Unit - III

Cryptography: It is the science of using mathematics to encrypt and decrypt data. Cryptography enables you to store sensitive information or transmit it across insecure n/ws so that it cannot be read by anyone except the intended receipient.

Cryptography is the science of securing data, Cryptanalysis is the science of analyzing and breaking secure communication.

A cryptographic algorithm or cipher is a mathematical function used in the encryption and decryption process. A cryptographic algo works in combination with key - a word, number, or phrase to encrypt the plaintext. The same plaintext encrypts to different ciphertext with different keys.

The security of encrypted data is entirely dependent on the two things: the strength of the cryptographic algorithm and the secrecy of the key.

A cryptographic algo, plus all possible keys and all the protocols that make it work comprise a cryptosystem.

An original msg is known as plaintext, while the coded msg is called ciphertext. The process of converting from plaintext to ciphertext is called "enciphering" or "encryption", restoring the plaintext from ciphertext is called "deciphering" or "decryption".

The many schemes used for encryption constitute the area of study known as "Cryptography". Such a scheme is known as Cryptographic system or cipher.

Techniques used for deciphering a msg without any knowledge of the enciphering details fall into the area of cryptanalysis.

The areas of cryptography and cryptanalysis together are called "cryptology".

(1) Symmetric Key Cryptography [Conventional Cryptography]

In the secret key or symmetric key cryptography, one key is used both for encryption and decryption.

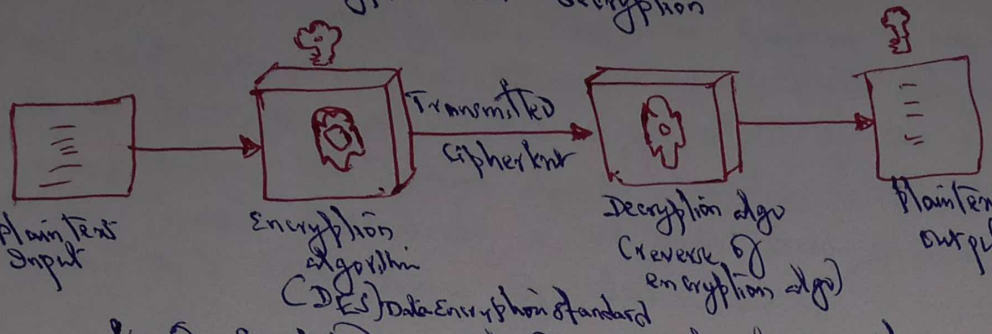


Fig ① Simplified model of Conventional Encryption

A symmetric encryption scheme has five ingredients (Fig ①):

- **Plaintext**: It is the original intelligible message or data that is fed into the algorithm as input.
- **Encryption Algorithm**: The encryption algorithm performs various substitutions and transformations on the plaintext.
- **Secret key**: The secret key also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algo produce different output depending on the specific key being used at the time. The exact substitutions & transformation performed by the algo depend on the key.
- **Ciphertext**: This is a scrambled msg produced as o/p. It depends on the plaintext and the key. For a plaintext, two different keys will produce two different ciphertexts
- **Decryption algorithm**: This is essentially the encryption algorithm run in reverse. It takes the ciphertext and secretkey and produce the original plaintext.

There are two requirements for secure use of Conventional encryption:

① "We need a strong encryption algorithm". At a minimum, we would like the algo to be such that an opponent who knows the algo and has access to one or more ciphertexts would be unable to decipher the ciphertext or figure out the key. In the stronger form: The opponent should be unable to decrypt ciphertext or discover the key even if he/she is in possession of a no of ciphertexts together with the plaintext that produced each ciphertext.

② Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure. If someone can discover the key and knows the algo, all communication using this key is readable.

We do not need to keep the algo secret, we need to keep only the key secret. With the use of symmetric encryption, the principal security problem is maintaining the secrecy of the key.

Below fig (2), A source produces a msg in plaintext, $X = [X_1, X_2, \ldots, X_M]$ The M elements in X are letters in some finite alphabet. For encryption a key of the form $K = [K_1, \ldots, K_J]$ is generated. If the key is generated at the message source, then it must also be provided to the destination by means of some secure channel. Alternatively a 3rd party could generate the key and securely deliver it to both source & destination

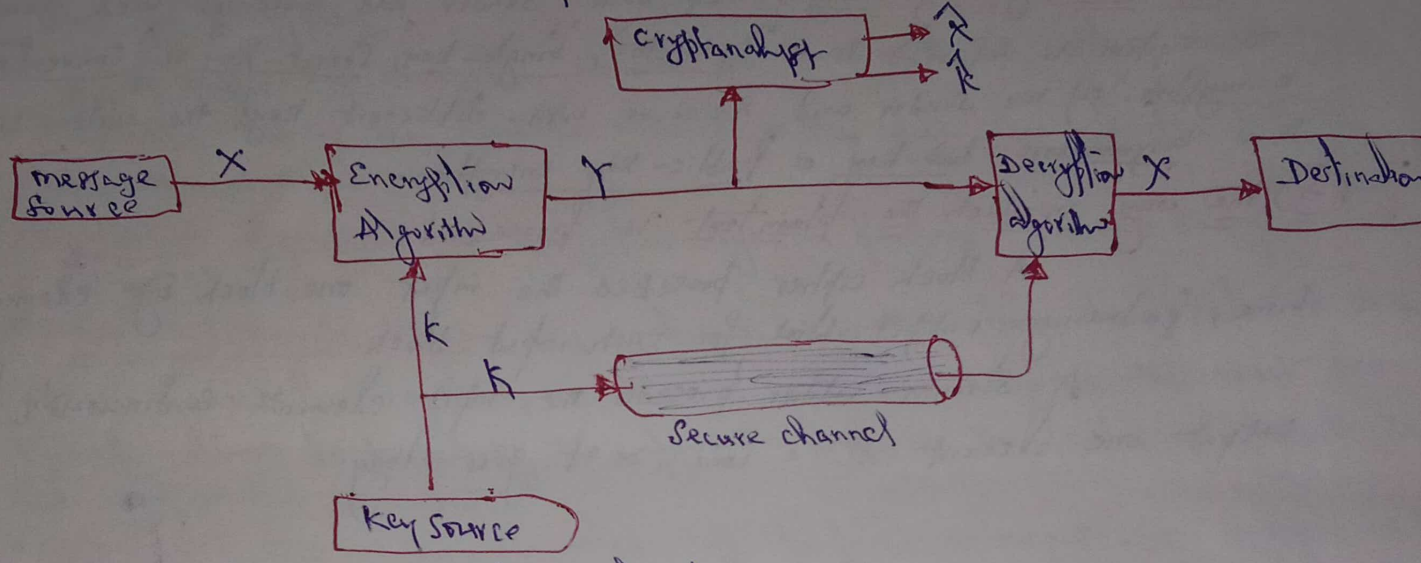


fig (2) model of conventional cryptosystem

With the msg X and the encryption key K as input, the encryption algo forms the ciphertext $Y = [Y_1, \ldots, Y_N]$ we can write this as

$$Y = E(K, X)$$

The intended receiver, in possession of the key, is able to invert the transformation

$$X = D(K, Y)$$

An opponent, observing Y but not having access to X or K, may attempt to recover X or K or both X and K. It is assumed that the opponent knows the encryption (E) and decryption algorithm (D).

Cryptographic Systems are ~~categorized~~ characterized along 3-independent dimensions:

① The type of Operation used for transforming plaintext to ciphertext:

All encryption algorithms are based on two general principles: "Substitution", in which each element in the plaintext (bit, letter, group of bits, or letters) is mapped into another element, and "Transposition", in which elements in the plaintext are "rearranged". The fundamental requirement is that no information be lost. most systems, referred to as product systems, involve multiple stages of substitutions and transpositions.

② The number of keys used: If both Sender and Receiver used same key, the system is referred to as "Symmetric", "Single-key", "Secret-key" or "Conventional encryption." If the Sender and Receiver used different keys, the system is referred to as "asymmetric", "two-key", or "public-key encryption".

③ The way in which the plaintext is processed:

A block cipher processes the input one block of elements at a time, producing output block for each input block.

A stream cipher processes the input elements continuously, producing output one element at a time, as it goes along

Asymmetric Key Cryptography [Public-Key Cryptography]

Asymmetric key algorithms rely on one key for encryption and a different but related key for decryption. These algorithms have the following important characteristic

- It is computationally infeasible to determine the decryption key given only the knowledge of the cryptographic algorithm and the encryption key

In addition, some algorithms such as RSA, also exhibit the following characteristic:

- Either of the two related keys can be used for encryption, with the other used for decryption

* A public-key encryption scheme has 6-ingredients shown below fig (1)

- **Plaintext**: This is the readable message, or the data that is fed into the algorithm as input
- **Encryption algorithm**: The encryption algorithm performs various transformations on the plaintext.
- **Public and private keys**: This is pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the algorithm depend on the public or private key that is provided as input.
- **Ciphertext**: This is the scrambled msg produced as output. It depends on the plaintext and the key. For a given msg, two different keys will produce two different ciphertexts
- **Decryption algorithm**: This algorithm accepts the ciphertext and the matching key and produces the original plaintext

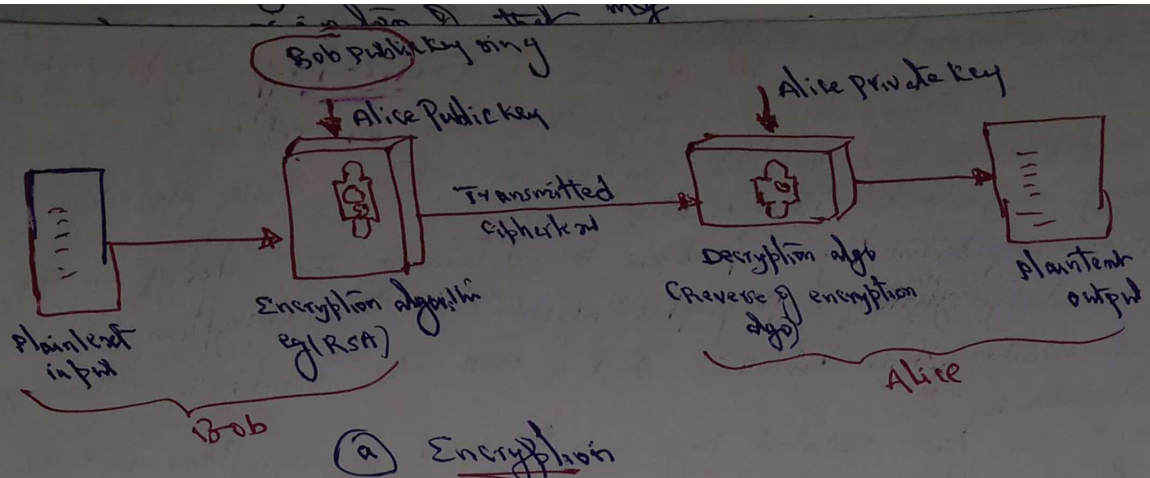
* **The Essential steps are as follows:**

(1) Each user generates a pair of keys to be used for the encryption and decryption of msgs

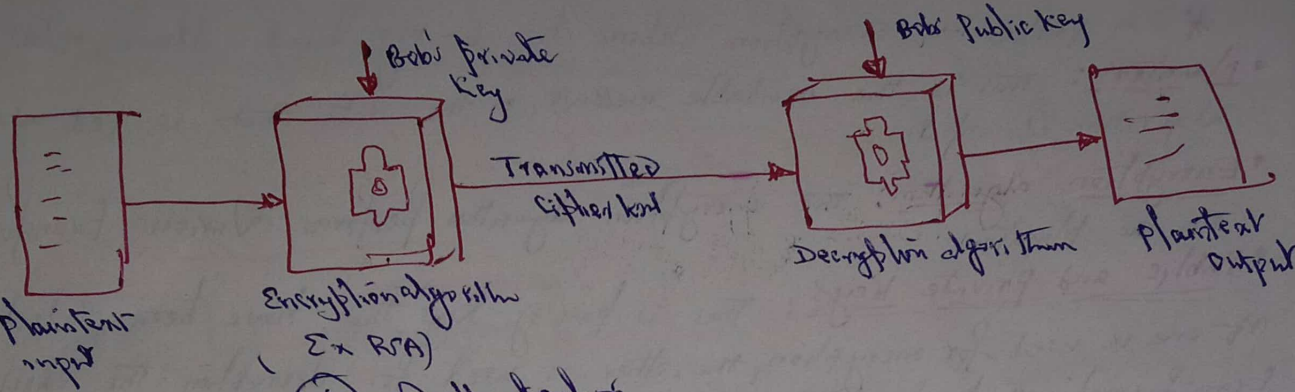
(2) Each user placed one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private. As fig (1)(A) suggests each user maintains a collection of public keys obtained from other

(3) If Bob wishes to send confidential msg to Alice, Bob encrypts the msg using Alice's public key

(4) When Alice receives the msg, she decrypts it using her private key. No other recipient can decrypt the msg because only Alice knows Alice's private key



(a) Encryption



(b) Authentication

fig① Public-key Cryptography

with this approach, all participants have access to public keys and private keys are generated locally by each participant and ∴ need never be distributed. As long as user's private key remains protected and secret, incoming communication is secure. At any time, a system can change its private key and publish the companion public key to replace its old public key.

Let us take a closer look at the essential elements of a public-key encryption scheme, using below fig (2). There is some source A that produces a msg in plaintext, $X = [X_1, X_2, \ldots X_m]$. The m elements of X are letters in some finite alphabet. The msg is intended for destination B. B generates a related pair of keys: public key PU_b, and a private key PR_b. PR_b known only to B, whereas PU_b is publicly available and ∴ accessible by A.

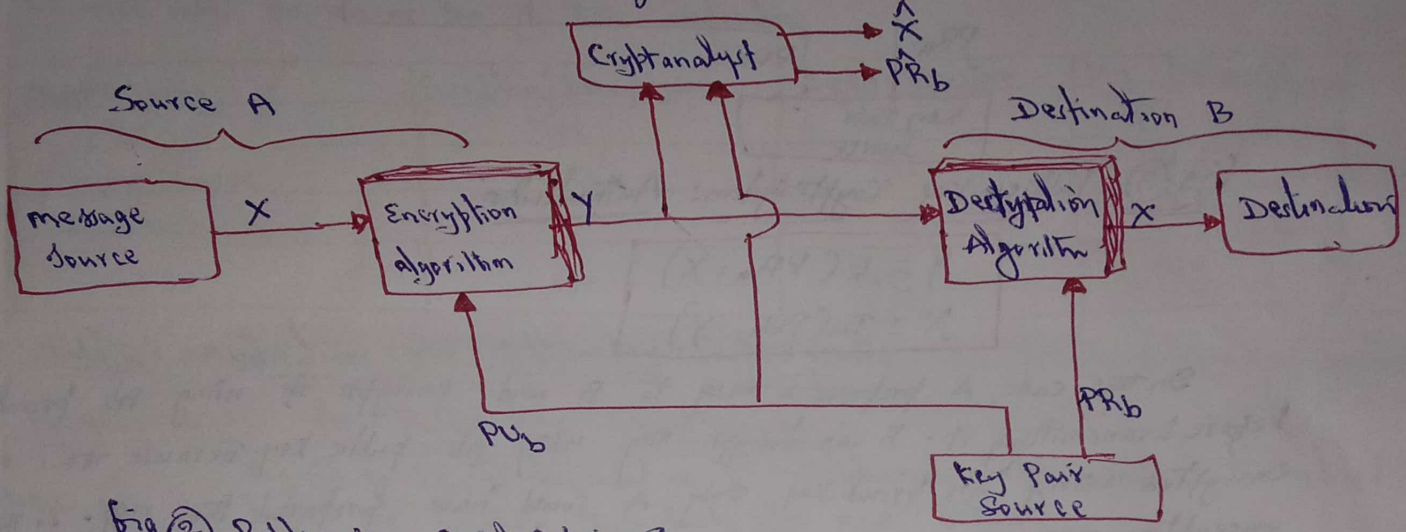


fig (2) Public-Key Cryptosystem: Secrecy

With the msg X and the encryption key PU_b as input, A forms the ciphertext $Y = [Y_1, \ldots Y_N]$

$$Y = E(PU_b, X)$$

The intended receiver, in possession of the matching private key, is able to invert the transformation:

$$X = D(PR_b, Y)$$

An adversary, observing Y and having access to PU_b, but not having access to PR_b or X, must attempt to recover X and/or PR_b. It is assumed that adversary does have knowledge of the encryption (E) and decryption (D) algorithms. If the adversary is interested only in this particular msg, then the focus of effort is to recover X, by generating a plaintext estimate $\hat{X}$. If adversary is interested in being able to read future msgs as well, in which case an attempt is made to recover PR_b by generating an estimate $\hat{PR}_b$.

The receiver of message, [illegible]

Either of the two related keys can be used for encryption, with the other being used for decryption. This enabled a rather different cryptographic scheme to be implemented. Whereas the scheme illustrated in fig ② provides confidentiality, fig ① ⓑ and fig ③ below use of public key encryption to provide authentication:

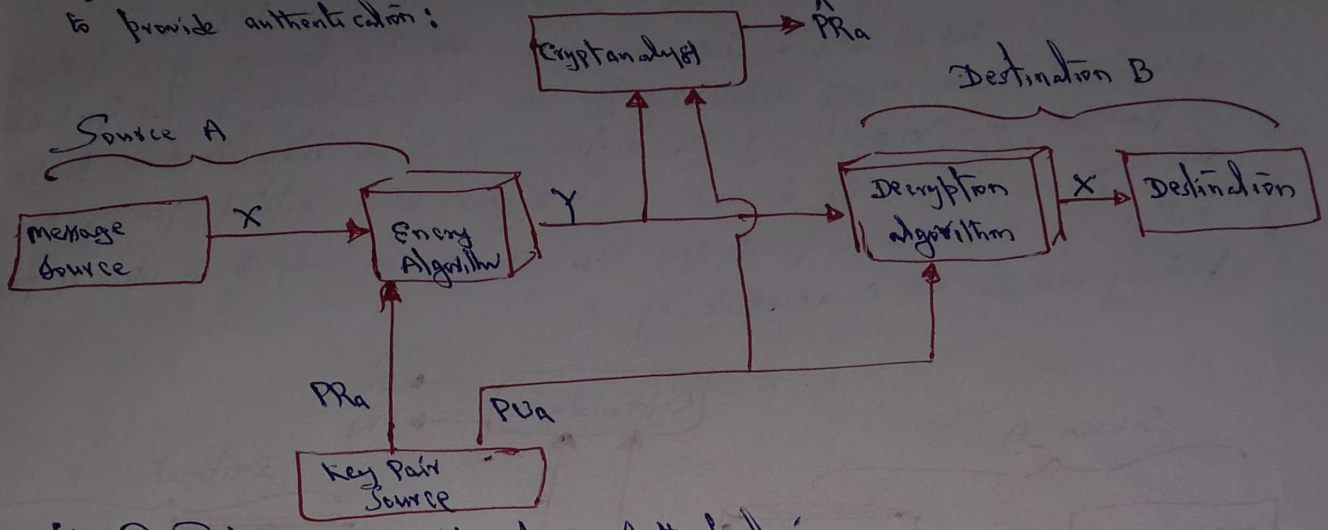
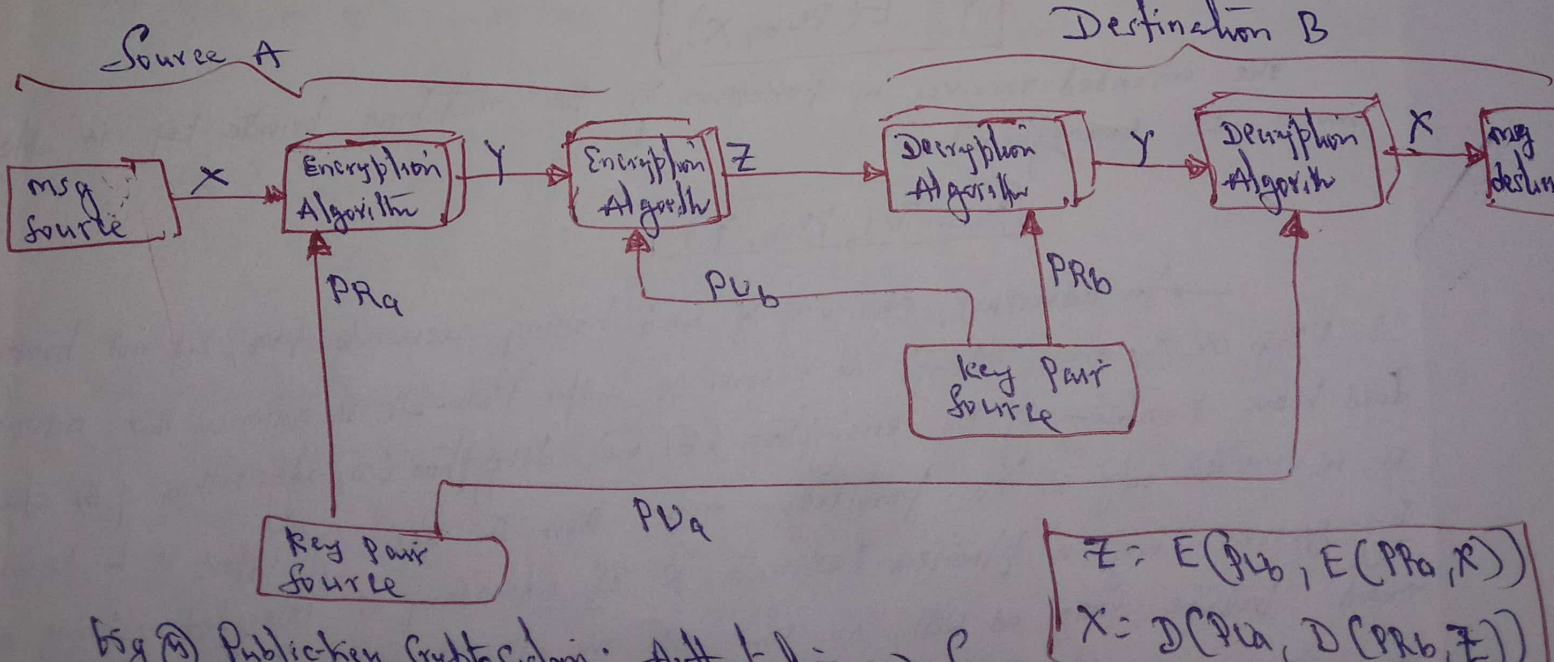


fig ③ Public-key Cryptosystem: Authentication

$$Y = E(PR_a, X)$$
$$X = D(PU_a, Y)$$

In this case, A prepared a msg to B and encrypts it using A's private key before transmitting it. B can decrypt msg using A's public key. Because the msg was encrypted using A's private key. Only A could have prepared the msg. ∴ the entire encryption msg served as a "digital signature"

It is possible to provide both authentication function and Confidentiality by double use of public-key scheme



$$Z = E(PU_b, E(PR_a, X))$$
$$X = D(PU_a, D(PR_b, Z))$$

fig ④ Public-key Cryptosystem: Authentication and Secrecy

* Applications for Public-key Cryptosystems

We can classify the use of public-key cryptosystems into three categories:

- **Encryption/Decryption:** The sender encrypts the msg with recipients public key.
- **Digital Signature:** The sender "signs" a msg with its private key. Signing is achieved by a cryptographic algorithm applied to the msg or to a small block of data that is a function of the msg.
- **Key Exchange:** Two sides cooperate to exchange a session key. Several different approaches are possible, involving the private key(s) of one or both parties.

Some algorithms are suitable for all 3-applications, whereas other can be used only for one or two of these applications

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Table: Applications for Public-key Cryptosystems

* Requirements for Public-Key Cryptography:

Fig (2) to fig (4) depends on a cryptographic algo based on two related keys. They did layout the conditions that such algo must fulfill:

(1) It is computationally easy for the ~~sender A~~, Party B to generate a pair (Public key PU_b, Private key PR_b) of keys

(2) It is computationally easy for a sender A, knowing the public key and the msg to be encrypted, M, to generate the corresponding ciphertext:

$$C = E(PU_b, M)$$

(3) It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original msg:

$$M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$$

(4) It is computationally infeasible for an adversary, knowing the public key PU_b, to determine the private key PR_b.

(5) It is computationally infeasible for an adversary, knowing the public key, PU_b, and ciphertext C, to recover the original msg M.

(6) The two keys can be applied in either order:

$$M = D[PU_b, E(PR_b, M)] = D[PR_b, E(PU_b, M)]$$

Message Authentication

It involves the use of a secret key to generate a small fixed-sized block of data known as a cryptographic checksum or MAC that is appended to the msg. This technique assumes that two communicating parties, say A and B share a common secret key K. When A has a message to send to B, it calculates the MAC as a function of the message and the key: MAC = C(K,M), where

M = input msg, C = MAC function, K = shared secret key

MAC = message Authentication code.

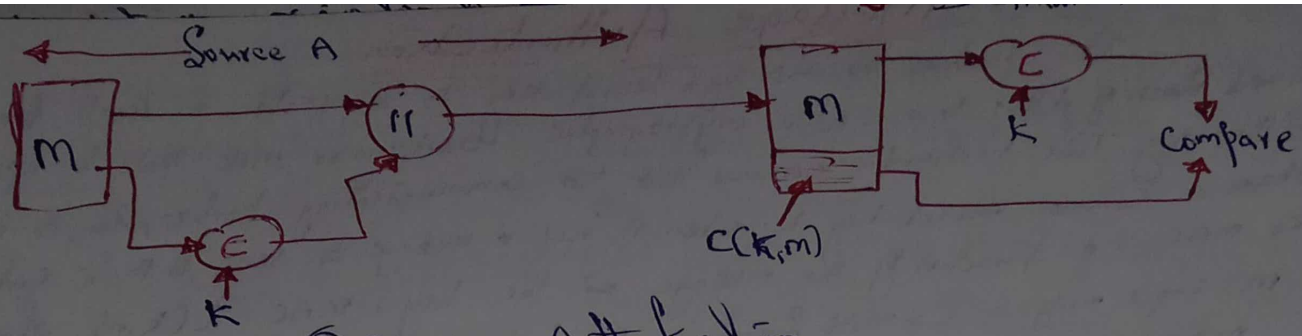
The message plus MAC are transmitted to the intended recipient. The recipient performs the same calculation on the received msg, using the same secret key to generate a new MAC. The received MAC is compared to the calculated MAC (Fig ① a below). If we assume that only the receiver and the sender know the identity of the secret key, and if the received MAC matches the calculated MAC, then

① The receiver is assured that the msg has not been altered. If an attacker alters the msg but does not alter the MAC, then the receiver's calculation of the MAC will differ from the received MAC. Because the attacker is assumed not to know the secret key, the attacker cannot alter the MAC to correspond to the alterations in the msg

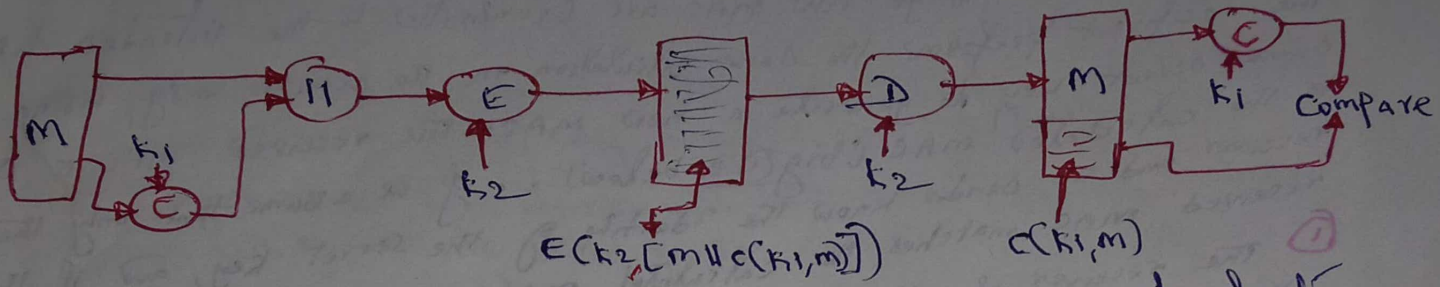
② The receiver is assured that the msg is from the alleged sender. Because no one else knows the secret key, no one else could prepare a msg with a proper MAC

③ If the msg includes a sequence number (such as is used with HDLC, X.25 and TCP) then the receiver can be assured of the proper sequence because an attacker cannot successfully alter the sequence numbe

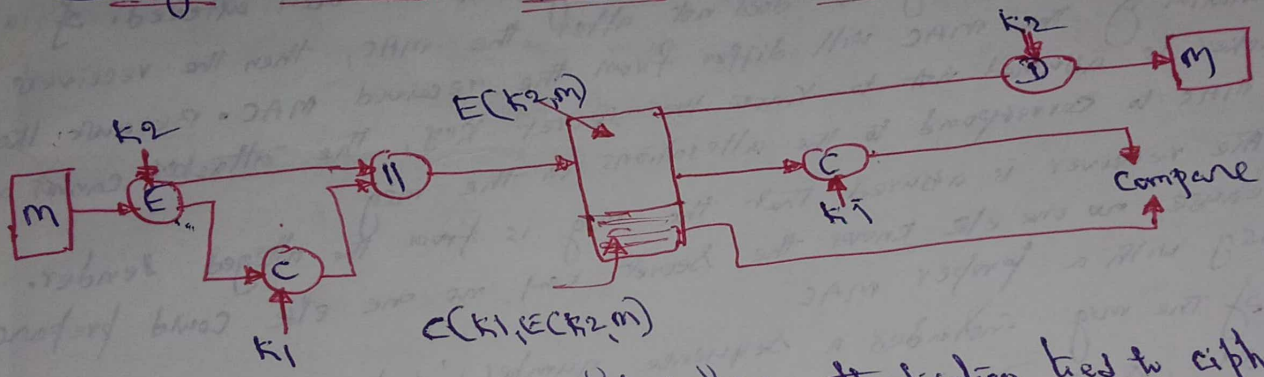
A MAC function is similar to encryption. One difference is that the MAC algo need not be reversible as it must for decryption. In general, the MAC function is a many-to-one function. The domain of the function consists of msgs of some arbitrary length, whereas the range consists of all possible MACs and all possible keys. If an n-bit MAC is used, then there are 2^n possible MACs, whereas there are N possible msgs with $N \gg 2^n$. Furthermore with a k-bit key, there are 2^k possible keys



(a) Message Authentication



(b) msg Authentication and Confidentiality: authentication tied to plaintext



(c) msg Authentication & Confidentiality: authentication tied to ciphertext

fig (1) Basic use of message Authentication code (MAC)

for ex, suppose that we are using 100-bit msgs and a 10-bit MAC. Then there are total of 2^{100} different msgs but only 2^{10} different MACs, so, on average, each MAC value is generated by a total of $2^{100}/2^{10} = 2^{90}$ different msg. If 5-bit key is used, there are $2^5 = 32$ diff msg from the set of msg to the set of MACs values

Because of mathematical properties of the authentication function, it is less vulnerable to being broken than encryption.

The process depicted in fig ① ④ provides authentication but not confidentiality, because the msg as a whole is transmitted in the clear. Confidentiality can be provided by performing msg encryption either after (fig ① ⑤) or before (fig ① ⑥) the MAC algorithm. In both these cases, two seperate keys are needed, each of which is shared by the Sender and the Receiver. In the first case, the MAC is calculated with the msg as input and is then concatenated to the msg. The entire block is then encrypted. In the Second case, the msg is encrypted first. Then the MAC is calculated using the resulting ciphertext and is concatenated to the ciphertext to form the transmitted block. It is preferable to tie the authentication directly to the plaintext, so the method of fig ① ③ is used.

Because, Symmetric encryption will provide authentication and because it is widely used with readily available products, why not simply use this instead of a seperate MAC? [DAVI89] Suggests 3-situations in which a MAC is used:

① There are a no. of applications, in which the same msg is broadcast to a no. of destinations. Ex are notification to user for the n/w is unavailable or an alarm signal in a military control center. It is cheaper and more reliable to have only one destination responsible for monitors authenticity. Also, the msg must be broadcasted in plaintext with an associated MAC. The responsible system has the secret key and performs authentication. If a violation occurs, the other destination systems are alerted by a general alarm.

② Another possible scenario is an exchange in which one side has a heavy load and cannot afford the time to decrypt all incoming messages. Authentication is carried out on a selective basis, msgs being chosen at random for checking

$A \rightarrow B: M \| C(K, M)$

- Provides Authentication
  - only A and B share key K.

(a) message Authentication

$A \rightarrow B: E(K_2, [M \| C(K_1, M)])$

- Provides authentication
  - only A and B share K_1
- Provides Confidentiality
  - Only A and B share K_2

(b) msg authentication and confidentiality: authentication tied to plaintext

$A \rightarrow B: E(K_2, M) \| C(K_1, E(K_2, M))$

- Provides authentication
  - using K_1
- Provides Confidentiality
  - using K_2

(c) msg authentication and Confidentiality: authentication tied to cipher text

Table 2: Basic use of MAC (fig 1)

Digital Signature

A digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer. In the real world, the receiver of message must be sure that the message belongs to the sender and he should not be able to ... [illegible] ... that msg.

(3) (2) Authentication of a computer program in plaintext is an alternative service. The computer program can be executed without having to decrypt it every time, which would be wasteful of processor resources. If MAC were attached to the program, it could be checked whenever assurance was required of the integrity of the program.

3- other rationales may be added as follows:

(4) for some applications, it may not be of concern to keep msgs secret, but it is imp to authenticate msgs. Ex is the SNMP V3, which separate functions of confidentiality of authentication.

(5) seperation of confidentiality of authentication functions affords architectural flexibility. for ex. it may be desired to perform authentication at the application level but to provide confidentiality at a lower level, such as the Transport layer

(6) A user may wish to prolong the period of protection beyond the time of reception and yet allow processing of msg contents. with msg encryption, the protection is lost when the msg is decrypted, so the msg is protected against fraudulent modification only in transit but not within the target system.

Finally note that the MAC does not provide a digital signature because both sender and receiver share the same key

Table (1) summarizes the confidentiality and authentication implications of the approaches illustrated in fig (1)

A digital signature is a cryptographic value that ...
from the data and a secret key known only to the signer. In the real world, the receiver of a message must be sure that the message was sent by the claimed sender. Like a manual signature, the digital signature must be unique to the owner and must be able to ... that ...

(3) Authentication of a computer program in plaintext is an attractive service. The computer program can be executed without having to decrypt it every time, which would be wasteful of processor resources. If MAC were attached to the program, it could be checked whenever assurance was required of the integrity of the program.

3- other rationales may be added as follows:

(4) for some applications, it may not be of concern to keep msgs secret, but it is imp to authenticate msgs. Ex is the SNMP V3, which separate functions of confidentiality of authentication.

(5) Separation of confidentiality of authentication functions affords architectural flexibility. For ex, it may be desired to perform authentication at the application level but to provide confidentiality at a lower level, such as the transport layer

(6) A user may wish to prolong the period of protection beyond the time of reception and yet allow processing of msg contents. With msg encryption, the protection is lost when the msg is decrypted, so the msg is protected against fraudulent modification only in transit but not within the target system.

Finally note that the MAC does not provide a digital signature because both sender and receiver share the same key

Table (1) summarizes the confidentiality and authentication implications of the approaches illustrated in fig (1)

Digital Signature

(1)

A digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer. In the real world, the receiver of message must be sure that the message belongs to the sender and he should not be able to [illegible] that msg.

(3) Authentication of a computer program in plaintext is an attractive service. The computer program can be executed without having to decrypt it every time, which would be wasteful of processor resources. If MAC were attached to the program, it could be checked whenever assurance was required of the integrity of the program.

3- other rationales may be added as follows:

(4) for some applications, it may not be of concern to keep msgs secret, but it is imp to authenticate msgs. Ex is the SNMP V3, which separate functions of confidentiality & authentication.

(5) Separation of confidentiality & authentication functions affords architectural flexibility. for ex, it may be desired to perform authentication at the application level but to provide confidentiality at a lower level, such as the Transport layer

(6) A user may wish to prolong the period of protection beyond the time of reception and yet allow processing of msg contents. With msg encryption, the protection is lost when the msg is decrypted, so the msg is protected against fraudulent modification only in transit but not within the target system.

Finally note that the MAC does not provide a digital signature because both sender and receiver share the same key

Table (1) summarizes the Confidentiality and authentication implications of the approaches illustrated in fig (1)

(3) Authentication of a computer program in plaintext is an alternative service. The computer program can be executed without having to decrypt it every time, which would be wasteful of processor resources. If MAC were attached to the program, it could be checked whenever assurance was required of the integrity of the program.

3- other rationales may be added as follows:

(4) for some applications, it may not be of concern to keep msgs secret, but it is imp to authenticate msgs. Ex is the SNMP V3, which seperate function of confidentiality of authentication.

(5) Seperation of confidentiality of authentication function affords architectural flexibility. for ex, it may be desired to perform authentication at the application level but to provide confidentiality at a lower level, such as the transport layer

(6) A user may wish to prolong the period of protection beyond the time of reception and yet allow processing of msg contents. with msg encryption, the protection is lost when the msg is decrypted, so the msg is protected against fraudulent modification only in transit but not within the target system.

Finally note that the MAC does not provide a digital signature because both sender and receiver share the same key

Table (1) summarizes the confidentiality and authentication implications of the approaches illustrated in fig (1)

Digital Signature

A digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer. In the real world, the receiver of message need assurance that the msg belongs to the sender and he should not be able to repudiate the origination of that msg.

The digital signature scheme is based on public key cryptography.

The important reason to implement digital signature to communication is:

- Authentication
- Non-Repudiation
- Integrity

Authentication: Authentication is a process which verifies the identity of a user who want to access the system. In the digital signature, authentication helps to authenticate the source of msg

Non-Repudiation: means assurance of something that cannot be denied. It ensures that someone in a contract or communication cannot later deny the authenticity of their signature on a document or in a file or the sending of a msg that they originated

Integrity: It ensures that the msg is real, accurate and safeguards from unauthorized user modification during the transmission

*Algorithms in Digital Signature:

A digital signature consists of 3 algorithms:

① Key-generation algorithm:

The key generation algorithm selects private key randomly from a set of possible private keys. This algo provides the private key and its corresponding public key

② Signing algorithm:

It produces a signature for the document

③ Signature Verifying algorithm:

A signature verifying algorithm either accepts or rejects the document authenticity

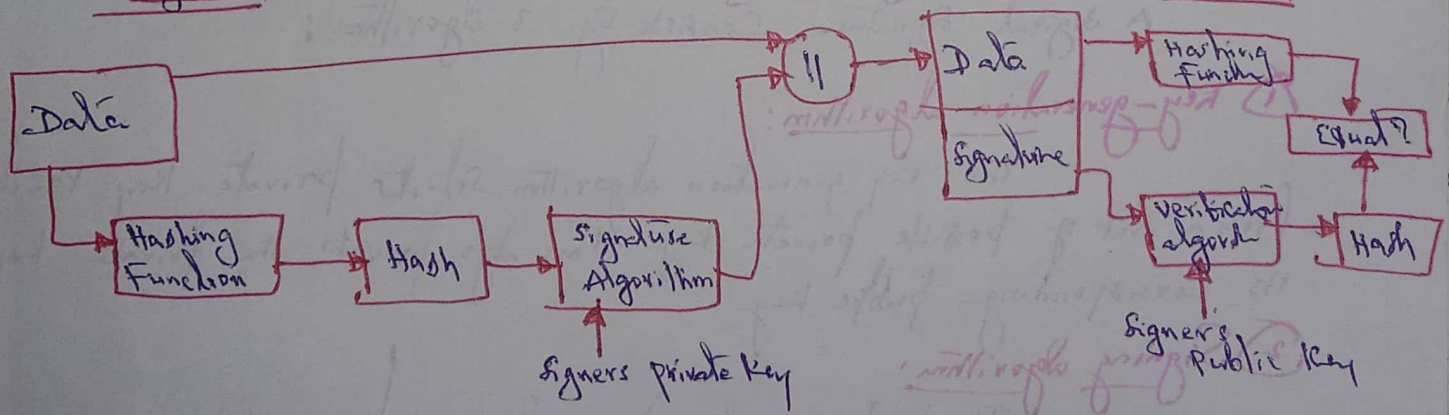
How Digital Signature works

Digital signatures are created and verified by using public-key cryptography, also known as asymmetric cryptography. By using public key algo (RSA) we can generate two keys that are mathematically linked - one is private key and other is public key.

The user who creates the digital signature use their own private key to encrypt the signature-related document.

The steps which are followed in creating a digital signature are:

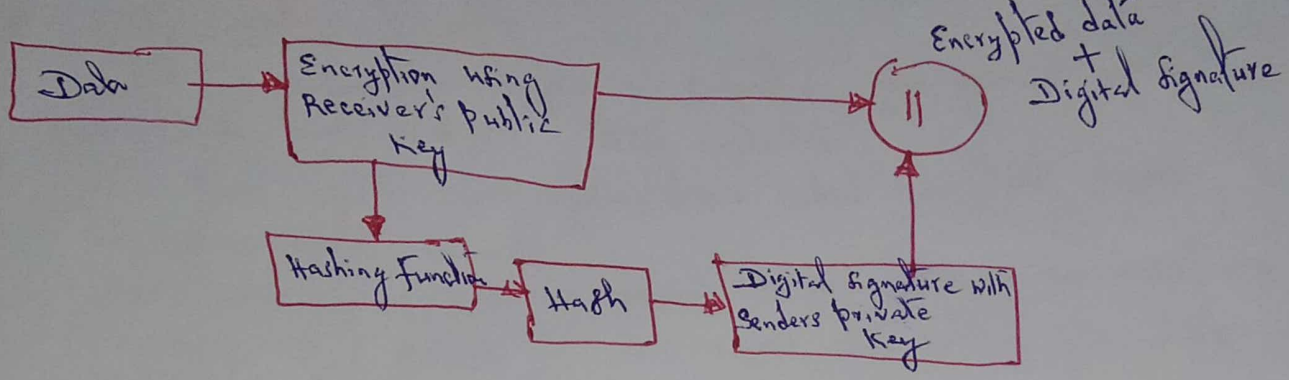
1. Select a file to be digitally signed
2. The hash value of the msg or file content is calculated. This message or file content is encrypted by using a private key of the sender to form the digital signature
3. Now, the original msg or file content along with the digital signature is transmitted.
4. The receiver decrypt the digital signature by using the public key of sender.
5. The receiver now has the msg or file content and can compute it.
6. Comparing these computed msg or file content with the original computed msg. The comparison needs to be the same for ensuring integrity



Digital Signature Encryption

Sending an encrypted msg is a safer choice than sending plaintext. In public-key encryption, the sender's public key is openly available and anyone can spoof their identity to send encrypted msg. The best way is to use Digital Signature along with encrypted msgs

Sender's side:



Types of Digital Signature:

(1) **Simple:** It is a digital signature in its simplest form because it is not protected by any encryption method. The most common example is a ~~wet~~ wet signature scanned by an electronic device and then inserted into a document. Another ex of a simple digital signature is the email signature that we often add at the end of the email, and check the terms of conditions box in s/w installation process

These are very easy to duplicate or fake (not encrypted). Both in terms of Security & legality, the use of digital signature in this type is not recommended

(2) **Basic:** Digital basic signatures don't have much difference compared to simple digital signatures. Advantage over simple is only their ability to show changes that occur after the document is signed. This is still cannot guarantee the security of identity because it cannot refer to verified identity. It don't optimally verify the user's identity (even using PKC). It is not through 2-factor authentication

It doesn't have legal power of legal Consequences

(3) **Advanced & Qualified:**

It is the safest DS as it has legal strength equivalent to a wet signature on paper. These signatures made with asymmetric key cryptographic technology and public key infrastructure

This DS provider more speed in the process of verifying the identity of the users they are applying. So these providers are required to impose a 2-factor authentication before the document can be signed by the user

The authentication method used also varies from sending OTP via sms to biometric scanning on mobile phones

Overview of Firewalls

Firewall can be an effective means of protecting a local system or n/w of systems from n/w-based security threats while at the same time affording access to the outside world via wide area n/w and the Internet

Firewall Design Principles:

Information System in Corporations, govt agencies, and other organizations have undergone a steady evaluation:

- Centralized data processing system, with a central mainframe supporting a no. of directly connected terminals.
- LAN interconnecting PCs and terminals to each other and the mainframe
- Premises n/w, consisting of a no. of LANs, interconnecting PCs, servers and perhaps a mainframe or two.
- Enterprise-wide n/w, consisting of multiple, geographically distributed premises n/ws interconnected by a private WAN
- Internet connectivity, in which the various premises n/ws all hook into the Internet and may or may not also be connected by a private WAN

Internet connectivity is no longer optional for organizations. The info and services available are essential to the organization. moreover, individual users within the organization want and need Internet access, and if this is not provided via their LAN, they use dial-up capability from their PC to an Internet Service Providers (ISP). While Internet access provides benefits to the organization, it enables the outside world to reach and interact with local n/w assets. This creates a threat to the organization. While it is possible to equip each workstation and server on the premises n/w with strong security features, such as intrusion protection, this is not a practical approach.

Consider a n/w with hundreds or even thousands of systems, running a mix of various versions of UNIX, plus windows. When a security flaw is discovered, each potentially effected system must be upgraded to fix that flaw. The alternative, increasingly accepted, is the firewall. The firewall is inserted b/w the premises n/w and the Internet to establish a controlled link and to erect an outer security wall or perimeter. The aim of this perimeter is to protect the premises n/w from Internet-based attacks and to provide a single choke point where security and audit can be imposed. The firewall may be a single computer system or a set of two or more systems that cooperate to perform the firewall function

What is a Firewall?

A firewall can be defined as a special type of n/w security device or a s/w program that monitors and filters incoming and outgoing n/w traffic based on a defined set of security rules. It acts as a barrier b/w Internal private n/w and external source (public Internet).

A firewall can be a n/w security device or a s/w program on a computer. Means that the firewall comes at both levels i.e h/w and s/w.

A h/w firewall is a physical device that sits either b/w a "Computer n/w" and a "gateway" ex: broadband router

A s/w firewall is a simple program installed on a computer that works through port numbers and other installed s/w.
Apart from that, there are cloud-based firewalls. They are commonly referred to as FaaS (Firewall as a service). Advantage is that they can be managed centrally. Like h/w firewalls, cloud-based firewalls are best known for providing perimeter security

Why Firewall?

Firewalls are primarily used for to prevent malware and n/w-based attacks, also block n/w-layer attacks

Firewalls are designed in such a way that they can react quickly to detect and counter-attack throughout the n/w.

Firewall Characteristics

The following are the design goals for a firewall:

1. All traffic from inside to outside, and vice versa, must pass through the firewall. This is achieved by physically blocking all access to the local n/w except via the firewall. Various configurations are possible, ad
2. Only authorized traffic, as defined by the local security policy, will be allowed to pass. Various types of firewalls are used, which implement various types of security policies.
3. The firewall itself is immune to penetration. This implied that use of a trusted system with a secure OS.

There are 4 general techniques that firewalls use to control access and enforce the site's security policy. Originally, firewalls focused primarily on service control, but they have since evolved to provide all four:

- **Service Control:**
  Determines the types of Internet services that can be accessed, inbound or outbound. The firewall may filter traffic on the basis of IP address and TCP port number; may provide proxy s/w that receives and interprets each service request before passing it on; or may host the server s/w itself, such as a web or a mail service.

- **Direction Control:**
  Determines the direction in which particular service requests may be initiated and allowed to flow through the firewall

- **User Control:** controls access to service according to which user is attempting to access it. This feature is typically applied to user inside the firewall perimeter (local users). It may also be applied to incoming traffic from external users, the latter requires some form of secure authentication technology, such as is provided in IPsec.

- **Behavior Control:** controls how particular services are used. For ex, the firewall may filter e-mail to eliminate spam, or it may enable external access to only a portion of the information on a local web server.

Functions of Firewall:

The firewall works as a gatekeeper. It analyzes every attempt coming to gain access to our OS and prevent traffic from unwanted or non-recognized sources

Since the firewall acts as a barrier or filter b/w the Computer System and other n/ws (public Internet), we can consider it as a traffic controller. ∴ a firewall primary function is to secure our n/w and info by controlling n/w traffic, and validating access by assessing n/w traffic for malicious things such as trackers, malware

Generally most OS (windows) and security sw come with built-in firewall support. Fire-walls become so powerful, and include a variety of functions and capabilities with built-in features:

- N/w threat prevention
- Application and Identity-based Control
- Hybrid cloud Support
- Scalable performance
- N/w traffic mgmt and control
- Access validation
- Record and Report on Events

Limitations of Firewall

- Firewalls cannot stop users from accessing malicious websites, making it vulnerable to internal threats or attacks
- Firewalls cannot protect against the transfer of virus-infected files or software
- Firewalls cannot prevent misuse of passwords
- Firewalls cannot protect if security rules are misconfigured
- Firewalls cannot protect against non-technical security risks such as social engineering
- Firewalls cannot stop or prevent attackers with modems from dialing in to or out of the internal n/w
- Firewalls cannot secure the system which is already infected.

An example of a circuit-level gateway implementation is the SOCKS package. Version 5 of socks is defined in RFC 1928. RFC defines SOCKS in the following fashion:

→ Protocol described here is designed to provide a framework for client-server application in both the TCP and UDP domains to conveniently and securely use the services of a n/w firewall. The protocol is conceptually "shim-layer" b/w the "application layer" and the "transport layer", & does not provide n/w-layer gateway services, such as forwarding of ICMP msgs.

SOCKS consists of the following components:

- The "SOCKS server", runs on a UNIX-based firewall
- The "SOCKS client library", runs on internal hosts protected by a firewall
- SOCKS-ified version of several standard client programs such as FTP and TELNET.

SOCKS Server TCP 1080 Port

When a TCP based client wishes to establish a connection to an object that is reachable only via a firewall, it must open a TCP connection to the appropriate SOCKS port on the SOCKS server system. The SOCKS service is located on TCP port no. 1080. If the connection request succeeds, the client enters a negotiation for the authentication method to be used, authenticates with the chosen method, and then sends a relay request. The SOCKS server evaluates the request and either establishes the appropriate connection or denies it. UDP exchanges are handled in a similar fashion.

In essence, TCP connection is opened to authenticate a user to send and receive UDP segments, and the UDP segments are forwarded as long as the TCP connection is open.

Bastion Host: It is a system identified by the firewall administrator as a critical strong point in the n/w security. Typically, the bastion host serves as a platform for an application-level or circuit-level gateway.

platform

Some common characteristics of bastion host includes;

- It's h/w platform executes a secure version of its OS, making it a trusted system
- Only the services that the n/w administrator considers essential are installed on the bastion host. These includes proxy applications such as Telnet, DNS, FTP, SMTP and user authentication
- It may require additional authentication before a user is allowed access the proxy services. In addition, each proxy service may require its own authentication before granting user access
- Each proxy is configured to support only a subset of the standard application command set
- Each proxy is configured to allow access only to specific host system. This means that limited command/feature set may be applied only to a subset of systems on the protected n/w
- Each proxy maintains detailed audit info by logging all traffic, each connection and the duration of each connection, and the audit log is an essential tool for discovering and terminating intruder attacks
- Each proxy module is a very small s/w package specifically designed n/w securing
- Each proxy is independent of other proxies on the bastion host. If there is a problem with the operation of any proxy or if a future vulnerability is discovered, it can be uninstalled without effecting the operation of the other proxy applications
- A proxy generally performs no disk access other than to read its initial configuration file. This makes it difficult for an intruder to install Trojan horses, sniffer or other dangerous files on the bastion host
- Each proxy runs as a non-privileged user in a private and secured directory on the bastion host.

Proxy: In computer n/wing, a proxy server is a server application that acts as an intermediary b/w a client requesting a resource and the server providing that resource

Proxy server is a system or router that provides a gateway b/w user and the internet. ∴ it helps prevent cyber attackers from entering a private n/w. It is a server, referred to as an "intermediary" because it goes b/w end-users and the webpages they visit online

Proxy Service: A proxy service is an intermediary role played by s/w or a dedicated computer b/w an endpoint device and a client which is requesting the service. The proxy service may exist on the same m/c or on a separate server

Types of Firewalls

① Packet-filtering Router:

A packet filtering router applies a set of rules to each incoming and outgoing IP packet and then forwards or discards the packet. The router is typically configured to filter packets going in both the direction. Filtering rules are based on information contained in a network packet:

- **Source IP address:** The IP address of the system that originates the IP packet
- **Destination IP address:** The IP address of the system the IP packet is trying to reach.
- **Source and Destination transport-level address:** The transport level (TCP or UDP) port number, which defined applications such as SNMP or TELNET
- **IP Protocol field:** Defines the transport protocol.
- **Interface:** For a router with 3 or more ports, which interface of the router the packet came from or which interface of the router the packet is defined for

The packet filter is typically set up as a list of rules based on matches to fields in the IP or TCP header. If there is a match to one of the rules, that rule is invoked to determine whether to forward or discard the packet. If there is no match to any rule, then a default action is taken. Two default policies are possible:

- **Default = discard:** That which is no explicitly permitted is prohibited
- **Default = forward:** That which is not explicitly prohibited is permitted

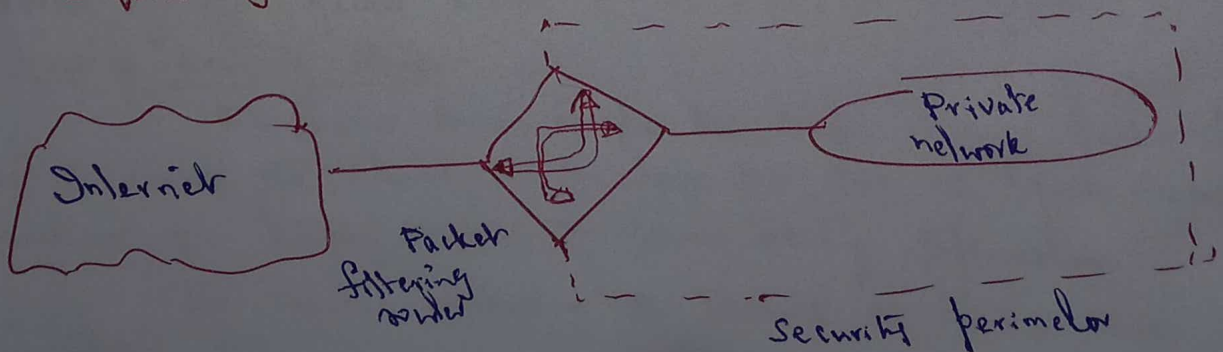


Fig ① Packet filtering router

One advantage of a packet-filtering router is its simplicity. Packet-filters are typically app. transparent to users and are very fast. Following are the weaknesses of packet-filtering firewalls:

- Because packet filter firewalls do not examine upper-layer data, they cannot prevent attacks that employ application-specific vulnerabilities or functions
- Because of the limited info available to the firewalls, the logging functionality present in packet filter firewalls is limited. Packet filter logs normally contain the same info used to make access control decisions
- most packet filter firewalls do not support advanced user authentication schemes this limitation is mostly due to the lack of upper-layer functionality by the firewalls
- they are generally vulnerable to attacks and exploits that take advantage of problems within the TCP/IP specification and protocol stack, such as network layer address spoofing
- Due to small no. of variables used in access control decisions, packet filter firewalls are susceptible to security breaches caused by improper configurations

Some of the attacks that can be made on packet-filtering routers and the appropriate countermeasures are the following:

- **IP address spoofing:**

The intruder transmits packets from the outside with a source IP address field containing an address of an internal host. The attacker hopes that the use of a spoofed address will allow penetration of systems that employ simple source add security, in which packets from specific trusted internal hosts are accepted

The countermeasure is to discard packets with an inside source address if the packet arrive on an external interface

- **Source routing attacks:**

The source station specifies the route that a packet should take as it crosses the Internet, in the hopes that this will bypass security measures that do not analyze the source routing information

The counter measure is to discard all packets that use this option

Tiny Fragment Attacks:

The intruder uses the IP fragmentation option to create extremely small fragments and force the TCP header into a seperate packet fragment. This attack is designed to circumvent filtering rules that depend on TCP header info

Typically the packet filter will make a filtering decision on the first fragment of a packet. All subsequent fragments of that packet are filtered out solely on the basis that they are part of the packet whose first fragment was rejected

A tiny fragment attack can be defeated by enforcing a rule that the first fragment of a packet must contain predefined min amount of the transport header

If the first fragment is rejected, the filter can remember the packet & discard all subsequent fragments

Ex: Stateful Inspection Firewalls:

A traditional packet filter makes filtering decisions on individual packet basis and not take into consideration any higher layer context

(2) Application-Level Gateway:

Also called a proxy server, acts as a relay of application-level traffic.

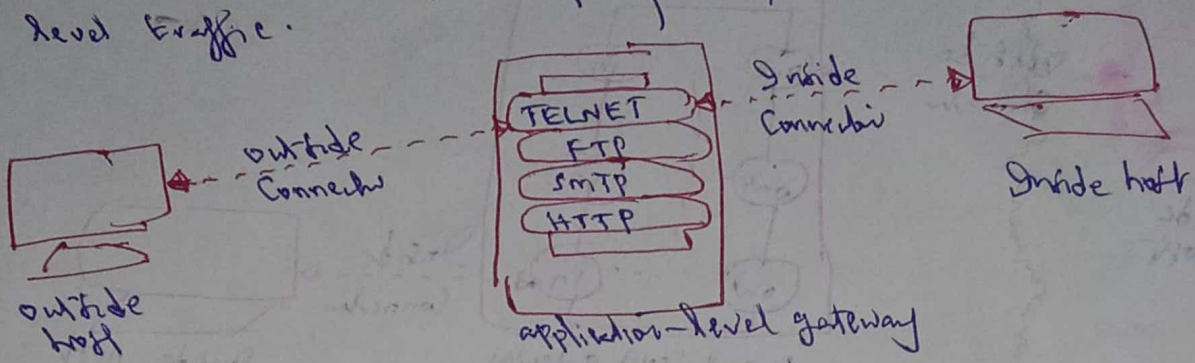


fig (2) Application-Level Gateway

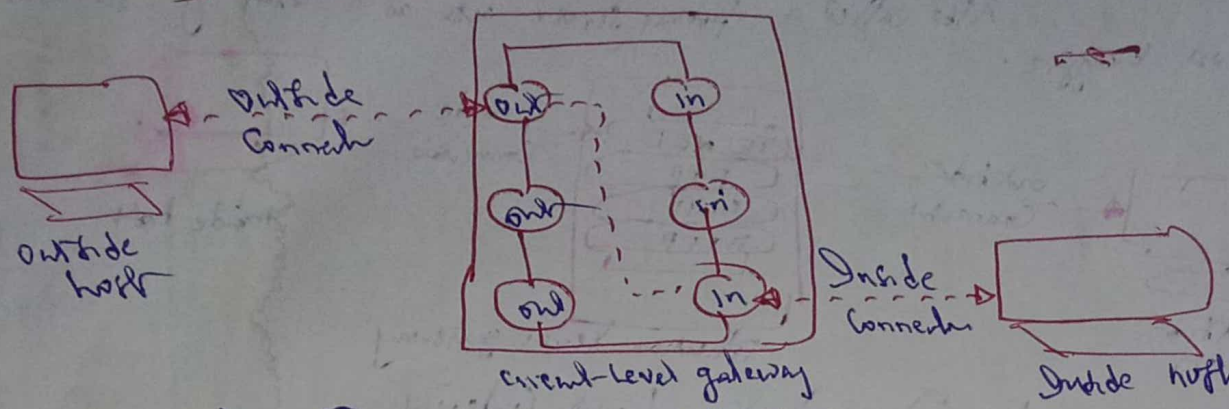
The user contacts the gateway using a TCP/IP application, such as TELNET or FTP, and the gateway asks the user for the name of the remote host to be accessed. When the user responds and provides a valid user ID and authentication info, the gateway contacts the application on the remote host and relays TCP segments containing the application data b/w the two end points.

If the gateway doesn't implement the proxy code for a specific application, the service is not supported and cannot be forwarded across the firewall. Further, the gateway can be configured to support only specific features of an application that the n/w administrator considers acceptable while denying all other features.

This gateway is more secure than packet filter. It only scrutinize few allowable applications. It is easy to log and audit all incoming traffic at the application level.

A prime disadvantage is the additional processing overhead on each connection. Gateway must examine and forward all traffic in both directions.

③ Circuit-Level Gateway [Proxy Server]



by ③ Circuit-Level Gateway

Circuit-Level firewall typically operate at the Session level of the OSI model by verifying TCP connection and Sessions. This firewall do not check actual Content.

Circuit-Level gateway works at the Session layer of the OSI model, or as a "thin-layer" b/w the application layer and the transport layer of the TCP/IP stack. They monitor TCP handshaking b/w packets to determine whether a requested session is legitimate

It helps in providing the Security b/w UDP and TCP using the connection.

Circuit level gateway doesn't filter individual packets It provides session-level control over n/w traffic: of operate at a higher level layer

This can be a standalone system or it can be a specialized function performed by an application-level gateway for certain application

It doesn't permit an end-to-end TCP connection rather the gateway setsup two TCP connection, one b/w itself and a TCP user on an inner host and one b/w 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 connection will be allowed

A typical use of circuit-level gateway is a situation in which the system administrator trusts the internal user. The gateway can be configured to support application-level or proxy service on inbound connections and circuit level functions for outbound connections