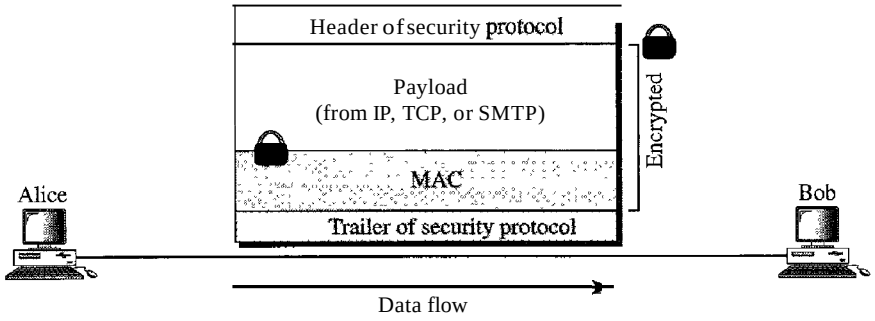
CHAPTER 32

Security in the Internet: IPSec, SSUTLS, PGp, VPN, and Firewalls

In this chapter, we want to show how certain security aspects, particularly privacy and message authentication, can be applied to the network, transport, and application layers of the Internet model. We briefly show how the IPSec protocol can add authentication and confidentiality to the IP protocol, how SSL (or TLS) can do the same for the TCP protocol, and how PGP can do it for the SMTP protocol (e-mail).

In all these protocols, there are some common issues that we need to consider. First, we need to create a MAC. Then we need to enCl)'pt the message and, probably, the MAC. This means, that with some minor variations, the three protocols discussed in this chapter take a packet from the appropriate layer and create a new packet which is authenticated and encrypted. Figure 32.1 shows this general idea.

Figure 32.1 *Common structure of three security protocols*



Note that the header or the trailer of the security protocol mayor may not be included in the encryption process. Note also that some protocols may need more information in the secured packet; the figure shows only the general idea.

One common issue in all these protocols is *security parameters.* Even the simplified structure in Figure 32.1 suggests that Alice and Bob need to know several pieces of information, security parameters, before they can send secured data to each other. In particular, they need to know which algorithms to use for authentication and encryption/decryption.

Even if these algorithms can be predetermined for everyone in the world, which they are not as we will see, Bob and Alice still need at least two keys: one for the MAC and one for encryption/decryption. In other words, the complexity of these protocols lies not in the way the MAC data are calculated or the way encryption is performed; it lies in the fact that before calculating the MAC and performing encryption, we need to create a set of security parameters between Alice and Bob.

At first glance, it looks as if the use of any of these protocols must involve an infinite number of steps. To send secured data, we need a set of security parameters. The secure exchange of security parameters needs a second set of security parameters. The secure exchange of the second set of security parameters needs a third set of security parameters. And so on ad infinitum.

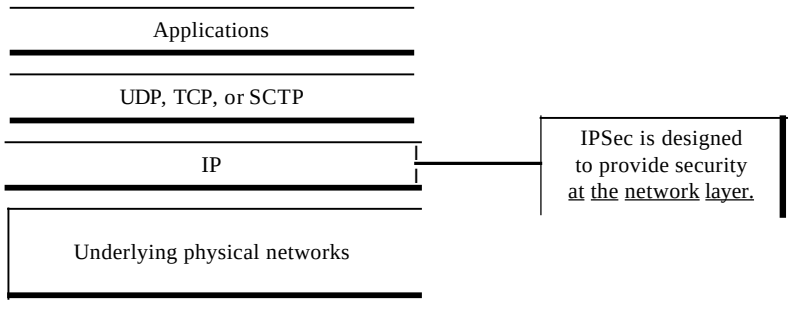
To limit the steps, we can use public-key cryptography if each person has a private and public key pair. The number of steps can be reduced to one or two. In the one-step version, we can use session keys to create the MAC and encrypt both data and MAC. The session keys and the list of algorithms can be sent with the packet but encrypted by using public-key ciphers. In the two-step version, we first establish the security parameters by using public-key ciphers. We then use the security parameters to securely send actual data. One of the three protocols, PGP, uses the first approach; the other two protocols, IPSec and SSLITLS, use the second.

We also discuss a common protocol, the virtual private network (VPN), that uses the IPSec. At the end of the chapter, we discuss the firewall, a mechanism for preventing the attack on the network of the organization.

32.1 IPSecurity (IPSec)

IPSecurity (IPSec) is a collection of protocols designed by the Internet Engineering Task Force (IETF) to provide security for a packet at the network level. IPSec helps to create authenticated and confidential packets for the IP layer as shown in Figure 32.2.

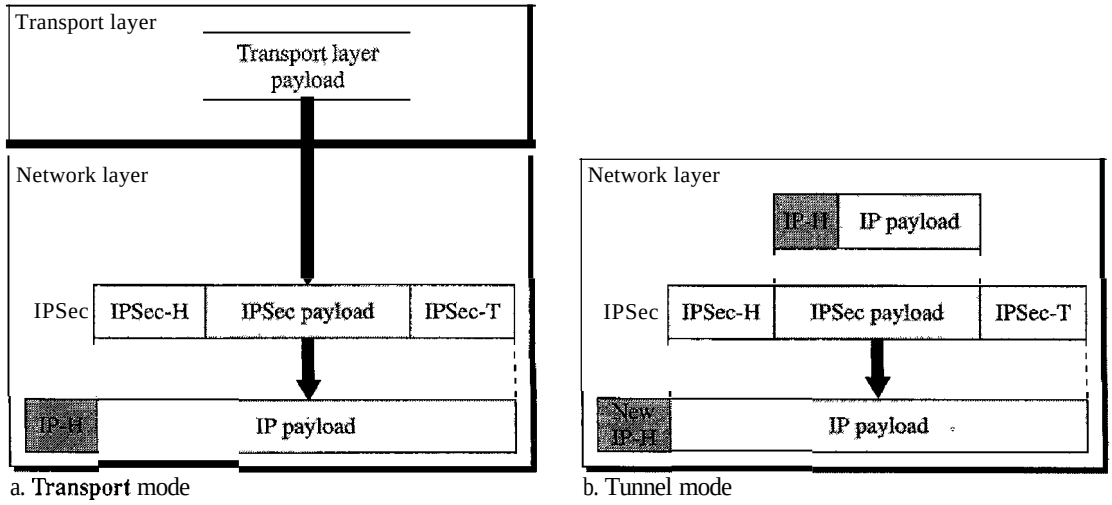
Figure 32.2 *TCPIIP protocol suite and IPSec*



Two Modes

IPSec operates in one of two different modes: the transport mode or the tunnel mode as shown in Figure 32.3.

Figure 32.3 *Transport mode and tunnel modes of IPSec protocol*



Transport Mode

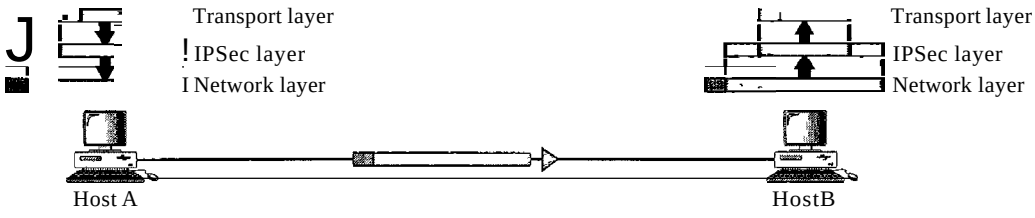
In the transport mode, IPSec protects what is delivered from the transport layer to the network layer. In other words, the transport mode protects the network layer payload, the payload to be encapsulated in the network layer.

Note that the transport mode does not protect the IP header. In other words, the transport mode does not protect the whole IP packet; it protects only the packet from the transport layer (the IP layer payload). In this mode, the IPSec header and trailer are added to the information corning from the transport layer. The IP header is added later.

IPSec in the transport mode does not protect the IP header;
it only protects the information coming from the transport layer.

The transport mode is normally used when we need host-to-host (end-to-end) protection of data. The sending host uses IPSec to authenticate and/or encrypt the payload delivered from the transport layer. The receiving host uses IPSec to check the authentication andlor decrypt the IP packet and deliver it to the transport layer. Figure 32.4 shows this concept.

Figure 32.4 *Transport mode in action*

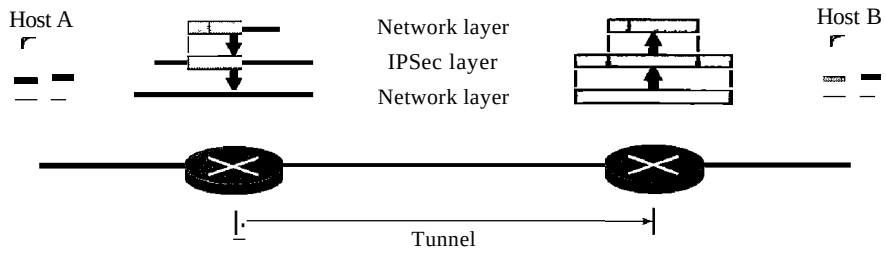


Tunnel Mode

In the tunnel mode, IPSec protects the entire IP packet. It takes an IP packet, including the header, applies IPSec security methods to the entire packet, and then adds a new IP header as shown in Figure 32.5.

The new IP header, as we will see shortly, has different information than the original IF header. The tunnel mode is normally used between two routers, between a host and a router, or between a router and a host as shown in Figure 32.5.

Figure 32.5 *Tunnel mode in action*



In other words, we use the tunnel mode when either the sender or the receiver is not a host. The entire original packet is protected from intrusion between the sender and the receiver. It's as if the whole packet goes through an imaginary tunnel.

IPSec in tunnel mode protects the original IP header.

Two Security Protocols

IPSec defines two protocols-the Authentication Header (AH) Protocol and the Encapsulating Security Payload (ESP) Protocol-to provide authentication and/or encryption for packets at the IP level.

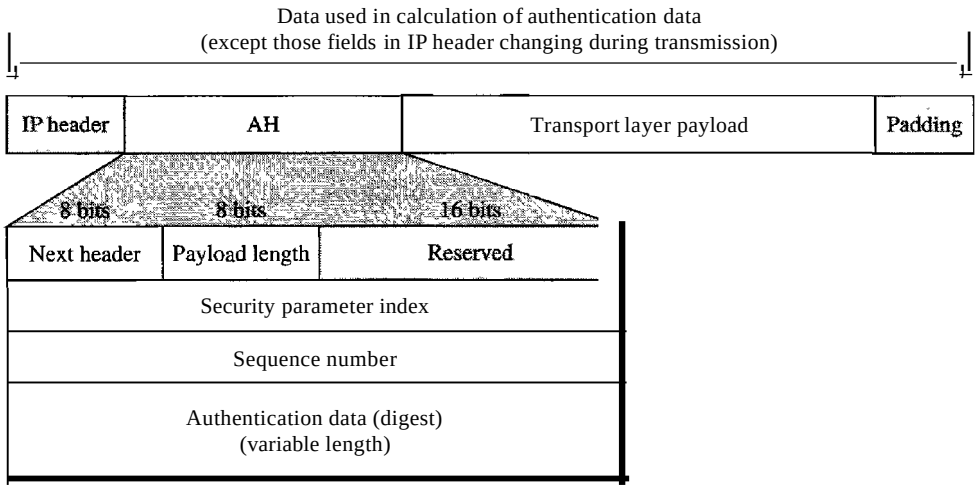
Authentication Header (AH)

The Authentication Header (AH) Protocol is designed to authenticate the source host and to ensure the integrity of the payload carried in the IP packet. The protocol uses a hash function and a symmetric key to create a message digest; the digest is inserted in the authentication header. The AH is then placed in the appropriate location based on the mode (transport or tunnel). Figure 32.6 shows the fields and the position of the authentication header in the transport mode.

When an IP datagram carries an authentication header, the original value in the protocol field of the IP header is replaced by the value 51. A field inside the authentication header (the next header field) holds the original value of the protocol field (the type of payload being carried by the IP datagram). The addition of an authentication header follows these steps:

1. An authentication header is added to the payload with the authentication data field set to zero.

Figure 32.6 *Authentication Header (AH) Protocol in transport mode*



2. Padding may be added to make the total length even for a particular hashing algorithm.
3. Hashing is based on the total packet. However, only those fields of the IP header that do not change during transmission are included in the calculation of the message digest (authentication data).
4. The authentication data are inserted in the authentication header.
5. The IP header is added after the value of the protocol field is changed to 51.

A brief description of each field follows:

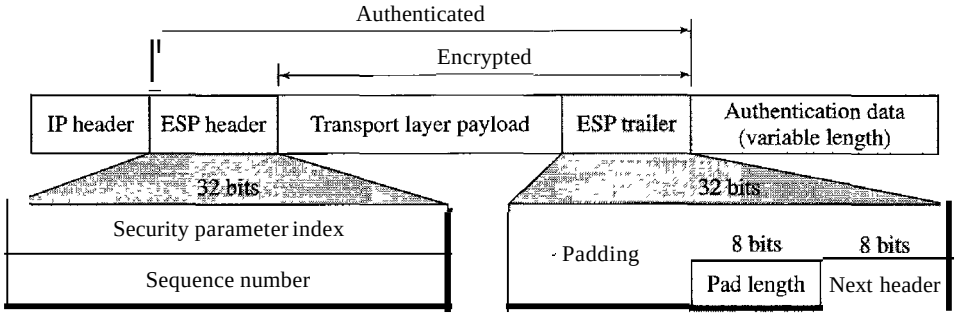
- Next header. The 8-bit next-header field defines the type of payload carried by the IP datagram (such as TCP, UDP, ICMP, or OSPF). It has the same function as the protocol field in the IP header before encapsulation. In other words, the process copies the value of the protocol field in the IP datagram to this field. The value of the protocol field in the new IP datagram is now set to 51 to show that the packet carries an authentication header.
- Payload length. The name of this 8-bit field is misleading. It does not define the length of the payload; it defines the length of the authentication header in 4-byte multiples, but it does not include the first 8 bytes.
- Security parameter index. The 32-bit security parameter index (SPI) field plays the role of a virtual-circuit identifier and is the same for all packets sent during a connection called a security association (discussed later).
- Sequence number. A 32-bit sequence number provides ordering information for a sequence of datagrams. The sequence numbers prevent a playback. Note that the sequence number is not repeated even if a packet is retransmitted. A sequence number does not wrap around after it reaches 2^{32}; a new connection must be established.
- Authentication data. Finally, the authentication data field is the result of applying a hash function to the entire IP datagram except for the fields that are changed during transit (e.g., time-to-live),

The AH Protocol provides source authentication and data integrity, but not privacy.

Encapsulating Security Payload (ESP)

The AH Protocol does not provide privacy, only source authentication and data integrity. IPSec later defined an alternative protocol that provides source authentication, integrity, and privacy called Encapsulating Security Payload (ESP). ESP adds a header and trailer. Note that ESP's authentication data are added at the end of the packet which makes its calculation easier. Figure 32.7 shows the location of the ESP header and trailer.

Figure 32.7 *Encapsulation Security Payload (ESP) Protocol in transport mode*



When an IP datagram carries an ESP header and trailer, the value of the protocol field in the IP header is 50. A field inside the ESP trailer (the next-header field) holds the original value of the protocol field (the type of payload being carried by the IP datagram, such as TCP or UDP). The ESP procedure follows these steps:

1. An ESP trailer is added to the payload.
2. The payload and the trailer are encrypted.
3. The ESP header is added.
4. The ESP header, payload, and ESP trailer are used to create the authentication data.
5. The authentication data are added to the end of the ESP trailer.
6. The IP header is added after the protocol value is changed to 50.

The fields for the header and trailer are as follows:

- Security parameter index. The 32-bit security parameter index field is similar to that defined for the AH Protocol.
- Sequence number. The 32-bit sequence number field is similar to that defined for the AH Protocol.
- Padding. This variable-length field (0 to 255 bytes) of 0s serves as padding.
- Pad length. The 8-bit pad length field defines the number of padding bytes. The value is between 0 and 255; the maximum value is rare.
- Next header. The 8-bitnext-header field is similar to that defined in theAH Protocol. It serves the same purpose as the protocol field in the IP header before encapsulation.

- Authentication data. Finally, the authentication data field is the result of applying an authentication scheme to parts of the datagram. Note the difference between the authentication data in AH and ESP. In AH, part of the IP header is included in the calculation of the authentication data; in ESP, it is not.

ESP provides source authentication, data integrity, and privacy.

IPv4 and IPv6

IPSec supports both IPv4 and IPv6. In IPv6, however, AH and ESP are part of the extension header.

AH Versus ESP

The ESP Protocol was designed after the AH Protocol was already in use. ESP does whatever AH does with additional functionality (privacy). The question is, Why do we need AH? The answer is, We don't. However, the implementation of AH is already included in some commercial products, which means that AH will remain part of the Internet until the products are phased out.

Services Provided by IPSec

The two protocols, AH and ESP, can provide several security services for packets at the network layer. Table 32.1 shows the list of services available for each protocol.

Table 32.1 *IPSec services*

Services	*AH*	*ESP*
Access control	Yes	Yes
Message authentication (message integrity)	Yes	Yes
Entity authentication (data source authentication)	Yes	Yes
Confidentiality	No	Yes
Replay attack protection	Yes	Yes

Access Control IPSec provides access control indirectly by using a Security Association Database (SADB) as we will see in the next section. When a packet arrives at a destination, and there is no security association already established for this packet, the packet is discarded.

Message Authentication The integrity of the message is preserved in both AH and ESP by using authentication data. A digest of data is created and sent by the sender to be checked by the receiver.

Entity Authentication The security association and the keyed-hashed digest of the data sent by the sender authenticate the sender of the data in both AH and ESP.

Confidentiality The encryption of the message in ESP provides confidentiality. AH, however, does not provide confidentiality. If confidentiality is needed, one should use ESP instead of AH.

Replay Attack Protection In both protocols, the replay attack is prevented by using sequence numbers and a sliding receiver window. Each IPSec header contains a unique sequence number when the security association is established. The number starts from **O** and increases until the value reaches 2^{32} - 1 (the size of the sequence number field is 32 bits). When the sequence number reaches the maximum, it is reset to zero and, at the same time, the old security association (see the next section) is deleted and a new one is established. To prevent processing of duplicate packets, IPSec mandates the use of a fixed-size window at the receiver. The size of the window is determined by the receiver with a default value of 64.

Security Association

As we mentioned in the introduction to the chapter, each of three protocols we discuss in this chapter (IPSec, SSLffLS, and PGP) needs a set of security parameters before it can be operative. In IPSec, the establishment of the security parameters is done via a mechanism called security association (SA).

IP, as we have seen, is a connectionless protocol: Each datagram is independent of the others. For this type of communication, the security parameters can be established in one of three ways.

1. Security parameters related to each datagram can be included in each datagram. The designer of IPSec did not choose this option probably because of overhead. Adding security parameters to each datagram creates a large overhead, particularly if the datagram is fragmented several times during its journey.
2. A set of security parameters can be established for each datagram. This means that before each datagram is transmitted, a set of packets needs to be exchanged between the sender and receiver to establish security parameters. This is probably less efficient than the first choice, and it is not used in IPSec.
3. IPSec uses the third choice. A set of security parameters can be established between a sender and a particular receiver the first time the sender has a datagram to send to that particular receiver. The set can be saved for future transmission of IP packets to the same receiver.

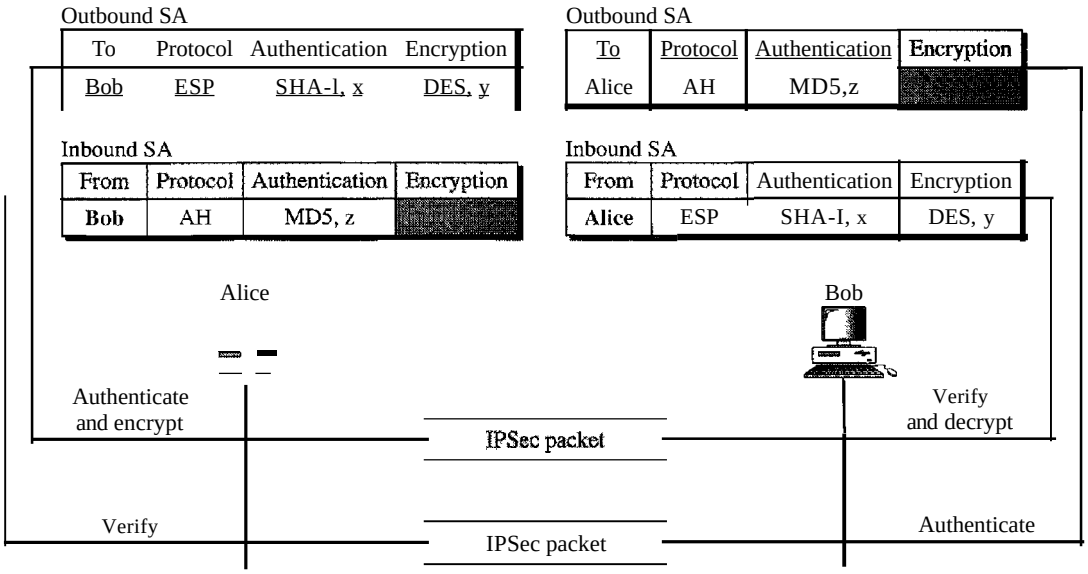
Security association is a very important aspect of IPSec. Using security association, IPSec changes a connectionless protocol, IP, to a connection-oriented protocol. We can think of an association as a connection. We can say that when Alice and Bob agree upon a set of security parameters between them, they have established a logical connection between themselves (which is called association). However, they may not use this connection all the time. After establishing the connection, Alice can send a datagram to Bob today, another datagram a few days later, and so on. The logical connection is there and ready for sending a secure datagram. Of course, they can break the connection, or they can establish a new one after a while (which is a more secure way of communication).

A Simple Example

A security association is a very complex set of pieces of information. However, we can show the simplest case in which Alice wants to have an association with Bob for use in

a two-way communication. Alice can have an outbound association (for datagrams to Bob) and an inbound association (for datagrams from Bob). Bob can have the same. In this case, the security associations are reduced to two small tables for both Alice and Bob as shown in Figure 32.8.

Figure 32.8 *Simple inbound and outbound security associations*



The figure shows that when Alice needs to send a datagram to Bob, she uses the ESP Protocol of IPSec. Authentication is done by using SHA-l with key x. The encryption is done by using DES with key y. When Bob needs to send a datagram to Alice, he uses the AR Protocol of IPSec. Authentication is done by using MD5 with key z. Note that the inbound association for Bob is the same as the outbound association for Alice, and vice versa.

Security Association Database (SADB)

A security association can be very complex. This is particularly true if Alice wants to send messages to many people and Bob needs to receive messages from many people. In addition, each site needs to have both inbound and outbound SAs to allow bidirectional communication. In other words, we need a set of SAs that can be collected into a database. This database is called the security association database (SADB). The database can be thought of as a two-dimensional table with each row defining a single SA. Normally, there are two SADBs, one inbound and one outbound.

Security Parameter Index

To distinguish one association from the other, each association is identified by a parameter called the security parameter index (SPI). This parameter, in conjunction with the destination address (outbound) or source address (inbound) and protocol (AR or ESP), uniquely defines an association.

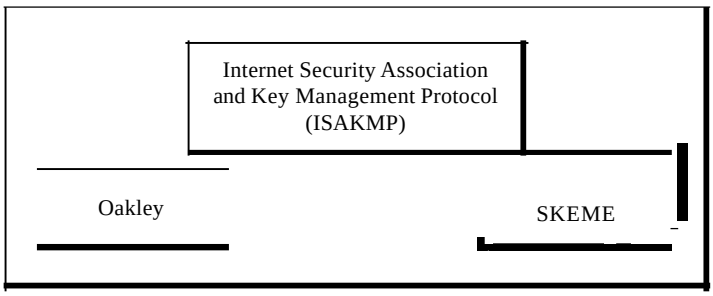
Internet Key Exchange (IKE)

Now we come to the last part of the puzzle-how SADBs are created. The Internet Key Exchange (IKE) is a protocol designed to create both inbound and outbound security associations in SADBs.

IKE creates SAs for IPSec.

IKE is a complex protocol based on three other protocols-Oakley, SKEME, and ISAKMP-as shown in Figure 32.9.

Figure 32.9 *IKE components*



The Oakley Protocol was developed by Hilarie Orman. It is a key creation protocol based on the Diffie-Hellman key-exchange method, but with some improvements. Oakley is a free-formatted protocol in the sense that it does not define the format of the message to be exchanged.

SKEME, designed by Hugo Krawcyzk, is another protocol for key exchange. It uses public-key encryption for entity authentication in a key-exchange protocol.

The Internet Security Association and Key Management Protocol (ISAKMP) is a protocol designed by the National Security Agency (NSA) that actually implements the exchanges defined in IKE. It defines several packets, protocols, and parameters that allow the IKE exchanges to take place in standardized, formatted messages to create SAs.

One may ask how ISAKMP is carried from the sender to the receiver. This protocol is designed so as to be applicable with any underlying protocol. For example, the packet can be used as the payload in the network layer or transport layer. When we use IPSec, it is natural that this packet be considered as a payload for the IP protocol and carried in the datagram. Now the next question is, How are the datagrams that carry ISAKMP securely exchanged? The answer is that there is no need. There is nothing in the ISAKMP packets that needs to be secured.

Virtual Private Network

Virtual private network (VPN) is a technology that is gaining popularity among large organizations that use the global Internet for both intra- and interorganization communication, but require privacy in their internal communications. We discuss VPN here because it uses the IPSec Protocol to apply security to the IP datagrams.

Private Networks

A private network is designed for use inside an organization. It allows access to shared resources and, at the same time, provides privacy. Before we discuss some aspects of these networks, let us define two commonly used, related terms: *intranet* and *extranet.*

Intranet An intranet is a private network (LAN) that uses the Internet model. However, access to the network is limited to the users inside the organization. The network uses application programs defined for the global Internet, such as HTTP, and may have Web servers, print servers, file servers, and so on.

Extranet An extranet is the same as an intranet with one major difference: Some resources may be accessed by specific groups of users outside the organization under the control of the network administrator. For example, an organization may allow authorized customers access to product specifications, availability, and online ordering. A university or a college can allow distance learning students access to the computer lab after passwords have been checked.

Addressing A private network that uses the Internet model must use IP addresses. Three choices are available:

1. The network can apply for a set of addresses from the Internet authorities and use them without being connected to the Internet. This strategy has an advantage. If in the future the organization decides to be connected to the Internet, it can do so with relative ease. However, there is also a disadvantage: The address space is wasted in the meantime.
2. The network can use any set of addresses without registering with the Internet authorities. Because the network is isolated, the addresses do not have to be unique. However, this strategy has a serious drawback: Users might mistakenly confuse the addresses as part of the global Internet.
3. To overcome the problems associated with the first and second strategies, the Internet authorities have reserved three sets of addresses, shown in Table 32.2.

Table 32.2 *Addresses for private networks*

Prefix	*Range*	*Total*
10/8	10.0.0.0 to 10.255.255.255	2^{24}
172.16/12	172.16.0.0 to 172.31.255.255	2^{20}
192.168/16	192.168.0.0 to 192.168.255.255	2^{16}

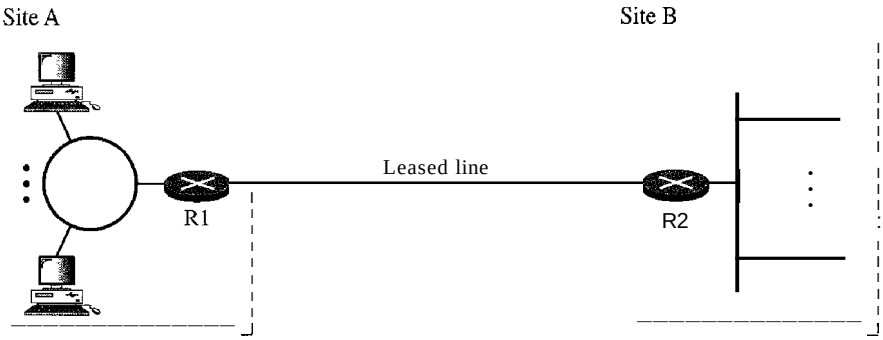
Any organization can use an address out of this set without permission from the Internet authorities. Everybody knows that these reserved addresses are for private networks. They are unique inside the organization, but they are not unique globally. No router will forward a packet that has one of these addresses as the destination address.

Achieving Privacy

To achieve privacy, organizations can use one of three strategies: private networks, hybrid networks, and virtual private networks.

Private Networks An organization that needs privacy when routing information inside the organization can use a private network as discussed previously. A small organization with one single site can use an isolated LAN. People inside the organization can send data to one another that totally remain inside the organization, secure from outsiders. A larger organization with several sites can create a private internet. The LANs at different sites can be connected to each other by using routers and leased lines. In other words, an internet can be made out of private LANs and private WANs. Figure 32.10 shows such a situation for an organization with two sites. The LANs are connected to each other by routers and one leased line.

Figure 32.10 *Private network*



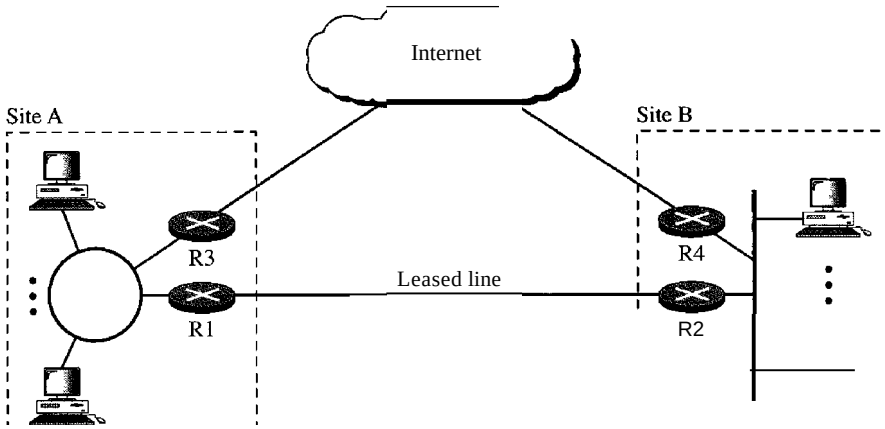
In this situation, the organization has created a private internet that is totally isolated from the global Internet. For end-to-end communication between stations at different sites, the organization can use the Internet model. However, there is no need for the organization to apply for IP addresses with the Internet authorities. It can use private IP addresses. The organization can use any IP class and assign network and host addresses internally. Because the internet is private, duplication of addresses by another organization in the global Internet is not a problem.

Hybrid Networks Today, most organizations need to have privacy in intraorganization data exchange, but, at the same time, they need to be connected to the global Internet for data exchange with other organizations. One solution is the use of a hybrid network. A hybrid network allows an organization to have its own private internet and, at the same time, access to the global Internet. Intraorganization data are routed through the private internet; interorganization data are routed through the global Internet. Figure 32.11 shows an example of this situation.

An organization with two sites uses routers R1 and R2 to connect the two sites privately through a leased line; it uses routers R3 and R4 to connect the two sites to the rest of the world. The organization uses global IP addresses for both types of communication. However, packets destined for internal recipients are routed only through routers R1 and R2. Routers R3 and R4 route the packets destined for outsiders.

Virtual Private Networks Both private and hybrid networks have a major drawback: cost. Private wide-area networks (WANs) are expensive. To connect several sites, an organization needs several leased lines, which means a high monthly fee. One solution

Figure 32.11 *Hybrid network*

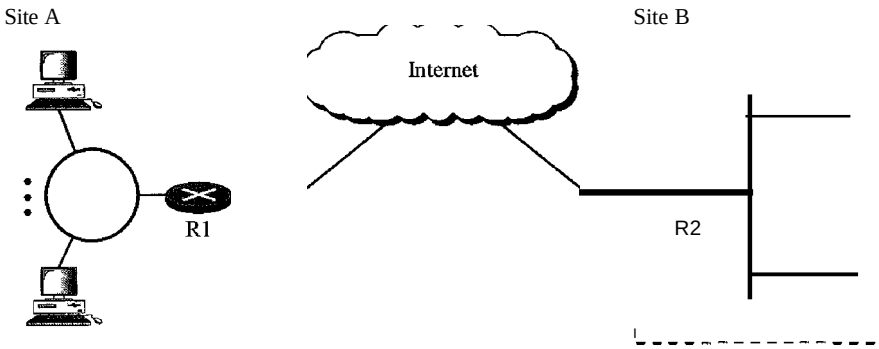


is to use the global Internet for both private and public communications. A technology called virtual private network allows organizations to use the global Internet for both purposes.

VPN creates a network that is private but virtual. It is private because it guarantees privacy inside the organization. It is virtual because it does not use real private WANs; the network is physically public but virtually private.

Figure 32.12 shows the idea of a virtual private network. Routers R1 and R2 use VPN technology to guarantee privacy for the organization.

Figure 32.12 *Virtual private network*

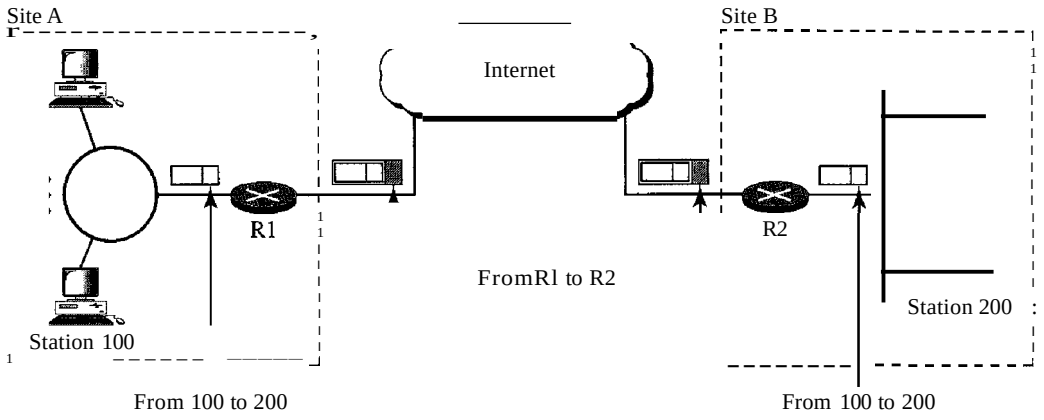


VPN Technology

VPN technology uses IPSec in the tunnel mode to provide authentication, integrity, and privacy.

Tunneling To guarantee privacy and other security measures for an organization, VPN can use the IPSec in the tunnel mode. In this mode, each IP datagram destined for private use in the organization is encapsulated in another datagram. To use IPSec in tunneling, the VPNs need to use two sets of addressing, as shown in Figure 32.13.

Figure 32.13 *Addressing in a VPN*



The public network (Internet) is responsible for carrying the packet from R1 to R2. Outsiders cannot decipher the contents of the packet or the source and destination addresses. Deciphering takes place at R2, which finds the destination address of the packet and delivers it.

32.2 SSL/TLS

A transport layer security provides end-to-end security services for applications that use a reliable transport layer protocol such as TCP. The idea is to provide security services for transactions on the Internet. For example, when a customer shops online, the following security services are desired:

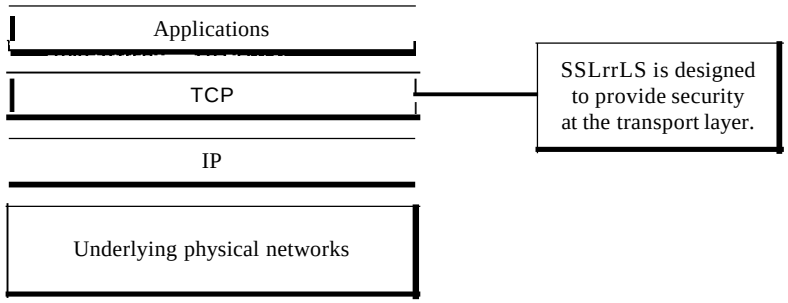
1. The customer needs to be sure that the server belongs to the actual vendor, not an imposter. The customer does not want to give an imposter her credit card number (entity authentication). Likewise, the vendor needs to authenticate the customer.
2. The customer and the vendor need to be sure that the contents of the message are not modified during transition (message integrity).
3. The customer and the vendor need to be sure that an imposter does not intercept sensitive information such as a credit card number (confidentiality).

Two protocols are dominant today for providing security at the transport layer: the Secure Sockets Layer (SSL) Protocol and the Transport Layer Security (TLS) Protocol. The latter is actually an IETF version of the former. First we discuss SSL, then we briefly mention the main differences between SSL and TLS. Figure 32.14 shows the position of SSL and TLS in the Internet model.

SSL Services

Secure Socket Layer (SSL) is designed to provide security and compression services to data generated from the application layer. Typically, SSL can receive data from any application layer protocol, but usually the protocol is HTTP. The data received from the

Figure 32.14 *Location of SSL and TLS in the Internet model*



application are compressed (optional), signed, and encrypted. The data are then passed to a reliable transport layer protocol such as TCP. Netscape developed SSL in 1994. Versions 2 and 3 were released in 1995. In this chapter, we discuss SSLv3. SSL provides several services on data received from the application layer.

Fragmentation

First, SSL divides the data into blocks of 2^{14} bytes or less.

Compression

Each fragment of data is compressed by using one of the lossless compression methods negotiated between the client and server. This service is optional.

Message Integrity

To preserve the integrity of data, SSL uses a keyed-hash function to create a MAC.

Confidentiality

To provide confidentiality, the original data and the MAC are encrypted using symmetric-key cryptography.

Framing

A header is added to the encrypted payload. The payload is then passed to a reliable transport layer protocol.

Security Parameters

When we discussed IPSec in the previous section, we mentioned that each of the two parties involved in data exchange needs to have a set of parameters for each association (SA). SSL has a similar goal, but a different approach. There are no SAs, but there are cipher suites and cryptographic secrets that together make the security parameters.

Cipher Suite

The combination of key exchange, hash, and encryption algorithms defines a **cipher suite** for each SSL session. Each suite starts with the term *SSL*, followed by the key-exchange

algorithm. The word *WITH* separates the key exchange algorithm from the encryption and hash algorithms. For example,

SSL_DHE_RSA_*WITH*_DES_CBC_SHA

defines DHE_RSA (ephemeral Diffie-Hellman with RSA digital signature) as the key exchange with DES_CBC as the encryption algorithm and SHA as the hash algorithm. Note that DH is fixed Diffie-Hellman, DHE is ephemeral Diffie-Hellman, and DH-anon is anonymous Diffie-Hellman. Table 32.3 shows the suites used in the United States. We have not included those that are used for export. Note that not all combinations of key-exchange algorithms (to establish keys for message authentication and encryption), encryption algorithms, and authentication algorithms are included in the cipher suite list. We have not defined or discussed several algorithms you can find in the table, but we wish to describe the whole picture so that the reader can have an idea of how general the suite is.

Table 32.3 *SSL cipher suite list*

Cipher Suite	*Key Exchange Algorithm*	*Encryption Algorithm*	*Hash Algorithm*
SSL_NULL_WITH_NULL_NULL	NULL	NULL	NULL
SSL_RSA_*WITH_NULL_MD5*	RSA	NULL	MD5
SSL_RSA_WITH_NULL_SHA	RSA	NULL	SHA
SSL_RSA_*WITH_RC4_128_MD5*	RSA	RCU28	MD5
SSL_RSA_WITH_RC4_128_SHA	RSA	RC4_128	SHA
SSL_RSA_WITH_IDEA_CBC_SHA	RSA	IDEA_CSC	SHA
SSL_RSA_WITH_DES_CBC_SHA	RSA	DES_eSC	SHA
SSL_RSA_*WITH_3DES_EDE*_CBC_SHA	RSA	3DES_EDE_CBC	SHA
SSL_DH3ll0ll_WITH_RC4_128_MD5	DH_anon	RC4_128	MD5
SSL_DH_anon_*WITH_DES*_CRC_SHA	DH_anon	DES_CBC	SHA
SSL_DH_anon_*WITH_3DES_EDE_CBC_SHA*	DH_anon	3DES_EDE_CBC	SHA
SSL_DHE_RSA_WITH_DES_CBC_SHA	DHE_RSA	DES_CBC	SHA
SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA	DHE_RSA	3DES_EDE_CBC	SHA
SSL_DHE_DSS_WITH_DES_CBCSHA	DHE_DSS	DES_CBC	SHA
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DHE_DSS	3DBS_EDE_CSC	SHA
SSL_DH_RSA_WITH_DES_CRC_SHA	DH_RSA	DES_CBC	SHA
SSL_DH_RSA_WITH_3DES_EDE_CBC_SHA	DH_RSA	30BS_EDE_CSC	SHA
SSL_DH_DSS_*WITH_DES_CBC_SHA*	DH_DSS	DES_CEC	SHA
SSL_DH_DSS_WITH_3DES_EDE_CBC_SHA	DH_DSS	3DES_EDE_CBC	SHA
SSL_FORTEZZA_DMS_WITH_NULL_SHA	FORTEZZA_DMS	NULL	SHA
SSL_FORTEZZA_DMS_WITH_FORTEZZA_CRC_SHA	FORTEZZA_DMS	FORTEZZA_CBC	SHA
SSL]ORTEZZA_DMS_WITH_RC4_128_SHA	FORTEZZA_DMS	RC4_128	SHA

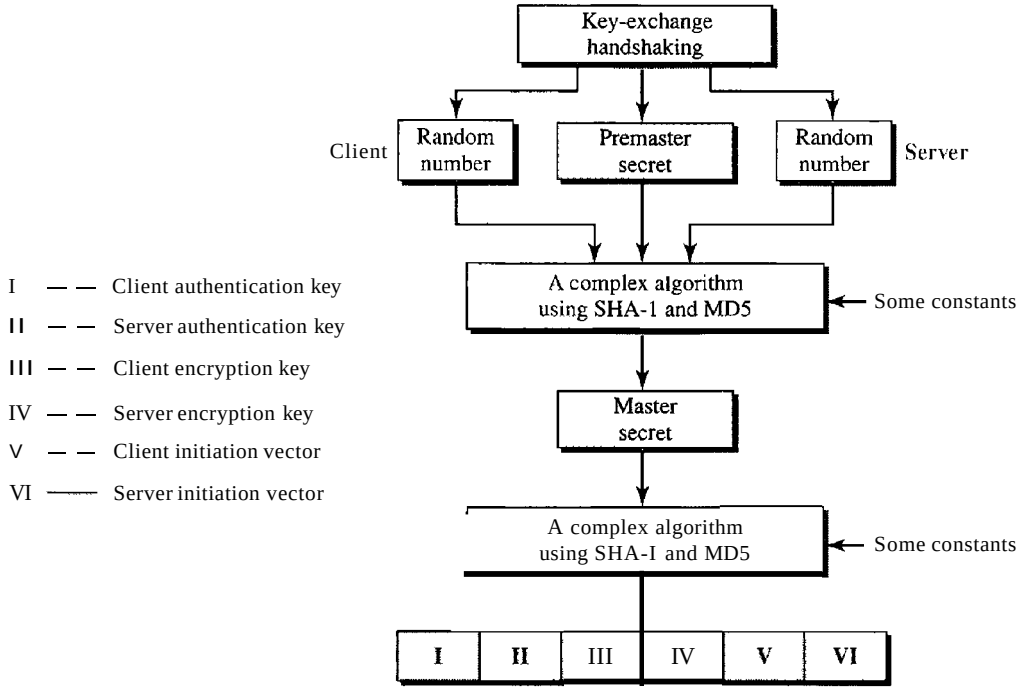
Cryptographic Secrets

The second part of security parameters is often referred to as cryptographic secrets. To achieve message integrity and confidentiality, SSL needs six cryptographic secrets, four keys, and two IVs.

The client and the server have six different cryptography secrets.

The process of creating these secrets is shown in Figure 32.15. The client needs one key for message authentication, one key for encryption, and one IV for block encryption. The server needs the same. SSL requires that the keys for one direction be different from those for the other direction. If there is an attack in one direction, the other direction is not affected. These parameters are generated by using a negotiation protocol, as we will see shortly.

Figure 32.15 *Creation of cryptographic secrets in SSL*



1. The client and server exchange two random numbers; one is created by the client and the other by the server.
2. The client and server exchange one premaster secret by using one of the key-exchange algorithms we discussed previously.
3. A 48-byte master secret is created from the premaster secret by applying two hash functions (SHA-1 and MD5).
4. The master secret is used to create variable-length secrets by applying the same set of hash functions and prepending with different constants.

Sessions and Connections

The nature of IP and TCP protocols is different. IP is a connectionless protocol; TCP is a connection-oriented protocol. An association in IPSec transforms the connectionless IP to a connection-oriented secured protocol. TCP is already connection-oriented.

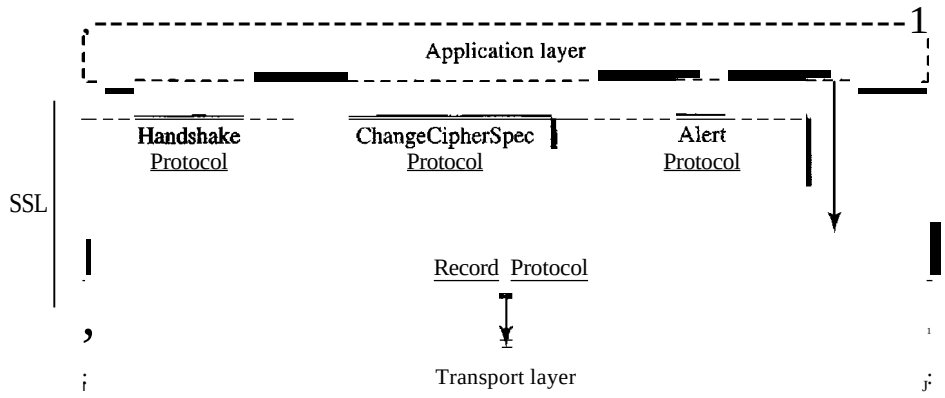
However, the designers of SSL decided that they needed two-levels of connectivity: **session and connection.** A session between two systems is an association that can last for a long time; a connection can be established and broken several times during a session.

Some of the security parameters are created during the session establishment and are in effect until the session is terminated (for example, cipher suite and master key). Some of the security parameters must be recreated (or occasionally resumed) for each connection (for example, six secrets).

Four Protocols

We have discussed the idea of SSL without showing how SSL accomplishes its tasks. SSL defines four protocols in two layers, as shown in Figure 32.16. The **Record Protocol** is the carrier. It carries messages from three other protocols as well as the data coming from the application layer. Messages from the Record Protocol are payloads to the transport layer, normally TCP. The **Handshake Protocol** provides security parameters for the Record Protocol. It establishes a cipher set and provides keys and security parameters. It also authenticates the server to the client and the client to the server, if needed. The **ChangeCipherSpec Protocol** is used for signaling the readiness of cryptographic secrets. The **Alert Protocol** is used to report abnormal conditions. We will briefly discuss these protocols in this section.

Figure 32.16 *Four SSL protocols*



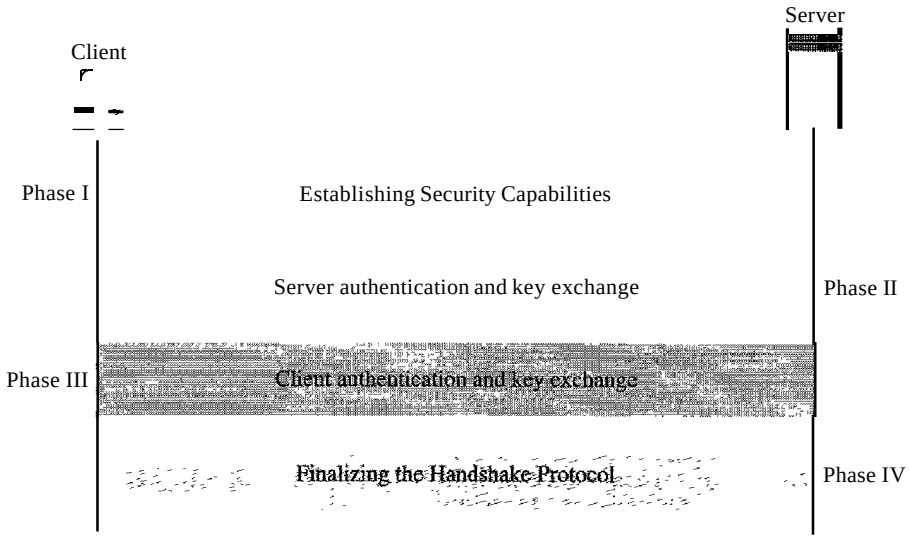
Handshake Protocol

The Handshake Protocol uses messages to negotiate the cipher suite, to authenticate the server to the client and the client to the server (if needed), and to exchange information for building the cryptographic secrets. The handshaking is done in four phases, as shown in Figure 32.17.

ChangeCipherSpec Protocol

We have seen that the negotiation of the cipher suite and the generation of cryptographic secrets are formed gradually during the Handshake Protocol. The question now is, When can the two parties use these parameter secrets? SSL mandates that the parties

Figure 32.17 *Handshake Protocol*



not use these parameters or secrets until they have sent or received a special message, the ChangeCipherSpec message, which is exchanged during the Handshake Protocol and defined in the ChangeCipherSpec Protocol. Before the exchange of any ChangeCipherSpec messages, only the pending columns have values.

Alert Protocol

SSL uses the Alert Protocol for reporting errors and abnormal conditions. It has only one message type, the alert message, that describes the problem and its level (warning or fatal).

Record Protocol

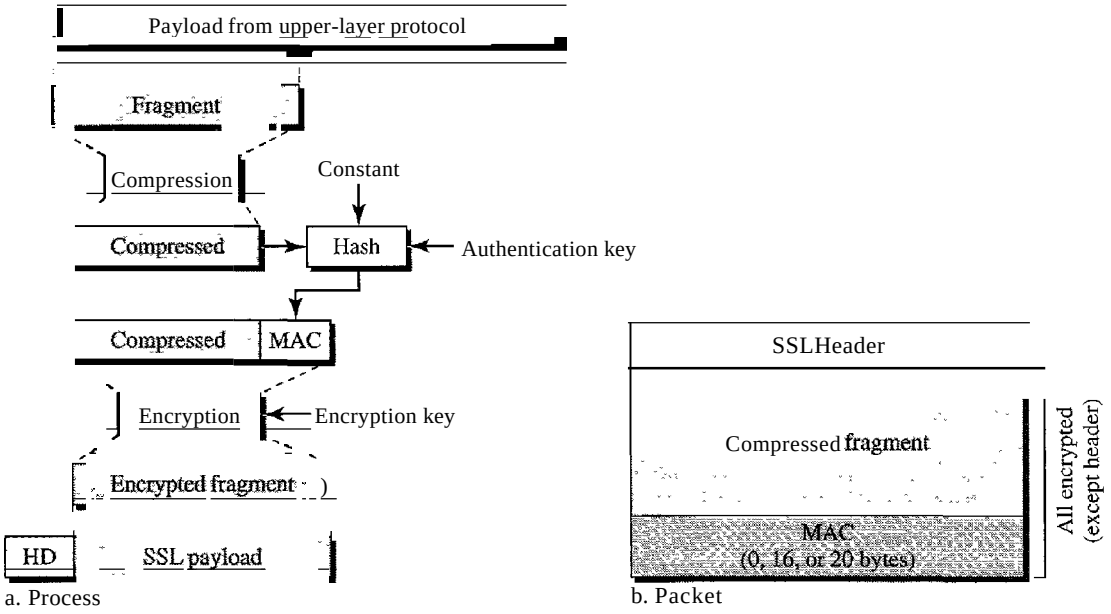
The Record Protocol carries messages from the upper layer (Handshake Protocol, ChangeCipherSpec Protocol, Alert Protocol, or application layer). The message is fragmented and optionally compressed; a MAC is added to the compressed message by using the negotiated hash algorithm. The compressed fragment and the MAC are encrypted by using the negotiated encryption algorithm. Finally, the SSL header is added to the encrypted message. Figure 32.18 shows this process at the sender. The process at the receiver is reversed.

Transport Layer Security

Transport Layer Security (TLS) is the IETF standard version of SSL. The two are very similar, with slight differences. We highlight the differences below:

- Version. The SSLv3.0 discussed in this section is compatible with TLSv1.0.
- Cipher Suite. TLS cipher suite does not support Fortezza.
- Cryptography Secret. There are several differences in the generation of cryptographic secrets. TLS uses a pseudorandom function (PRF) to create the master key and the key materials.

Figure 32.18 *Processing done by the Record Protocol*

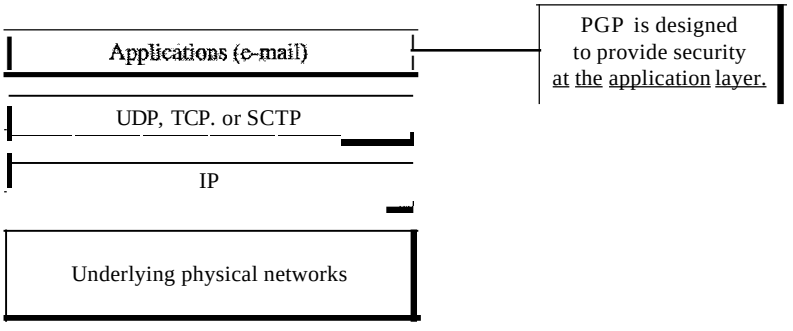


- Alert Protocol. TLS deletes some alert messages and adds some new ones.
- Handshake Protocol. The details of some messages have been changed in TLS.
- Record Protocol. Instead of using MAC, TLS uses the HMAC as defined in Chapter 31.

32.3 PGP

One of the protocols to provide security at the application layer is Pretty Good Privacy (PGP). PGP is designed to create authenticated and confidential e-mails. Figure 32.19 shows the position of PGP in the TCP/IP protocol suite.

Figure 32.19 *Position of PGP in the TCPlIP protocol suite*



Sending an e-mail is a one-time activity. The nature of this activity is different from those we have seen in the previous two sections. In IPSec or SSL, we assume that the two parties create a session between themselves and exchange data in both directions. In e-mail,there is no session. Alice and Bob cannot create a session. Alice sends a message to Bob; sometime later, Bob reads the message and mayor may not send a reply. We discuss the security of a unidirectional message because what Alice sends to Bob is totally independent of what Bob sends to Alice.

Security Parameters

If e-mail is a one-time activity, how can the sender and receiver agree on the security parameters to use for e-mail security? If there is no session and no handshaking to negotiate the algorithms for encryption and authentication, how can the receiver know which algorithm the sender has chosen for each purpose? How can the receiver know the values of the keys used for encryption and authentication?

Phil Zimmerman, the designer and creator of PGP, has found a very elegant solution to the above questions. The security parameters need to be sent with the message.

In PGP, the sender of the message needs to include the identifiers of the algorithms used in the message as well as the values of the keys.

Services

PGP can provide several services based on the requirements of the user. An e-mail can use one or more of these services.

Plaintext

The simplest case is to send the e-mail message in plaintext (no service). Alice, the sender, composes a message and sends it to Bob, the receiver. The message is stored in Bob's mailbox until it is retrieved by him.

Message Authentication

Probably the next improvement is to let Alice sign the message. Alice creates a digest of the message and signs it with her private key. When Bob receives the message, he verifies the message by using Alice's public key. Two keys are needed for this scenario. Alice needs to know her private key; Bob needs to know Alice's public key.

Compression

A further improvement is to compress the message and digest to make the packet more compact. This improvement has no security benefit, but it eases the traffic.

Confidentiality with One-Time Session Key

As we discussed before, confidentiality in an e-mail system can be achieved by using conventional encryption with a one-time session key. Alice can create a session key, use the session key to encrypt the message and the digest, and send the key itself with the message. However, to protect the session key, Alice encrypts it with Bob's public key.

Code Conversion

Another service provided by PGP is code conversion. Most e-mail systems allow the message to consist of only ASCII characters. To translate other characters not in the ASCII set, PGP uses Radix 64 conversion. Each character to be sent (after encryption) is converted to Radix 64 code.

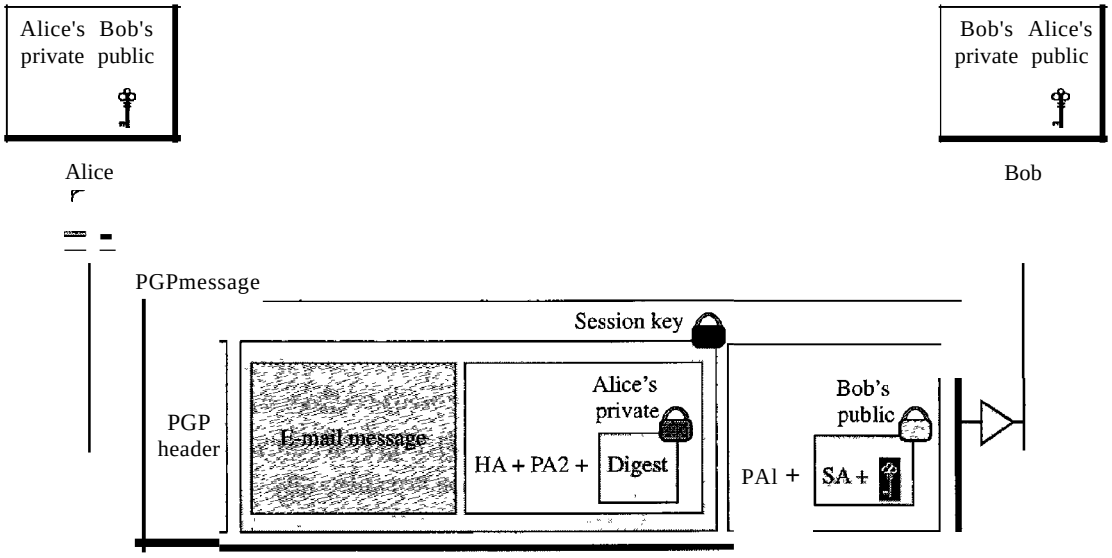
Segmentation

PGP allows segmentation of the message after it has been converted to Radix 64 to make each transmitted unit the uniform size allowed by the underlying e-mail protocol.

A Scenario

Let us describe a scenario that combines some of these services, authentication and confidentiality. The whole idea of PGP is based on the assumption that a group of people who need to exchange e-mail messages trust one another. Everyone in the group somehow knows (with a degree of trust) the public key of any other person in the group. Based on this single assumption, Figure 32.20 shows a simple scenario in which an authenticated and encrypted message is sent from Alice to Bob.

Figure 32.20 *A scenario in which an e-mail message is authenticated and encrypted*



PA1: Public-key algorithm 1 (for encrypting session key)
PA2: PubliC-key algorithm (for encrypting the digest)
SA: Symmetric-key algorithm identification (for encrypting message and digest)
HA: Hash algorithm identification (for creating digest)

Sender Site

The following shows the steps used in this scenario at Alice's site:

1. Alice creates a session key (for symmetric encryption/decryption) and concatenates it with the identity of the algorithm which will use this key. The result is encrypted

with Bob's public key. Alice adds the identification of the public-key algorithm used above to the encrypted result. This part of the message contains three pieces of information: the session key, the symmetric encryption/decryption algorithm to be used later, and the asymmetric encryption/decryption algorithm that was used for this part.

2.
   a. Alice authenticates the message (e-mail) by using a public-key signature algorithm and encrypts it with her private key. The result is called the signature. Alice appends the identification of the public key (used for encryption) as well as the identification of the hash algorithm (used for authentication) to the signature. This part of the message contains the signature and two extra pieces of information: the encryption algorithm and the hash algorithm.
   b. Alice concatenates the three pieces of information created above with the message (e-mail) and encrypts the whole thing, using the session key created in step 1.
3. Alice combines the results of steps 1 and 2 and sends them to Bob (after adding tue appropriate PGP header).

Receiver Site

The following shows the steps used in this scenario at Bob's side after he has received the PGP packet:

1. Bob uses his private key to decrypt the combination of the session key and symmetric-key algorithm identification.
2. Bob uses the session key and the algorithm obtained in step 1 to decrypt the rest of the PGP message. Bob now has the content of the message, the identification of the public algorithm used for creating and encrypting the signature, and the identification of the hash algorithm used to create the hash out of the message.
3. Bob uses Alice's public key and the algorithm defined by PA2 to decrypt the digest.
4. Bob uses the hash algorithm defined by HA to create a hash out of message he obtained in step 2.
5. Bob compares the hash created in step 4 and the hash he decrypted in step 3. **If** the two are identical, he accepts the message; otherwise, he discards the message.

PGP Algorithms

Table 32.4 shows some of the algorithms used in POP. The list is not complete; new algorithms are continuously added.

Table 32.4

Algorithm	*ID*	*Description*
Public key	1	RSA (encryption or signing)
	2	RSA (for encryption only)
	3	RSA (for signing only)
	17	*DSS* (for signing)

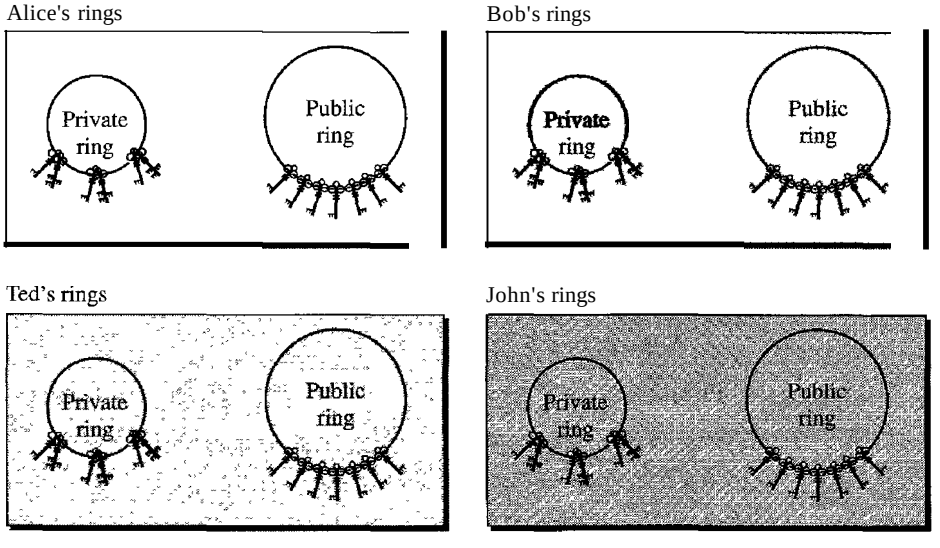
Table 32.4 *(continued)*

Algorithm	*ID*	*Description*
Hash algorithm	1	MD5
	2	SHA-l
	3	RIPE-MD
Encryption	0	No encryption
	1	IDEA
	2	Triple DES
	9	AES

Key Rings

In the previous scenarios, we assumed that Alice needed to send a message to only Bob. That is not always the case. Alice may need to send messages to many people. In this case, Alice needs a key ring of public keys, with a key belonging to each person with whom Alice needs to correspond (send or receive messages). In addition, the PGP designers specified a ring of private/public keys. One reason is that Alice may wish to change her pair of keys from time to time. Another reason is that Alice may need to correspond with different groups of people (friends, colleagues, and so on). Alice may wish to use a different key pair for each group. Therefore, each user needs to have two sets of rings: a ring of private/public keys and a ring of public keys of other people. Figure 32.21 shows a community of four people, each having a ring of pairs of private/public keys and, at the same time, a ring of four public keys belonging to the other four people in the community. The figure shows seven public keys for each public ring. Each person in the ring can keep more than one public key for each other person.

Figure 32.21 *Rings*



Alice, for example, has several pairs of private/public keys belonging to her and public keys belonging to other people. Note that everyone can have more than one public key. Two cases may arise.

1. Alice needs to send a message to one of the persons in the community.
   a. She uses her private key to sign the digest.
   b. She uses the receiver's public key to encrypt a newly created session key.
   c. She encrypts the message and signs the digest with the session key created.
2. Alice receives a message from one of the persons in the community.
   a. She uses her private key to decrypt the session key.
   b. She uses the session key to decrypt the message and digest.
   c. She uses her public key to verify the digest.

PGP Certificates

To trust the owner of the public key, each user in the PGP group needs to have, implicitly or explicitly, a copy of the certificate of the public-key owner. Although the certificate can come from a certificate authority (CA), this restriction is not required in PGP. PGP has its own certificate system.

Protocols that use X509 certificates depend on the hierarchical structure of the trust. There is a predefined chain of trust from the root to any certificate. Every user fully trusts the authority of the CA at the root level (prerequisite). The root issues certificates for the CAs at the second level, a second-level CA issues a certificate for the third level, and so on. Every party that needs to be trusted presents a certificate from some CA in the tree. **If** Alice does not trust the certificate issuer for Bob, she can appeal to a higher-level authority up to the root (which must be trusted for the system to work). In other words, there is one single path from a fully trusted CA to a certificate.

In PGP, there is no need for CAs; anyone in the ring can sign a certificate for anyone else in the ring. Bob can sign a certificate for Ted, John, Anne, and so on. There is no hierarchy of trust in PGP; there is no tree. As a result of the lack of hierarchical structure, Ted may have one certificate from Bob and another certificate from Liz. **If** Alice wants to follow the line of certificates for Ted, it has two paths: one starts from Bob and the other starts from Liz. An interesting point is that Alice may fully trust Bob, but only partially trust Liz. There can be multiple paths in the line of trust from a fully or partially trusted authority to a certificate. In PGP, the issuer of a certificate is usually called an introducer.

In PGP, there can be multiple paths from runy or partially trusted authorities to any subject.

Trusts and Legitimacy

The entire operation of PGP is based on introducer trust, the certificate trust, and the legitimacy of the public keys.

Introducer Trust Levels With the lack of a central authority, it is obvious that the ring cannot be very large if every user in the PGP ring of users has to fully trust everyone else.

(Even in real life we cannot fully trust everyone that we know.) To solve this problem, PGP allows different levels of trust. The number of levels is mostly implementation-dependent, but for simplicity, let us assign three levels of trust to any introducer: *none*, *partial*, and *full*. The introducer trust level specifies the trust levels issued by the introducer for other people in the ring. For example, Alice may fully trust Bob, partially trust Anne, and not trust John at all. There is no mechanism in PGP to determine how to make a decision about the trustworthiness of the introducer; it is up to the user to make this decision.

Certificate Trust Levels When Alice receives a certificate from an introducer, she stores the certificate under the name of the subject (certified entity). She assigns a level of trust to this certificate. The certificate trust level is normally the same as the introducer trust level that issued the certificate. Assume Alice fully trusts Bob, partially trusts Anne and Janette, and has no trust in John. The following scenarios can happen.

1. Bob issues two certificates, one for Linda (with public key K1) and one for Lesley (with public key K2). Alice stores the public key and certificate for Linda under Linda's name and assigns a *full* level of trust to this certificate. Alice also stores the certificate and public key for Lesley under Lesley's name and assigns a full level of trust to this certificate.
2. Anne issues a certificate for John (with public key K3). Alice stores this certificate and public key under John's name, but assigns a *partial* level for this certificate.
3. Janette issues two certificates, one for John (with public key K3) and one for Lee (with public key K4). Alice stores John's certificate under his name and Lee's certificate under his name, each with a *partial* level of trust. Note that John now has two certificates, one from Anne and one from Janette, each with a *partial* level of trust.
4. John issues a certificate for Liz. Alice can discard or keep this certificate with a signature trust of *none*.

Key Legitimacy The purpose of using introducer and certificate trusts is to determine the legitimacy of a public key. Alice needs to know how legitimate are the public keys of Bob, John, Liz, Anne, and so on. PGP defines a very clear procedure for determining key legitimacy. The level of the key legitimacy for a user is the weighted trust level of that user. For example, suppose we assign the following weights to certificate trust levels:

1. A weight of 0 to a nontrusted certificate
2. A weight of $\frac{1}{2}$ to a certificate with partial trust
3. A weight of 1 to a certificate with full trust

Then to fully trust an entity, Alice needs one fully trusted certificate or two partially trusted certificates for that entity. For example, Alice can use John's public key in the previous scenario because both Anne and Janette have issued a certificate for John, each with a certificate trust level of $\frac{1}{2}$. Note that the legitimacy of a public key belonging to an entity does not have anything to do with the trust level of that person. Although Bob can use John's public key to send a message to him, Alice cannot accept any certificate issued by John because, for Alice, John has a trust level of *none*.

Starting the Ring

You might have realized a problem with the above discussion. What if nobody sends a certificate for a fully or partially trusted entity? For example, how can the legitimacy of Bob's public key be determined if no one has sent a certificate for Bob? In PGP, the key legitimacy of a trusted or partially trusted entity can be also determined by other methods.

1. Alice can physically obtain Bob's public key. For example, Alice and Bob can meet personally and exchange a public key written on a piece of paper or to a disk.
2. If Bob's voice is recognizable to Alice, Alice can call him and obtain his public key on the phone.
3. A better solution proposed by PGP is for Bob to send his public key to Alice by e-mail. Both Alice and Bob make a 16-byte MD5 (or 20-byte SHA-l) digest from the key. The digest is normally displayed as eight groups of four digits (or 10 groups of four digits) in hexadecimal and is called a **fingerprint.** Alice can then call Bob and verify the fingerprint on the phone. If the key is altered or changed during the e-mail transmission, the two fingerprints do not match. To make it even more convenient, PGP has created a list of words, each representing a four-digit combination. When Alice calls Bob, Bob can pronounce the eight words (or 10 words) for Alice. The words are carefully chosen by PGP to avoid those similar in pronunciation; for example, if *sword* is in the list, *word* is not.
4. In PGP, nothing prevents Alice from getting Bob's public key from a CA in a separate procedure. She can then insert the public key in the public-key ring.

Web of Trust

PGP can eventually make a **web of trust** between a group of people. If each entity introduces more entities to other entities, the public-key ring for each entity gets larger and larger and entities in the ring can send secure e-mail to one another.

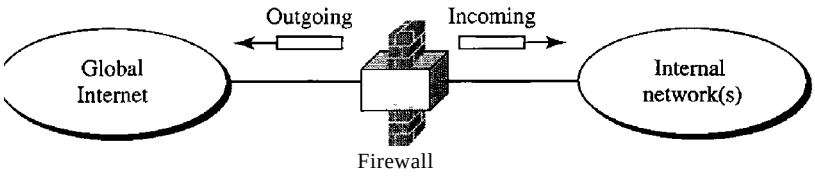
Key Revocation

It may become necessary for an entity to revoke his or her public key from the ring. This may happen if the owner of the key feels that the key is compromised (stolen, for example) or just too old to be safe. To revoke a key, the owner can send a revocation certificate signed by herself. The revocation certificate must be signed by the old key and disseminated to all the people in the ring who use that public key.

32.4 FIREWALLS

All previous security measures cannot prevent Eve from sending a harmful message to a system. To control access to a system, we need firewalls. A **firewall** is a device (usually a router or a computer) installed between the internal network of an organization and the rest of the Internet. It is designed to forward some packets and filter (not forward) others. Figure 32.22 shows a firewall.

Figure 32.22 *Firewall*



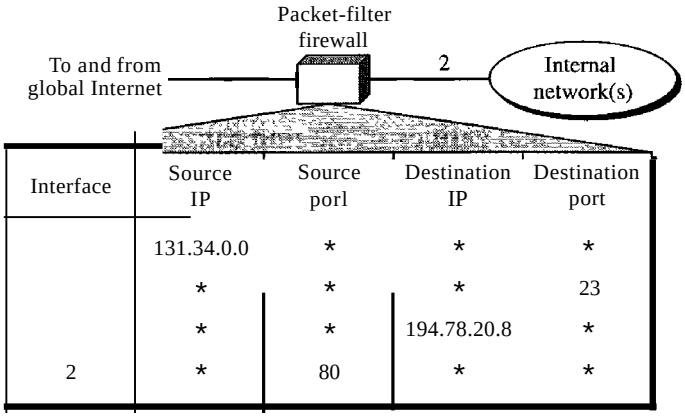
For example, a firewall may filter all incoming packets destined for a specific host or a specific server such as HTTP. A firewall can be used to deny access to a specific host or a specific service in the organization.

A firewall is usually classified as a packet-filter firewall or a proxy-based firewall.

Packet-Filter Firewall

A firewall can be used as a packet filter. It can forward or block packets based on the information in the network layer and transport layer headers: source and destination IP addresses, source and destination port addresses, and type of protocol (TCP or UDP). A packet-filter firewall is a router that uses a filtering table to decide which packets must be discarded (not forwarded). Figure 32.23 shows an example of a filtering table for this kind of a firewall.

Figure 32.23 *Packet-filter firewall*



Interface	Source IP	Source porl	Destination IP	Destination port
	131.34.0.0	*	*	*
	*	*	*	23
	*	*	194.78.20.8	*
2	*	80	*	*

According to Figure 32.23, the following packets are filtered:

). Incoming packets from network 131.34.0.0 are blocked (security precaution). Note that the * (asterisk) means "any."

2. Incoming packets destined for any internal TELNET server (port 23) are blocked.

3. Incoming packets destined for internal host 194.78.20.8 are blocked. The organization wants this host for internal use only.

4. Outgoing packets destined for an HTfP server (port 80) are blocked. The organization does not want employees to browse the Internet.

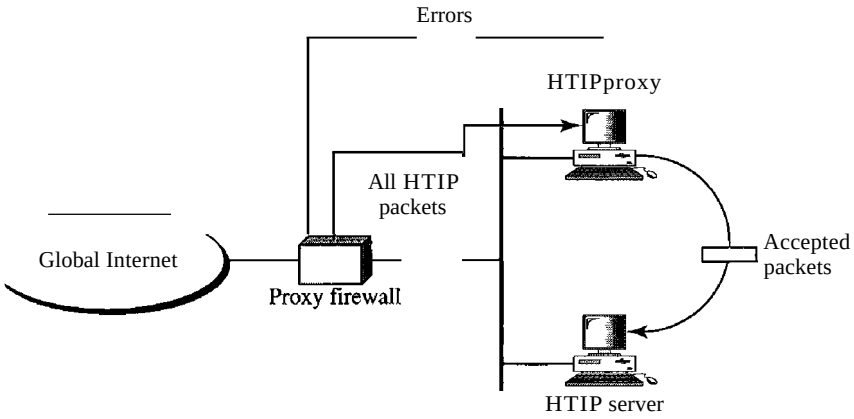
A **packet-filter** firewall filters at the network or transport layer.

Proxy Firewall

The packet-filter firewall is based on the information available in the network layer and transport layer headers (IP and TCPIUDP). However, sometimes we need to filter a message based on the information available in the message itself (at the application layer). As an example, assume that an organization wants to implement the following policies regarding its Web pages: Only those Internet users who have previously established business relations with the company can have access; access to other users must be blocked. In this case, a packet-filter firewall is not feasible because it cannot distinguish between different packets arriving at TCP port 80 (HTTP). Testing must be done at the application level (using URLs).

One solution is to install a proxy computer (sometimes called an application gateway), which stands between the customer (user client) computer and the corporation computer shown in Figure 32.24.

Figure 32.24 *Proxy firewall*



When the user client process sends a message, the proxy firewall runs a server process to receive the request. The server opens the packet at the application level and finds out if the request is legitimate. If it is, the server acts as a client process and sends the message to the real server in the corporation. If it is not, the message is dropped and an error message is sent to the external user. In this way, the requests of the external users are filtered based on the contents at the application layer. Figure 32.24 shows a proxy firewall implementation.

A proxy firewall filters at the application layer.

PDF-XChange Viewer Click to buy NOW! www.docu-track.com

32.5 RECOMMENDED READING

For more details about subjects discussed in this chapter, we recommend the following books. The brackets, [...], refer to the reference list at the end of the text.

Books

IPSec is discussed in Chapter 7 of [Rhe03], Section 18.1 of [PHS03], and Chapters 17 and 18 of [KPS02]. A full discussion ofIPSec can be found in [DH03]. SSLITLS is discussed in Chapter 8 of [Rhe03], and Chapter 19 of [KPS02]. A full discusion of SSL and TLS can be found in [ResOl] and [ThoOO]. PGP is discussed in Chapter 9 of [Rhe03], and Chapter 22 of [KPS02]. Firewalls are discussed in Chapter 10 of [Rhe03] and Chapter 23 of [KPS02]. Firewalls are fully discussed in [CBR03]. Virtual private networks are fully discussed in [YSO1] and [SWE99].

32.6 KEY TERMS

Alert Protocol
Authentication Header (AH) Protocol
certificate trust
ChangeCipherSpec Protocol
cipher suite
connection
Encapsulating Security Payload (ESP)
extranet
fingerprint
firewall
Handshake Protocol
hybrid network
Internet Key Exchange (IKE)
Internet Security Association and Key Management Protocol (lSAKMP)
intranet
introducer
introducer trust
IP Security (IPSec)
key legitimacy
key material
key ring
master secret
Oakley
packet-filter firewall
premaster secret
Pretty Good Privacy (PGP)
private network
proxy firewall
pseudorandom function (PRF)
Record Protocol
replay attack
Secure Socket Layer (SSL)
security association (SA)
security association database (SADB)
security parameter index (SPI)
session
SKEME
Transport Layer Security (TLS)
transport mode
tunnel mode
tunneling
virtual private network (VPN)
web of trust