St. Joseph's College of Engineering & Technology Palai

Department of Computer Science & Engineering

S8 CS

RT801 Security in Computing

Module 4

Security in Computing - Module 4

Syllabus

Network & Application Security: Kerberos - X509 Authentication service -

IP security Architecture - Secure socket layer -

Electronic mail security - Pretty Good privacy - S/MIME -

Secure Electronic Transactions -

Firewalls - Security mechanisms in JAVA platform -

Applet security - Security policy and Security Manager.

Contents

Part I. Kerberos

Kerberos is a network authentication protocol. It was developed at MIT. For example consider the following situation,

fig

In such a distributed environment, consistiong of a number of computers (PCs, work stations, web servers, file servers etc..), authentication can be implemented using Kerberos.

For example, if Alice wants to log onto a remote file server and access files or other services, Kerberos performs authentication.

Kerberos uses only symmetric encryption to transfer authentication information between computers.

A Kerberos system consists of ,

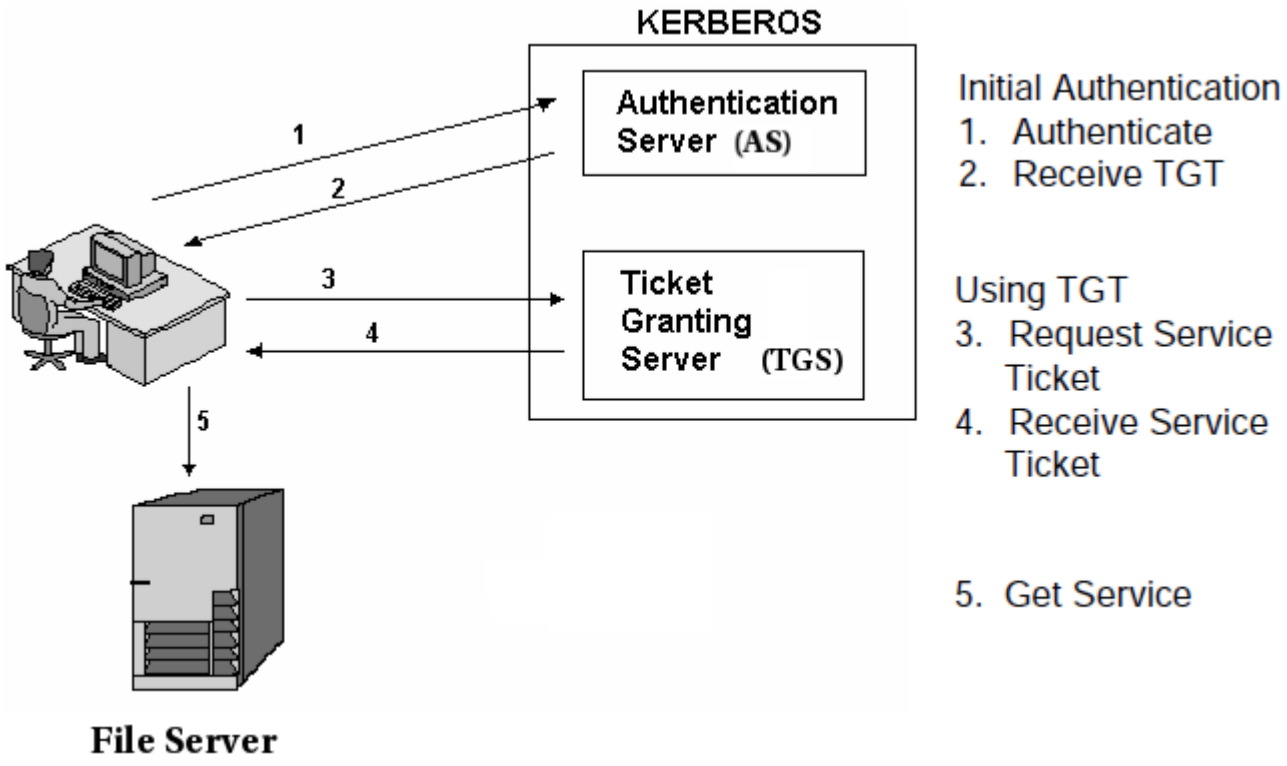
an authentication server (AS), and

a ticket granting server (TGS).

fig

All the users and servers are registered with Kerberos server. That is, a users's password is stored at a Kerberos server.

An overview of Kerberos (version 4) operation is shown below:

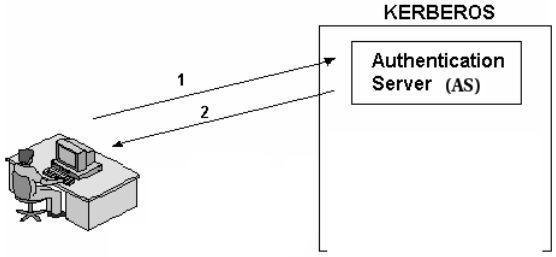


1. User requests AS to provide a ticket (TGT) to access TGS.
2. AS sends the TGT to user.
3. User sends the TGT to TGS requesting access to file server.
4. TGS issues the ticket to user.
5. User sends the ticket to file server and accesses the files.

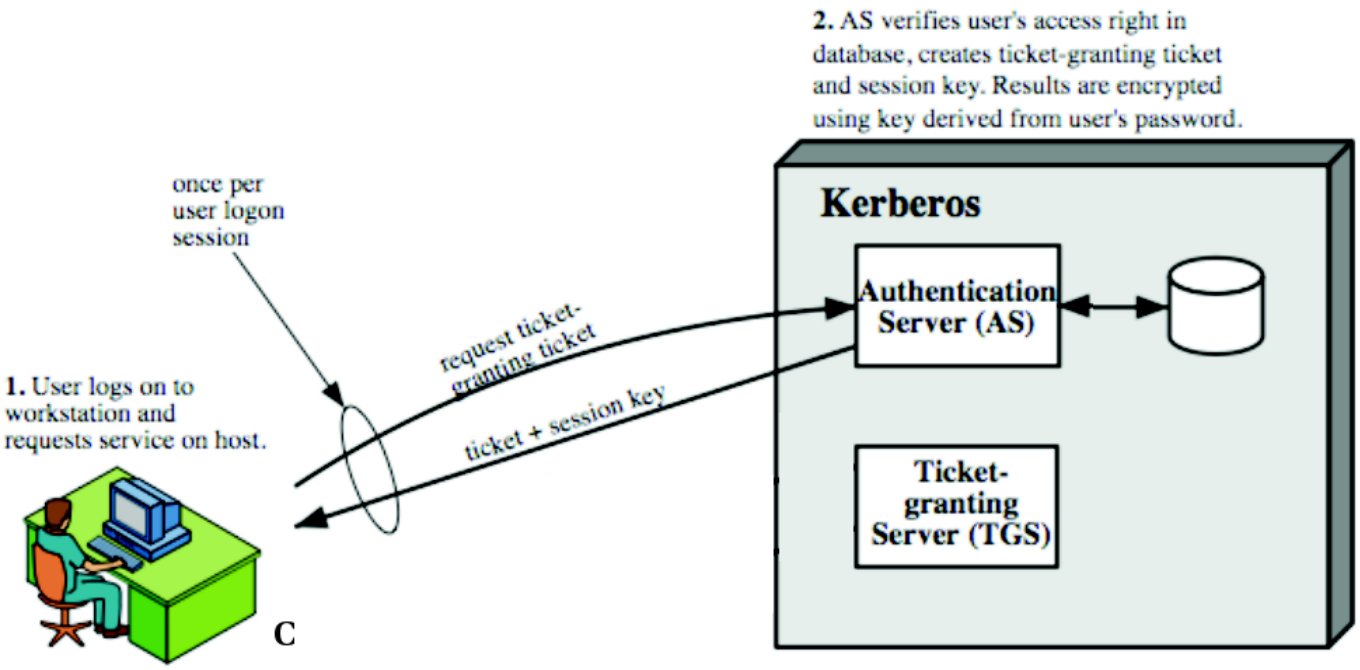
A detailed explanation of the operation of Kerberos is given below:

1 Initial Authentication

Initial authentication is shown below:



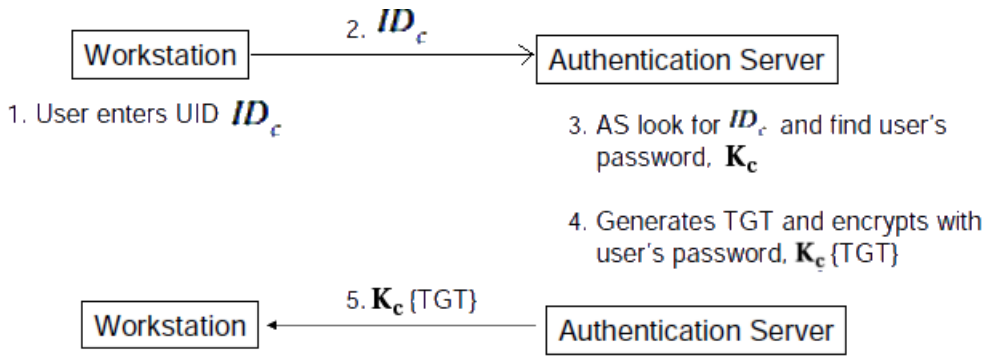
Initial Authentication
1. Authenticate
2. Receive TGT



(1) C → AS $ID_c \| ID_{tgs} \| TS_1$

(2) AS → C $E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$$



6. User enters password
7. Application decypts K_c {TGT} using password.
8. User now have TGT

Thus in Kerberos, authentication is done using user's password.

In the above communication, user's password is never transmitted.

User knows his password and Kerberos AS has a copy of it stored in its database.

TGT is encrypted using user's password.

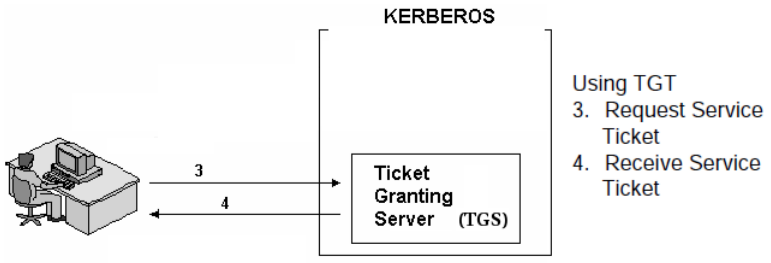
Ticket Granting Ticket (TGT)

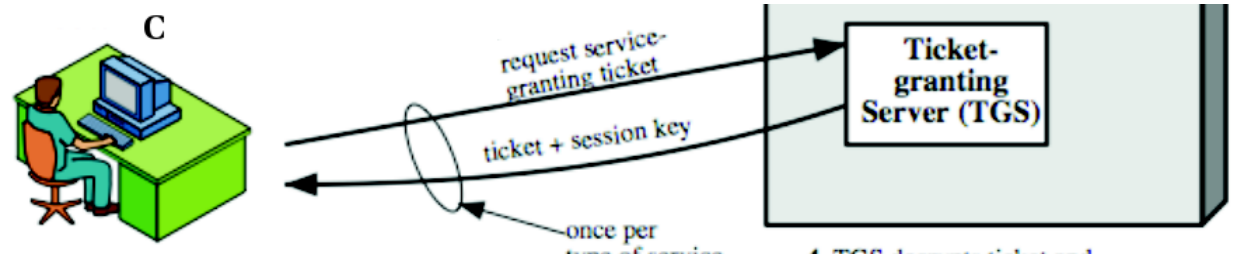
It consists of a session key, $K_{c,tgs}$, ticket for TGS which is encrypted with both the session key and TGS's key.

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$$

After getting TGT, user's workstation can contact Kerberos TGS to obtain ticket for any of the services. For example, if user Alice wants to access a file from a file server, it can contact TGS to obtain a ticket.

2 Using TGT





3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.

4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.

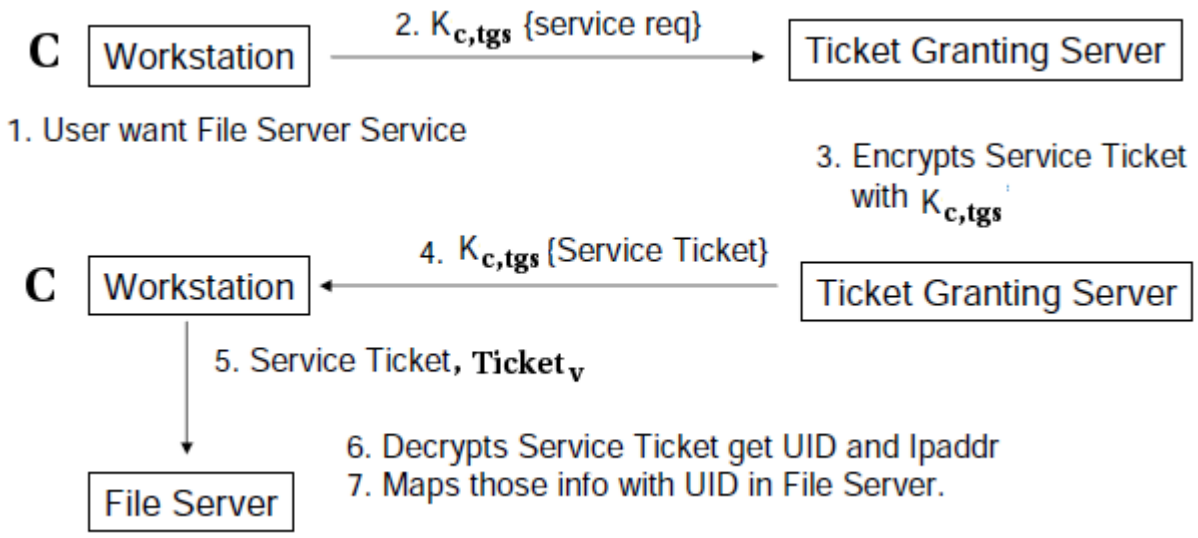
(3) C → TGS $ID_v \| Ticket_{tgs} \| Authenticator_c$

(4) TGS → C $E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_1 \| Ticket_v])$

$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} || ID_c || AD_c || ID_{tgs} || TS_2 || Lifetime_2])$

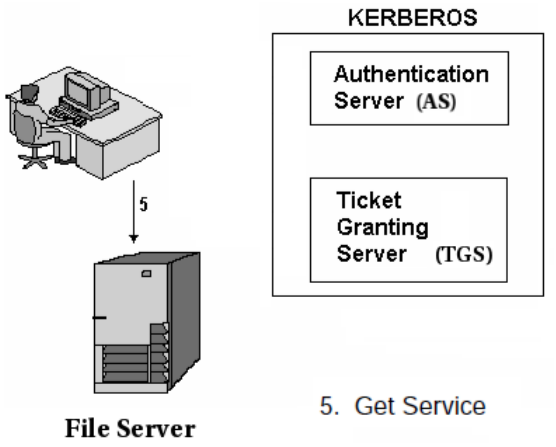
$Ticket_v = E(K_v, [K_{c,v} || ID_c || AD_c || ID_v || TS_4 || Lifetime_4])$

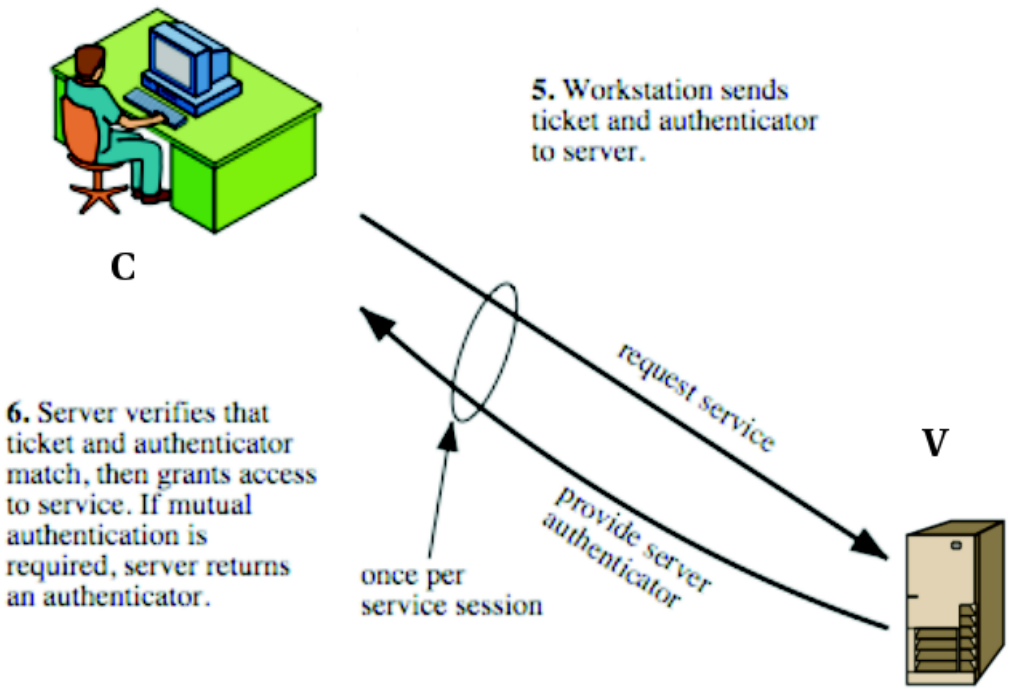
$Authenticator_c = E(K_{c,tgs}, [ID_c || AD_c || TS_3])$



TGS can establish user's identitiy because the request is encrypted using the session key. This is possible only if user can decrypt the TGT from AS.

3 Accessing Service using the Ticket from TGS





(5) C → V $\quad Ticket_v \parallel Authenticator_c$

(6) V → C $\quad E_{K_{C,V}}[\,TS_5 + 1\,]$

$$Ticket_v = E(K_v, [K_{c,v} \,||\, ID_c \,||\, AD_c \,||\, ID_v \,||\, TS_4 \,||\, Lifetime_4])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_c \,||\, AD_c \,||\, TS_3])$$

File server, V can establish user's identity because ticket is encrypted with file server's key and it contains the details of user.

In the communication, there are two types of tickets,

Ticket Granting Ticket (TGT), $Ticket_{tgs}$

and

Service Ticket, $Ticket_v$.

Drawbacks of Kerberos

Every network service must be modified to use with Kerberos.

Kerberos authentication does not work well in a time shared environment.

This mechanism requires a secure Kerberos server and also Kerberos server must be continuously available.

Kerberos server stores all passwords encrypted with a single key.

Part II. X.509 Digital Certificates

X.509 is a protocol which describes the details of using digital certificates.

Certificates

A certificate is a piece of information that proves the identity of owner.

A passport can be treated as a certificate that proves a person's identity. A passport is issued by the passport authority under the government. A passport has a unique serial number, name, address of the person, issue date, expiry date, official seal etc.. On seeing checking its seal and serial number, a person's identity can be proved. After the expiry date, the passport cannot be used. The person needs to apply for another passport.

A driving license is a certificate that proves a driver's identity. It contains details such as name, address, issue date, expiry date, photograph and signature, seal. A driving licence is issued by a state's transport authority.

A degree certificate is a proof that a person has undergone and passed a degree. It contains the details of the person, official seal and signature of vice chancellor. It is issued by an approved university. On examining the degree certificate, one can tell that the person has passed a degree conducted by a valid university.

4 X.509 Certificates

In a computer network, we use the term digital certificates. A digital certificate is an electronic document which conforms to the ITU's X.509 specification. A digtial certificate is issued from a certificate authority (CA). This mechanism uses public key encryption.

For example, communication through an https connection uses digital certificates (between a mail server and browser, between a bank computer server and browser etc..).

The use of a digital certificate ensures a browser that the web page is coming from a valid web server. The user names and passwords are sent to a valid server.

It is a document which typically contains the owner's name and public key, the expiration date of the public key, the serial number of the certificate, and the name and digital signature of the organization which issued the certificate. The digital certificate binds together the owner's name and a pair of electronic keys (a public key and a private key) that can be used to encrypt and sign documents.

All browsers (Firefox, Internet Explorer) have certificates from all known trustworthy CAs built into their browsers. If a certificate is sent from a web site to a browser, and if the browser does not know it, then a warning message is given to the user that the certificate is not valid. This is shown below:

Certificate Authority (CA)

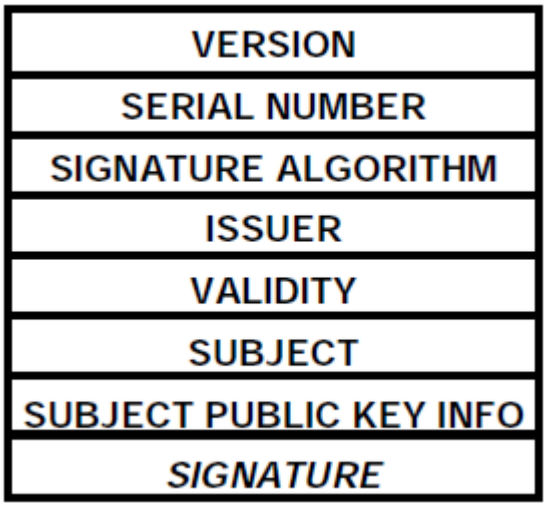
A number of CAs are present over the Internet. CAs issue digital certificates. Examples for CAs are VeriSign, User Trust Network etc..

A browser is already loaded with the certificates from many root CAs.

For example, in the Firefox web browser, see "Edit -> Preferences -> Advanced -> Encryption -> View Certificates -> Authorities

Structure of a Certificate

Following shows the structure of a X.509 digital certificate:



Following is an example of a digital certificate sent from a web server to a browser.

```
Certificate:
Data:
 Version: v3 (0x2)
 Serial Number: 3 (0x3)
 Signature Algorithm: PKCS #1 MD5 With RSA Encryption
 Issuer: OU=Ace Certificate Authority, O=Ace Industry, C=US
 Validity:
  Not Before: Fri Oct 17 18:36:25 1997
  Not  After: Sun Oct 17 18:36:25 1999
 Subject: CN=Jane Doe, OU=Finance, O=Ace Industry, C=US
 Subject Public Key Info:
  Algorithm: PKCS #1 RSA Encryption
  Public Key:
   Modulus:
    00:ca:fa:79:98:8f:19:f8:d7:de:e4:49:80:48:e6:2a:2a:86:
    ed:27:40:4d:86:b3:05:c0:01:bb:50:15:c9:de:dc:85:19:22:
    43:7d:45:6d:71:4e:17:3d:f0:36:4b:5b:7f:a8:51:a3:a1:00:
    98:ce:7f:47:50:2c:93:36:7c:01:6e:cb:89:06:41:72:b5:e9:
    73:49:38:76:ef:b6:8f:ac:49:bb:63:0f:9b:ff:16:2a:e3:0e:
    9d:3b:af:ce:9a:3e:48:65:de:96:61:d5:0a:11:2a:a2:80:b0:
    7d:d8:99:cb:0c:99:34:c9:ab:25:06:a8:31:ad:8c:4b:aa:54:
    91:f4:15
   Public Exponent: 65537 (0x10001)
  Extensions:
   Identifier: Certificate Type
    Critical: no
    Certified Usage:
     SSL Client
   Identifier: Authority Key Identifier
    Critical: no
    Key Identifier:
     f2:f2:06:59:90:18:47:51:f5:89:33:5a:31:7a:e6:5c:fb:36:
     26:c9
 Signature:
  Algorithm: PKCS #1 MD5 With RSA Encryption
  Signature:
   6d:23:af:f3:d3:b6:7a:df:90:df:cd:7e:18:6c:01:69:8e:54:65:fc:06:
   30:43:34:d1:63:1f:06:7d:c3:40:a8:2a:82:c1:a4:83:2a:fb:2e:8f:fb:
   f0:6d:ff:75:a3:78:f7:52:47:46:62:97:1d:d9:c6:11:0a:02:a2:e0:cc:
   2a:75:6c:8b:b6:9b:87:00:7d:7c:84:76:79:ba:f8:b4:d2:62:58:c3:c5:
   b6:c1:43:ac:63:44:42:fd:af:c8:0f:2f:38:85:6d:d6:59:e8:41:42:a5:
   4a:e5:26:38:ff:32:78:a1:38:f1:ed:dc:0d:31:d1:b0:6d:67:e9:46:a8:
   d:c4
```

The fields in an X.509 certificate are,

Version

Public-key certificates have gone through a development by defining new versions. The public-key certificate shown above is a so-called version-3 public-key certificate. The version shall be provided in the version field.

Serial number

Every public-key certificate issued by a CA must have a unique serial number placed in the serial number field. Every certificate issued by a CA has a serial number that is unique among the certificates issued by that CA.

Algorithm identifier

This component identifies the signature algorithm used by the issuer to construct the signature on the certificate. This is a redundant component, as the same information is available in the signature itself.

Issuer

This component hold the distinguished name of the CA that issued and signed the certificate.

Validity

Each certificate is valid only for a limited amount of time. This period is described by a start date and time and an end date and time, and can be as short as a few seconds or almost as long as a century. The validity period chosen depends on a number of factors, such as the strength of the private key used to sign the certificate or the amount one is willing to pay for a certificate. This is the expected period that entities can rely on the public value, if the associated private key has not been compromised.

Subject

This component holds the name of the entity for which the certificate is issued. It is an X.500 distinguished name. In principle this name has to be globally unique. However, there is no naming authority in place to ensure that. At least, the CA should ensure that it does not use the same name for different entities.

Public key information

The public key information (subjectPublicKeyInfo) component holds the public key associated with the subject entity. It holds information about the encryption algorithm for which this key is to be used.

Signature

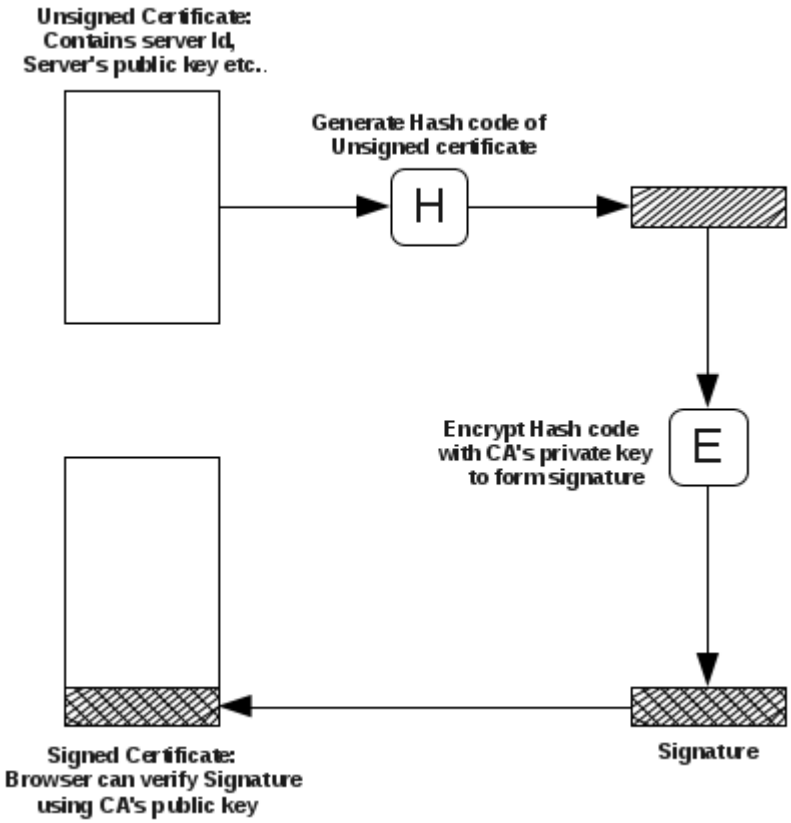
The signature section includes the following information:

The cryptographic algorithm, or cipher, used by the issuing CA to create its own digital signature.

The CA's digital signature, obtained by hashing all of the data in the certificate together and encrypting it with the CA's private key as shown below.

5 Creating a Digital Certificate

Consider the gmail server. A CA (for gmail, the CA is User Trust Network) had issued a digital certificate to the gmail server. The following figure shows how the CA created the digital certificate for gmail server.



In the beginning when gmail server came into operation (may be in 2004), gmail server presented its name, id to CA. Gmail server or CA creates a pair of public and private keys for the gmail server. The CA creates the certificate with details such as version, serial number, issuer, public key etc..

1. CA computes the hash code of this certificate.
2. This hash code is encrypted with CA's private key. This forms the signature of the certificate.
3. This signature is appended at the end of the certificate and it becomes a signed certificate.

Since CA has encrypted the hash code using its private key, decryption of the signature section of the certificate can be done using CA's public key only. If somebody changes the contents of the certificate, decrypted value will not match with the contents of the certificate.

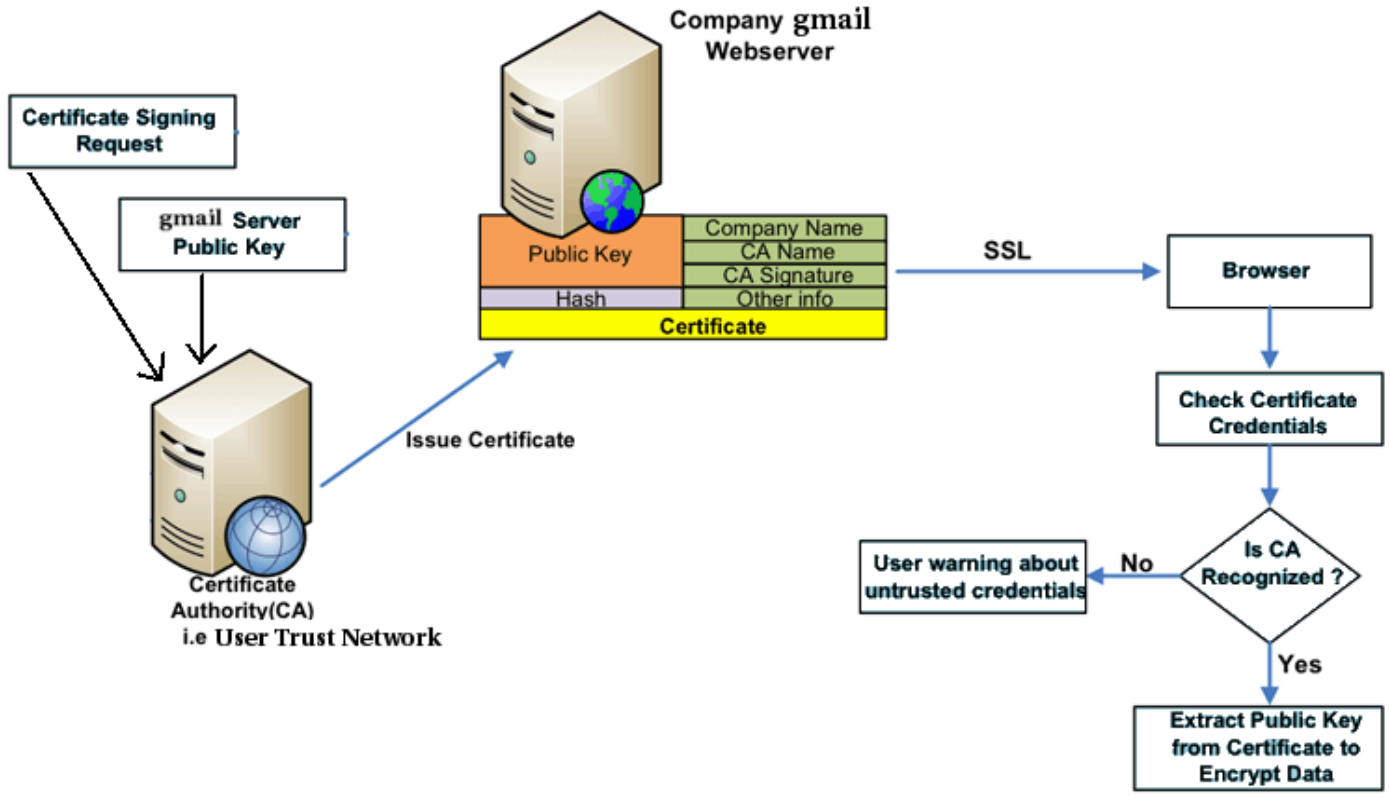
This digital certificate is sent from gmail server to a browser when a user want to access his mail.

When we install a browser in our computer, it stores the certiifcates from a number of CAs (namely the public keys of CAs).

When the browser gets the certiifcate from gmail server, it performs the following steps:

1. Browser computes the hash code of the certificate part.
2. The browser decrypts the signature part of the certificate using the public key of the CA. Then it gets a hash code.
3. If the hash code in step 1 matches with the hash code in step 2, it is assured that browser is communicating with the actual gmail server.

This ensures the browser that it is the actual gmail server which is in communication.



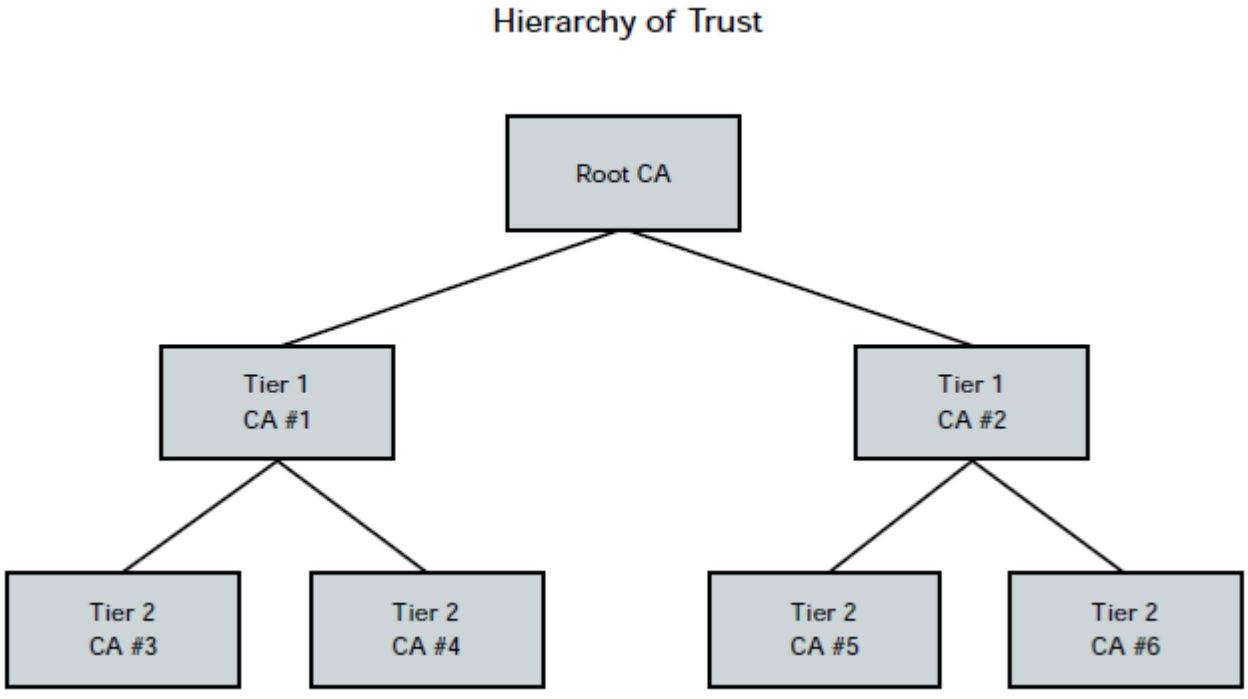
6 Certificate Authority (CA)

A number of CAs are present over the Internet. Examples for CAs are VeriSign, User Trust Network etc..

A browser is already loaded with the certificates from many root CAs.

For example, in the Firefox web browser, see "Edit -> Preferences -> Advanced -> Encryption -> View Certificates -> Authorities

CAs issue digital certificates for other CAs and for ordinary web sites.



Self Signed Certificates

Certain CAs are root CAs. They use their own certificate, where subject and issuer part of the certificate are identical. A root CA must be a well known CA.

Intermediate CA

A root CA issue certificates for other CAs. They are called intermediate CAs.

Thus root CA and intermediate CAs from a tree where the leaves of the tree are ordinary certificates.

A certificate chain is a sequence of certificates $C_1, C_2, C_3.....C_n,$

where

C_1 is a root CA certificate,

C_n is an ordinary certificate.

Holder of C_1 is the issuer of $C_2,$

Holder of C_2 is the issuer of $C_3,$

Holder of C_3 is the issuer of $C_4,$ and so on.

Holder of C_{n-1} is the issuer of $C_n.$

To validate C_3's certificate, public key from C_3 is used to to decrypt the signature of $C_2's$ certificate. Then public key from $C_2's$ certificate is used to decrypt signature of $C_1's$ certificate. If all of this work correctly, the ordinary certificate C_3 is authenticated.

7 Certificate Revocation

Certificates are sometimes problematic. For example:

The certificate was wrongly issued (because of information that was not known during the vetting process).

The certificate has become compromised for some reason (such as someone stealing the private key).

In these cases, the certificate should be invalidated. If the end of the certificate's valid period is far in the future, this need becomes critical.

A popular method of invalidating a signed certificate is described below.

Certificate Revocation List (CRL)

The most common method for revoking a certificate is through a Certificate Revocation List (CRL). This is a list that a CA creates listing all the certificates it has signed that it now wants to revoke. This list is made available to all browsers, and can be automatically downloaded from the CA's Web site. In theory, browsers download and check a CA's CRL before trusting a certificate presented to it. In practice, this does not always happen properly.

There are two problems with CRLs:

Downloading a long list of revoked certificates from a Web site is a slow process. This ultimately slows the Web browsing experience.

CAs only issue CRLs periodically. If a CA wants to revoke a certain certificate but the next CRL update is scheduled in another two weeks, this leaves a two-week window during which a bad certificate is still seen by browsers as trustworthy.

In reality, CRL usage is spotty at best, and many browsers do not properly implement this functionality due to the inherent problems. This creates vulnerability for all SSL users.

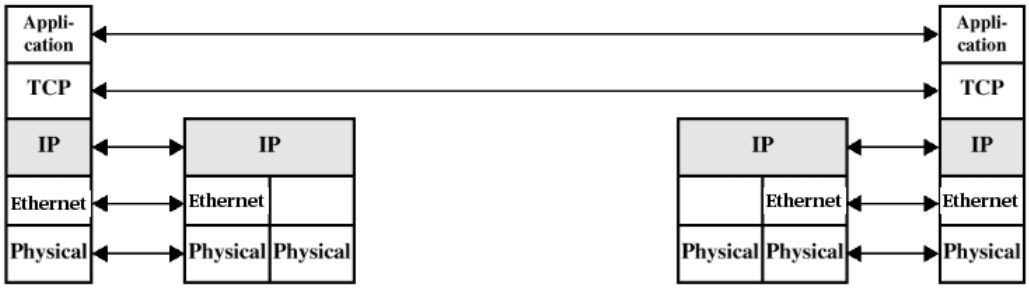
Part III. IP Security

IPsec (Internet Protocol Security) is a protocol suite for securing communication by authenticating and encrypting every IP packet of a communication session.

IPsec operates in the network layer of TCP/ IP protocol suite. It can be used in protecting data flow between 2 remote computers, between 2 routers, or between a router and a computer.

Network Layer without Security

Operation of IP which we already learned is shown below,



IP provides unreliable service. -

No guarantee that all data packets will be delivered.

Delivered packets may arrive in wrong order.

Higher layer (TCP) must recover from any errors.

Provides great deal of flexibility,

No reliability requirements of subnets.

Packets can follow different paths.

8 IPsec

IPsec consists of three aspects:

1. Authentication: insures that the received packet was transmitted by the party identified in the header.
2. Confidentiality: Enables communicating nodes to encrypt messages.
3. Key management: secure key exchange.

IPSec provides a set of security algorithms plus a general framework that allows a pair of communicating computers to use. The network can choose between algorithms to provide security appropriate for the communication.

IPSec can encrypt and authenticate all traffic at IP level.

Distributed applications (like remote login, e-mail, file transfers, web accesss etc.) can be secured.

Applications of IPSec

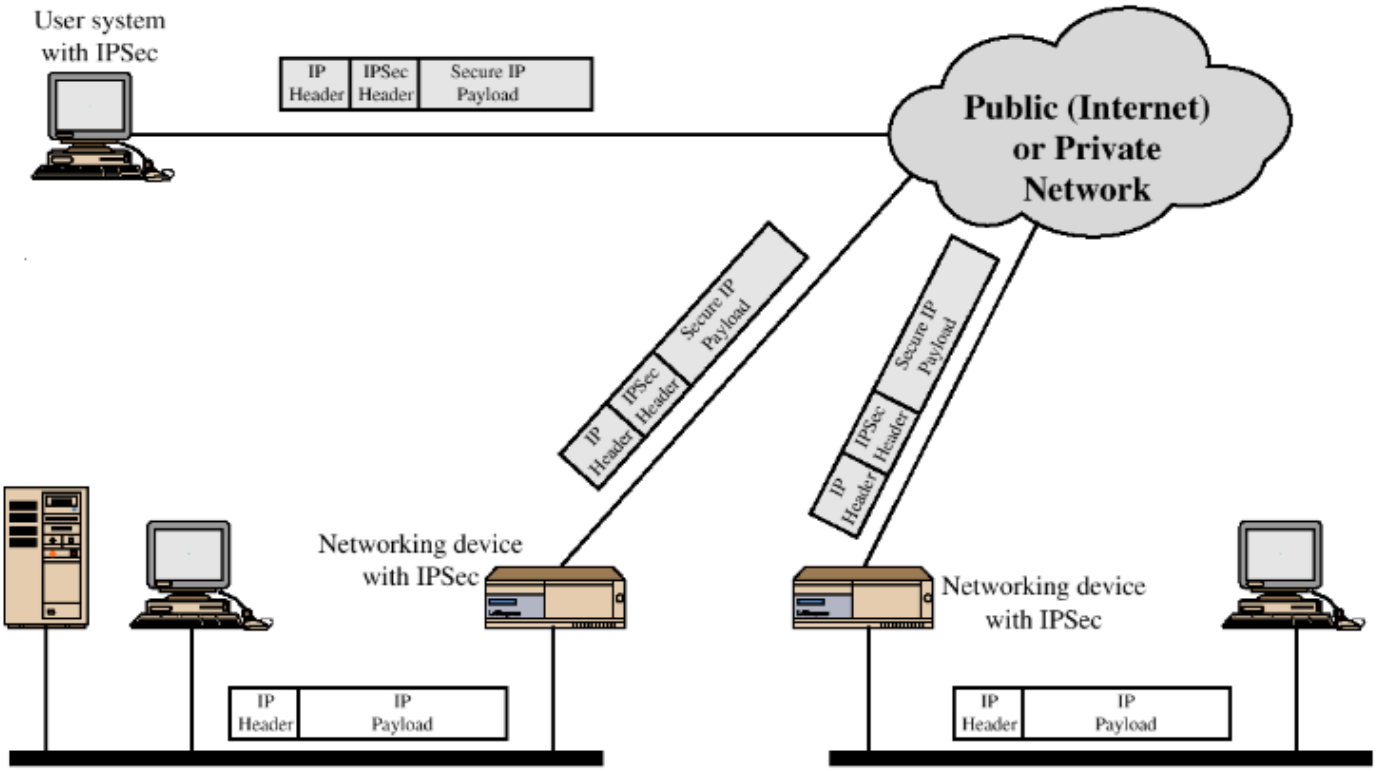
A company can build a secure private virtual network over the Internet.

An end user can gain secure access to other company's network over the Internet.

An Example Operation

IPSec protocols operate in networking devices that connect a LAN to Internet (like router or firewalls).

It encrypts all traffic leaving a LAN and decrypt traffic incoming to a LAN. IPSec operations are transparent to workstations and servers.



IPsec and Routers

IPSec plays an important role in routing. IPSec can assure that:

A router or neighbor advertisement (Hello packet) comes from an authorized router,

A redirect message (Echo packet) comes from the router to which the initial packet was sent.

A routing update is not forged.

9 IPsec Protocols

IPSec uses two protocols to provide security:

1. Authentication Header (AH): an authentication protocol.
2. Encapsulating Security Payload (ESP): a combined encryption and authentication protocol.

Authentication Header (AH)

AH provides support for:

1. Data integrity of a packet.

   Modification to packets while in transit are not possible.

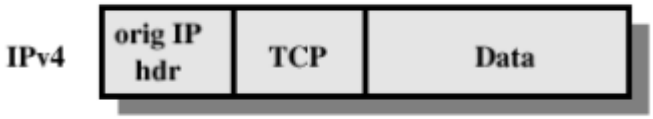
2. Authentication of a packet.

   End system can verify the sender.

   Prevents address spoofing attacks.

3. Also guards against replay attacks.

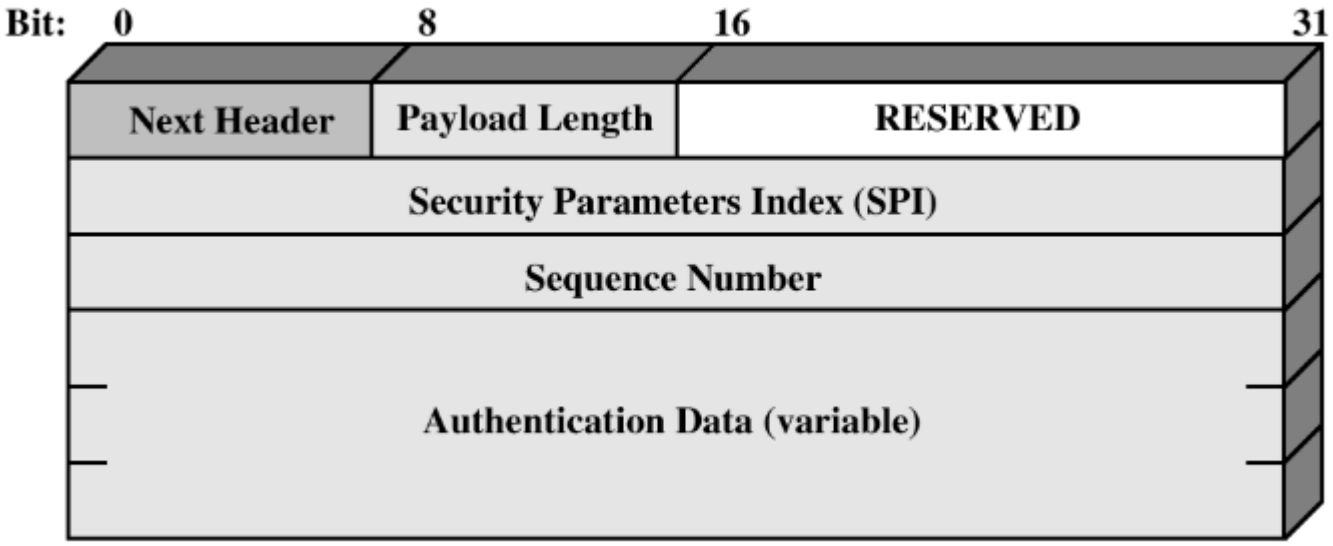
An IP packet before appying AH is shown below:



Here two end systems share a protected secret key for authentication.

Authentication header is inserted after the original IP header.

AH format



Next header (8): Identifies the type of header immediately following this.

Payload length (8): Length of the AH in 32-bit words minus 2.

Reserved (16): Reserved for future use.

Security Parameter Index (32): Identifies a security association.

Sequence number (32): A monotonically increasing counter value.

Authentication data (variable): Contains integrity check value (ICV) or MAC for this packet.

Encapsulating Security Payload (ESP)

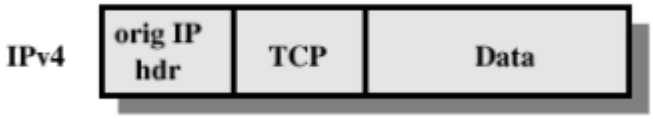
ESP provides support for,

Provides confidentiality services,

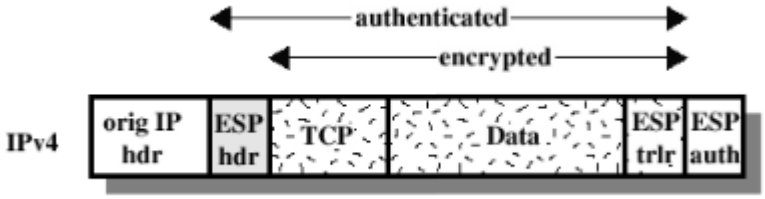
Provides limited authentication service,

Also provides limited traffic confidentiality.

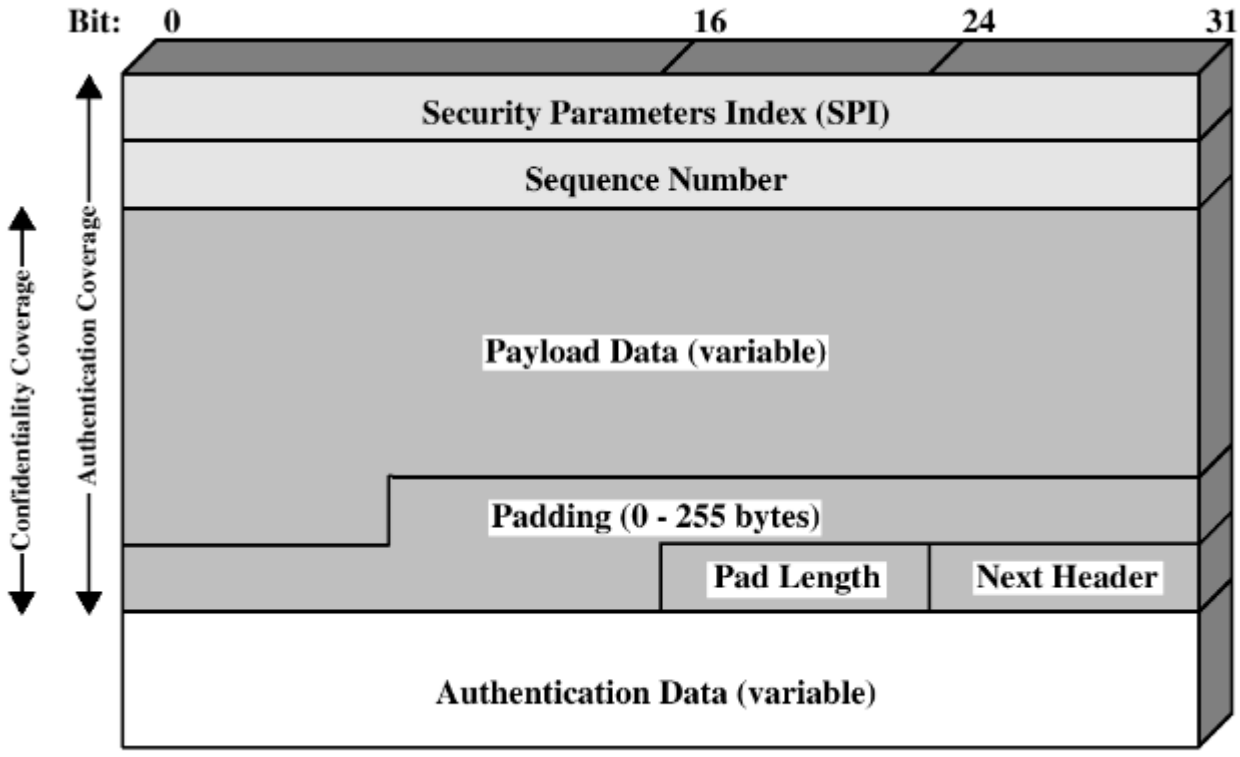
An IP packet before appying ESP is shown below:



An IP packet after appying ESP is shown below:



ESP Format



Security Parameter Index (32): Identifies a security association.

Sequence number (32): A monotonically increasing counter value.

Payload data (Variable): packet data.

Padding (0-255): Characters are padded in the packet to bring to some multiples of required number of bytes.

Pad length (8): The number of byte padded in this packet.

Next Header (8): Identifies the type of data contained in the payload data field.

Authentication data (variable): Contains the integrity check value of the packet.

Encryption Mechanism

Payload data, Padding, Padding length, and Next Header fields are encrypted using one of the following:

DES with CBC

Three-key triple DES

RC5

IDEA

Three-key triple IDEA

CAST

Blowfish

Authentication Mechanism

ESP supports the use of a MAC with default length of 96 bits. It supports: HMAC-MD5-96 and HMAC-SHA-1-96.

10 Working of IPsec

IPsec consists of two parts:

Part 1: Does Authentication and Key Management.

Consists of two phases:

Phase 1: Authentication and key agreement.

Phase 2: Setting up bulk encryption parameters.

Part 2: Provides Bulk data encryption, confidentiality, and message integrity.

Two parties exchange encrypted and authenticated messages.

Part 1: (Authentication and Key Management)

Phase 1: (Negotiate cryptographic parameters and secret keys).

Mainly consists of plain text messages.

Two parties agree on a shared DH secret and authenticate each other.

Also these parties construct IKE SA.

After Phase 1, two parties have set up an authenticated secret channel between them.

With the shared secret, they derive three keys: 1. an encryption key, 2. an authentication key, and 3. an additional secret value (for second set of secret keys).

Example:

Bob sends Alice his public DH value, a random value (RV), and a proposes cryptographic methods he wants to use in the phase 2 (DES, TripleDES, MD5, . . .). Alice uses Bob's public DH value, her private DH value, Bob's RV, and her RV to generate a shared secret key (SKEYID).

Alice sends to Bob her public DH value, her RV, and signed proof that she calculated SKEYID. Alice sends a signed hash of (SKEYID, Bob's RV, Alice's RV).

Bob uses Alice's public DH value, his private DH value, his RV, and Alice's RV to generate a shared secret key (SKEYID). Bob verifies the signed proof that Alice sent in the previous step and authenticates that she calculated the identical shared secret (SKEYID). Bob sends signed proof that he calculated SKEYID. Alice verifies the signed proof that Bob sent and authenticates that he calculated identical SKEYID.

Bob and Alice use the following:

SKEYID, a keyed hash function, and other values from phase 1 to compute three additional keys:

1. SKEYID_a: used as a HMAC secret key to authenticate all Phase 2 messages.
2. SKEYID_e: used to encrypt all Phase 2 messages.
3. SKEYID_d: used to derive the secret bulk encryption key used in Part 2.

Phase 2: (Setting up bulk encryption parameters.).

All phase 2 messages are protected by IKE SA.

They are encrypted with the encryption key and authenticated with the authentication key.

Example:

Bob sends Alice one or more SA proposals. He also sends a newly generated random number (Nonce) to prevent replay attacks. Alice responds that she agrees with one of the Bob's SA proposals and sends her newly generated random number.

Bob responds in a similar manner as Alice responded. Bob and Alice independently and simultaneously generate the secret key material, KEYMAT which they use in bulk encryption. (KEYMAT depends upon SKEYID_d).

Part 2: (Bulk data encryption, confidentiality, and message integrity)

Two parties exchange encrypted messages using the parameters and secret keys established in Phase 2.

Part IV. Secure Sockets Layer (SSL)

With SSL protocol, secure connections can be established between clients and servers. For example, between a web browser and a web server. SSL ensures that all data passed between a web server and browser remain private and integral. SSL was developed by Netscape in 1994.

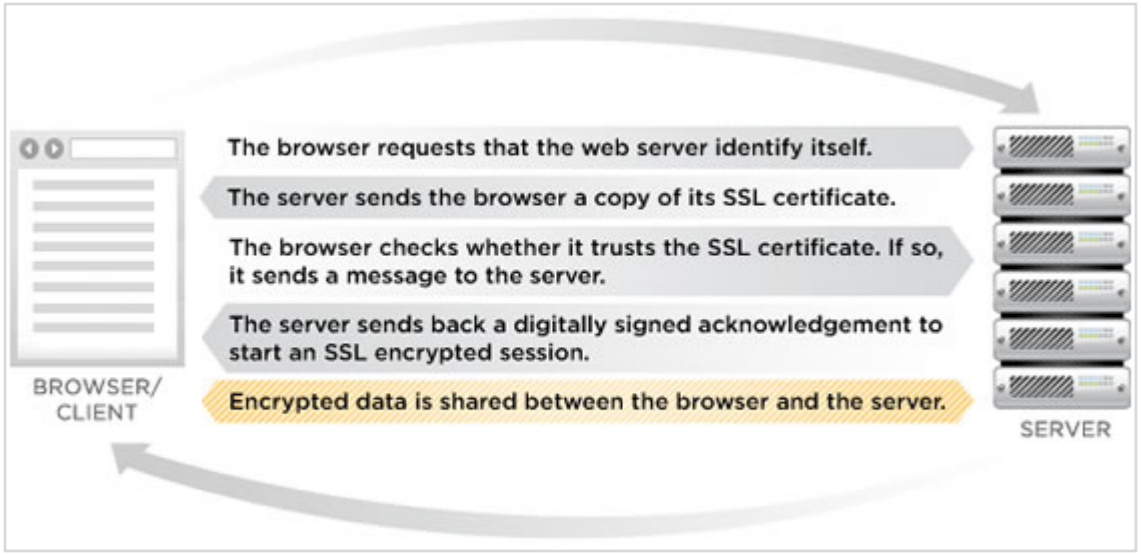
SSL is used by millions of web sites. Other applications such as Telnet, FTP can also use SSL protocol.

If a connection is using SSL, a user can know this by seeing a lock icon on the lower right hand corner of the browser. Clicking on this icon will display the SSL certificate. Also the address of the website starts with 'https' instead of 'http'. This assures that the communication is secure and all data sent over the connection are encrypted.

The above features can be seen on the web browser, when we access e mails (using gmail, yahoomail) or when accessing bank accounts online.

11 Overview of SSL

Following explains on overview of how SSL operates:



When a browser connects to a web server in which SSL is enabled, SSL begins to work.

Server sends its SSL certificate to the browser. When browser gets this certificate, it checks for the validity of the certificate. If the certificate is valid, browser generates a random number to use as secret key. this secret key is encrypted using the public key present in the SSL certificate and send it to web server. Web server decrypts the encrypted value using its private key and communication starts. All data sent between server and browser are encrypted using this secret key.

Following explains a detailed explanation of how SSL operates:

SSL Certificate Creation

A person running the web server such as gmail performs the following steps:

1. Generate a key pair, a public key and a private key. Place the public key in a certificate. Other details such as website domain name, address are added in the certificate.

2. Send this certificate to a CA such as VeriSign, Mountain View, California, GeoTrust Inc.

3. CA verifies the certificate and confirms the identity of certificate through outside means and verifies the sender's authority to own the certificate.

4. After verification, CA adds its signature to the certificate. For this, CA uses its private key to encrypt information in to the certificate. CA's private key is highly guarded and nobody except CA knows its private key.

5. The above signed certificate is the SSL certificate and it is sent to the person running the web site.

If a web server such as gmail needs a secure connection, it requires an SSL certificate. For this the owner of the web server approaches a Certificate Authority (CA). The CA will ask a number of questions to confirm the identity of the webserver and the company.

After this, web server creates two keys- a private key and a public key. The public key is placed in a Cerficate Signing Request (CSR) data file. CSR contains the details of the web server. This CSR is submitted to the CA.

Issuing SSL Certificate

CA will validate the the detials submitted in CSR and issues an SSL certificate containing the details of the web server. The web server matches the issued certificate with its private key.

Next, the web server can establish an encrypted link between its web site and a web browser.

An SSL certificate is an X.509 certificate that contains the web server's domain name, address, expiration date, issue date, details of CA which issued it, public key etc..

Web Browser - Web server Communication

When a browser connects to a secure web site of a web server, web server sends its SSL certificate to the browser. Browser checks and assures that certificate has not expired, it has been issued by a CA the browser trusts. Browser already has the public key of the CA which issued this certificate. Using CA's public key signature in the certificate is checked. If all are correct, certificate is validated.

If any one of these tests fails, browser will issue a warning to the user to inform him that the site is not secured by SSL.

Creating the Secret key

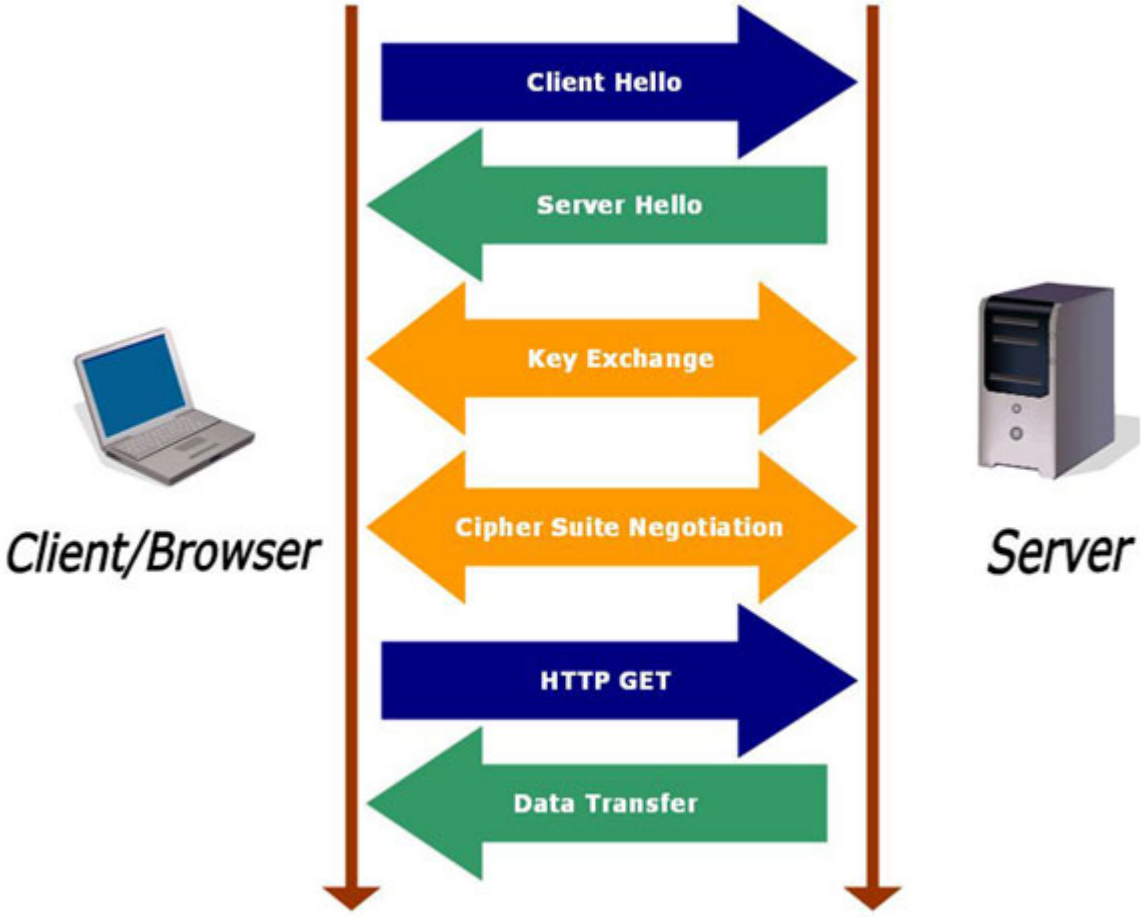
Next browser generates a secret key at random. Browser uses the public key in the SSL certificate to encrypt this secret key. This encrypted secret key is sent to the web server. Web server decrypts the encrypted value using its public key.

Now browser and web server have the secret key. Others do not know this key value.

Next data can be transferred between web server and browser. All these data are encrypted using the secret key.

Once the communication is over, web server and browser discard the secret key.

12 SSL in Detail



Suppose Alice wants to connect to a secure Web site to buy something online:

When Alice visits a Web site secured with SSL (typically indicated by a URL that begins with “https:”), her browser sends a “Client Hello” message to the Web server indicating that a secure session (SSL) is requested.

The Web server responds by sending Alice its server certificate (which includes its public key).

Alice’s browser will verify that the server’s certificate is valid and has been signed by a Certification Authority (CA) like Entrust, whose certificate is in the browser’s database or that has been cross certified by a root whose certificate is in the browser’s database (and who Alice trusts). It will also verify that the CA certificate has not expired.

If the certificate is valid, Alice’s browser will generate a one-time, unique “session” key and encrypt it with the server’s public key. Her browser will then send the encrypted session key to the server so that they will both have a copy.

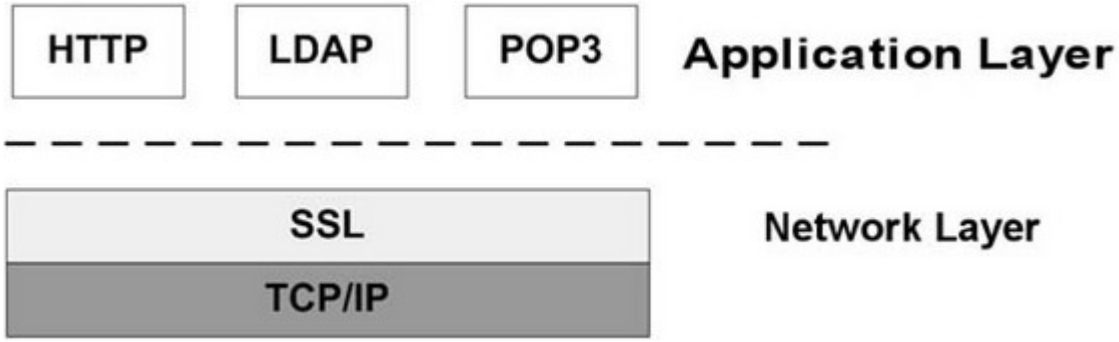
The server will decrypt the message using its private key and recover the session key. At this point Alice can be confident about two things:

The Web site she is communicating with has been vetted to confirm the identity of the organization requesting the certificate and the domain on which the server has been established.

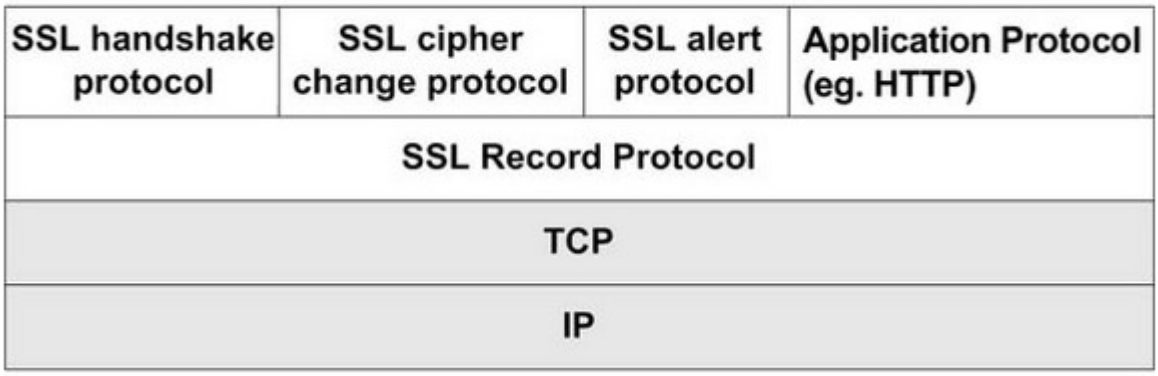
Only Alice’s browser and the Web server have a copy of the session key.

Once the SSL “handshake” is complete, then a secure communications “pipe” is established. Alice’s browser and the Web server can now use the session key to send encrypted information back and forth, knowing that their communications are protected. The entire process of establishing the SSL connection typically happens transparently to the user and takes only seconds.

SSL protocol lies above TCP in the transport layer as shown below:



SL protocol stack is shown below:



Establishing SSL Connections

SSL handshake protocol,

SSL cipher change protocol,

SSL alert protocol

are designed to establish an SSL session.

SSL Alert Protocol

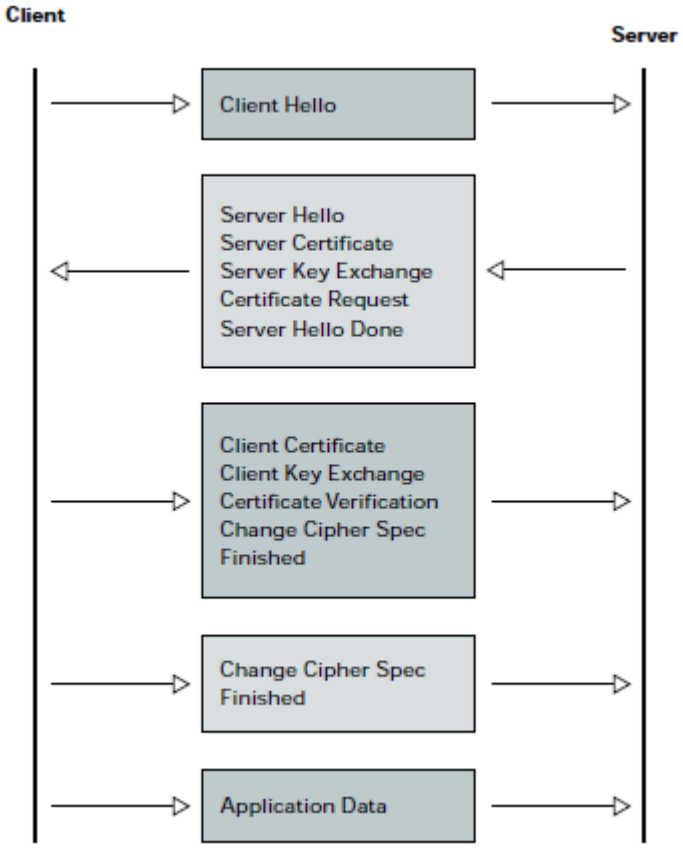
It is used to convey session messages associated with data exchange.

SSL Cipher change Protocol

It consists of a single message. This message is used to cause the pending session state to be established as a fixed state.

SSL Handshake Protocol

It is used to initiate a session between a client and a server.



Step 1:

Client sends the server a client_hello message. This message contains details such as method of key exchange such as Diffie Hellma algorithm, encryption algorithm such as DES or Triple DES, function for obtaining MAC value.

Step 2:

Server sends the client its SSL certificate for authorisation.

Step 3:

Client on getting the SSL certificate verifies the certificate. This is done as follows:

Client checks the issue and expiry dates presnt in the certificate.

Client checks whether the CA in the certificate is included in its trusted CAs. If the client does not have the CA name, it sends an error message. If CA is presnt in client's list, client validates the certificate as follows:

Client has the public key of CA. Using it, client decrypts the signature in the certificate and hash code is found. Client computes the hash code in the certificate. These hash codes are matched. If they are same, the certiifcate is a valid one.

Cleint checks whether the website domain name in the certificate matches with the domain name it requested.

On successful completion, server is authenticated.

Step 4;

After successful verification of server, client sends client_key_exchange message. In this, client specifies the method of exchanging the secret key such as Diffie hellman protocol or others.

Step 5:

Client sends a change_cipher_spec message and sets up algorithm parameters and keys.

Server on getting this replies with the same message.

At this stage, TCP session between client and server is closed, but a session state is maintained allowing it to resume communication using the already established parameters.

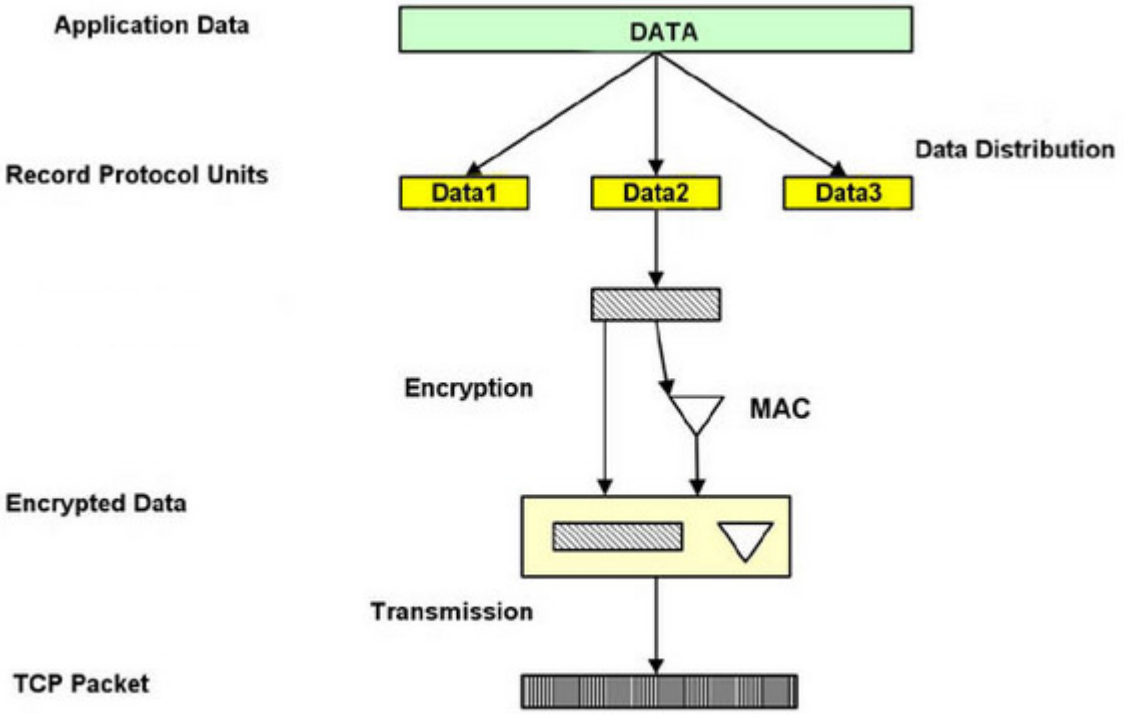
SSL Record Protocol

This layer performs

fragmentation,

adding headers and

Encryption.



Creating a packet under SSL record protocol

The layer takes an application message and fragments it into 16Kb data fragments.

Every fragment is added a header . The header contains the length of the data record, MAC value.

Next the resulting block is encrypted using DES or Triple DES.

The encrypted data block is added with an SSL header. It has the following fields:

Content type: It identifies the type of data in the block. Possible values are change_cipher_spec, alert, handshake and application_data.

Major version:

Minor version:

The resulting record is given to TCP.

Part V. Electronic Mail Security

To protect email messages, two schemes are in widespread use. They are,

PGP (Pretty Good Privacy), and

S/MIME (Secure/ Multipurpose Internet Mail Extensions).

13 Pretty Good Privacy (PGP)

PGP is a software application which allows people to exchange messages and files with high security. It was developed by Phil Zimmerman.

13.1 PGP Modes

PGP consists of the following modes:

1. Authentication

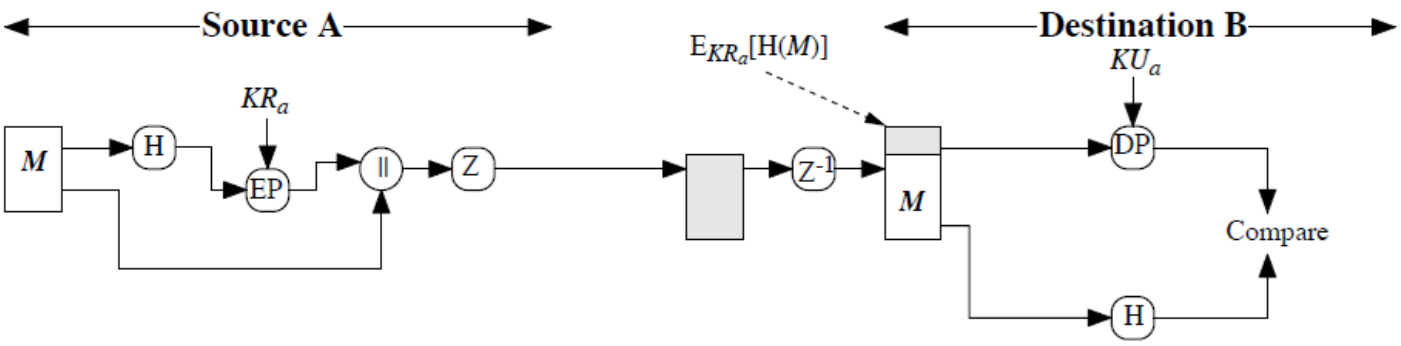
   It checks whether the sender is the actual one by adding a digital signature.

2. Confidentiality

   It protects the message using encryption.

3. Authentication and Confidentiality

Authentication



At the source, A,

Here a hash code, H of the actual message, M is created. This hash code, is H

Hash function used is SHA-1. It creates a 160 bit hash code, H.

H is encrypted using sender's private key, KR_a. This encrypted code, EP acts as the digital signature.

Encryption is done using RSA or DSS.

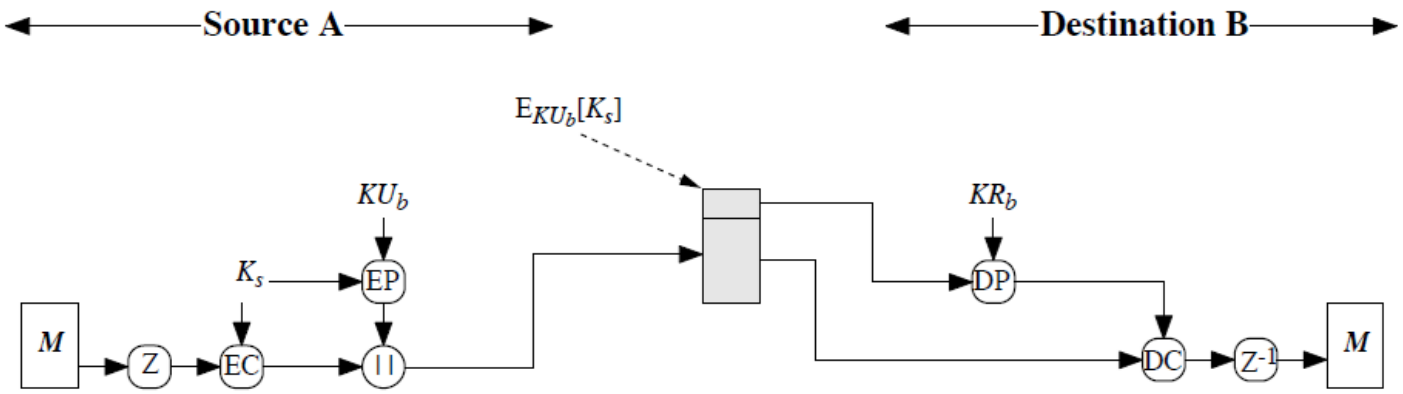
At the destination, B,

B decrypts the encrypted hash code, EP using A's public key, KU_a. The result is DP.

B computes the hash code of the message, M. This hash code is H

H is compared with DP. If they match, message is assumed as authentic.

Confidentiality



At the source, A,

A compresses the message, M. The result is Z.

compression is done using the compression algorithm, ZIP.

Z is encrypted using a randomly generated secret key, K_s. The result is EC.

Encryption is done using CAST-128, IDEA or 3DES.

Secret key, K_s is encrypted using B's public key, KU_b. The result is EP.

Here encryption is done using RSA.

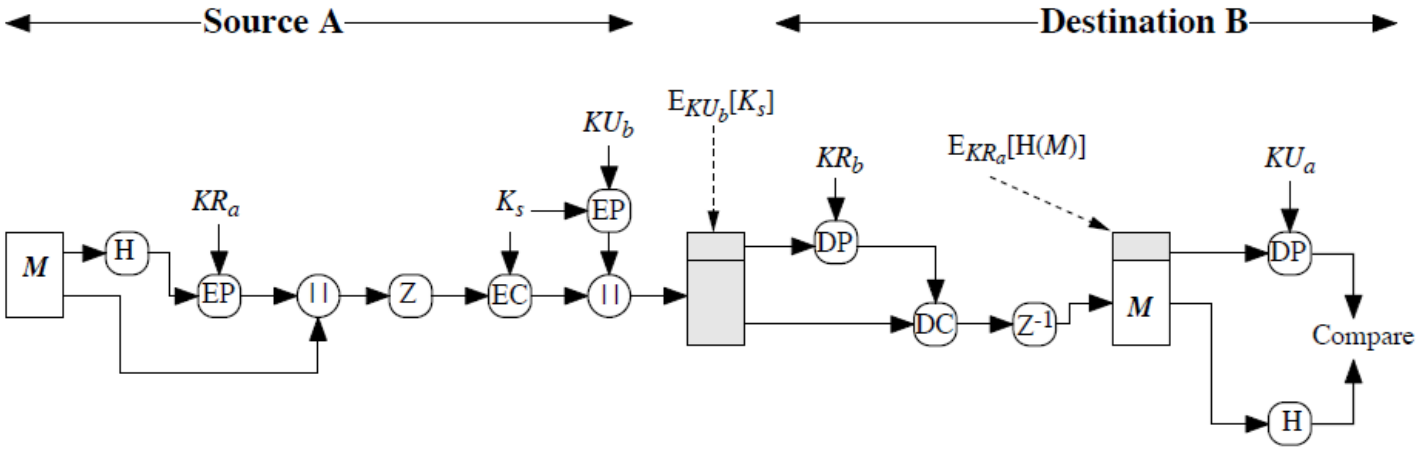
EC and EP are sent to the destination, B.

At destination, B,

Secret key, K_s is found by decrypting EP using B's private key, KR_b. The result is DP.

Using the secret key, DP original message, M is produced by decrypting EC.

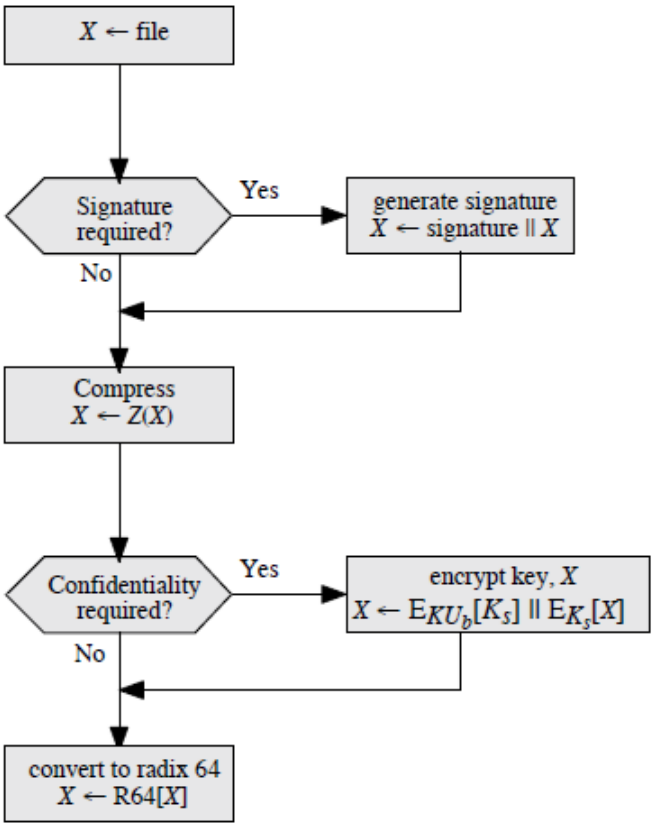
Confidentiality and Authentication



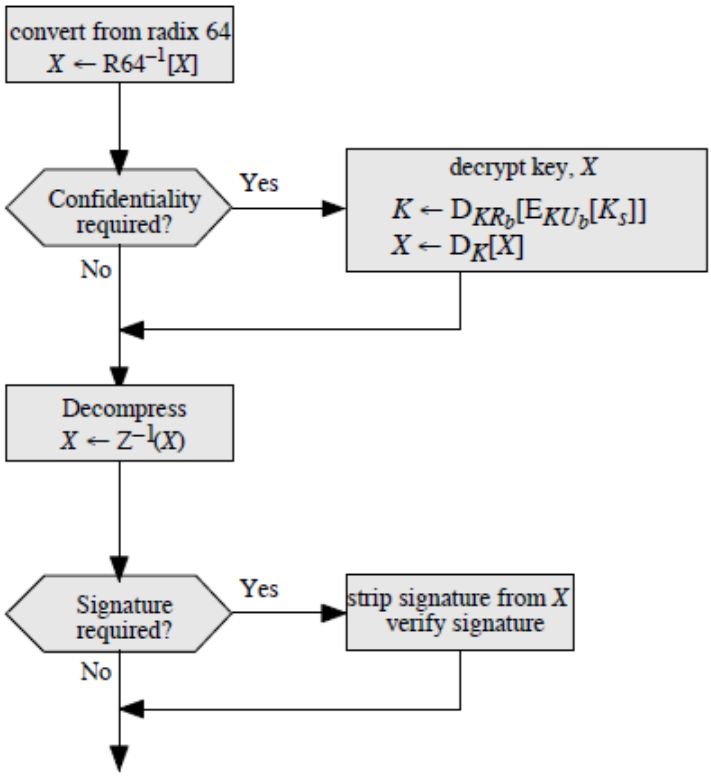
Above two processes are combined and both authentication and confidentiality can be achieved.

1. A signature, EP is generated and added to the message.
2. The result is compresed.
3. The compressed result is encrypted using CAST-128, IDEA or 3DES and the secret session key is encrypted.

Transmission and Reception of PGP messages

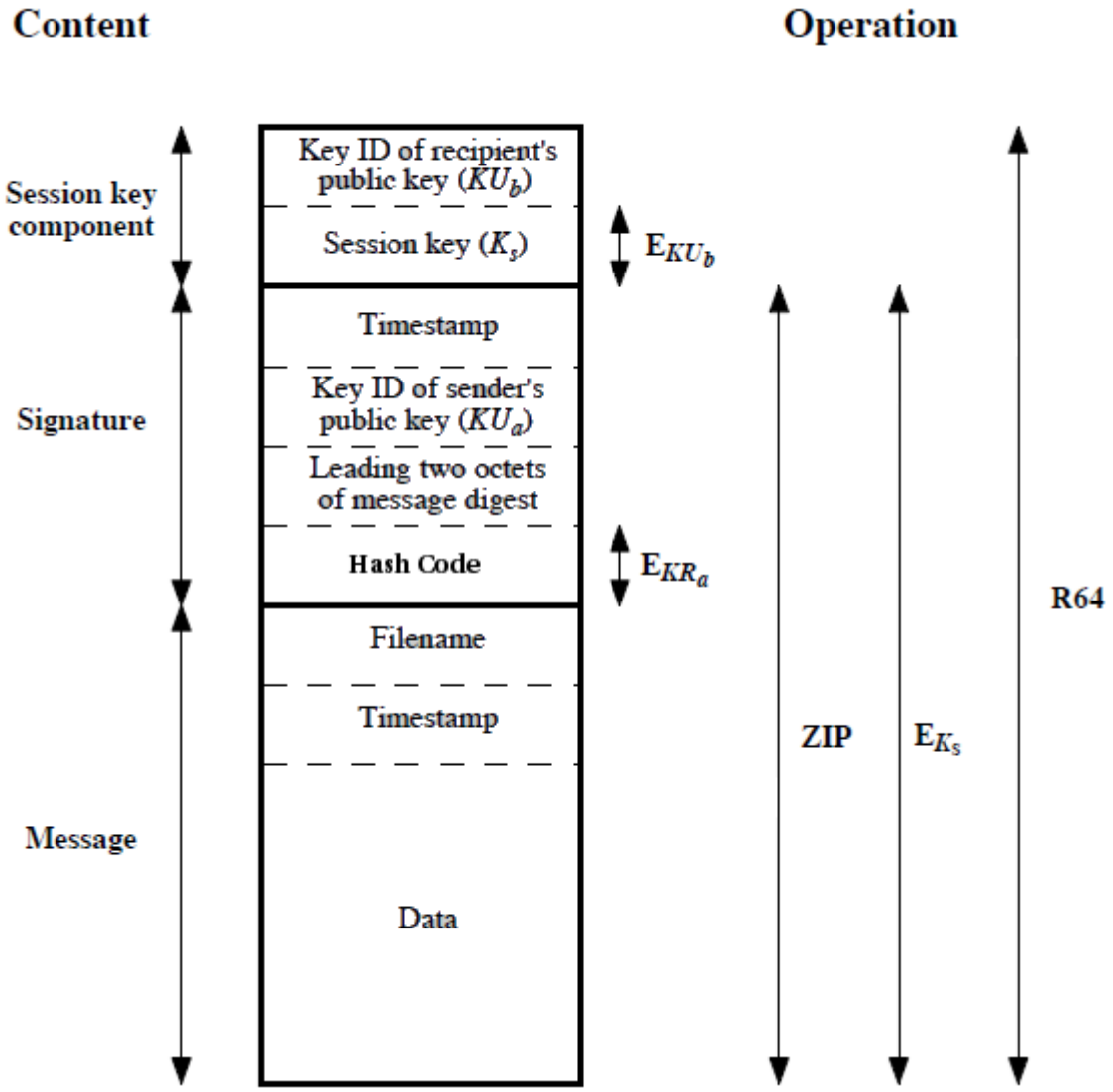


Generic Transmission Diagram (from A)



Generic Reception Diagram (to B)

Format of PGP Messages



14 S/MIME

S/MIME (Secure/ Multipurpose Internet Mail Extension) is an enhancement to the MIME standard. S/ MIME provides security to MIME.

We have already learned MIME (S6 CS Computer Networks). Learn us look at MIME once again.

14.1 MIME (Multipurpose Internet Mail Extensions)

This is a standard for the format of the mail message. When Bob sends an email message to Alice, he may include information such as Alice's address, his own address, subject and the date. These headers can be included in an email message. They are defined in MIME.

A typical message header looks like this:

From: bob@yahoo.com

To: alice@gmail.com

Subject: searching for a job

After the message header, a blank line follows, and then the message body follows. Each header line contains a keyword followed by a colon followed by a value. In the above header, From is a keyword, it is followed by a colon and then the value, i.e., the email address of Bob.

For sending multimedia messages such as images, audio and video extra headers have been defined.

The 2 key MIME headers for supporting multimedia are

content type: header, and

content transfer encoding: header.

Example:

Let Bob wants to send a JPEG image to Alice. For this, Bob invokes his user agent (eg. Outlook Express) for email, specifies Alice's email address, specifies the subject of the message and inserts the JPEG image in to the message body of the message. When Bob finishes composing his message, he clicks on 'send'. Bob's user agent then generates a MIME message, which is as follows:

From: bob@yahoo.com

To: alice@gmail.com

Subject: Picture of a tiger

MIME-Version: 1.0

Content-Transfer-Encoding: base64

Content-Type: image/jpeg

Here Bob's user agent encoded the JPEG image using base64 encoding. This is one of the encoding schemes for converting to 7 bit ASCII format.

When Alice reads his mail with her user agent, her user agent also follows MIME standard. When Alice's user agent observes the Content-Transfer-encoding: base64 header line, it decodes the base64 encoded message body. The message includes a content-Type: image/jpeg header line; this indicates to the Alice's user agent that the message body should be JPEG decompressed. The MIME-Version header indicates the MIME version that is being used.

The use of Content-Type header

This header has the following format:

Content-Type: type/subtype; parameters

It is used to specify the nature of the data in the body of a MIME entity. Currently there are 7 top level types defined. We describe 3 of the sub types below.

1. text:

The text type is used to indicate to the recipient that the message body contains textual information. The following are the common type/ subtype pairs.

text / plain

text / html

2. image:

The image type is used to indicate to the receiving user agent that the message body is an image. The popular type / subtype pairs are

image / jpeg

image / gif

3. application:

It is often used for data that must be processed by an application before it is viewable by a user. For example

application / msword

When the receiving user agent sees this, it launches Microsoft Office word application.

4. multipart

One important MIME type is multipart type.

An email message can contain many objects such as text, images and so on. When a message contains more than one object, the message uses the header line

Content-Type : multipart / mixed.

This header line indicates to the receiver that the message contains multiple objects.

Example: Suppose Bob wants to send a message to Alice consisting of some text, followed by a JPEG image, followed by some text again. His user agent generates a message as below.

From : bob@yahoo.com

To: alice@gmail.com

Subject: Picture of tiger with details

Mime-Version: 1.0

Content-Type: multipart / mixed; boundary =ABCD

- - ABCD

Content-Type: text/plain

dear alice,

please find picture of a tiger

- - ABCD

Content-Transfer-Encoding: base64

Content-Type: image/jpeg

———base64 encoded data————————————-

————————image of tiger is encoded here————

————————————————————————————

———base64 encoded data————————————-

- - ABCD

Content-Type: text/plain

Let me know if you would like the tiger.

How are you doing Alice?

- - ABCD - -

Here the Content-type: line with the boundary parameter in the header indicates how the various parts in the message are separated. The separation always begins with 2 dashes.

14.2 S/MIME

S/MIME is a standard for protecting email messages. This is done by signing and encrypting MIME messages.

S/MIME adds the following content types:

multipart/signed,

application/pkcs7 etc..

to indicate that the email is signed, encrypted etc..

Once again, consider the following MIME mail:

From: "Bob Simpson" <bob@yahoo.com>

To: "Alice Fern" <alice@gmail.com>

Subject: Project submission

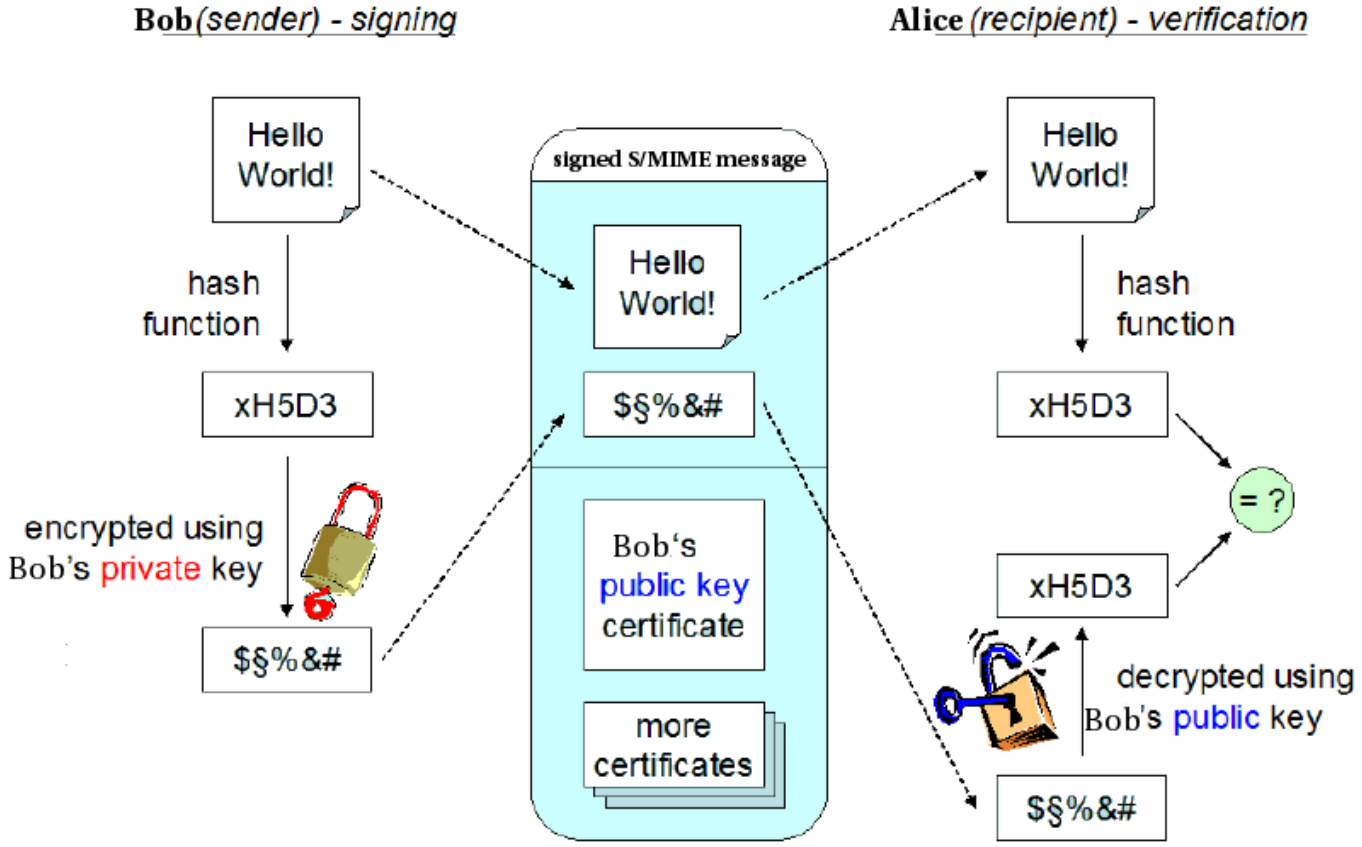
MIME-Version: 1.0

Content-Type: text/plain

Hello World!

S/MIME Signed Message

If S/MIME is used, it can sign the message as follows:



From the figure, the steps for signing a message are,

Bob write the message as clear text,

Hash code of the message is calculated (using SHA-1 or MD5),

This hash code is encrypted using Bob's private key.

Above MIME message is signed and the resulting S/MIME message is as follows:

From: "Bob Simpson" <bob@yahoo.com>

To: "Alice Fern" <alice@gmail.com>

Subject: S/MIME message

MIME-Version: 1.0

Content-Type: multipart/signed;

protocol="application/x-pkcs7-signature";

micalg=SHA1; boundary=ABCDE

- - ABCDE

Content-Type: text/plain

Hello World!

- - ABCDE

Content-Type: application/x-pkcs7-signature

Content-Transfer-Encoding: base64

MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMI ggJfoAMCA

QICDnW7AAAAAjGqNhNz5/78MA0GCSqGSIb3DQ MA4GA1UECBM

HSGFtYnVyZzEQMA4GA1UEBxMHSGFtYnVyZz [...]

- -ABCDE- -

Note the last part of the email message. That is,

- - ABCDE

Content-Type: application/x-pkcs7-signature

Content-Transfer-Encoding: base64

MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMI ggJfoAMCA

QICDnW7AAAAAjGqNhNz5/78MA0GCSqGSIb3DQ MA4GA1UECBM

HSGFtYnVyZzEQMA4GA1UEBxMHSGFtYnVyZz [...]

- -ABCDE- -

This part contains

sender's public key certificate,

algorithm identifier,

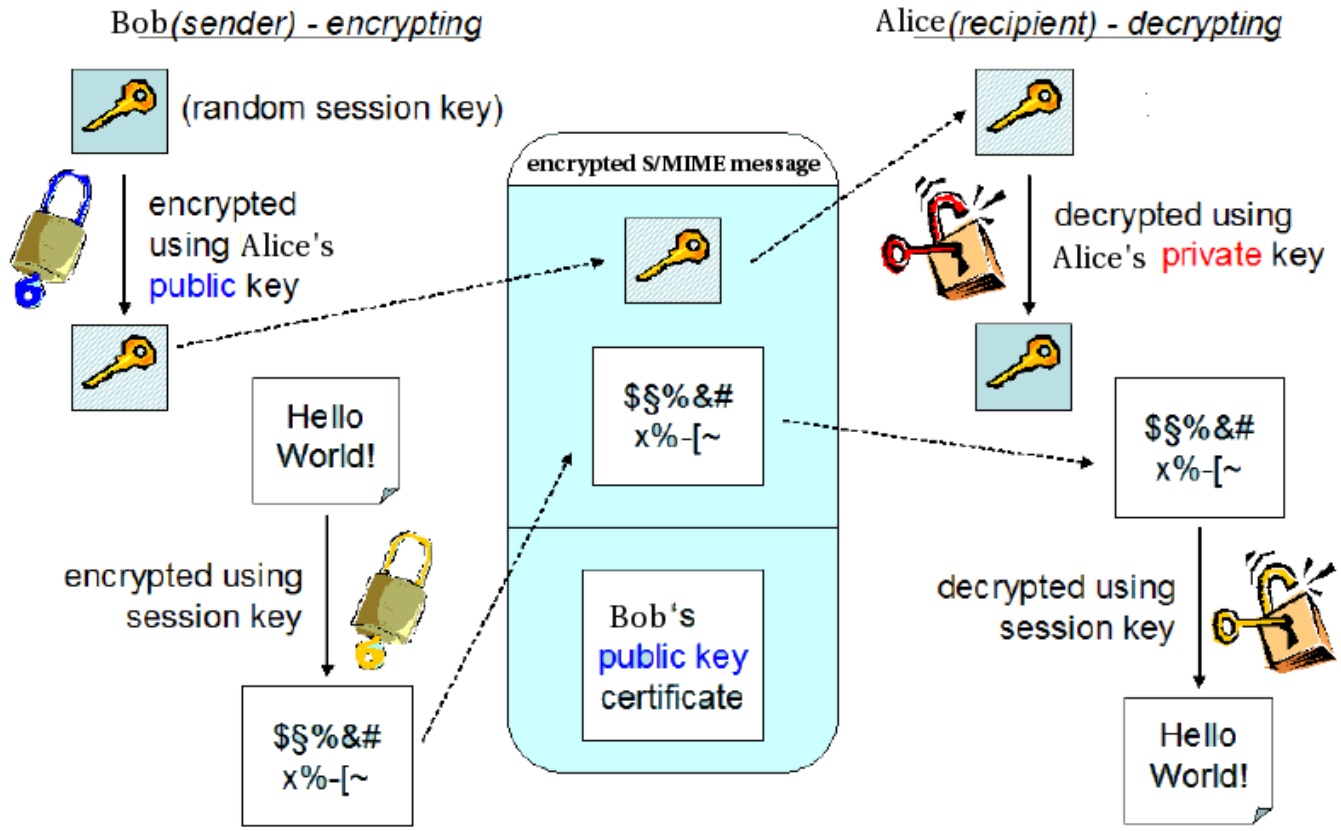
encrypted hash code of message.

Here, only digital signature is attached. Receiver can verify whether the message was being sent from the actual sender using the digital signature.

Also, message text is sent as plain text. It can be read by any mail client. But no modification can be done. This is because hash code of the new message will be different.

S/MIME Encrypted Message

S/MIME can encrypt the above email as follows:



The following steps are taken in order to create an encrypted message:

1. Bob writes the message as clear-text.
2. A random session key is being created at Bob (tripleDES or RC2)
3. The message is being encrypted using the random session key.
4. For every recipient (here only Alice), the session key is being encrypted using the recipient's public key (DH or RSA).

The resulting S/MIME message is shown below:

From: "Bob Simpson" <bob@yahoo.com>

To: "Alice Fern" <alice@gmail.com>

Subject: S/MIME message

MIME-Version: 1.0

Content-Type: application/x-pkcs7-mime;

smime-type=enveloped-data

Content-Transfer-Encoding: base64

MIAGCSqGSIb3DQEHA6CAMIACAQAxggJUMIIBJgIBADCBzz

BAgTB0hhbWJ1cmcxEDAOBgNVBAcTB0hhbWJ1cmcxOjA4B

g ciBTZWN1cml0eSBpbiBEYXRhIE5ldHdvcmtzIEdtYkgxIj [...]

This message contains

the (symmetrically) encrypted message text,

the (asymmetrically) encrypted secret key,

the algorithm identifier, and

the sender's public key.

Due to the message being encrypted, the message cannot be read by anyone but the intended recipient (i.e. the owner of the private key corresponding to the public key used to encrypt the session key). However, the encrypted message text can be replaced by another encrypted message text, because the public key used to encrypt the message text is usually available to everyone.

S/MIME Encrypted and Signed Message

An encrypted and signed message is first being signed and then encrypted. Therefore, an encrypted and signed message looks exactly like the above example of an encrypted message.

S/MIME can sign and encrypt the above email as follows:

From: "Bob Simpson" <bob@yahoo.com>

To: "Alice Fern" <alice@gmail.com>

Subject: S/MIME message

MIME-Version: 1.0

Content-Type: application/x-pkcs7-mime;

smime-type=enveloped-data

Content-Transfer-Encoding: base64

MIAGCSqGSIb3DQEHA6CAMIACAQAxggJUMIIBJgIBADCBzz

BAgTB0hhbWJ1cmcxEDAOBgNVBAcTB0hhbWJ1cmcxOjA4B

g ciBTZWN1cml0eSBpbiBEYXRhIE5ldHdvcmtzIEdtYkgxIj [...]

Being encrypted, the message can not be read by any unauthorized person.

Being signed, the message cannot be modified by an unauthorized person, because digital signature will be lost if modified.

Hash Functions

To compute the hash code, S/MIME uses,

SHA (Secure Hash Algorithm), or

MD5 (Message Digest Algorithm).

Digital Signature

To encrypt hash code, S/MIME uses

DSS (Digital Signature Standard), or

RSA (Rivest Shamir Adleman).

Content Encryption

To encrypt message content, S/MIME uses

Triple DES, or

RC2.

Key Encryption

To encrypt the secret session key, S/MIME uses

DH (Diffie Hellman) algorithm, or

RSA.

Part VI. Secure Electronic Transaction (SET)

SET is a standard protocol for securing credit card transactions over Internet.

SET was jointly developed by VISA and MasterCard. But VISA and MasterCard do not use this protocol now. Now they use 3D secure scheme.

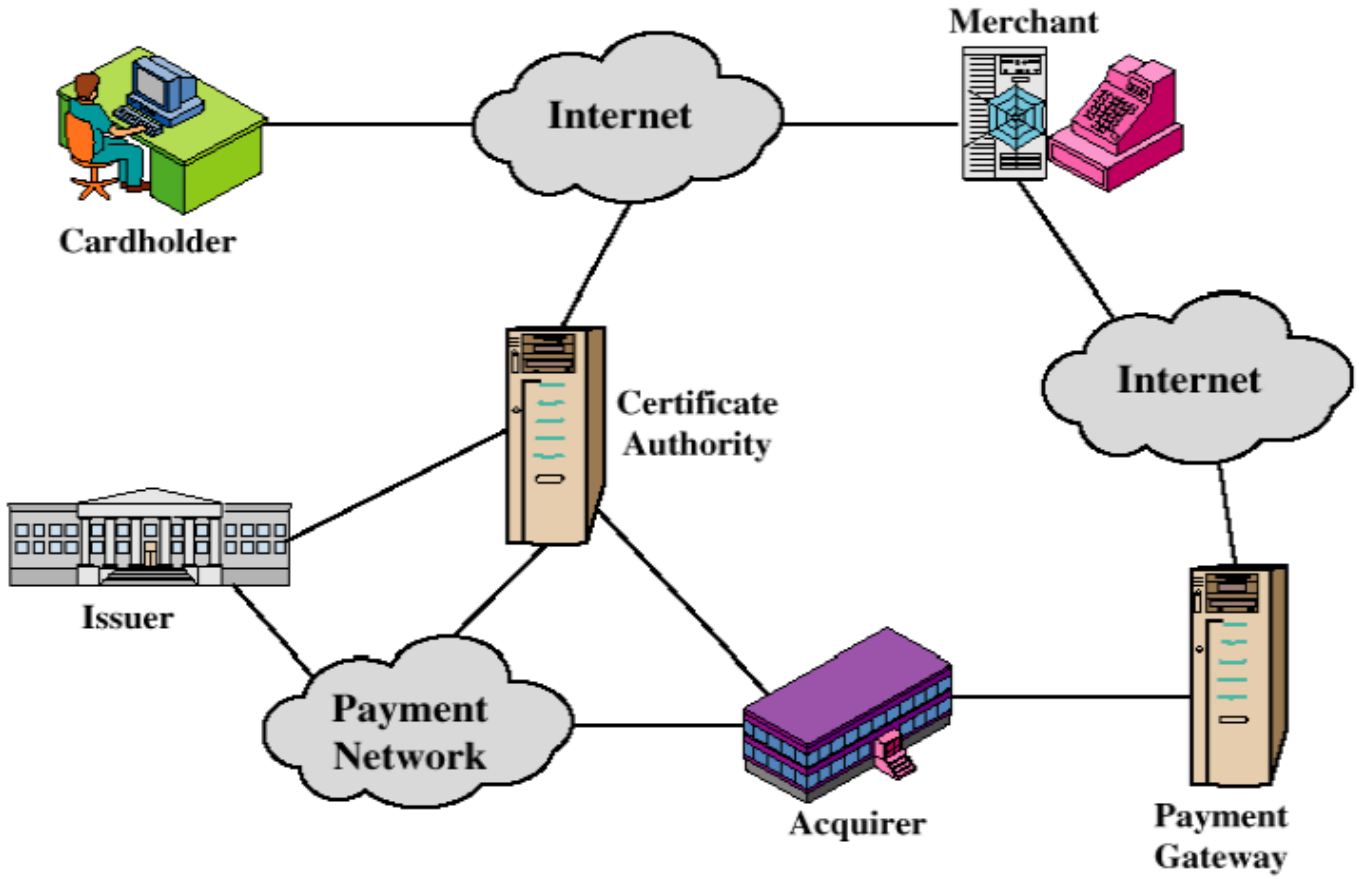
SET is based on cryptography. Consider the following example:

Alice wants to purchase a laptop displayed on the Amazon.com website. Amazon.com is an Internet retailer.

Alice uses her MasterCard and provides credit card number and details to the Amazon.com web site. Amazon validates her credit card details; money is withdrawn from her bank account. Then a laptop is delivered to her by post.

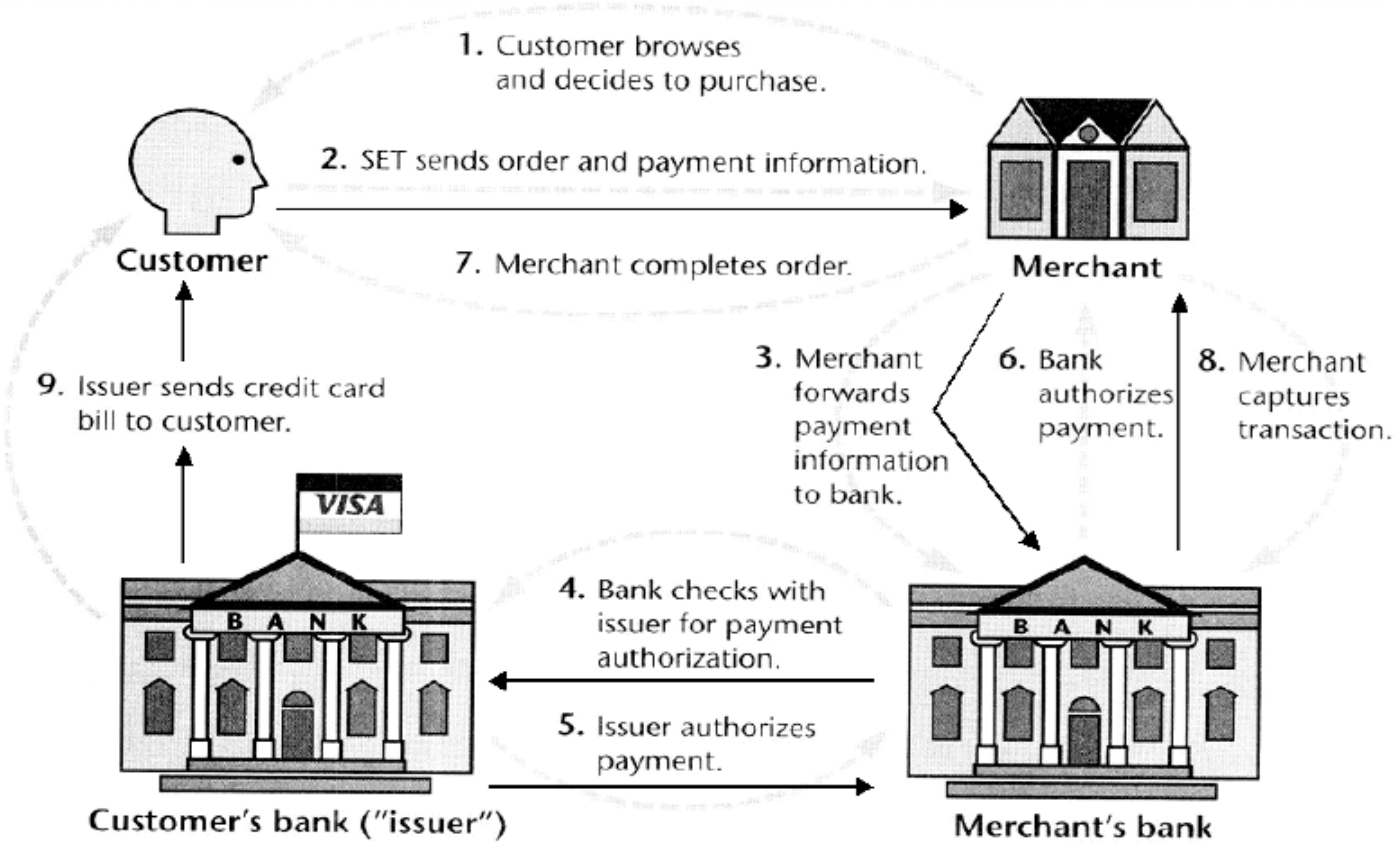
15 Participants In the SET system

Above operation is performed as shown in the following figure:



Transaction

The sequence of events required for such a transaction are as follows:



The customer opens an account with a card issuer. –MasterCard, Visa, etc.

•The customer receives a digital certificate signed by a bank.

•A merchant who accepts a certain brand of card must possess two digital certificates. –One for signing & one for key exchange

•The customer places an order for a product or service with a merchant.

•The merchant sends a copy of its certificate for verification.

•The customer sends order and payment information to the merchant.

•The merchant requests payment authorization from the payment gateway prior to shipment.

•The merchant confirms order to the customer.

•The merchant provides the goods or service to the customer.

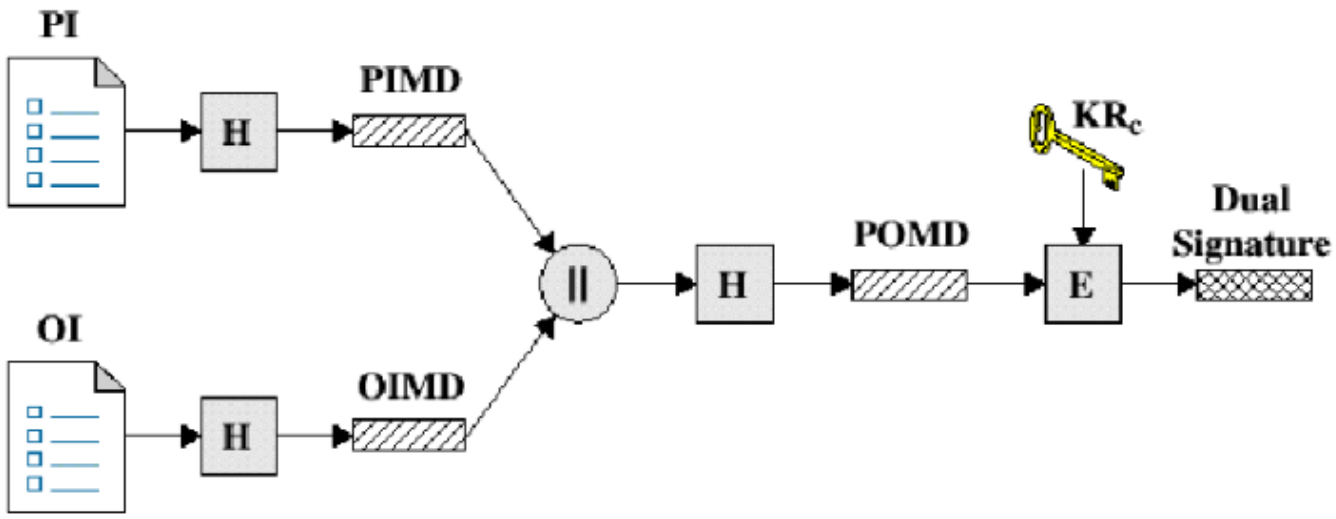
•The merchant requests payment from the payment gateway.

Dual Signature

Customer sends order information (OI) to the merchant and payment information (PI) to the bank.

Merchant does not need to know customer's credit card number; also bank does not need to know customer's order. Also some linking between OI and PI is needed to solve disputes.

For this purpose, digital signature (DS) is used.



Customer finds the hash code of PI and hash code of OI. These two hash codes are joined and the hash code of the result is taken.

Customer encrypts this hash code with his private key. The result is the dual signature (DS).

Thus,

$$DS = E_{KPR_C}[H(H(PI)||H(OI))]$$

Merchant gets DS, OI and PIMD. On getting these values, merchant can compute,

$D_{KPUB_C}(DS)$, and

$H(PIMD||H[OI])$

If these are equal, merchant has verified the signature.

Bankt gets DS, PI and OIMD. On getting these values, bank can compute,

$D_{KPUB_C}(DS)$, and

$H(OIMD||H[PI])$

If these are equal, bank has verified the signature.

Purchase Request

Customer browses the webiste of a merchant (eg. Amazon.com). He selects an item and ordering is done.

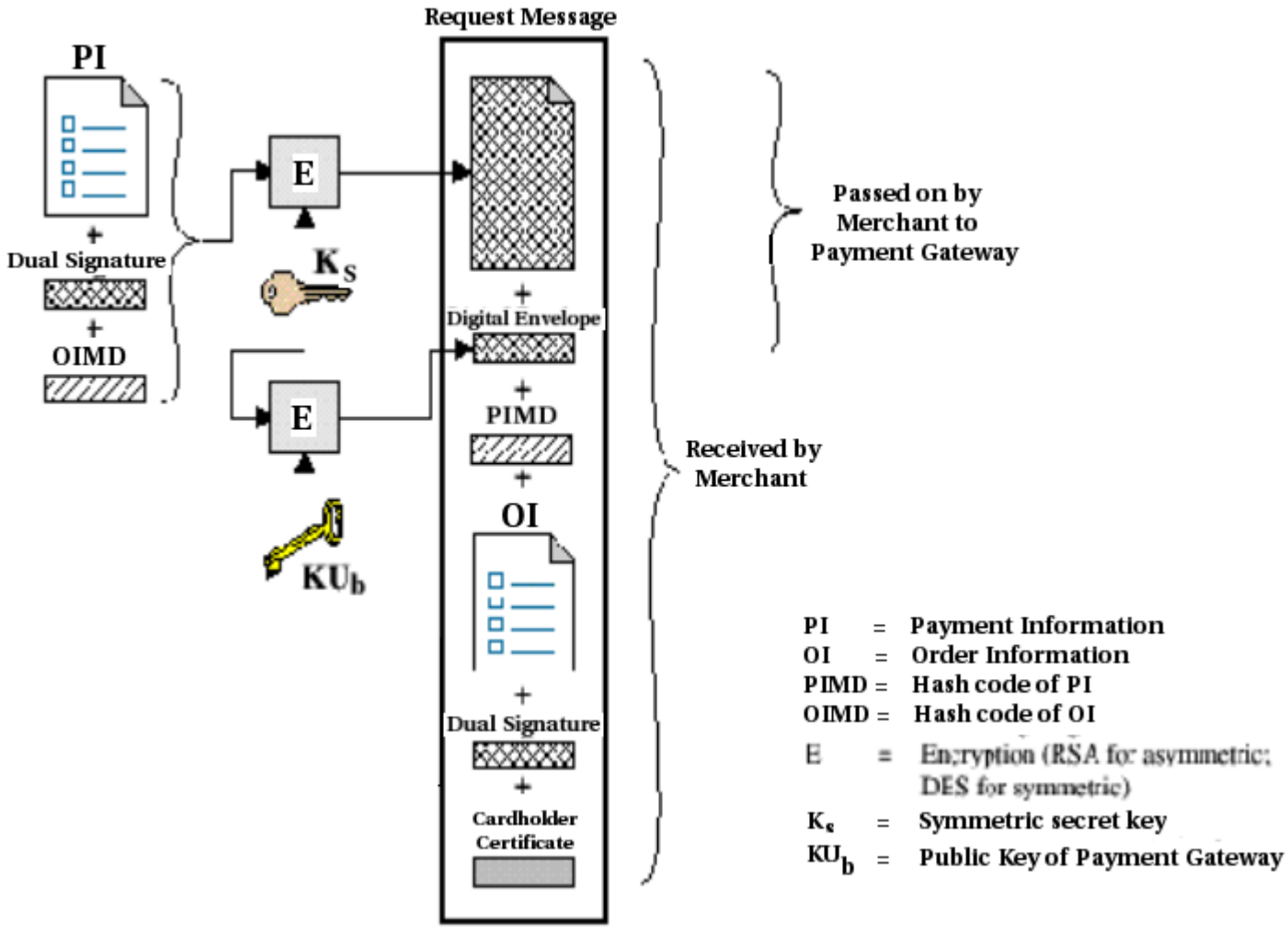
Purchasing Involves 4 Messages:

Initiate Request

Initiate Response

Purchase Request

Purchase Response



Following information is sent from customer to payment gateway through the merchant:

Customer generates a secret key, K_S.

Customer joins together PI, DA and OIMD.

The result is encrypted using a symmetric secret key, K_S.

Customer encrypts the secret key, K_S with the public key, KU_b of payment gateway. The result is the digital envelope.

Customer sends these two items to the merchant. Merchant sends these to the payment gateway.

Following information is sent from customer to merchant:

OI,

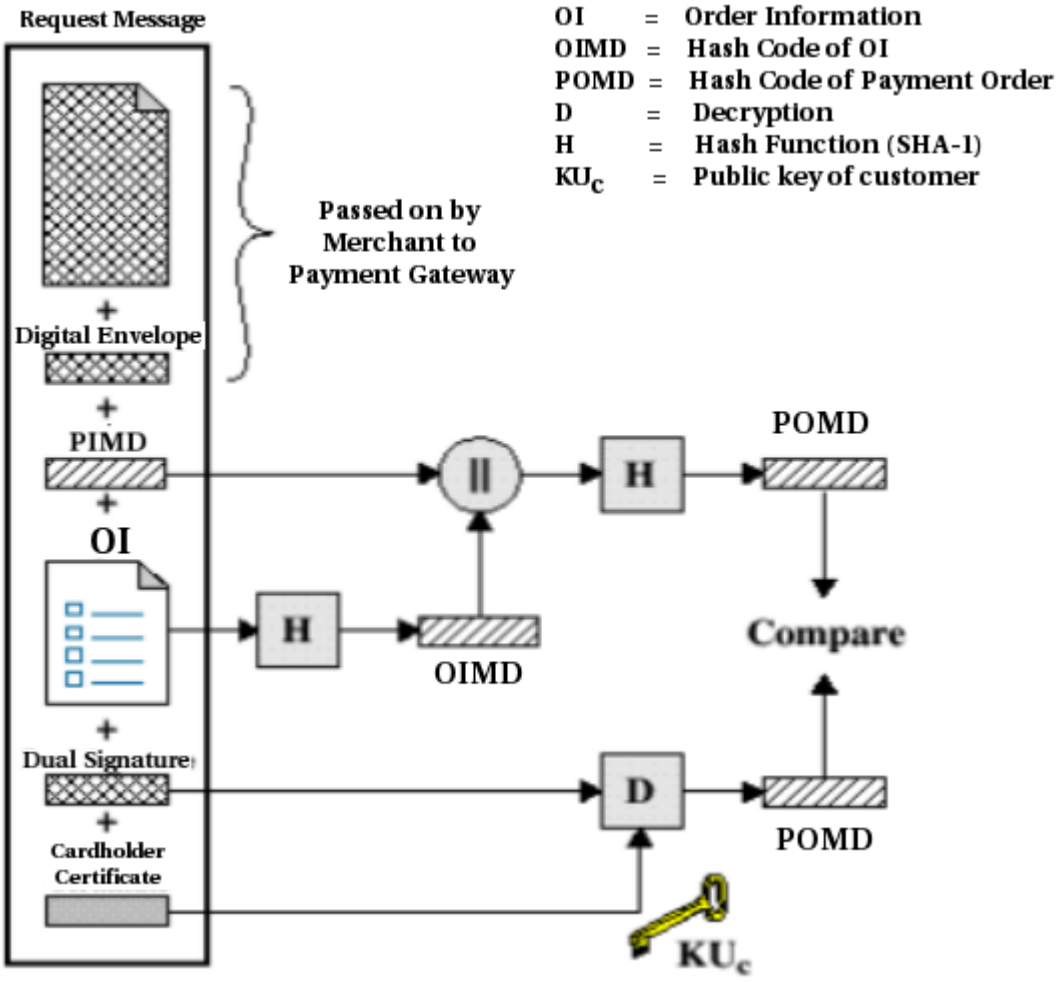
DS,

PIMD, and

Cardholder certificate.

Cardholder certificate contains the public key of customer. It is needed by merchant and payment gateway.

When the merchant receives the Purchase Request message, it performs the following actions:



Verify the cardholder certificates by means of its CA signatures,

Verifies the dual signature using the customer's public key signature.

Processes the order and forwards the payment information to the payment gateway for authorization.

Sends a purchase response to the cardholder.

Purchase Response

Purchase response message acknowledges the Order and References Corresponding Transaction Number.

Response response message is signed by Merchant using its Private Key. Message and signature are sent to customer along with Merchant's Signature Certificate. Upon reception customer verifies Merchant's certificate, signature on message.

Payment Process

The payment process is broken down into two steps:

Payment authorization,

Payment capture.

Payment Authorisation

The merchant sends an authorization request message to the payment gateway consisting of the following:

Purchase-related information,

PI,

Dual signature,

OIMD,

The digital envelope,

Authorization-related information,

Certificates

Cardholder's signature key certificate

Merchant's signature key certificate

Merchant's key exchange certificate.

On getting the above, payment gateway performs the following steps:

Verify all Certificates,

Verify Merchant signature on authorization message,

Decrypt payment message Digital Envelope to obtain symmetric key and decrypt message,

Verify dual signature on Payment message,

Verify whether the received transaction ID received from Merchant matches PI received from customer,

Request and receive issuer authorization.

With this authorisation from the payment gateway, merchant can provide goods to the customer.

Payment Capture

Merchant needs to get the payment. For this, it performs a capture transaction with payment gateway.

Payment capture consists of,

capture request,

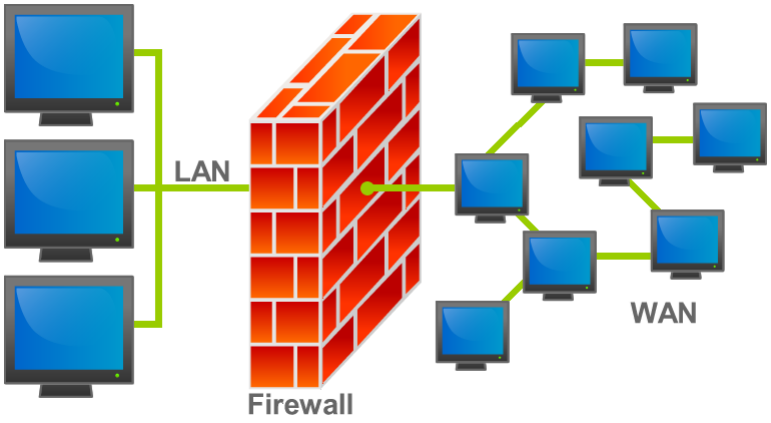
capture response.

Capture Request Here merchant provides all the details to the payment gateway. Payment gateway checks for consistency and checks all the details given. Then a request is sent to issuer. Then issuer transfers funds to the merchant's account.

Capture Response After funds are transferred, payment gateway notifies the merchant of payment.

Part VII. Firewalls

Firewall is a means of protecting a local system or a network from security threats. A firewall forms a barrier or a wall through which the traffic going in each direction must pass.



A firewall is a device or software designed to permit or deny network transmissions based upon a set of rules and is frequently used to protect networks from unauthorized access.

Many operating systems include software-based firewalls to protect against threats from the public Internet. Many routers that pass data between networks contain firewall components.

16 Firewall Characteristics

1. All traffic from inside and outside of a network must pass through firewall.
2. Only authorised traffic is allowed to pass.
3. The firewall itself is immune to penetration.

17 Capabilities of Firewalls

1. A firewall keeps unauthorised users out of the protected network, prohibits vulnerable services from reaching the network, and protects from various IP spoofing and routing attacks.
2. Firewall provides monitoring security related events. Audits and alrarms can be implemented.

18 Types of Firewalls

Three types of firewalls are,

Packet filters,

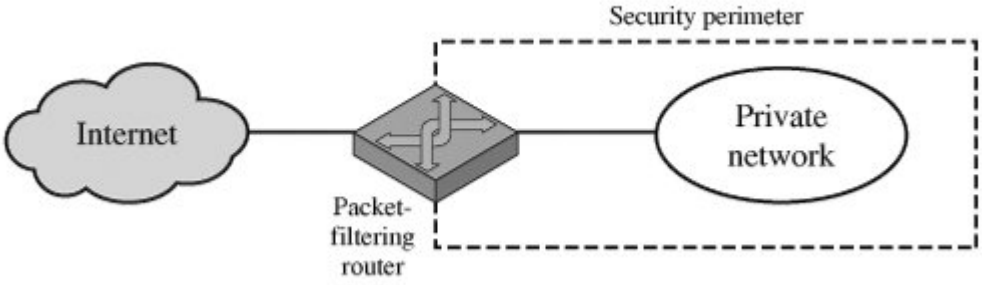
Application level gateways, and

Circuit level gateways.

Packet Filters

Packet filters act by inspecting the "packets" which travel between computers on the Internet. If a packet matches the packet filter's set of rules, the packet filter will drop (silently discard) the packet, or reject it (send "error message" to the

source).



It filters each packet based on information contained in the packet itself (for eg. packet's source and destination IP addresses, source and destination port numbers, protocol value etc..).

It can control traffic such as web browsing, remote printing, email transmission, file transfer.

Packet filtering firewalls work mainly on the first three layers of the OSI reference model, which means most of the work is done between the network and physical layers, with a little bit of peeking into the transport layer to figure out source and destination port numbers. When a packet originates from the sender and filters through a firewall, the device checks for matches to any of the packet filtering rules that are configured in the firewall and drops or rejects the packet accordingly.

When the packet passes through the firewall, it filters the packet on a protocol/port number basis. For example, if a rule in the firewall exists to block web browsing, then the firewall will block the TCP protocol for port number 80.

The firewall administrator may define the rules.

Consider a rule given below:

	action	ourhost	port	theirhost	port	comment
A	block	*	*	SPIGOT	*	we don't trust these people
	allow	OUR-GW	25	*	*	connection to our SMTP port

First entry means packets from a computer named 'SPIGOT' are blocked.

Second inblound mail (port 25 for SMTP) is allowed but only to a gateway computer.

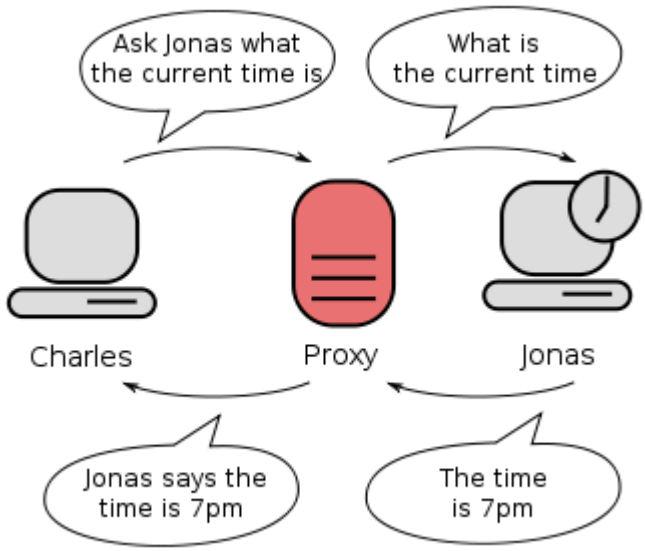
Consider another rule,

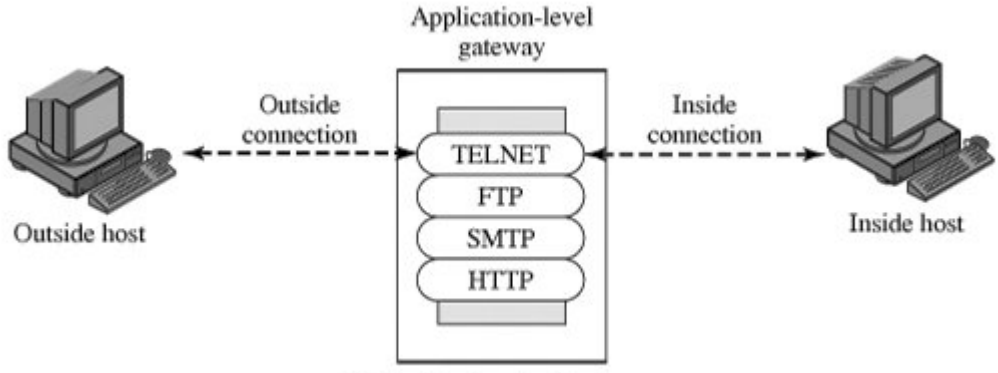
	action	ourhost	port	theirhost	port	comment
C	allow	*	*	*	25	connection to their SMTP port

This rule says that any inside computer can send emails to outside.

Application Level Gateway

It is a type of firewall. It is also called a proxy server. It acts as a relay of application level traffic.



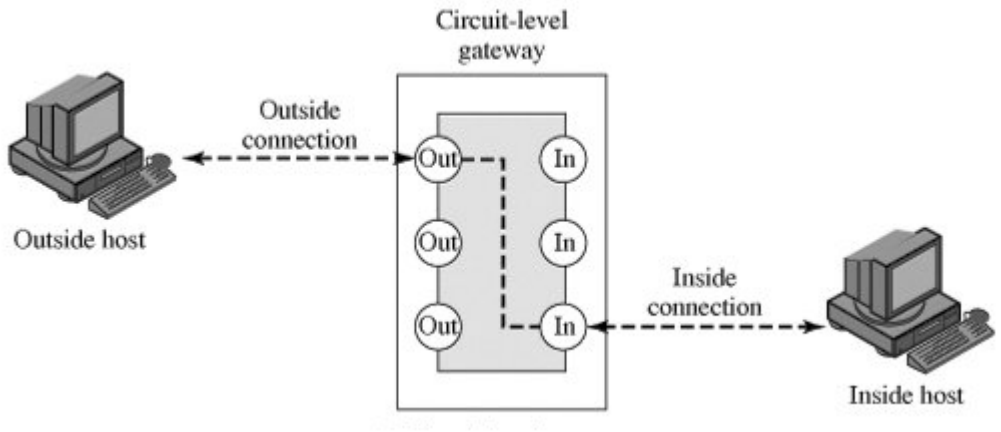


A proxy server may act as a firewall by responding to input packets (connection requests, for example) in the manner of an application, while blocking other packets. A client connects to the proxy server, requesting some service, such as a file, connection, web page, or other resource available from a different server. The proxy server evaluates the request according to its filtering rules. For example, it may filter traffic by IP address or protocol. If the request is validated by the filter, the proxy provides the resource by connecting to the relevant server and requesting the service on behalf of the client.

This type of firewall needs to monitor only a few allowable applications. It is easy to log andd audit all incoming traffic at the application level.

Proxies make tampering with an internal system from the external network more difficult.

Circuit Level Gateway



A circuit-level gateway does not permit an end-to-end TCP connection; rather,

the gateway sets up two TCP connections,

one between itself and a TCP user on an inner host, and

one between itself and a TCP user on an outside host.

Once the two connections are established, the gateway typically relays TCP segments from one connection to the other without examining the contents. The security function consists of determining which connections will be allowed.

19 More complex Firewall Configurations

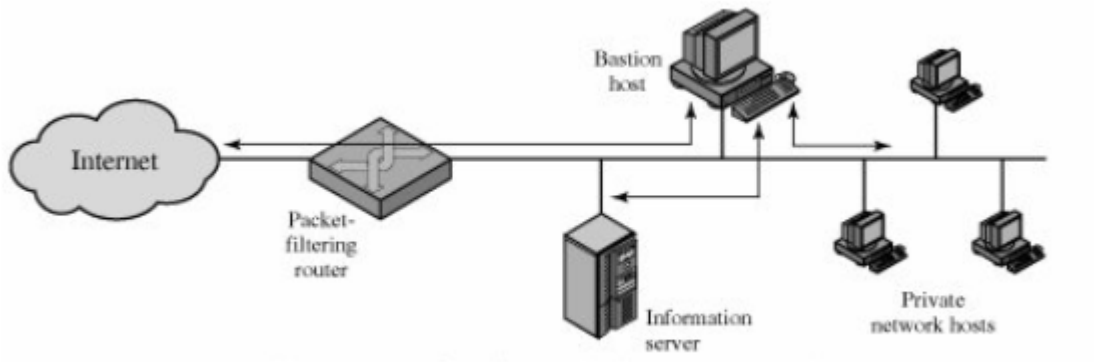
In addition to the use of a simple configuration consisting of a single system,more complex configurations are possible. Three of them are,

Screened host firewall system (single homed bastion host),

screened host firewall system (dual homed bastion host), and

screened subnet firewall system.

Screened host firewall system (single homed bastion host)



The firewall consists of two systems:

a packet-filtering router and

a bastion host.

Typically, the router is configured so that

For traffic from the Internet, only IP packets destined for the bastion host are allowed in.

For traffic from the internal network, only IP packets from the bastion host are allowed out.

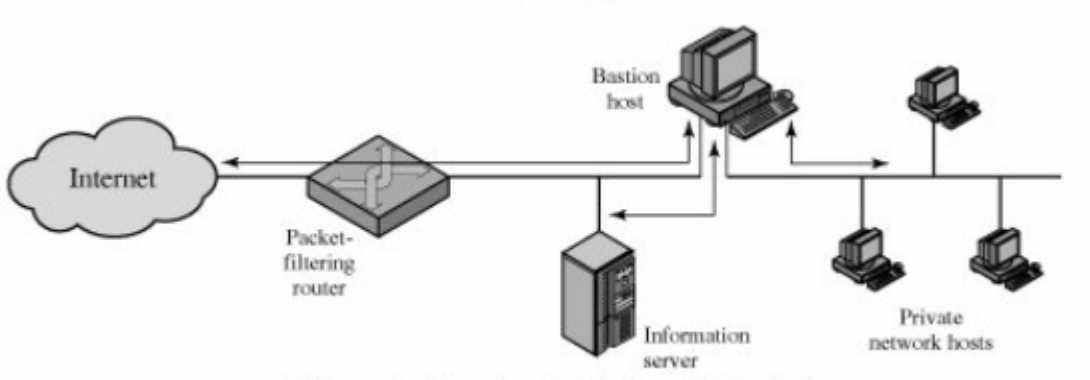
The bastion host performs authentication and proxy functions.

This configuration has greater security than simply a packet-filtering router or an application-level gateway alone, for two reasons.

First, this configuration implements both packet-level and application-level filtering, allowing for considerable flexibility in defining security policy.

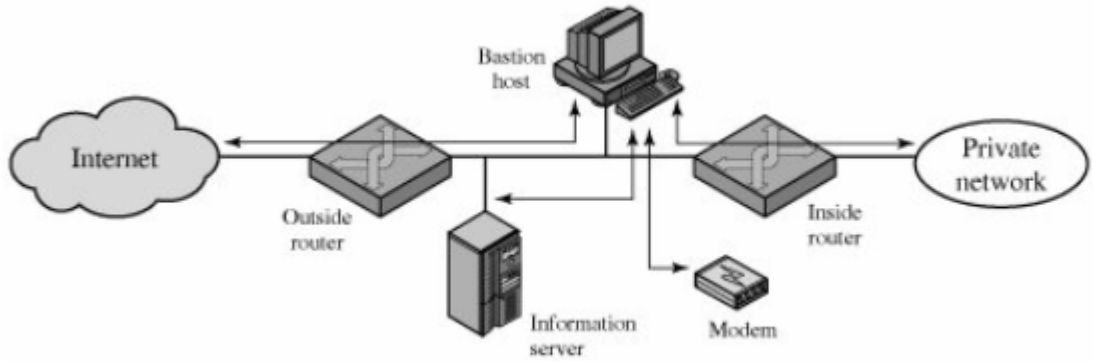
Second, an intruder must generally penetrate two separate systems before the security of the internal network is compromised.

Screened host firewall system (Dual homed bastion host)



Here, if the packet-filtering router is completely compromised, traffic cannopt flow directly through the router between the Internet and other hosts on the private network. This configuration physically prevents such a security breach.

Screened subnet firewall system



In this configuration, two packet-filtering routers are used,

one between the bastion host and the Internet and

one between the bastion host and the internal network.

This configuration creates an isolated subnetwork, which may consist of simply the bastion host but may also include one or more modems for dial-in capability. Typically, both the Internet and the internal network have access to hosts on the screened subnet, but traffic across the screened subnet is blocked. This configuration offers several advantages:

There are now three levels of defense to thwart intruders.

The outside router advertises only the existence of the screened subnet to the Internet; therefore, the internal network is invisible to the Internet.

Similarly, the inside router advertises only the existence of the screened subnet to the internal network; therefore, the systems on the inside network cannot construct direct routes to the Internet.

Questions

MGU/May2012

1. What are firewalls (4marks)?

2. Write briefly on applet security (4marks).

3a. Explain with an example, Email security.

OR

b. Explain briefly: i)Kerberos ii)S/MIME (12marks).

MGU/May2010

1. What is secure socket layer (4marks)?

2. What are firewalls?

3a. Explain about the electronic mail security with an example.

OR

b. Discuss in detail about security policy and security manager (12marks).

MGU/Nov2009

1. What is an applet? Explain (4marks).

2. Curve the X-509 format (4marks).

3a. Explain IP security architecture in detail.

OR

b. Explain the firewall design principles and its characteristics in detail (12marks).

MGU/June 2009

1. What is a socket? Explain about its creation (4marks).

2. List the design goals of a firewall (4marks).

3a. Explain briefly the X-509 authentication service.

OR

b. Explain the security mechanisms in JAVA platform (12marks).

MGU/May2008

1. What is Kerberos (4marks)?

2. What are firewalls (4marks)?

3a. Explain secure socket layer architecture briefly.

OR

b. How do firewalls help to achieve security? Discuss their limitations (12marks).

MGU/Nov2008

1. What are firewalls (4marks)?

2a. Describe briefly about secure electronic transaction.

OR

b. Briefly describe the IP security architecture (12marks).

MGU/July2007

1. What are Kerberos (4marks)?

2a. Describe in detail the IP security architecture.

OR

b. What are firewalls and how does it help in a secure transition (12marks)?

MGU/Jan2007

1. What are Kerberos (4marks)?

2. Define firewalls (4marks).

3a. Describe X509 authentication service.

OR

b. Explain electronic mail security aspects (12marks).

MGU/July2006

1. What is IP security architecture (4marks)?

2. Explain features in pretty good privacy (4marks).

3a. Describe the security mechanisms in JAVA platforms.

OR

b. Explain Kerberos authentication mechanism in the distributed system (12marks).

References

.

Stallings, W (2006). Cryptography and Network Security. Pearson Education.

Ramachandran, J (2002). Designing Security Architecture Solutions. Wiley Dreamtech.

website: http://sites.google.com/site/sjcetcssz