UNIT V - SECURITY PRACTICE AND SYSTEM SECURITY

Electronic Mail security – PGP, S/MIME – IP security – Web Security - SYSTEM SECURITY: Intruders – Malicious software – viruses – Firewalls.

Electronic Mail security

5.1.1 PRETTY GOOD PRIVACY (PGP)

PGP provides the confidentiality and authentication service that can be used for electronic mail and file storage applications.

The steps involved in PGP are

□ Select the best available cryptographic algorithms as building blocks.

□ Integrate these algorithms into a general purpose application that is independent of operating system and processor and that is based on a small set of easy-touse commands.

□ Make the package and its documentation, including the source code, freely available via the internet, bulletin boards and commercial networks.

□ Enter into an agreement with a company to provide a fully compatible, low cost commercial version of PGP.

PGP has grown explosively and is now widely used.

A number of reasons can be cited for this growth.

□ It isavailable free worldwide in versions that run on a variety of platform.

□ It is based on algorithms that have survived extensive public review and are considered extremely secure.e.g., RSA, DSS and Diffie Hellman for public key encryption

□ It has a wide range of applicability.

□ It was not developed by, nor it is controlled by, any governmental or standardsorganization.

5.1.1.1. Operational description

The actual operation of PGP consists of five services:

1. Authentication
2. Confidentiality
3. Compression
4. E-mail compatibility
5. Segmentation.

1. Authentication: The sequence for authentication is as follows:

□ The sender creates the message

□ SHA-1 is used to generate a 160-bit hash code of the message

□ The hash code is encrypted with RSA using the sender‟s private key and the result is pretended to the message.

- The receiver uses RSA with the sender‟s public key to decrypt and recover the hash code.
- The receiver generates a new hash code for the message and compares it with the decrypted hash code. If the two match, the message is accepted as authentic.

2. Confidentiality

Confidentiality is provided by encrypting messages to be transmitted or to be stored locally as files. In both cases, the conventional encryption algorithm CAST-128 may be used.

The 64-bit cipher feedback (CFB) mode is used. In PGP, each conventional key is used only once. That is, a new key is generated as a random 128-bit number for each message. Thus although this is referred to as **a session key**, it is in reality a **onetime key**. To protect the key, it is encrypted with the receiver‟s public key.

The sequence for confidentiality is as follows:

- The sender generates a message and a random 128-bit number to be used as a session key for this message only.
- The message is encrypted using CAST-128 with the session key.
- The session key is encrypted with RSA, using the receiver‟s public key and is prepended to the message.
- The receiver uses RSA with its private key to decrypt and recover the session key.
- The session key is used to decrypt the message.

Confidentiality and authentication

Here both services may be used for the same message. First, a signature is generated for the plaintext message and prepended to the message. Then the plaintext plus the signature is encrypted using CAST-128 and the session key is encrypted using RSA.

3. Compression

As a default, PGP compresses the message after applying the signature but before encryption. This has the benefit of saving space for both e-mail transmission and for file storage.

The signature is generated before compression for two reasons:

- It is preferable to sign an uncompressed message so that one can store only the uncompressed message together with the signature for future verification. If one signed a compressed document, then it would be necessary either to store a compressed version of the message for later verification or to recompress the message when verification is required.
- Even if one were willing to generate dynamically a recompressed message from verification, PGP‟s compression algorithm presents a difficulty. The algorithm is not deterministic; various implementations of the algorithm achieve different tradeoffs in running speed versus compression ratio and as a result, produce different compression forms.

Message encryption is applied after compression to strengthen cryptographic security. Because the compressed message has less redundancy than the original plaintext, cryptanalysis is more difficult. The compression algorithm used is ZIP

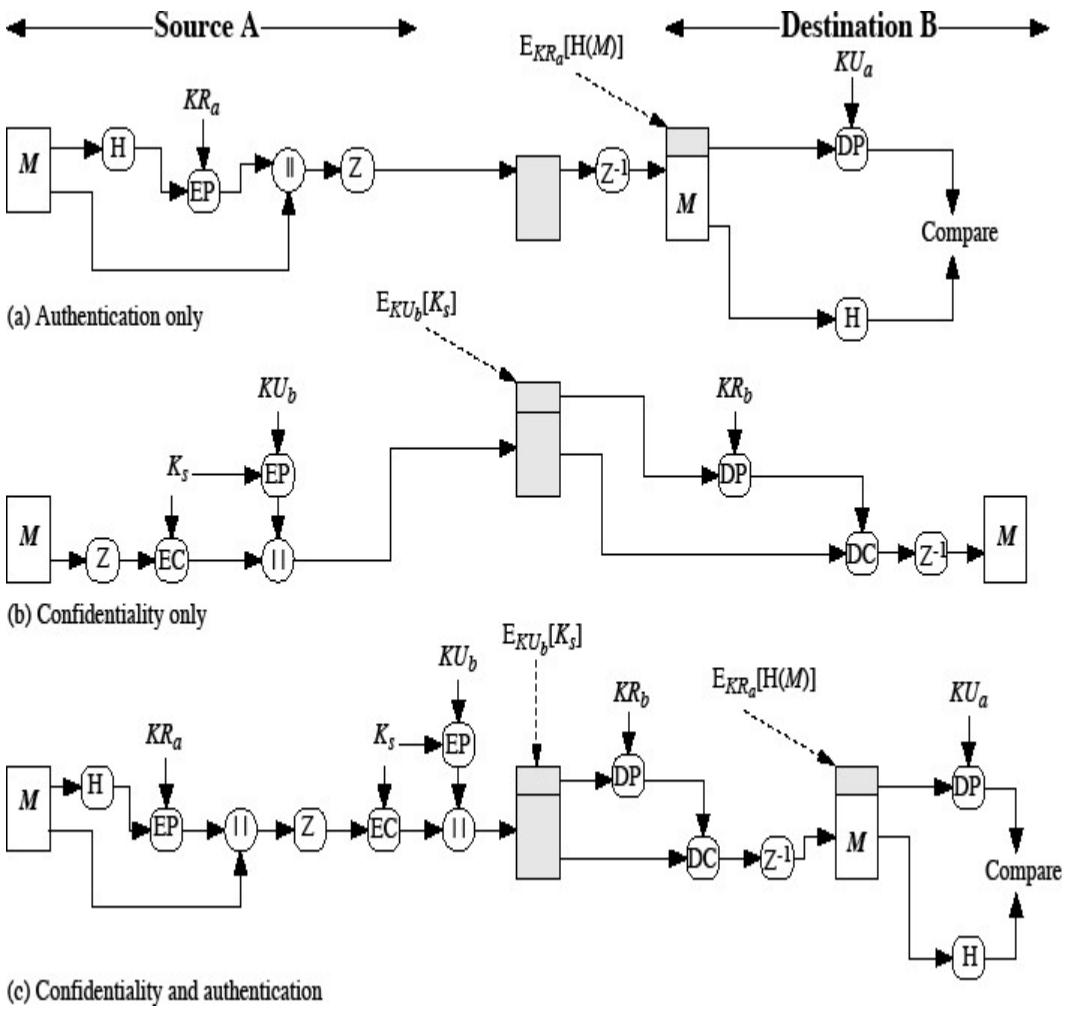


Fig 5.1: PGP Cryptographic Functions

4. E-mail compatibility

Many electronic mail systems only permit the use of blocks consisting of ASCII texts. To accommodate this restriction, PGP provides the service of converting the raw 8-bit binarystream to a stream of printable ASCII characters. The scheme used for this purpose is **radix-64 conversion**. Each group of three octets of binary data is mapped into four ASCII characters.

5. Segmentation and reassembly

E-mail facilities often are restricted to a maximum length. E.g., many of the facilities accessible through the internet impose a maximum length of 50,000 octets. Any message longer than that must be broken up into smaller segments, each of which is mailed separately.

To accommodate this restriction, PGP automatically subdivides a message that is too large into segments that are small enough to send via e-mail. The segmentation is done after all the other processing, including the radix-64 conversion.

At the receiving end, PGP must strip off all e-mail headers and reassemble the entire original block before performing the other steps.

5.1.1.2. Cryptographic keys and key rings

Three separate requirements can be identified with respect to these keys:

- A means of generating unpredictable session keys is needed.
- It must allow a user to have multiple public key/private key pairs.
- Each PGP entity must maintain a file of its own public/private key pairs as well asa file of public keys of correspondents.

a. Session key generation

Each session key is associated with a single message and is used only for the purposeof encryption and decryption of that message. Random 128-bit numbers are generated using CAST-128 itself.

The input to the random number generator consists of a 128-bit key and two 64-bit blocks that are treated as plaintext to be encrypted. Using cipher feedback mode, the CAST-128 produces two 64-bit cipher text blocks, which are concatenated to form the 128-bit session key. The plaintext input to CAST-128 is itself derived from a stream of 128-bit randomized numbers. These numbers are based on the keystroke input from the user.

b. Key identifiers

If multiple public/private key pair are used, then how does the recipient know which of the public keys was used to encrypt the session key?

One simple solution would be to transmit the public key with the message but, it is unnecessary wasteful of space. Another solution would be to associate an identifier with each public key that is unique at least within each user.

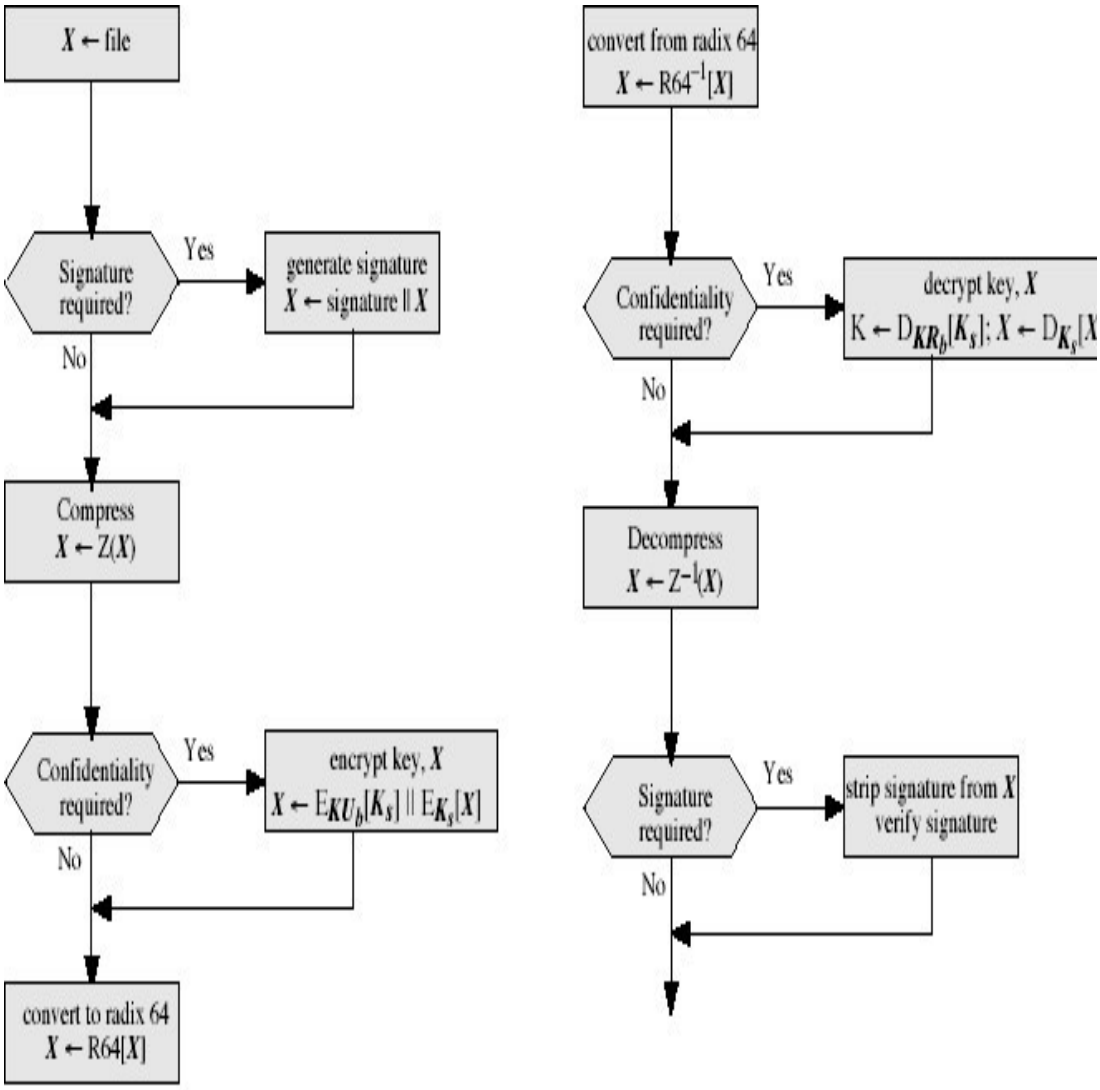
The solution adopted by PGP is to assign a key ID to each public key that is, with very high probability, unique within a user ID. The key ID associated with each public key consists of its least significant 64 bits. i.e., the key ID of public key KU_a is ($KU_a \bmod 2^{64}$).

A message consists of three components.

- **Message component —** includes actual data to be transmitted, as well as the filename and a timestamp that specifies the time of creation
- **Session key component** — includes session key and the identifier of the recipient public key.
- **Signature component** – includes the following
  - **Timestamp** – time at which the signature was made.
  - **Message digest** – hash code.
  - **Two octets of message digest —** to enable the recipient to determine if the correct public key was used to decrypt the message.
  - **Key ID of sender's public key** – identifies the public key

Notation :

- **EkU_b**= encryption with user B"s Public key
- **EKR_a**= encryption with user A"s private key
- **EK_s** = encryption with session key
- **ZIP** = Zip compression function
- **R64** = Radix- 64 conversion function



(a) Generic Transmission Diagram (from A)

(b) Generic Reception Diagram (to B)

Fig 5.2: Transmission and Reception of PGP message

PGP provides a pair of data structures at each node, one to store the public/private key pair owned by that node and one to store the public keys of the other users known at that node. These data structures are referred to as private key ring and public key ring.

The general structures of the private and public key rings are shown below:

Timestamp - the date/time when this entry was made. **Key ID** - the least significant bits of the public key.
Public key - public key portion of the pair.
Private Key - private key portion of the pair.

User ID - the owner of the key

Key legitimacy field — indicates the extent to which PGP will trust that this is a valid public key for this user.

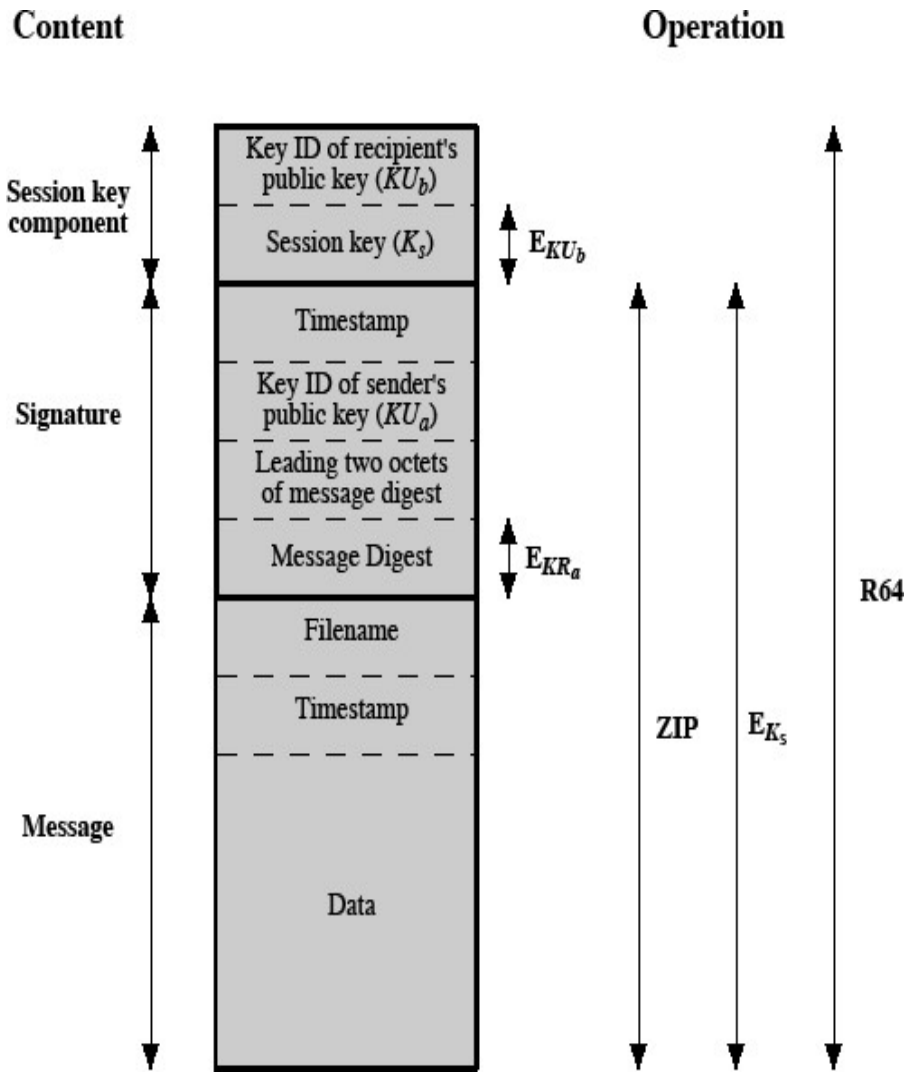


Fig 5.3: General Format of PGP message (From A to B)

Signature trust field – indicates the degree to which this PGP user trusts the signer to certify public key.

Owner trust field - indicates the degree to which this public key is trusted to sign other public key certificates.

PGP message generation First consider message transmission and assume that the message is to be both signed and encrypted. The sending PGP entity performs the following steps:

1. Signing the message

- PGP retrieves the sender‟s private key from the private key ring using user ID as an index. If user ID was not provided, the first private key from the ring is retrieved.
- PGP prompts the user for the passphrase (password) to recover the unencrypted private key.
- The signature component of the message is constructed.

Private Key Ring

Timestamp	Key ID*	Public Key	Encrypted Private Key	User ID*
• • •	• • •	• • •	• • •	• • •
T_i	PU_i mod 2^{64}	PU_i	E(H(P_i), PR_i)	User i
• • •	• • •	• • •	• • •	• • •

Public Key Ring

Timestamp	Key ID*	Public Key	Owner Trust	User ID*	Key Legitimacy	Signature(s)	Signature Trust(s)
• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •
T_i	PU_i mod 2^{64}	PU_i	$trust\_flag_i$	User i	$trust\_flag_i$		
• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •

* = field used to index table

Fig 5.4: General structure of private and public key Rings

2. Encrypting the message

- PGP generates a session key and encrypts the message.
- PGP retrieves the recipient‟s public key from the public key ring using user ID as index.

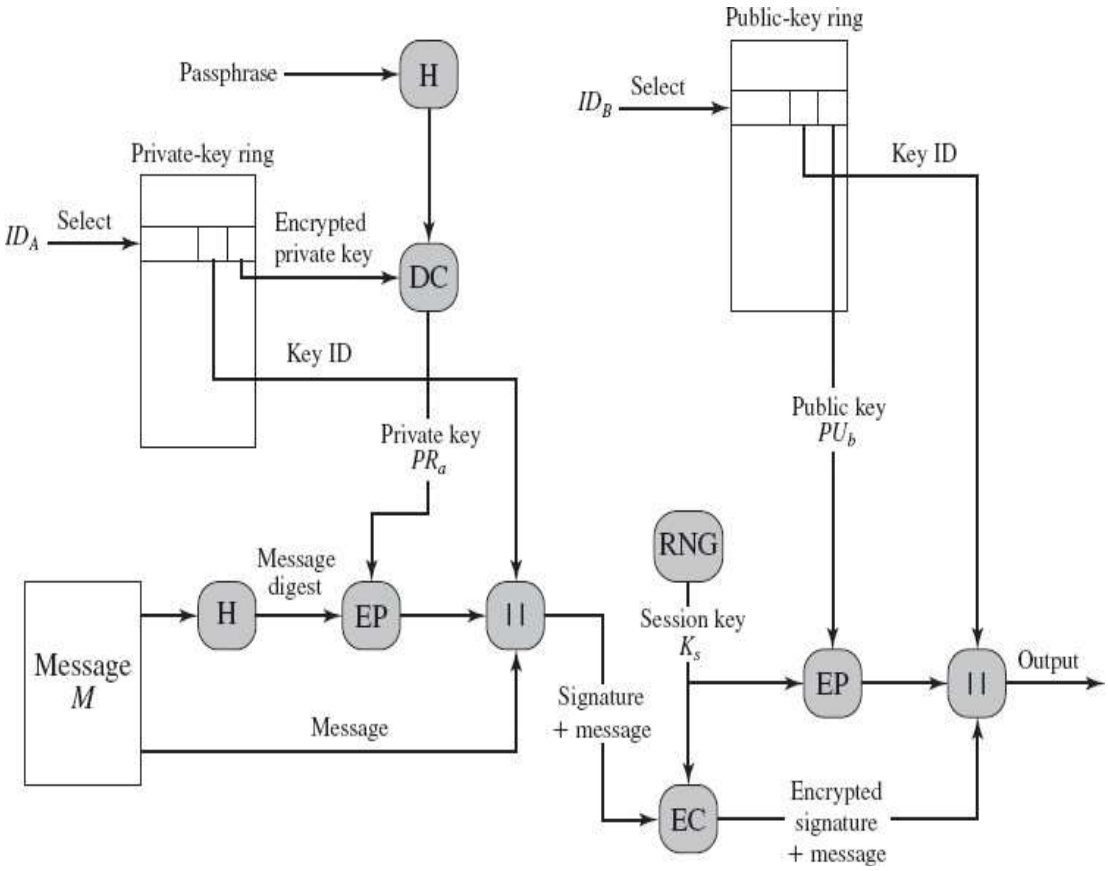


Fig 5.5: PGP message generation

The receiving PGP entity performs the following steps:

1. Decrypting the message

- PGP retrieves the receiver‟s private key from the private key ring, using the key ID fieldin the session key component of the message as an index.
- PGP prompts the user for the passphrase (password) to recover the unencrypted private key.
- PGP then recovers the session key and decrypts the message.

2. Authenticating the message

- PGP retrieves the sender‟s public key from the public key ring, using the key ID field inthe signature key component of the message as an index.
- PGP recovers the transmitted message digest.
- PGP computes the message digest for the received message and compares it to the transmitted message digest to authenticate.

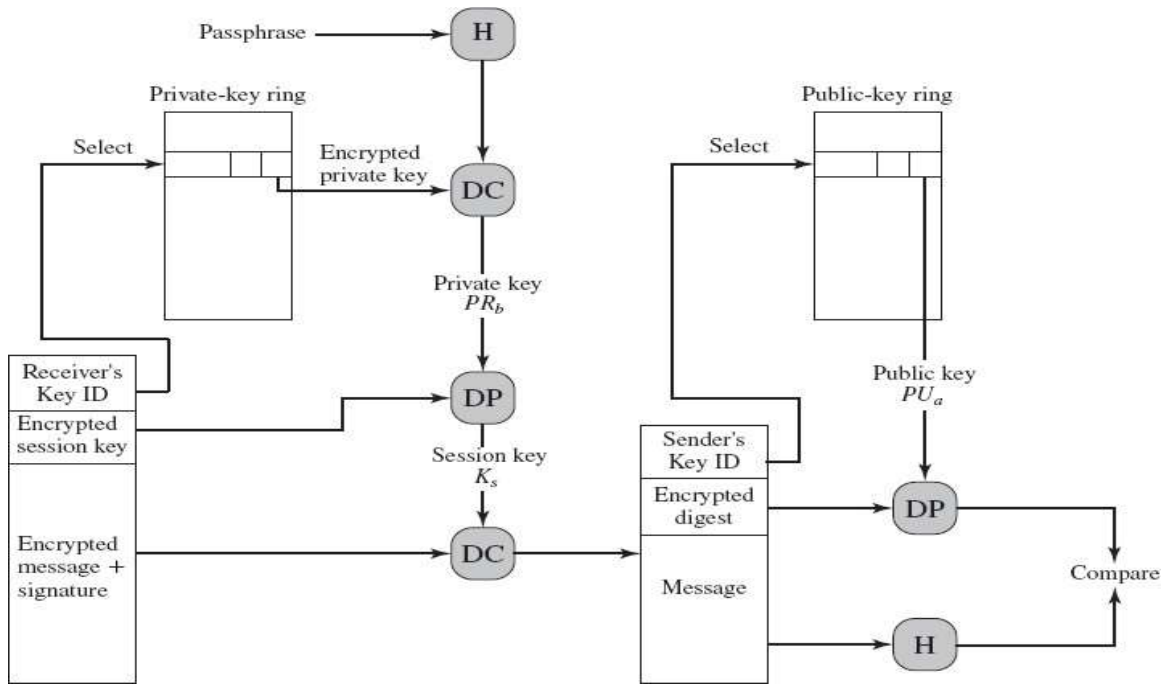


Fig 5.6: PGP message reception

5.1.2. S/MIME

S/MIME (Secure/Multipurpose Internet Mail Extension) is a security enhancement to the MIME Internet e-mail format standard, based on technology from RSA Data Security.

5.1.2.1 Multipurpose Internet Mail Extensions

MIME is an extension to the RFC 822 framework that is intended to address some of the problems and limitations of the use of SMTP (Simple Mail Transfer Protocol) or some other mail transfer protocol and RFC 822 for electronic mail.

Following are the limitations of SMTP/822 scheme:

1. SMTP cannot transmit executable files or other binary objects.
2. SMTP cannot transmit text data that includes national language characters because these are represented by 8-bit codes with values of 128 decimal or higher, and SMTP is limited to 7-bit ASCII.
3. SMTP servers may reject mail message over a certain size.
4. SMTP gateways that translate between ASCII and the character code EBCDIC do not use a consistent set of mappings, resulting in translation problems.
5. SMTP gateways to X.400 electronic mail networks cannot handle non textual data included in X.400 messages.
6. Some SMTP implementations do not adhere completely to the SMTP standards defined in RFC 821. Common problems include:

- Deletion, addition, or reordering of carriage return and linefeed
- Truncating or wrapping lines longer than 76 characters
- Removal of trailing white space (tab and space characters)

- Padding of lines in a message to the same length

- Conversion of tab characters into multiple space characters

MIME is intended to resolve these problems in a manner that is compatible with existing RFC 822 implementations. The specification is provided in RFCs 2045 through 2049.

5.1.3 OVERVIEW

The MIME specification includes the following elements:

1. **Five new message header** fields are defined, which may be included in an RFC 822 header. These fields provide information about the body of the message.
2. **A number of content formats** are defined, thus standardizing representations thatsupport multimedia electronic mail.
3. **Transfer encodings** are defined that enable the conversion of any content format into a form that is protected from alteration by the mail system.

In this subsection, we introduce the five message header fields. The next two subsections deal with content formats and transfer encodings.

The five header fields defined in MIME are as follows:

- **MIME-Version**: Must have the parameter value 1.0. This field indicates that the message conforms to RFCs 2045 and 2046.
- **Content-Type**: Describes the data contained in the body with sufficient detail.
- **Content-Transfer-Encoding**: Indicates the type of transformation that has been used to represent the body of the message in a way that is acceptable for mail transport.
- **Content-ID:** Used to identify MIME entities uniquely in multiple contexts.
- **Content-Description**: A text description of the object with the body; this is useful when theobject is not readable (e.g., audio data).

5.1.3.1 MIME Content Types

There are seven different major types of content and a total of 15 subtypes

MIME Content Types		
Type	**Subtype**	**Description**
Text	Plain	Unformatted text; may be ASCII or ISO 8859.
	Enriched	Provides greater format flexibility.
Multipart	Mixed	The different parts are independent but are to be transmitted together. They should be presented to the receiver in the order that they appear in the mail message.
	Parallel	Differs from Mixed only in that no order is defined fordelivering the parts to the receiver.
	Alternative	The different parts are alternative versions of the same information. They are ordered in increasing faithfulness to the original, and the recipient's mail system should display the "best" version to the user.
	Digest	Similar to Mixed, but the default type/subtype of each part is message/rfc822.
Message	rfc822	The body is itself an encapsulated message that conformsto RFC 822.

	Partial	Used to allow fragmentation of large mail items, in a waythat is transparent to the recipient.
	External-body	Contains a pointer to an object that exists elsewhere.
Image	jpeg	The image is in JPEG format, JFIF encoding.
	Gif	The image is in GIF format.
Video	mpeg	MPEG format.
Audio	Basic	Single-channel 8-bit ISDN mu-law encoding at a sample rate of 8 kHz.
Application	PostScript	Adobe Postscript.
	octet-stream	General binary data consisting of 8-bit bytes.

For the **text type** of body, no special software is required to get the full meaning of the text, aside from support of the indicated character set. The primary subtype is plain text, which is simply a string of ASCII characters or ISO 8859 characters. The enriched subtype allows greater formatting flexibility.

The **multipart type** indicates that the body contains multiple, independent parts. The Content-Type header field includes a parameter, called boundary, that defines the delimiter between body parts.

The **multipart/digest subtype** is used when each of the body parts is interpreted as an RFC 822 message with headers. This subtype enables the construction of a message whose parts are individual messages. For example, the moderator of a group might collect e-mail messages from participants, bundle these messages, and send them out in one encapsulating MIME message.

The **message type** provides a number of important capabilities in MIME. The message/rfc822 subtype indicates that the body is an entire message, including header and body. Despite the name of this subtype, the encapsulated message may be not only a simple RFC 822 message, but also any MIME message.

The **message/partial subtype** enables fragmentation of a large message into a number of parts, which must be reassembled at the destination. For this subtype, three parameters are specified in the Content-Type: Message/Partial field: an id common to all fragments of the same message, a sequence number unique to each fragment, and the total number of fragments.

The **message/external-body subtype** indicates that the actual data to be conveyed in this message are not contained in the body. Instead, the body contains the information neededto access the data.

5.1.3.2. MIME Transfer Encodings

MIME Transfer Encodings

7bit	The data are all represented by short lines of ASCII characters.
8bit	The lines are short, but there may be non-ASCII characters (octets with the high-order bit set).
binary	Not only may non-ASCII characters be present but the lines are not necessarily short enough for SMTP transport.

quoted-printable	Encodes the data in such a way that if the data being encoded are mostly ASCII text, the encoded form of the data remains largely recognizable by humans.
base64	Encodes data by mapping 6-bit blocks of input to 8-bit blocks ofoutput, all of which are printable ASCII characters.
x-token	A named nonstandard encoding.

The quoted-printable transfer encoding is useful when the data consists largely of octets that correspond to printable ASCII characters. In essence, it represents non safe characters by the hexadecimal representation of their code and introduces reversible (soft) line breaks to limit message lines to 76 characters.

The base64 transfer encoding, also known as radix-64 encoding, is a common one for encoding arbitrary binary data in such a way as to be invulnerable to the processing by mail transport programs.

5.1.3.3. Canonical Form

An important concept in MIME and S/MIME is that of canonical form. Canonical form is a format, appropriate to the content type that is standardized for use between systems. This is in contrast to native form, which is a format that may be peculiar to a particular system.

5.1.3.4. S/MIME Functionality

In terms of general functionality, S/MIME is very similar to PGP. Both offer the ability to sign and/or encrypt messages.

Functions:

S/MIME provides the following functions:

- **Enveloped data:** This consists of encrypted content of any type and encrypted-content encryption keys for one or more recipients.
- **Signed data**: A digital signature is formed by taking the message digest of the content to be signed and then encrypting that with the private key of the signer. The content plus signature are then encoded using base 64 encoding. A signed data message can only be viewed by a recipient with S/MIME capability.
- **Clear-signed data:** As with signed data, a digital signature of the content is formed. However, in this case, only the digital signature is encoded using base 64. As a result, recipients without S/MIME capability can view the messagecontent, although they cannot verify the signature.
- **Signed and enveloped data:** Encrypted data may be signed and signed data or clear-signed data may be encrypted.

5.1.3.5. Cryptographic Algorithms

Table 1 summarizes the cryptographic algorithms used in S/MIME. S/MIME uses the following terminology, taken from RFC 2119 to specify the requirement level:

- **Must:** The definition is an absolute requirement of the specification. An implementation must include this feature or function to be in conformance with the specification.

- **Should:** There may exist valid reasons in particular circumstances to ignore this feature or function, but it is recommended that an implementation include the feature or function.

Table1: Cryptographic Algorithms Used in S/MIME	
Function	**Requirement**
Create a message digest to be used in forming a digital signature. Encrypt message digest to form digital signature.	MUST support SHA-1. Receiver SHOULD support MD5 forbackward compatibility. Sending and receiving agents MUST support DSS. Sending agents SHOULD support RSA encryption. Receiving agents SHOULD support verification of RSA signatures with key sizes 512 bits to 1024 bits.
Encrypt session key for transmission with message.	Sending and receiving agents SHOULD support Diffie-Hellman. Sending and receiving agents MUST support RSA encryption with key sizes512 bits to 1024 bits.
Encrypt message for transmission with one-time session key.	Sending and receiving agents MUST support encryption with triple DES Sending agents SHOULD support encryption with AES. Sending agents SHOULD support encryption with RC2/40.
Create a message authentication code	Receiving agents MUST support HMACwith SHA-1. Receiving agents SHOULD support HMAC with SHA-1.

5.1.3.6 S/MIME MESSAGES

S/MIME makes use of a number of new MIME content types, which are shown in Table 2. All of the new application types use the designation PKCS. This refers to a set of public-key cryptography specifications issued by RSA Laboratories and made available for the S/MIME effort.

Table 2: S/MIMEContent Types			
Type	**Subtype**	**SMIME Parameter**	**Description**
Multipart	Signed		A clear-signed message in two parts: one is the message and the other is the signature.
Application	PKCS 7-MIME	Signed Data	A signed S/MIME entity.
	PKCS 7-MIME	Enveloped Data	An encrypted S/MIME entity.
	PKCS 7-MIME	degenerate signed Data	An entity containing only public-key certificates.
	PKCS 7-MIME	Compressed Data	A compressed S/MIME entity
	PKCS 7-SIGNATURE	signed Data	The content type of the signature subpart of a multipart/signed message.

5.1.4 SECURING A MIME ENTITY

S/MIME secures a MIME entity with a signature, encryption, or both. A MIME entity may be an entire message (except for the RFC 822 headers), or if the MIME content type ismultipart, then a MIME entity is one or more of the subparts of the message. In all cases, the message to be sent is converted to canonical form.

In particular, for a given type and subtype, the appropriate canonical form is used for the message content. For a multipart message, the appropriate canonical form is used for each subpart.

The use of transfer encoding requires special attention.

1) Enveloped Data

1. Generate a pseudorandom session key for a particular symmetric encryption algorithm (RC2/40 or triple DES).
2. For each recipient, encrypt the session key with the recipient's public RSA key.
3. For each recipient, prepare a block known as Recipient Info that contains an identifier of the recipient's public-key certificate, an identifier of the algorithm used to encrypt the session key, and the encrypted session key.
4. Encrypt the message content with the session key.

The Recipient Info blocks followed by the encrypted content constitute the enveloped Data. This information is then encoded into base 64. To recover the encrypted message, the recipient first strips off the base64 encoding. Then the recipient's private key is used to recover the session key. Finally, the message content is decrypted with the session key.

2) Signed Data

The steps for preparing a signed Data MIME entity are as follows:

- Select a message digest algorithm (SHA or MD5).
- Compute the message digest, or hash function, of the content to be signed
- Encrypt the message digest with the signer's private key.
-
- 4. Prepare a block known as Signer Info that contains the signer's public-key certificate, an identifier of the message digest algorithm, an identifier of the algorithm used to encrypt the message digest, and the encrypted message digest

To recover the signed message and verify the signature, the recipient first strips off the base64 encoding. Then the signer's public key is used to decrypt the message digest.

The recipient independently computes the message digest and compares it to the decrypted message digest to verify the signature.

3) Clear Signing

- Clear signing is achieved using the multipart content type with a signed subtype.
- As was mentioned, this signing process does not involve transforming the message to be signed, so that the message is sent "in the clear."
- Thus, recipients with MIME capability but not S/MIME capability are able to read the incoming message.

A multipart/signed message has two parts.

The first part can be any MIME type but must be prepared so that it will not be altered during transfer from source to destination. This means that if the first part is not 7bit, then it needs to be encoded using base64 or quoted-printable.

This second part has a MIME content type of application and a subtype of PKCS7-signature The protocol parameter indicates that this is a two-part clear-signed entity. The receiver can verify the signature by taking the message digest of the first part and comparingthis to the message digest recovered from the signature in the second part.

5.1.4.1 Registration Request

The user will apply to a certification authority for a public-key certificate. The S/MIME entity is used to transfer a certification request.

- The certification request includes certification Request Info block, followed by an identifier of the public-key encryption algorithm, followed by the signature of the certification Request Info block, made using the sender's private key.
- The certification Request Info block includes a name of the certificate subject (the entity whose public key is to be certified) and a bit-string representation of the user's public key.

Certificates-Only Message

A message containing only certificates or a certificate revocation list (CRL) can be sent in response to a registration request. The message is an application/PKCS7-MIME type/subtype with an SMIME-type parameter of degenerate. The steps involved are the same as those for creating a signed Data message, except that there is no message content and the signer Info field is empty.

S/MIME Certificate Processing

S/MIME uses public-key certificates that conform to version 3 of X.509 The key-management scheme used by S/MIME is in some ways a hybrid between a strict X.509 certification hierarchy and PGP's web of trust.

User Agent Role

An S/MIME user has **several key-management functions** to perform:

Key generation:

The user of some related administrative utility (e.g., one associated with LAN management) MUST be capable of generating a key pair from a good source of nondeterministic random input and be protected in a secure fashion.

Registration:

A user's public key must be registered with a certification authority in order to receive an X.509 public-key certificate.

Certificate storage and retrieval:

A user requires access to a local list of certificates in order to verify incoming signatures and to encrypt outgoing messages. Such a list could be maintained by the user or by some local administrative entity on behalf of a number of users.

VeriSign Certificates

There are several companies that provide certification authority (CA) services. There are a number of Internet-based CAs, including VeriSign, GTE, and the U.S. Postal Service.

VeriSign provides a CA service that is intended to be compatible with S/MIME and a variety of other applications. VeriSign issues X.509 certificates with the product name VeriSign Digital ID.

The information contained in a Digital ID depends on the type of Digital ID and its use. At a minimum, each Digital ID contains

- Owner's public key
- Owner's name or alias
- Expiration date of the Digital ID
- Serial number of the Digital ID
- Name of the certification authority that issued the Digital ID
- Digital signature of the certification authority that issued the Digital ID

Digital IDs can also contain other user-supplied information, including

- Address
- E-mail address
- Basic registration information (country, zip code, age, and gender)

VeriSign provides three levels, or classes, of security for public-key certificates. A user requestsa certificate online at VeriSign's Web site or other participating Web sites. Class 1 and Class 2 requests are processed on line, and in most cases take only a few seconds to approve. Briefly, the following procedures are used:

- For Class 1 Digital IDs, VeriSign confirms the user's e-mail address by sending a PIN and Digital ID pick-up information to the e-mail address provided in the application.

- For Class 2 Digital IDs, VeriSign verifies the information in the application through an automated comparison with a consumer database in addition to performing all of the checking associated with a Class 1 Digital ID.

    - Finally, confirmation is sent to the specified postal address alerting the user thata Digital ID has been issued in his or her name.

- For Class 3 Digital IDs, VeriSign requires a higher level of identity assurance. An individual must prove his or her identity by providing notarized credentials or applying in

person.

5.1.4.2 Enhanced Security Services

Three enhanced security services have been proposed in an Internet draft.

Signed receipts:

A signed receipt may be requested in a Signed Data object. Returning a signed receipt provides proof of delivery to the originator of a message and allows the originator to demonstrate to a third party that the recipient received the message.

Security labels:

A security label may be included in the authenticated attributes of a SignedData object. A security label is a set of security information regarding the sensitivity of the content that is protected by S/MIME encapsulation. The labels may be used for access control, by indicating which users are permitted access to an object.

Secure mailing lists:

When a user sends a message to multiple recipients, a certain amount of per-recipient processing is required, including the use of each recipient's public key.

The user can be relieved of this work by employing the services of an S/MIME Mail List Agent (MLA). An MLA can take a single incoming message, perform the recipient-specific encryption for each recipient, and forward the message.

The originator of a message need only send the message to the MLA, with encryption performed using the MLA's public key.

5.1.5 NON REPUDIATION

Non-repudiation is the assurance that someone cannot deny something. Typically, non-repudiation refers to the ability to ensure that a party to a contract or a communication cannot deny the authenticity of their signature on a document or the sending of a message that they originated.

To repudiate means to deny. On the Internet, a digital signature is used not only to ensure that a message or document has been electronically signed by the person that purportedto sign the document, but also, since a digital signature can only be created by one person, to ensure that a person cannot later deny that they furnished the signature.

Since no security technology is absolutely fool-proof, some experts warn that a digital signature alone may not always guarantee non-repudiation. It is suggested that multiple approaches be used, such as capturing unique biometric information and other data about the sender or signer that collectively would be difficult to repudiate.

Email non-repudiation involves methods such as email tracking that is designed toensure that the sender cannot deny having sent a message and/or that the recipient cannotdeny having received it

5.2 IP SECURITY

5.2.1 OVERVIEW OF IPSEC

5.2.1.1 Applications of IPSec

IPSec provides the capability to secure communications across a LAN, across private and public WANs, and across the Internet. Examples of its use include the following:

- Secure branch office connectivity over the Internet
- 2 Secure remote access over the Internet
- Establishing extranet and intranet connectivity with partners
- Enhancing electronic commerce security

5.2.1.2 Benefits of IPSec:

- When IPSec is implemented in a firewall or router, it provides strong security
- IPSec in a firewall is resistant to bypass if all traffic from the outside must use IP, and the firewall is the only means of entrance from the Internet into the organization.
- IPSec is below the transport layer (TCP, UDP) and so is transparent to applications. There is no need to change software on a user or server system when IPSec is implemented in the firewall or router.
- IPSec can be transparent to end users. There is no need to train users onsecurity mechanisms
- IPSec can provide security for individual users if needed.

5.2.1.3 Routing Applications

IPSec can play a vital role in the routing architecture required for internet working.

The following are examples of the use of IPSec. IPSec can assure that

- A router advertisement (a new router advertises its presence) comes from an authorized router
- A neighbor advertisement (a router seeks to establish or maintain a neighbor relationship with a router in another routing domain) comes from an authorized router.
- A redirect message comes from the router to which the initial packet was sent.
- A routing update is not forged.

5.2.2 IP SECURITY ARCHITECTURE

5.2.2.1 IPSec Documents

The IPSec specification consists of numerous documents. The most important of these, issued in November of 1998, are RFCs 2401, 2402, 2406, and 2408:

- **RFC 2401:** An overview of a security architecture
- **RFC 2402:** Description of a packet authentication extension to IPv4 and IPv6
- **RFC 2406:** Description of a packet encryption extension to IPv4 and IPv6
- **RFC 2408:** Specification of key management capabilities

The documents are divided into seven groups:

5.2.2.2 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:

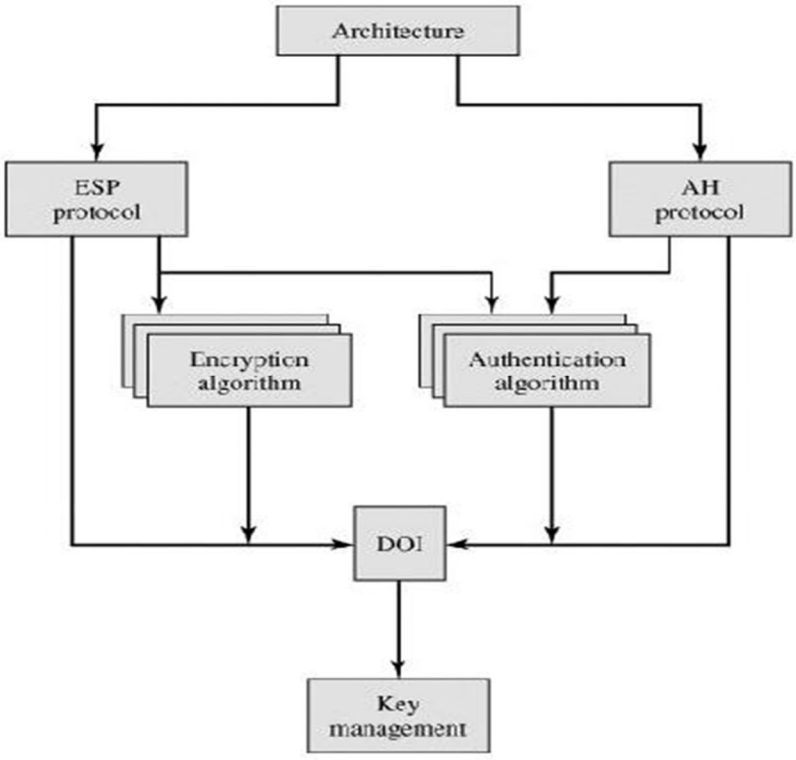
A set of documents that describe how various authentication algorithms are used for AH and for the authentication option of ESP.

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.

Fig 5.7: IP security Document overview

IPSec Services

IPSec provides security services at the IP layer by enabling a system to select required security protocols, determine the algorithm(s) to use for the service(s), and put in place any cryptographic keys required to provide the requested services.

Two protocols are used to provide security:

- An authentication protocol: Designated by the header of the protocol, Authentication Header (AH);
- Encryption/authentication protocol designated by the format of the packet for that protocol, Encapsulating Security Payload (ESP).

The services are

- Access control
- Connectionless integrity
- Data origin authentication
- Rejection of replayed packets (a form of partial sequence integrity)
- Confidentiality (encryption)
- Limited traffic flow confidentiality

Security Associations

A key concept that appears in both the authentication and confidentiality mechanisms for IP is the security association (SA). An association is a one-way relationship between a sender and a receiver that affords security services to the traffic carried on it.

A security association is uniquely identified by three parameters:

- Security Parameters Index (SPI)
- IP Destination Address
- Security Protocol Identifier

SA Parameters

A security association is normally defined by the following parameters:

a) Sequence Number Counter:

A 32-bit value used to generate the Sequence Number field in AH or ESP headers.

b) Sequence Counter Overflow:

A flag indicating whether overflow of the Sequence Number Counter should generate an auditable event and prevent further transmission of packets on this SA.

c) Anti-Replay Window:

Used to determine whether an inbound AH or ESP packet is a replay.

d) AH Information:

Authentication algorithm, keys, key lifetimes, and related parameters being used with AH (required for AH implementations).

e) ESP Information:

Encryption and authentication algorithm, keys, initialization values, key lifetimes, and related parameters being used with ESP (required for ESP implementations).

f) Lifetime of This Security Association:

A time interval or byte count after which an SA must be replaced with a new SA or terminated, plus an indication of which of these actions should occur .

g) IPSec Protocol Mode: Tunnel, transport.
h) Path MTU: Any observed path maximum transmission unit and aging variables.

5.2.2.3. Modes of Transfer

Both AH and ESP support two modes of use: transport and tunnel mode.

Transport Mode:

Transport mode provides protection primarily for upper-layer protocols. That is, transport mode protection extends to the payload of an IP packet.

Tunnel Mode:

Tunnel mode provides protection to the entire IP packet. To achieve this, after the AH or ESP fields are added to the IP packet, the entire packet plus security fields is treated as the payload of new "outer" IP packet with a new outer IP header.

The entire original, or inner, packet travels through a "tunnel" from one point of an IP network to another; no routers along the way are able to examine the inner IP header. Because the original packet is encapsulated, the new, larger packet may have totally different source and destination addresses, adding to the security.

5.2.2.4. Authentication Header

The Authentication Header provides support for data integrity and authentication of IP packets. The Authentication Header consists of the following fields:

- **Next Header (8 bits):** Identifies the type of header immediately following this header.
- **Payload Length (8 bits):** Length of Authentication Header in 32-bit words, minus 2.
- **Reserved (16 bits):** For future use.
- **Security Parameters Index (32 bits):** Identifies a security association.
- **Sequence Number (32 bits):** A monotonically increasing counter value.
- **Authentication Data (variable):** A variable-length field (must be an integral numberof 32-bit words) that contains the Integrity Check Value (ICV), or MAC

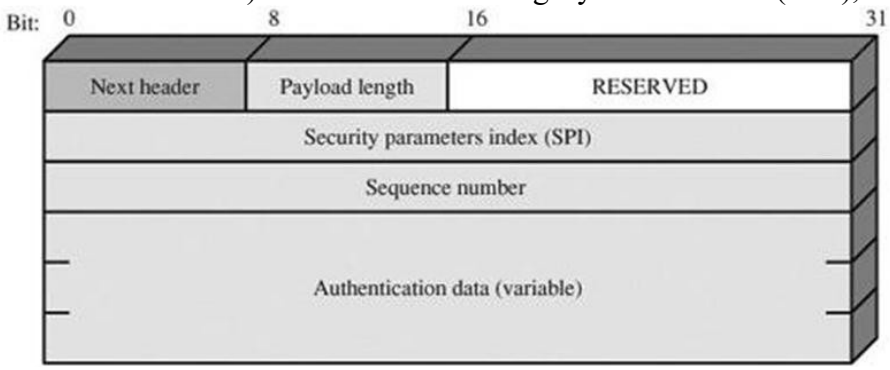


Fig 5.8: IPSec Authentication Header

Anti-Replay Service

A replay attack is one in which an attacker obtains a copy of an authenticated packet and later transmits it to the intended destination.

When a new SA is established, the sender initializes a sequence number counter to 0. Each time that a packet is sent on this SA, the sender increments the counter and places the value in the Sequence Number field. Thus, the first value to be used is 1.

If anti-replay is enabled (the default), the sender must not allow the sequence number to cycle past 2^{32}-1 back to zero. Otherwise, there would be multiple valid packets with the same sequence number. If the limit of 2^{32}-1 is reached, the sender should terminate this SA and negotiate a new SA with a new key.

Integrity Check Value

The Authentication Data field holds a value referred to as the Integrity Check Value. The ICV is a message authentication code or a truncated version of a code produced by a MAC algorithm.

Transport and Tunnel Modes

For transport mode AH using IPv4, the AH is inserted after the original IP header and before the IP payload

For tunnel mode AH, the entire original IP packet is authenticated, and the AH isinserted between the original IP header and a new outer IP header

5.2.2.5. Encapsulating Security Payload

The Encapsulating Security Payload provides confidentiality services, including confidentiality of message contents and limited traffic flow confidentiality.

The diagram shows the format of an ESP packet. It contains the following fields:

- **Security Parameters Index (32 bits):** Identifies a security association.
- **Sequence Number (32 bits):** A monotonically increasing counter value; this provides an anti-replay function, as discussed for AH.
- **Payload Data (variable):** This is a transport-level segment (transport mode) or IPpacket (tunnel mode) that is protected by encryption.
- **Padding (0255 bytes):**The purpose of this field is discussed later.
- **Pad Length (8 bits):** Indicates the number of pad bytes immediately preceding this field.
- **Next Header (8 bits):** Identifies the type of data contained in the payload data field by identifying the first header in that payload (for example, an extension header in IPv6, or an upper-layer protocol such as TCP).
- **Authentication Data (variable):** A variable-length field (must be an integral number of 32-bit words) that contains the Integrity Check Value computed over the ESP packet minus the Authentication Data field.

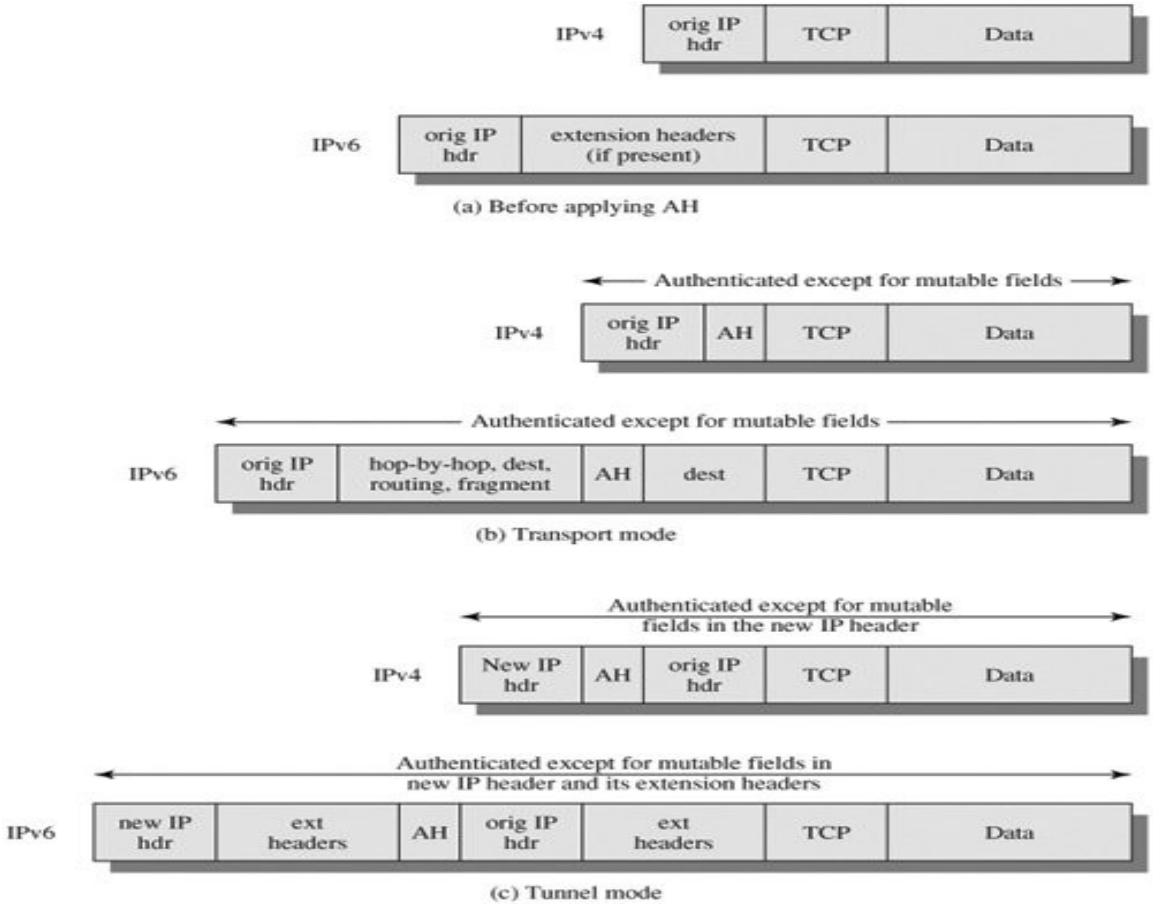


Fig 5.9: Scope of AH Authentication

Padding:

The Padding field serves several purposes:

- If an encryption algorithm requires the plaintext to be a multiple of some number of bytes (e.g., the multiple of a single block for a block cipher), the Padding field is used to expand the plaintext (consisting of the Payload Data, Padding, Pad Length, and Next Header fields) to the required length.
- The ESP format requires that the Pad Length and Next Header fields be right aligned within a 32-bit word. Equivalently, the ciphertext must be an integer multiple of 32 bits. The Padding field is used to assure this alignment.
- Additional padding may be added to provide partial traffic flow confidentiality by concealing the actual length of the payload.

Transport and Tunnel Modes

ESP service can be used. In the upper part of the figure, encryption (and optionally authentication) is provided directly between two hosts.

The diagram shows how tunnel mode operation can be used to set up a virtual private network.

In this example, an organization has four private networks interconnected across the Internet.

Hosts on the internal networks use the Internet for transport of data but do not interact with other Internet-based hosts.

By terminating the tunnels at the security gateway to each internal network, the configuration allows the hosts to avoid implementing the security capability. The former technique is support by a transport mode SA, while the latter technique uses a tunnel mode SA.

Transport Mode ESP:

For this mode using IPv4, the ESP header is inserted into the IP packet immediatelyprior to the transport-layer header (e.g., TCP, UDP, ICMP) and an ESP trailer (Padding, Pad Length, and Next Header fields) is placed after the IP packet; if authentication is selected, the ESP Authentication Data field is added after the ESP trailer.

The entire transport-level segment plus the ESP trailer are encrypted. Authentication covers all of the cipher text plus the ESP header.

For this mode using IPv4, the ESP header is inserted into the IP packet immediatelyprior to the transport-layer header (e.g., TCP, UDP, ICMP) and an ESP trailer (Padding, Pad Length, and Next Header fields) is placed after the IP packet; if authentication is selected, the ESP Authentication Data field is added after the ESP trailer.

The entire transport-level segment plus the ESP trailer are encrypted. Authentication covers all of the cipher text plus the ESP header.

Transport mode operation provides confidentiality for any application that uses it, thus avoiding the need to implement confidentiality in every individual application. This mode of operation is also reasonably efficient, adding little to the total length of the IP packet. One drawback to this mode is that it is possible to do traffic analysis on the transmitted packets.

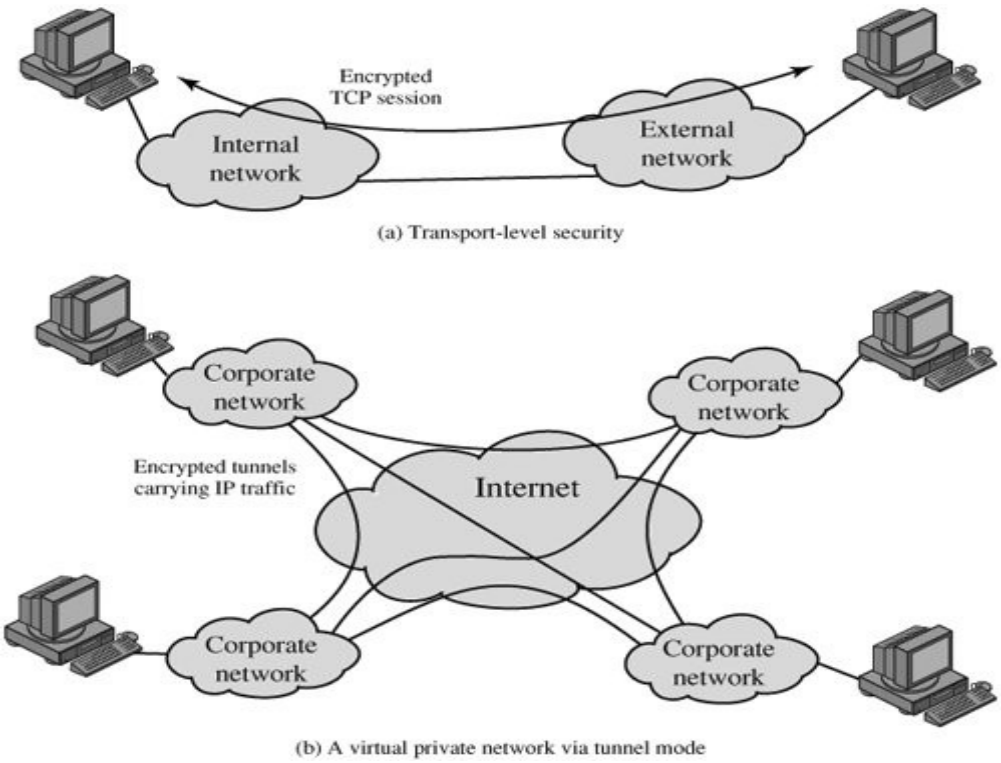


Fig 5.10: Transport Mode

Tunnel Mode ESP

Tunnel mode ESP is used to encrypt an entire IP packet. For this mode, the ESP header is prefixed to the packet and then the packet plus the ESP trailer is encrypted. This method can be used to counter traffic analysis.

Because the IP header contains the destination address and possibly source routing directives and hop-by-hop option information, it is not possible simply to transmit the encrypted IP packet prefixed by the ESP header. Intermediate routers would be unable to process such a packet.

Therefore, it is necessary to encapsulate the entire block (ESP header plus cipher text plus Authentication Data, if present) with a new IP header that will contain sufficient information for routing but not for traffic analysis.

5.2.3 COMBINING SECURITY ASSOCIATIONS

An individual SA can implement either the AH or ESP protocol but not both. Sometimes a particular traffic flow will call for the services provided by both AH and ESP. Further, a particular traffic flow may require IPsec services between hosts and, for that same flow, separate services between security gateways, such as firewalls.

Security associations may be combined into bundles in two ways:

Transport adjacency:

Refers to applying more than one security protocol to the same IP packet withoutinvoking tunneling. This approach to combining AH and ESP allows for only one level ofcombination; further nesting yields no added benefit since the processing is performed at one IPsec instance: the (ultimate) destination.

Iterated tunneling:

Refers to the application of multiple layers of security protocols effected through IP tunneling. This approach allows for multiple levels of nesting, since each tunnel can originate or terminate at a different IPsec site along the path.

The two approaches can be combined, for example, by having a transport SA between hosts travel part of the way through a tunnel SA between security gateways.

One interesting issue that arises when considering SA bundles is the order in which authentication and encryption may be applied between a given pair of endpoints and the ways of doing so. We examine that issue next. Then we look at combinations of SAs that involve at least one tunnel.

Authentication plus Confidentiality

Encryption and authentication can be combined in order to transmit an IP packet thathas both confidentiality and authentication between hosts. We look at several approaches.

5.2.3.1 ESP with Authentication Option

This approach is illustrated in diagram. In this approach, the user first applies ESP to the data to be protected and then appends the authentication data field. There are actually two subcases:

Transport mode ESP: Authentication and encryption apply to the IP payload delivered to the host, but the IP header is not protected.

Tunnel mode ESP: Authentication applies to the entire IP packet delivered to the outer IP destination address (e.g., a firewall), and authentication is performed at that destination. The entire inner IP packet is protected by the privacy mechanism for delivery to the inner IP destination.

For both cases, authentication applies to the cipher text rather than the plaintext.

Transport Adjacency

Another way to apply authentication after encryption is to use two bundled transportSAs, with the inner being an ESP SA and the outer being an AH SA. In this case, ESP is used without its authentication option. Because the inner SA is a transport SA, encryption is appliedto the IP payload. The resulting packet consists of an IP header (and possibly IPv6 header extensions) followed by an ESP.AH is then applied in transport mode,

so that authentication covers the ESP plus the original IP header (and extensions)except for mutable fields. The advantage of this approach over simply using a single ESP SA with the ESP authentication option is that the authentication covers more fields, including the source and destination IP addresses.The disadvantage is the overhead of two SAs versus one SA.

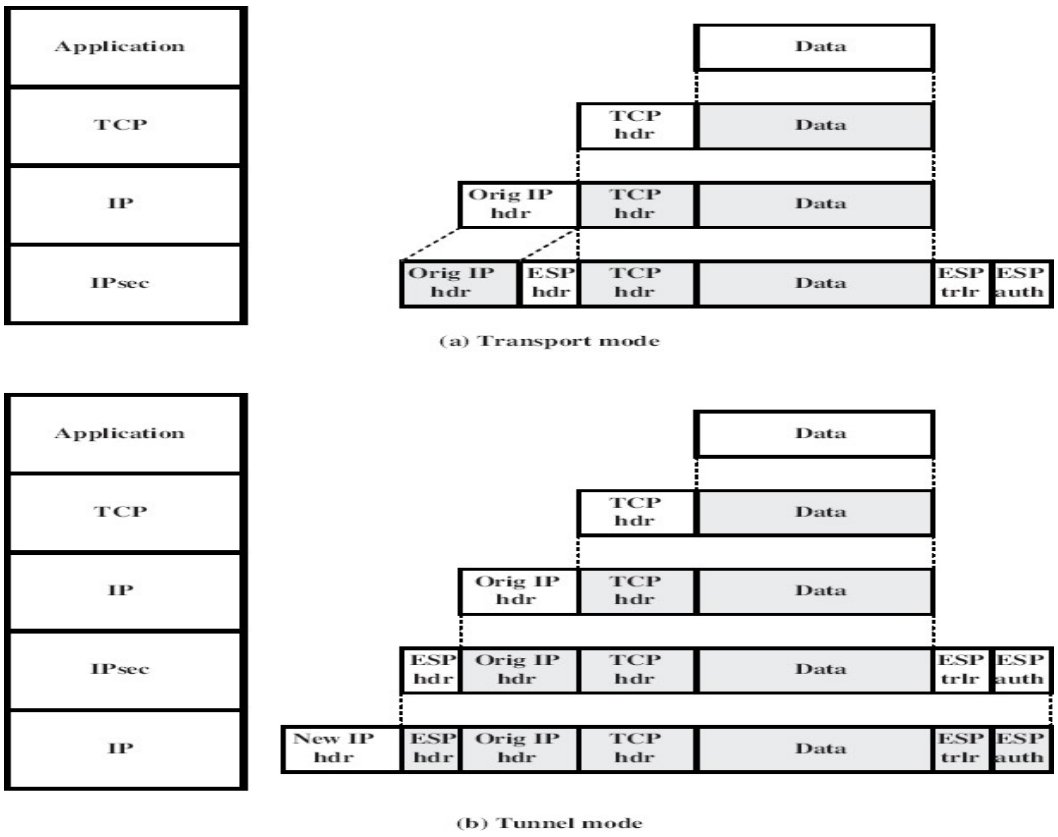


Fig 5.11: Protocol Operation for ESP

Basic Combinations of Security Associations

The IPsec Architecture document lists four examples of combinations of SAs that must be supported by compliant IPsec hosts (e.g., workstation, server) or security gateways (e.g. firewall, router).These are illustrated in Figure .

The lower part of each case in the figure represents the physical connectivity of the elements; the upper part represents logical connectivity via one or more nested SAs. Each SA

can be either AH or ESP. For host-to-host SAs, the mode may be either transport or tunnel; otherwise it must be tunnel mode.

Case 1: All security is provided between end systems that implement IPsec.

For any two end systems to communicate via an SA, they must share the appropriate secret keys. Among the possible combinations are

- AH in transport mode
- ESP in transport mode
- ESP followed by AH in transport mode (an ESP SA inside an AH SA)
- Any one of a, b, or c inside an AH or ESP in tunnel mode

We have already discussed how these various combinations can be used to support authentication, encryption, authentication before encryption, and authentication after encryption.

Case 2: Security is provided only between gateways (routers, firewalls, etc.) and no hosts implement IPsec. This case illustrates simple virtual private network support. The security architecture document specifies that only a single tunnel SA is needed for this case. The tunnel could support AH, ESP, or ESP with the authentication option. Nested tunnels are not required, because the IPsec services apply to the entire inner packet.

Case 3: This builds on case 2 by adding end-to-end security. The same combinations discussed for cases 1 and 2 are allowed here. The gateway-to-gateway tunnel provides either authentication, confidentiality, or both for all traffic between end systems.When the gateway-to-gateway tunnel is ESP, it also provides a limited form of traffic confidentiality. Individual hosts can implement any additional IPsec services required for given applications or given users by means of end-to-end SAs.

Case 4: This provides support for a remote host that uses the Internet to reach an organization‟s firewall and then to gain access to some server or workstation behind the firewall. Only tunnel mode is required between the remote host and the firewall. As in case 1, one or two SAs may be used between the remote host and the local host.

5.2.4 KEY MANAGEMENT

The key management portion of IPSec involves the determination and distribution of secret keys. 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.

(firewall, router).These are illustrated in Figure .The lower part

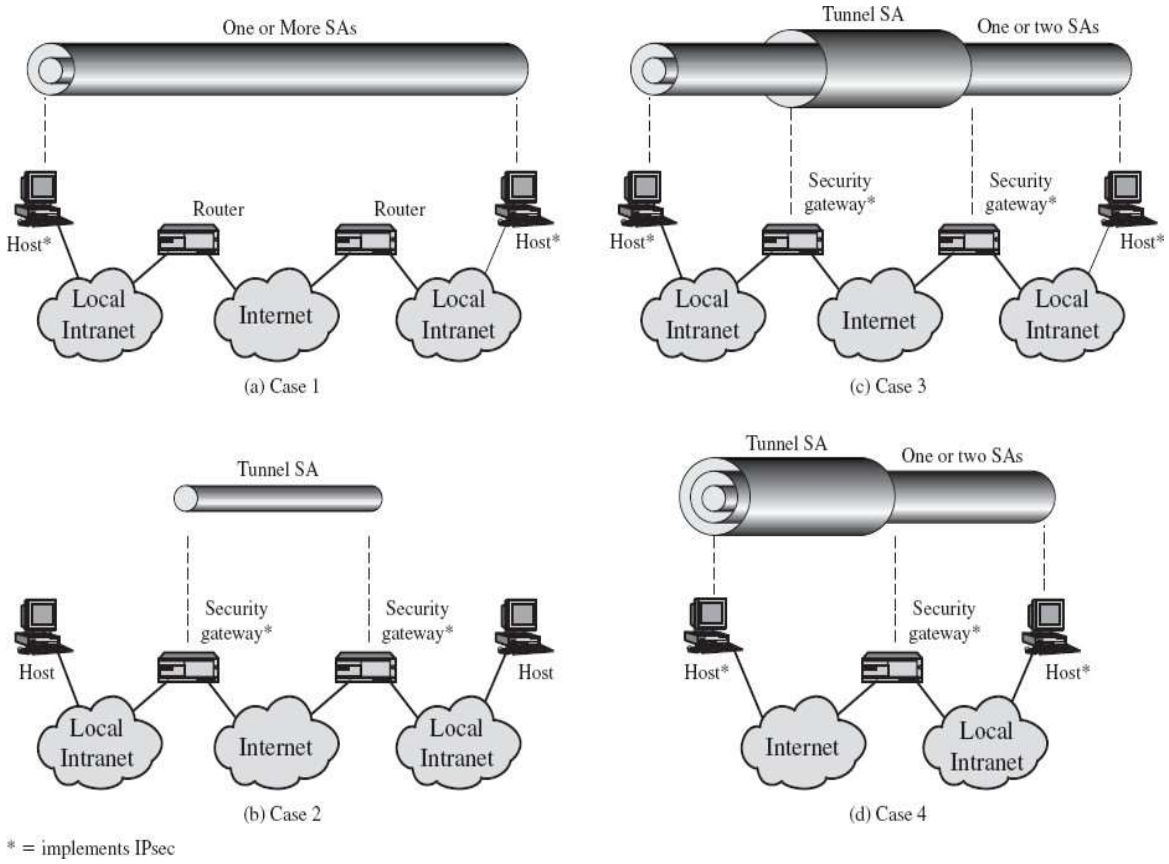


Fig 5.12: Basic Combinations of Security Associations

5.2.4.1. Oakley Key Determination Protocol

Oakley is a refinement of the Diffie-Hellman key exchange algorithm. There is prior agreement on two global parameters: q, a large prime number; and 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 keysfor a long period of time, exposing them to increased vulnerability.
- The exchange requires no preexisting infrastructure other than an agreement on the global parameters.

5.2.4.2. Features of Oakley

The Oakley algorithm is characterized by five important features:

- It employs a mechanism known as cookies to thwart clogging attacks.
- It enables the two parties to negotiate a group; this, in essence, specifies the global parameters of the Diffie-Hellman key exchange.

- It uses nonces to ensure against replay attacks.
- It enables the exchange of Diffie-Hellman public key values.
- It authenticates the Diffie-Hellman exchange to thwart man-in-the-middle attacks.Three different authentication methods can be used with Oakley:

- Digital signatures
- Public-key encryption
- Symmetric-key encryption

5.2.4.3. ISAKMP

ISAKMP defines procedures and packet formats to establish, negotiate, modify, and delete security associations.

ISAKMP Header Format

An ISAKMP message consists of an ISAKMP header followed by one or more payloads. It consists of the following fields:

- **Initiator Cookie (64 bits):** Cookie of entity that initiated SA establishment, SAnotification, or SA deletion.
- **Responder Cookie (64 bits)**: Cookie of responding entity; null in first message from initiator.
- **Next Payload (8 bits):** Indicates the type of the first payload in the message; payloads are discussed in the next subsection.
- **Major Version (4 bits)**: Indicates major version of ISAKMP in use.
- **Minor Version (4 bits):** Indicates minor version in use.
- **Exchange Type (8 bits):** Indicates the type of exchange.
- **Flags (8 bits):** Indicates specific options set for this ISAKMP exchange.
- **Message ID (32 bits):** Unique ID for this message.

- **Length (32 bits):** Length of total message (header plus all payloads) in octets.

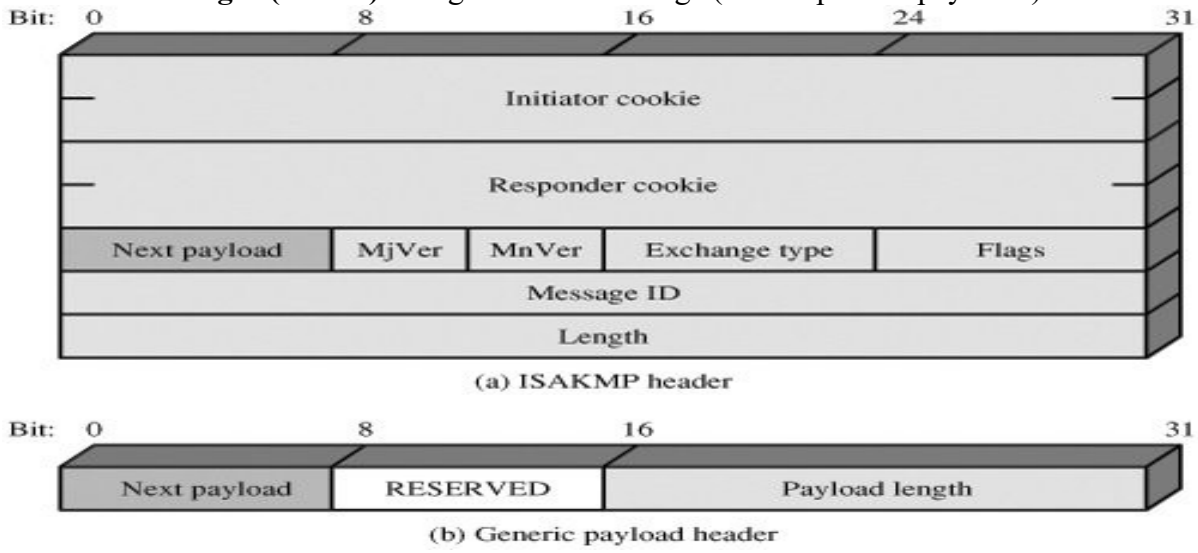


Fig 5.13: ISAKMP Header Format

5.3 WEB SECURITY

5.3.1 WEB SECURITY CONSIDERATIONS

The World Wide Web is fundamentally a client/server application running over the Internet and TCP/IP intranets.

Web Security Threats

A Comparison of Threats on the Web

	Threats	Consequences	Countermeasures
Integrity	Modification of user data Trojan horse browser Modification of memory Modification ofmessage traffic in transit	Loss of information Compromise of machine Vulnerability to all other threats	Cryptographic checksums
Confidentiality	Eavesdropping on the Net Theft of info from server Theft of data from client Info about network configuration Info about which client talks to server	Loss of informationLoss of privacy	Encryption, web proxies
Denial of Service	Killing of user threads Flooding machine with Bogus requests Filling up disk or memory Isolating machine by DNS attacks	Disruptive Annoying Prevent user from getting work done	Difficult to prevent
Authentication	Impersonation o f legitimate users Data forgery	Misrepresentation of user Belief that false information is valid	Cryptographic techniques

Two types of attacks are:

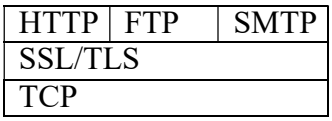
Passive attacks include eavesdropping on network traffic between browser and server and gaining access to information on a Web site that is supposed to be restricted.

Active attacks include impersonating another user, altering messages in transitbetween client and server, and altering information on a Web site.

5.3.2 WEB TRAFFIC SECURITY APPROACHES

One way to provide Web security is to use IP Security. The advantage of using IPSec is

that it is transparent to end users and applications and provides a general-purpose solution.



IP

HTTP	FTP	SMTP
TCP		
IP/IPSec		

(a) Network level

(b) Transport Level

	S/MIME	PGP	SET
Kerberos	SMTP		HTTP
UDP	TCP		
IP			

(c) Application Level

Fig5.14: Relative Facilities in the TCP/IP

Location of Security Protocal Stack

5.3.3 SECURE SOCKET LAYER AND TRANSPORT LAYER SECURITY

5.3.3.1 SSL Architecture

SSL is designed to make use of TCP to provide a reliable end-to-end secure service.

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 SpecProtocol, and the Alert Protocol.

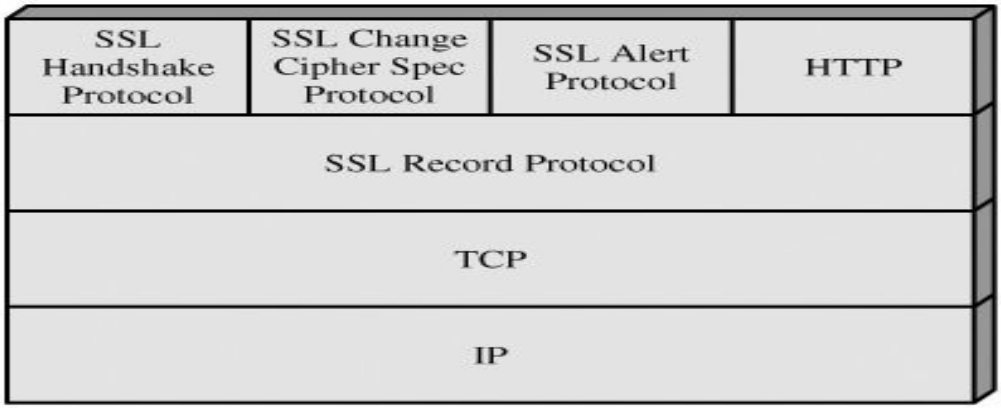


Fig 5.15: SSL Protocol Stack

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 createdby 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.

A session state is defined by the following parameters

- Session identifier
- Peer certificate
- Compression method
- Cipher spec
- Master secret
- Is resumable

A connection state is defined by the following parameters:

- Server and client random
- Server write MAC secret
- Client write MAC secret
- Server write key
- Client write key.
- Initialization vectors
- Sequence numbers

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 diagram indicates the overall operation of the SSL Record Protocol. 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 first step is fragmentation. Each upper-layer message is fragmented into blocks of 2^{14} 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. In SSLv3 (as well asthe current version of TLS), no compression algorithm is specified, so the default compression algorithm is null.

The next step in processing is to compute a **message authentication code** over the compressed data.

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 is3.
- **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 $2^{14} + 2048$.

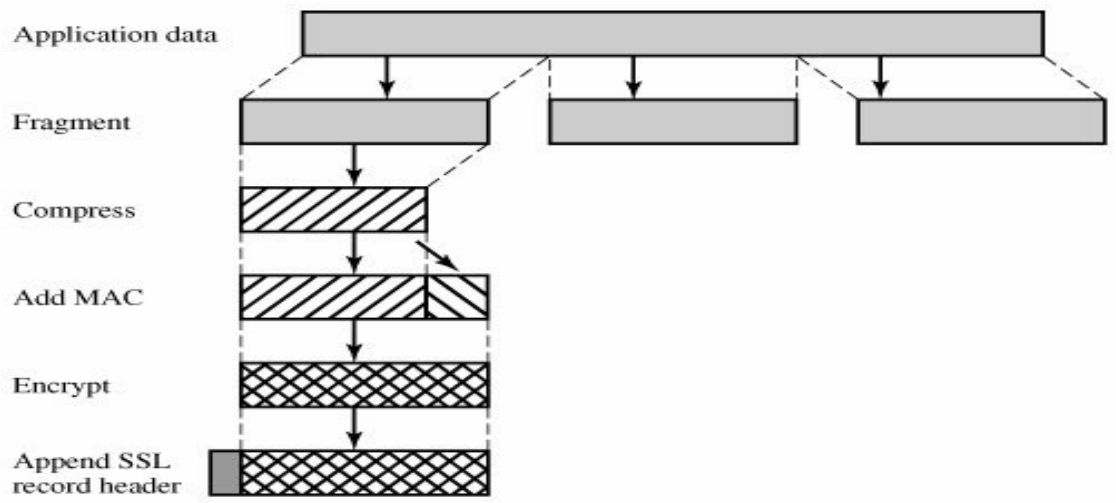


Fig 5.16: SSL Record Protocol Operation

Change Cipher Spec Protocol

This protocol consists of a single message which consists of a single byte with the value 1.

Alert Protocol

The Alert Protocol is used to convey SSL-related alerts to the peer entity.

Each message in this protocol consists of two bytes The first byte takes the value warning(1) or fatal(2) to convey the severity of the message. The second byte contains a code that indicates the specific alert.

- **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 is the following:
  - **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 right 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.

Handshake Protocol

This protocol allows the server and client to authenticate each other and to negotiate an encryption and MAC algorithm and cryptographic keys to be used to protect data sent in an SSL record. The Handshake Protocol is used before any application data is transmitted.

The Handshake Protocol consists of a series of messages exchanged by client and server.

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:

1. **Version:** The highest SSL version understood by the client.
2. **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.
3. **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.
4. **CipherSuite:** This is a list that contains the combinations of cryptographic algorithms supported by the client, in decreasing order of preference.
5. **Compression Method:** This is a list of the compression methods the client supports. After sending the client_hello message, the client waits for the server_hello message, which contains the same parameters as the client_hello message.

Server Authentication and Key Exchange

The server begins this phase by sending its certificate; The certificate message isrequired for any agreed-on key exchange method except anonymous Diffie-Hellman.

Next, a server_key_exchange message may be sent if it is required. The certificate request message includes two parameters: certificate_type and certificate_authorities.

Client Authentication and Key Exchange

If the server has requested a certificate, the client begins this phase by sending a certificate message.Next is the client_key_exchange message, which must be sent in thisphase.

Finally, in this phase, the client may send a certificate_verify message to provide explicit verification of a client certificate.

SSL Handshake Protocol Message Types

Message Type	Parameters
hello_request	null
client_hello	version, random, session id, cipher suite, compression method
server_hello	version, random, session id, cipher suite, compression method
certificate	chain of X.509v3 certificates
server_key_exchange	parameters, signature

SSL Handshake Protocol Message Types	
Message Type	**Parameters**
certificate_request	type, authorities
server_done	null
certificate_verify	signature
client_key_exchange	parameters, signature
finished	hash value

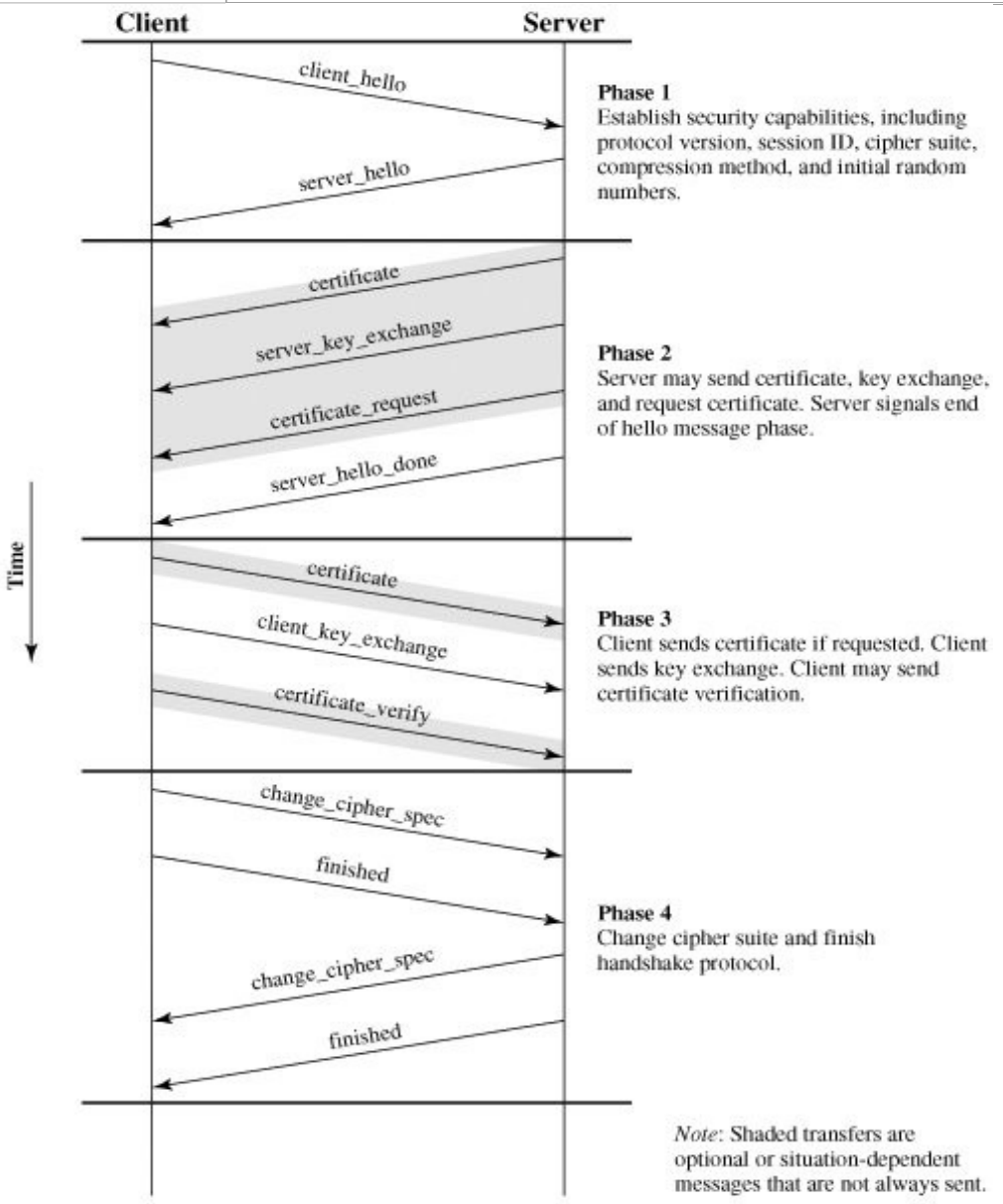


Fig 5.17: Handshake Protocol Action

Finish

This phase completes the setting up of a secure connection. The client sends a change_cipher_spec message and copies the pending CipherSpec into the current CipherSpec.The client then immediately sends the finished message under the new algorithms, keys, and secrets.

Cryptographic Computations

Master Secret Creation

The shared master secret is a one-time 48-byte value (384 bits) generated for this session by means of secure key exchange. The creation is in two stages.

First, a pre_master_secret is exchanged. Second, the master_secret is calculated by both parties. For pre_master_secret exchange, there are two possibilities:

- **RSA:** A 48-byte pre_master_secret is generated by the client, encrypted with the server's public RSA key, and sent to the server. The server decrypts the ciphertext using its private key to recover the pre_master_secret.

- **Diffie-Hellman:** Both client and server generate a Diffie-Hellman public key. After these are exchanged, each side performs the Diffie-Hellman calculation to create the shared pre_master_secret.

5.3.4 SECURE ELECTRONIC TRANSACTION

SET is an open encryption and security specification designed to protect credit card transactions on the Internet.

SET is not itself a payment system. Rather it is a set of security protocols and formats that enables users to employ the existing credit card payment infrastructure on an open network, such as the Internet, in a secure fashion.

SET provides three services:

- Provides a secure communications channel among all parties involved in a transaction
- Provides trust by the use of X.509v3 digital certificates
- Ensures privacy because the information is only available to parties in a transaction when and where necessary

5.3.4.1 Key Features of SET

- Confidentiality of information
- Integrity of data
- Cardholder account authentication
- Merchant authentication

5.3.4.2 SET Participants

- **Cardholder:** A cardholder is an authorized holder of a payment card (e.g., MasterCard, Visa) that has been issued by an issuer.
- **Merchant:** A merchant is a person or organization that has goods or services to sell to the cardholder.
- **Issuer**: This is a financial institution, such as a bank, that provides the cardholder with the payment card.
- **Acquirer**: This is a financial institution that establishes an account with a merchant and

processes payment card authorizations and payments.

- **Payment gateway**: This is a function operated by the acquirer or a designated third party that processes merchant payment messages.
- **Certification authority (CA):** This is an entity that is trusted to issue X.509v3 public-key certificates for cardholders, merchants, and payment gateways.

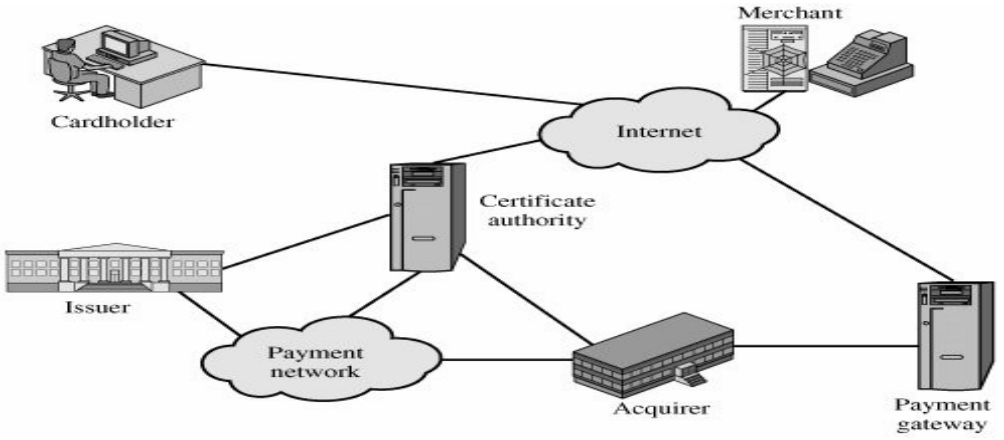


Fig 5.18: Secure Electronic Commerce Components

5.3.4.3 SET Transaction

- **Customer opens account:** The customer obtains a credit card account, such as MasterCard or Visa, with a bank that supports electronic payment and SET.
- **Customer receives a certificate:** After verification the customer receives X.509V3, digital certificate which is signed by the bank. This certificate verifies the customer‟s RSApublic key and expiration date.
- **Merchants have their own certificates:**
    - ✓ Merchants who accepts card need to have 2 certificates for 2 public keys owned by them.
    - ✓ One certificate is used for signing of message and the other is used for key exchange.
    - ✓ The merchants also need the copy of payment gateway‟s public key certificate.
- **Customer places an order:**
    - ✓ The customer places the order containing the list of items to be purchased to the merchant.
    - ✓ The merchant returns the order form having the items, price, total price and order number.
- **Merchant is verified:** The merchant along with the order form sends its certificate copy. The customer can verify the same.
- **Order and payment are sent:**
    - ✓ The customer sends order and payment information into the merchant along with customer‟s certificate.
    - ✓ This is order conformation of the order form.
    - ✓ The payment contains the card details. This is encrypted, so it cannot be read by the merchant.
    - ✓ The certificate sent can be verified by the merchant.

- **Merchant requests payment authorization:** The merchant sends the payment information to the payment gateway. The merchant requests for authentication of the customer, credit limit, validity.
- **Merchant confirms order:** The merchant sends conformation of the order to the customer.
- **Merchant provides goods or service**
- **Merchant requests payment**

5.3.4.4 Dual Signature

The purpose of the dual signature is to link two messages that are intended for two different recipients. In this case, the customer wants to send the order information (OI) to the merchant and the payment information (PI) to the bank. The merchant does not need to know the customer's credit card number, and the bank does not need to know the details of the customer's order.

The customer takes the hash (using SHA-1) of the PI and the hash of the OI. These two hashes are then concatenated and the hash of the result is taken. Finally, the customer encrypts the final hash with his or her private signature key, creating the dual signature. The operation can be summarized as

$$DS = E(PR_c, [H (H (PI) || H (O I)])$$

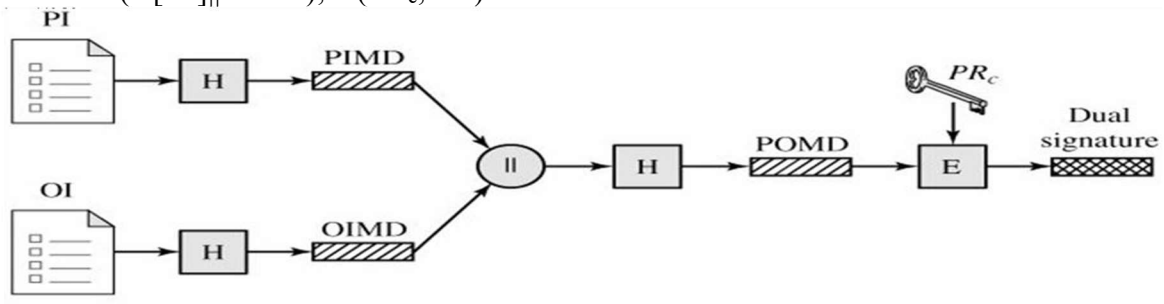
Where PR_c is the customer's private signature key. Now suppose that the merchant is in possession of the dual signature (DS), the OI, and the message digest for the PI (PIMD). The merchant also has the public key of the customer, taken from the customer's certificate. Then the merchant can compute the quantities

$$H (PIMS || H[OI]); D(PU_c, DS)$$

Where PU_c is the customer's public signature key. If these two quantities are equal, then the merchant has verified the signature.

Similarly, if the bank is in possession of DS, PI, the message digest for OI (OIMD), and the customer's public key, then the bank can compute

$$H(H[OI]||OIMD); D(PU_c, DS)$$



PI = Payment information
OI = Order information
H = Hash function(SHA-1)
II = Concatenation
PIMD = PI message digest
OIMD = OI message digest
POMD = Payment order message digest
E = Encryption(RSA)

PR_C = Customer‟s private signature key

Payment Processing

- Purchase request
- Payment authorization
- Payment capture

Purchase Request

Before the Purchase Request exchange begins, the cardholder has completed browsing, selecting, and ordering. The end of this preliminary phase occurs when the merchant sends a completed order form to the customer.

The purchase request exchange consists of four messages: Initiate Request, Initiate Response, Purchase Request, and Purchase Response.

- verifies cardholder certificates using CA sigs
- verifies dual signature using customer's public signature key to ensure order has not been tampered with in transit & that it was signed using cardholder's private signature key
- processes order and forwards the payment information to the payment gateway for authorization (described later)
- sends a purchase response to cardholder

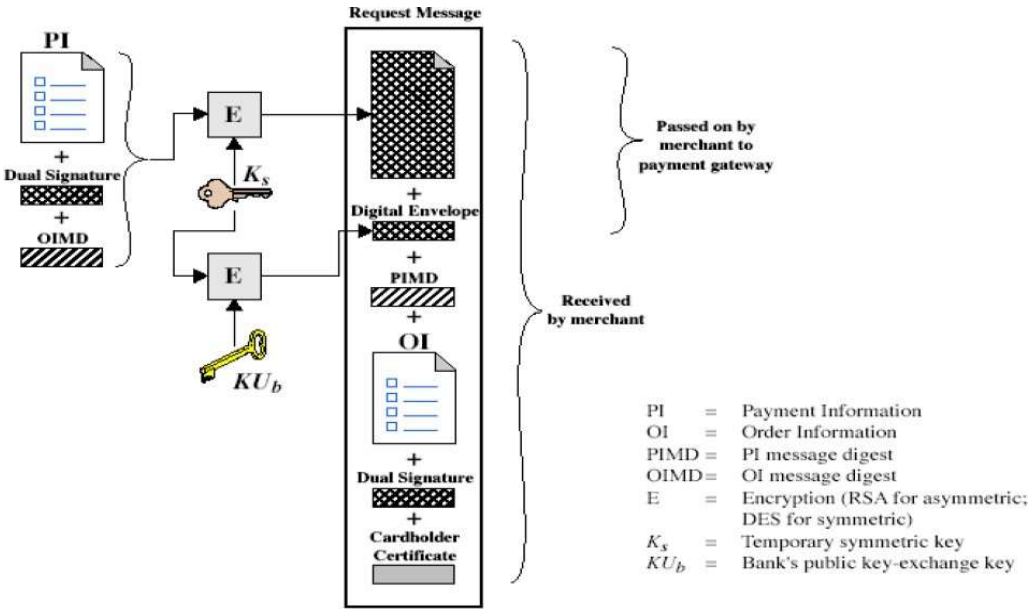


Fig 5.19: Purchase Request – Customer

Payment Authorization

The payment authorization ensures that the transaction was approved by the issuer. This authorization guarantees that the merchant will receive payment; the merchant can therefore provide the services or goods to the customer. The payment authorization exchange consists of two messages: Authorization Request and Authorization response.

- Verifies all certificates

- Decrypts digital envelope of authorization block to obtain symmetric key & thendecrypts authorization block
- Verifies merchant's signature on authorization block
- Decrypts digital envelope of payment block to obtain symmetric key & then decryptspayment block
- Verifies dual signature on payment block
- Verifies that transaction ID received from merchant matches that in PI received (indirectly) from customer
- Requests & receives an authorization from issuer
- Sends authorization response back to merchant

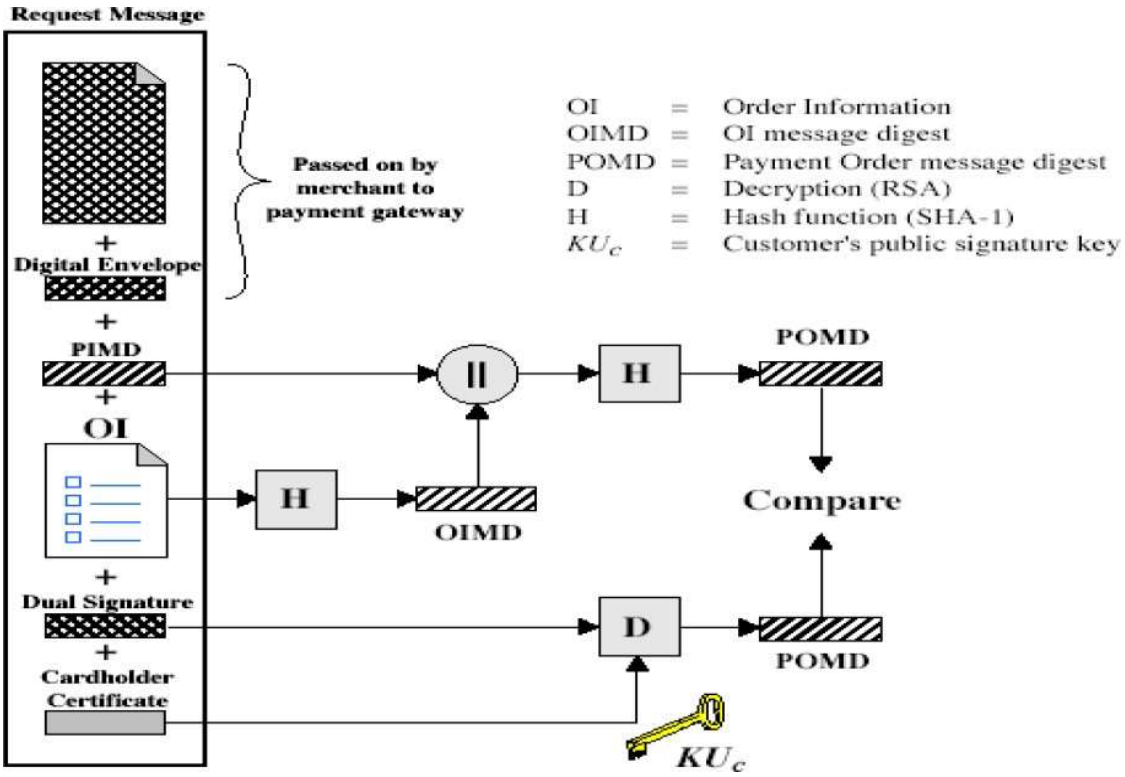


Fig 5.20: Purchase Request – Merchant

Payment Capture

To obtain payment, the merchant engages the payment gateway in a payment capture transaction, consisting of a capture request and a capture response message.

- Merchant sends payment gateway a payment capture request
- Gateway checks request
- Then causes funds to be transferred to merchants account
- Notifies merchant using capture response

Firewalls

Firewall are frequently used to prevent unauthorized Internet users from accessing private networks connected to the Internet,

All messages entering or leaving the intranet pass through the Firewall, which examines each message & blocks those do not meet the specified security criteria.

Firewall is a term used for a "barrier" between a network of machines & users that operate under a common security policy & generally trust each other & the outside world.

Related terminology

1- Rules ⟶ AXS GUARD
2- Policies
3- Security level
4- Firewall Rights
5- Firewall GUI : ⟶ a) Create, Edit
b) Assign Firewall policies
c) Consult Firewall logs & Firewall status.

Characteristics

1. Traffic from inside to outside & vice versa must pass through the Firewall.
2. Only authorized traffic will be allowed to pass
3. Firewall itself is immune to penetration

1) Two filtering routers

1. one between bastion host & Internet
2. one between bastion host & Internal n/w

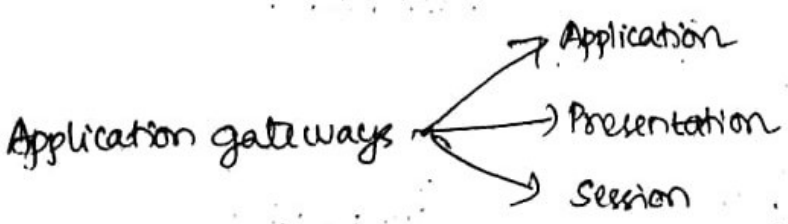
2) Both Internet & the internal network have access to the host on the screened subnet but traffic across the screened subnet is blocked.

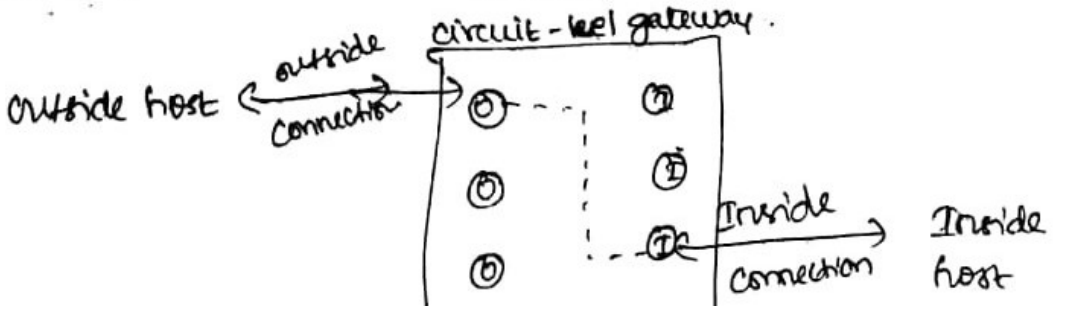
Ref

outer ⟶ screened subnet to Internet.

inside ⟶ screened subnet to Internal n/w

Packet Filters	Application-level gateway	circuit-level gateways.
1. simplest & least secure Firewall mech.	1. Most secure approach	More secure than packet, but less
2. Many routers provide this function.	2. Unique pgm for each application	not as secure as application
3. Passes or rejects packets based on rules	3. Good for authentication, logging:	Relay Tcp connection
4. Hard to manage.	4. Not always transparent to users.	permission granted by Port address.
	Used for e-mail, FTP, Telnet, WWW	No application-level checking





Routers

Routers route packets of data from one network to another. Some routers even control the internet's infrastructure.

Gateways

Gateways are the portals that computers use to connect to the internet. One computer can be a gateway for others which is made possible through ICS (Internet connection sharing)

Differences

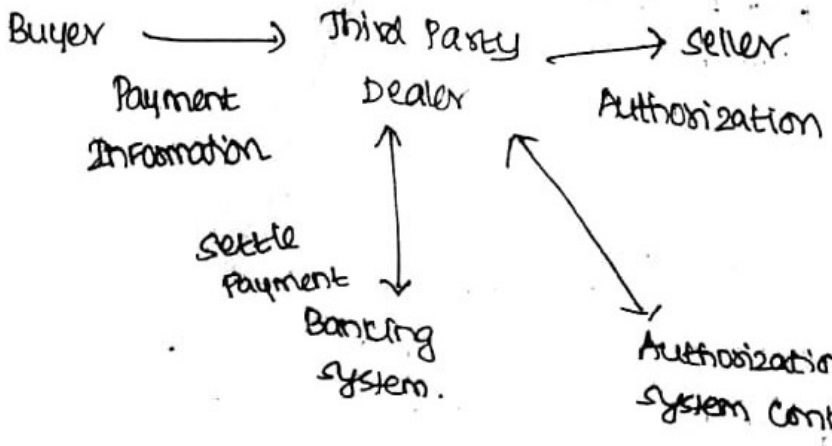
Router coordinates data transfer from one computer to another within a n/w & to other n/w.

Gateway is any device specifically designed to provide all the computer in a n/w with access to WWW.

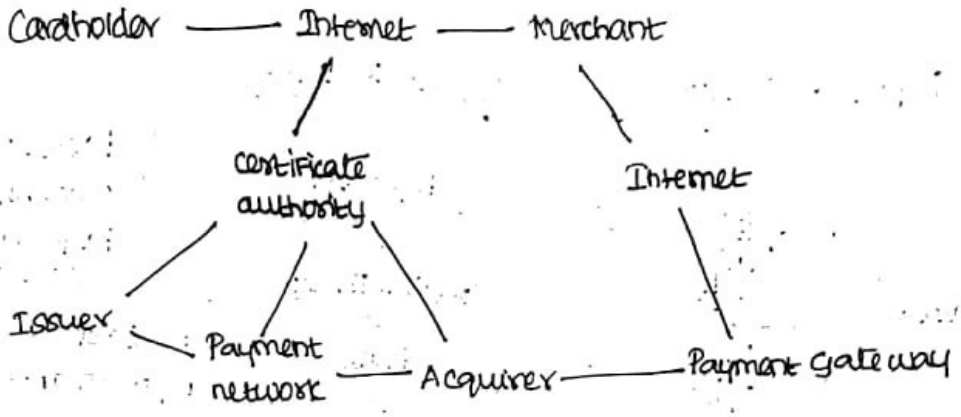
SET For E-Commerce Transactions

The Secure Electronic Transaction (SET) is the credit card payment protocol that securely transfers the money from customer to the merchant with high integrity.

Structure



SET components



Requirements

* Payment Confidentiality
* Ordering Confidentiality
* Transmitted Data Integrity
* Transmitted data privacy
* Card holder Buyer Authentication
* Seller Authentication

Dual signature

Authorizing that particular payment is intended for particular order.

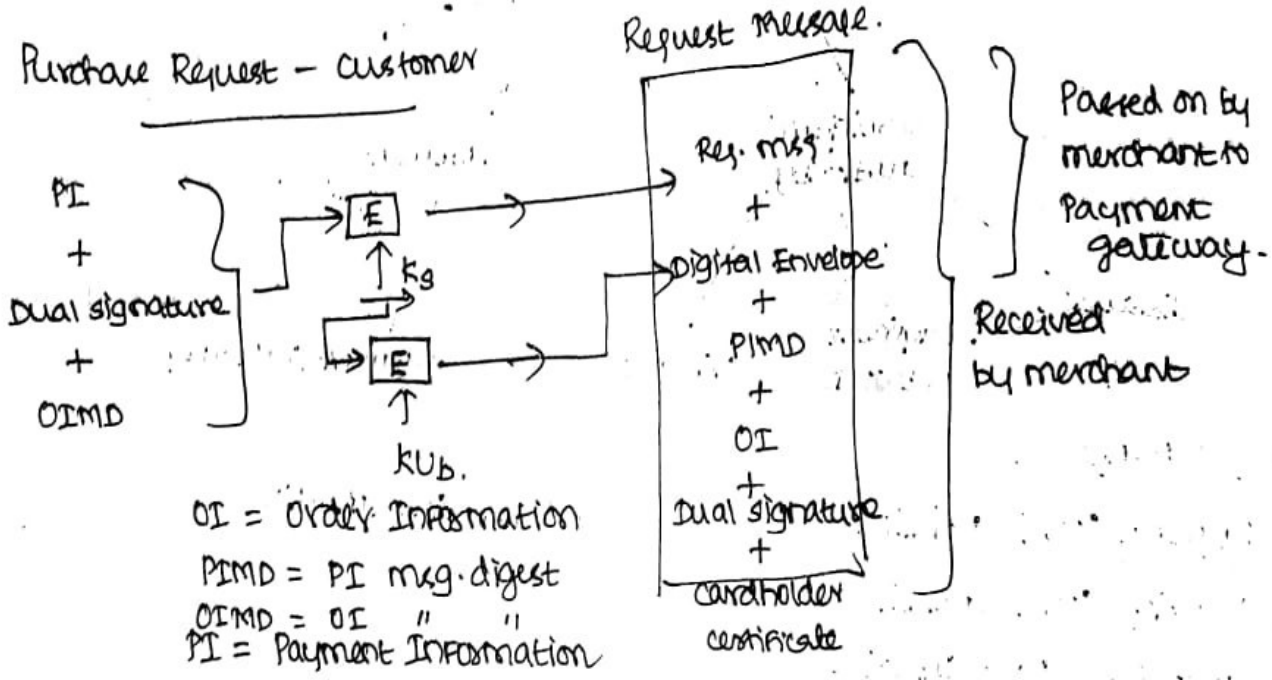
1. Payment Information (PI) for bank
2. Order Information (OI) for merchant

" The purpose of dual signature is to link two messages that are intended for two different recipients.

Customer wants to send the Order Information (OI) to the merchant & the payment Information (PI) to the bank. Merchant does not need to know the customer's credit-card number, and the bank does not need to know the details of the customer's order.

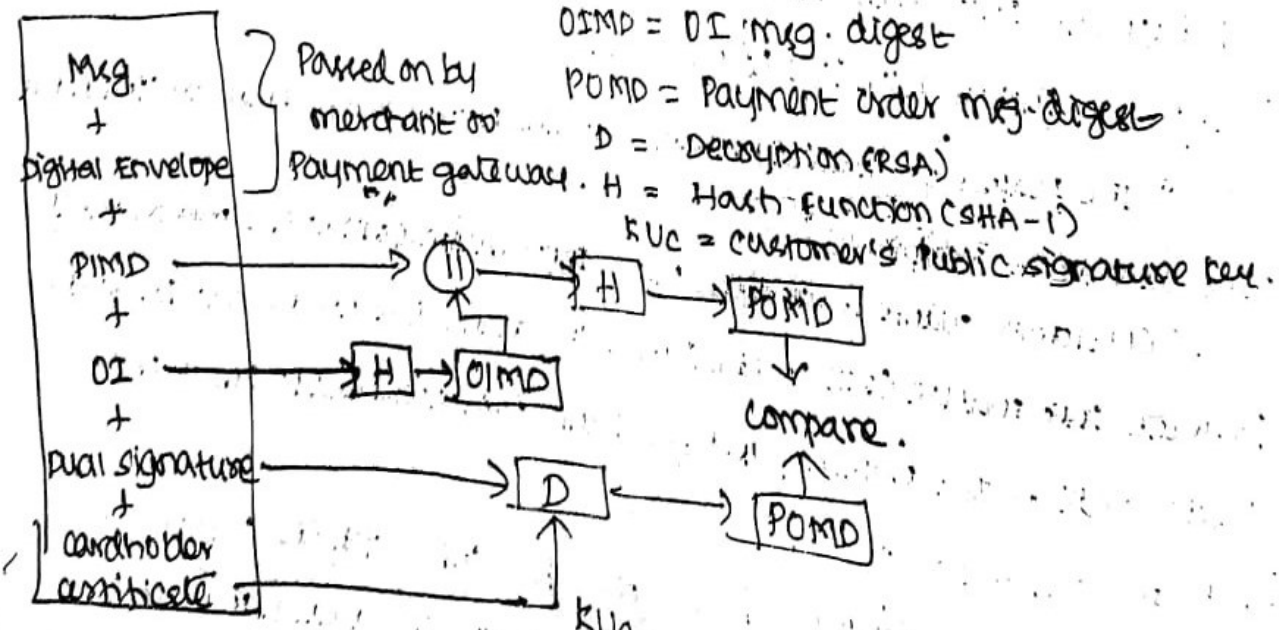
The dual signature is the encrypted MD (with the customer's secret key) of the concatenated MD's of PI and OI. The dual signature is sent to both the merchant and the bank. The protocol arranges for the merchant to see the MD of the PI without seeing the PI itself, and the bank sees the MD of the OI but not the OI itself. The dual signature can

be verified using the MD of the OI or PI. The MD does not reveal the content of the OI or PI, thus privacy is preserved.



OI = Order Information
PIMD = PI msg. digest
OIMD = OI " "
PI = Payment Information
E = Encryption (RSA for asymmetric; DES for symmetric).
Ks = Temporary symmetric key.
KUb = Bank's Public key-exchange key.

Purchase Request – Merchant



OI = Order Information
OIMD = OI msg. digest
POMD = Payment order msg. digest
D = Decryption (RSA)
H = Hash function (SHA-1)
KUc = customer's public signature key.

1. verifies cardholder certificates using CA
2. verifies dual Signature using customer's Public Signature Key to ensure order has not been tampered with in transit & that it was signed using cardholder's Private Signature key.
3. Processes order & forward the Payment information to the Payment gateway for authorization
4. Sends a Purchase response to Cardholder.

Intrusion Detection

Intruders:

1. Masquerader: Person or a system that is not authorized to use the computer but penetrates a system and access controls to exploit an authorized users account.
2. Misfeasor: An authorized user who misuses the privileges by accessing data, programs or resources for which such access is not authorized.
3. clandestine user: An individual who seizes supervisory control of the system to evade auditing & access controls or to suppress audit collection.

Techniques

The main aim of any Intruder is to gain access to a system or to increase the range of privileges accessible on a system.

The Intruder needs to know certain details that are protected in the system. This can be done by gaining access to the password file for each authorized user.

Techniques (intrusion)

→ Target acquisition & Information gathering
→ Initial access
→ Privilege escalation
→ Covering tracks

Password guessing

→ one of the most common attacks

→ attackers knows a login (from email / web page etc.)

→ try default passwords shipped with systems

→ try all short passwords

→ then try by searching dictionaries of common words

→ Intelligent searches try passwords associated with the user (variations on names, birthday, phone, common words / interfaces)

Password capture

→ watching over shoulder as password is entered

→ using a trojan horse program to collect

→ extracting recorded info after successful login (web history/cache, last number dialed etc)

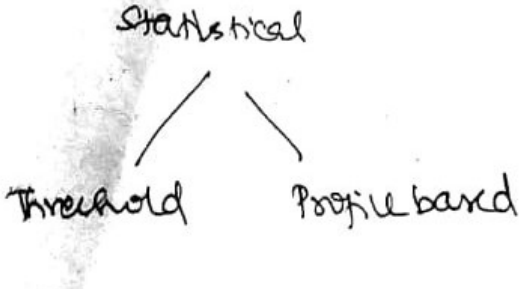
Approaches to Intrusion detection

1. Statistical Anomaly Detection:

Involves collection of data relating to the behaviour of authorized users over a period of time

2. Rule based detection

Involves an attempt to define a set of rules that can be used to decide that a given behavior is that of an intruder.



Statistical

a) Threshold detection

Involves counting the number of occurence of a specific event type over an interval of time.

b) Profile based

Profile of the activity of each user is developed and used to detect the changes in the past behaviour of individual accounts.

Metrics used

1. Counter — keeps a count of certain event types is kept over a particular period of time
2. Gauge — Used to measure the current value of some entity
3. Interval timer — Notes the length of time between two related events
4. Resource utilization — Quantity of resource that is consumed during a period of time

Rule Based

a) Rule based Anomaly detection

→ Historical Audit records are analyzed to identify usage patterns & to generate automatically rules that described patterns.

→ Based on observing past behaviour & assumes that the future will be like the past.

b) Penetration Identific

Rules for identifying known penetration that would exploit known weaknesses.

Rules specific to machine & O.S

Audit Records → Fundamental tool

1. Native Audit record. : Any multiuser os includes accounting s/w that collects the information on user activity
2. Detection specific audit. Facility that can be implemented by generated audit records.

fields

1-subject → Initiate an action Eg: terminal user, group of users

2. Action → Performed by subject Eg: login, read, execute

3. Object → Entities on which Action is performed Eg: files, Programs, rec

4. Exception condition → Raised

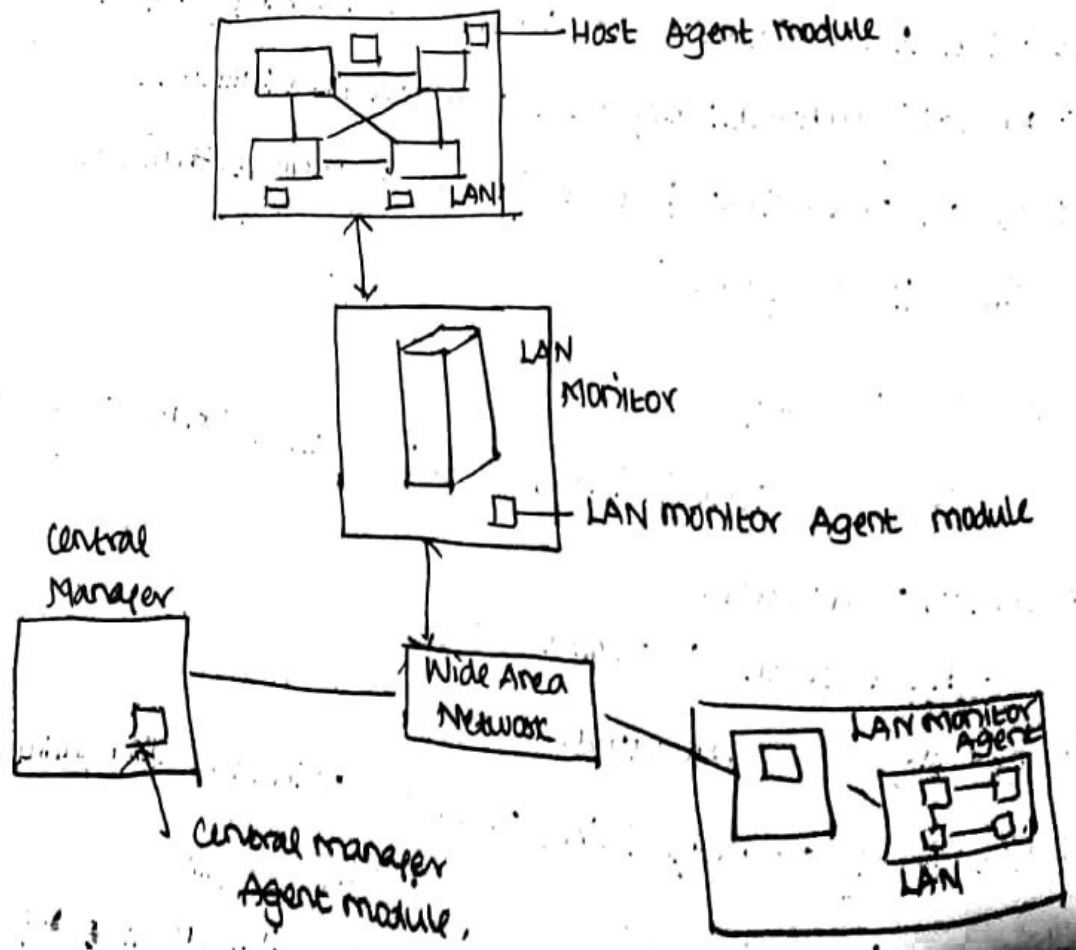
5. Usage → quantitative elements amount list of resources used

6. Time stamp. → Action took place.

Shyam	Read	<Shyam> Hello.Exe	0	CPU = 0005	1106872165
subject	Action	Object	Exception	Resource	TS

Distributed Intrusion Detection method

Architecture of the Intrusion Detection system for Distributed Network is defined by



Three components

1. Host Agent module 2. LAN monitor Agent module 3. Central manager Agent module.

1. Each host system is configured with the agent module to monitor and collect the audit information.

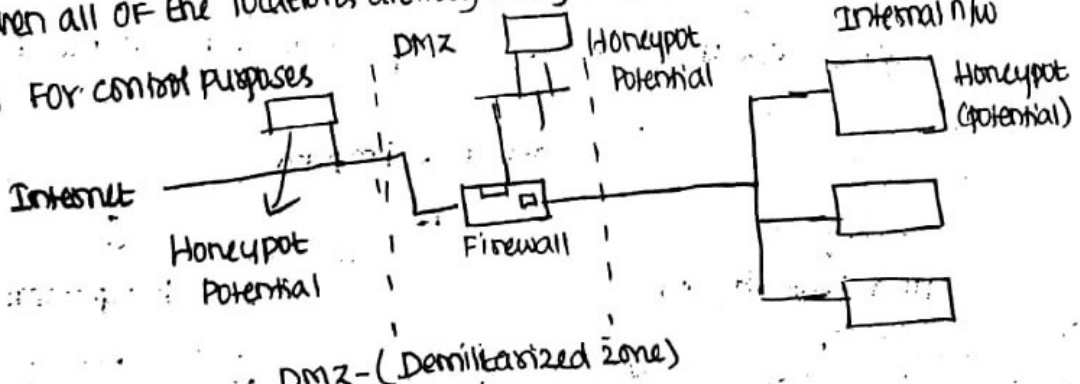
2. Each LAN network has LAN monitor Agent module that analyzes the user agent module & reposts the same to central Manager. ~~Agent module~~

3. The central manager Agent, module collects the repost from LAN monitor agent module & compares the reposte with its predefined reposts to detect the intrusion.

Honeypots

Another Intrusion Detection system. Honeypot systems are decoy servers or systems setup to gather information regarding an attacker or intruder into your system.

Honeypots can be setup inside, outside or in the DMZ of a Firewall design or even all of the locations although they are most often deployed inside of a Firewall for control purposes

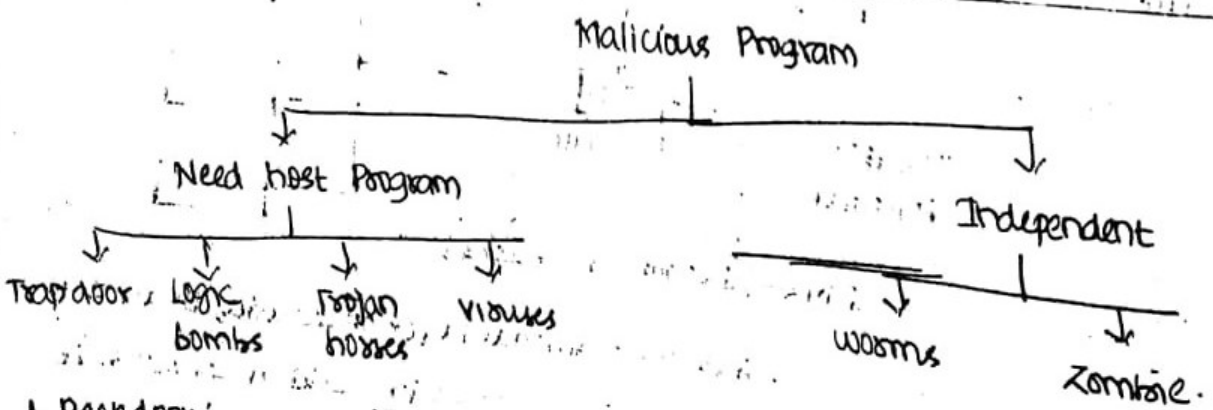


Add an additional layer of security to an LAN

To Designed, to divert an attacker from accessing critical systems. It collects information about the attackers activity, then it encourages the attackers to say stay on the system long enough for administrator to respond.

Viruses and Related threats

	Virus	Worms
How does it infect a computer system?	It inserts itself into a file or executable program	It exploits a weakness in an application or OS by replicating itself
How can it spread?	It has to rely on users transferring infected files/programs to other computer systems	It can use a n/w to replicate itself to other computer system without user intervention
Does it infect files?	Yes, it deletes or modifies files. sometimes a virus also change the location of files.	Usually not. Worms usually monopolize the CPU & memory.
Whose speed is more?	Virus is slower than worm	Worm is faster than virus. Eg: code red worm affected 3 lak PC in 14 hrs.
Definition	The virus is the program code that attaches itself to application program & when application pgm run it runs along with it.	The worm is code that replicate itself in order to consume resources to bring it down.



1. Back door:

Backdoor is also called a trapdoor and it is a secret entry point in the program to gain access with program output. It completely damages the system with unimaginable number of intruders use back door to give access to the programs.

Types of virus / Classification

1. Parasitic virus [Attaches to Executable Files & replicates]
2. Memory Resident virus : Lodges in main memory as a part of resident system Pgm.
3. Boot sector virus : Infects a master boot record or spreads a system is booted from dsk
4. Stealth virus : [To Hide from detection by antivirus s/w]
5. Polymorphic virus : mutates with every Infection ["signature" different antivirus cannot detect]
6. Email virus : Spreading Email virus made use of a ms word attachment - "Melissa virus"
7. Metamorphic virus : [Similar to Polymorphic, gets more Power after each infection to rewrites its own code.
8. Macro virus : Infect Ms word document.

Worms : Remote Execution capability : [worms executes a copy of itself on another system.

Remote login capability : A worm logs onto a remote system as a user & then uses commands to copy itself from one system to another -

Virus Countermeasures

1. Antivirus methods — Virus Detection, Virus Identification, Virus Removal or Deletion
2. Techniques for Antivirus.

Generation of Antivirus s/w

1. First generation - Basic virus scanner [Uses virus signature & detect only specific virus
2. Second - Heuristic : Not using signature & uses heuristic rules to find.
3. Third - Activity : Identify not by structure by the action performed.
4. Fourth - Full Featured : Antivirus s/w techniques that are Parallely executed [Contain capability to restrict reproduce in the system]

Techniques for Antivirus method [To develop antivirus]

1) Generic Decryption method
2) Digital Immune method
3) Behaviour Blocking s/w

1) General Decryption Method :

Detect & Fastly scan the Polymorphic virus from the system. Generally the virus must decrypt itself to be activated in the system.

1. CPU Emulator : (Interprets the instruction in executable file rather than on Processor)
2. Virus signature scanner [Scans the code for known virus signature]
3. Emulation control module [Controls execution of virus infected code]

2) Digital Immune method. [Designed for Internet based virus]

Information from virus Infected client m/c

↓

Administrative m/c.

↓

Received by virus Analysis m/c Engine

↓

Analyze the virus Behaviour

↓

Analyze the virus structure.

↓

Extract the virus signature.

↓

Derive virus description to Individual user.

3) Behaviour Blocking method. [Integrates with OS of a computer & monitors for malicious actions.

1) Modifying critical system settings such as startup
2) Operating, view, delete or modify Files.
3) Initiating N/w communications.
4. Formatting disk.
5. Modify Exec Files.
6. Scripting of e mail & msg to client.

Practical Implementation of cryptography

1. Code book: Code replaces word or phrase with character.

* symbol

∧ % @ ∞ ⌧ ≠

and msg. Night Tanks Bombrun Tmw.

pt. T => Tanks and bomb run tmw. Night.

cod => ∞∧⌧≠@.

2. Nomenclators : Use elements of substitution cipher.

a b c

0 ≠ ∧ . .

3. Smart cards

& For building entry system, ATM's etc.

4. Biometrics : Authenticating an individual by personal characteristics.

Token is unique - [Replace password - based authentication]