE tunnel cannot be established.
- The second exchange in main mode performs DH key establishment between peers. It exchanges two values called nonces, which are hashes that only the other party can decrypt. This confirms that the message is sent by the same hosts as the previous exchange.
- The third and last exchange authenticates the peers using the agreed-on methods: public keys signatures, public key encryption, or a preshared secret. This exchange is protected by an encryption method that was selected in the first exchange.

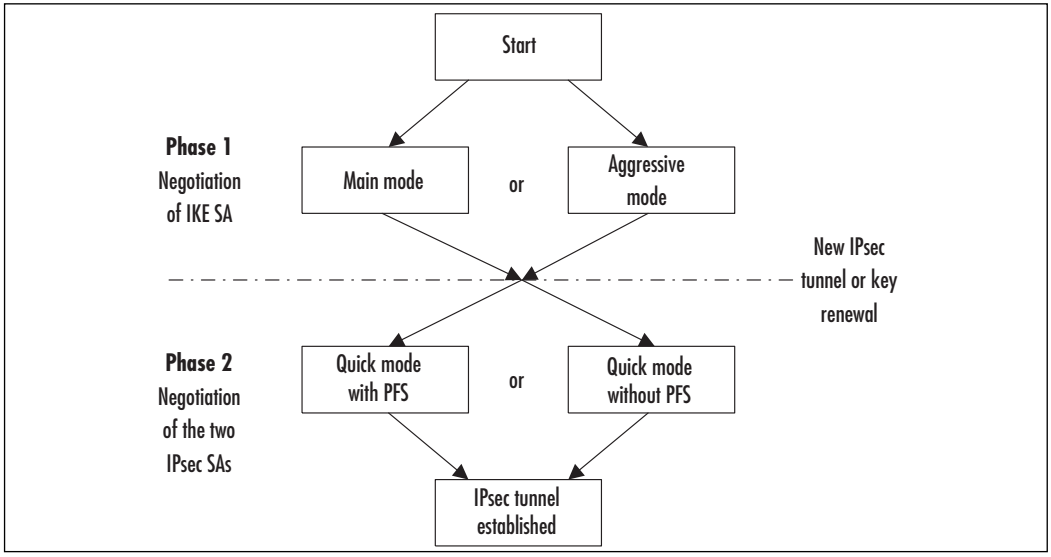
RFC 2408 provides more details on the packet format and algorithms used. At the end of the first phase, each host has an IKE SA, which specifies all parameters for this IKE tunnel: the authentication method, the encryption and hashing algorithm, the Diffie-Hellman group used, the lifetime for this IKE SA, and the key values.

Aggressive mode exchanges only three packets instead of six, so it is faster but not as secure. Fewer packets are sent because the first two packets in this exchange include almost everything in one message; each host sends a proposed protection set, Diffie-Hellman values, and authentication values. The third packet is sent only for confirmation and after the IKE SA is already established. The weakness in aggressive mode is that everything is sent in clear text and can be captured. However, the only thing the attacker can achieve is to DoS one of the peers, because it is not possible to discover the keys that are established by the Diffie-Hellman protocol. There have been recent attacks against VPN endpoints that relied on the properties of aggressive mode.

The most important mode of Phase 2 is quick mode. It can be repeated several times using the same IKE SA established in Phase 1. Each exchange in this mode establishes two IPSec SAs by each peer. One of these SAs is used for inbound protection, and the other is used for outbound protection. During the exchange, peers agree on the IPSec SA parameters and send each other a new nonce, which is used for deriving Diffie-Hellman keys from the ones established in Phase 1. When the

IPSec SA lifetime expires, a new SA is negotiated in the same manner. Figure 5.13 summarizes the flow of the IKE protocol.

Figure 5.13 IKE Phases and Modes



NOTE

Quick mode can use Perfect Forward Secrecy (PFS). PFS dictates that new encryption keys are not derived from previous ones, so even if one key is discovered, only the traffic protected by that key will be exposed. PFS is achieved by performing a new Diffie-Hellman key establishment in each quick mode.

Security Associations

Previous sections assumed that an IPSec connection was already established and all parameters such as authentication and encryption keys were known to both parties. The data flow in each direction is associated with an entity called a *security association* (SA). Each party has at least two IPSec SAs: the sender has one for outgoing packets and another for incoming packets from the receiver, and the receiver has one SA for incoming packets from the sender and a second SA for outgoing packets to the sender.

Each SA has three parameters:

- The Security Parameter Index (SPI), which is always present in AH and ESP headers
- The destination IP address
- The IPSec protocol, AH or ESP (so if both protocols are used in communication, each has to have its own SA, resulting in a total of four SAs for two-way communication)

Each peer maintains a separate database of active SAs for each direction (inbound and outbound) on each of its interfaces. This database is known as the Security Association Database (SAD). SAs from these databases decide which encryption and authentication parameters are applied to the sent or received packet. SAs may be fixed for the time of traffic flow (called *manual IPSec* in some documents), but when a key management protocol is used, they are renegotiated many times during the connection. For each SA, the SAD entry contains the following data:

- The destination address
- The SPI
- The IPSec transform (protocol and algorithm used—for example; AH, HMAC-MD5)
- The key used in the algorithm
- The IPSec mode (tunnel or transport)
- The SA lifetime (in kilobytes or in seconds); when this lifetime expires, the SA must be terminated, and a new SA established
- The anti-reply sequence counters
- Some extra parameters such as Path MTU

The selection of encryption parameters and corresponding SAs is governed by the Security Policy Database (SPD). An SPD is maintained for each interface and is used to decide on the following:

- Selection of outgoing traffic to be protected
- Checking if incoming traffic was properly protected
- The SAs to use for protecting this traffic
- What to do if the SA for this traffic does not exist

The SPD consists of a numbered list of policies. Each policy is associated with one or more selectors, which are implemented as an access-lists. A *permit* statement means that IPSec should be applied to the matching traffic; a *deny* statement means that the packet should be forwarded without applying IPSec. The resulting map and a crypto access-list are applied to the interface, creating an SPD for this interface.

For outgoing traffic, when IPSec receives data to be sent, it consults the SPD to determine if the traffic has to be protected. If it does, the SPD uses an SA that corresponds to this traffic. If the SA exists, its characteristics are taken from the SAD and applied to the packet. If the SA does not exist yet, IKE establishes a new SA to protect the packet.

For incoming IPSec traffic, the SPI is culled from the AH or ESP header to find a corresponding SA in the SAD. If it does not exist, the packet is dropped. If an SA exists, the packet is checked/decrypted using the parameters provided by this SA. Finally, the SPD is checked to ensure this packet was correctly protected—for example, that it should have been encrypted using 3DES and authenticated with MD5 and nothing else.

Designing & Planning…

Cryptographic Algorithms in IPSec and Their Relative Strengths

Three types of cryptography algorithms are used in all IPSec implementations:

- Encryption
- Message authentication
- Key establishment

Encryption algorithms encipher clear-text messages, turning them into cipher text and deciphering them back to their original content via cryptographic keys. The simplest type of encryption algorithms is *symmetric encryption* where messages are encrypted and decrypted using the same key. This key must be kept a secret and well protected; otherwise, anybody can decrypt and read the message. The longer the key, the more difficult it is to "crack."

DES is an example of symmetric encryption. DES was adopted by the U.S. government as an official standard, but has now adopted the Advanced Encryption Standard (AES) for much stronger encryption. DES is obsolete and weak since messages encrypted with standard 56-bit DES can easily be cracked.

Continued

Triple DES (3DES) is a better solution, as it encrypts a message three times using DES, each time using a different 56-bit key. 3DES is still considered a strong cipher, although we see it being phased out in favor of AES.

Public-key cryptography uses complex exponential calculations and appears slow compared with symmetric-key ciphers such as 3DES or AES-128. Public-key cryptography uses two keys: one for encryption and a completely separate one for decryption. Only the decryption key (known as the *private key*) needs to be kept secret; the encryption key (known as the *public key*) can be made public. For example, if anyone wants to send Alice an encrypted message, he can use her public key to encrypt the message, but only Alice knows the key that allows her to decrypt the message. One widespread algorithm based on public keys is the Rivest, Shamir, and Adelman (RSA) algorithm.

Message authentication algorithms protect the integrity of a message. IPSec uses two types: keyed message hash algorithms and public signature algorithms. *Keyed message hashing* combines a message with a key and reduces it to a fixed-length digest. (Adding a key gives these algorithms the name *keyed*.) A *hashing algorithm* makes it almost impossible to create a spoofed message that will yield the same digest as the original message. When a receiver wants to ensure the message was not altered in transit, it performs the same calculation on the message and compares the result with the received digest. If they are the same, the message is authentic; a spoofed one would have a different digest.

IPSec uses MD5, which produces 128-bit output, and the stronger SHA-1, which produces 160-bit output. Although SHA-1 is cryptographically stronger than MD5, it requires more processing to compute the hash. IPSec uses modified versions of each, HMAC-MD5 and HMAC-SHA-1, which perform hashing twice, each time differently combining the message with the key.

Key establishment protocols securely exchange symmetric keys by both sides via an insecure medium (such as the Internet). In IPSec, this task is accomplished using the Diffie-Hellman (DH) algorithm. DH is based on exponential computations. During the process, both sides exchange digits, allowing both peers to derive the same key, but nobody who sees these numbers can do the same. DH in IPSec can work with keys of different lengths: 768-bit (DH Group 1), 1024-bit (DH Group 2), and 1536-bit (DH Group 5). Group 5 keys are stronger, but require more processing power.

Pros of IPSec

The IPSec protocol, as defined by the IETF, is "a framework of open standards for ensuring private, secure communications over Internet Protocol networks, through the use of cryptographic security services." This means that IPSec is a set of standards used for encrypting data so it can pass securely through a public medium, such as the Internet. Unlike other methods of secure communications, IPSec is not bound to any

particular authentication method or algorithm, which is why it is considered an "open standard." In addition, unlike older security standards that were implemented at the application layer of the OSI model, IPSec is implemented at the network layer.

NOTE

Remember that IPSec is implemented at the network layer, not the application layer.

The advantage to IPSec being implemented at the network layer (versus the application layer) is that it is not application-dependent, meaning users do not have to configure each application to IPSec standards.

IPSec can be used to secure any protocol that makes use of IP. It also enjoys the support of the medium over which IP runs. Other encryption schemes to secure data, like PGP, expect a user to remember his or her passphrase, ensure the passphrase is safe, and the user must follow procedures to validate the correspondent's keys. IPSec is independent of the overhead in terms of expectation from a user to secure data. It is transparent to a user. IPSec authentication mechanism also provides prevention against many attacks on a high-level protocol. For example, a man-in-the-middle attack is not possible for an application using IPSec.

Cons of IPSec

The IPSec protocol is an open protocol. The different design choices among different vendors have often resulted in IPSec-compliant products that differ from each other, which will cause these products to not operate with each other. IPSec-based VPN is tightly coupled with the operating system, so there is a longer packet processing time. IPSec has been designed to provide authentication between computers. It does not provide the concept of user ID, or support authentication of users, which is required for many other security mechanisms. If we want to design some sort of access control to our e-mail server or database server, a non-IPSec mechanism will be desired. IPSec provides encryption at the IP layer between two computers, which again is different from encrypting messages between users or between applications. For example, to secure e-mail, PGP is still preferred.

To ensure the integrity of data being transmitted using IPSec, there has to be a mechanism in place to authenticate end users and manage secret keys. This mechanism is called Internet Key Exchange (IKE). IKE is used to authenticate the two ends of a secure tunnel by providing a secure exchange of a shared key before IPSec transmissions begin.

For IKE to work, both parties must use a password known as a *pre-shared* key. During IKE negotiations, both parties swap a *hashed* version of a pre-shared key. When they receive the hashed data, they attempt to recreate it. If they successfully recreate the hash, both parties can begin secure communications.

IPSec also has the capability to use digital signatures. A digital signature is a certificate signed by a trusted third party (CA) that offers authentication and *nonrepudiation*, meaning the sender cannot deny that the message came from him. Without a digital signature, one party can easily deny he was responsible for messages sent.

Although *public key cryptology* ("User A" generates a random number and encrypts it with "User B's" *public key*, and User B decrypts it with his *private key*) can be used in IPSec, it does not offer nonrepudiation. The most important factor to consider when choosing an authentication method is that *both parties must agree on the method chosen*. IPSec uses an SA to describe how parties will use AH and encapsulating security payload to communicate. The security association can be established through manual intervention or by using the Internet Security Association and Key Management Protocol (ISAKMP). The Diffie-Hellman key exchange protocol is used for secure exchange of pre-shared keys.

Certain fields like source and destination gateway address, packet size, and so forth in IPSec can be used for traffic analysis. IPSec is prone to traffic analysis. IPSec cannot provide all the functionality of other security protocol working at upper layers. For example, IPSec cannot be used to digitally sign a document. IPSec and the applications that make use of IPSec are still prone to DoS attacks. Another serious drawback of IPSec VPN is the inability to work behind NAT devices. The authentication header in the IPSec mode hashes the source addresses during the authentication process. If NAT changes the source address, the VPN on the other end will see a different hash when it receives the packet. It will drop the packet, thinking it has been tampered with. Errors due to mismatched hashes because of a changed address can be avoided by running IPSec in tunnel mode using only Encapsulating Security Payload (ESP). IPSec cannot be used with non-IP protocols like AppleTalk, IPX, NetBIOS, and DECnet.

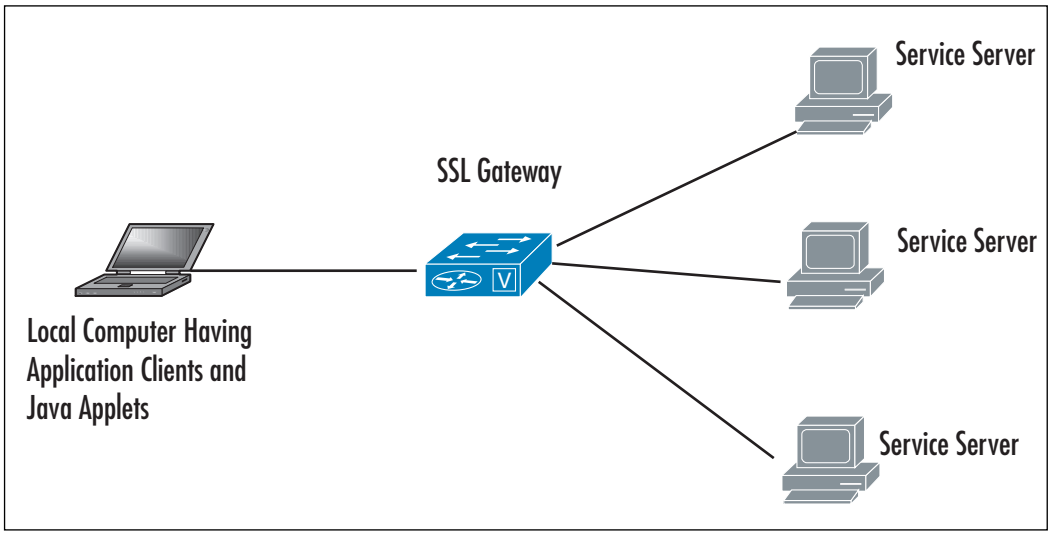
SSL VPNs

Many years ago, accessing corporate resources and being productive while away from the office was a dream. With the advent of the IPSec VPN, accessing resources remotely is becoming a reality. However, using IPSec, company had several hundred or even a thousand employees who all needed remote access. There was software to install and update the policies to create. Generally speaking, when you deploy IPSec client software you must also purchase licenses. This can become extremely costly if

you have a fairly large user base. The ability to access a company's resources while on the go is now at an all-time high.

This is where SSL VPN comes into play. SSL VPN allows you to secure your internal resources behind a single entry point device; the remote users only require a Web browser capable of SSL encryption. The user connects to the SSL-VPN gateway and begins his or her secure session. At this point, the user can access many different types of resources. This provides secure ubiquitous client access and because you don't have to deploy a client, you can easily deploy access to thousands of users in a matter of hours (Figure 5.14).

Figure 5.14 SSL-Based VPN



Technical Description

A secure tunnel between computers provides secure communication channel between two computers. SSL uses asymmetric cryptography to share secrets between the local computers and then uses symmetric keys to encrypt the communication between the SSL gateways. To rehash, an encrypted tunnel between two computers over an insecure network such as the Internet is known as a virtual private network. SSL-VPN thus creates a secure tunnel by making sure both the users are authenticated before allowing access, and encrypting all data transmitted to and from the users by using SSL.

Earlier, we discussed the IPSec-based VPN. The difference between the IPSec-based VPN and the SSL-based VPN is that IPSec operates at the IP layer or at net-

work layers, and SSL-VPN establishes connection using SSL, which works at the transport and session layers. They can also encapsulate information at the presentation and application layers. Thus, you can see that SSL-based VPN is the most versatile.

SSL between client and server as shown in Figure 5.14 can in turn be divided into two phases: handshake and data exchange. The handshake phase between the local machine and the server requires three phases.

First Phase

During the first phase, client and server exchange hello, which in turn enables the client and server to exchange information about the encryption ciphers and the compression algorithms.

- **Client's hello** Comprised of protocol version supported, Session ID, list of supported data and key encryption ciphers, supported compression methods, and a nonce.
- **Server's hello message** Protocol version to be used, Session ID, one cipher for data and one for key exchange, one compression method and a nonce.

Based on the cryptography and compression algorithms, the client and server decide to cancel or proceed with the session. The next handshake phase involves authentication and key exchange between both the parties.

Second Phase

The second phase involves the authentication**,** between client and server, and is done by exchanging digital certificates.

Server's authentication Server certificate or Server's public key, certificate request, "hello done" notification.

Client's authentication Clients certificate or client's public key, certificate verification.

A digital certificate is issued and signed by the private key of the CA and comprises the following:

- Owners public key
- Owner's name
- Expiration date of the public key

- Name of the issuer (the CA that issued the digital certificate)
- Serial number of the digital certificate
- Digital signature of the issuer

The CA can be some trusted third party such as VeriSign. The client must possess the public keys of the trusted party to verify that it has the public keys of the correct server. Digital certificates then help in handing over the public keys in a secure manner. The client will then use the public keys of the server to encrypt a pre-master secret and send it to the server. This pre-master secret is then used to generate a master secret, which aids in the generation of symmetric keys for data exchange. The symmetric keys between client and the server are then used to encrypt data.

Third Phase

In the third phase, client and server wrap up the communication. Closing communication is performed by sending a 1-byte value that conveys finished notification.

Server Finish is comprised of change cipher spec, which is a 1-byte value, "finished notification." Client Finish in turn is comprised of change cipher spec and "finished" specifications. Once the client and server have finished authentication, the next stage involves the data exchange stage of SSL, which involves various stages.

First, data is fragmented into 18kB and then compressed. After compression, SSL appends a message authentication code MAC to the compressed data:

```
MAC{data} = hash { secret_key + hash{ secret_key  + data + time_stamp}}
```

The message authentication code is added to the packet and is then forwarded to the next layer, which involves encryption of the message. After encryption is complete, the SSL header is added to the packet and sent to the SSL layer. The packet is ready to be sent to the other side.

SSL Tunnels in Linux

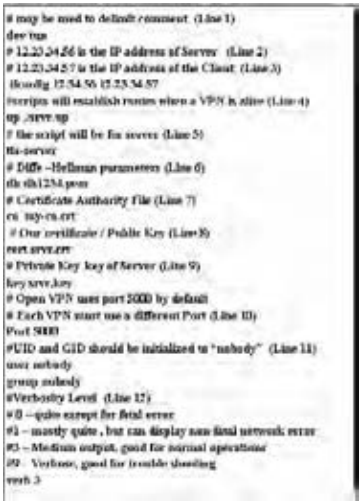
One of the most commonly used open source SSL VPNs is Open VPN, which uses TAP and TUN virtual drivers. For Linux version 2.4.*x* or later, these driver are already bundled with the kernel. Open VPN tunnels traffic over the UDP port 5000. Open VPN can either use TUN driver to allow the IP traffic or TAP driver to pass the Ethernet traffic. Open VPN requires configuration to be set in the configuration files. Open VPN has two secure modes. The first is based on SSL/TLs security using public keys like RSA, and the second is based on using symmetric keys or pre-shared secrets. RSA certificates and the keys for the first mode can be generated by

using the **openssl** command. Details about these certificates or the private keys are stored in our *.cnf files to establish VPN connection.

The .crt extension will denote the certificate file, and .key will be used to denote private keys. The SSL-VPN connection will be established between two entities, one of which will be a client, which can be your laptop, and the other will be a server running at your office or lab. Both these computers will have .conf files, which define the parameters required to establish SSL-VPN connection.

For the server side, let's call the file tls-srvr.conf, details of which are shown in Figure 5.15.

Figure 5.15 Configuration of the *.conf File on Server Side



The configuration of srvr.up, which is mentioned after line 4, is shown in Figure 5.16.

The *.cnf file (let's call it clt.cnf) on the client side will look similar to Figure 5.12. However, there will be modifications in some of the parameters in the file. After line 3, the parameters of ifconf will change to ifconfig 12.1.0.2 12.1.0.1 # from client side to server side. 12.23.34.57 is the IP address of the client, and # 12.23.34.56 is the IP address of the server.
After line 4, modification will be

```
up ./cnt.up
```

After line 5, modification will be

```
tls-client
```

Figure 5.16 Configuration of the srvr.up File

```
# !/bin/bash
route add –net  12.0.1.0 netmask 255.255.255.0 gw $5
```

Again, the certificate on the client side will point to the certificate of the client. If local.crt is storing the certificate of client and the private key of client is key local.key, then

```
cert home.crt
key local.key
```

will have to be added after line 8 and line 9.
The remaining part of the configuration file for the client side will remain the same.

The configuration of the clt.up to start a VPN server is shown in Figure 5.17.

Figure 5.17 Configuration of the clt.up File

```
#!/bin/bash
route add –net 12.23.34.0 netmask 255.255.255.0 gw $5
```

Once these files are configured, to start a VPN at the server side execute the command

$ open vpn –config tls-srvr.cnf

and similarly to start at the client side, use
$ openvpn –config tls-clt.cnf

Pros

SSL VPN is one way to transfer the information since a web browser can be used to establish an SSL VPN connection. Since SSL VPN is clientless, it will result in cost savings and can be configured to allow access from corporate laptops, home desktops, or any computer in an Internet café. SSL VPNs also provide support for authentication methods and protocols, some of which include:

- Active Directory (AD)
- Lightweight Directory Access Protocol (LDAP)
- Windows NT LAN Manager (NTLM)
- Remote Authentication Dial-In User Service (RADIUS)
- RSA Security's RSA ACE/Server and RSA SecurID

Many SSL VPNs also provide support for single sign-on (SSO) capability. More sophisticated SSL VPN gateways provide additional network access through downloadable ActiveX components, Java applets, and installable Win32 applications. These add-ons help remote users access a wide range of applications, including:

- Citrix MetaFrame
- Microsoft Outlook
- NFS
- Remote Desktop
- Secure Shell (SSH)
- Telnet

However, note that not all SSL VPN products support all applications.

SSL VPN can also block traffic at the application level, blocking worms and viruses at the gateway. SSL VPN is again not bound to any IP address; hence, unlike IPSec VPN, connections can be maintained as the client moves. SSL VPN differs from IPSec VPN in that it provides fine-tuned access control. By using SSL VPN, each resource can be defined in a very granular manner, even as far as a URL. This feature of SSL VPN enables remote workers to access internal Web sites, applications, and file servers. This differs from IPSec VPN, since the entire corporate net-

work can be defined in a single statement. SSL-based VPN uses Secure HTTP, TCP port 443. Many corporate network firewall policies allow outbound access for port 443 from any computer in the corporate network. In addition, since HTTPS traffic is encrypted, there will be limited restrictive firewall rules for SSL VPN.

Cons

As you know, SSL-based VPN offers a greater choice of client platforms and is easy to use. However, an organization that wants to be sure their communication channel is encrypted and well secured will never assume that any computer in an Internet café is trusted. This in turn requires a trust association with an un-trusted client connection. To address the concern of an untrusted client, whenever a client from an untrusted platform connects to the VPN, a small java applet is downloaded to the client that searches for malicious files, processes, or ports. Based on the analysis of the computer, the applet can also restrict the types of client that can connect. This may sound feasible theoretically; practically, it requires the mapping of policies of one anti-virus and anti-spyware tool into an endpoint security tool used by VPN. In addition, these applets are prone to evasion and can be bypassed. However, note it carefully; you also need to have administrative access to perform many of the operations like deleting temporary files, deleting cookies, clearing cache, and so forth. If you have administrative rights in an Internet café, be assured that the system will be infected with keystroke loggers, sophisticated malicious remote access tools like Back Orifice using ICMP as a communication channel and RC4 to encrypt the payload.

By using SSL VPN, a user can download sensitive files or confidential, proprietary corporate data. This sensitive data has to be deleted from the local computer when an SSL VPN is terminated. To ensure the safety of confidential data, a sandbox is proposed and used. A sandbox is used to store any data downloaded from a corporate network via SSL VPN. After the SSL VPN session is terminated, the data in the sandbox is securely deleted. After a session is terminated, all logon credentials require deletion as well. You know that SSL VPN can be established even from a cyber café. It might happen that a user can leave the system unconnected. To prevent such issues, periodic authentication is required in some systems. As SSL VPN works on the boundary of Layers 4 and 5, each application has to support its use. In IPSec VPN, a large number of static IP address can be assigned to the remote client using RADIUS. This in turn provides the flexibility to filter and control the traffic based on source IP address. In the case of SSL VPN, the traffic is normally proxies from a single address, and all client sessions originate from this single IP. Thus, a network administrator is unable to allocate privileges using a source IP address. SSL-based VPN allows more firewall configurations as compared to IPSec VPN to control access to internal resources. Another cause of concern with SSL-based VPN is packet

drop performance. IPSec will drop the malformed packet at the IP layer, whereas SSL will take it up the layer in the OSI model before dropping it. Hence, a packet will have to be processed more before it is dropped. This behavior of SSL-based VPN can be misused, used to execute DoS attacks, and if exploited, can result in a high capacity usage scenario.

Layer 2 Solutions

A Layer 2 solution from Microsoft and Cisco makes use of both the Point-to-Point Protocol and Cisco Layer 2 protocols. Since the Layer 2 VPN solution provides a significant amount of revenue for the independent local exchange carriers (ILECs) and PTT (Post, Telephone, and Telegraph) service providers, the need for Layer 2 VPN has been increasing. However, the connections for a Layer 2 solution are costly, and the customers want more effective cost solutions. To aid customers, ILECS and PTT are using more effective solutions such as Multiprotocol Label Switching (MPLS), which offers Layer 2 VPN services. L2TP, as the name suggests, operates at the data link layer of the OSI networking model. L2TP is discussed in more detail in the following section. In the Layer 2 VPN solutions, there is no separate private IP network over which traffic is sent. Layer 2 VPNs take existing Layer 2 traffic and send it through point-to-point tunnels on an MPLS network backbone. Layer 2 MPLS VPNs are also called as Transparent LAN Services (TLS) or VPLS Virtual Private LAN Services.

Some vendors who provide MPLS VPN include Avici Systems (www.avivi.com), Cisco Systems (www.cisco.com), CoSine Communications (www.cosineco.com), Juniper Networks (www.juniper.com), Lucent Technology (www.lucent.com), Nortel Networks (www.nortelnetworks.com), and Riverstone Networks (www.riverstonenetworks.com).

L2TP

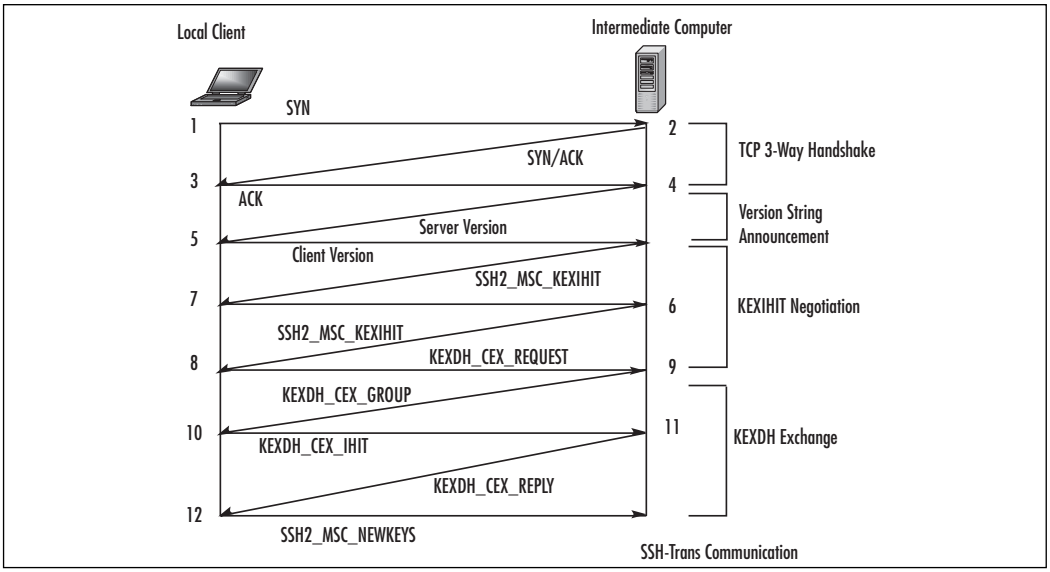
L2TP is a combination of PPTP and Layer 2 Forwarding (L2F), put forth by Cisco Systems. L2TP can encapsulate PPP frames just as PPTP can, but in contrast can then be sent over IP, ATM, or frame relay. It is rather more complicated than PPTP, and more secure.

The IPSec Encapsulating Security Payload (ESP) protocol is used to encrypt L2TP traffic. As you can see in Figure 5.18, one advantage of IPSec is that it encrypts more than just the PPP data packet.

As to security, L2TP is extremely strong. In addition to requiring user authentication through PPP, L2TP requires machine authentication via certificates. Although

certificates are covered in Chapter 3, you need to understand the following requirements for an L2TP implementation of a LAN-to-LAN VPN. First, a user certificate needs to be installed on the calling router, and a computer certificate needs to be installed on the answering router.

Figure 5.18 An L2TP Packet



TIP

If the answering router is a member server in a domain, a computer certificate is required for L2TP. However, if the router is a domain controller (DC), a DC certificate is needed.

PPTP versus L2TP

When choosing which layering protocol to use for a secure VPN, you should understand some of the differences between them. One of the largest differences between PPTP and L2TP is the method of encryption each uses. PPTP uses MPPE, and L2TP uses IPSec ESP.

When PPTP negotiations happen between a client and the VPN server, the authentication phase is not encrypted, even when using the strongest form of MPPE (128-bit RSA RC4). IPSec encryption, however, is negotiated even before the L2TP

connection is established. This allows the securing of both data and passwords. Moreover, IPSec can be configured to use Triple DES (3-DES), which is based on three separately generated 56-bit keys, for true 168-bit encryption. It is the strongest encryption method natively supported by Windows Server 2003.

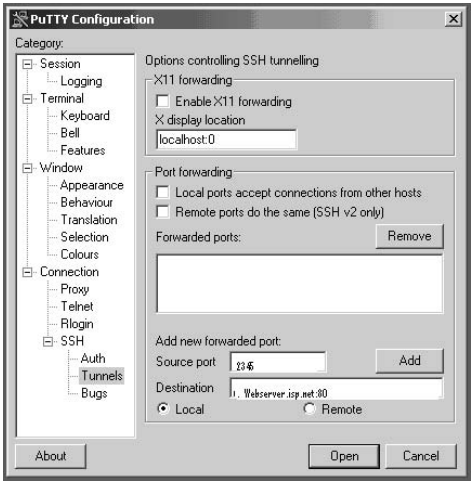
Another consideration when choosing between L2TP and PPTP is how to implement packet filtering. In RRAS, packet filters can be implemented through the external interface's property sheet, located in the General IP Routing section. To allow only PPTP traffic through, the VPN server requires the dropping of all traffic except TCP port 1723 and protocol ID number 47 on both the input and output filters. L2TP, however, is more complicated. It requires the dropping of all traffic except UDP ports 500, 4500, and 1701.

Even though the implementation of L2TP is more administrative work than PPTP, it is recommended for all high-security environments. However, keep in mind that both L2TP and PPTP can be used on the same VPN server. It is also recommended that you use packet filtering and firewalls on all LAN-to-LAN and remote access VPNs.

Technical Description for MPLS

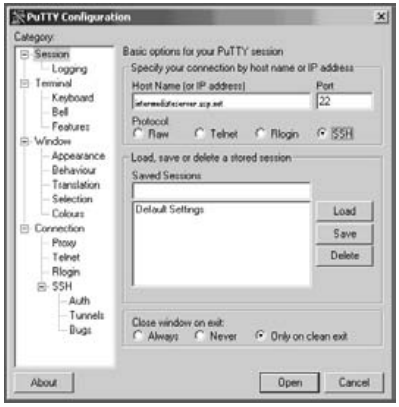
Figure 5.19 shows the architecture for Layer 2 in a Layer 2 VPN. For the rest of the discussion about a Layer 2 solution, CE will represent the customer edge router, and PE will correspond to the provider edge router. PE performs the functionality of egress/ingress routing. The devices that perform the functionality of transit routing are called as provider routers, or P. Provider routers are less complex than PE.

Figure 5.19 The Connection between Different Provider Edge Routers when There Are Three Customers' Sites



As shown in Figure 5.19, in a Layer 2 solution, traffic is forwarded to the provider edge PE router in a Layer 2 format. Interior Gateway Protocol (IGP) or static routes are enabled on the provider edge routers. The traffic is carried in MPLS format over the provider's network and is converted back to the Layer 2 traffic at the sending computer. MPLS works by pre-pending packets with an MPLS header, containing one or more "labels"—called a label stack. Figure 5.20 shows the structure of the MPLS stack. The label stacks as shown in Figure 5.20 contain four fields. The first field is a 20-bit label value. The next field is of size 3 bits; currently, this is reserved for any future use. Following the EXP field is 1-bit stack flag. If the stack flag is set (s=1), it signifies the current label is the last. Following the stack flag, is an 8-bit TTL (time to live) field.

Figure 5.20 MPLS Packet Structure



Instead of lookup in the IP Tables, MPLS packets are forwarded by label lookup. When the ingress router encounters an unlabeled packet, it inserts the MPLS header. The packet is then forwarded to the next hop. The MPLS router, based on the contents of the MPLS packet, can perform three operations: SWAP, PUSH, or POP. The routers can also have built-in lookup tables that in turn can aid in deciding which kind of operations to perform based on the topmost label of the incoming packet so they can process the packet very quickly. In a PUSH operation, a new label is pushed on to the top of the label. This in turn aids in hierarchical routing of packets. For a SWAP operation, the packet label is replaced with the other label. For POP operation, the packet label is removed. The process of removing the label from the MPLS header is called *decapsulation*. At the egress router, the popped label is the last label of the packet. When the last label is removed from the MPLS packet, the packet contains only the payload. Therefore, the egress router must contain the

information about the routing of the packet without any label lookup. In a Layer 2 VPN, IPSec, and more specifically its ESP protocol, provides the encryption for L2TP tunnels. L2TP also requires digital certificates, which in turn also computer authentication.

Pros

A Layer 2 solution service provider provides only a Layer 2 solution to the customers. Hence, in a Layer 2 solution, routing of the packets, which is done at Layer 3, is the responsibility of the customer or the local host. This in turn results in privacy of routing, and customers are free to choose their own Layer 3 protocol. Also notice that overhead in maintaining information on the service provider router is also reduced in terms that they will not have to do anything to keep a customer's route separate from other customers or from the Internet. As shown in Figure 5.12, each PE in Layer 2 will transfer small information about every CE, that it is connected to every PE. Each PE will have to keep information from each CE in each VPN and keep a single "route" to every site in every VPN. In a Layer 2 VPN, if customers believe the Layer 2 service is insecure, they can use IPSec on top of a Layer 2 solution.

Cons

The important problem with Layer 2 VPNs is that they will tie up the service provider VPN to Layer 2 circuits; for example, x.25, frame relay, and ATM (Asynchronous Transfer Mode). If there are n local hosts, and each is connected to each other (i.e., meshed network), the complexity of configuring is O (n*n), and is exponential in nature. Therefore, as the number of local hosts increases, the complexity of configuration increases exponentially. For n CEs, n*(n–1) /2 DLCI PVC must be provisioned across the service provider network, and at each CE, (n–1) DLCIs must be configured to reach each of other CEs. In addition, when a new CE is added, n new DLCI PVCs must be provisioned. Existing CEs must also be updated with a new DLCI to reach the new CE. (See the upcoming "Notes from the Underground" sidebar for more information on PVC, DLCI, and CDs.)

The Layer 2 solution is costly for the provider, and hence the topologies in a Layer 2 solution can be dictated by the cost rather than traffic patterns. Multiple Layer 2 solutions can result in an increase of administrative costs. In a Layer 2 VPN, if a CE is under the control of a customer, he may decide to use IPSec to secure his communication channel. However, the overhead involved in providing this extra security can result in slightly slower performance than PPTP. The client has to perform two

authentications for dial-in users with the VPN carrier L2TP model; one when it encounters VPN carrier POP, and on contact with Enterprise gateway security.

Notes from the Underground…

What Are PVC (Permanent Virtual Circuits) , DLCI (Data Link Connection Identifier), and CE (Customer Edge Router)?

PVC provides frame relay service. It is a data link connection that is predefined on the both ends of the connection. The actual path taken through the network may alter; however, the beginning and end point of the circuits remain the same.

PVCs are identified by the DLCI, which is a 10-bit channel number attached to a data frame that aids in routing the data. Frame relays are multiplexed statistically, which results in transmission of one frame at a time. The DLCI, helps in logical connection of data to the connection; when a data goes to the network, the network knows where to send it.

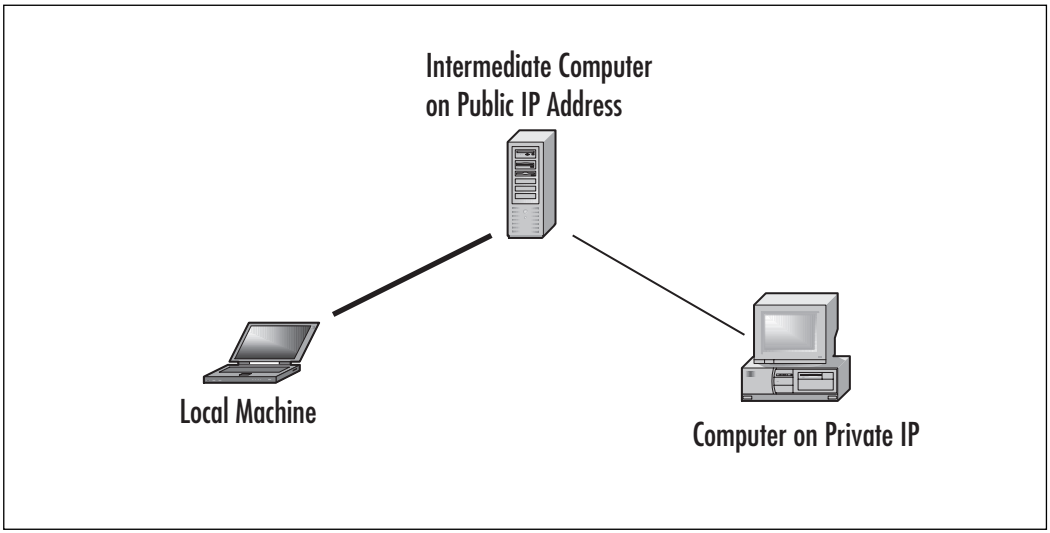
A CE router interfaces the customer network with the provider network. Using it, a customer can limit the number of MAC addresses to the provider network.

SSH Tunnels

Let's take the case of an organization in which all computers on the network have public IP addresses. This means that you can access any computer from anywhere in the world. This definitely is convenient for the mobile workforce or the employees because they can directly connect to the computers in their offices, research labs, and so forth (see Figure 5.21).

Public IP addresses can also cause problems. Since the computers on public IP addresses are universally accessible, they could be attacked by anyone on the global Internet. These computers could be attacked by viruses or worms, and thereby become infected and capable of spreading the infection to others.

Figure 5.21 Connections between Local Machines and between Computers on a Private IP Using SSH Tunnel



By using SSH tunnels, as you can see in Figure 5.21, you can access the computers that have public IP addresses. You can then forward the traffic to a computer with a private IP address such as an office or research lab computer. This method in turn provides the security of a private IP address, while retaining the convenience of a public IP address. An SSH tunnel provides the same functionality as a VPN, but with a simpler configuration.

Technical Description

An SSH tunnel is a connection that takes traffic from an arbitrary port on one machine and sends it through an intermediate machine to a remote machine. Since it uses SSH to create the tunnel, between your computer and the computer on a public IP address, all your data is encrypted.

There are three basic steps to creating a tunnel to a privately addressed machine, and it requires three computers: your local computer, an intermediate computer with a public IP address, and the privately addressed destination computer to which you want to establish a connection.

Before you start an SSH connection from your local computer to the intermediate computer with a public IP address, you have to install SSH clients on your local computer. Some of the common SSH clients are Putty for Windows, MacSSH, MacSFTP, and Nifty Telnet for Mac OS. There are two main, incompatible versions

of the SSH protocol: SSH1 (1.5) and SSH2. SSH1 uses CRC32 (cyclic redundancy check) to check the integrity of a message. CRC32s are prone to collision and are normally used to detect accidental errors in transmissions (IP, TCP, and UDP, for example, use a checksum in their headers). SSH2 (which is the latest version of SSH), on the other hand, uses MACs to check the integrity of messages. Integrity of messages in SSH2, is strengthened by using a cryptographic hash such as MD5 or SHA1. Since SSH1 and SSH use two different schemes to ensure the integrity of the message, make sure you use the recent version of SSH2, or the SSH1 between the client on a local host and on the server are the same.

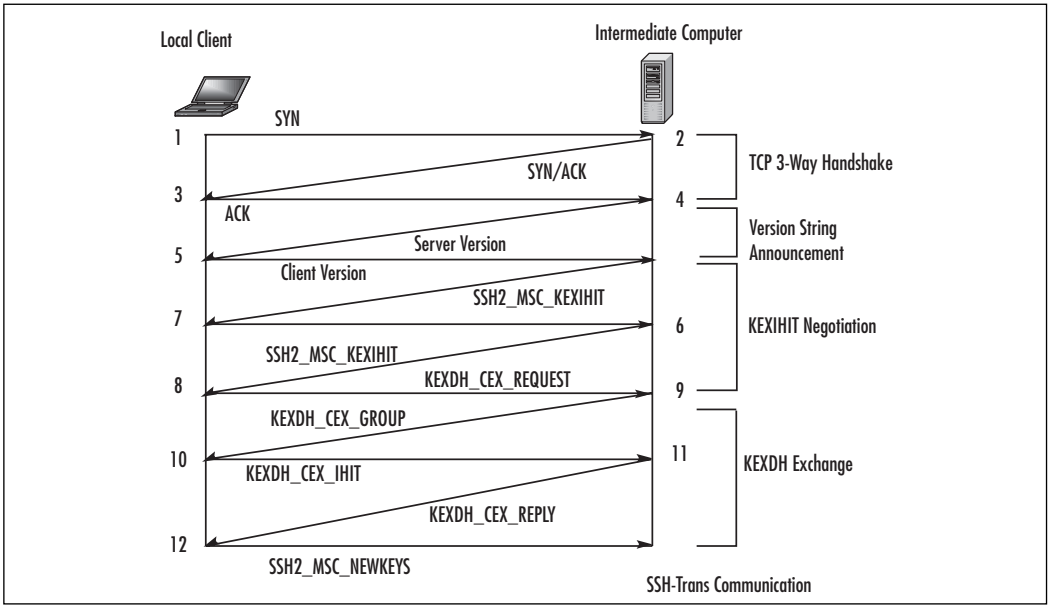
Care has to be taken when you establish the connection to a computer having a public IP address for the first time—make sure you are connecting to the right computer. The SSH2 client will prompt with a warning that it has never seen that computer before. It will then store the public key of the computer having a public IP address in a cache so that on follow-up connections it can compare the received public key with the cached version and verify it hasn't changed.

Figure 5.22 shows the packets exchanged while establishing an SSH connection. Notice that packets 1, 2, and 3 are being used to establish the TCP 3-way handshake. As previously discussed, an SSH connection is successfully established only when both the client and the server have the same version number; if not, either peer can force termination of connection. Packets 4 and 5 as per Figure 5.22 while establishing SSH connection are being used for version string announcement. The server sends its version number first, and then the client sends it. A special code "1.99:" demonstrates that the server supports both SSH1 and SSH2. After the version number is verified, the next phase involves key exchange, bulk data encryption, message integrity, and compression. The primary objective of the SSH2_MSG_KEXINIT exchange (packets 6, 7) the primary objective is to negotiate the algorithms for key exchange, bulk data encryption, message integrity, and compression. The peers will also let each other know the accepted host key types. As mentioned earlier, if diffie-hellman-group-exchange-sha1 is selected as the key exchange method, in SSH2_MSG_KEXDH_GEX_REQUEST (packet 8) the client notifies the server of its minimum, preferred, and maximum prime size for the group.

SSH2_MSG_KEXDH_GEX_GROUP (packet 9) is the server's response to the request and contains an appropriate size for the group's prime packet, and two multiprocessing integers containing the prime to be used (p) and the corresponding generator (g). After receiving this message, both client and server know the Diffie Hellman group to use. There are only two remaining packets in the key exchange (packets 10 and 11) before enough parameters are negotiated to start encrypting data. The client receives p and g, generates a random number x, such that $1 < x < (p-1)/2$, and then calculates $e = g^x \bmod p$. The value of e is sent in

SSH2_MSG_KEXDH_GEX_INIT (packet 10). After the server receives the client's SSH2_MSG_KEXDH_GEX_INIT message, the server generates its own random number y, calculates f = g^y mod p, and sends "f" to the client in SSH2_MSG_KEXDH_GEX_REPLY (packet 11). The server also calculates k = e^y mod p, which is the value of the shared secret. The client, after receiving the reply SSH2 _MSG_KEXDH_GEX_REPLY, does the same, using formula k = f^x mod p. If everything goes right, the client and server should compute identical values for k. This is very important, because k is one of the elements used to create the exchange hash signature, which is the primary factor in server authentication. SSH2_MSG_NEWKEYS (packet 12) contains the notice that keying materials and algorithms should go into effect from this point on.

Figure 5.22 SSH Packet Exchange Diagram between Client and Intermediate Machine



Once you have successfully established an SSH connection with the intermediate computer, the next step is to configure the connection to listen for traffic, to some port on your local machine. This port on the local machine is called forwarded. In the second step, the forwarded port is bound to the local host. When a process connects to the local host on forwarded port on the client machine, the /usr/bin/ssh client program accepts the connection. The SSH client informs the SSH server, over the encrypted channel, to create a connection to the remote

computer (or the computer having a private IP address as shown in Figure 5.22). The client takes any data to the forwarded port, and sends it to the SSH server on a public IP, inside the encrypted SSH session. The SSH server after receiving the data decrypts it and then sends it in the clear to the computer on a remote IP address. The SSH server also takes any data received from the remote computer having a private IP and sends it inside the SSH session back to the client, who decrypts and sends them in the clear to the process connected to the client's on the forwarded port.

On your local machine, use the application you want to connect to the remote computer, and tell it to use the forwarded port on your local computer. When you connect to the local port, it will look like you have established connection to the destination computer on a private IP.

SSH Tunnel in Linux

This section will help you understand how to create an SSH tunnel in Linux. SSH tunneling requires wrapping a TCP connection inside an SSH session. You will have to first configure your computer to send traffic to the tunnel instead of to the Internet. To establish an SSH tunnel, you will have to pick up a port on your computer that is called as a forwarded port. For this section, we will be using port 2345 as a forwarded port. Before doing so, ensure that no other application is listening on port 2345. This can be done by using netcat. Type the command **nc localhost 2345** at the command prompt. If the result of the command is **connection refused**, no other application is listening on port 2345 and it can be used for port forwarding.

Next, you have to set up a tunnel with SSH, and finally you should connect to the tunnel using the application you want to access the remote machine. SSH provides an option -L port:host:hostport. This option specifies that the given port on the local (client) host is to be forwarded to the given host and host port on the remote side. Host will have a private IP and will be streaming data at the host port. By allocating a socket to listen to the port on the local host, whenever a connection is made, the connection is forwarded over the secure channel to the host port from the local machine.

```
dummy$ ssh -L 2345:mailserver.isp.net:110 intermediateserver.usp.net
$ dummy@intermediateserverpassword: **********
dummy@intermediateserver $ hostname
intermediateserver
```

When the command

```
$ ssh -L 2345:mailserver.isp.net:110 intermediateserver.usp.net
```

is executed, the SSH client logs in to the intermediate server. After entering the password for dummy, authentication is complete.

The SSH client also binds to the port 2345 on loopback interface. When any process tries to access connection to 127.0.0.1 on port 2345 on the client machine, the /usr/bin/ssh client program accepts the connection.

Open a different window on your local host and establish a connection to the local computer using netcat.

```
dummy@desktop$ nc localhost 2345
 +OK POP3 mail server (mailserver.isp.net) ready.
 USER <Type POP3 user name>
 +OK
 PASS  <Type password>
```

Now we know that port 2345 is bounded by our SSH process, and the TCP connection to local port 2345 is tunneled through SSH to the other remote mail server. The local host takes data sent to port 2345, and forwards it to the intermediate server inside the encrypted SSH session. The intermediate server then decrypts the data and sends it in the clear to the destination computer, on port 110 of the Mail server.

The intermediate server also takes data received from the Mail server's port 110, and sends it inside the SSH tunnel back to the client , who decrypts and sends it in the clear to the process connected to the client's bound port, port 2345.

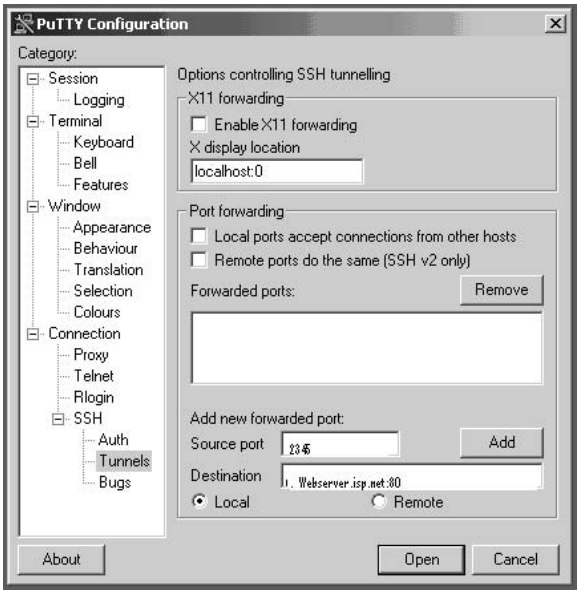
SSH Tunnel in Windows

To establish SSH tunnel in Windows, you will have to ensure an SSH client in installed on your computer. We will be using Putty as our ssh client.(Putty is a free ssh client, and can be downloaded from www.chiark.greenend.org.uk/~sgtatham/). In the previous section, we used an SSH tunnel to secure access to our mail. For Windows, we will be discussing in depth how to establish a secure SSH tunnel from your local computer to access Web pages from a remote Web server on a private IP address.

After **Putty** is successfully installed on your computer, in the Category pane of the application window, click Session, and as shown in Figure 5.23, type **localhost** in the Host Name box. Ensure SSH is selected as your protocol. In the field Source

port , enter the port number where you wish to forward the connection. Here we will be using 2345 as the forwarded port. After adding the port, click **Add**.

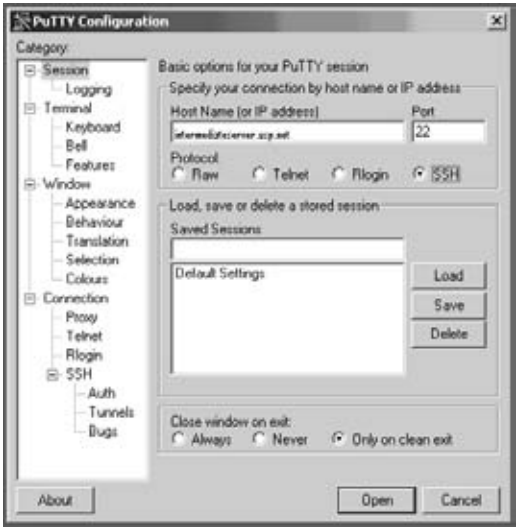
Figure 5.23 Entries to Create SSH Tunnels



After adding the configuration, next in the Category pane, click **Session** and, as shown in Figure 5.24, enter the host name of the computer with a public IP address through which you want to establish your tunnel. Here we will be using intereme-diate.isp.net. Make sure this computer is running sshd. Select SSH as your protocol, which will automatically set the port number to 22. Figure 5.24 shows the image of the configuration. Once you click **Open**, it will prompt for username and password. Enter your username and password and log in to the intermediateserver.isp.net on a public IP address.

To receive Web pages from an SSH tunnel (or from computer on a private IP address (WebServer.isp.net) running a web server), open your Web browser, and in the location bar enter **http://localhost:<PORTNUMBER>/,** where <PORT-NUMBER> is the number of the port on your local machine that you forwarded to the remote machine when you established your connection. In this case, we are using 2345. When you enter, the browser communicates with the remote computer on a private IP address, via an SSH tunnel. It fetches the Web pages served by web-server.isp.net and appears in the browser of your local computer.

Figure 5.24 Establishing an SSH Connection to an Intermediate Computer on a Public IP Address



Notes from the Underground…

Can I View Web Pages Using an SSH Tunnel in Linux?

Yes, you can. To view Web pages from the Web server running on a remote computer, use the command **ssh -L 2345:webserver.isp.net:80 intermediate-server.isp.net**. After successful login to the intermediate server, specify **http://webserver.isp.net: 2345** in your browser.

Pros

SSH operates at the application layer, whereas IPSec operates at the network layer. An SSH tunnel provides advantages in that any application with a fixed port number can be tunneled for a session. Hence, an SSH tunnel is an excellent way to tunnel insecure protocols through a secure communication channel. Various services like POP3 (port 110), FTP (port 21), SMTP (port 25),HTTP (port 80), Telnet (port 23), NNTP (port 119), VNC (port 5900), and NTP (port 123) can be tunneled by using SSH. An SSH tunnel reduces the unnecessary network overhead of encrypting

all the applications as compared with IP VPNS. Tunneling of static TCP ports can also be automated. Moreover, another important advantage of SSH tunneling is that SSH was developed outside the United States, so it does not fall within U.S. government restrictions. Secure Shell, which provides support of encryption in an SSH tunnel, also provides support to many software/algorithms to generate one-time passwords, some of which include SecurID, S/Key, Kerberos, and TIS. SSH provides support for a wide variety of encryption algorithms such as RSA public key, Triple DES, IDEA, and Blowfish. You can also use port forwarding to secure some games; for example, *Age of Empires II* (port 23978) *Baldur's Gate* (port 15000), and *Dark Reign 2* (port 26214).

Cons

Even though an SSH tunnel is lightweight, and can be used to secure insecure protocols like IMAP, SMTP, POP3, HTTP, and FTP, SSH only tunnels applications that use TCP for communication. SSH cannot be used to tunnel applications that do not have known ports, like applications using UDP, port ranges, or dynamic ports. Hence, applications like NFS cannot be protected by using port forwarding. An SSH daemon also does not provide any access control or any restriction against what port or ports can or cannot be forwarded as per the user.

In some computers, SSH is compiled with TCPWRAP options. TCPWRAP controls who can access a particular service on a computer. If a wrapper disallows a user from accessing a service, the SSH tunnel connection will be aborted. In SSH, a user can specify the encryption algorithms he can use for encrypting the channel. Many encrypting protocols can be used in SSH; for example, IDEA, DES, 3DES, and Blowfish. Even though IDEA is considered one of the most secure algorithms available in SSH, it is slowest while transmitting the data. Other algorithms like Blowfish are considered fastest; however, they come at the cost of security. Hence, in SSH tunnel, you will have to compromise on latency of transmitting data.

Others

Besides the aforementioned SSL VPN, and SSH tunnels, there are other solutions to establish a VPN tunnel to a secure communication channel. One of them is CIPE, which stands for Crypto IP encapsulation. CIPE works by tunneling IP packets in UDP packets, and in a similar manner as IPSec. It too provides encryption and tunneling at the IP layer. This is different from SSL or SSH tunnels, which provide tunneling at the TCP layer. Even though CIPE gives much better performance in

respect to IPSEC, IPSEC is more standardized and has more interoperability. CIPE makes use of Blowfish and 128-bit IDEA to secure the communication channel.

Point-to-Point Tunneling (PPTP) is the Microsoft proposed version of VPN, and is comprised of two channels. The first channel is called as the control channel over which link management information is passed, and the second channel is called as data channel over which private data network traffic is passed. The control channel connects to port 1723 on the server, and the data channel uses a generic encapsulation protocol. Although PPTP support is built into Windows, RRAS (Routing and Remote Access Server) needs to be installed and configured, which is a major upgrade on the Windows VPN server.

Microsoft implementation of the PPTP is considered unsecure, and the authentication protocol is prone to dictionary attack. By using a standard sniffer, the following information can be obtained from a Microsoft PPTP server:

- Client machine IP address
- Server machine IP address
- Number of available PPTP virtual tunnels at the server
- Client machine RAS number
- Client machine NetBIOS name
- Client vendor identification
- Server vendor identification
- Internal Virtual Tunnel IP address handed to the client
- Internal DNB servers handed to the client
- Client username
- Enough information to retrieve user's password hash
- Information to retrieve the initialization value used inside MPPE
- Current value of the encrypted packet for the client before RC4 re-initialization
- Current value of the encrypted packet for the server before RC4 re-initialization

To prevent information leakage, you can encrypt the control channel, or remove the channel, and everything the channel does can be done via PPP negotiations or the unused portions of the GRE header. There is no authentication of the control channel. Implementation of the encryption scheme uses output-feedback-mode

stream cipher, whereas a cipher block-chaining-mode block cipher would have been more appropriate. Encryption key is a function of the user password instead of using a key-exchange algorithm like Diffie-Hellman. During the cryptanalysis of the PPTP protocol done by Bruce Schneier, they were able to open connections through a firewall by abusing PPTP negotiations. They were able to crash the Windows NT server by sending the malicious crafted packets from outside the firewall without any authentication. Some of the malicious crafted packets included sending invalid values in the PPTP control packet header, or iterating through all the valid and nonvalid values that could be held in the Packet Type field inside the PptpPacket header. Microsoft's Point-to-Point Encryption protocol provides a way to encrypt PPTP packets. It assumes the existence of a secret key shared by both ends of the connection and uses RC4 with either 40 bits or a 128-bit key. In the cryptanalysis attack against these encryption schemes, they claim to completely negate the usefulness of the encryption protocol. For further details, readers are encouraged to read Bruce Schneier's paper, "Cryptanalysis of Microsoft's Point-to-Point Tunneling Protocol (PPTP)," available at www.schneier.com/paper-pptp.pdf.

The PPP negotiation occurs before and after the encryption can be applied. A PPP CCP packet is being used for the resynchronization of the keys. Since there is no authentication of packets, spoofing the configuration packet containing the DNS server can be exploited to force all name resolution to happen through a malicious name server. Windows 95 can be classified as an obsolete version of Windows; however, the Win 95 client fails to properly sanitize the buffer and the information leaks in the protocol messages. As per the PPTP documentation, characters after the host name and the vendor string should be 0x00 (in the PPTP_START_SESSION_REQUEST) packet. Unfortunately, Windows 95 fails to do it. For Windows NT, all these bytes are set to null.

PPP-SSH VPN is also one of the common methods to establish VPN connection. In the next section, we discuss CIPE and PPP-SSH.

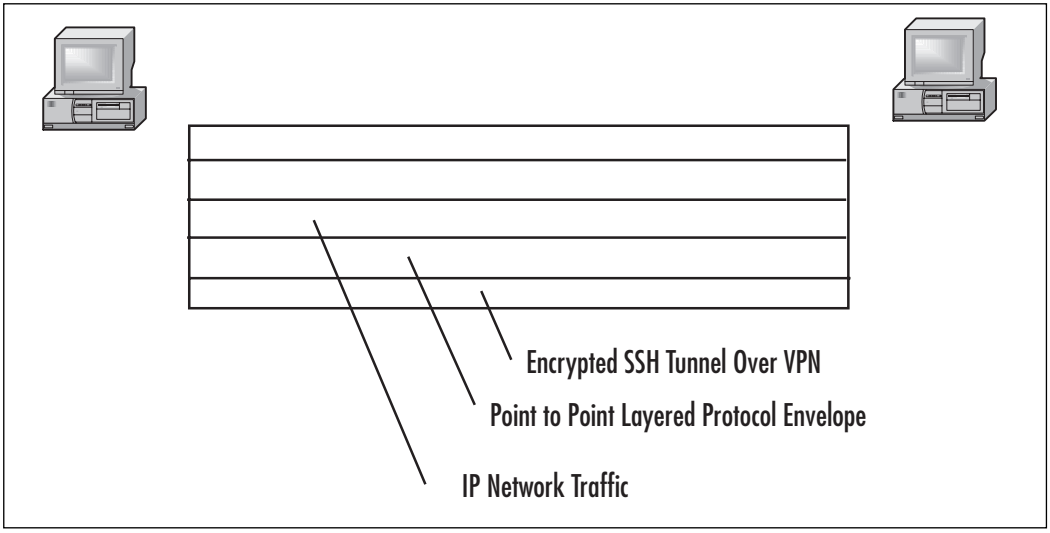
Technical Description

The CIPE protocol involves two parts. The first part is encryption and checksumming of data packets and dynamic key exchange. In CIPE, an IP datagram is padded with the zero to seven random octet so length is congruent to three modulo eight. The packet is then padded with a value of P, which again is one octet. CRC 32 bit checksum is then calculated. The packet is then encrypted with the 64-bit block cipher in CBC mode. IDEA is generally used to perform this. It also makes use of Blowfish with 132-bit keys. The value of P, which is appended to the packet, is calculated as follows: Bits 4, 5, and 6 indicate the length of the packet between the

original packet and P; bits 2 and 1 indicate the type of packet. Value 00 denotes the packet is a data packet, 01 denotes the packet is a key exchange packet, and 10 is reserved. Bits 0, 3, and 7 are reserved and must be zero. CIPE is available for download at http://sourceforge.net/projects/cipe-linux, and CIPE for Windows version is available at http://cipe-win32.sourceforge.net/. Details about the installation procedure for CIPE for a Linux computer can be found at www.redhat.com/docs/manuals/linux/RH-9-Manuals/security-guide/sl-vpn-cipe-install.html.

PPP is generally configured and designed to support single dial-up users Figure 5.25). A PPP connection is established when a user runs the PPP program. By using SSH, you can log on to a remote computer and run a program on the server. SSH ensures that the data stream between the client and the server is encrypted. If the locally and reciprocally running program on a LAN is PPP, and is triggered by SSH, the communication channel becomes encrypted and we can call the data transmission channel to be PPP-SSH VPN. In this scheme, SSH is being used to create a tunnel connection, and pppd is used to run TCP/IP traffic. Further details about the installation procedure for PPP-SSH can be found at www.tldp.org/HOWTO/ppp-ssh/index.html.

Figure 5.25 Packet Encapsulation in a PPP Connection



In PPP-SSH when a network load becomes very high, one TCP connection may get all the bandwidth, resulting in timeouts and dropping of other connections. Overheads and latency on PPP-SSH are also very high. If the ping time on 57.6 modem connections is in 135–175 ms range, the PPP-SSH ping time is around

310—340 ms range. Keep alives are the small packets used to find out if the connections between the computers are alive. In the case of PPP-SSH if the network load is too high, there will be latency in the arrival of packets. This in turn will delay the keep alive packets, and hence you cannot reliably tell if the network connections are down. When your SSH TCP connection is broken, the VPN connection is also disconnected, and all tunneled TCP connections are broken. PPP-SSH is running IP over TCP stream. This may result in weird delays, dropouts, and oscillations, and definitely the concept of IP over TCP is against the concept of the OSI model.

Pros

Besides IPv4, CIPE also provides support for IPv6. CIPE uses Blowfish, which again is not prone to U.S. export restrictions. CIPE is a software-based VPN, and any computer able to run Linux can be used as a CIPE gateway. Organizations can save money by not purchasing dedicated hardware-based VPNs. CIPE has been designed to work with iptables, ipchains, and other rule-based firewalls. CIPE configuration is done through text files, which makes configuration easy. While graphical tools look good, they are often a burden over a network.

PPP-SSH comes with most of the Linux distributions, and many of the Linux kernels are preconfigured, thus saving time when installing, configuring, or recompiling the kernel. By ensuring distinct IP addresses for each tunnel to a single computer, you can establish multiple tunnels to a computer. PPP-SSH can establish VPN connections over dynamic IP addresses. You can have a VPN connection over a dialup connection. If the rules of your firewall are configured to allow SSH, PPP-SSH will work in that configuration as well. Routing in the case of SSH-PPP is a simple task; pppd automatically establishes the routing. If you require complex routing, you will have to update the Perl script with the custom routing command; PPP-SSH VPN thus can be termed as the poor man's VPN.

Cons

Connections making use of CIPE will be slow on dialup modems. It is generally recommended to turn off the compression in the modem. A DoS attack is possible in CIPE. Vulnerability against the key generation process exists in CIPE, and happens when the sender is overrun with bogus packets. Hence, the sender will be using the static keys for a long period of time. To prevent this situation, stop sending the data completely after a long burst of static key sends.

Summary

VPNs have quickly come to supplant traditional WAN technologies such as frame relay, leased lines, and dialup networks. They reduce the total cost of ownership of the WAN by eliminating recurring costs associated with those technologies and using the underlying and nascent IP technology a company has deployed. IPSec is the one of the most commonly used VPNs. Other methodologies to secure communication include SSL VPN, SSH Tunnel, and Layer 2 solutions.

SSL VPN, being the clientless VPN, is the most versatile VPN, whereas SSH Tunnel helps to secure nonsecure protocols. Each of these techniques to secure communication has its advantages and disadvantages; the best scheme to secure a channel depends on a user or an organization.

Solutions Fast Track

Solution IPSec

- ☑ The four topologies for VPNs are mesh (both fully and partially), star topology, hub-and-spoke, and remote access.
- ☑ The difference between a star topology and a hub-and-spoke topology is that in a star topology, the branch or stub networks are not able to communicate with one another. They can only communicate with the central corporate network.
- ☑ IPSec is not capable of traversing NAT devices without some modification. The problem comes when the NAT device changes information in the IP header of the IPSec packet. The changes will result in an incorrect IPSec checksum that is calculated over parts of the IP header. There are workarounds for this problem, however.
- ☑ When the number of VPNs connecting to the router, firewall, or VPN appliance becomes sufficiently large, it might be necessary to install a VPN accelerator module (VAM) into the device to offload many of the cryptographic functions used in the VPN.
- ☑ There are three main choices for encryption schemes in IPSec: DES, 3DES, and AES. AES deployment is not as widely used at present, so it might not be possible to use that encryption algorithm. DES has been proven insecure

against an attack with sufficient resources. 3DES is the only current algorithm that is widely available and provably secure.

- ☑ Message integrity is provided using the MD5, SHA-1, or HMAC hash algorithms.
- ☑ Before an IPSec VPN tunnel can be established, the session parameters must be negotiated using Internet Key Exchange.
- ☑ IPSec security policies define the traffic permitted to enter the VPN tunnel.

Solution SSL VPN

- ☑ SSL VPN is a clientless VPN. A VPN connection can be established from an Internet café.
- ☑ SSL VPN provides very fine-tuned access control over the resources.
- ☑ In SSL VPN, care has be taken to ensure that the downloaded sensitive data or information has been deleted from the clients.

Solution SSH Tunnels

- ☑ SSH tunnels can be used to secure insecure protocols like POP3 and HTTP traffic.
- ☑ SSH tunnels can secure only applications using fixed ports.
- ☑ In SSH tunnels, security of insecure channel comes with the cost of latency. If you are using encryption algorithms that provide more security, latency on the network will be high.

Solution Layer 2 Solution

- ☑ In Layer 2 solutions, the customer takes care of Layer 3 functionalities such as routing.
- ☑ Configuring of provider edge routers in a Layer 2 solution increases as new nodes are added.
- ☑ MPLS packets are forwarded by label lookup.

Others

- ☑ CIPE provides tunneling in UDP packets. Like IPSec, it also works at the IP layer.
- ☑ PPP-SSH comes with many standard Linux distributions, and can be considered a poor man?s VPN. PPP-SSH is a TCP connection over TCP; even though it is secure, connection is unreliable.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: What is the significance of terminating a VPN tunnel on a firewall's internal interface?

A: Terminating a VPN tunnel on a firewall's internal interface allows all VPN traffic to access the internal directory in one hop. This might not be desirable, and if IP filters cannot be applied to VPN tunnel traffic, other methods, such as having the VPN tunnel terminate within an isolated VLAN, must be employed to restrict the traffic.

Q: How does IKE work?

A: The Internet Key Exchange (IKE) protocol is designed to provide mutual authentication of systems and the establishment of a shared secret key to create in IPSec SA. IKE operates in two phases. Phase 1 provides mutual authentication of the systems and the establishment of session keys and is known as the ISAKMP SA. Phase 2 provides for setting up the IPSec SA.

Q: In SSH VPN, is the communication channel between a computer on a public IP and a computer on a private IP encrypted?

A: No. SSH VPN is being used to protect the computers on a private address from being accessed by the Internet. The communication channel between the client and the computer on a public IP is secured using SSH.

Q: Is it possible for many clients on a computer to use IPSec simultaneously?

A: Yes; however, there might be minor problems. IPSec defines a parameter for identifying the traffic by the Security parameter Index. (SPI). Unfortunately, SPI for inbound traffic is different from outbound traffic. Thus, there is no association between inbound and outbound traffic; hence, the connection for many clients on a computer to use IPSec is not reliable.

Q: Does IPSec uses the same encryption algorithm as SSL? Why or Why not?

A: No. IPSec works at THE IP layer which is a loss environment. SSL uses stream ciphers like RC4, which depend on the endpoint synchronization and are suitable for reliable connections like HTTP over TCP. In an environment where there are chances of packets getting lost, a block cipher like 3DES, CAST-128 is used.

Q: Does SSL and IPSec VPNs work at the same layer in the OSI model?

A: No. SSL works at the application layer, and IPSec works at the network layer.

Q: Can I use VPN to secure my wireless network?

A: Yes. Create a separate LAN that will connect all the access points and one more Ethernet interface on the VPN server. You also have to ensure that DHCP service is provided to the wireless LAN. Use dhrelay to provide DHCP service. Ensure that the DHCP server is updated with the new Subnet details and there exists a route to the VPN wireless LAN interface. This can be done by using the **Route add** command. Update the VPN server by adding the DNSname service to ensure clients can access the VPN server by name.