Unit I Cryptographic Tools

Definition of Computer Security

Computer Security:

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

This definition introduces three key objectives that are at the heart of computer security:

1. **Confidentiality:** This term covers two related concepts:

Data confidentiality: Assures that private or confidential information is not made available or disclosed to unauthorized individuals.

Privacy:

Assures that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed.

2. **Integrity:** This term covers two related concepts:

Data integrity:

Assures that information and programs are changed only in a specified and authorized manner.

System integrity:

Assures that a system performs its intended function in an unimpaired manner, free from

deliberate or inadvertent unauthorized manipulation of the system.

3. **Availability:**

Assures that systems work promptly and service is not denied to authorized users.

Confidentiality with Symmetric Encryption:

The universal technique for providing confidentiality for transmitted or stored data is symmetric encryption.

Symmetric Encryption:

Symmetric encryption also referred to as conventional encryption or single-key encryption was the only type of encryption in use prior to the introduction of public-key encryption. For example to present-day diplomatic, military, and commercial users, have used symmetric encryption for secret communication.

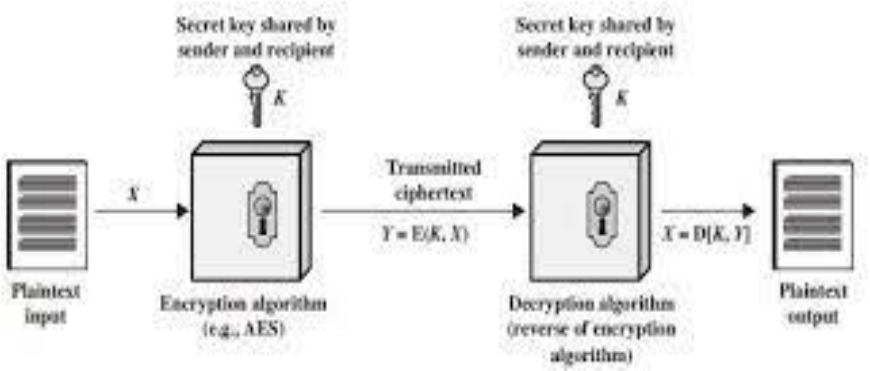


Fig1: Simplified Model of Symmetric Encryption

A symmetric encryption scheme has five ingredients (Figure 1)

Plaintext: This is the original 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 is also input to the encryption algorithm. The exact substitutions and transformations performed by the algorithm depend on the key.
- **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different cipher texts.

• **Decryption algorithm**: This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext.

There are two requirements for secure use of symmetric encryption:

1. We need a strong encryption algorithm. At a minimum, we would like the algorithm to be such that an opponent who knows the algorithm and has access to one or more cipher texts would be unable to decipher the cipher text or figure out the key. This requirement is usually stated in a stronger form.

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 algorithm, all communication using this key is readable.

There are two general approaches to attacking a symmetric encryption scheme.

1. The first attack is known as **cryptanalysis**. Cryptanalytic attacks rely on the nature of the algorithm plus perhaps some knowledge of the general characteristics of the plaintext or even some sample plaintext-cipher text pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used. If the attack succeeds in deducing the key, the effect is catastrophic: All future and past messages encrypted with that key are compromised.

2.The second method, known as the **brute-force attack**, is to try every possible key on a piece of cipher text until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success. It is important to note that there is more to a brute-force attack than simply running through all possible keys. Thus, to supplement the brute-force approach, some degree of knowledge about the expected plaintext is needed.

Symmetric Encryption Algorithms:

Block cipher and **stream cipher** are members of the family of **symmetric key ciphers,** essentially encryption techniques used for directly transforming the **plaintext** into **ciphertext**.

Block cipher

A block cipher processes the plaintext input in fixed-size blocks and produces a block of ciphertext of equal size for each plaintext block. The algorithm processes longer plaintext amounts as a series of fixed-size blocks. The most important symmetric algorithms, all of which are block ciphers, are the Data Encryption Standard (DES), triple DES, and the Advanced Encryption Standard.

DES :- Data Encryption Standard (DES) adopted in 1977 by the National Bureau of Standards. DES takes a plaintext block of 64 bits and a key of 56 bits, to produce a ciphertext block of 64 bits.

Triple DES :- The life of DES was extended by the use of triple DES (3DES), which involves repeating the basic DES algorithm three times, using either two or three unique keys, for a key size of 112 or 168 bits. 3DES was first standardized for use in financial applications

3DES has two attractions :- First, it uses 168-bit key length, which overcomes the vulnerability to brute-force attack of DES. Second, the algorithm in 3DES is required more scrutiny than any other encryption algorithm over a longer period of time, due to that effective cryptanalytic attack is not possible.

Advanced Encryption Standard

Advanced Encryption Standard (AES), which should have a security strength equal to or better than 3DES and significantly improved efficiency.

AES must be a symmetric block cipher with a block length of 128 bits and support for key lengths of 128, 192, and 256 bits. Evaluation criteria included security, computational

efficiency, memory requirements, hardware and software suitability, and flexibility.

The output of the generator, called a keystream, is combined one byte at a time with the plaintext stream using the bitwise exclusiveOR (XOR) operation

Stream cipher

A stream cipher processes the input elements continuously, producing output one element at a time, as it goes along. i.e. In this structure a key is input to a pseudorandom bit generator that produces a stream of 8-bit numbers that are apparently random.

The primary advantage of a stream cipher is that stream ciphers are almost always faster and use far less code than do block ciphers. The advantage of a block cipher is that you can reuse keys. For applications that require encryption/decryption of a stream of data, such as over a data communications channel or a browser/Web link, a stream cipher might be the better alternative. For applications that deal with blocks of data, such as file transfer, e-mail, and database, block ciphers may be more appropriate. However, either type of cipher can be used in virtually any application.

Message Authentication and Hash Functions:

Encryption protects against passive attack. A different requirement is to protect against active attack. Protection against such attacks is known as message or data authentication.

A message, file, document, or other collection of data is said to be authentic when it is genuine and came from its alleged source. Message or data authentication is a procedure that allows communicating parties to verify that received or stored messages are authentic. The two important aspects are to verify that the contents of the message have not been altered and that the source is authentic.

Authentication Using Symmetric Encryption:

If we assume that only the sender and receiver share a key (which is as it should be), then only the genuine sender would be able to encrypt a message successfully for the other participant, provided the receiver can recognize a valid message. If the message includes an error-detection code and a sequence number, the receiver is assured that no alterations have been made and that sequencing is proper. If the message also includes a timestamp, the receiver is assured that the message has not been delayed beyond that normally expected for network transit.

Message Authentication without Message Encryption:

This approach does not encrypt the message and due to that message confidentiality is not provided.

There are three situations in which message authentication without confidentiality is preferable:

1. There are a number of applications in which the same message is broadcast to a number of destinations. It is cheaper and more reliable to have only one destination responsible for monitoring authenticity. Thus, the message must be broadcast in plaintext with an associated message authentication tag. The responsible system performs authentication.

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, with messages being chosen at random for checking.

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. However, if a message authentication tag were attached to the program, it could be checked whenever assurance is required of the integrity of the program.

Message Authentication Code:

One authentication technique involves the use of a secret key to generate a small block of data, known as a message authentication code that is appended to the message. The process of Message Authentication Code (MAC) involves the following steps (Figure 2):

1. Message preparation: The sender prepares the message that needs to be authenticated. This could be any data that needs to be sent, such as a file, document, or email.
2. Key generation: A secret key is generated by the sender that will be used to create the MAC. The key should be strong and kept confidential to ensure the security of the system.
3. MAC generation: The sender generates a MAC using a cryptographic hash function and the secret key. The hash function generates a fixed-size value from the message, and the secret key is used to create a tag that is appended to the message. The resulting MAC is unique to the message and the secret key.
4. Sending message and MAC: The sender sends the message and the MAC to the receiver. The receiver can use the MAC to verify the authenticity of the message.
5. MAC verification: The receiver receives the message and the MAC. The receiver then computes the MAC using the same algorithm and secret key as the sender. The receiver compares the computed MAC to the received MAC. If the MACs match, the receiver can be confident that the message is authentic and has not been altered in transit.
6. Message processing: If the MAC verification is successful, the receiver processes the message as required. If the MAC verification fails, the receiver rejects the message.

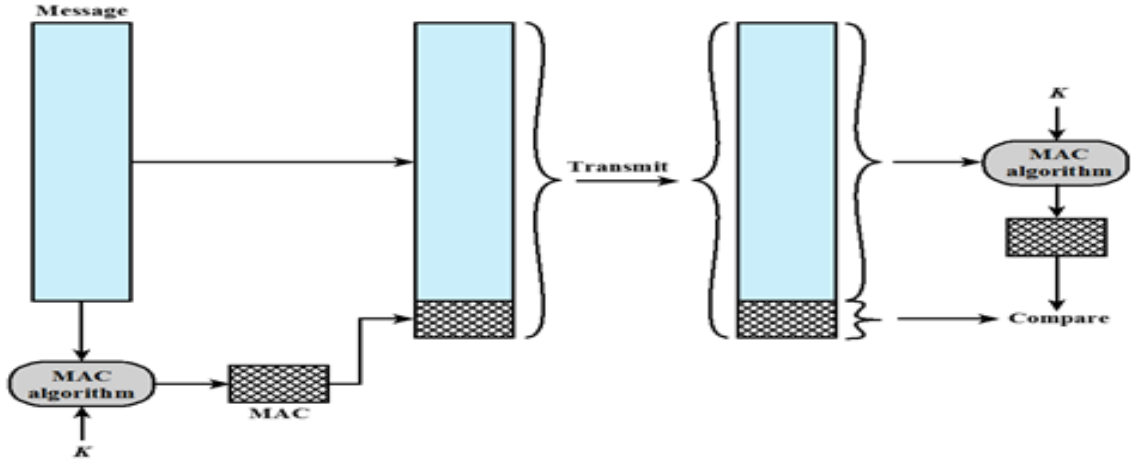


Fig 2 : Message authentication using message authentication code

One-Way Hash Function:

An alternative to the message authentication code is the one-way hash function. With the message authentication code, a hash function accepts a variable-size message M as input and produces a fixed-size message digest. Typically, the message is padded out to an integer multiple of some fixed length (e.g. 1024 bits) and the padding includes the value of the length of the original message in bits. The length field is a security measure to increase the difficulty for an attacker to produce an alternative message with the same hash value.

A hash function does not take a secret key as input. Three ways in which the message can be authenticated using a hash function. The message digest can be encrypted using symmetric encryption. If it is assumed that only the sender and receiver share the encryption key, then authenticity is assured. The message digest can also be encrypted using public-key encryption. The public-key approach has two advantages: It provides a digital signature as well as message authentication; and it does not require the distribution of keys to communicating parties.

But an even more common approach is the use of a technique that avoids encryption.

Security of Hash Functions As with symmetric encryption, there are two approaches to attacking a secure hash function:

cryptanalysis and brute-force attack. As with symmetric encryption algorithms, cryptanalysis of a hash function involves exploiting logical weaknesses in the algorithm. The strength of a hash function against brute-force attacks depends solely on the length of the hash code produced by the algorithm.

Secure Hash Function

The one-way hash function, or secure hash function, is important not only in message authentication but in digital signatures.

Hash Function Requirements

The purpose of a hash function is to produce a "fingerprint" of a file, message, or other block of data. To be useful for message authentication, a hash function H must have the following properties:

1. H can be applied to a block of data of any size.
2. H produces a fixed-length output.
3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. For any given code h, it is computationally infeasible to find x such that H(x) = h. A hash function with this property is referred to as one-way or preimage resistant.
5. For any given block x, it is computationally infeasible to find y ≠ x with H(y) = H(x). A hash function with this property is referred to as second preimage resistant. This is sometimes referred to as weak collision resistant.
6. It is computationally infeasible to find any pair (x, y) such that H(x) = H(y). A hash function with this property is referred to as collision resistant. This is sometimes referred to as strong collision resistant.

If all these properties are satisfied then we can say that it is strong hash function.

Secure HASH Function Algorithms :

The most widely used hash function is Secure Hash Algorithm (SHA). SHA was developed by the National Institute of Standards and Technology (NIST). SHA-1 produces a hash value of 160 bits.

Three new versions of SHA are defined, with hash value lengths of 256, 384, and 512 bits, known as SHA-256, SHA-384, and SHA-512. These new versions, collectively known as SHA-2. SHA-2, particularly the 512-bit version, would appear to provide unassailable security. This new hash function, known as SHA-3, was published in 2012 and is now available as an alternative to SHA-2.

Other Applications of Hash Functions:

Here are two other examples of secure hash function applications:

• **Passwords:** A hash of a password is stored by an operating system rather than the password itself. Thus, the actual password is not retrievable by a hacker who gains access to the password file. In simple terms, when a user enters a password, the hash of that password is compared to the stored hash value for verification. This application requires preimage resistance and perhaps second preimage resistance.

• **Intrusion detection:** Store the hash value for a file, H (F), for each file on a system and secure the hash values (e.g., on a CD-R that is kept secure). One can later determine if a file has been modified by recomputing H(F). An intruder would need to change F without changing H (F). This application requires weak second preimage resistance.

Public-Key Encryption:

Public-key encryption uses in message authentication and key distribution. Public key encryption, or public key cryptography, is a method of encrypting data with two different keys and making one of the keys, the public key, available for anyone to use. The other key is known as the private key.

When the two parties communicate to each other to transfer the intelligible or sensible message, referred to as plaintext, is converted into apparently random nonsense for security purpose referred to as **ciphertext**.

Encryption:

The process of changing the plaintext into the ciphertext is referred to as encryption.The encryption process consists of an algorithm and a key. The key is a value independent of the plaintext.

The security of conventional encryption depends on the major two factors:

The Encryption algorithm

Secrecy of the key.

Once the ciphertext is produced, it may be transmitted. The Encryption algorithm will produce a different output depending on the specific key being used at the time. Changing the key changes the output of the algorithm.

Once the ciphertext is produced, it may be transmitted. Upon reception, the ciphertext can be transformed back to the original plaintext by using a decryption algorithm and the same key that was used for encryption.

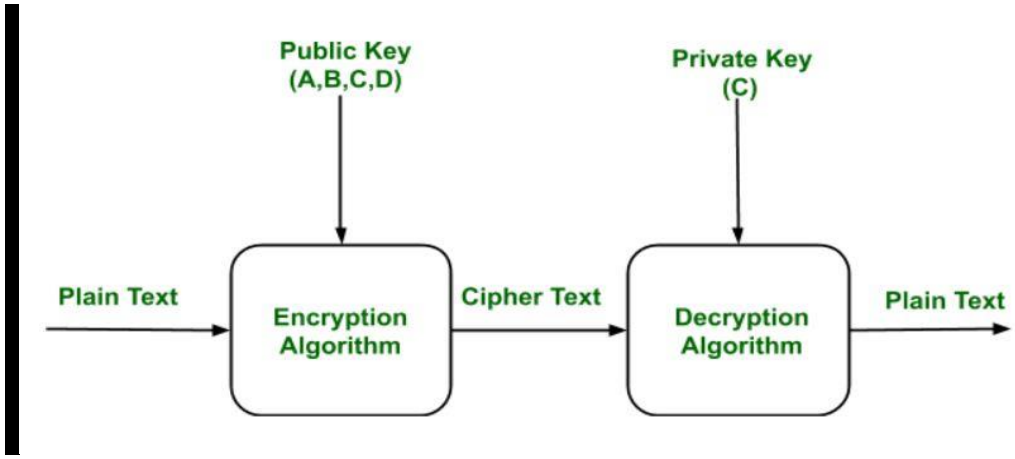
Asymmetric is a form of Cryptosystem in which encryption and decryption are performed using different keys-Public key (known to everyone) and Private Key (Secret key). This is known as Public Key Encryption.

Characteristics of Public Encryption key:

1. Public key Encryption is important because it is infeasible to determine the decryption key given only the knowledge of the cryptographic algorithm and encryption key.
2. Either of the two key (Public and Private key) can be used for encryption with other key used for decryption.
3. Due to Public key cryptosystem, public keys can be freely shared, allowing users an easy and convenient method for encrypting content and verifying digital signatures, and private keys can be kept secret, ensuring only the owners of the private keys can decrypt content and create digital signatures.

4. The most widely used public-key cryptosystem is RSA (Rivest–Shamir–Adleman). The difficulty of finding the prime factors of a composite number is the backbone of RSA.

Example:
Public keys of every user are present in the Public key Register. If B wants to send a confidential message to C, then B encrypt the message using C Public key. When C receives the message from B then C can decrypt it using its own Private key. No other recipient other than C can decrypt the message because only C know C's private key.



Components of Public Key Encryption:
Plain Text:
This is the message which is readable or understandable. This message is given to the Encryption algorithm as an input.
Cipher Text:
The cipher text is produced as an output of Encryption algorithm. We cannot simply understand this message.
Encryption Algorithm:
The encryption algorithm is used to convert plain text into cipher text.
Decryption Algorithm:
It accepts the cipher text as input and the matching key (Private Key or Public key) and produces the original plain text
Public and Private Key:

One key either Private key (Secret key) or Public Key (known to everyone) is used for encryption and other is used for decryption.

Applications of the Public Key Encryption:

Encryption/Decryption:

Confidentiality can be achieved using Public Key Encryption. In this the Plain text is encrypted using receiver public key. This will ensures that no one other than receiver private key can decrypt the cipher text.

Digital signature:

Digital signature is for senders authentication purpose. In this sender encrypt the plain text using his own private key. This step will make sure the authentication of the sender because receiver can decrypt the cipher text using sender's pubic key only.

Key exchange:

This algorithm can use in both Key-management and securely transmission of data.

Asymmetric Encryption Algorithms

RSA

One of the first public-key schemes was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT.It is the most widely accepted and implemented approach to public-key encryption. RSA is a block cipher in which the plaintext and ciphertext are integers between 0 and n – 1 for some n.
Currently, a 1024-bit key size (about 300 decimal digits) is considered strong enough for virtually all applications.

Diffie-Hellman Key Agreement

The purpose of the algorithm is to enable two users to securely reach agreement about a shared secret that can be used as a secret key for subsequent symmetric encryption of messages. The algorithm itself is limited to the exchange of the keys.

Digital Signature Standard

The DSS makes use of SHA-1 and presents a new digital signature technique, the Digital Signature Algorithm (DSA). The DSS uses an algorithm that is designed to provide only the digital signature function. Unlike RSA, it cannot be used for encryption or key exchange.

Elliptic Curve Cryptography

The principal attraction of ECC compared to RSA is that it appears to offer equal security for a far smaller bit size, thereby reducing processing overhead.

Digital Signatures and Key Management:-

Public-key algorithms are used in a variety of applications. In broad terms, these applications fall into two categories: digital signatures, and various techniques to do with key management and distribution. With respect to key management and distribution, there are at least three distinct aspects to the use of public-key encryption in this regard:
The secure distribution of public keys.
The use of public-key encryption to distribute secret keys.
The use of public-key encryption to create temporary keys for message encryption.

The overview of digital signatures and the various types of key management and distribution are as follows.

Digital Signature:-

Public-key encryption can be used for authentication in Digital signature. In the physical world, it is common to use handwritten signatures on handwritten or typed messages. They are used to bind signatory to the message.

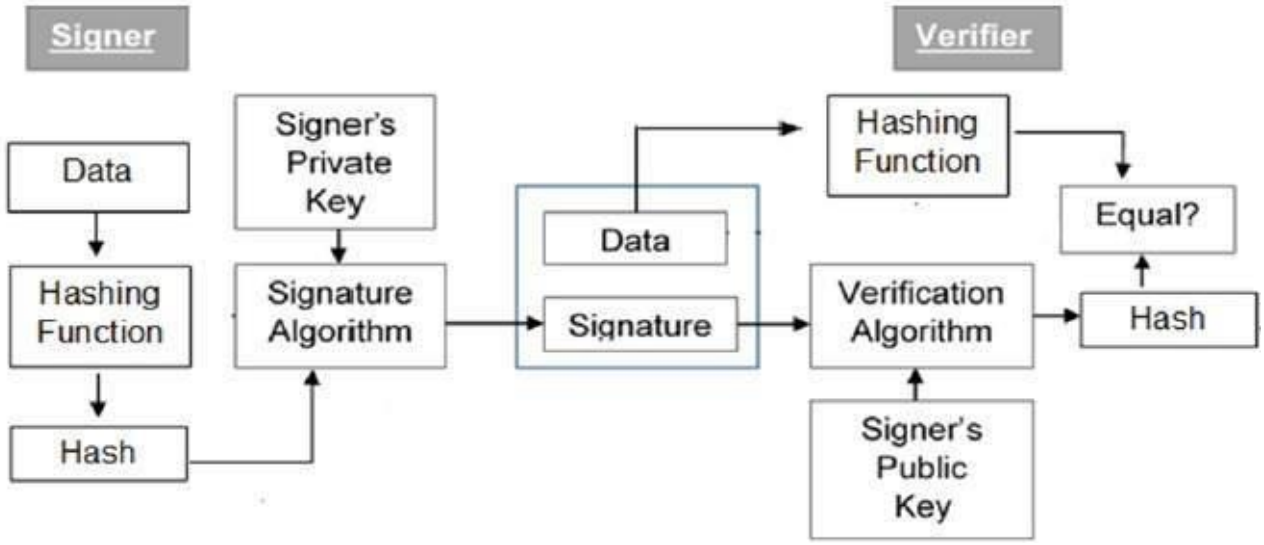
Similarly, a digital signature is a technique that binds a person/entity to the digital data. This binding can be independently verified by receiver as well as any third party.

Digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer.

In real world, the receiver of message needs assurance that the message belongs to the sender and he should not be able to repudiate the origination of that message. This requirement is very crucial in business applications, since likelihood of a dispute over exchanged data is very high.

Model of Digital Signature

As mentioned earlier, the digital signature scheme is based on public key cryptography. The model of digital signature scheme is depicted in the following illustration −



The following points explain the entire process in detail −

- Each person adopting this scheme has a public-private key pair.
- Generally, the key pairs used for encryption/decryption and signing/verifying are different. The private key used for signing is referred to as the signature key and the public key as the verification key.
- Signer feeds data to the hash function and generates hash of data.
- Hash value and signature key are then fed to the signature algorithm which produces the digital signature on given hash. Signature is appended to the data and then both are sent to the verifier.

- Verifier feeds the digital signature and the verification key into the verification algorithm. The verification algorithm gives some value as output.
- Verifier also runs same hash function on received data to generate hash value.
- For verification, this hash value and output of verification algorithm are compared. Based on the comparison result, verifier decides whether the digital signature is valid.
- Since digital signature is created by 'private' key of signer and no one else can have this key; the signer cannot repudiate signing the data in future.

Public-Key Certificates:-

A public key certificate is a digitally signed document that serves to validate the sender's authorization and name. It uses a cryptographic structure that binds a public key to an entity, such as a user or organization.

A certificate consists of a public key plus a user ID of the key owner, with the whole block signed by a trusted third party. The certificate also includes some information about the third party plus an indication of the period of validity of the certificate. Typically, the third party is a certificate authority (CA) that is trusted by the user community, such as a government agency or a financial institution. A user can present his or her public key to the authority in a secure manner and obtain a signed certificate. The user can then publish the certificate. Anyone needing this user's public key can obtain the certificate and verify that it is valid by means of the attached trusted signature.

How does a public key certificate work?

Public key certificates form a part of a public key infrastructure (PKI) system that uses encryption technology to secure messages and data. A public key certificate uses a pair of encryption keys, one public and one private. The public key is made available to anyone who wants to verify the identity of

the certificate holder, while the private key is a unique key that is kept secret. This enables the certificate holder to digitally sign documents, emails and other information without a third party being able to impersonate them. The four main components of PKI are public key encryption, trusted third parties such as the CA, the registration authority and the certificate database or store.

The key steps can be summarized as follows:

1. User software (client) creates a pair of keys: one public and one private.
2. Client prepares an unsigned certificate that includes the user ID and user's public key.
3. User provides the unsigned certificate to a CA in some secure manner. This might require a face-to-face meeting, the use of registered e-mail, or happen via a web form with e-mail verification.
4. CA creates a signature as follows:
    a. CA uses a hash function to calculate the hash code of the unsigned certificate. A hash Function is one that maps a variable-length data block or message into a fixed-length value called a hash code.
    b. CA encrypts the hash code with the CA's private key.
5. CA attaches the signature to the unsigned certificate to create a signed certificate.
6. CA returns the signed certificate to client.
7. Client may provide the signed certificate to any other user.
8. Any user may verify that the certificate is valid as follows:
    a. User calculates the hash code of certificate (not including signature).
    b. User decrypts the signature using CA's known public key.
    c. User compares the results of (a) and (b). If there is a match, the certificate is valid.

Digital Envelopes:

Digital envelope, is used to protect a message without needing to first arrange for sender and receiver to have the same secret key. The technique is referred to as a digital envelope. Suppose Bob wishes to send a confidential message to Alice, but they do not share a symmetric secret key. Bob does the following:

1. Prepare a message.
2. Generate a random symmetric key that will be used this one time only.
3. Encrypt that message using symmetric encryption the one-time key.
4. Encrypt the one-time key using public-key encryption with Alice's public key.
5. Attach the encrypted one-time key to the encrypted message and send it to Alice.

Random and Pseudorandom Numbers:-

Random numbers play an important role in the use of encryption for various network security applications.

The Use of Random Numbers

A number of network security algorithms based on cryptography make use of random numbers. For example,

- Generation of keys for the RSA public-key encryption algorithm and other public-key algorithms.
- Generation of a stream key for symmetric stream cipher.
- Generation of a symmetric key for use as a temporary session key or in creating a digital envelope.
- Session key generation, whether done by a key distribution center or by one of the principals.

The following two criteria are used to validate that a sequence of numbers is random:

- Uniform distribution:

  The distribution of numbers in the sequence should be uniform; that is, the frequency of occurrence of each of the numbers should be approximately the same.
- **Independence:**

No one value in the sequence can be inferred from the others.

Random versus Pseudorandom:

Cryptographic applications typically make use of algorithmic techniques for random number generation. These algorithms are deterministic and therefore produce sequences of numbers that are not statistically random. However, if the algorithm is good, the resulting sequences will pass many reasonable tests of randomness. Such numbers are referred to as pseudorandom numbers.