UNIT- V

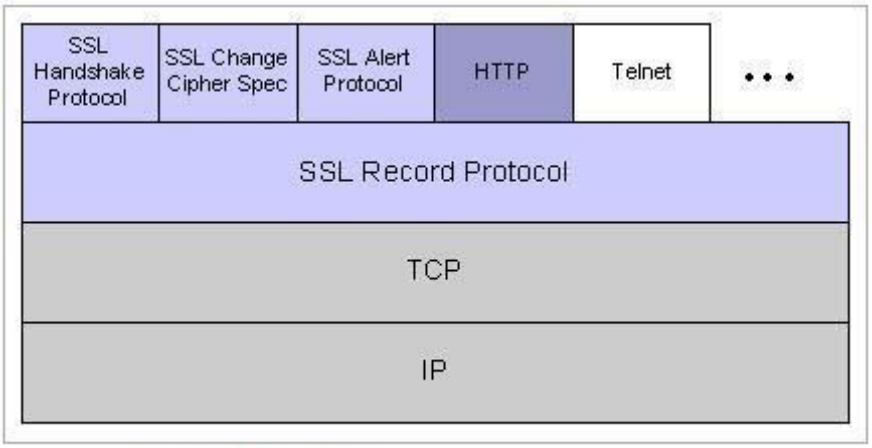
Security at the Transport Layer(SSL and TLS) : SSL Architecture, Four Protocols, SSL Message Formats, Transport Layer Security, HTTPS, SSH
Security at the Network layer (IPSec): Two modes, Two Security Protocols, Security Association, Security Policy, Internet Key Exchange.
System Security: Description of the system, users, Trust and Trusted Systems, Buffer Overflow and Malicious Software, Malicious Programs, worms, viruses, Intrusion Detection System(IDS), Firewalls

---

SSL Architecture :

IPSec provides security at the network level and the main advantage is that it is transparent to end users and applications. In addition, IPSec includes a filtering capability so that only selected traffic can be processed. **Secure Socket Layer or Transport Layer Security (SSL/TLS)** provides security just above the TCP at transport layer. Two implementation choices are present here. Firstly, the SSL/TLS can be implemented as a part of TCP/IP protocol suite, thereby being transparent to applications. Alternatively, SSL can be embedded in specific packages like SSL being implemented by Netscape and Microsoft Explorer browsers. **Secure Electronic Transaction (SET)** approach provides application-specific services i.e., according to the security requirements of a particular application. The main advantage of this approach is that service can be tailored to the specific needs of a given application.

SSL provides for secure communication between client and server by allowing mutual authentication, the use of digital signatures for integrity and encryption for privacy. SSL protocol has different versions such as SSLv2.0, SSLv3.0, where SSLv3.0 has an advantage with the addition of support for certificate chain loading. SSL 3.0 is the basis for the Transport Layer Security [TLS] protocol standard. SSL is designed to make use of TCP to provide a reliable end-to-end secure service. SSL is not a single protocol, but rather two layers of protocols as shown below:



SSL Protocol Stack

The SSL Record Protocol provides basic security services to various higher-layer protocols. In particular, the Hypertext Transfer Protocol (HTTP), which provides the transfer service for Web client/server interaction, can operate on top of SSL. Three higher-layer protocols are defined as part of SSL: the Handshake Protocol, The Change Cipher Spec Protocol, and the Alert Protocol. Two important SSL concepts are the SSL session and the SSL connection, which are defined in the specification as follows:

Connection: A connection is a transport (in the OSI layering model definition) that provides a suitable type of service. For SSL, such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.

Session: An SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol. Sessions define a set of cryptographic security parameters, which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.

An SSL session is *stateful*. Once a session is established, there is a current operating state for both read and write (i.e., receive and send). In addition, during the Handshake Protocol, pending read and write states are created. Upon successful conclusion of the Handshake Protocol, the pending states become the current states. An SSL session may include multiple secure connections; in addition, parties may have multiple simultaneous sessions.

A **session state** is defined by the following parameters:
Session identifier: An arbitrary byte sequence chosen by the server to identify an active or resumable session state.

Peer certificate*:* An X509.v3 certificate of the peer. This element of the state may be null.
Compression method: The algorithm used to compress data prior to encryption.

Cipher spec*:* Specifies the bulk data encryption algorithm (such as null, AES, etc.) and a hash algorithm (such as MD5 or SHA-1) used for MAC calculation. It also defines cryptographic attributes such as the hash_size.

Master secret*:* 48-byte secret shared between the client and server.

Is resumable*:* A flag indicating whether the session can be used to initiate new connections.

A **connection state** is defined by the following parameters:
Server and client random: Byte sequences that are chosen by the server and client for each connection.

Server write MAC secret: The secret key used in MAC operations on data sent by the server.

Client write MAC secret: The secret key used in MAC operations on data sent by the client.

Server write key: The conventional encryption key for data encrypted by the server and decrypted by the client.

Client write key: The conventional encryption key for data encrypted by the client and decrypted by the server.
***Initialization vectors*:** When a block cipher in CBC mode is used, an initialization vector (IV) is maintained for each key. This field is first initialized by the SSL Handshake Protocol. Thereafter the final ciphertext block from each record is preserved for use as the IV with the following record.

Sequence numbers: Each party maintains separate sequence numbers for transmitted and received messages for each connection. When a party sends or receives a change cipher spec message, the appropriate sequence number is set to zero. Sequence numbers may not exceed 264-1.

Four Protocols

SSL Record Protocol
The SSL Record Protocol provides two services for SSL connections:
Confidentiality: The Handshake Protocol defines a shared secret key that is used for conventional encryption of SSL payloads.

Message Integrity: The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC).

The Record Protocol takes an application message to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC, encrypts, adds a header, and transmits the resulting unit in a TCP segment. Received data are decrypted, verified, decompressed, and reassembled and then delivered to higher-level users. The overall operation of the SSL Record Protocol is shown below:

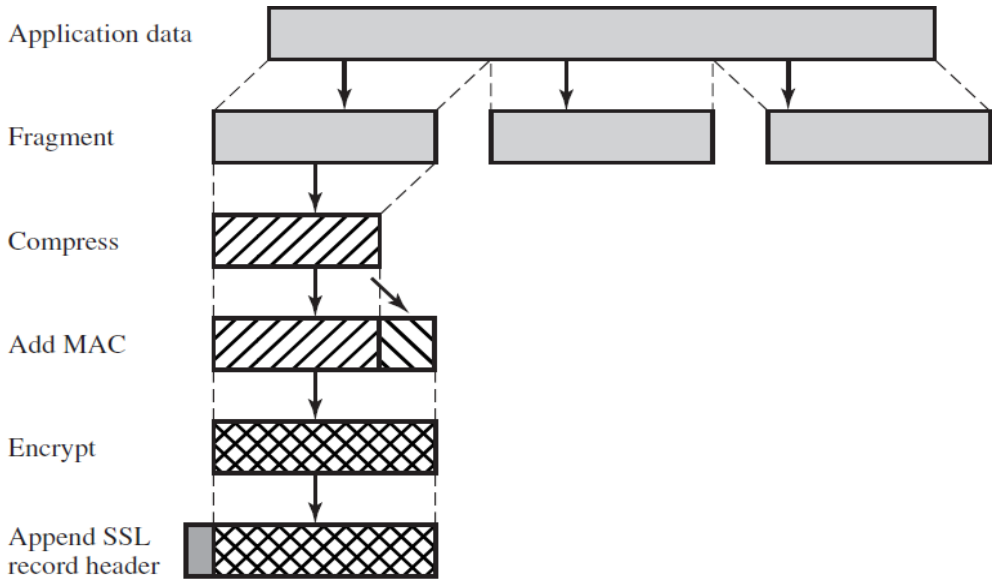
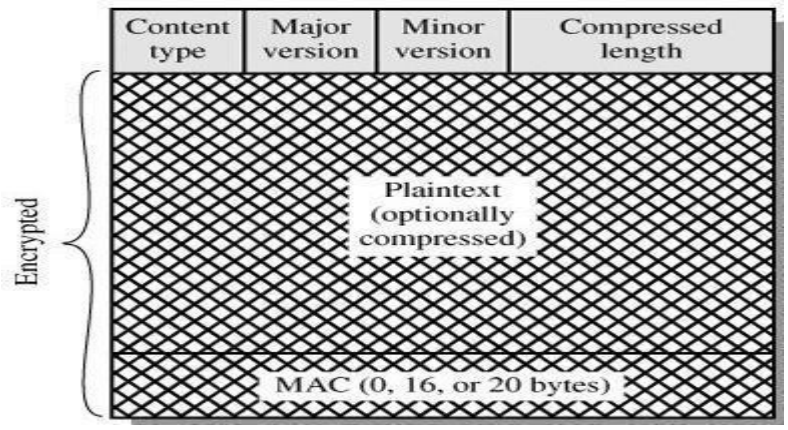


Figure 16.3 SSL Record Protocol Operation

The first step is fragmentation. Each upper-layer message is fragmented into blocks of 214 bytes (16384 bytes) or less. Next, compression is optionally applied. Compression must be lossless and may not increase the content length by more than 1024 bytes. The next step in processing is to compute a message authentication code over the compressed data. For this purpose, a shared secret key is used.

SSL Record Format



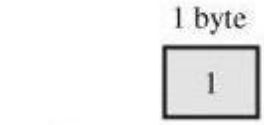
The final step of SSL Record Protocol processing is to prepend a header, consisting of the following fields:
Content Type (8 bits): The higher layer protocol used to process the enclosed fragment.
Major Version (8 bits): Indicates major version of SSL in use. For SSLv3, the value is 3.

Minor Version (8 bits): Indicates minor version in use. For SSLv3, the value is 0.
Compressed Length (16 bits): The length in bytes of the plaintext fragment (or compressed fragment if compression is used). The maximum value is 214 + 2048.

The content types that have been defined are change_cipher_spec, alert, handshake, and application data.

SSL Change Cipher Spec Protocol

The Change Cipher Spec Protocol is one of the three SSL-specific protocols that use the SSL Record Protocol, and it is the simplest. This protocol consists of a single message, which consists of a single byte with the value 1.



(a) Change Cipher Spec Protocol

The sole purpose of this message is to cause the pending state to be copied into the current state, which updates the cipher suite to be used on this connection.

SSL Alert Protocol

The Alert Protocol is used to convey SSL-related alerts to the peer entity. As with other applications that use SSL, alert messages are compressed and encrypted, as specified by the current state. Each message in this protocol consists of two bytes.

(b) Alert Protocol

The first byte takes the value warning(1) or fatal(2) to convey the severity of the message. If the level is fatal, SSL immediately terminates the connection. Other connections on the same session may continue, but no new connections on this session may be established. The second byte contains a code that indicates the specific alert. The fatal alerts are listed below

unexpected_message: An inappropriate message was received.

bad_record_mac: An incorrect MAC was received.

decompression_failure: The decompression function received improper input (e.g., unable to decompress or decompress to greater than maximum allowable length).

handshake_failure: Sender was unable to negotiate an acceptable set of security parameters given the options available.

illegal_parameter: A field in a handshake message was out of range or inconsistent with other fields. The remainder of the alerts are given below:

close_notify: Notifies the recipient that the sender will not send any more messages on this connection. Each party is required to send a close_notify alert before closing the write side of a connection.

• **no_certificate**: May be sent in response to a certificate request if no appropriate certificate is available.

bad_certificate: A received certificate was corrupt (e.g., contained a signature that did not verify).

unsupported_certificate: The type of the received certificate is not supported.

certificate_revoked: A certificate has been revoked by its signer.

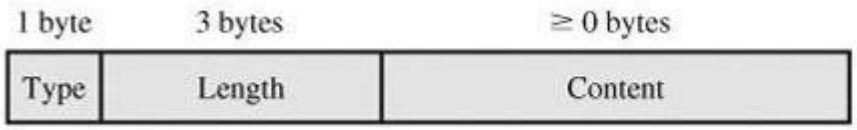
certificate_expired: A certificate has expired.

certificate_unknown: Some other unspecified issue arose in processing the certificate, rendering it unacceptable.

SSL Handshake Protocol

SSL Handshake protocol ensures establishment of reliable and secure session between client and server and also allows server & client to: authenticate each other to negotiate encryption & MAC algorithms to negotiate cryptographic keys to be used .

The Handshake Protocol consists of a series of messages exchanged by client and server. All of these have the format shown below and each message has three fields:



(c) Handshake Protocol

Type (1 byte): Indicates one of 10 messages.

Length (3 bytes): The length of the message in bytes.

Content (>=0 bytes): The parameters associated with this message

The following figure shows the initial exchange needed to establish a logical connection between client and server. The exchange can be viewed as having four phases. in phases
o Establish Security Capabilities
o Server Authentication and Key Exchange
o Client Authentication and Key Exchange
o Finish

Phase 1. Establish Security Capabilities
This phase is used to initiate a logical connection and to establish the security capabilities that will be associated with it. The exchange is initiated by the client, which sends a client_hello message with the following parameters:

Version: The highest SSL version understood by the client.

Random: A client-generated random structure, consisting of a 32-bit timestamp and 28 bytes generated by a secure random number generator. These values serve as nonces and are used during key exchange to prevent replay attacks.

Session ID: A variable-length session identifier. A nonzero value indicates that the client wishes to update the parameters of an existing connection or create a new connection on this session. A zero value indicates that the client wishes to establish a new connection on a new session.

Cipher Suite: This is a list that contains the combinations of cryptographic algorithms supported by the client, in decreasing order of preference. Each element of the list (each cipher suite) defines both a key exchange algorithm and a CipherSpec.

Compression Method: This is a list of the compression methods the client supports.

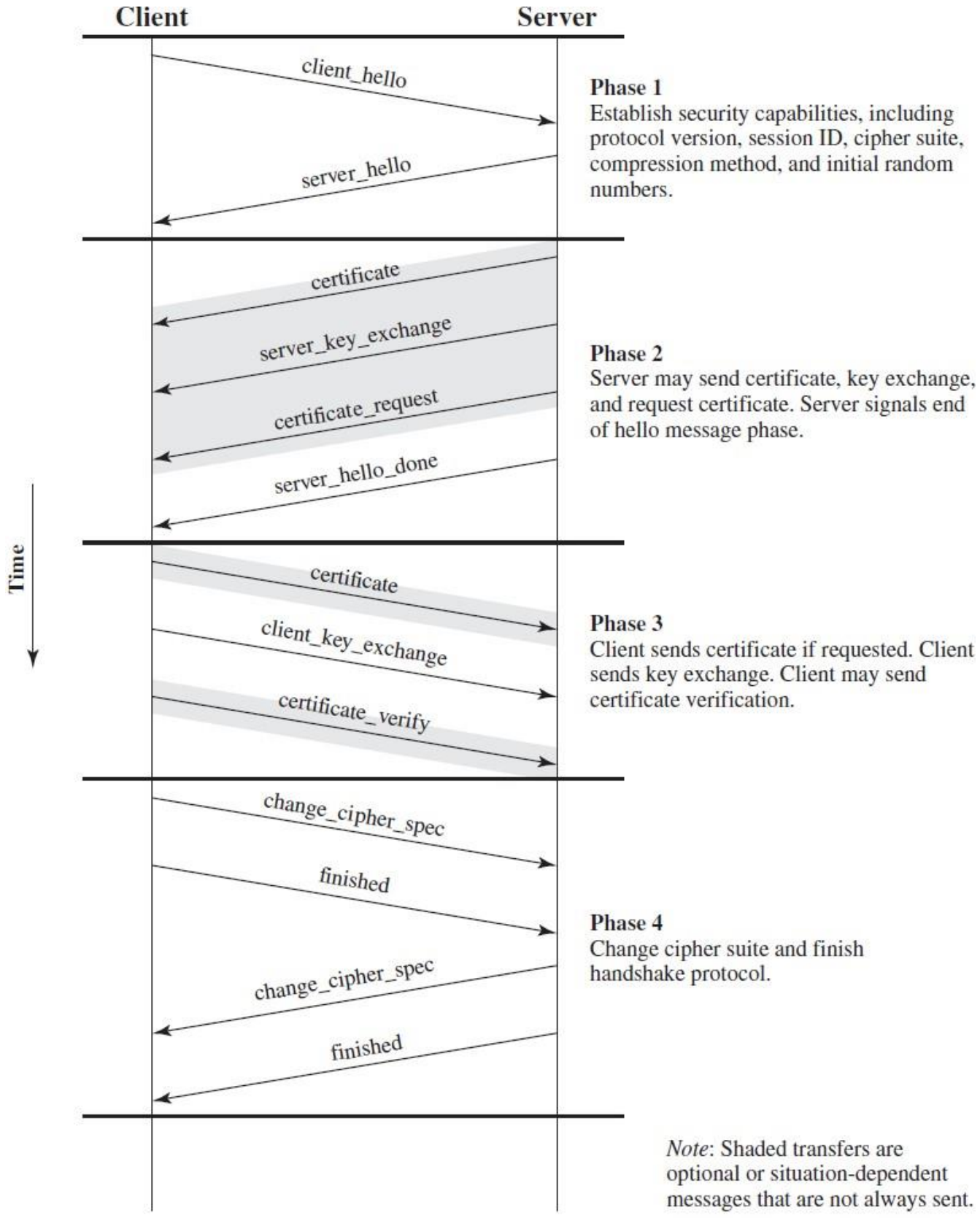


Figure 16.6 Handshake Protocol Action

SSL Message Formats

Transport Layer Security :

In order to provide an open Internet standard of SSL, IETF released The Transport Layer Security (TLS) protocol in January 1999. TLS is defined as a proposed Internet Standard in RFC 5246.

Salient Features

- TLS protocol has same objectives as SSL.
- It enables client/server applications to communicate in a secure manner by authenticating, preventing eavesdropping and resisting message modification.
- TLS protocol sits above the reliable connection-oriented transport TCP layer in the networking layers stack.
- The architecture of TLS protocol is similar to SSLv3 protocol. It has two sub protocols: the TLS Record protocol and the TLS Handshake protocol.
- Though SSLv3 and TLS protocol have similar architecture, several changes were made in architecture and functioning particularly for the handshake protocol.

Comparison of TLS and SSL Protocols

There are main eight differences between TLS and SSLv3 protocols. These are as follows −

- **Protocol Version** − The header of TLS protocol segment carries the version number 3.1 to differentiate between number 3 carried by SSL protocol segment header.
- **Message Authentication** − TLS employs a keyed-hash message authentication code (H-MAC). Benefit is that H-MAC operates with any hash function, not just MD5 or SHA, as explicitly stated by the SSL protocol.
- **Session Key Generation** − There are two differences between TLS and SSL protocol for generation of key material.
    - Method of computing pre-master and master secrets is similar. But in TLS protocol, computation of master secret uses the HMAC standard and pseudorandom function (PRF) output instead of ad-hoc MAC.
    - The algorithm for computing session keys and initiation values (IV) is different in TLS than SSL protocol.
- Alert Protocol Message −
    - TLS protocol supports all the messages used by the Alert protocol of SSL, except *No certificate* alert message being made redundant. The client sends empty certificate in case client authentication is not required.
    - Many additional Alert messages are included in TLS protocol for other error conditions such as *record_overflow*, *decode_error* etc.
- **Supported Cipher Suites** − SSL supports RSA, Diffie-Hellman and Fortezza cipher suites. TLS protocol supports all suits except Fortezza.
- **Client Certificate Types** − TLS defines certificate types to be requested in a *certificate_request* message. SSLv3 support all of these. Additionally, SSL support certain other types of certificate such as Fortezza.
- CertificateVerify and Finished Messages −

- In SSL, complex message procedure is used for the *certificate_verify*message. With TLS, the verified information is contained in the handshake messages itself thus avoiding this complex procedure.
- Finished message is computed in different manners in TLS and SSLv3.

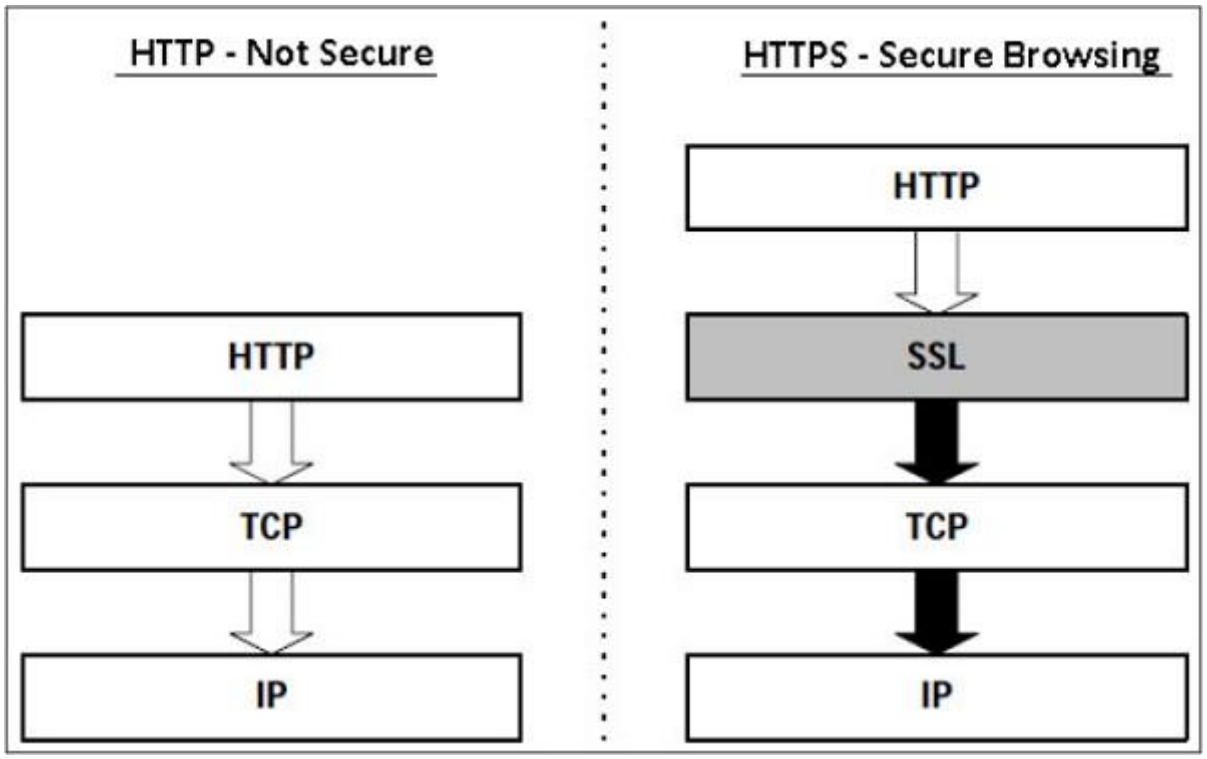
- **Padding of Data** − In SSL protocol, the padding added to user data before encryption is the minimum amount required to make the total data-size equal to a multiple of the cipher‘s block length. In TLS, the padding can be any amount that results in data-size that is a multiple of the cipher‘s block length, up to a maximum of 255 bytes.

The above differences between TLS and SSLv3 protocols are summarized in the following table.

	SSL v3.0	TLS v1.0
Protocol version in messages	3.0	3.1
Alert protocol message types	12	23
Message authentication	ad hoc	standard
Key material generation	ad hoc	PRF
CertificateVerify	complex	simple
Finished	ad hoc	PRF
Baseline cipher suites	includes Fortezza	no Fortezza

HTTPS

Hyper Text Transfer Protocol (HTTP) protocol is used for web browsing. The function of HTTPS is similar to HTTP. The only difference is that HTTPS provides –secure| web browsing. HTTPS stands for HTTP over SSL. This protocol is used to provide the encrypted and authenticated connection between the client web browser and the website server.



The secure browsing through HTTPS ensures that the following content are encrypted −

- URL of the requested web page.

- Web page contents provided by the server to the user client.
- Contents of forms filled in by user.
- Cookies established in both directions.

Working of HTTPS

HTTPS application protocol typically uses one of two popular transport layer security protocols - SSL or TLS. The process of secure browsing is described in the following points.

- You request a HTTPS connection to a webpage by entering https:// followed by URL in the browser address bar.
- Web browser initiates a connection to the web server. Use of https invokes the use of SSL protocol.
- An application, browser in this case, uses the system port 443 instead of port 80 (used in case of http).
- The SSL protocol goes through a handshake protocol for establishing a secure session as discussed in earlier sections.
- The website initially sends its SSL Digital certificate to your browser. On verification of certificate, the SSL handshake progresses to exchange the shared secrets for the session.
- When a trusted SSL Digital Certificate is used by the server, users get to see a padlock icon in the browser address bar. When an Extended Validation Certificate is installed on a website, the address bar turns green.

- Once established, this session consists of many secure connections between the web server and the browser.

Use of HTTPS

- Use of HTTPS provides confidentiality, server authentication and message integrity to the user. It enables safe conduct of e-commerce on the Internet.
- Prevents data from eavesdropping and denies identity theft which are common attacks on HTTP.

Present day web browsers and web servers are equipped with HTTPS support. The use of HTTPS over HTTP, however, requires more computing power at the client and the server end to carry out encryption and SSL handshake.

SSH

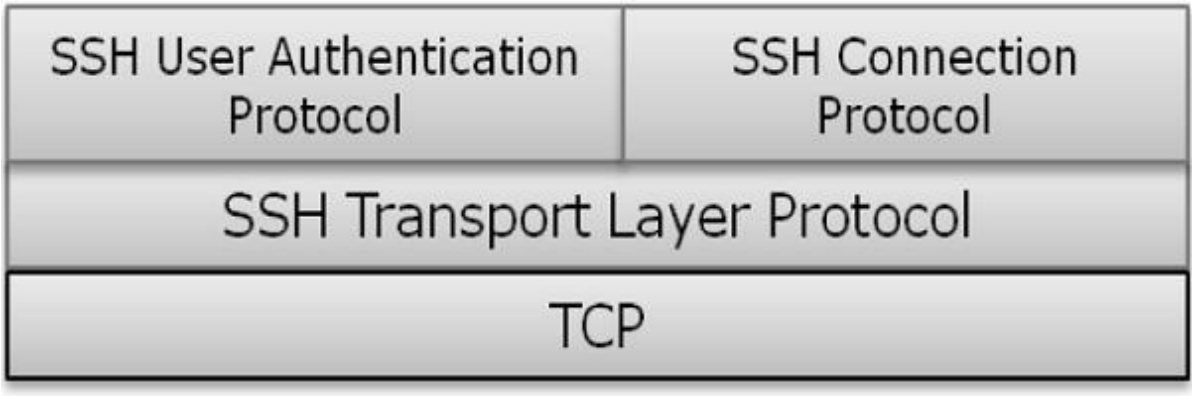
Secure Shell Protocol (SSH)

The salient features of SSH are as follows −

- SSH is a network protocol that runs on top of the TCP/IP layer. It is designed to replace the TELNET which provided unsecure means of remote logon facility.
- SSH provides a secure client/server communication and can be used for tasks such as file transfer and e-mail.
- SSH2 is a prevalent protocol which provides improved network communication security over earlier version SSH1.

SSH Defined

SSH is organized as three sub-protocols.

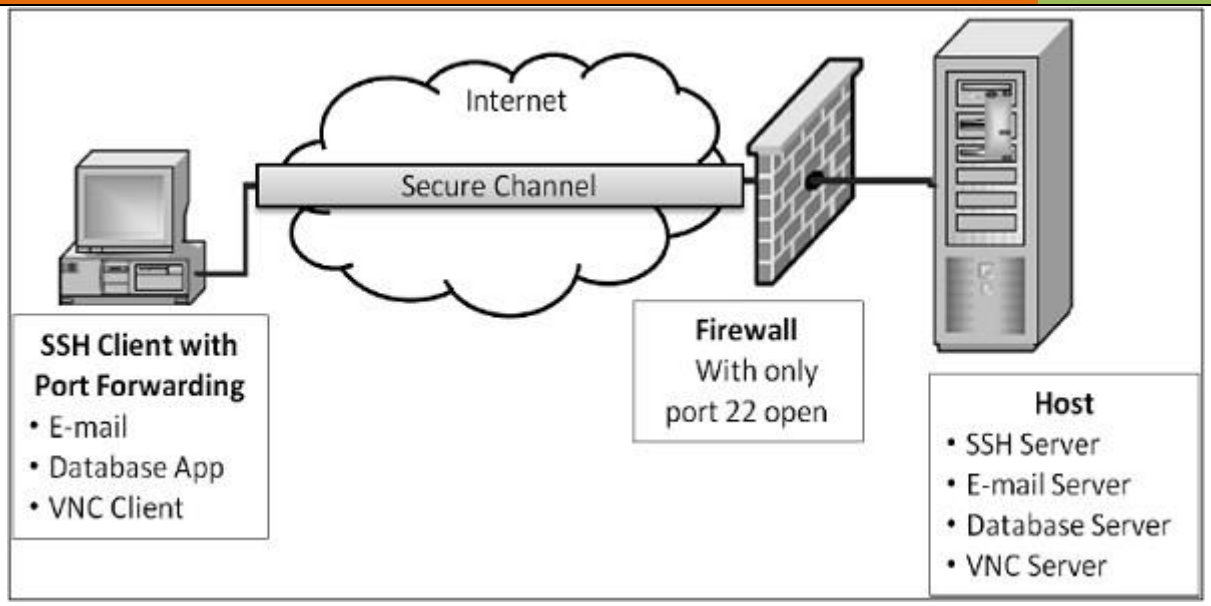


- **Transport Layer Protocol** − This part of SSH protocol provides data confidentiality, server (host) authentication, and data integrity. It may optionally provide data compression as well.
    - **Server Authentication** − Host keys are asymmetric like public/private keys. A server uses a public key to prove its identity to a client. The client verifies that contacted server is a ‒known‖ host from the database it maintains. Once the server is authenticated, session keys are generated.
    - **Session Key Establishment** − After authentication, the server and the client agree upon cipher to be used. Session keys are generated by both the client and the server. Session keys are generated before user authentication so that usernames and passwords can be sent encrypted. These keys are generally replaced at regular intervals (say, every hour) during the session and are destroyed immediately after use.
    - **Data Integrity** − SSH uses Message Authentication Code (MAC) algorithms to for data integrity check. It is an improvement over 32 bit CRC used by SSH1.
- **User Authentication Protocol** − This part of SSH authenticates the user to the server. The server verifies that access is given to intended users only. Many authentication methods are currently used such as, typed passwords, Kerberos, public-key authentication, etc.
- **Connection Protocol** − This provides multiple logical channels over a single underlying SSH connection.

SSH Services

SSH provides three main services that enable provision of many secure solutions. These services are briefly described as follows −

- **Secure Command-Shell (Remote Logon)** − It allows the user to edit files, view the contents of directories, and access applications on connected device. Systems administrators can remotely start/view/stop services and processes, create user accounts, and change file/directories permissions and so on. All tasks that are feasible at a machine‘s command prompt can now be performed securely from the remote machine using secure remote logon.
- **Secure File Transfer** − SSH File Transfer Protocol (SFTP) is designed as an extension for SSH-2 for secure file transfer. In essence, it is a separate protocol layered over the Secure Shell protocol to handle file transfers. SFTP encrypts both the username/password and the file data being transferred. It uses the same port as the Secure Shell server, i.e. system port no 22.
- **Port Forwarding (Tunneling)** − It allows data from unsecured TCP/IP based applications to be secured. After port forwarding has been set up, Secure Shell reroutes traffic from a program (usually a client) and sends it across the encrypted tunnel to the program on the other side (usually a server). Multiple applications can transmit data over a single multiplexed secure channel, eliminating the need to open many ports on a firewall or router.



Benefits & Limitations

The benefits and limitations of employing communication security at transport layer are as follows −

- Benefits
  - Transport Layer Security is transparent to applications.
  - Server is authenticated.
  - Application layer headers are hidden.
  - It is more fine-grained than security mechanisms at layer 3 (IPsec) as it works at the transport connection level.
- Limitations
  - Applicable to TCP-based applications only (not UDP).
  - TCP/IP headers are in clear.
  - Suitable for direct communication between the client and the server. Does not cater for secure applications using chain of servers (e.g. email)
  - SSL does not provide non-repudiation as client authentication is optional.
  - If needed, client authentication needs to be implemented above SSL.

(Part – 2)

Applications of IPSec

IPSec provides the capability to secure communications across a LAN, across private and public wide area networks (WAN‘s), and across the Internet.

Secure branch office connectivity over the Internet: A company can build a secure virtual private network over the Internet or over a public WAN. This enables a business to rely heavily on the Internet and reduce its need for private networks, saving costs and network management overhead.

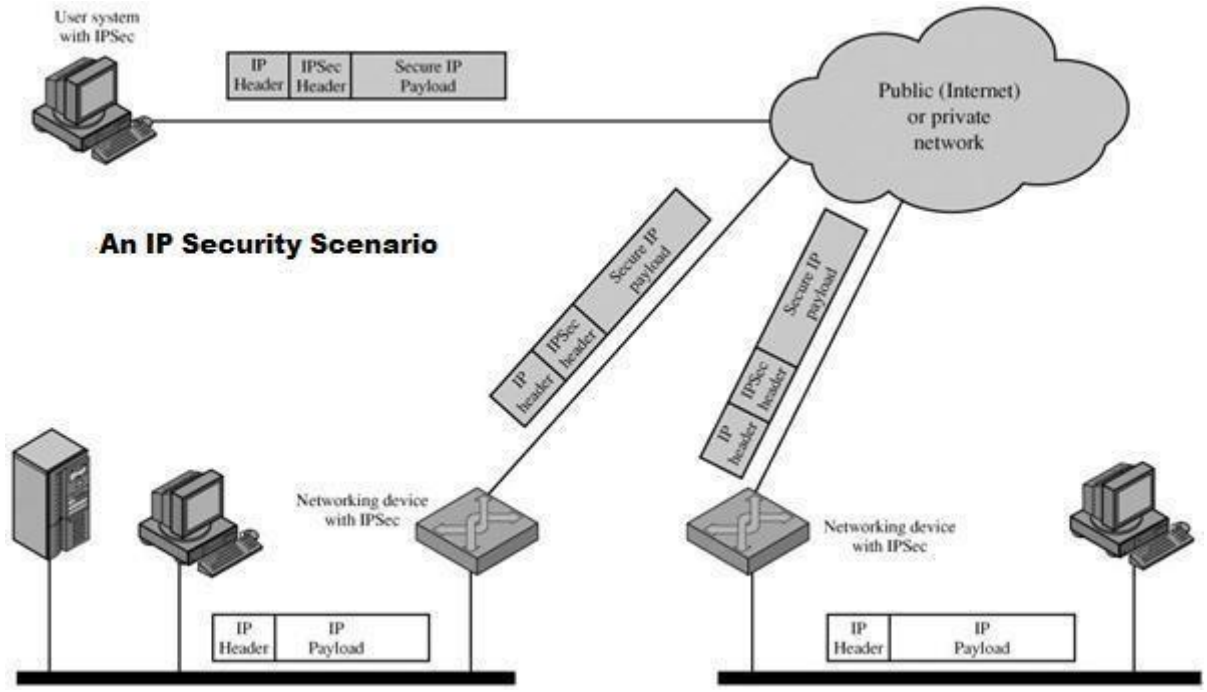
Secure remote access over the Internet: An end user whose system is equipped with IP security protocols can make a local call to an Internet service provider (ISP) and gain secure access to a company network. This reduces the cost of toll charges for travelling employees and telecommuters.

Establishing extranet and intranet connectivity with partners: IPSec can be used to secure communication with other organizations, ensuring authentication and confidentiality and providing a key exchange mechanism.

Enhancing electronic commerce security: Even though some Web and electronic commerce applications have built-in security protocols, the use of IPSec enhances that security.

The principal feature of IPSec enabling it to support varied applications is that it can encrypt and/or authenticate all traffic at IP level. Thus, all distributed applications, including remote logon, client/server, e-mail, file transfer, Web access, and so on, can be secured.

The following figure shows a typical scenario of IPSec usage. An organization maintains LANs at dispersed locations. Non secure IP traffic is conducted on each LAN.



The IPSec protocols operate in networking devices, such as a router or firewall that connect each LAN to the outside world. The IPSec networking device will typically encrypt and compress all traffic going into the WAN, and decrypt and decompress traffic coming from the WAN; these operations are transparent to workstations and servers on the LAN. Secure transmission is also possible with individual users who dial into the WAN. Such user workstations must implement the IPSec protocols to provide security.

Benefits of IPSec

The benefits of IPSec are listed below:
IPSec in a firewall/router provides strong security to all traffic crossing the perimeter
IPSec in a firewall is resistant to bypass
IPSec is below transport layer(TCP,UDP), hence transparent to applications
IPSec can be transparent to end users
IPSec can provide security for individual users if needed (useful for offsite workers and setting up a secure virtual subnetwork for sensitive applications)

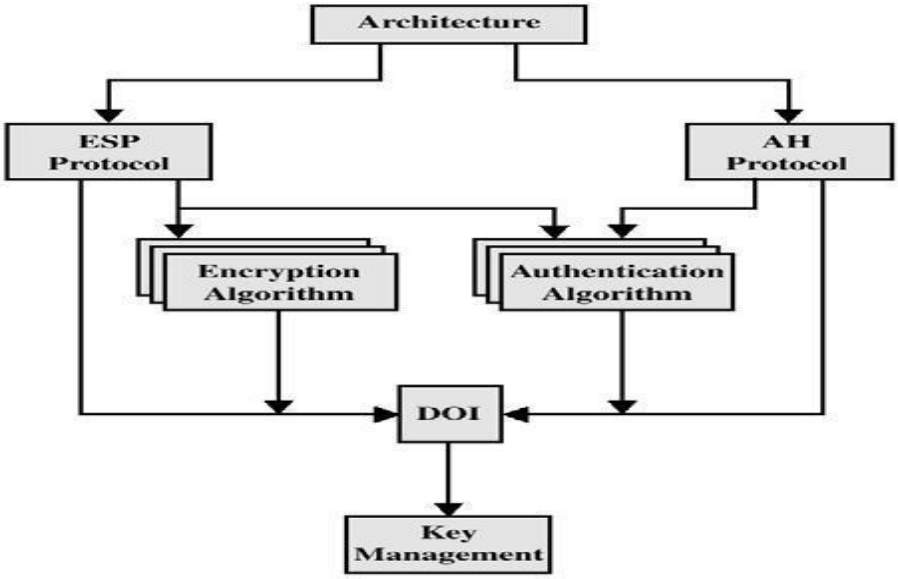
Routing Applications

IPSec also plays a vital role in the routing architecture required for internetworking. It assures that:

- router advertisements come from authorized routers
- neighbor advertisements come from authorized routers

- redirect messages come from the router to which initial packet was sent
- A routing update is not forged

Support for these features is mandatory for IPv6 and optional for IPv4. In both cases, the security features are implemented as extension headers that follow the main IP header. The extension header for authentication is known as the Authentication header; that for encryption is known as the Encapsulating Security Payload (ESP) header. In addition to these four RFCs, a number of additional drafts have been published by the IP Security Protocol Working Group set up by the IETF. The documents are divided into seven groups, as depicted in following figure:



Architecture: Covers the general concepts, security requirements, definitions, and mechanisms defining IPSec technology

Encapsulating Security Payload (ESP): Covers the packet format and general issues related to the use of the ESP for packet encryption and, optionally, authentication.

Authentication Header (AH): Covers the packet format and general issues related to the use of AH for packet authentication.

Encryption Algorithm: A set of documents that describe how various encryption algorithms are used for ESP.

Authentication Algorithm: A set of documents that describe how various authentication algorithms are used for AH and for the authentication option of ESP.

Key Management: Documents that describe key management schemes.

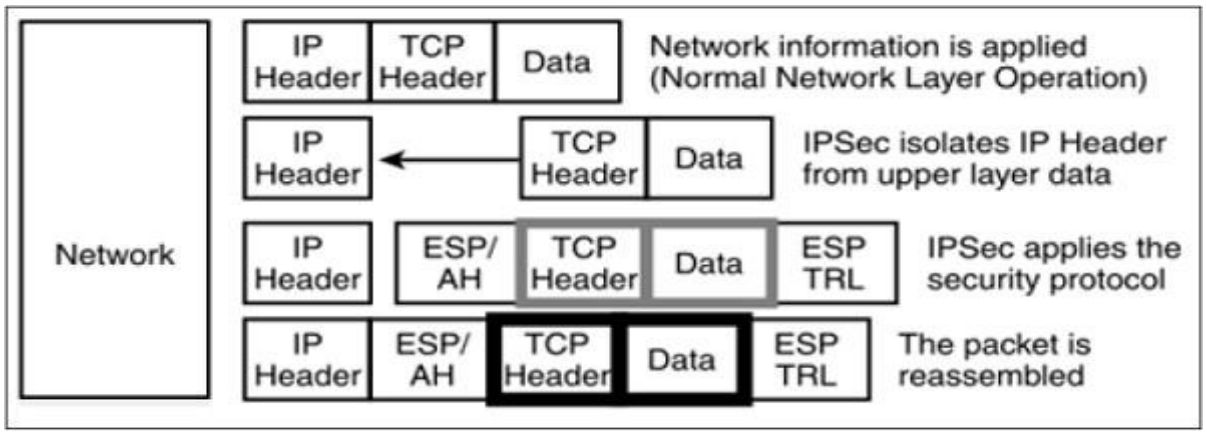
Domain of Interpretation (DOI): Contains values needed for the other documents to relate to each other. These include identifiers for approved encryption and authentication algorithms, as well as operational parameters such as key lifetime.

Two modes (IPsec Communication Modes)

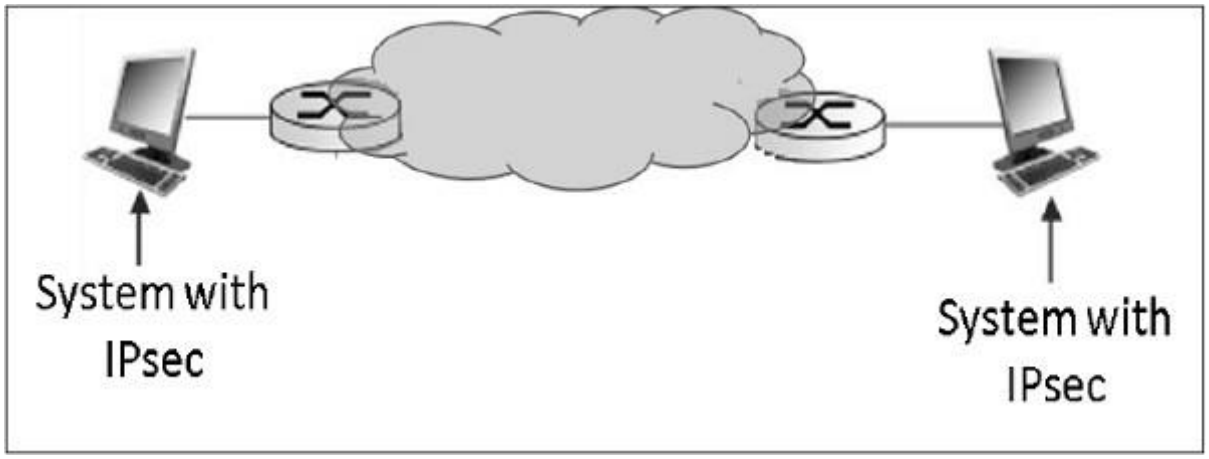
IPsec Communication has two modes of functioning; transport and tunnel modes. These modes can be used in combination or used individually depending upon the type of communication desired.

Transport Mode

- IPsec does not encapsulate a packet received from upper layer.
- The original IP header is maintained and the data is forwarded based on the original attributes set by the upper layer protocol.
- The following diagram shows the data flow in the protocol stack.

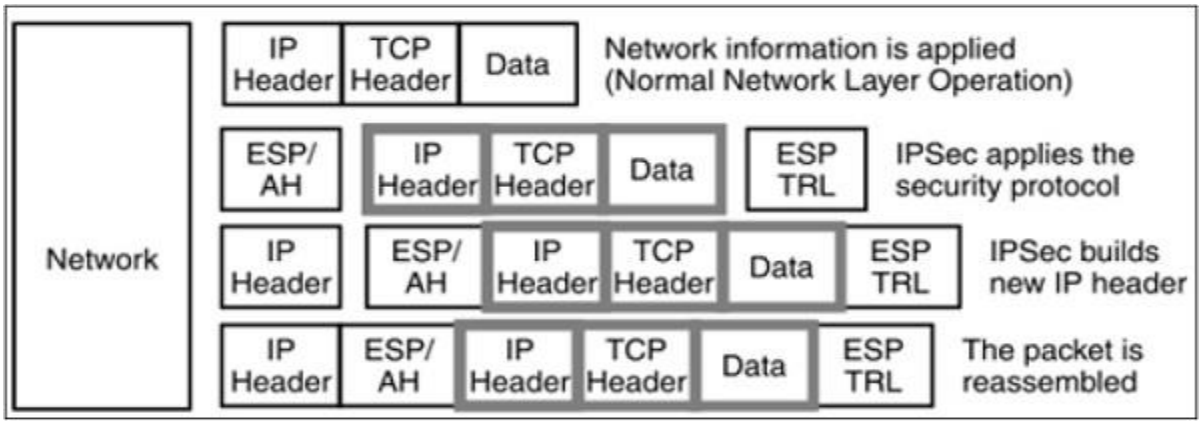


- The limitation of transport mode is that no gateway services can be provided. It is reserved for point-to-point communications as depicted in the following image.

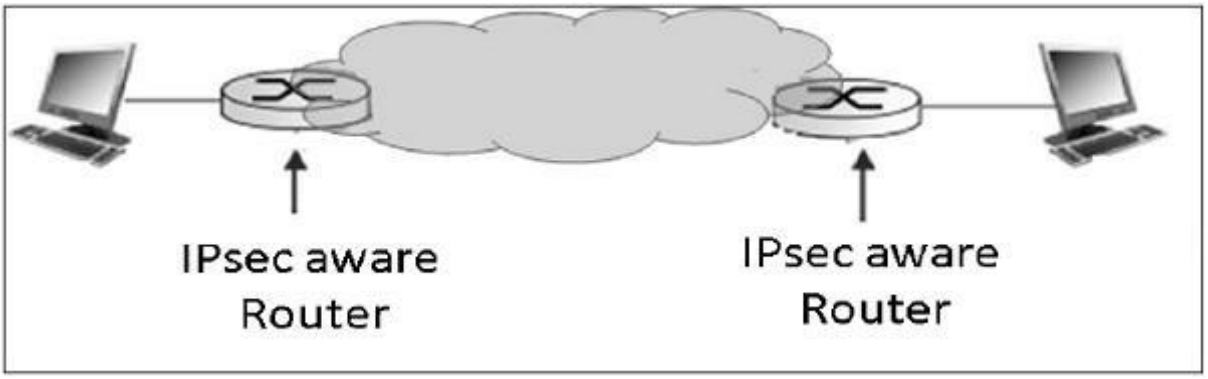


Tunnel Mode

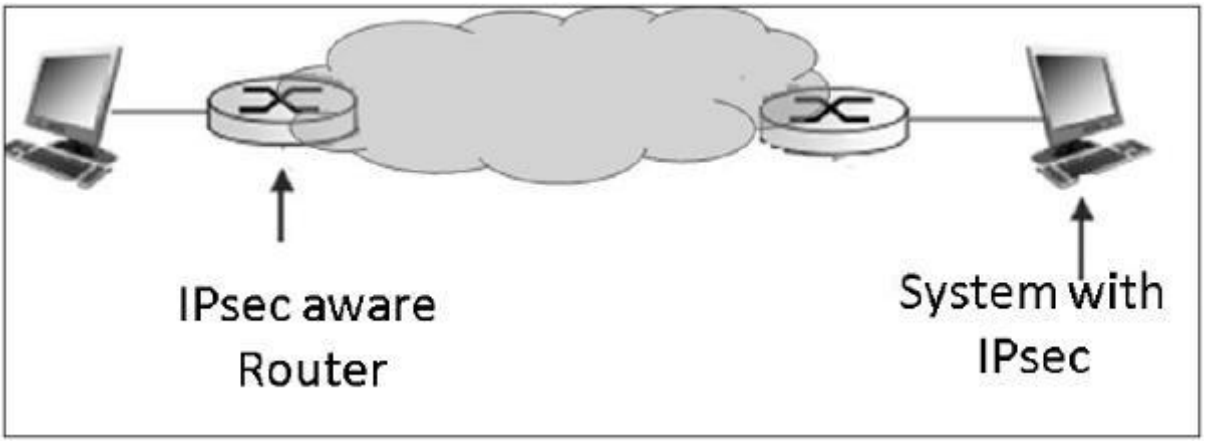
- This mode of IPsec provides encapsulation services along with other security services.
- In tunnel mode operations, the entire packet from upper layer is encapsulated before applying security protocol. New IP header is added.
- The following diagram shows the data flow in the protocol stack.



- Tunnel mode is typically associated with gateway activities. The encapsulation provides the ability to send several sessions through a single gateway.
- The typical tunnel mode communication is as depicted in the following diagram.



- As far as the endpoints are concerned, they have a direct transport layer connection. The datagram from one system forwarded to the gateway is encapsulated and then forwarded to the remote gateway. The remote associated gateway de-encapsulates the data and forwards it to the destination endpoint on the internal network.
- Using IPsec, the tunneling mode can be established between the gateway and individual end system as well.



Two Security Protocols (IPsec Protocols)

IPsec uses the security protocols to provide desired security services. These protocols are the heart of IPsec operations and everything else is designed to support these protocol in IPsec.

Security associations between the communicating entities are established and maintained by the security protocol used.

There are two security protocols defined by IPsec — Authentication Header (AH) and Encapsulating Security Payload (ESP).

Authentication Header

The AH protocol provides service of data integrity and origin authentication. It optionally caters for message replay resistance. However, it does not provide any form of confidentiality.

AH is a protocol that provides authentication of either all or part of the contents of a datagram by the addition of a header. The header is calculated based on the values in the datagram. What parts of the

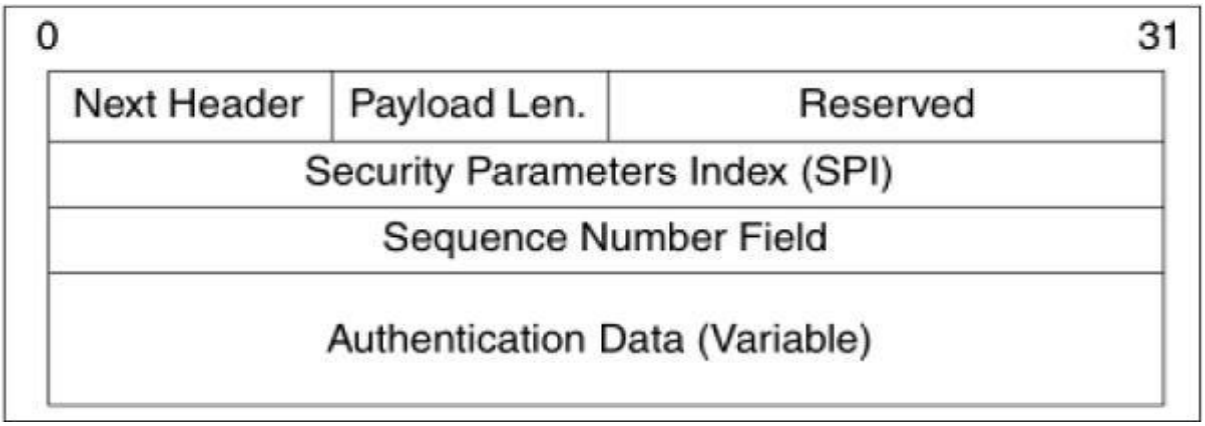
datagram are used for the calculation, and where to place the header, depends on the mode cooperation (tunnel or transport).

The operation of the AH protocol is surprisingly simple. It can be considered similar to the algorithms used to calculate checksums or perform CRC checks for error detection.

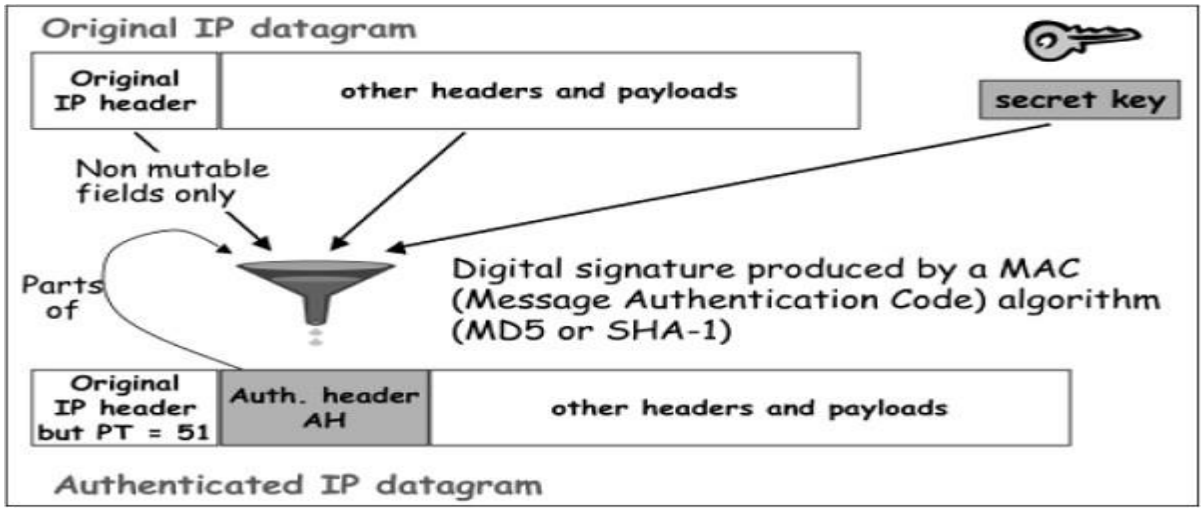
The concept behind AH is the same, except that instead of using a simple algorithm, AH uses special hashing algorithm and a secret key known only to the communicating parties. A security association between two devices is set up that specifies these particulars.

The process of AH goes through the following phases.

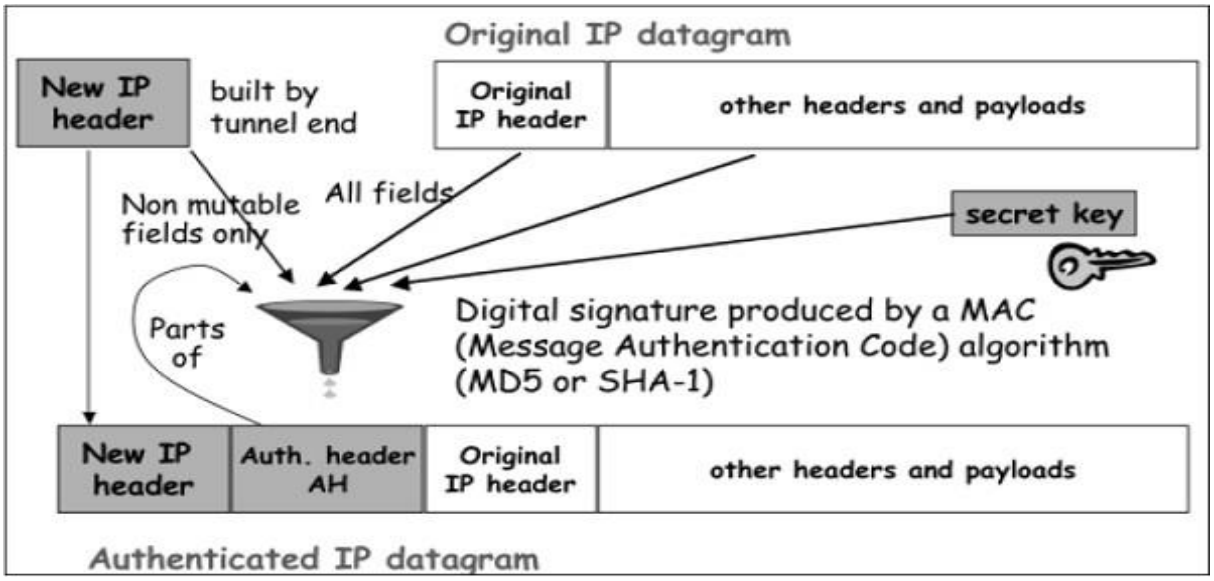
- When IP packet is received from upper protocol stack, IPsec determine the associated Security Association (SA) from available information in the packet; for example, IP address (source and destination).
- From SA, once it is identified that security protocol is AH, the parameters of AH header are calculated. The AH header consists of the following parameters −



- The header field specifies the protocol of packet following AH header. Sequence Parameter Index (SPI) is obtained from SA existing between communicating parties.
- Sequence Number is calculated and inserted. These numbers provide optional capability to AH to resist replay attack.
- Authentication data is calculated differently depending upon the communication mode.
- In transport mode, the calculation of authentication data and assembling of final IP packet for transmission is depicted in the following diagram. In original IP header, change is made only in protocol number as 51 to indicated application of AH.



- In Tunnel mode, the above process takes place as depicted in the following diagram.



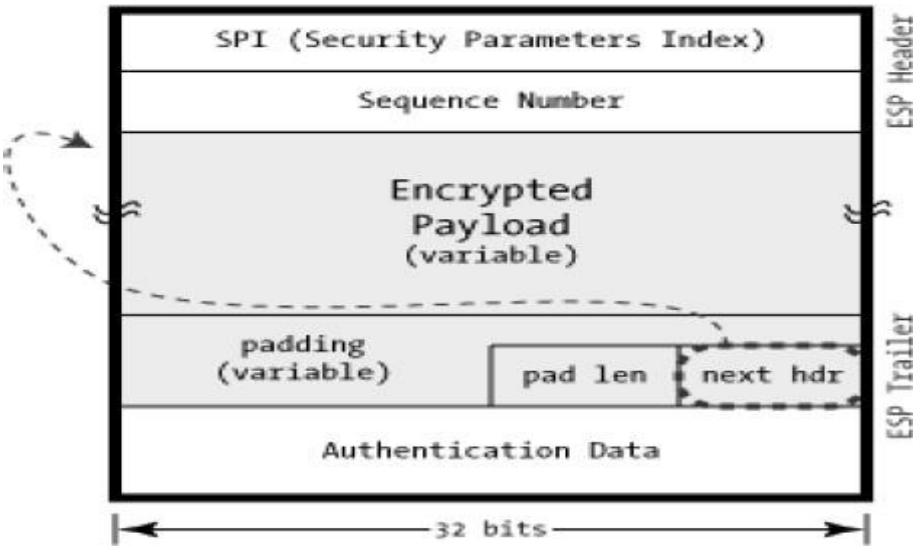
Encapsulation Security Protocol (ESP)

ESP provides security services such as confidentiality, integrity, origin authentication, and optional replay resistance. The set of services provided depends on options selected at the time of Security Association (SA) establishment.

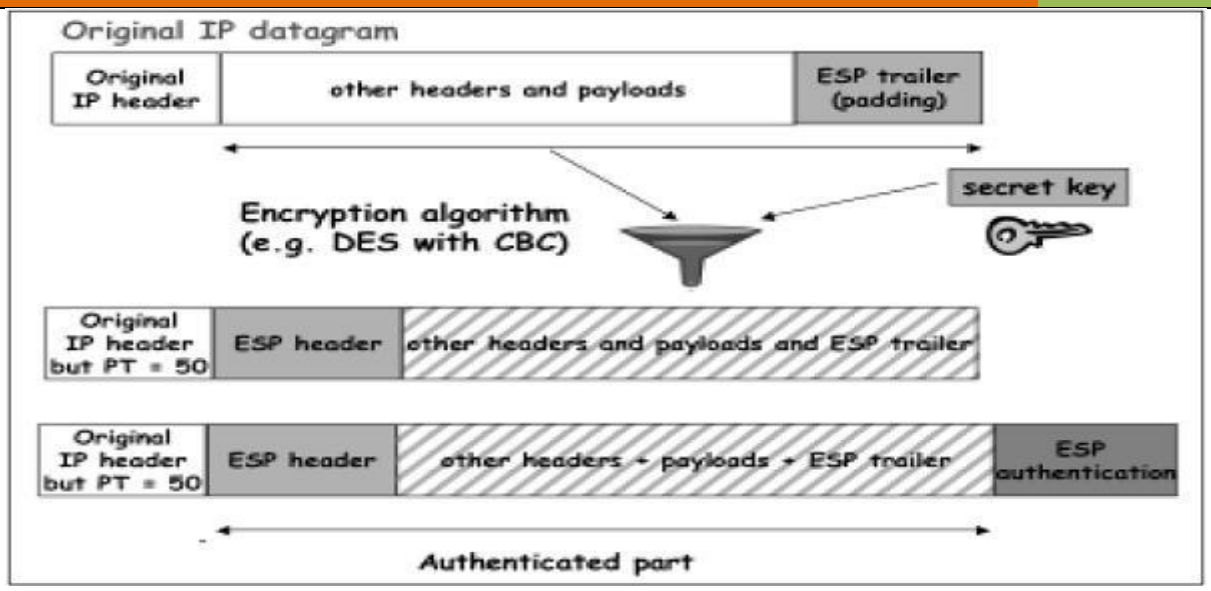
In ESP, algorithms used for encryption and generating authenticator are determined by the attributes used to create the SA.

The process of ESP is as follows. The first two steps are similar to process of AH as stated above.

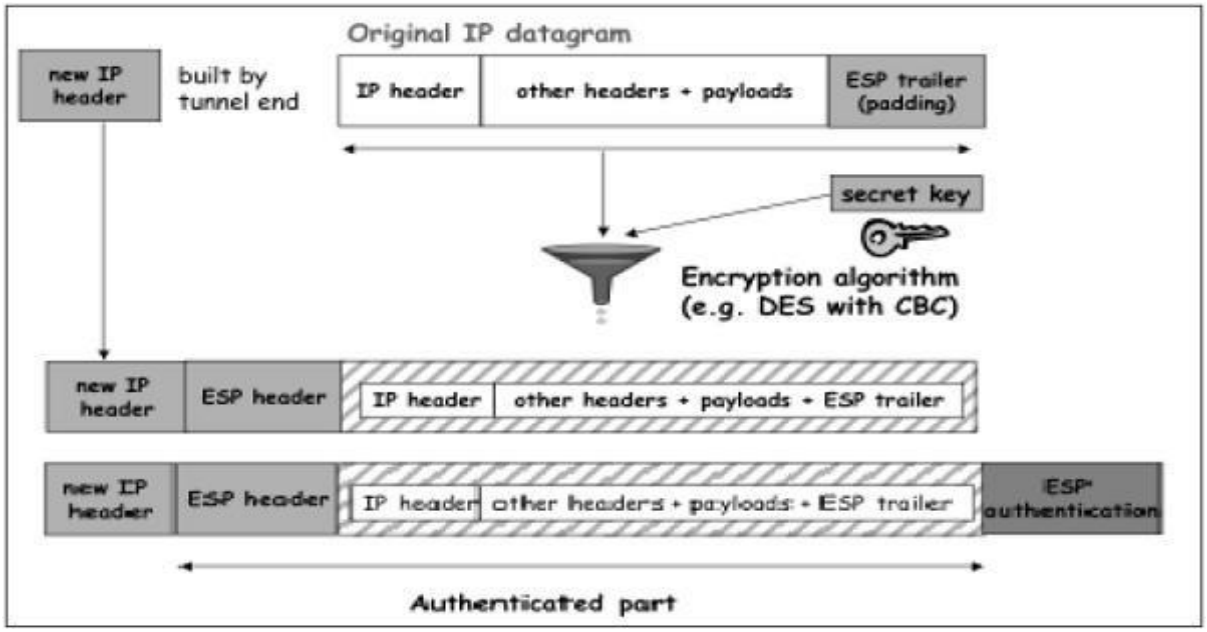
- Once it is determined that ESP is involved, the fields of ESP packet are calculated. The ESP field arrangement is depicted in the following diagram.



- Encryption and authentication process in transport mode is depicted in the following diagram.



- In case of Tunnel mode, the encryption and authentication process is as depicted in the following diagram.



Although authentication and confidentiality are the primary services provided by ESP, both are optional. Technically, we can use NULL encryption without authentication. However, in practice, one of the two must be implemented to use ESP effectively.

The basic concept is to use ESP when one wants authentication and encryption, and to use AH when one wants extended authentication without encryption.

Security Association

Security Association (SA) is the foundation of an IPsec communication. The features of SA are −

- Before sending data, a virtual connection is established between the sending entity and the receiving entity, called ‑Security Association (SA)‖.
- IPsec provides many options for performing network encryption and authentication. Each IPsec connection can provide encryption, integrity, authenticity, or all three services. When the security service is determined, the two IPsec peer entities must determine exactly which algorithms to use

(for example, DES or 3DES for encryption; MD5 or SHA-1 for integrity). After deciding on the algorithms, the two devices must share session keys.

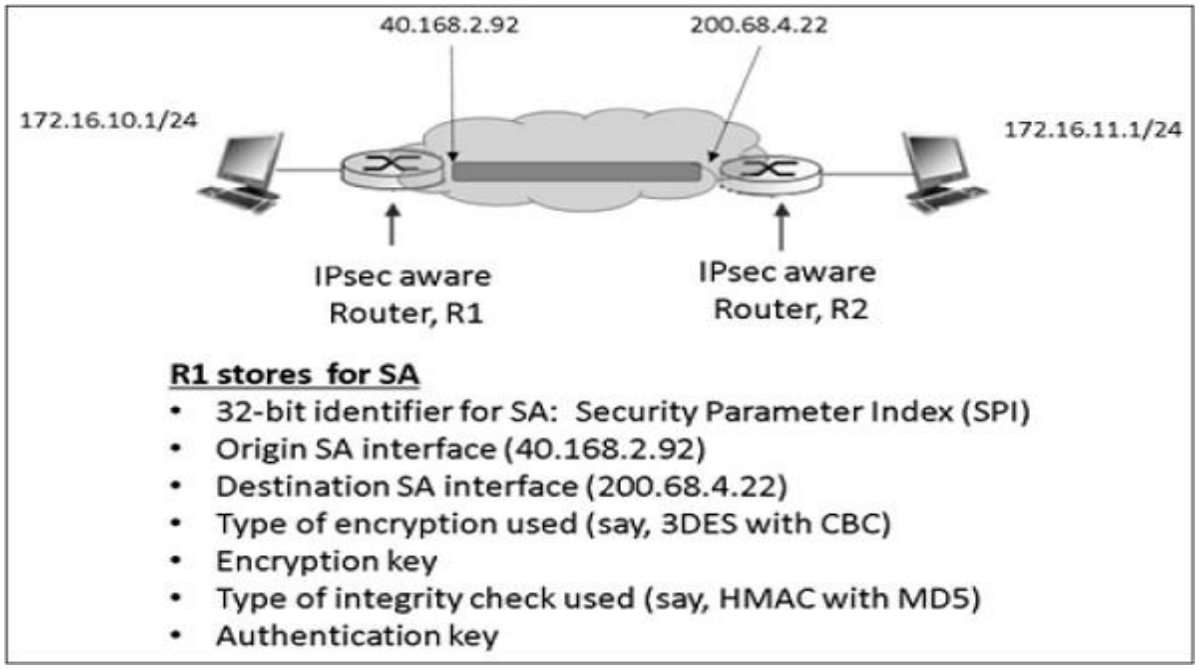
- SA is a set of above communication parameters that provides a relationship between two or more systems to build an IPsec session.
- SA is simple in nature and hence two SAs are required for bi-directional communications.
- SAs are identified by a Security Parameter Index (SPI) number that exists in the security protocol header.
- Both sending and receiving entities maintain state information about the SA. It is similar to TCP endpoints which also maintain state information. IPsec is connection-oriented like TCP.

Parameters of SA

Any SA is uniquely identified by the following three parameters −

- Security Parameters Index (SPI).
  - It is a 32-bit value assigned to SA. It is used to distinguish among different SAs terminating at the same destination and using the same IPsec protocol.
  - Every packet of IPsec carries a header containing SPI field. The SPI is provided to map the incoming packet to an SA.
  - The SPI is a random number generated by the sender to identify the SA to the recipient.
- **Destination IP Address** − It can be IP address of end router.
- **Security Protocol Identifier** − It indicates whether the association is an AH or ESP SA.

Example of SA between two router involved in IPsec communication is shown in the following diagram.



Security Administrative Databases

In IPsec, there are two databases that control the processing of IPsec datagram. One is the Security Association Database (SAD) and the other is the Security Policy Database (SPD). Each communicating endpoint using IPsec should have a logically separate SAD and SPD.

Security Association Database

In IPsec communication, endpoint holds SA state in Security Association Database (SAD). Each SA entry in SAD database contains nine parameters as shown in the following table −

S.No.	Parameters & Description
1	**Sequence Number Counter** For outbound communications. This is the 32-bit sequence number provided in the AH or ESP headers.
2	**Sequence Number Overflow Counter** Sets an option flag to prevent further communications utilizing the specific SA
3	**32-bit anti-replay window** Used to determine whether an inbound AH or ESP packet is a replay
4	**Lifetime of the SA** Time till SA remain active
5	**Algorithm - AH** Used in the AH and the associated key
6	**Algorithm - ESP Auth** Used in the authenticating portion of the ESP header
7	**Algorithm - ESP Encryption** Used in the encryption of the ESP and its associated key information
8	**IPsec mode of operation** Transport or tunnel mode
9	**Path MTU(PMTU)** Any observed path maximum transmission unit (to avoid fragmentation)

All SA entries in the SAD are indexed by the three SA parameters: Destination IP address, Security Protocol Identifier, and SPI.

Security Policy

Security Policy Database

SPD is used for processing outgoing packets. It helps in deciding what SAD entries should be used. If no SAD entry exists, SPD is used to create new ones.

Any SPD entry would contain −

- Pointer to active SA held in SAD.
- Selector fields – Field in incoming packet from upper layer used to decide application of IPsec. Selectors can include source and destination address, port numbers if relevant, application IDs, protocols, etc.

Outgoing IP datagrams go from the SPD entry to the specific SA, to get encoding parameters. Incoming IPsec datagram get to the correct SA directly using the SPI/DEST IP/Protocol triple, and from there extracts the associated SAD entry.

SPD can also specify traffic that should bypass IPsec. SPD can be considered as a packet filter where the actions decided upon are the activation of SA processes.

Internet Key Exchange

The key management portion of IPSec involves the determination and distribution of secret keys. The IPSec Architecture document mandates support for two types of key management:

• Manual: A system administrator manually configures each system with its own keys and with the keys of other communicating systems. This is practical for small, relatively static environments.

• Automated: An automated system enables the on-demand creation of keys for SAs and facilitates the use of keys in a large distributed system with an evolving configuration.

The default automated key management protocol for IPSec is referred to as ISAKMP/Oakley and consists of the following elements:

• Oakley Key Determination Protocol: Oakley is a key exchange protocol based on the Diffie-Hellman algorithm but providing added security. Oakley is generic in that it does not dictate specific formats.

• Internet Security Association and Key Management Protocol (ISAKMP): ISAKMP provides a framework for Internet key management and provides the specific protocol support, including formats, for negotiation of security attributes.

Oakley Key Determination Protocol Oakley is a refinement of the Diffie-Hellman key exchange algorithm. The Diffie-Hellman algorithm has two attractive features:

• Secret keys are created only when needed. There is no need to store secret keys for a long period of time, exposing them to increased vulnerability.

• The exchange requires no pre-existing infrastructure other than an agreement on the global parameters. However, Diffie-Hellman has got some weaknesses:

• No identity information about the parties is provided.
• It is possible for a man-in-the-middle attack
• It is computationally intensive. As a result, it is vulnerable to a clogging attack, in which an opponent requests a high number of keys.

Oakley is designed to retain the advantages of Diffie-Hellman while countering its weaknesses.

Features of Oakley The Oakley algorithm is characterized by five important features:
1. It employs a mechanism known as cookies to thwart clogging attacks.
2. It enables the two parties to negotiate a group; this, in essence, specifies the global parameters of the Diffie-Hellman key exchange.
3. It uses nonces to ensure against replay attacks.
4. It enables the exchange of Diffie-Hellman public key values.
5. It authenticates the Diffie-Hellman exchange to thwart man-in-the-middle attacks.

Aggressive Oakley Key Exchange Aggressive key exchange is a technique used for exchanging the message keys and is so called because only three messages are allowed to be exchanged at any time.

Key Determination Protocol

IKE key determination is a refinement of the Diffie-Hellman key exchange algorithm. Recall that Diffie-Hellman involves the following interaction between users A and B. There is prior agreement on two global parameters: q, a large prime number; and α, a primitive root of q. A selects a random integer X_A as its private key and transmits to B its public key $Y_A = \alpha^{X_A} \bmod q$. Similarly, B selects a random integer X_B as its private key and transmits to A its public key $Y_B = \alpha^{X_B} \bmod q$. Each side can now compute the secret session key:

$$K = (Y_B)^{X_A} \bmod q = (Y_A)^{X_B} \bmod q = \alpha^{X_A X_B} \bmod q$$

The Diffie-Hellman algorithm has two attractive features:

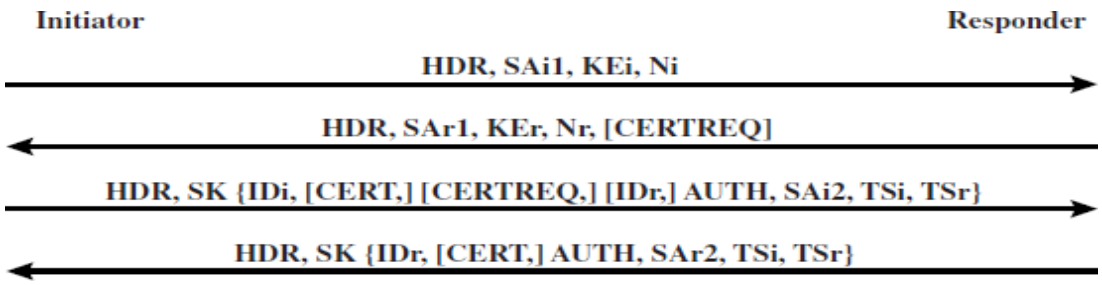
- Secret keys are created only when needed. There is no need to store secret keys for a long period of time, exposing them to increased vulnerability.
- The exchange requires no pre-existing infrastructure other than an agreement on the global parameters.

However, there are a number of weaknesses to Diffie-Hellman, as pointed out in

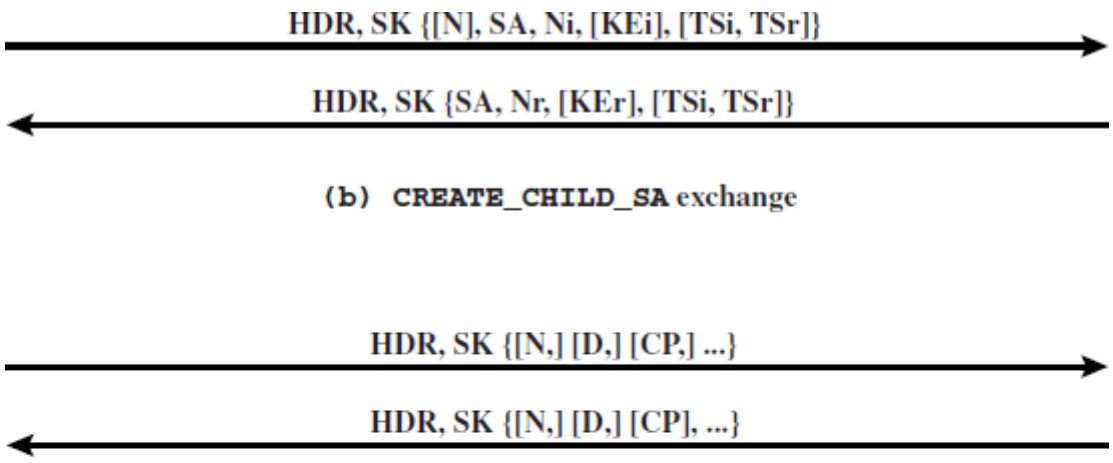
IKE key determination employs **nonces** to ensure against replay attacks. Each nonce is a locally generated pseudorandom number. Nonces appear in responses and are encrypted during certain portions of the exchange to secure their use.

Three different **authentication** methods can be used with IKE key determination:

- **Digital signatures:** The exchange is authenticated by signing a mutually obtainable hash; each party encrypts the hash with its private key. The hash is generated over important parameters, such as user IDs and nonces.
- **Public-key encryption:** The exchange is authenticated by encrypting parameters such as IDs and nonces with the sender's private key.
- **Symmetric-key encryption:** A key derived by some out-of-band mechanism can be used to authenticate the exchange by symmetric encryption of exchange parameters.



(a) Initial exchanges



HDR = IKE header
SAx1 = offered and chosen algorithms, DH group
KEx = Diffie-Hellman public key
Nx= nonces
CERTREQ = Certificate request
IDx = identity
CERT = certificate

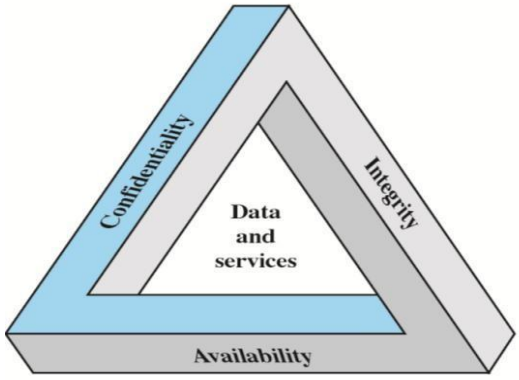
SK {...} = MAC and encrypt
AUTH = Authentication
SAx2 = algorithms, parameters for IPsec SA
TSx = traffic selectors for IPsec SA
N = Notify
D = Delete
CP = Configuration

Figure 19.11 IKEv2 Exchanges

(Part - 3)

Description of the system

Computer Security: The protection afforded to an automated information system in order to attain the applicable objectives of preserving the **integrity**, **availability** and **confidentiality** of information system resources (i.e. hardware, software, firmware, information/data, and telecommunications)



•*Integrity* - Assets can be modified by authorized parties only

•*Availability* - Assets be available to authorized parties

•*Confidentiality* - Requires information in a computer system only be accessible by authorized parties. Individuals set their own privacy requirements.

Addl. requirements:

•*Authenticity* - Requires that a computer system be able to verify the identity of a user

•*Accountability* - Requires the detection and tracing of a security breach to a responsible party.

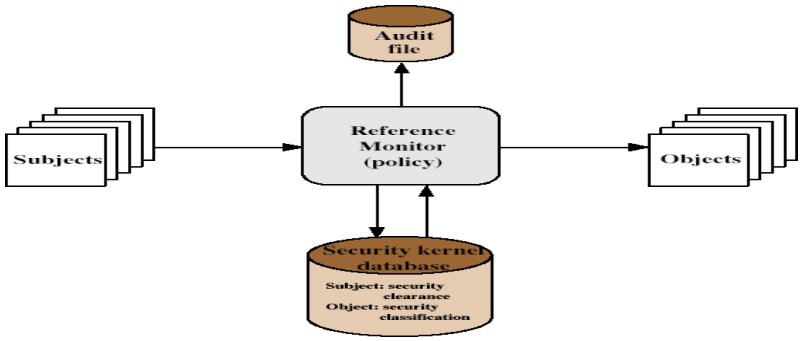
Security is a concern of organizations with assets that are controlled by computer systems. By accessing or altering data, an attacker can steal tangible assets or lead an organization to take actions it would not otherwise take. By merely examining data, an attacker can gain a competitive advantage, without the owner of the data being any the wiser.

users

Trust and Trusted Systems :

information security is increasingly important have varying degrees of sensitivity of information cf military info classifications: confidential, secret etc subjects (people or programs) have varying rights of access to objects (information)want to consider ways of increasing confidence in systems to enforce these rights known as multilevel security subjects have maximum & current security level objects have a fixed security level classification one of the most famous security models implemented as mandatory policies on system has two key policies:

no read up (simple security property) a subject can only read/write an object if the current security level of the subject dominates (>=) the classification of the object no write down (*-property) a subject can only append/write to an object if the current security level of the subject is dominated by (<=) the classification of the object



- governments can evaluate IT systems
- against a range of standards:
  - TCSEC, IPSEC and now Common Criteria
- define a number of –levels| of evaluation with increasingly stringent checking
- have published lists of evaluated products
  - though aimed at government/defense use
  - can be useful in industry also

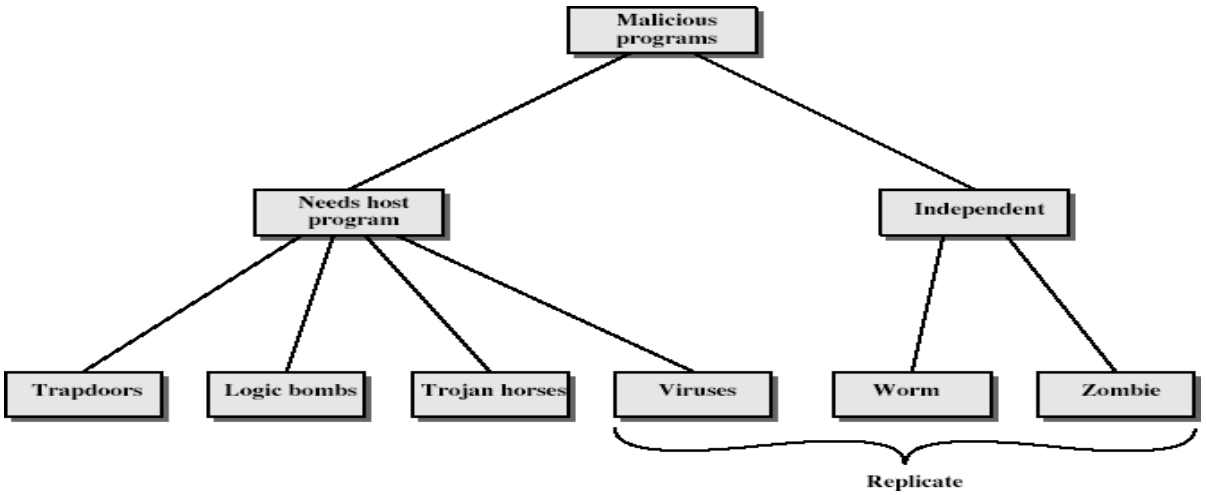
Buffer Overflow and Malicious Software :

- computer viruses have got a lot of publicity
- one of a family of malicious software
- effects usually obvious
- have figured in news reports, fiction, movies (often exaggerated)
- getting more attention than deserve
- are a concern though

Trapdoors: secret entry point into a program

- allows those who know access bypassing usual security procedures
- have been commonly used by developers
- a threat when left in production programs allowing exploited by attackers
- very hard to block in O/S
- requires good s/w development & update

Logic Bomb



Logic Bomb

- one of oldest types of malicious software
- code embedded in legitimate program
- activated when specified conditions met
  - eg presence/absence of some file
  - particular date/time
  - particular user
- when triggered typically damage system
  - modify/delete files/disks

Trojan Horse

- program with hidden side-effects
- which is usually superficially attractive
  - eg game, s/w upgrade etc
- when run performs some additional tasks
  - allows attacker to indirectly gain access they do not have directly
- often used to propagate a virus/worm or install a backdoor
- or simply to destroy data

Zombie

- program which secretly takes over another networked computer
- then uses it to indirectly launch attacks
- often used to launch distributed denial of service (DDoS) attacks
- exploits known flaws in network systems

Viruses

- a piece of self-replicating code attached to some other code
  - cf biological virus
- both propagates itself & carries a payload
  - carries code to make copies of itself
  - as well as code to perform some covert task

Types of Viruses

- can classify on basis of how they attack parasitic virus
- memory-resident virus
- boot sector virus
- stealth
- polymorphic virus

- macro virus

Macro Virus

- **macro code** attached to some **data file**
- interpreted by program using file
  - eg Word/Excel macros
  - esp. using auto command & command macros
- code is now platform independent
- is a major source of new viral infections
- blurs distinction between data and program files making task of detection much harder
- classic trade-off: "ease of use" vs "security"

Email Virus

- spread using email with attachment containing a macro virus
  - cf Melissa
- triggered when user opens attachment
- or worse even when mail viewed by using scripting features in mail agent
- usually targeted at Microsoft Outlook mail agent & Word/Excel documents

Worms

- replicating but not infecting program
- typically spreads over a network
  - cf Morris Internet Worm in 1988
  - led to creation of CERTs
- using users distributed privileges or by exploiting system vulnerabilities
- widely used by hackers to create **zombie PC's**, subsequently used for further attacks, esp DoS
- major issue is lack of security of permanently connected systems, esp PC's

Intrusion Detection System(IDS)

Intruders:

- significant issue for networked systems is hostile or unwanted access
- either via network or local
- can identify classes of intruders:
  - masquerader
  - misfeasor
  - clandestine user
- varying levels of competence
- clearly a growing publicized problem
  - from –Wily Hacker in 1986/87
  - to clearly escalating CERT stats
- may seem benign, but still cost resources
- may use compromised system to launch other attacks
- awareness of intruders has led to the development of CERTs

Intrusion Techniques:

- aim to gain access and/or increase privileges on a system
- basic attack methodology
  - target acquisition and information gathering
  - initial access
  - privilege escalation
  - covering tracks
- key goal often is to acquire passwords
- so then exercise access rights of owner

Password Capture

- another attack involves **password capture**
  - watching over shoulder as password is entered
  - using a trojan horse program to collect
  - monitoring an insecure network login
    - eg. telnet, FTP, web, email
  - extracting recorded info after successful login (web history/cache, last number dialed etc)
- using valid login/password can impersonate user
- users need to be educated to use suitable precautions/countermeasures

Intrusion Detection:

- inevitably will have security failures

- so need also to detect intrusions so can
  - block if detected quickly
  - act as deterrent
  - collect info to improve security
- assume intruder will behave differently to a legitimate user
  - but will have imperfect distinction between

Approaches to Intrusion Detection

- statistical anomaly detection
  - threshold
  - profile based
- rule-based detection
  - anomaly
  - penetration identification

Audit Records :

- fundamental tool for intrusion detection
- native audit records
  - part of all common multi-user O/S
  - already present for use
  - may not have info wanted in desired form
- detection-specific audit records
  - created specifically to collect wanted info
  - at cost of additional overhead on system

Statistical Anomaly Detection

- threshold detection
  - count occurrences of specific event over time
  - if exceed reasonable value assume intrusion
  - alone is a crude & ineffective detector
- profile based
  - characterize past behavior of users
  - detect significant deviations from this
  - profile usually multi-parameter

Audit Record Analysis

- foundation of statistical approaches
- analyze records to get metrics over time
  - counter, gauge, interval timer, resource use
- use various tests on these to determine if current behavior is acceptable
  - mean & standard deviation, multivariate, markov process, time series, operational
- key advantage is no prior knowledge used

Rule-Based Intrusion Detection

- observe events on system & apply rules to decide if activity is suspicious or not
- rule-based anomaly detection
  - analyze historical audit records to identify usage patterns & auto-generate rules for them
  - then observe current behavior & match against rules to see if conforms
  - like statistical anomaly detection does not require prior knowledge of security flaws
- rule-based penetration identification
  - uses expert systems technology
  - with rules identifying known penetration, weakness patterns, or suspicious behavior
  - compare audit records or states against rules
  - rules usually machine & O/S specific
  - rules are generated by experts who interview & codify knowledge of security admins
  - quality depends on how well this is done

Honeypots:

- decoy systems to lure attackers
  - away from accessing critical systems
  - to collect information of their activities
  - to encourage attacker to stay on system so administrator can respond
- are filled with fabricated information
- instrumented to collect detailed information on attackers activities
- single or multiple networked systems
- cf IETF Intrusion Detection WG standards

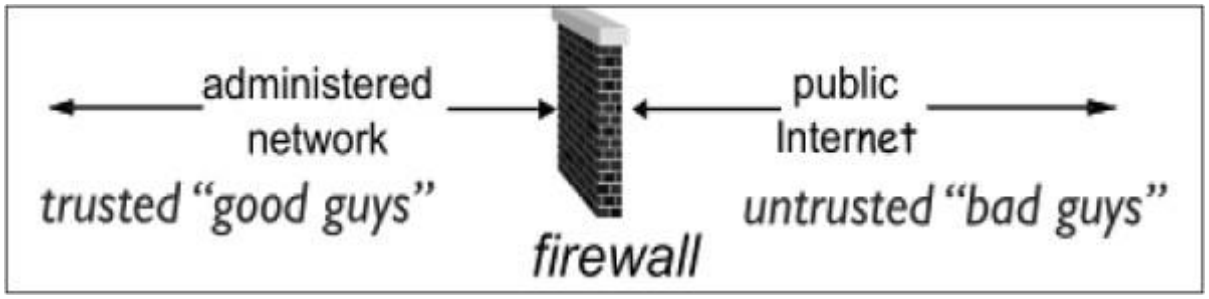
Firewalls

Almost every medium and large-scale organization has a presence on the Internet and has an organizational network connected to it. Network partitioning at the boundary between the outside Internet and the internal network is essential for network security. Sometimes the inside network (intranet) is referred to as the ―trusted‖ side and the external Internet as the ―un-trusted‖ side.

Types of Firewall

Firewall is a network device that isolates organization‘s internal network from larger outside network/Internet. It can be a hardware, software, or combined system that prevents unauthorized access to or from internal network.

All data packets entering or leaving the internal network pass through the firewall, which examines each packet and blocks those that do not meet the specified security criteria.



Deploying firewall at network boundary is like aggregating the security at a single point. It is analogous to locking an apartment at the entrance and not necessarily at each door.

Firewall is considered as an essential element to achieve network security for the following reasons −

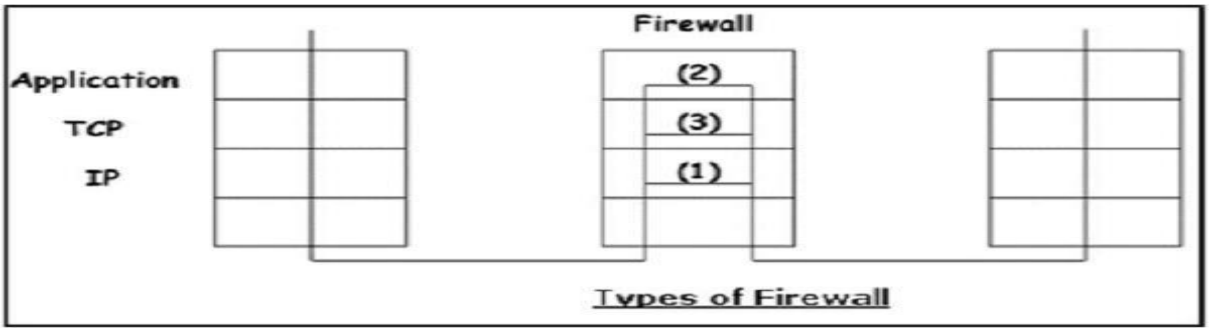
- Internal network and hosts are unlikely to be properly secured.

- Internet is a dangerous place with criminals, users from competing companies, disgruntled ex-employees, spies from unfriendly countries, vandals, etc.
- To prevent an attacker from launching denial of service attacks on network resource.
- To prevent illegal modification/access to internal data by an outsider attacker.

Firewall is categorized into three basic types −

- Packet filter (Stateless & Stateful)
- Application-level gateway
- Circuit-level gateway

These three categories, however, are not mutually exclusive. Modern firewalls have a mix of abilities that may place them in more than one of the three categories.



Stateless & Stateful Packet Filtering Firewall

In this type of firewall deployment, the internal network is connected to the external network/Internet via a router firewall. The firewall inspects and filters data packet-by-packet.

Packet-filtering firewalls allow or block the packets mostly based on criteria such as source and/or destination IP addresses, protocol, source and/or destination port numbers, and various other parameters within the IP header.

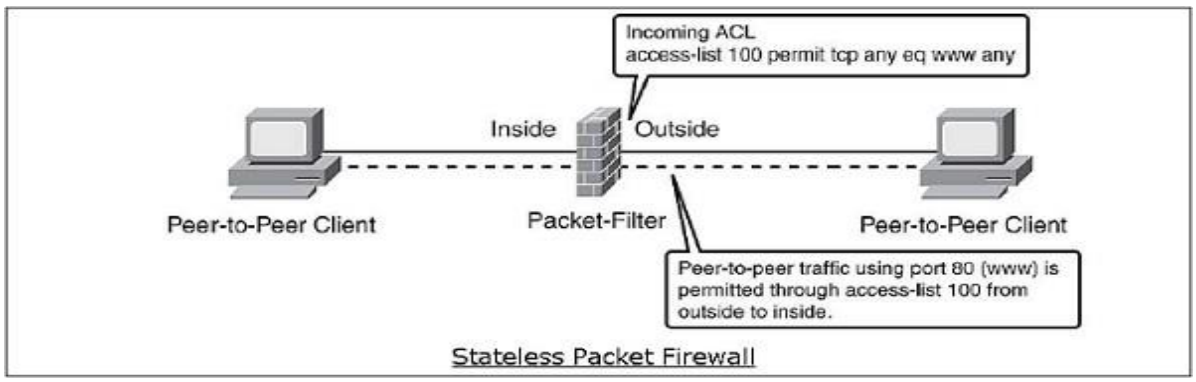
The decision can be based on factors other than IP header fields such as ICMP message type, TCP SYN and ACK bits, etc.

Packet filter rule has two parts −

- **Selection criteria** − It is a used as a condition and pattern matching for decision making.
- **Action field** − This part specifies action to be taken if an IP packet meets the selection criteria. The action could be either block (deny) or permit (allow) the packet across the firewall.

Packet filtering is generally accomplished by configuring Access Control Lists (ACL) on routers or switches. ACL is a table of packet filter rules.

As traffic enters or exits an interface, firewall applies ACLs from top to bottom to each incoming packet, finds matching criteria and either permits or denies the individual packets.



Stateless firewall is a kind of a rigid tool. It looks at packet and allows it if its meets the criteria even if it is not part of any established ongoing communication.

Hence, such firewalls are replaced by **stateful firewalls** in modern networks. This type of firewalls offer a more in-depth inspection method over the only ACL based packet inspection methods of stateless firewalls.

Stateful firewall monitors the connection setup and teardown process to keep a check on connections at the TCP/IP level. This allows them to keep track of connections state and determine which hosts have open, authorized connections at any given point in time.

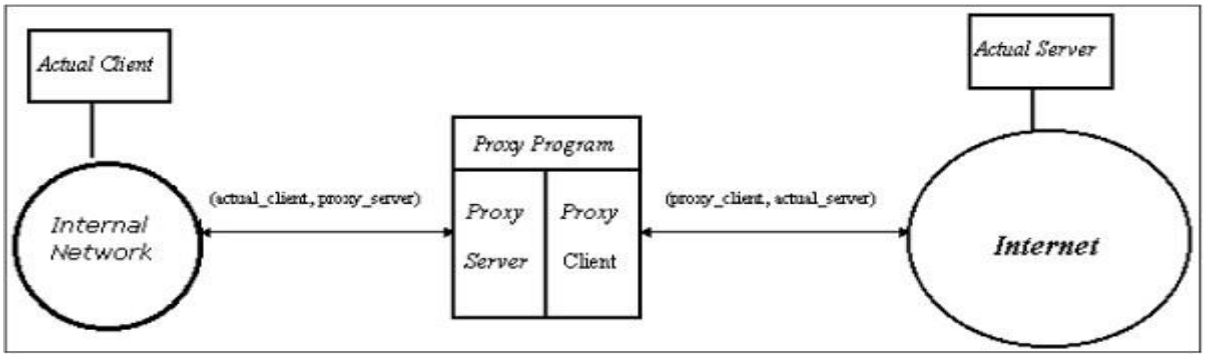
They reference the rule base only when a new connection is requested. Packets belonging to existing connections are compared to the firewall's state table of open connections, and decision to allow or block is taken. This process saves time and provides added security as well. No packet is allowed to trespass the firewall unless it belongs to already established connection. It can timeout inactive connections at firewall after which it no longer admit packets for that connection.

Application Gateways

An application-level gateway acts as a relay node for the application-level traffic. They intercept incoming and outgoing packets, run proxies that copy and forward information across the gateway, and function as a ***proxy server***, preventing any direct connection between a trusted server or client and an untrusted host.

The proxies are application specific. They can filter packets at the application layer of the OSI model.

Application-specific Proxies



An application-specific proxy accepts packets generated by only specified application for which they are designed to copy, forward, and filter. For example, only a Telnet proxy can copy, forward, and filter Telnet traffic.

If a network relies only on an application-level gateway, incoming and outgoing packets cannot access services that have no proxies configured. For example, if a gateway runs FTP and Telnet proxies, only packets generated by these services can pass through the firewall. All other services are blocked.

Application-level Filtering

An application-level proxy gateway, examines and filters individual packets, rather than simply copying them and blindly forwarding them across the gateway. Application-specific proxies check each packet that passes through the gateway, verifying the contents of the packet up through the application layer. These proxies can filter particular kinds of commands or information in the application protocols.

Application gateways can restrict specific actions from being performed. For example, the gateway could be configured to prevent users from performing the _FTP put' command. This can prevent modification of the information stored on the server by an attacker.

Transparent

Although application-level gateways can be transparent, many implementations require user authentication before users can access an untrusted network, a process that reduces true transparency. Authentication may be different if the user is from the internal network or from the Internet. For an internal network, a simple list of IP addresses can be allowed to connect to external applications. But from the Internet side a strong authentication should be implemented.

An application gateway actually relays TCP segments between the two TCP connections in the two directions (Client ↔ Proxy ↔ Server).

For outbound packets, the gateway may replace the source IP address by its own IP address. The process is referred to as Network Address Translation (NAT). It ensures that internal IP addresses are not exposed to the Internet.