UNIT III PUBLIC KEY CRYPTOGRAPHY

MATHEMATICS OF ASYMMETRIC KEY CRYPTOGRAPHY: Primes – Primality Testing – Factorization – Euler's totient function, Fermat's and Euler's Theorem – Chinese Remainder Theorem – Exponentiation and logarithm

ASYMMETRIC KEY CIPHERS: RSA cryptosystem – Key distribution – Key management – Diffie Hellman key exchange -ElGamal cryptosystem – Elliptic curve arithmetic-Elliptic curve cryptography.

MATHEMATICS OF ASYMMETRIC KEY CRYPTOGRAPHY

PRIME NUMBER

An integer $p > 1$ is a prime number if and only if its only divisors are ±1 and ±p. Any integer $a > 1$ can be factored in a unique way as

$$a = p_1^{a_1} \times p_2^{a_2} \times \cdots \times p_t^{a_t}$$

where $p_1 < p_2 < \ldots < p_t$ are prime numbers and where each a_i is a positive integer.

Eg, $91 = 7 * 13$
$3600 = 2^4 * 3^2 * 5^2$
$11011 = 7 * 11^2 * 13$

If P is the set of all prime numbers, then any positive integer a can be written uniquely in the following form:

$$a = \prod_{p \in P} p^{a_p} \quad \text{where each } a_p \geq 0$$

It is easy to determine the greatest common divisor of two positive integers if we express each integer as the product of primes

Eg $300 = 2^2 * 3^1 * 5^2$
$18 = 2^1 * 3^2$
$\gcd(18, 300) = 2^1 * 3^1 * 5^0 = 6$

The following relationship always holds: If $k = \gcd(a, b)$, then $k_p = \min(a_p, b_p)$ for all p.

TESTING FOR PRIMALITY

For many cryptographic algorithms, it is necessary to select one or more very large prime numbers at random. Thus, we are faced with the task of determining whether a given large number is prime. There is no simple yet efficient means of accomplishing this task.

Miller-Rabin Algorithm

The algorithm due to Miller and Rabin [MILL75, RABI80] is typically used to test a large number for primality.

TEST (*n*)

1. Find integers k, q, with $k > 0$, q odd, so that $(n - 1 = 2^k q)$;
2. Select a random integer a, $1 < a < n - 1$;
3. if $a^q \bmod n = 1$ **then** return("inconclusive");
4. for $j = 0$ **to** $k - 1$ **do**
5. if $a^{2^j q} \bmod n = n - 1$ **then** return("inconclusive");
6. return("composite");

Example 1: Let us apply the test to the prime number $n = 29$.
$(n - 1) = 28 = 2^2(7) = 2^k q$.
First, let us try $a = 10$.
Compute $10^7 \bmod 29 = 17$,
$(10^7)^2 \bmod 29 = 28$, and the test returns inconclusive.
So n is prime number.

FERMAT AND EULER'S THEOREM

Two theorems that play important roles in public-key cryptography are Fermat's theorem and Euler's theorem

Fermat's Theorem (also called as Fermat's little Theorem)

Definition:

If P is prime and a is a positive integer not divisible by p, then $\mathbf{a^{p-1} \equiv 1 \pmod p}$

Fermat's little theorem is the basis for the Fermat primality test and is one of the fundamental results of elementary number theory.

This theorem is useful in generating public key in RSA and Primality testing

Proof:

1) Consider the set of positive integers less than *p:*{1,2,3..p-1}
2) Multiply each element by *a, modulo p* to get the set

   X ={ a mod p,2a mod p….(p-1) a mod p}.

   - None of the elements of X is equal to zero because *p* does not divide *a*.
   - No two of the integers in X are equal.
   - We know that (p-1) elements of X are all positive integers with no two elements are equal. So, we can conclude X consists of the set of integers {1,2,……..,p-1} in some order

3) Multiplying the numbers in both sets (p and X) and taking the result mod *p* yields.

$$a * 2a * \ldots * (p-1)a \equiv [(1*2*\ldots*(p-1)] \pmod p$$
$$\{1 * 2 * \ldots * (p-1)\}\ a^{p-1} \equiv [(1*2*\ldots*(p-1)] \pmod p$$
$$(p-1)!\ a^{p-1} \equiv (p-1)! \pmod p$$
$$\mathbf{a^{p-1} \equiv 1 \pmod p}$$

Example

$a = 7, p = 19$ $\quad a^{p-1} \equiv 1 \pmod p.$

- $7^2 = 49 \equiv 11 \pmod{19}$
- $7^4 = 7^2 \times 7^2 = 121 \equiv 7 \pmod{19}$
- $7^8 = 7^4 \times 7^4 = 7 \times 7 = 49 \equiv 11 \pmod{19}$
- $7^{16} \equiv 7^8 \times 7^8 = 11 \times 11 = 121 \equiv 7 \pmod{19}$
- $a^{p-1} = 7^{18} = 7^{16} * 7^2 \equiv 7 * 11 \equiv 1 \pmod{19}$

An alternative form of Fermat's theorem:

If p is prime and a is a positive integer, $\mathbf{a^p \equiv a \pmod p}$

Eg: a=3, p=5
$a^p = 3^5 = 243 \equiv 3 \pmod 5 = a \pmod p$
$a^p \equiv a \pmod p$

Euler's Totient function

Euler's totient function written as $\emptyset(n)$, (called as phi) is defined as the number of positive integers less than n and relatively prime (Co-Prime) to ***n***.

The Properties are as follows

1) ø(1) = 1
2) ø(p)=p-1 for p (p prime)
3) ø(p.q)=(p-1)x(q-1) for p.q (p,q prime)

Suppose that we have two prime numbers *p* and *q*, with *p* not equal to *q*. Then we can show that

n=pq.

ø(n)= ø(pq)= ø(p)* ø(q)=(p-1)*(q-1)

Examples:

1) ø(37) = 36 {ø(p)=p-1 for p (p prime)]
2) ø(21) = ø(3)* ø(7)= (3–1)x(7–1) = 2x6 = 12 where the 12 integers are {1,2,4,5,8,10,11,13,16,17, 19, 20}

[ø(p.q)=(p-1)x(q-1) for p.q (p,q prime)]

Sample Examples

1. What is the value of Φ(13)?
   Because 13 is a prime, Φ (13) = (13 −1) = 12.
2. What is the value of Φ (10)?
   We can use the third rule: Φ (10) = Φ (2) × Φ (5) = 1 × 4 = 4, because 2 and 5 are primes.
3. What is the number of elements in Z14*?
   The answer is Φ (14) = Φ (7) × Φ (2) = 6 × 1 = 6. The members are 1, 3, 5, 9, 11, and 13.

Euler's theorem

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

$$\mathbf{a^{ø(n)} \equiv 1 \pmod n}$$

Proof:

The above equation is true, if n is prime, because in that case *ø(n)=(n-1)* and Fermat's theorem holds. However it holds for any integer *n*.

1)Recall that ø(n)is the number of positive integers less than n that are relatively prime to *n*. consider the set of such integers, labeled as follows:

$$R=\{x_1, x_2 \ldots . x_{ø(n)}\}$$

That is, each element x_i of R is a unique positive integer less than *n* with $\gcd(x_i, n)=1$.

2)Now multiply each element by *a modulo n:*

$$S=\{(ax_1 \bmod n), (ax_2 \bmod n), \ldots . (ax_{ø(n)}) \bmod n)\}$$

3) The set S is a permutation of R, by the following reasons:
1.Because *a* is relatively prime to *n* and x_i is relatively prime to *n*,ax_i must also be relatively prime to *n*. thus all the members of S are integers that are less than n and that are relatively prime to n.
2. There are no duplicates in S. if $ax_i \bmod n = ax_i \bmod n$, then $x_i = x_j$

$$\prod_{i=1}^{\phi(n)} (ax_i \bmod n) = \prod_{i=1}^{\phi(n)} x_i$$

$$\prod_{i=1}^{\phi(n)} ax_i \equiv \prod_{i=1}^{\phi(n)} x_i \pmod{n}$$

$$a^{\phi(n)} \times \left[\prod_{i=1}^{\phi(n)} x_i\right] \equiv \prod_{i=1}^{\phi(n)} x_i \pmod{n}$$

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

An alternative form of the theorem is also useful:

$$a^{\phi(n)+1} \equiv a \pmod{n}$$

$$a = 3; n = 10; \phi(10) = 4\ a^{\phi(n)} = 3^4 = 81 = 1 \pmod{10} = 1 \pmod{n}$$
$$a = 2; n = 11; \phi(11) = 10\ a^{\phi(n)} = 2^{10} = 1024 = 1 \pmod{11} = 1 \pmod{n}$$

THE CHINESE REMAINDER THEOREM

The Chinese remainder theorem (CRT) is used to solve a set of congruent equations with one variable but different moduli, which are relatively prime, as shown below:

Let $m_1, m_2 \ldots\ldots m_k$ be integers with $\gcd(m_i, m_j) = 1$, whenever $i \neq j$. Let $a_1, a_2 \ldots\ldots a_k$ be integers, there exists exactly one solution x (mod $m_1, m_2 \ldots\ldots m_k$) to the simultaneous congruences

$$x \equiv a_1 \pmod{m_1}$$
$$x \equiv a_2 \pmod{m_2}$$
$$x \equiv a_k \pmod{m_k}$$

If $n_1,n_2,..,n_k$ are positive integers that are pairwise co-prime and $a_1,a_2,\ldots,a_k$ are any integers, then CRT is used to find the values of x that solves the following congruence simultaneously.

Value of $x=(a_1m_1y_1+a_2m_2y_2+\ldots+a_km_ky_k) \bmod M$
Where $M=n_1n_2n_3..n_k$

$m_i=M/n_i$

$m_iy_i=1 \bmod n_i$

Example 1:
Find the solution to the simultaneous equations: $x \equiv 1 \pmod 5$, $x \equiv 2 \pmod 6$, $x \equiv 3 \pmod 7$.
Solution

$M=n_1n_2n_3$

$M=5*6*7=210$

$m_i = M/n_i$

m1=210/5=42

m2=210/6=35

m3=210/7=30

$m_i y_i = 1 \bmod n_i$

$42y_1 = 1 \bmod 5$

$y_1 = 2$

$35y_2 = 1 \bmod 6$

$y_2 = 5$

$30y_3 = 1 \bmod 7$

$y_3 = 2$

$x = (a_1 m_1 y_1 + a_2 m_2 y_2 + a_3 m_3 y_3) \bmod M$

$= ((1*42*2)+(2*35*5)+(3*30*3)) \bmod 210$

$= 193$

Example 2:

Find an integer that has a remainder of 3 when divided by 7 and 13, but is divisible by 12.
Solution: This is a CRT problem. We can form three equations and solve them to find the value of x.

$$x = 3 \bmod 7$$
$$x = 3 \bmod 13$$
$$x = 0 \bmod 12$$

If we follow the four steps, we find x = 276. We can check that 276 = 3 mod 7, 276 = 3 mod 13 and 276 is divisible by 12 (the quotient is 23 and the remainder is zero).

Example 3:

A bag has contained number of pens if you take out 3 pens at a time 2 pens are left. If you take out 4 pens at a time 1 pen is left and if you take out 5 pens at a time 3 pens are left in the bag. What is the number of pens in the bag.

$x \equiv 2 \bmod 3$

$x \equiv 1 \bmod 4$

$x \equiv 3 \bmod 5$

$a_1 = 2$

$a_2 = 1$

$a_3 = 3$

$n_1 = 3$

$n_2 = 4$

$n_3 = 5$

$M = n_1 n_2 n_3$

$M = 3*4*5 = 60$

$m_i=M/n_i$
m1=60/3=20
m2=60/4=15
m3=60/5=12

$m_iy_i=1 \bmod n_i$
$20y_1=1 \bmod 3$
$Y_1=2 \bmod 3$

$15y_2= 1 \bmod 4$
$y_2=3 \bmod 4$

$12y_3=1 \bmod 5$
$y_3=3 \bmod 5$

$x=(a_1m_1y_1+a_2m_2y_2+ a_3m_3y_3) \bmod M$
=((2*20*2)+(1*15*3)+(3*12*3)) mod 60
=233 mod 60
=53

DISCRETE LOGARITHMS.

Discrete logarithms are fundamental to a number of public-key algorithms. Discrete logarithms are analogous to ordinary logarithms but are defined using modular arithmetic.

Discrete logarithms are fundamental to a number of public-key algorithms, including Diffie-Hellman key exchange and the digital signature algorithm (DSA)

The Powers of an Integer, Modulo *n*

Recall from Euler's theorem that, for every and that are relatively prime,

$$a^{\phi(n)} = 1 \pmod{n}$$

Where $\phi(n)$, Euler's totient function, is the number of positive integers less than and relatively prime to . Now consider the more general expression:

$$a^m = 1 \pmod{n}$$

If a and n are relatively prime, then there is at least one integer m that satisfies Equation, namely M= $\phi(n)$, . The least positive exponent m for which Equation holds is referred to in several ways:

- The order of a(mod n)
- The exponent to which a belongs (mod n)
- The length of the period generated by a

The highest possible exponent to which a number can belong (mod n) is $\phi(n)$. If a number is of this order, it is referred to as a **primitive root** of n .The importance of this notion is that if is a primitive root of , then its powers

$$a, a^2, \ldots, a^{\phi(n)}$$

Logarithms for Modular Arithmetic

A primitive root of a prime number p is one whose powers modulo p generate all the integers from 1 to p - 1. That is, if 'a' is a primitive root of the prime number p, then the numbers

a mod p, a^2 mod p,…, a^{p-1} mod p

are distinct and consist of the integers from 1 through p - 1 in some permutation.

For any integer b and a primitive root a of prime number p, we can find a unique exponent i such that

$b \equiv a^i \pmod p$ where 0 … i … (p - 1)

The exponent i is referred to as the discrete logarithm of b for the base a, mod p.

We denote this value as dloga,p(b)

Note the following:

dlog$_{a,p}(a)$ = 1 because $a^1 \bmod p = a$

dlog$_{a,p}(1)$ = 0 because $a^0 \bmod p = 1 \bmod p = 1$

Calculation of Discrete Logarithms

Consider the equation

$$y = gx \bmod p$$

Given g, x , and p , it is a straightforward matter to calculate y. At the worst, we must perform repeated multiplications, and algorithms exist for achieving greater efficiency.

PUBLIC KEY CRYPTOGRAPHY

Introduction to Public key Cryptography:

- Public key cryptography also called as **asymmetric cryptography**.
- It was invented by whitfield **Diffie** and Martin **Hellman** in 1976. Sometimes this cryptography also called as **Diffie-Helman Encryption**.
- Public key algorithms are based on mathematical problems which admit no efficient solution that are inherent in certain integer factorization, discrete logarithm and Elliptic curve relations.

Public key Cryptosystem Principles:

- The concept of public key cryptography in invented for two most difficult problems of Symmetric key encryption.
  - *The Key Exchange Problem*
  - *The Trust Problem*

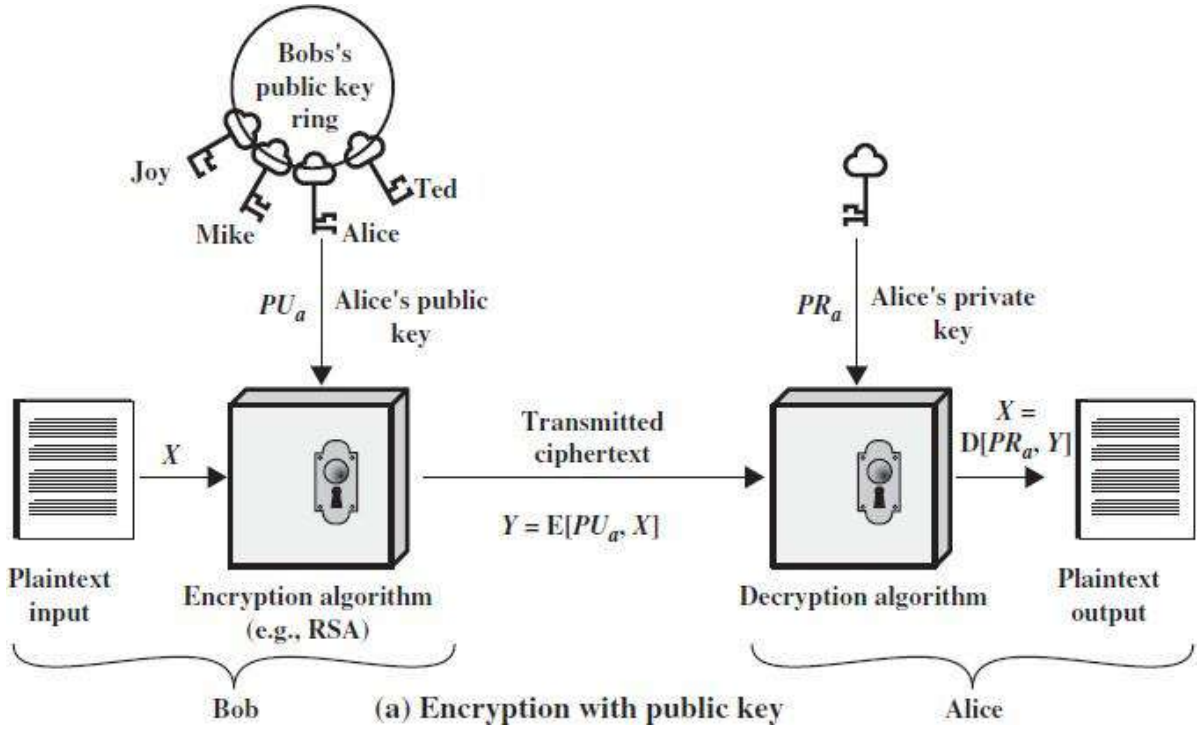
  The Key Exchange Problem: *The key exchange problem arises from the fact that communicating parties must somehow share a secret key before any secure communication can be initiated, and both parties must then ensure that the key remains secret. Of course, direct key exchange is not always feasible due to risk, inconvenience, and cost factors.*

  The Trust Problem: *Ensuring the integrity of received data and verifying the identity of the source of that data can be very important. Means in the symmetric key cryptography system, receiver doesn't know whether the message is coming for particular sender.*

- This public key cryptosystem uses two keys as pair for encryption of plain text and Decryption of cipher text.
- These two keys are names as "**Public key**" and "**Private key**". The private key is kept secret whereas public key is distributed widely.
- A message or text data which is encrypted with the public key can be decrypted only with the corresponding private-key
- This two key system very useful in the areas of confidentiality (secure) and authentication

A **public-key encryption** scheme has six ingredients		
1	**Plaintext**	This is the readable message or data that is fed into the algorithm as input.
2	**Encryption algorithm**	The encryption algorithm performs various transformations on the plaintext.
3	**Public key**	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
4	**Private Key**	
5	**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.
6	**Decryption algorithm**	This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

Public key cryptography for providing confidentiality (secrecy)



The essential steps are the following.

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.
2. Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private. So above fig states that each user maintains a collection of public keys obtained from others.
3. If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.
4. When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key.

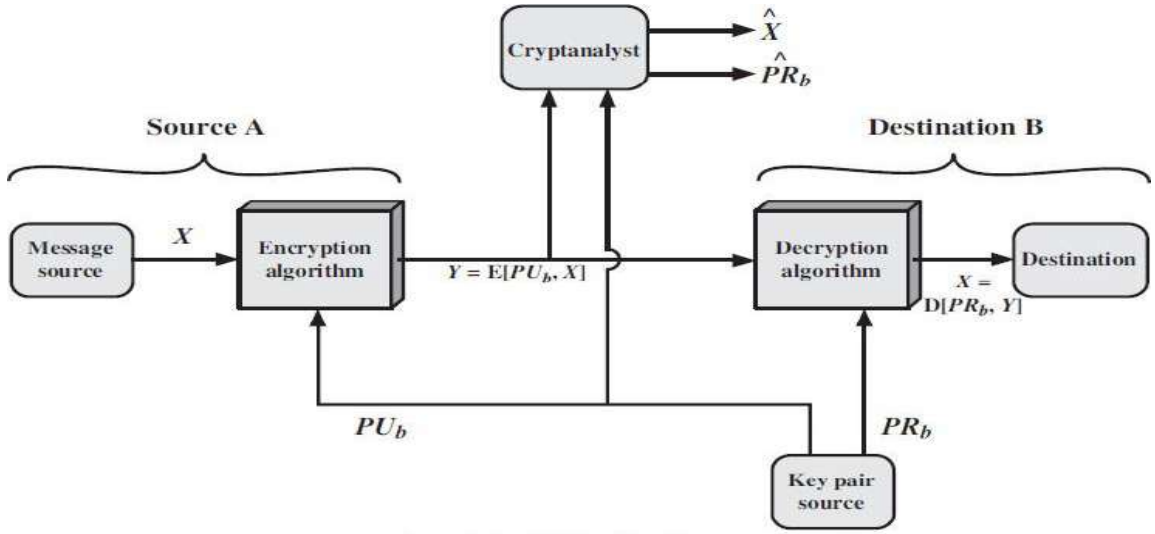


Figure 9.2 Public-Key Cryptosystem: Secrecy

There is some source A that produces a message in plaintext $\boldsymbol{X = [X1, X2, \ldots, XM]}$.

The $\boldsymbol{M}$ elements of $\boldsymbol{X}$ are letters in some finite alphabet. The message is intended for destination **B**. B generates a related pair of keys: a public key, $\boldsymbol{PU_b}$, and a private key, $\boldsymbol{PR_b}$.

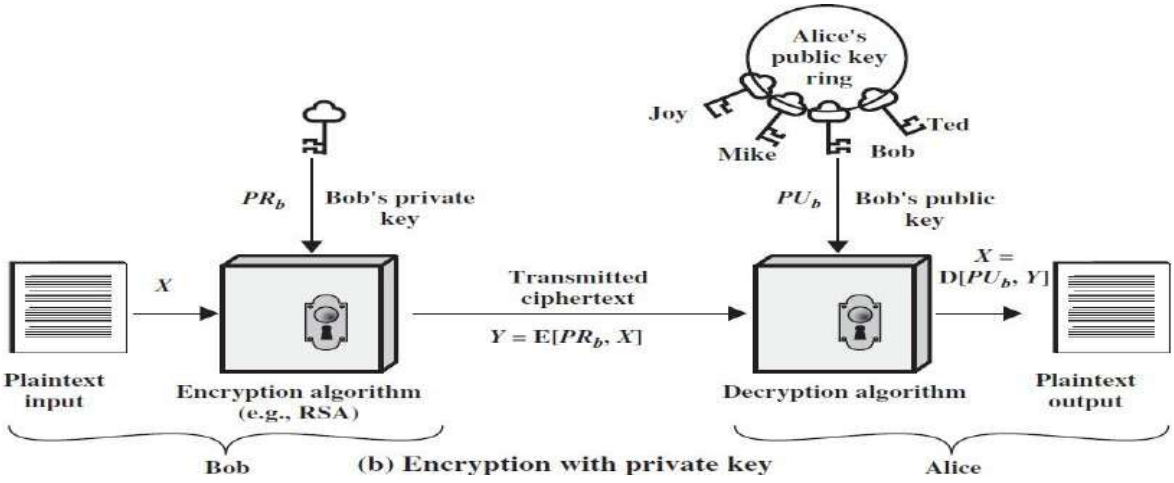
$\boldsymbol{PR_b}$ is known only to B, whereas $\boldsymbol{PU_b}$ is publicly available and therefore accessible by A. With the message X and the encryption key $\boldsymbol{PU_b}$ as input, A forms the ciphertext $Y = [Y1, Y2, \ldots, Y_N]$:

$$Y = \mathrm{E}(PU_b, X)$$

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

$$X = \mathrm{D}(PR_b, Y)$$

Public key cryptography for proving Authentication:



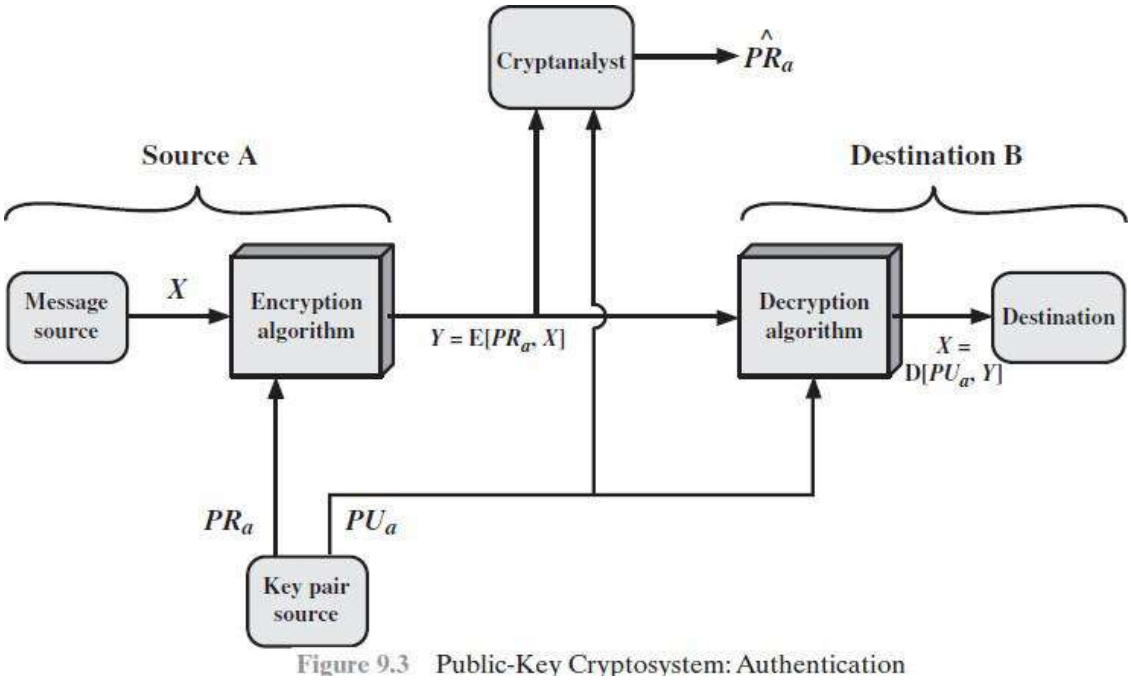


Figure 9.3 Public-Key Cryptosystem: Authentication

The above diagrams show the use of public-key encryption to provide authentication:

$$Y = \mathrm{E}(PR_a, X)$$

$$X = \mathrm{D}(PU_a, Y)$$

- In this case,A prepares a message to B and encrypts it using A's private key before transmitting it. B can decrypt the message using A's public key. Because the message was encrypted using A's private key, only A could have prepared the message. Therefore, the entire encrypted message serves as a **digital signature.**
- It is impossible to alter the message without access to A's private key, so the message is authenticated both in terms of source and in terms of data integrity.

Public key cryptography for both authentication and confidentiality (Secrecy)

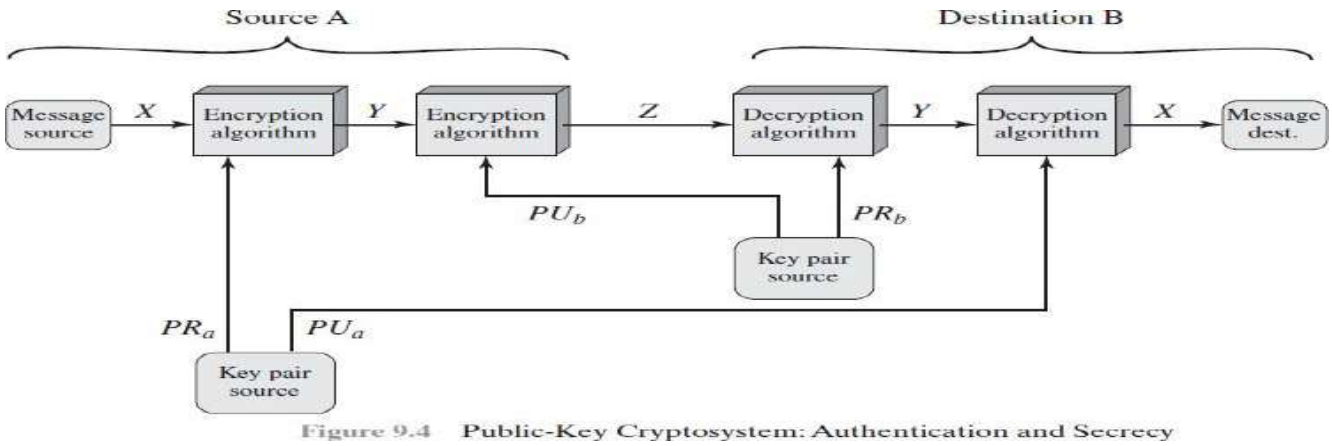


Figure 9.4 Public-Key Cryptosystem: Authentication and Secrecy

It is, however, possible to provide both the authentication function and confidentiality by a double use of the public-key scheme (above figure):

$$Z = \mathrm{E}(PU_b, \mathrm{E}(PR_a, X))$$

$$X = \mathrm{D}(PU_a, \mathrm{D}(PR_b, Z))$$

In this case, we begin as before by encrypting a message, using the sender's private key. This provides the digital signature. Next, we encrypt again, using the receiver's public key. The final ciphertext can be decrypted only by the intended receiver, who alone has the matching private key. Thus, confidentiality is provided.

Applications for Public-Key Cryptosystems

The use of **public-key cryptosystems** into three categories

• ***Encryption /decryption***: The sender encrypts a message with the recipient's public key.

• ***Digital signature***: The sender "signs" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message.

• ***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.

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

RSA

- It is the most common public key algorithm.
- This RSA name is get from its inventors first letter (Rivest (R), Shamir (S) and Adleman (A)) in the year 1977.
- The RSA scheme is a block cipher in which the plaintext & ciphertext are integers between 0 and n-1 for some 'n'.
- A typical size for 'n' is 1024 bits or 309 decimal digits. That is, n is less than 2^{1024}

Description of the Algorithm:

- RSA algorithm uses an expression with exponentials.

- In RSA plaintext is encrypted in blocks, with each block having a binary value less than some number n. that is, the block size must be less than or equal to **$\log_2(n)$**
- **RSA** uses two exponents 'e' and 'd' where e is public and d is private.
- Encryption and decryption are of following form, for some PlainText 'M' and CipherText block 'C'

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

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

$$M = C^d \bmod = (M^e \bmod n)^{d} \text{ mon } n = (M^e)^d \bmod n = M^{ed} \bmod n$$

- Both sender and receiver must know the value of n.
- The sender knows the value of 'e' & only the reviver knows the value of 'd' thus this is a public key encryption algorithm with a

  Public key PU={e, n}

  Private key PR={d, n}

Requirements:

The RSA algorithm to be satisfactory for public key encryption, the following requirements must be met:

1. It is possible to find values of e, d n such that " $\mathbf{M^{ed} \bmod n = M}$ " for all M<n
2. It is relatively easy to calculate " $\mathbf{M^e \bmod n}$ " and " $\mathbf{C^d \bmod n}$ "for M<n
3. It is infeasible to determine "d" given 'e' & 'n'. The " $\mathbf{M^{ed} \bmod n = M}$ " relationship holds if 'e' & 'd' are multiplicative inverses modulo Ø(n).

   Ø(n) is Euler Totient function

   For p,q primes where p*q and p≠q.

   Ø(n)= Ø(pq)=(p-1)(q-1)

Then the relation between 'e' & 'd' can be expressed as " $\boxed{ed \bmod \emptyset(n)=1}$ "
this is equivalent to saying

$$ed \equiv 1 \bmod \emptyset(n)$$

$$d \equiv e^{-1} \bmod \emptyset(n)$$

That is 'e' and 'd' are multiplicative inverses mod Ø(n).

Note: according to the rules of modular arithmetic, this is true only if 'd' (and 'e') is relatively prime to Ø(n).

Equivalently **gcd(Ø(n), d)=1.**

Steps of RSA algorithm:

Step 1 Select 2 prime numbers p & q

Step 2 Calculate n=pq

Step 3Calculate Ø(n)=(p-1)(q-1)

Step 4 Select or find integer e (public key) which is relatively prime to Ø(n) i.e., e with gcd (Ø(n), e)=1 where 1<e< Ø(n).

Step 5 Calculate "d" (private key) by using following condition. $ed \equiv 1 \bmod \emptyset(n)$

d< Ø(n).

Step 6 Perform encryption by using $C = M^e \bmod n$

Step 7 perform Decryption by using $M = C^d \bmod n$

Example:

1. Select two prime numbers, $p = 17$ and $q = 11$.

2. Calculate $n = pq = 17 \times 11 = 187$.

3. Calculate $\emptyset(n) = (p - 1)(q - 1) = 16 \times 10 = 160$.

4. Select e such that e is relatively prime to $\emptyset(n) = 160$ and less than $\emptyset(n)$; we choose $e = 7$.

5. Determine d such that $de \equiv 1 \pmod{160}$ and $d < 160$. The correct value is $d = 23$, because $23 * 7 = 161 = (1 \times 160) + 1$; d can be calculated using the extended Euclid's algorithm

The resulting keys are public key $PU = \{7, 187\}$ and private key $PR = \{23, 187\}$.

The example shows the use of these keys for a plaintext input of $M = 88$. For encryption, we need to calculate $C = 88^7 \bmod 187$. Exploiting the properties of modular arithmetic, we can do this as follows.

$$88^7 \bmod 187 = [(88^4 \bmod 187) \times (88^2 \bmod 187) \times (88^1 \bmod 187)] \bmod 187$$

$$88^1 \bmod 187 = 88$$

$$88^2 \bmod 187 = 7744 \bmod 187 = 77$$

$$88^4 \bmod 187 = 59{,}969{,}536 \bmod 187 = 132$$

$$88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187 = 894{,}432 \bmod 187 = 11$$

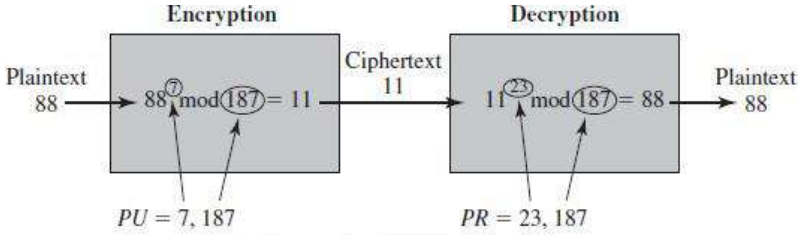


Figure 9.6 Example of RSA Algorithm

For decryption, we calculate $M = 11^{23} \bmod 187$:

$$11^{23} \bmod 187 = [(11^1 \bmod 187) \times (11^2 \bmod 187) \times (11^4 \bmod 187) \times (11^8 \bmod 187) \times (11^8 \bmod 187)] \bmod 187$$

$$11^1 \bmod 187 = 11$$

$$11^2 \bmod 187 = 121$$

$$11^4 \bmod 187 = 14{,}641 \bmod 187 = 55$$

$$11^8 \bmod 187 = 214{,}358{,}881 \bmod 187 = 33$$

$$11^{23} \bmod 187 = (11 \times 121 \times 55 \times 33 \times 33) \bmod 187 = 79{,}720{,}245 \bmod 187 = 88$$

RSA Attacks

There are four possible approaches to attack the RSA:

- **Brute force:** This involves trying all possible private keys. The defence against this attack is the use of large key space.
- **Mathematical attacks**: There are several approaches, all equivalent in effort to factoring the product of two primes. Three approaches that could be identified of this type are:
    - Factor n into its two prime factors which enables calculation of $\phi(n) = (p - 1) \times (q - 1)$, which in turn enables determination of $d \equiv e^{-1} \pmod{\phi(n)}$.
    - Determine $\phi(n)$ directly, without first determining p and q. Again, this enables determination of $d \equiv e^{-1} \pmod{\phi(n)}$.
    - Determine d directly, without first determining $\phi(n)$.
- **Timing attacks**: These depend on the running time of the decryption algorithm. This attack is alarming for two reasons namely it comes from a completely unexpected direction, and it is a cipher text-only attack. Modular exponentiation algorithm that is accomplished bit by bit, with one modular multiplication performed every iteration and an additional modular multiplication performed for each 1 bit can be used to perform this attack. Simple counter measures could be used to overcome the timing attack. They are
    - ***Constant exponentiation time***: Ensure that all exponentiations take the same amount of time before returning a result. This is a simple fix but does degrade performance.
    - ***Random delay***: Better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack. Kocher points out that if defenders don't add enough noise, attackers could still succeed by collecting additional measurements to compensate for the random delays.
    - ***Blinding***: Multiply the cipher text by a random number before performing exponentiation. This process prevents the attacker from knowing what cipher text bits are being processed inside the computer and therefore prevents the bit-by-bit analysis essential to the timing attack.
- **Chosen cipher text attacks (CCAs)**: This type of attack exploits properties of the RSA algorithm. It is defined as an attack in which the adversary chooses a number of ciphertexts and is then given the corresponding plaintexts, decrypted with the target's

private key. Thus, the adversary could select a plaintext, encrypt it with the target's public key, and then be able to get the plaintext back by having it decrypted with the private key. Clearly, this provides the adversary with no new information. Instead, the adversary exploits properties of RSA and selects blocks of data that, when processed using the target's private key, yield information needed for cryptanalysis.

To overcome this simple attack, practical RSA-based cryptosystems randomly pad the plaintext prior to encryption. More sophisticated CCAs are possible and simple padding with a random value is insufficient to provide the desired security. To counter such attacks modifying the plaintext using a procedure known as optimal asymmetric encryption padding will help.

Diffie-Hellman key exchange is the first published public key algorithm, also known as exponential key agreement. And it is based on mathematical principles. The purpose of the algorithm is to enable two users to exchange a key securely that can then be used for subsequent encryption of messages.This algorithm itself is limited to exchange of the keys. Security of algorithm depends on computing discrete logarithms values.

KEY MANAGEMENT

There are actually two distinct aspects to the use of public-key cryptography:

- The distribution of public keys
- The use of public-key encryption to distribute secret keys

1. Distribution of Public Keys

There are four different schemes

- Public announcement
- Publicly available directory
- Public-key authority
- Public-key certificates

A.Public announcement

Any participant can send his or her public key to any other participant or **broadcast** the key to the community. Uncontrolled Public-Key Distribution

Limitation : Anyone can forge such a public announcement. That is, some user could pretend to be user A and send a public key to another participant or broadcast such a public key. Authentication is needed to avoid this problem.

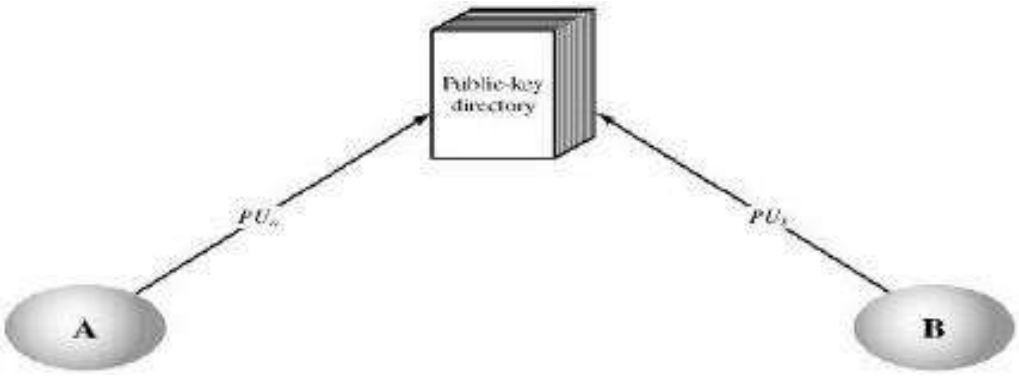
B. Publicly Available Directory

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.

i)The authority maintains a directory with a {name, public key} entry for each participant.

ii)Each participant registers a public key with the directory authority.

iii) Participants could also access the directory electronically.



Limitation :

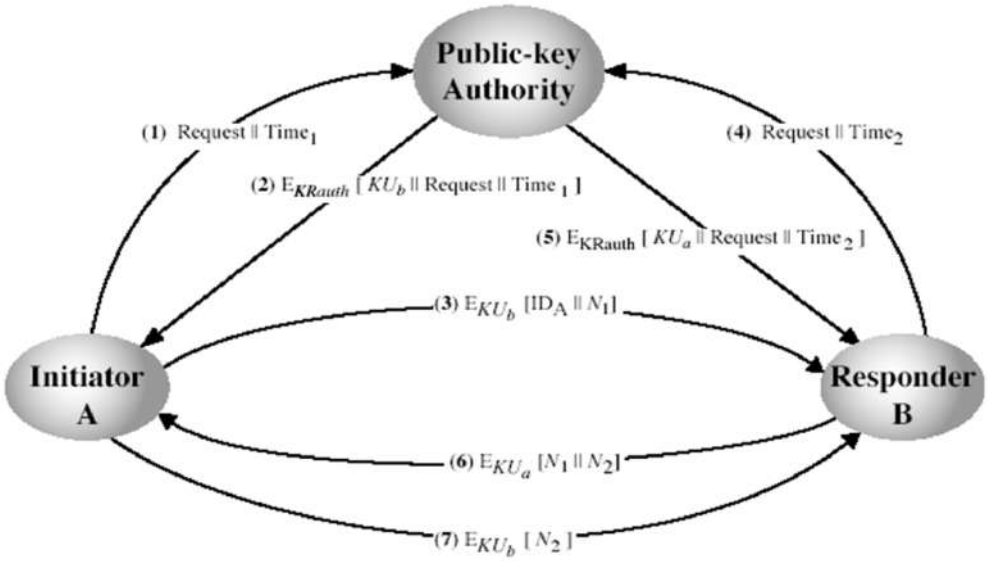
An Adversary may impersonate by stealing the private key of public key directory and falsely send the public key details.

An attacker may attack the records stored in the directory.

C. 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.

Each participant reliably knows a public key for the authority, with only the authority knowing the corresponding private key.



i)A sends a time stamped message to the public-key authority containing a request for the current public key of B.

ii) The authority responds with a message that is encrypted using the authority's private key, PRauth. Thus, A is able to decrypt the message using the authority's public key. Therefore, A is assured

that the message originated with the authority. The message includes the following:

- B's public key, PUb which A can use to encrypt messages destined for B
- The original request, to enable A to match this response with the corresponding earlier request and to verify that the original request was not altered before reception by the authority
- The original timestamp, so A can determine that this is not an old message from the authority containing a key other than B's current public key

iii) A stores B's public key and also uses it to encrypt a message to B containing an identifier of A (IDA) and a nonce (N1), which is used to identify this transaction uniquely.

iv) B retrieves A's public key from the authority in the same manner as A retrieved B's public key.

v) At this point, public keys have been securely delivered to A and B, and they may begin their protected exchange.

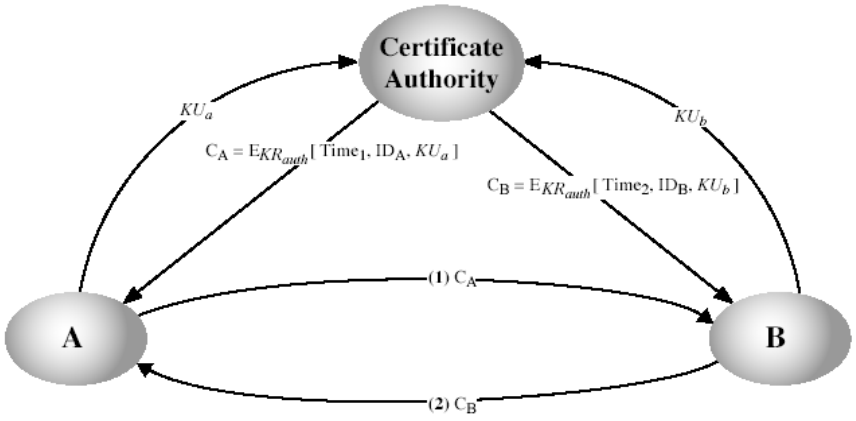
v) B sends a message to A encrypted with PUa and containing A's nonce (N1) as well as a new nonce generated by B (N2) Because only B could have decrypted message (3), the presence of N1 in message (6) assures A that the correspondent is B.

vi) A returns N2, encrypted using B's public key, to assure B that its correspondent is A.

Limitations : Bottleneck may occur in public authority. Tampering of records stored by the authority may take place.

D. Public key certificate

A certificate consists of a public key plus an identifier of the key owner, with the whole block signed by a trusted third party.



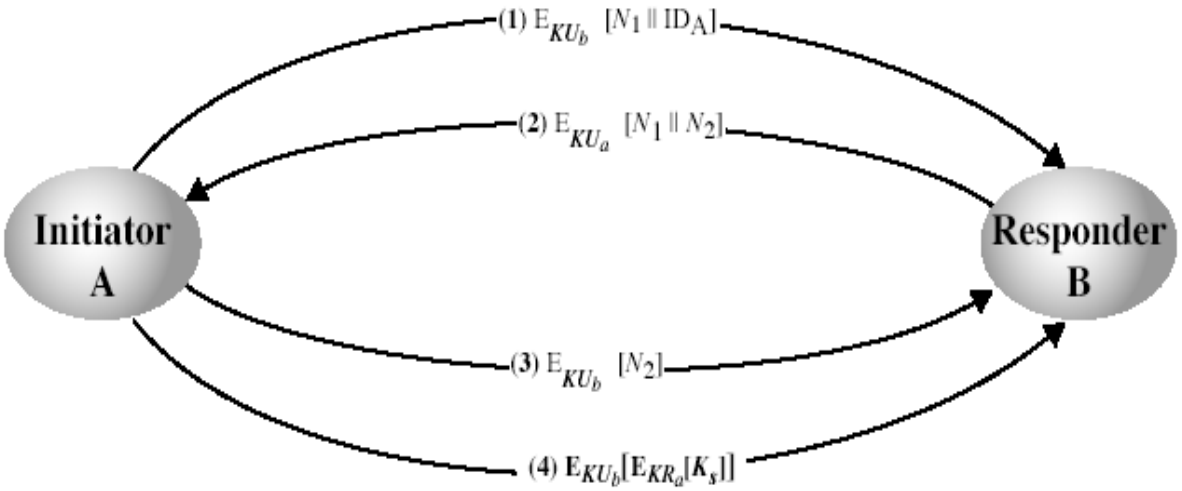
Typically, the third party is a certificate authority, such as a government agency or a financial institution, that is trusted by the user community.

A user can present his or her public key to the authority in a secure manner, and obtain a certificate. The user can then publish the certificate.

2. Secret Key Distribution with Confidentiality and Authentication

i) A uses B's public key to encrypt a message to B containing an identifier of A (IDA) and a nonce (N1), which is used to identify this transaction uniquely.

ii) B sends a message to A encrypted with PUa and containing A's nonce (N1) as well as a new nonce generated by B (N2)



iv) A returns *N*2 encrypted using B's public key, to assure B that its correspondent is A.
A selects a secret key *Ks* and sends *M* = E(*PUb*, 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.

v) B computes D(*PUa*, D(*PRb*, *M*)) to recover the secret key.

Elagamal Cryptographic system

- Public-key cryptosystem related to D-H
- uses exponentiation in a finite field
- with security based difficulty of computing discrete logarithms, as in D-H
- Used in number of standards including DSS (Digital Signature Standard) and S/MIME e-mail standard

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

Key Generation by Alice	
Select private X_A	$X_A < q - 1$
Calculate Y_A	$Y_A = \alpha^{X_A} \bmod q$
Public key	$\{q, \alpha, Y_A\}$
Private key	X_A

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < q$
Select random integer k	$k < q$
Calculate K	$K = (Y_A)^k \bmod q$
Calculate C_1	$C_1 = \alpha^k \bmod q$
Calculate C_2	$C_2 = KM \bmod q$
Ciphertext:	(C_1, C_2)

Decryption by Alice with Alice's Private Key	
Ciphertext:	(C_1, C_2)
Calculate K	$K = (C_1)^{X_A} \bmod q$
Plaintext:	$M = (C_2 K^{-1}) \bmod q$

How K is recovered by the decryption process:

$K = (Y_A)^k \bmod q$ — K is defined during the encryption process

$K = (\alpha^{X_A} \bmod q)^k \bmod q$ — substitute using $Y_A = \alpha^{X_A} \bmod q$

$K = \alpha^{kX_A} \bmod q$ — by the rules of modular arithmetic

$K = (C_1)^{X_A} \bmod q$ — substitute using $C_1 = \alpha^k \bmod q$

Next, using K, we recover the plaintext as

$$C_2 = KM \bmod q$$

$$(C_2 K^{-1}) \bmod q = KMK^{-1} \bmod q = M \bmod q = M$$

Example:

message.

For example, let us start with the prime field GF(19); that is, $q = 19$. It has primitive roots {2, 3, 10, 13, 14, 15}. We choose $\alpha = 10$.

Alice generates a key pair as follows:

1. Alice chooses $X_A = 5$.
2. Then $Y_A = \alpha^{X_A} \bmod q = \alpha^5 \bmod 19 = 3$.
3. Alice's private key is 5 and Alice's public key is $\{q, \alpha, Y_A\} = \{19, 10, 3\}$.

Suppose Bob wants to send the message with the value $M = 17$. Then:

1. Bob chooses $k = 6$.
2. Then $K = (Y_A)^k \bmod q = 3^6 \bmod 19 = 729 \bmod 19 = 7$.
3. So
   $C_1 = \alpha^k \bmod q = \alpha^6 \bmod 19 = 11$
   $C_2 = \text{KM} \bmod q = 7 \times 17 \bmod 19 = 119 \bmod 19 = 5$
4. Bob sends the ciphertext (11, 5).

For decryption:

1. Alice calculates $K = (C_1)^{X_A} \bmod q = 11^5 \bmod 19 = 161051 \bmod 19 = 7$.
2. Then K^{-1} in GF(19) is $7^{-1} \bmod 19 = 11$.
3. Finally, $M = (C_2 K^{-1}) \bmod q = 5 \times 11 \bmod 19 = 55 \bmod 19 = 17$.

DIFFIE- HELLMAN KEY EXCHANGE

Algorithm for Diffie-Hellman Key Exchange:

Step 1 two public known numbers q, α

q Prime number

α primitive root of q and α< q.

Step 2 if A & B users wish to exchange a key

a) User A select a random integer X_A<q and computes $Y_A = \alpha^{X_A} \bmod q$

b) User B independently select a random integer X_B <q and computes $Y_B = \alpha^{X_B} \bmod q$

c) Each side keeps the X value private and Makes the Y value available publicly to the outer side.

Step 3 User A Computes the key as $K = (Y_B)^{X_A} \bmod q$

User B Computes the key as $K = (Y_A)^{X_B} \bmod q$

Step 4 two calculation produce identical results

$K = (Y_B)^{X_A} \bmod q$

$K = (\alpha^{X_B} \bmod q)^{X_A} \bmod q$ (We know that $Y_B = \alpha^{X_B} \bmod q$)

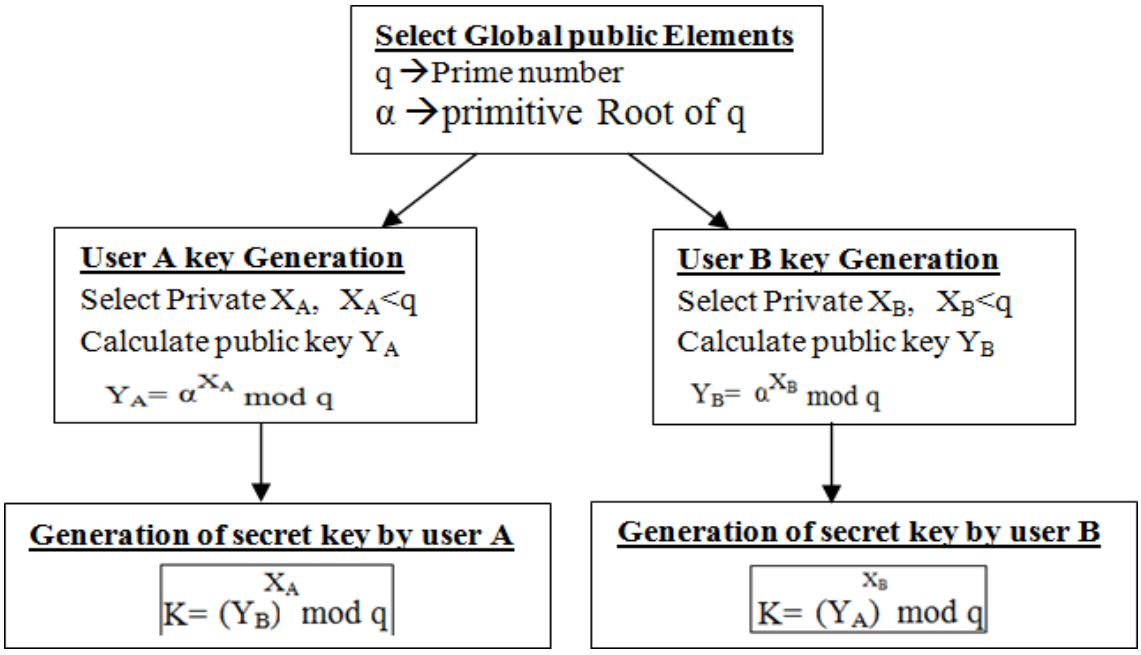
$= (\alpha^{X_B})^{X_A} \bmod q$

$= (\alpha^{X_A})^{X_B} \bmod q$

$= (\alpha^{X_A} \bmod q)^{X_B} \bmod q$

$= (Y_A)^{X_B} \bmod q$ (We know that $Y_A = \alpha^{X_A} \bmod q$)

The result is that the two sides have exchanged a secret key.



Example: 1

Here is an example. Key exchange is based on the use of the prime number $q = 353$ and a primitive root of 353, in this case $\alpha = 3$. A and B select secret keys $X_A = 97$ and $X_B = 233$, respectively. Each computes its public key:

A computes $Y_A = 3^{97} \bmod 353 = 40$.

B computes $Y_B = 3^{233} \bmod 353 = 248$.

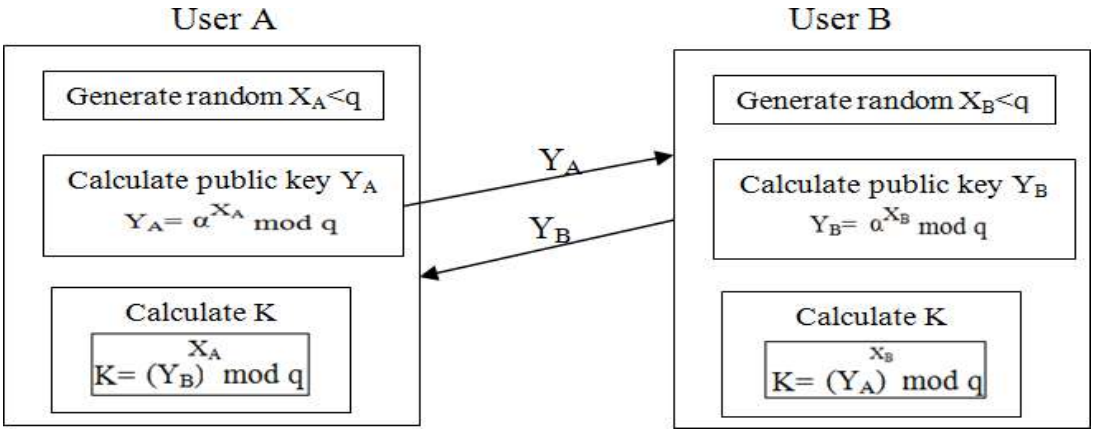
After they exchange public keys, each can compute the common secret key:

A computes $K = (Y_B)^{X_A} \bmod 353 = 248^{97} \bmod 353 = 160$.

B computes $K = (Y_A)^{X_B} \bmod 353 = 40^{233} \bmod 353 = 160$.

We assume an attacker would have available the following information:

$$q = 353; \alpha = 3; Y_A = 40; Y_B = 248$$



Example2:

Here is an example, taken from. Key exchange is based on the use of the prime number q=71 and a primitive root of 71, in this case $\alpha = 7$. A and B select private keys $X_A = 5$ and $X_B = 12$, respectively. Each computes its public key:

$$Y_A = 7^5 = 51 \bmod 71$$
$$Y_B = 7^{12} = 4 \bmod 71$$

After they exchange public keys, each can compute the common secret key:

$$K = (Y_B)^{X_A} \bmod 71 = 45 = 30 \bmod 71$$
$$K = (Y_A)^{X_B} \bmod 71 = 5112 = 30 \bmod 71$$

From {51, 4}, an attacker cannot easily compute 30.

Example 3:

User A and B exchange the key using Diffie-Hellman algorithm. Assume α=5 q=11 X_A=2 X_B=3. Find the value of Y_A, Y_B and k?

Soln:

$$Y_A = \alpha^{X_A} \bmod q$$
$$= 25 \bmod 11$$
$$= 3$$
$$Y_B = \alpha^{X_B} \bmod q$$
$$= 125 \bmod 11$$
$$= 4$$
$$K = (Y_A)^{X_B} \bmod q$$
$$= 27 \bmod 11$$
$$= 5$$
$$K = (Y_B)^{X_A} \bmod q$$
$$= 16 \bmod 11$$
$$= 5$$

MAN-IN-MIDDLE-ATTACK:

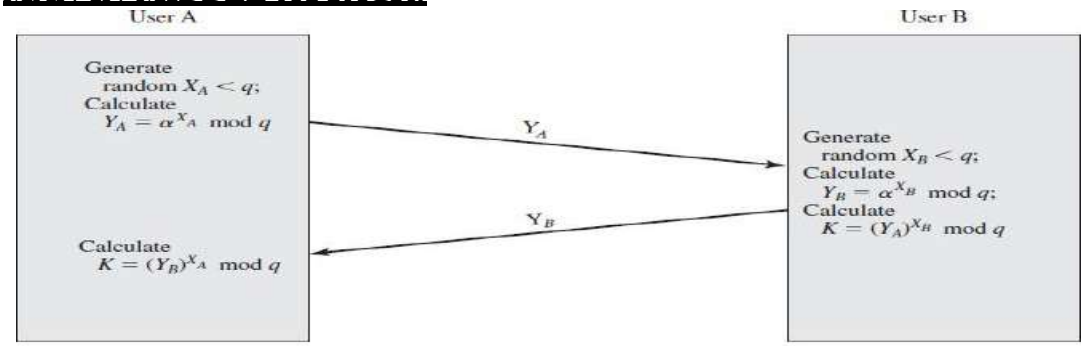


Figure 10.2 Diffie-Hellman Key Exchange

Definition: A man in the middle attack is a form of eavesdropping where communication between two users is monitored and modified by an unauthorized party.

Generally the attacker actively eavesdrops by intercepting (stoping) a public key message exchange.

The Diffie- Hellman key exchange is insecure against a "Man in the middle attack".

Suppose user 'A' & 'B' wish to exchange keys, and D is the adversary (opponent). The attack proceeds as follows.

1. 'D' prepares for the attack by generating two random private keys X_{D1} & X_{D2} and then computing the corresponding public keys Y_{D1} and Y_{D2}.
2. 'A' transmits 'Y_A' to 'B'
3. 'D' intercepts Y_A and transmits Y_{D1} to 'B'. and D also calculates $K2 = (Y_A)^{X_{D2}} \bmod q$.

4. 'B' receives Y_{D1} & calculate $K1 = (Y_{D1})^{X_B} \bmod q$.
5. 'B' transmits 'YB' to 'A"
6. 'D' intercepts 'YB' and transmits YD2 to 'A' and 'D' calculate K1 $K1 = (Y_B)^{X_{D1}} \bmod q$.
7. A receives Y_{D2} and calculates $K2 = (Y_{D2})^{X_A} \bmod q$

At this point, Bob and Alice think that they share a secret key, but instead Bob and Darth share secret key *K*1 and Alice and Darth share secret key *K*2. All future communication between Bob and Alice is compromised in the following way.

1. A sends an encrypted message *M*: E(*K*2, *M*) .
2. D intercepts the encrypted message and decrypts it to recover *M*.
3. D sends B $E(K1, M)$ or $E(K1, M')$, where M' is any message. In the first case, D simply wants to eavesdrop on the communication without altering it. In the second case, D wants to modify the message going to B

The key exchange protocol is vulnerable to such an attack because it does not authenticate the participants. This vulnerability can be overcome with the use of digital signatures and public-key certificates.

ELLIPTIC CURVE CRYPTOGRAPHY

Elliptic Curve Arithmetic:

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.

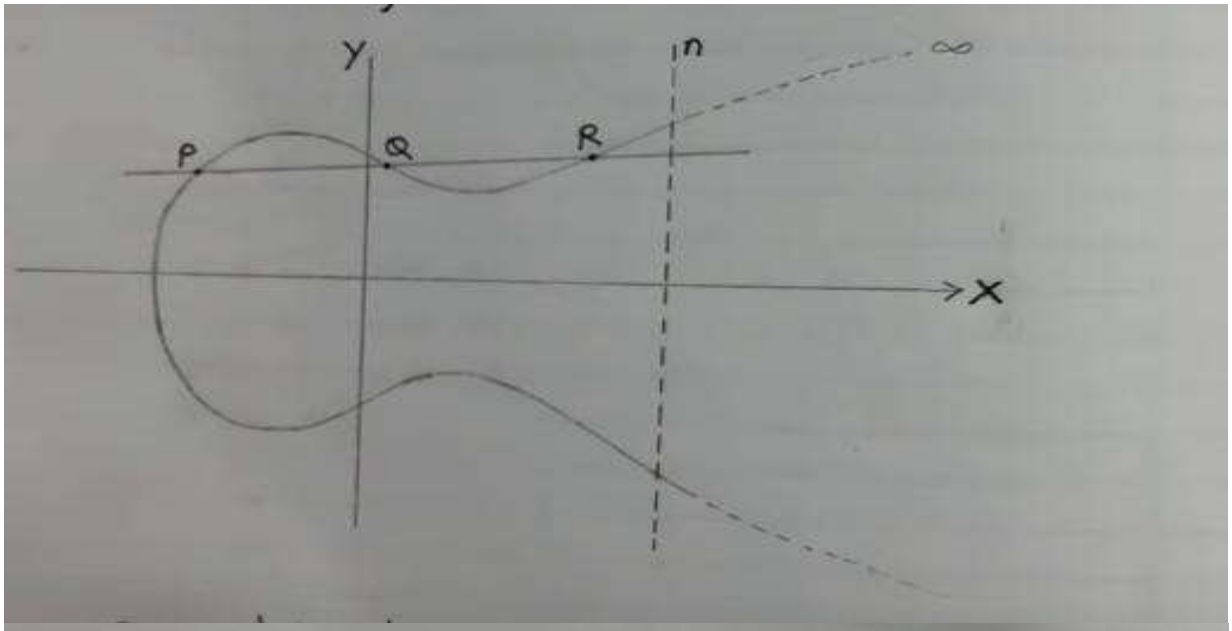
Majority of public-key crypto (RSA, D-H) use either integer or polynomial arithmetic with very large numbers/polynomials. It imposes a significant load in storing and processing keys and messages. An alternatives to use elliptic curves; it offers same security with smaller bit sizes.

Real Elliptic Curves

Elliptic Curve Cryptography (ECC)

- It is an asymmetric / public key cryptosystem
- It provides equal security with smaller key size as compared to non ECC algorithms
- It makes use of elliptic curves
- Elliptic curves are defined by some mathematical functions

Ex - $y^2 = x^3 + ax + b$



- symmetric to x-axis
- If we draw a line, it will touch a maximum of 3 points.

ECC Algorithm

ECC Key Exchange

Global Public Elements -

1) $E_q(a, b)$ - Elliptic curve with parameters a, b of q (prime number or an integer of the form 2^m)
2) G - Point on the elliptic curve

User A key generation

Select private key n_A $\quad n_A < n$

Calculate public key P_A $\quad P_A = n_A \times G$

User B key generation

Select private key n_B $\quad n_B < n$

Calculate public key P_B $\quad P_B = n_B \times G$

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 Encryption

- Let the message be M
- First encode this message M into a point on elliptic curve
  Let this point be P_m

- For encryption, choose a random positive integer k
  The cipher point will be

$$C_m = \{kG, P_m + kP_B\}$$

This point will be sent to the receiver

Decryption

- For decryption, multiply x-coordinate with receiver's secret key

$$kG \times n_B$$

Then subtract $(kG \times n_B)$ from y-coordinate of cipher point

$$P_m + kP_B - (kG \times n_B)$$

we know that $P_B = n_B \times G$

$$\therefore P_m + kP_B - kP_B$$

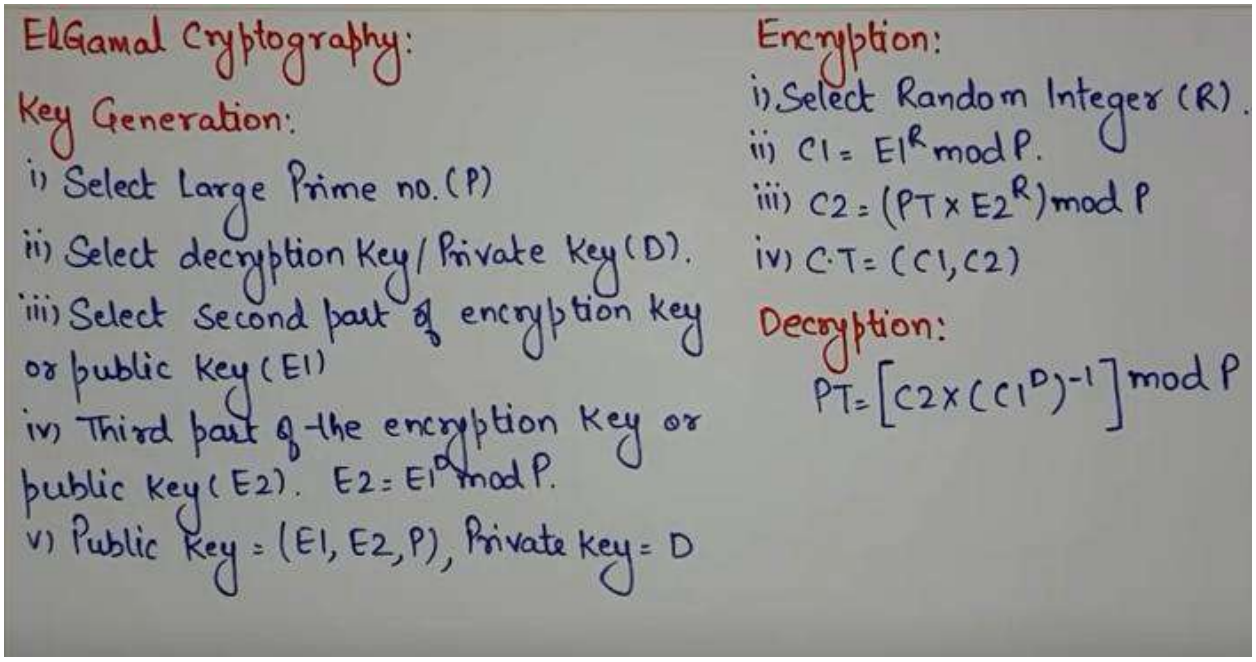
$$= P_m$$

So receiver gets the same point

Comparision of RSA/DSA (Diffie Hellman Algorithm

Key sizes with equivalent security levels

Minimum size (bits) of public keys		
DSA/DH	RSA	ECC
1024	1024	160
2048	2048	224
3072	3072	256
7680	7680	384
15360	15360	512



Elgamal Cryptography: Asymmetruc Key (Encrp-pub key ,Decrypt-Private key)

`Let P=11;D=3;E1=2

$E2=E1^D \bmod p$

$E2= 2^3 \bmod 11$

E2= 8 mod 11

E2=8

Now public key={E1,E2,P}={2,8,11}

Private Key=3

Encryption:

R=4

$C1=E1^R \bmod p$

$C1=2^4 \bmod 11$

C1=16 mod 11=5

Now Pt=7

C2={Pt $E2^R$ mod p}

C2= $7x8^4$ mod 11

C2=28672 mod 11 =6

C2=6

C.T ={5,6}

Decryption

Pt={C2 $(C1^D)^{-1}$ mod p}

Pt={6x$(5^3)^{-1}$)mod 11

First: $(5^3)^{-1}$)mod 11

$(125)^{-1}$)mod 11

125 *X mod 11=1

125*1 mod 11= 121 +4 mod 11=4

125*2 mod 11= 250 mod 11= 242+8 mod11=8

125*3 mod 11=375 mod 11= 374+1 mod 11=1

Pt={6*3 mod 11}

Pt=18 mod 11=7

Pt=7

So the Decrypt and encrypt Ptvalue is same .