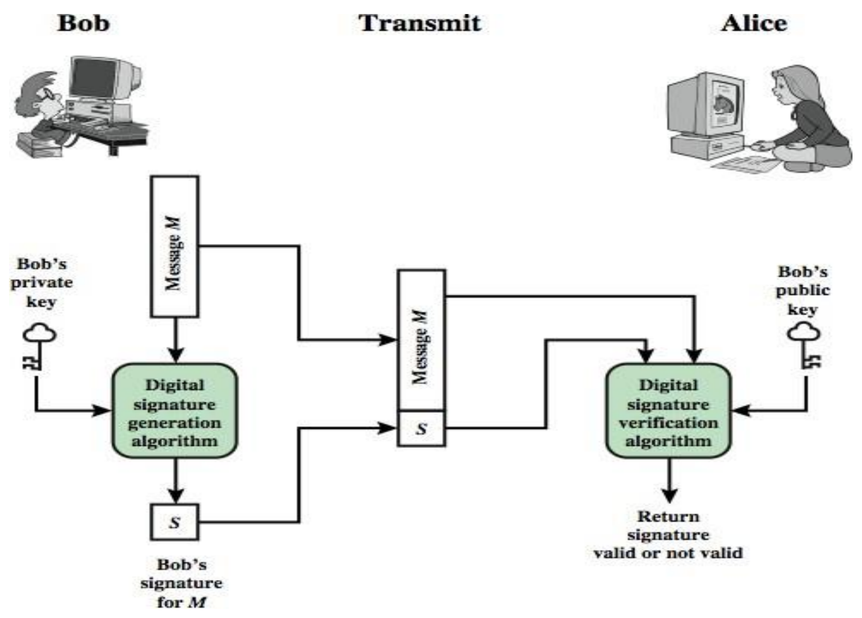
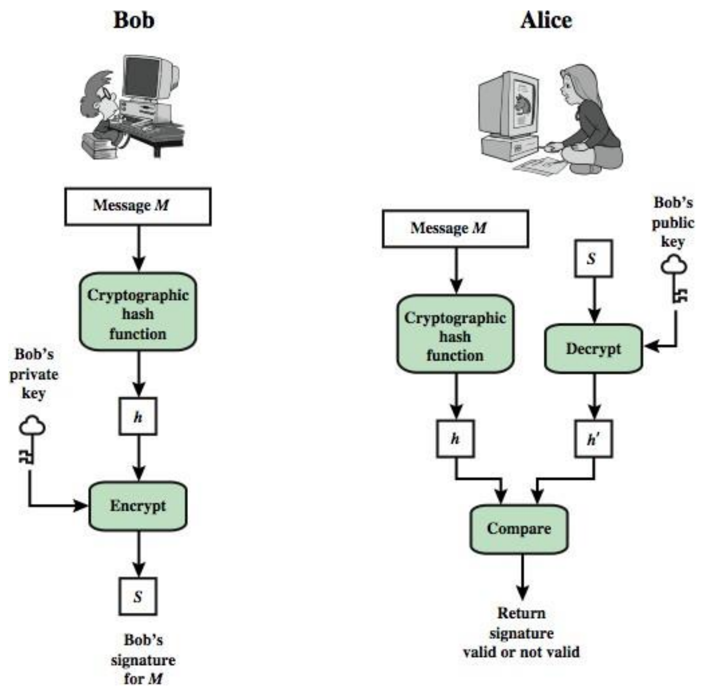
A **digital signature** is an authentication mechanism that enables the creator of a message to attach a code that acts as a signature. Typically the signature is formed by taking the hash of the message and encrypting the message with the creator's private key. The signature guarantees the source and integrity of the message.

Generic Model of the Digital Signature


Bob
Transmit
Alice
Message M
Bob's private key
Digital signature generation algorithm
S
Bob's signature for M
Message M
S
Bob's public key
Digital signature verification algorithm
Return signature valid or not valid

- Figure above is a generic model of the process of making and using digital signatures.
- Bob can sign a message using a digital signature generation algorithm.
- The inputs to the algorithm are the message and Bob's private key.
- Any other user, say Alice, can verify the signature using a verification algorithm, whose inputs are the message, the signature, and Bob's public key.
- In simplified terms, the essence of the digital signature mechanism is shown in Figure below.



Needs of Digital Signature

- Message authentication protects two parties who exchange messages from any third party.
- However, it does not protect the two parties against each other.
- Several forms of dispute between the two are possible.
  1. Mary may forge a different message and claim that it came from John. Mary would simply have to create a message and append an authentication code using the key that John and Mary share.
  2. John can deny sending the message. Because it is possible for Mary to forge a message, there is no way to prove that John did in fact send the message.
- Both scenarios are of legitimate concern.
- The sender pretends that the message was never sent.
- In situations where there is not complete trust between sender and receiver, something more than authentication is needed.
- The most attractive solution to this problem is the digital signature.

Properties of Digital Signature

- The digital signature must have the following properties:
  1. It must verify the author and the date and time of the signature.
  2. It must authenticate the contents at the time of the signature.
  3. It must be verifiable by third parties, to resolve disputes.
- Thus, the digital signature function includes the authentication function.

Security of Digital Signature

- Following types of attacks can be possible on digital signature.
- Here A denotes the user whose signature method is being attacked, and C denotes the attacker.
  1. **Key-only attack:** C only knows A’s public key.
  2. **Known message attack:** C is given access to a set of messages and their signatures.
  3. **Generic chosen message attack:** C chooses a list of messages before attempting to breaks A’s signature scheme, independent of A’s public key. C then obtains from A valid signatures for the chosen messages. The attack is generic, because it does not depend on A’s public key; the same attack is used against everyone.
  4. **Directed chosen message attack:** Similar to the generic attack, except that the list of messages to be signed is chosen after C knows A’s public key but before any signatures are seen.
  5. **Adaptive chosen message attack:** C is allowed to use A as an “oracle.” This means the A may request signatures of messages that depend on previously obtained message–signature pairs.
- If anyone success at breaking a signature scheme can do any of the following with a non-negligible probability:
  1) **Total break:** C determines A’s private key.
  2) **Universal forgery:** C finds an efficient signing algorithm that provides an equivalent way of constructing signatures on arbitrary messages.
  3) **Selective forgery:** C forges a signature for a particular message chosen by C.
  4) **Existential forgery:** C forges a signature for at least one message. C has no control over the message. Consequently, this forgery may only be a minor nuisance to A.

Digital Signature Requirements

- On the basis of the properties and attacks just discussed, we can formulate the following requirements for a digital signature.
  1) The signature must be a bit pattern that depends on the message being signed.
  2) The signature must use some information unique to the sender to prevent both forgery and denial.
  3) It must be relatively easy to produce the digital signature.
  4) It must be relatively easy to recognize and verify the digital signature.
  5) It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message.
  6) It must be practical to retain a copy of the digital signature in storage.

Direct Digital Signature

- The term **direct digital signature** refers to a digital signature scheme that involves only the communicating parties (source, destination).
- It is assumed that the destination knows the public key of the source.
- Confidentiality can be provided by encrypting the entire message plus signature with a shared secret key (symmetric encryption).
- Note that it is important to perform the signature function first and then an outer confidentiality function.
- In case of dispute, some third party must view the message and its signature.
- If the signature is calculated on an encrypted message, then the third party also needs access to the decryption key to read the original message.
- However, if the signature is the inner operation, then the recipient can store the plaintext message and its signature for later use in dispute resolution.
- The validity of the scheme just described depends on the security of the sender's private key.
- If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and that someone else forged his or her signature.
- Another threat is that some private key might actually be stolen from X at time T.
- The opponent can then send a message signed with X's signature and stamped with a time before or equal to T.
- The universally accepted technique for dealing with these threats is the use of a digital certificate and certificate authorities.

Elgamal Digital Signature Scheme

- The ElGamal signature scheme involves the use of the private key for encryption and the public key for decryption.
- As with ElGamal encryption, the global elements of **ElGamal digital signature** are a prime number **q** and **α**, which is a primitive root of **q**.
- User A generates a private/public key pair as follows.
  1. Generate a random integer X_A, such that $1 < X_A < q - 1$.
  2. Compute $Y_A = \alpha^{XA} mod\ q$.
  3. A's private key is X_A; A's pubic key is $\{q, \alpha, Y_A\}$.
- To sign a message **M**, user A first computes the hash $m = H(M)$, such that **m** is an integer in the range $0 < m < q - 1$.

- A then forms a digital signature as follows.
  1) Choose a random integer K such that $1 < K < q - 1$ and$\gcd(K, q - 1) = 1$. That is, K is relatively prime to q-1.
  2) Compute $S_1 = \alpha^K mod\ q$. Note that this is the same as the computation of C_1 for ElGamal encryption.
  3) Compute $K^{-1} mod\ (q - 1)$.That is, compute the inverse of K modulo q-1.
  4) Compute $S_2 = K^{-1}(m - X_A S_1) mod (q - 1)$.
  5) The signature consists of the pair (S_1, S_2).
- Any user B can verify the signature as follows.
  1. Compute $V_1 = \alpha^m mod\ q$.
  2. Compute $V_2 = (Y_A)^{S1}(S_1)^{S2}\ mod\ q$.
- The signature is valid if $V_1 = V_2$.

Example

- let us start with the q = 19 and α=10
- Alice generates a key pair as follows:
  1. Alice chooses $X_A = 16$.
  2. Then $Y_A = \alpha^{XA} mod\ q = 10^{16} mod\ 19 = 4$.
  3. Alice's private key is 16; Alice's pubic key is $\{q, \alpha, Y_A\} = \{19, 10, 4\}$.
- Suppose Alice wants to sign a message with hash value $m = 14$.
  1) Alice chooses $K = 5$, which is relatively prime to $q - 1 = 18$.
  2) $S_1 = \alpha^K mod\ q = 10^5 mod\ 19 = 3$.
  3) $K^{-1} mod\ (q - 1) = 5^{-1} mod\ (19 - 1) = 11$.
  4) $S_2 = K^{-1}(m - X_A S_1) mod (q - 1) = 11(14 - (16)(3)) mod (19 - 1) = -347\ mod\ 18 = 4$.
- Bob can verify the signature as follows.
  1. $V_1 = \alpha^m mod\ q = 10^{14} mod\ 19 = 16$.
  2. $V_2 = (Y_A)^{S1}(S_1)^{S2}\ mod\ q = (4)^3(3)^4\ mod\ 19 = 5184\ mod\ 19 = 16$.
- Thus, the signature is valid.

Schnorr Digital Signature Scheme

- Schnorr signature scheme is based on discrete logarithms.
- The Schnorr scheme minimizes the message-dependent amount of computation required to generate a signature.
- The main work for signature generation does not depend on the message and can be done during the idle time of the processor.
- The message-dependent part of the signature generation requires multiplying a 2n-bit integer with an n-bit integer.
- The scheme is based on using a prime modulus **p**, with **p-1** having a prime Factor **q** of appropriate size.
- The first part of this scheme is the generation of a private/public key pair, which consists of the following steps.
  1. Choose primes **p** and **q**, such that **q** is a prime factor of **p-1**.
  2. Choose an integer **α**, such that $\alpha^q = 1\ mod\ p$.The values **α**, **p**, and **q** comprise a global public key that can be common to a group of users.
  3. Choose a random integer **s** with $0 < s < q$.This is the user's private key.
  4. Calculate $v = \alpha^{-s} mod\ p$.This is the user's public key.
- A user with private key **s** and public key **v** generates a signature as follows.

1) Choose a random integer **r** with $0 < r < q$ and compute $x = \alpha^r mod\ p$. This computation is a preprocessing stage independent of the message **M** to be signed.
2) Concatenate the message with **x** and hash the result to compute the value **e**: $e = H(M||x)$.
3) Compute$y = (r + se) mod q$. The signature consists of the pair **(e, y)**.

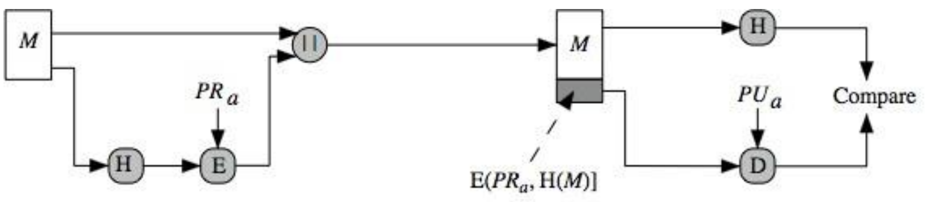
- Any other user can verify the signature as follows.
  1. Compute $x' = \alpha^y v^e mod\ p$.
  2. Verify that $e = H(M||x')$.

Digital Signature Standard

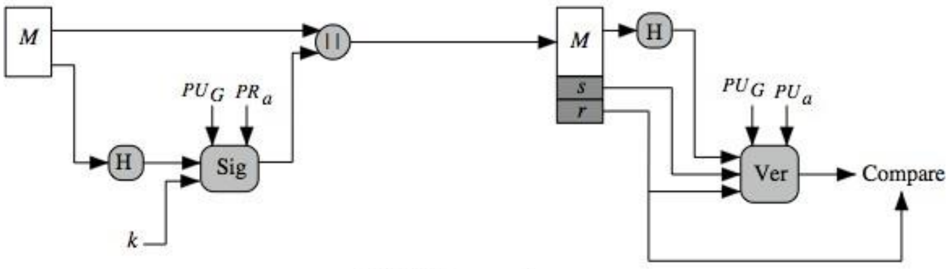
- The National Institute of Standards and Technology (NIST) has published Federal Information Processing Standard FIPS 186, known as the Digital Signature Standard (DSS).
- The DSS makes use of the SHA and presents a new digital signature technique, the **Digital Signature Algorithm (DSA)**.
- Latest version also incorporates digital signature algorithms based on RSA and on elliptic curve cryptography.

The DSS Approach

- 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. Nevertheless, it is a public-key technique.



(a) RSA Approach



(b) DSS Approach

- Figure contrasts the DSS approach for generating digital signatures to that used with RSA.

RSA approach

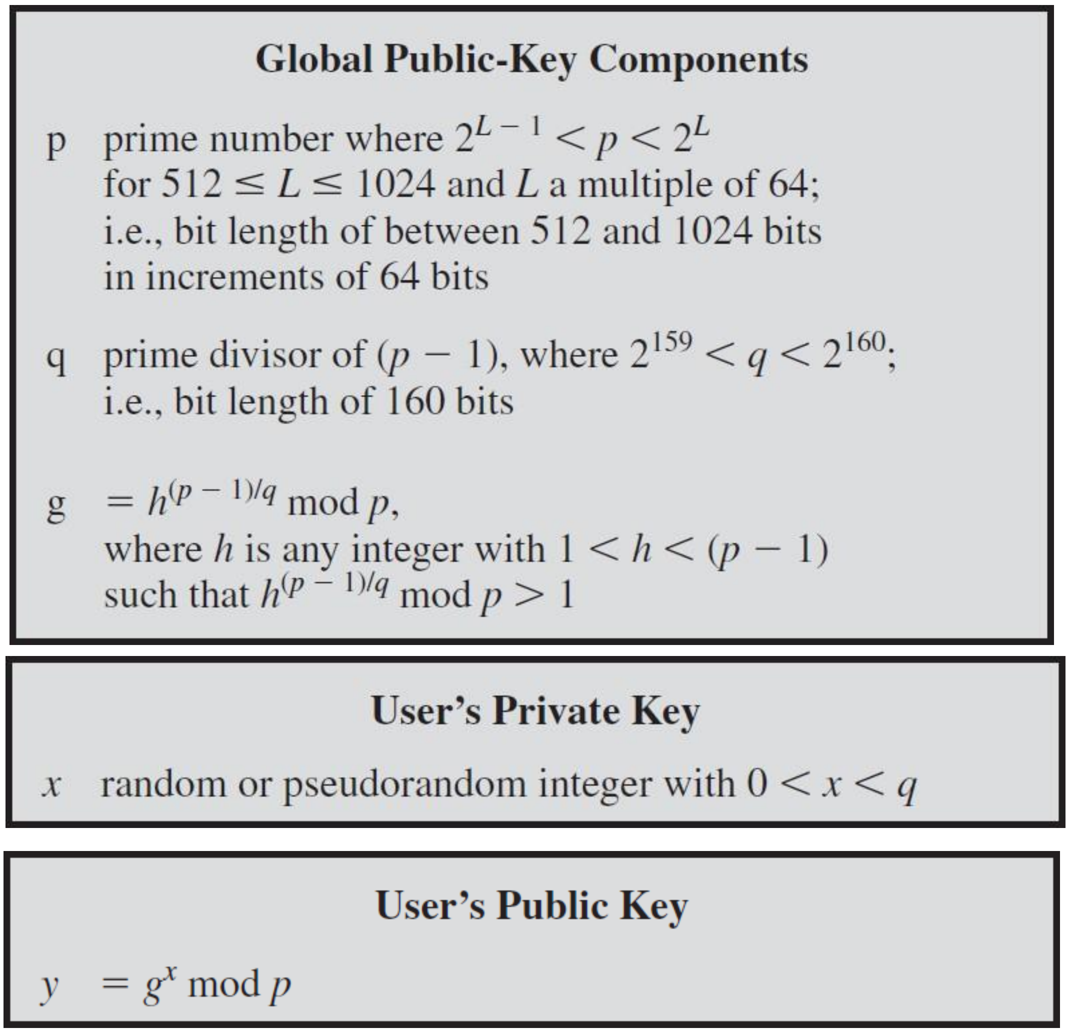
- In the RSA approach, the message to be signed is input to a hash function that produces a secure hash code of fixed length.
- This hash code is then encrypted using the sender’s private key to form the signature.
- Both the message and the signature are then transmitted.
- The recipient takes the message and produces a hash code.
- The recipient also decrypts the signature using the sender’s public key.
- If the calculated hash code matches the decrypted signature, the signature is accepted as valid.

DSS approach

- The DSS approach also makes use of a hash function.
- The hash code is provided as input to a signature function along with a random number k, generated for this particular signature.
- The signature function also depends on the sender's private key (PR_a), and a set of parameters known to a group of communicating principals.
- We can consider this set to constitute a global public key (PU_G).
- The result is a signature consisting of two components, labeled *s* and *r*.
- At the receiving end, the hash code of the incoming message is generated.
- This plus the signature is input to a verification function.
- The verification function also depends on the global public key as well as the sender's public key (PU_a), which is paired with the sender's private key.
- The output of the verification function is a value that is equal to the signature component r, if the signature is valid.
- The signature function is such that only the sender, with knowledge of the private key, could have produced the valid signature.

The Digital Signature Algorithm

- The DSA is based on the difficulty of computing discrete logarithms and is based on schemes originally presented by ElGamal and Schnorr.



Global Public-Key Components

p prime number where $2^{L-1} < p < 2^L$
for $512 \le L \le 1024$ and L a multiple of 64;
i.e., bit length of between 512 and 1024 bits
in increments of 64 bits

q prime divisor of $(p - 1)$, where $2^{159} < q < 2^{160}$;
i.e., bit length of 160 bits

g $= h^{(p-1)/q} \bmod p$,
where h is any integer with $1 < h < (p - 1)$
such that $h^{(p-1)/q} \bmod p > 1$

User's Private Key

x random or pseudorandom integer with $0 < x < q$

User's Public Key

$y = g^x \bmod p$

User's Per-Message Secret Number

k = random or pseudorandom integer with $0 < k < q$

Signing

$r = (g^k \bmod p) \bmod q$

$s = [k^{-1} (H(M) + xr)] \bmod q$

Signature $= (r, s)$

Verifying

$w = (s')^{-1} \bmod q$

$u_1 = [H(M')w] \bmod q$

$u_2 = (r')w \bmod q$

$v = [(g^{u1} y^{u2}) \bmod p] \bmod q$

TEST: $v = r'$

M	= message to be signed
$H(M)$	= hash of M using SHA-1
M', r', s'	= received versions of M, r, s

- Figure summarizes the algorithm.

Global Public-Key Components

- There are three parameters that are public and can be common to a group of users.
- A 160-bit prime number **q** is chosen.
- Next, a prime number **p** is selected with a length between 512 and 1024 bits such that **q** divides **(*p* - 1)**.
- Finally, g is chosen to be of the form $h^{(p-1)/q} mod\ p$, where h is an integer between 1 and **(*p* - 1)** with the restriction that **g** must be greater than 1.

User’s Private Key

- With these numbers in hand, each user selects a private key and generates a public key.
- The private key **x** must be a number from 1 to **(*q* - 1)** and should be chosen randomly.

User’s Public Key

- The public key is calculated from the private key as shown in figure.
- It should be computationally infeasible to determine **x**, from **y.**

User’s Per-Message Secrete Number

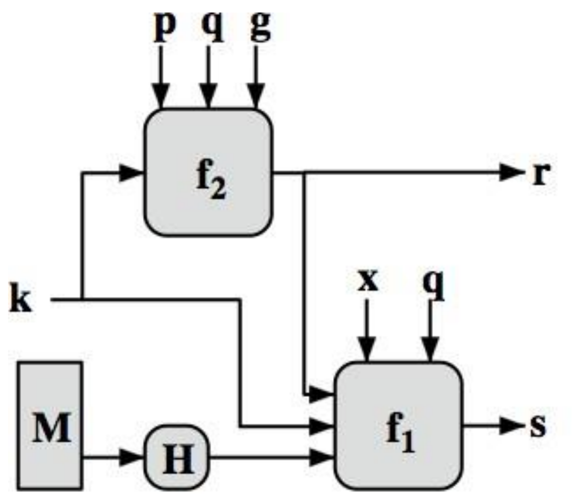
- Random number k is integer with 0 < k <q.

Creating Signature

- To create a signature, a user calculates two quantities, **r** and **s**, that are functions of the public key components **(p, q, g)**, the user’s private key **x**, the hash code of the message H(M), and an additional integer **k** that should be generated randomly and be unique for each signing.

Verification

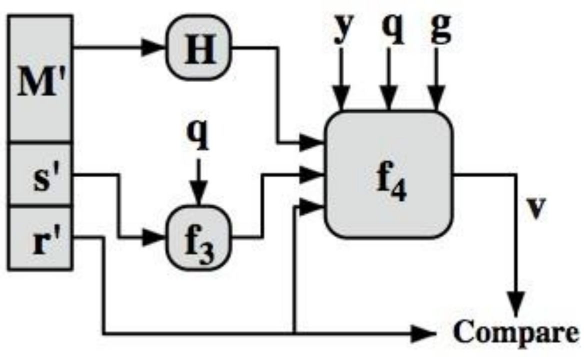
- At the receiving end, verification is performed using the formulas shown in Figure.
- The receiver generates a quantity that is a function of the public key components, the sender’s public key, and the hash code of the incoming message.
- If this quantity matches the component of the signature, then the signature is validated.
- Figure below depicts the functions of signing and verifying.



$$s = f_1(H(M), k, x, r, q) = (k^{-1}(H(M) + xr)) \bmod q$$

$$r = f_2(k, p, q, g) = (g^k \bmod p) \bmod q$$

(a) Signing



$$w = f_3(s', q) = (s')^{-1} \bmod q$$

$$v = f_4(y, q, g, H(M'), w, r')$$

$$= ((g^{(H(M')w) \bmod q}\, y^{r'w \bmod q}) \bmod p) \bmod q$$

(b) Verifying

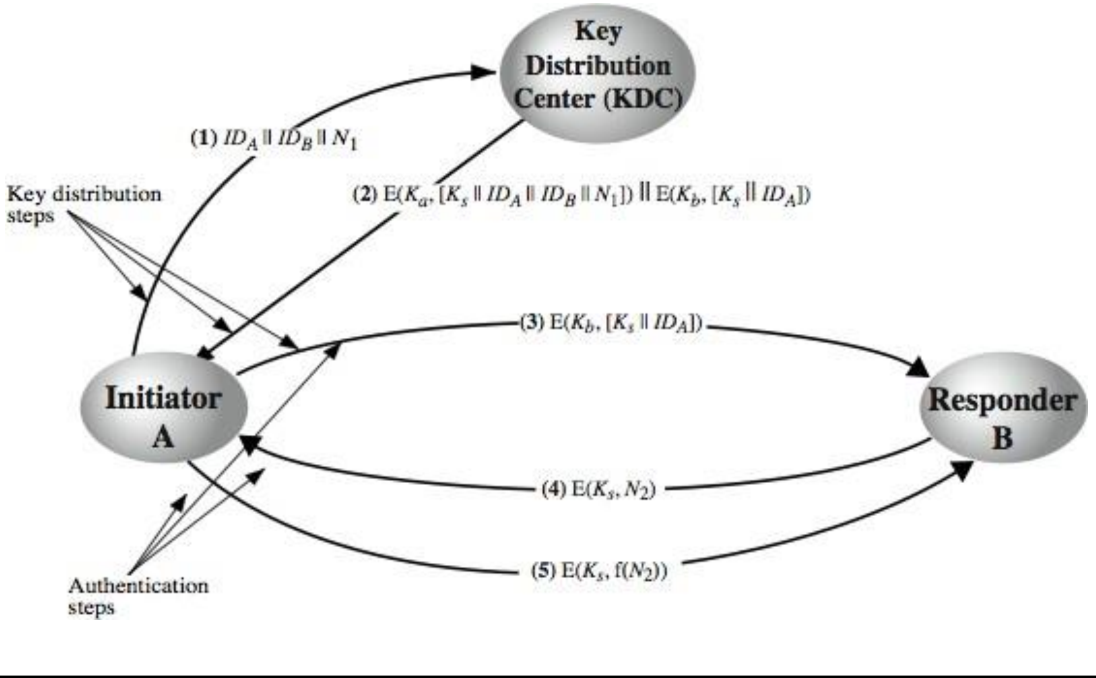
Unit-8 –Key Management and Distribution

Key distribution is the function that delivers a key to two parties who wish to exchange secure encrypted data. Some sort of mechanism or protocol is needed to provide for the secure distribution of keys.

Symmetric Key Distribution Using Symmetric Encryption

- For symmetric encryption two parties must share the same key securely.
- Frequent key changes are usually desirable to limit the amount of data compromised.
- Therefore, the strength of any cryptographic system rests with the *key distribution technique*.
- For two parties A and B, key distribution can be achieved in a number of ways, as follows:
  1. A can select a key and physically deliver it to B.
  2. A third party can select the key and physically deliver it to A and B.
  3. If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
  4. If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B.
- Options 1 and 2 call for manual delivery of a key.
- For **end-to-end encryption** over a network, manual delivery is awkward.
- If end-to-end encryption is done at a network or IP level, then a key is needed for each pair of hosts on the network that wish to communicate.
- Thus, if there are N hosts, the number of required keys is $[N(N-1)]/2$.
- If encryption is done at the application level, then a key is needed for every pair of users or processes that require communication.
- Option 3 is a possibility for either link encryption or end-to-end encryption, but if an attacker ever succeeds in gaining access to one key, then all subsequent keys will be revealed.
- Some variation on option 4 has been widely adopted.
- In this scheme, a key distribution center is responsible for distributing keys to pairs of users (hosts, processes, applications) as needed.
- Each user must share a unique key with the key distribution center for purposes of key distribution.

A Key Distribution Scenario



Key Distribution Center (KDC)
(1) $ID_A \| ID_B \| N_1$
Key distribution steps
(2) $E(K_a, [K_s \| ID_A \| ID_B \| N_1]) \| E(K_b, [K_s \| ID_A])$
(3) $E(K_b, [K_s \| ID_A])$
Initiator A
Responder B
(4) $E(K_s, N_2)$
(5) $E(K_s, f(N_2))$
Authentication steps