ASYMMETRIC CIPHERS

Modular arithmetic

Given any positive integer n and any integer a. if we divide a by n, we get quotient q and a remainder r that obeys the following relationship:

$$a = qn + r \quad 0 \le r < n; \quad q = \lfloor a/n \rfloor$$

Where $\lfloor x \rfloor$ is the largest integer less than or equal to x

Fermat's theorem

Fermat's theorem states the following: if p is Prime and a is a positive integer not divisible by p, then

$$a^{p-1} = 1 \bmod p, \text{ p doesn't divide a}$$

Euler's totient function

Before presenting Euler's theorem we need to introduce an important quantity in number theory referred to as Euler's totient function and written ø(n), where ø(n) is the number of positive integers less than n and relatively prime to n

It should be clear that for a prime number p.

$$ø(p) = p-1$$

Now suppose that we have two prime numbers p and q. then for n=pq

$$ø(n)=ø(pq) = (p-1)(q-1)$$

Euler's theorem

Euler's theorem states that for every a and n that are relatively prime

$$a^{\phi(n)} = 1 \bmod n \qquad \text{– reduces to Fermat's for n prime}$$

Public-Key Cryptography

The radical difference with Public-Key Cryptography is the use of two related keys but with very different roles and abilities. Anyone knowing the public key can encrypt messages or verify signatures, but cannot decrypt messages or create signatures.

Public-key/two-key/asymmetric cryptography involves the use of two keys:

- a public-key, which may be known by anybody, and can be used to encrypt messages, and verify signatures

- a private-key, known only to the recipient, used to decrypt messages, and sign (create) signatures

It is asymmetric because those who encrypt messages or verify signatures cannot decrypt messages or create signatures

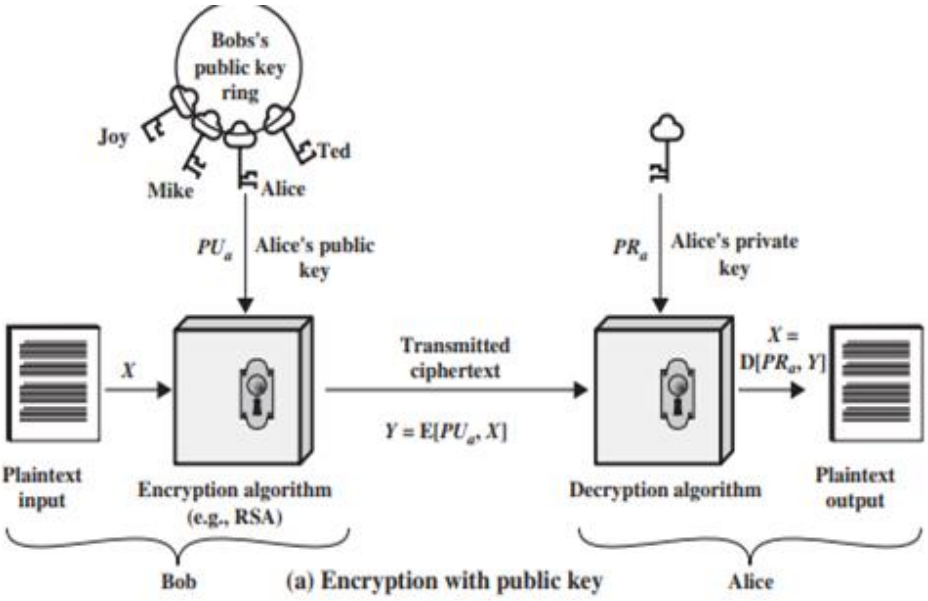


Fig. Public key encryption

Figure above "Public-Key Cryptography", shows that a public-key encryption scheme has six ingredients:

- Plaintext: This is the readable message or 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 a 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 message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.
- Decryption algorithm: This algorithm accepts the ciphertext and the matching key and produces the original plaintext

RSA algorithm

RSA is the best known, and by far the most widely used general public key encryption algorithm, and was first published by Rivest, Shamir & Adleman of MIT in 1978. Since that time RSA has reigned supreme as the most widely accepted and implemented general-purpose approach to public-key encryption. It is based on exponentiation in a finite (Galois) field over integers modulo a prime, using large integers (eg. 1024 bits). Its security is due to the cost of factoring large numbers.

RSA is a public key encryption algorithm based on exponentiation using modular arithmetic

- Key-Generation by each user consists of:
  - Selecting two large primes at random p, q
  - computing their system modulus n=p.q p, q primes, note ø(n)=(p-1)(q-1)
  - Selecting at random the encryption key e

    Where $1<e<ø(n)$, $gcd(e,ø(n))=1$
  - Solve following equation to find decryption key d

    $e.d=1 \bmod ø(n)$ and $0 \le d \le n$
  - publish their public encryption key: KU={e,n}
  - keep secret private decryption key: KR={d,n}

Key Generation

Select p, q	p and q both prime
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e)=1; 1 < e < \phi(n)$
Calculate d	$d=e^{-1} \bmod \phi(n)$
Public key	KU=$\{e,n\}$
Private key	KR=$\{d,n\}$

- Encryption of a message M to obtain ciphertext C is:

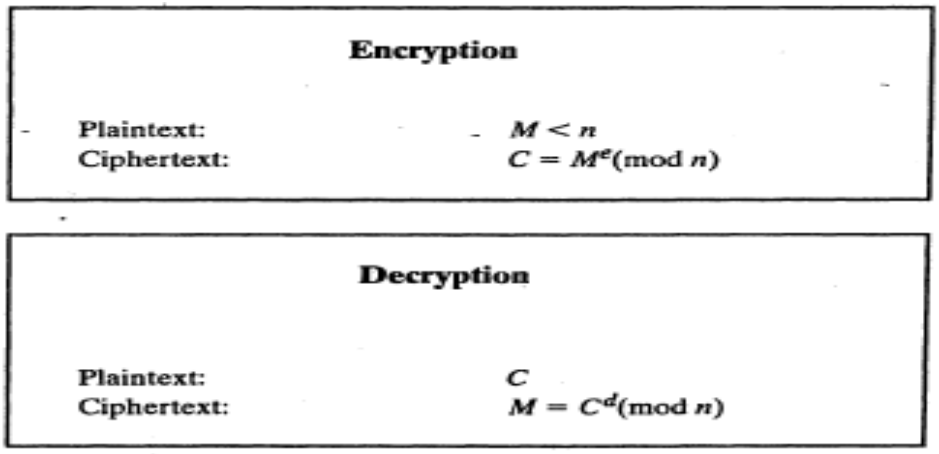
To encrypt a message M the sender obtains public key of recipient PU= {e,n} and computes:

$$C = M^e \bmod n, \text{ where } 0 \le M < n$$

- Decryption of a ciphertext C to recover the message M is:

To decrypt the ciphertext C the owner uses their private key PR={d,n} computes:

$$M = C^d \bmod n$$



Problem: In RSA algorithm system it is given that p=2, q= 11, e=7 and m=5. Find the cipher text "C" and decrypt "C" to set plain text M

Key management

One of the major roles of public-key encryption has been to address the problem of key distribution, with two distinct aspects:

- The distribution of public keys
- Use of public-key encryption to distribute secret keys.

1. Distribution of Public Keys

Several techniques have been proposed for the distribution of public keys, which can mostly be grouped into the categories shown below

- public announcement
- publicly available directory
- public-key authority
- public-key certificates

- **Public announcement**

The point of public-key encryption is that the public key is public; hence any participant can send his or her public key to any other participant, or broadcast the key to the community at large. Its major weakness is forgery, anyone can create a key claiming to be someone else and broadcast it, and until the forgery is discovered they can masquerade as the claimed user.

- **Publicly available directory**

A greater degree of security can be achieved by maintaining a publicly available dynamic directory of public keys. Maintenance and distribution of the public directory would have to be the responsibility of some trusted entity or organization. This scheme is clearly more secure than individual public announcements but still has vulnerabilities to tampering or forgery.

Directory must be trusted with properties:

- contains {name,public-key} entries
- participants register securely with directory
- participants can replace key at any time
- directory is periodically published
- directory can be accessed electronically

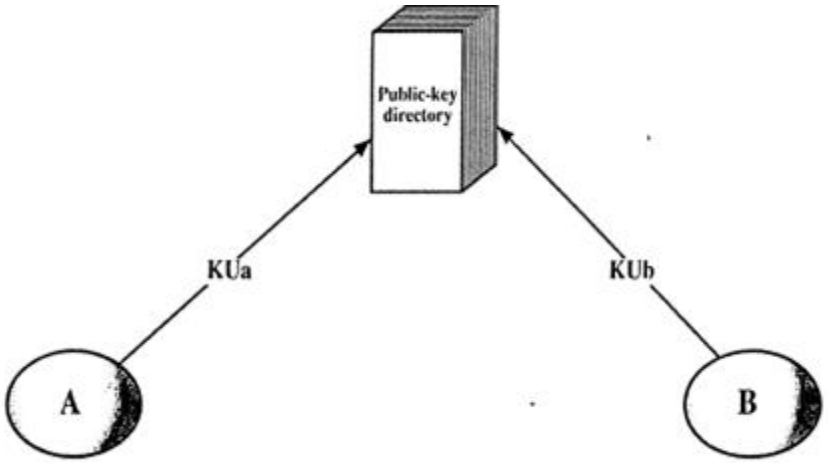
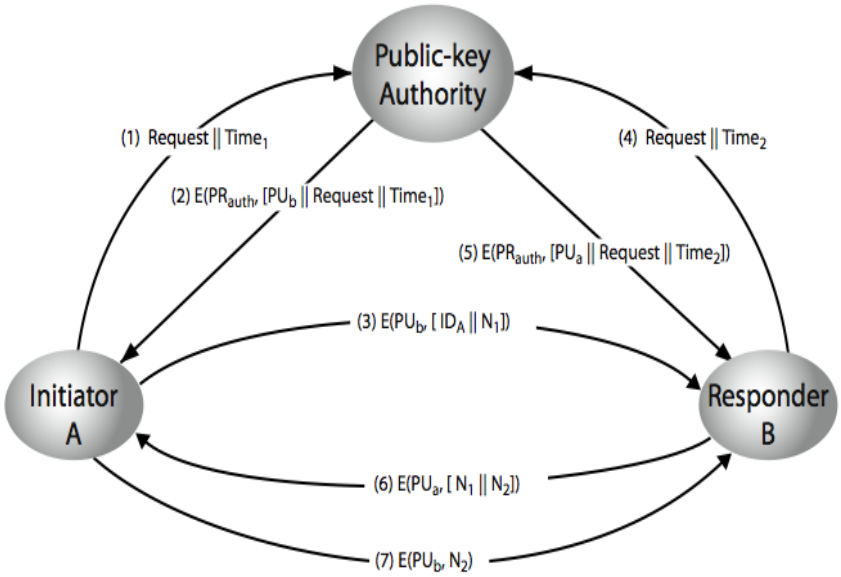


Figure Public-Key Publication.

- **Public-Key Authority**

Stronger security for public-key distribution can be achieved by providing tighter control over the distribution of public keys from the directory. It requires users to know the public key for the directory, and that they interact with directory in real-time to obtain any desired public key securely.



- **Public-key certificates**

An alternative approach first suggested by kohnfelder is to use certificates that can be used by participants to exchange keys without contacting a public –key authority. In a way that is as reliable as if the keys were obtained directly from a public key authority

Each certificate contains a public key and other information is created by a certificate authority and is given to the participant with the matching private key. A participant conveys its key information to another by transmitting its certificate .other participants can verify that the certificate was created by authority.

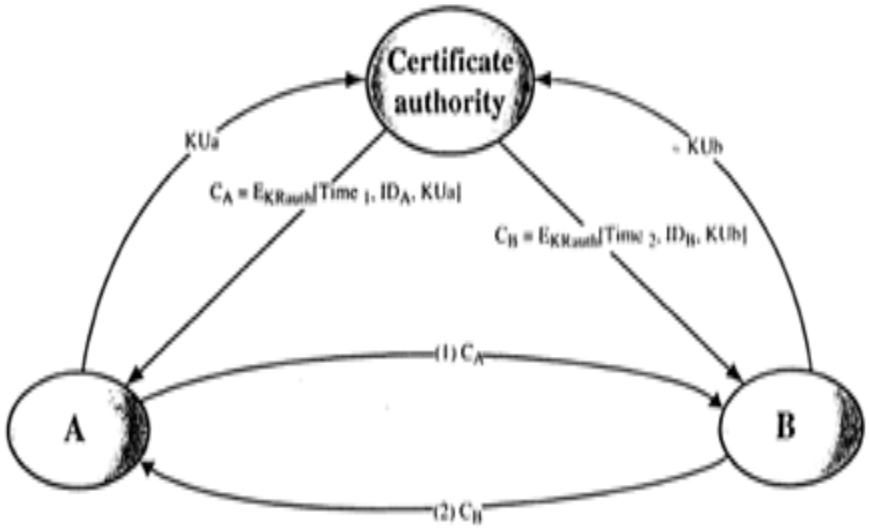


Figure Exchange of Public-Key Certificates.

2. Public-Key Distribution of Secret Keys

Once public keys have been distributed or have become accessible, secure communication that thwarts eavesdropping, tampering, or both, is possible. However, few users will wish to make exclusive use of public-key encryption for communication because of the relatively slow data rates that can be achieved. Accordingly, public-key encryption provides for the distribution of secret keys to be used for conventional encryption.

- **Simple Secret Key Distribution**
  An extremely simple scheme was put forward by Merkle. But it is insecure against an adversary who can intercept messages and then either relay the intercepted message or substitute another message. Such an attack is known as a man-in-the-middle attack

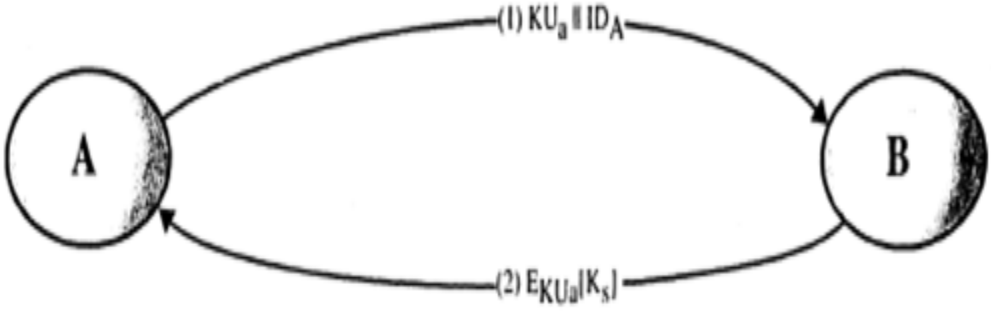


Figure Simple Use of Public-Key Encryption to Establish a Session Key.

- **Public-Key Distribution of Secret Keys**

- A uses B's public key to encrypt a message to B containing an identifier A (ID_A) of and a nonce N_1, which is used to identify this transaction uniquely.
- B sends a message to A encrypted with PU_A and containing A's nonce N_1 as well as a new nonce generated by B N_2. Because only B could have decrypted message (1), the presence of N_1 in message (2) assures A that the correspondent is B.
- A returns N_2, encrypted using B's public key, to assure B that its correspondent is A.
- A selects a secret key K_S and sends M = E (PU_b, E (PRa,Ks)) to B. Encryption of this message with B's public key ensures that only B can read it; encryption with A's private key ensures that only A could have sent it.
- B computes D (PUa, D (PRb, M)) to recover the secret key

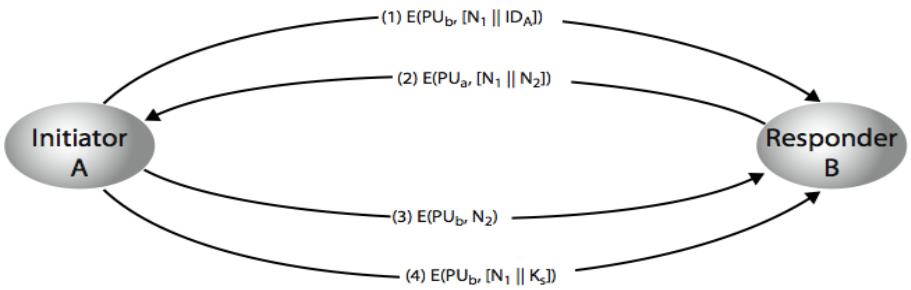


Figure. "Public-Key Distribution of Secret Keys

- **Hybrid Key Distribution**

  Yet another way to use public-key encryption to distribute secret keys is a hybrid approach .This scheme retains the use of a key distribution center (KDC) that shares a secret master key with each user and distributes secret session keys encrypted with the master key. A public key scheme is used to distribute the master keys. The addition of a public-key layer provides a secure, efficient means of distributing master keys. This is an advantage in a configuration in which a single KDC serves a widely distributed set of users.

Diffie-Hellman Key Exchange

The actual key exchange for either party consists of raising the others "public key' to power of their numbers, which means solving a discrete log, which is computationally infeasible given large enough numbers private key. The resulting number (or as much of as is necessary) is used as the key for a block cipher or other private key scheme. For an attacker to obtain the same value they need at least one of the secret.

Figure below summarizes the Diffie-Hellman key exchange algorithm. For this scheme,

- There are two publicly known numbers: a prime number q and an integer α that is a primitive root of q. Suppose the users A and B wish to exchange a key.
- User A selects a random integer $X_A < q$ and computes $Y_A = \alpha^{XA} \bmod q$.
- similarly, user B independently selects a random integer $X_B < q$ and computes $Y_B = \alpha^{XB} \bmod q$
- Each side keeps the value private and makes the value available publicly to the other side.
- User A computes the key as $K = (Y_B)^{XA} \bmod q$
- and user B computes the key as $K= (Y_A)^{XB} \bmod q$

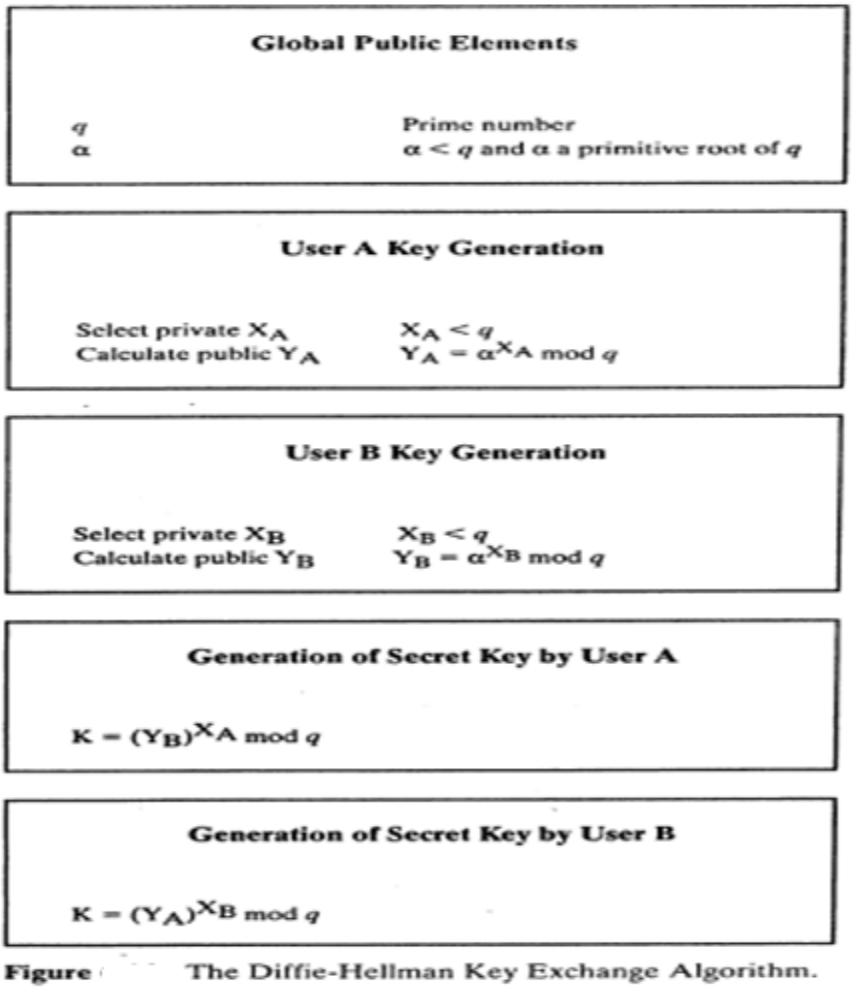


Figure The Diffie-Hellman Key Exchange Algorithm.

Key Exchange Protocols

- Users selects random private keys (X_A and X_B) each time they communicate
- Users create a known public key (Y_A and Y_B) and publish in a directory, then consulted and used to securely communicate with them
- Both of these are vulnerable to a meet-in-the-Middle Attack
- Authentication of the keys is needed
- By using public key and private key users secret key is generated

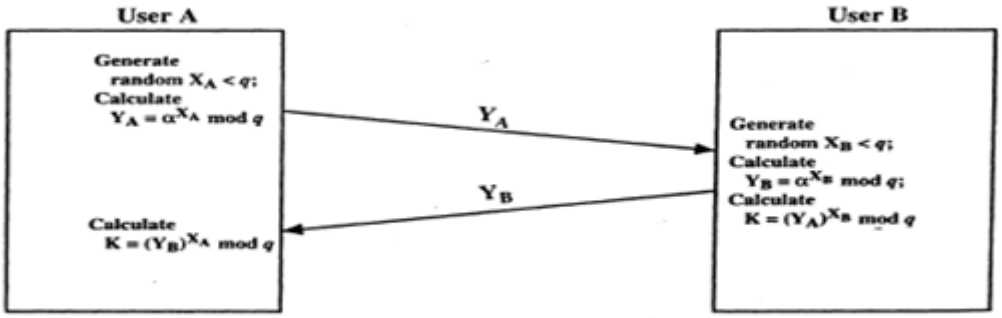


Figure Diffie-Hellman Key Exchange.

Elliptic Curve Cryptography

A major issue with the use of Public-Key Cryptography, is the size of numbers used, and hence keys being stored. Recently, an alternate approach has emerged, elliptic curve cryptography (ECC), which performs the computations using elliptic curve arithmetic instead of integer or polynomial arithmetic.

Elliptic Curve Cryptography uses addition as an analog of modulo multiply, and repeated addition as an analog of modulo exponentiation. But the "hard" problem is the elliptic curve logarithm problem.

Fig below illustrates the elliptic curve analog of Diffie-Hellman key exchange, which is a close analogy given elliptic curve multiplication equates to modulo exponentiation.

Global Public Elements	
$E_q(a, b)$	elliptic curve with parameters a, b, and q, where q is a prime or an integer of the form 2^m
G	point on elliptic curve whose order is large value n

User A Key Generation	
Select private n_A	$n_A < n$
Calculate public P_A	$P_A = n_A \times G$

User B Key Generation	
Select private n_B	$n_B < n$
Calculate public P_B	$P_B = n_B \times G$

Calculation of Secret Key by User A
$K = n_A \times P_B$

Calculation of Secret Key by User B
$K = n_B \times P_A$

ECC Diffie-Hellman Key Exchange