MODULE 2- Public Key Cryptography and RSA

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:

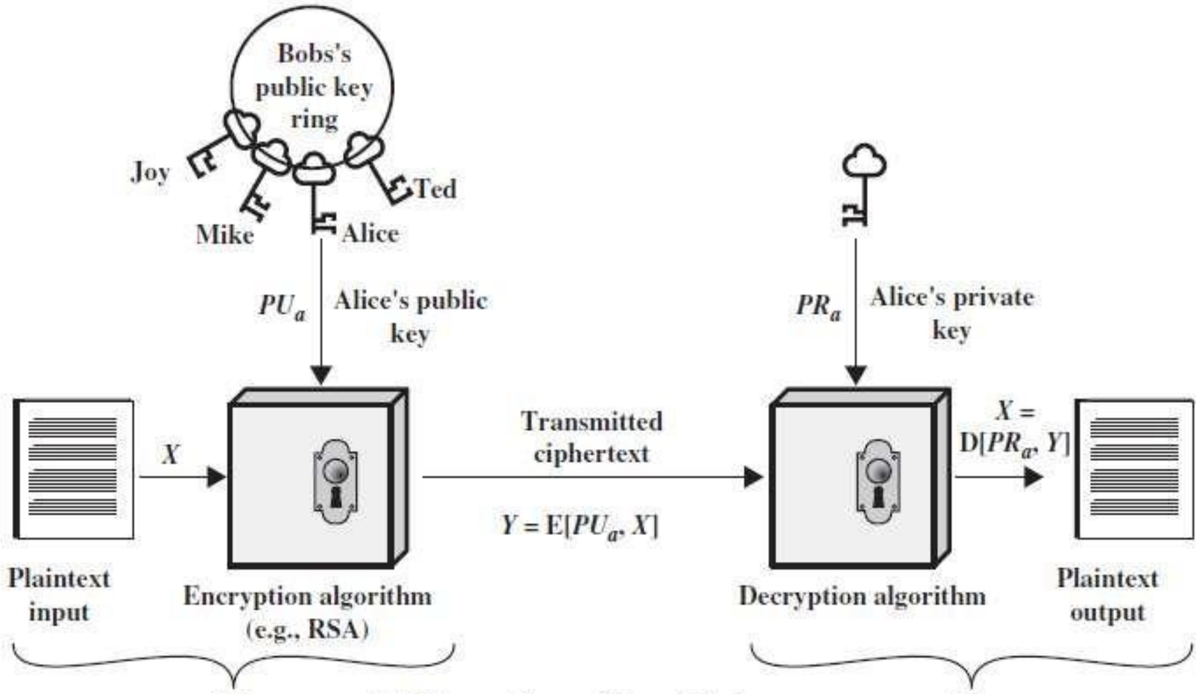
- The concept of public key cryptography in invented for two most difficult problems of Symmetric key encryption.
  - The Key Exchange Problem
  - The Trust Problem

  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 where as 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)



(a) Encryption with public key

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.As Figure 9.1a suggests, 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.

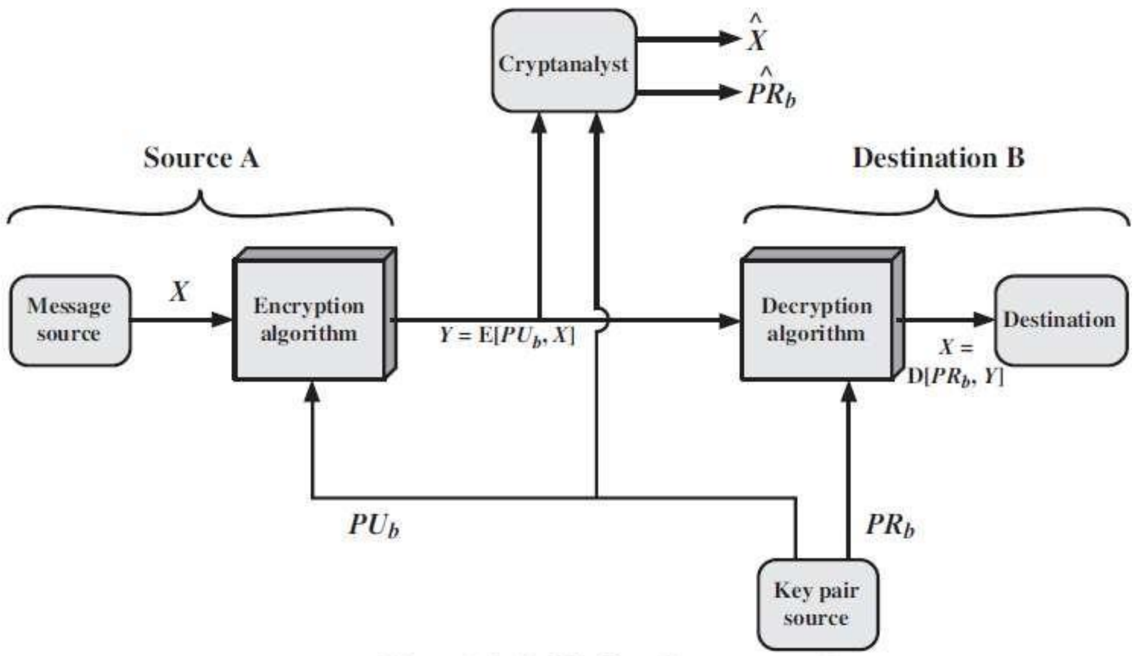


Figure 9.2 Public-Key Cryptosystem: Secrecy

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

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

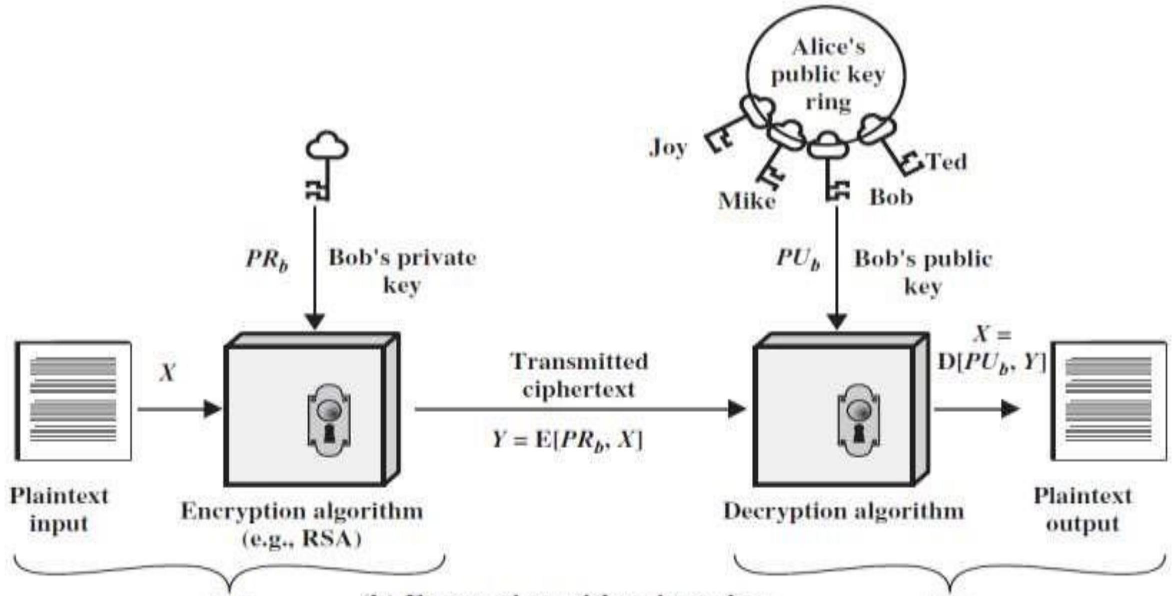
PR_b is known only to B, whereas PU_b is publicly available and therefore accessible by A.
With the message X and the encryption key PU_b as input, A forms the ciphertext $Y = [Y1, Y2, . . . , Y_N]$:

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

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:



(b) Encryption with private key

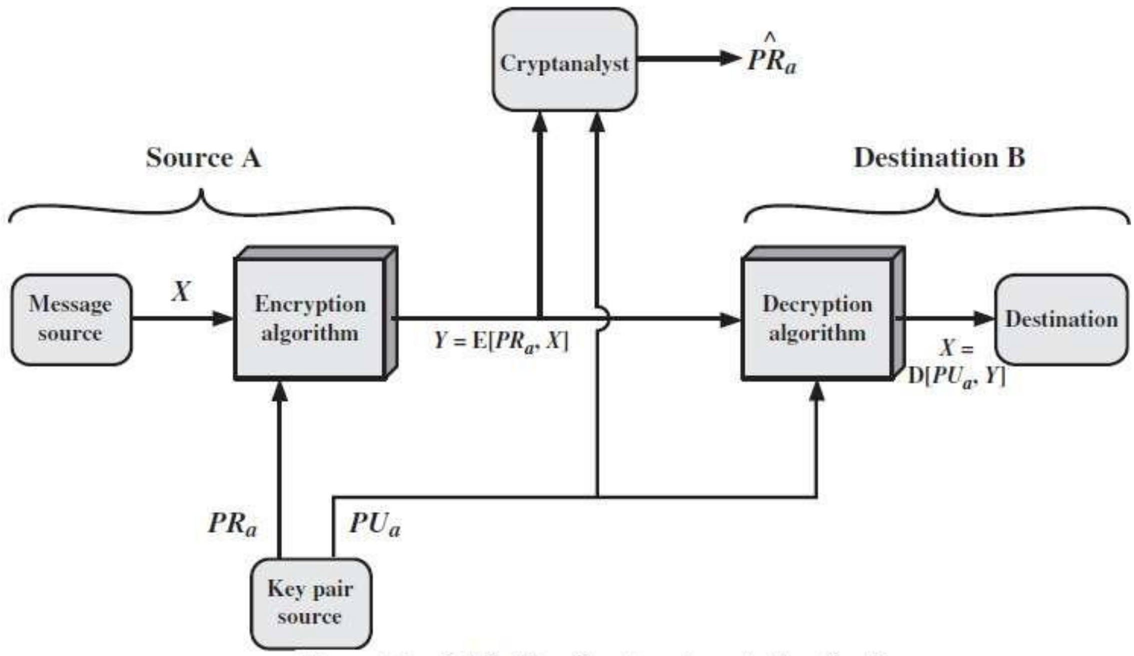


Figure 9.3 Public-Key Cryptosystem: Authentication

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)

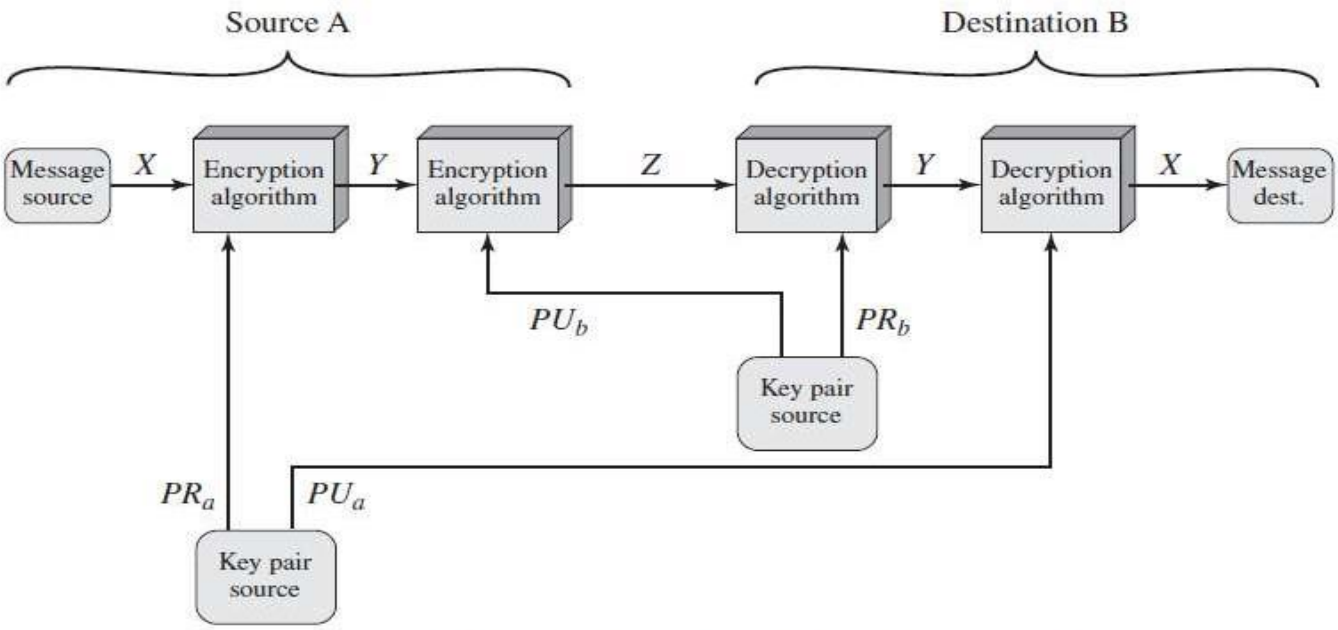


Figure 9.4 Public-Key Cryptosystem: Authentication and Secrecy

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

Public-key systems are characterized by
the use of a cryptographic algorithm with two keys, one held private and one available publicly. Depending on the application, the sender uses either the sender's
private key or the receiver's public key, or both, to perform some type of cryptographic function.
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.

Applications for Public-Key Cryptosystems

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

Public-Key Cryptanalysis

As with symmetric encryption, a public-key encryption scheme is vulnerable to a brute-force attack. The countermeasure is the same: Use large keys. However, there is a tradeoff to be considered. Public-key systems depend on the use of some sort of invertible mathematical function. The complexity of calculating these functions may not scale linearly with the number of bits in the key but grow more rapidly than that. Thus, the key size must be large enough to make brute-force attack impractical but small enough for practical encryption and decryption. In practice, the key sizes that have been proposed do make brute-force attack impractical but result in encryption/decryption speeds that are too slow for general-purpose use. Instead, as was mentioned earlier, public-key encryption is currently confined to key management and signature applications.

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 $\mathbf{\log_2(n)}$
- **RSA** uses two exponents 'e' and 'd' where e→public and d→private.
- Encryption and decryption are of following form, for some PlainText 'M' and CipherText block 'C'

$$\boxed{C = M^e \bmod n}$$

$$\boxed{M = C^d \bmod n}$$

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

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)→ 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 “ $\boxed{\mathbf{ed \bmod Ø(n)=1}}$ “
   this is equivalent to saying

$$\boxed{\mathbf{ed \equiv 1 \bmod Ø(n)}}$$
$$\boxed{\mathbf{d \equiv e^{-1} \bmod Ø(n)}}$$

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 3→Calculate Ø(n)=(p-1)(q-1)
Step 4→ Select or find integer e (public key) which is relatively prime to Ø(n).
ie., e with gcd (Ø(n), e)=1 where 1<e< Ø(n).
Step 5→ Calculate “d” (private key) by using following condition. $\boxed{\mathbf{ed \equiv 1 \bmod Ø(n)}}$ d< Ø(n).
Step 6→ Perform encryption by using $\boxed{C = M^{e} \bmod n}$
Step 7→ perform Decryption by using $\boxed{M = C^{d} \bmod n}$

Example:
1. Select two prime numbers, p = 17 and q = 11.
2. Calculate $n = pq$ = 17 × 11 = 187.
3. Calculate Ø(n) = (p - 1)(q - 1) = 16 × 10 = 160.
4. Select e such that e is relatively prime to Ø(n) = 160 and less than Ø (n); we choose e = 7.
5. Determine d such that ***de*** **≡1 (mod 160)** and d < 160.The correct value is d = 23, because 23 * 7 = 161 = (1 × 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.

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

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

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

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

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

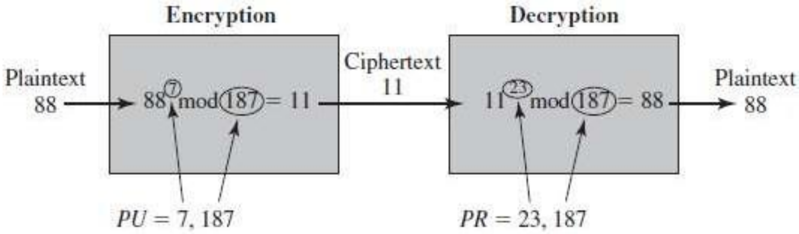


Figure 9.6 Example of RSA Algorithm

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

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

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

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

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

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

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

The Security of RSA

Four possible approaches to attacking the RSA algorithm are

- **Brute force:** This involves trying all possible private keys.
- **Mathematical attacks:** There are several approaches, all equivalent in effort to factoring the product of two primes.
- **Timing attacks:** These depend on the running time of the decryption algorithm.
- **Chosen ciphertext attacks:** This type of attack exploits properties of the RSA algorithm.

Diffie-Hellman Key Exchange:

- Diffie-Hellman key exchange is the first published public key algorithm
- This Diffie-Hellman key exchange protocol is 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.
- This algorithm depends for its effectiveness on the difficulty of computing discrete logarithms.
- The discrete logarithms are defined in this algorithm in the way of define a primitive root of a prime number.
    - Primitive root: we define a primitive root of a prime number P as one whose power generate all the integers form 1 to P-1 that is if 'a' is a primitive root of the prime number P, then the numbers $a \bmod P, a^2 \bmod P, a^3 \bmod P, \ldots\ldots a^{P-1} \bmod P$ are distinct and consist of the integers form 1 through P-1 in some permutation.
    For any integer 'b' and 'a', here 'a' is a primitive root of prime number P, then
    $b \equiv a^i \bmod P \qquad 0 \leq i \leq (P-1)$

The exponent i → is refer as discrete logarithm or index of b for the base a, mod P.
The value denoted as **$ind_{a,p}(b)$**

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

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

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 $Y_B = \alpha^{X_B} \bmod q$)

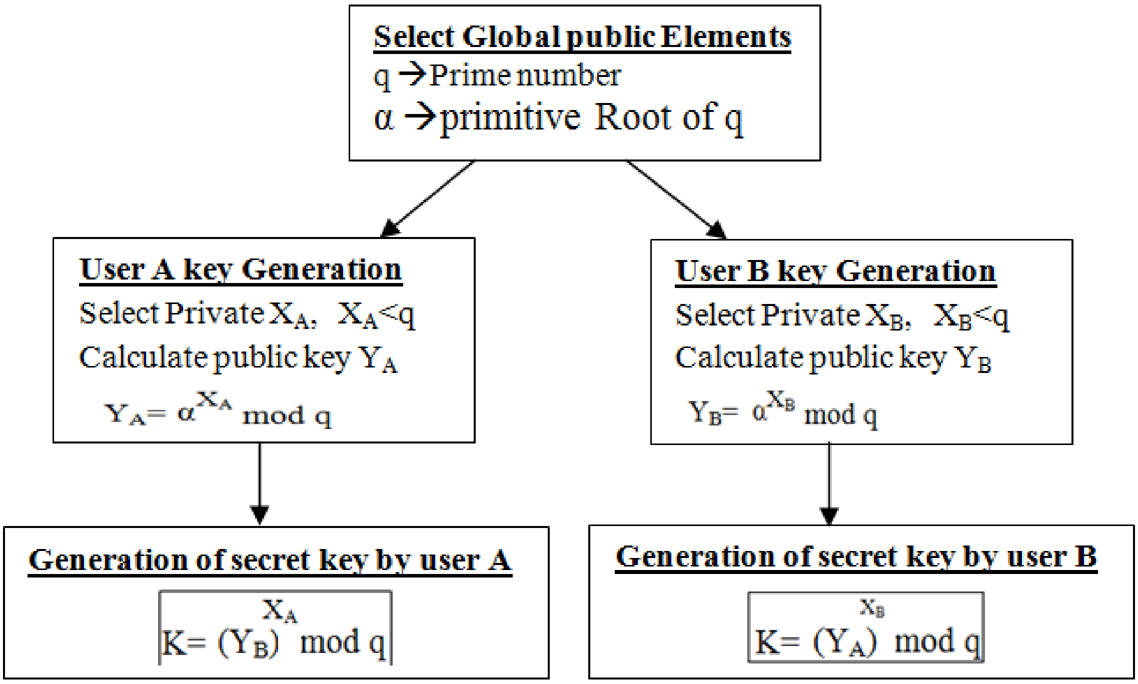
$$= (\alpha^{X_B})^{X_A} \bmod q$$

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

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

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

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



Example:

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

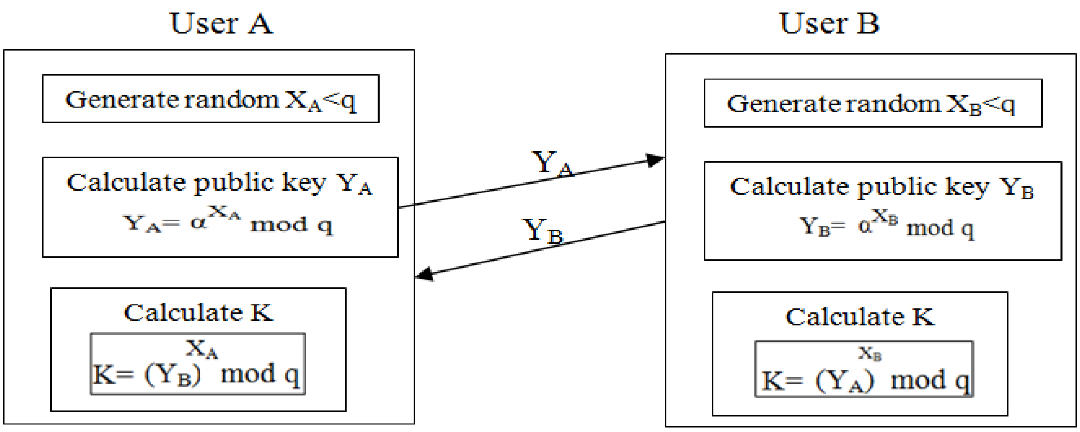
After they exchange public keys, each can compute the common secret key:

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:

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



MAN-in the Middle Attack (MITM)

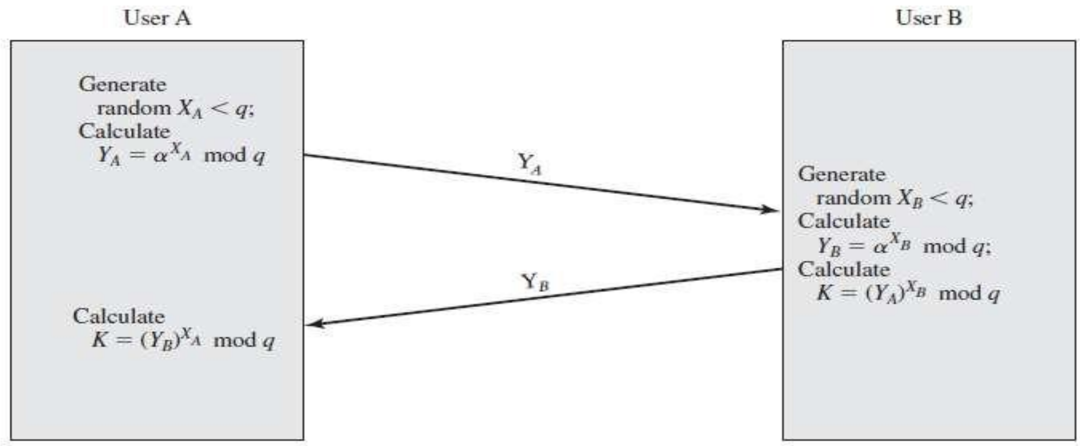


Figure 10.2 Diffie-Hellman Key Exchange

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 'Y_B' to 'A"
6. 'D' intercepts 'Y_B' and transmits Y_{D2} 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 $K1$ and Alice and Darth share secret key $K2$. All future communication between Bob and Alice is compromised in the following way.

1. A sends an encrypted message M: $E(K2, M)$.
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

- **Definition: Elliptic curve cryptography (ECC)** is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. These are analogy of existing public key cryptosystem in which modular arithmetic is replaced by operations defined over elliptic curve.
- The use of elliptic curves in cryptography was suggested independently by **Neal Koblitz** and **Victor S. Miller** in **1985.**
- Elliptic curve cryptography (ECC) is one of the most powerful but least understood types of cryptography in wide use today. An increasing number of websites make extensive use of ECC to secure everything from customers' HTTPS connections to how they pass data between data centers.

An elliptic curve is defined by an equation in two variables with coefficients. For cryptography, the variables and coefficients are restricted to elements in a finite field, which results in the definition of a finite abelian group.

Elliptic Curves over Real Numbers

Elliptic curves are not ellipses. They are so named because they are described by cubic equations,

$y^2 + axy + by = x^3 + cx^2 + dx + e$ is similar to equation of calculating circumference of an ellipse.

Where

a,b,c,d and e → real numbers.

X and Y are→taken on values in the real numbers.

For utilization of this in cryptography

$y^2 = x^3 + ax + b$ → EQ1, is sufficient.

Such equations are said to be cubic, or of degree 3, because the highest exponent they contain is a 3. Also included in the definition of an elliptic curve is a single element denoted *O* and called the *point at infinity* or the *zero point. To plot such a curve, we need to compute*

$y = \sqrt{x^3 + ax + b}$ For given values of and, the plot consists of positive and negative values of for each value of . Thus, each curve is symmetric about $y = 0$.

Two families of elliptic curves are used in cryptographic applications:

- Prime curves over $\mathbf{Z}_p$ ***[it is** Best for software application]*
- Binary curves over **GF(2^m)** ***[it is** Best for software application]*

Prime curves over $\mathbf{Z}_p$

In Prime curves over $\mathbf{Z}_p$, ***p*** ⇒referred to as a modulus.

we use a cubic equation in which the variables and coefficients all take on values in the set of integers from **0** through ***p* - 1** and in which calculations are performed modulo *p*.

from EQ1, in this case coefficients and variables limited to $\mathbf{Z}_p$.

$$y^2 \bmod p = (x^3 + ax + b) \bmod p$$ → eq2

Now consider the set E*p*(*a*, *b*) consisting of all pairs of integers (*x*, *y*) that satisfy Equation eq2 together with a point at infinity .The coefficients a and b and the variables x and y are all elements of Zp.

For example, Equation eq2 is satisfied for $a = 1$, $b = 1$, $x = 9$, $y = 7$, $\alpha = 1$ $p = 23$:

$$7^2 \bmod 23 = (9^3 + 9 + 1) \bmod 23$$
$$49 \bmod 23 = 739 \bmod 23$$
$$3 = 3$$

For example, let *p* = 23 and consider the elliptic curve *y*2 = *x*3 + *x* + 1 In this case, *a* = *b* = 1 For the set $E_{23}(1, 1)$, we are only interested in the nonnegative integers in the quadrant from (0, 0) through (*p* - 1, *p* - 1) that satisfy the equation mod *p*.

Elliptic Curves over GF(2^m):

A finite field GF(2^m) consists of 2m elements, together with addition & multiplication operations that can be defined over polynomials.

For elliptic Curves over GF(2^m), we use a cubic equation in which the variables and coefficients all take on values in GF(2^m), for some number m.
By this, the form of cubic equation appropriate for cryptographic application.

The form is $y^2 + xy = x^3 + ax^2 + b$ → EQ3.

To form a cryptographic system using elliptic curves, we need to find a "hard problem" corresponding to factoring the product of two primes or taking the discrete logarithm.

Consider the equation $Q = kP$ where $Q, P \in E_p(a, b)$ and $k < p$.
It is relatively easy to calculate *Q* given *k and P*
But it is relatively hard to determine given Q and P.
This is called the discrete logarithm problem for elliptic curves.

<u>ECC Diffie-Hellman Key Exchange</u>:

ECC can do key exchange, that is analogous to Diffie Hellman.
Key exchange using elliptic curves can be done in the following manner.
First pick a large integer q , which is either a prime number P or an integer of the form 2^m and elliptic curve parameters a & b for equation $y^2 \bmod p = (x^3 + ax + b) \bmod p$ or

$y^2 + xy = x^3 + ax^2 + b$.

This define elliptic group of point $E_q(a,b)$.
Pick a base point G=(x1,y1) in $E_p(a,b)$ whose order is a very large value n.
The order n of a point G on an elliptic curve is the smallest +ve integer n such that nG=0.$E_q(a,b)$

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

$K = n_B \times P_A$

Figure 10.7 ECC Diffie-Hellman Key Exchange

$$n_A \times P_B = n_A \times (n_B \times G) = n_B \times (n_A \times G) = n_B \times P_A$$

Elliptic Curve Encryption/Decryption:

Several approaches to encryption/decryption using elliptic curves have been analyzed in the literature. In this subsection, we look at perhaps the simplest. The first task in this system is to encode the plaintext message m to be sent as an x–y point P_m. It is the point P_m that will be encrypted as a ciphertext and subsequently decrypted. Note that we cannot simply encode the message as the x or y coordinate of a point, because not all such coordinates are in $E_q(a, b)$; for example, see Table 10.1. Again, there are several approaches to this encoding, which we will not address here, but suffice it to say that there are relatively straightforward techniques that can be used.

As with the key exchange system, an encryption/decryption system requires a point G and an elliptic group $E_q(a, b)$ as parameters. Each user A selects a private key n_A and generates a public key $P_A = n_A \times G$.

To encrypt and send a message P_m to B, A chooses a random positive integer k and produces the ciphertext C_m consisting of the pair of points:

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

Note that A has used B's public key P_B. To decrypt the ciphertext, B multiplies the first point in the pair by B's secret key and subtracts the result from the second point:

$$P_m + kP_B - n_B(kG) = P_m + k(n_B G) - n_B(kG) = P_m$$

A has masked the message P_m by adding kP_B to it. Nobody but A knows the value of k, so even though P_b is a public key, nobody can remove the mask kP_B. However, A also includes a "clue," which is enough to remove the mask if one knows the private key n_B. For an attacker to recover the message, the attacker would have to compute k given G and kG, which is assumed to be hard.

As an example of the encryption process (taken from [KOBL94]), take $p = 751$; $E_p(-1, 188)$, which is equivalent to the curve $y^2 = x^3 - x + 188$; and $G = (0, 376)$. Suppose that A wishes to send a message to B that is encoded in the elliptic point $P_m = (562, 201)$ and that A selects the random number $k = 386$. B's public key is $P_{B = (201, 5)}$. We have $386(0, 376) = (676, 558)$, and $(562, 201) + 386(201, 5) = (385, 328)$. Thus, A sends the cipher text $\{(676, 558), (385, 328)\}$.