ciphers. It is also used for the cryptanalysis of stream ciphers. The basic architecture of DES is based on substitution and diffusion.

3.10 LINEAR CRYPTANALYSIS

Another form of cryptanalysis technique is linear cryptanalysis which is based on linear approximations. This cryptanalysis technique can be used against both the stream and block ciphers. The loopholes in the cipher can be found out using linear cryptanalysis. This helps to improve the performance of the cipher. In this technique, both plaintext and ciphertext are used for cryptanalysis. The key is found out by using the plaintext through a simplified cipher in the complete ciphertext. XOR operation is used to get the key. Some bits of the plaintext are XOR, also some bits of the ciphertext are XOR. The result is XOR together. This result is same as the XOR of some bits of the key. This helps to get the complete key.

3.11 WEAK KEYS IN DES ALGORITHMS

The performance of any encryption algorithm is based on the keys used. But all the keys may not be strong. Some ciphertexts are relatively easy for cryptanalysis. The keys used for generation of such ciphertexts are called *weak keys*. The keys having high degree of similarity are called *simple weak key*.

For example, if any key is composed of:

- all bits are zeros,
- all bits are ones,
- alternating bits are ones and zeroes, and
- alternating bits are zeroes and ones, weak keys have one of the above combinations. For DES, there are total 2^{56} keys of which sixteen keys are considered as weak keys. With the above combinations, following 16 weak keys are produced:

Table 3.7 Weak keys

	L_0	L_0	L_0	L_0
R_0	0,0..	1,0..	0,1..	1,1..
R_0	0,0..	0,0..	0,0..	0,0..
R_0	1,0..	1,0..	1,0..	1,0..
R_0	0,1..	0,1..	0,1..	0,1..
R_0	**1,1..**	**1,1..**	**1,1..**	**1,1..**

From Table 3.7, two keys which have all the bits of L_0 and R_0 as zeroes or ones. These keys are weak because they have their own inverses. Permutation and shifting does not change the key. Therefore, subkeys of these two keys are the same keys. Therefore, all the rounds have the same key. Other than these two subkeys, there are two other keys having each half all ones or zeroes. That means left 28 bits are zeroes and right 28 bits are ones and vice-versa. These four keys are very weak keys and recommended for not to use. Other twelve keys are the combinations of zeroes and ones, such as alternate bits in the key are ones and zeroes as per the table. These twelve keys are called *semi-weak keys*. For good encryption it is recommended not to select such keys.

EXAMPLE 3.1 Let the message be M = COMPITDT and the key be K = COEPPUNE. USE DES algorithm to encrypt and decrypt the message.

Convert M to ASCII and rewrite it in binary format, we get the 64-bit block of plaintext:

M = 01100011 01101111 01101101 01110000 01101001 01110100 01100100 01110100
L = 01100011 01101111 01101101 01110000
R = 01101001 01110100 01100100 01110100

We first write the message in 8 X 8 matrix form as below:

01100011
01101111
01101101
01110000
01101001
01110100
01100100
01110100

The first bit of M is "0". The last bit is "0". We read from left to right.
Convert K to ASCII and rewrite it in binary format, we get the 64-bit key as:

K = 01100011 01101111 01100101 01110000 01110000 01110101 01101110 01100101
Write the key in 8 X 8 matrix form as below:

01100011
01101111
01100101
01110000
01110000
01110101
01101110
01100101

Solution The DES algorithm uses the following steps.

Step 1 *Generate 16* subkeys (48-bit length)

Round=1 key=111000001011111011101110110100000100000010011110
Round=2 key=111000001011011011110110100100011010110110000100
Round=3 key=111101001101111001110110001010000010011010010001
Round=4 key=111001101111001101110010011110110110000000000111
Round=5 key=101011101101011101110111001001100100000110001010
Round=6 key=111011110101001101011011100001000011000101000111
Round=7 key=001011111101001111111001111001101000001011100000
Round=8 key=100111101011001110110110101000010001111101001011

Round=9 key=000111110100101111011011011001001001010100001100
Round=10 key=001111110111100110011101100010000001010011101110
Round=11 key=000111110010110111001101010011001101101010100001
Round=12 key=010110110110110010111101000100100100110001111001
Round=13 key=110111011010110110101100100010111001100100010000
Round=14 key=110100101010111010101111100000010110011100110000
Round=15 key=111110011011111100010011001111001000010100000100
Round=16 key=1111000110111111000101110000100101000001001110100

Plaintext after rounds

10100010
00001111
11100011
01010000
01011100
11101111
01010011
00001110

Printing Ciphertext in int form

60 126 178 178 137 100 173 100

Ciphertext generated:

< ~ ² ² ‰ d d

---------DECRPTION-------------

After initial permutation

1 0 1 0 0 0 1 0
0 0 0 0 1 1 1 1
1 1 1 0 0 0 1 1
0 1 0 1 0 0 0 0
0 1 0 1 1 1 0 0
1 1 1 0 1 1 1 1
0 1 0 1 0 0 1 1
0 0 0 0 1 1 1 0

Plaintext matrix after Decryption

0 1 1 0 0 0 1 1
0 1 1 0 1 1 1 1
0 1 1 0 1 1 0 1
0 1 1 1 0 0 0 0
0 1 1 0 1 0 0 1
0 1 1 1 0 1 0 0
0 1 1 0 0 1 0 0
0 1 1 1 0 1 0 0

After Decryption
compitdt

algorithm. ... as weak key

EXERCISES

3.1 What is a block cipher? Explain various modes of the operation of block cipher.

3.2 What are the advantages of CTR mode?

3.3 What are the design parameters of Feistel cipher?

3.4 Explain the working of DES in detail.

3.5 Explain the key transformation in DES.

3.6 Discuss triple DES.

3.7 Explain the modes of operation in triple DES.

3.8 Discuss the design criteria for DES.

3.9 Explain differential cryptanalysis.

3.10 What is linear cryptanalysis?

3.11 Compare the modes of operation in triple DES and DES.

3.12 During the transmission of *C*4 (the fourth cipher block) an error in the 3rd bit occurred. How many plaintext blocks will be affected, if we are using:16-bit CFB mode for DES? Explain why?

curve groups, these user-defined operations are defined geometrically. The underlying fields can be created by the number of points on a curve.

8.4.1 Elliptic Curve Groups Over Real Numbers

Over a hundred people studied elliptic curves. An elliptic curve E over the real numbers is the set of points (x, y). It is a graph of an equation of the form:

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

where x, y, a and b are real numbers. It also includes a point at infinity.

Each choice of the numbers a and b yield a different elliptic curve. For example, $a = -3$ and $b = 3$ give the elliptic curve with equation $y^2 = x^3 - 3x + 3$; the graph of this curve is shown in Figure 8.5.

If the given equation for elliptic curve has no repeated factors, then the given equation of elliptic curve can be used to form a group. The corresponding points on a curve form a group over real numbers with a special point O. This point O is called the *point at infinity*.

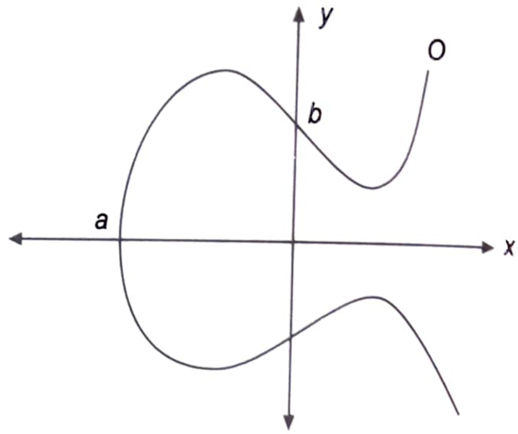

Figure 8.5 Elliptic curve.

8.4.2 Elliptic Curve Addition: A Geometric Approach

The basic function of elliptic curve groups is addition, so it is additive groups. The addition of any two points on the elliptic curve can be defined geometrically.

The negative of any point $P(x_p, y_p)$ lies on the elliptic curve is $-P(x_p, -y_p \bmod P)$. If any point P lies on the elliptic curve then point $-P$ also lies on the curve.

Adding Distinct Points P and Q

Suppose $P(x_P, y_P)$ and $Q(x_Q, y_Q)$ are two distinct points on the elliptic curve such that Q is not $-P$. The point where line PQ intersects the curve is $-R$ and its reflection against x-axis is R (Figure 8.6). Then

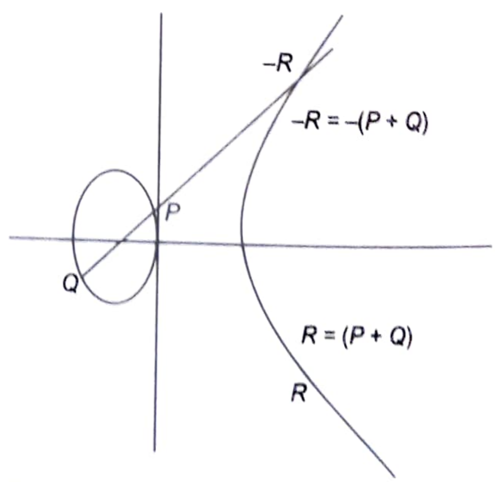

Figure 8.6 Adding distinct points P and Q.

$$P + Q = R$$

where R is the point where line PQ intersects the curve.

$$m = (y_P - y_Q)/(x_P - x_Q) \bmod P$$

$$x_R = m^2 - (x_P + x_Q) \bmod p \text{ and } y_R = -y_P + m(x_P - x_R) \bmod P$$

where m is the slope of the line PQ.

Adding the Points R and –R

If the two points R and $-R$ join by a vertical line, it does not intersect the elliptic curve at any point other than R and $-R$. Therefore, we cannot add R and $-R$ as P and Q. Due to this, the point at infinity O is added to the elliptic curve group. O is the additive identity of the elliptic curve group. All the elliptic curves have an additive identity.

By addition property,

$$R + (-R) = O.$$

Therefore, we get

$R + O = R$ is in the elliptic curve group.

Figure 8.7 illustrated this property.

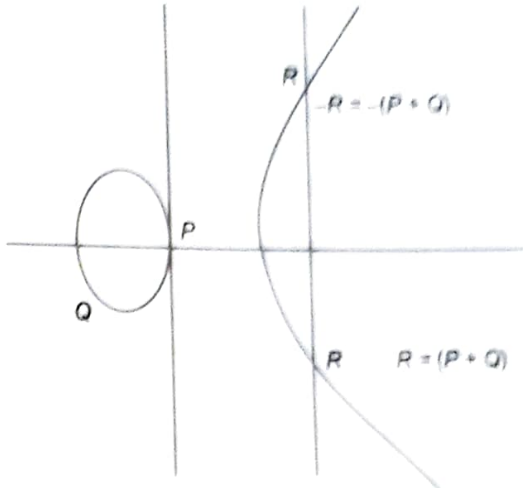

Figure 8.7 Adding the points R and $-R$.

Doubling the Point Q

Now, suppose we want to add a point Q in the group. Draw a tangent to the curve at point Q. If the y-coordinate of Q is not 0, then the tangent intersects the elliptic curve at exactly one other point. That point is $-R$. The reflection of $-R$ against x-axis is R. This is shown in Figure 8.8. This operation helps to double the point so it is called *doubling the point Q*.

The law for doubling a point on an elliptic curve group is defined by:

- If y-coordinate y_Q is 0, the tangent from Q is always vertical.
- If $y_Q = 0$, then doubling the point Q.
- If $y_q = 0$, then the tangent to the elliptic curve at Q is vertical and it does not intersect the elliptic curve at any other point as shown in Figure 8.9.

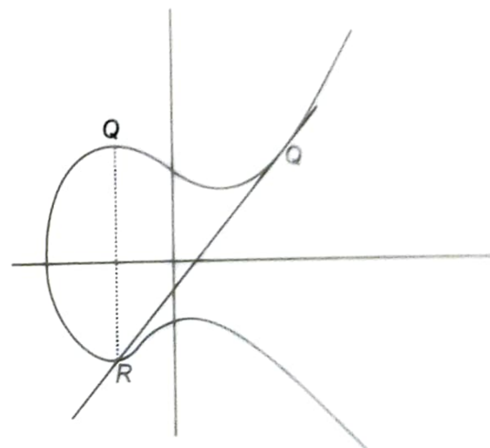

Figure 8.8 Doubling the point Q.

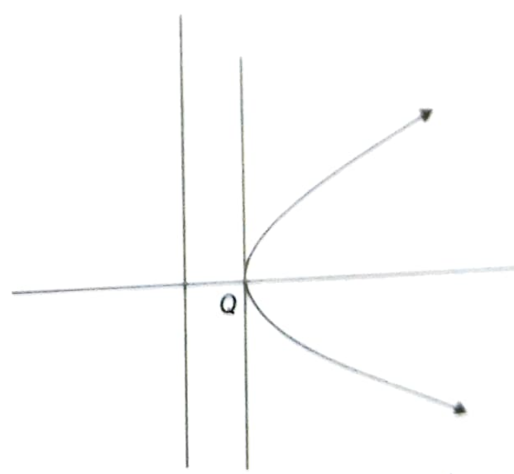

Figure 8.9 The tangent from Q is always vertical if $y_Q = 0$.

By definition, $2Q = 0$ for a given point Q.

If one wanted to find $3Q$ in this situation, one can add $2Q + Q$. This becomes $Q + 0 = Q$

Thus, $\quad 3Q = Q.$

$$3Q = Q,\ 4Q = 0,\ 5Q = Q,\ 6Q = 0,\ 7Q = Q,\ 8Q = 0,\ 9Q = 0, \text{etc}$$

8.4.3 Elliptic Curve Addition: An Algebraic Approach

Above approach of elliptic curves provides an excellent method of illustrating elliptic curve arithmetic, but it is not a practical method for implementing arithmetic computations. So, there should be a method to construct algebraic formulae to efficiently compute the geometric arithmetic.

Adding Distinct Points P and Q

When two points on the elliptic curve, P and Q are not negative of each other, then

$$P + Q = R$$

where

$$m = (y_P - y_Q)/(x_P - x_Q)$$

$$x_R = m^2 - x_P - x_Q \text{ and } y_R = -y_P + m(x_P - x_R)$$

Note that m is the slope of line PQ.

Doubling the Point P

When y_P is not 0,

$$2P = R$$

where

$$m = (3x_P^2 + a)/(2y_P).$$

$$x_R = m^2 - 2x_P \text{ and } y_R = -y_P + m(x_P - x_R)$$

We know one of the parameters chosen with the elliptic curve is a and m is the slope of tangent on the point P.

8.4.4 Elliptic Curve Groups over F_P

Above approach use real numbers which make the execution of the algorithm very slow. At the same time rounding off the real number gives approximate results. Due to all these reasons, if we use this approach for cryptography, the performance of the cryptographic algorithms deteriorate. Cryptographic algorithms require fast and precise arithmetic. Thus, the finite fields of F_P and F_2m are used in place of real number arithmetic. The field F_P uses the numbers from 0 to $P - 1$. The computations will result in an integer between 0 to $P - 1$.

For example, in F_{29} the field is composed of integers from 0 to 28, and any operation within this field will result an integer also in between 0 and 28.

An elliptic curve of F_P can be formed by selecting a and b as coefficients. The coefficients a and b are the integer numbers from 0 to $P - 1$, the field of F_P. The elliptic curve includes all points (x, y) which satisfy the elliptic curve equation modulo P (where x and y are numbers in F_P).

For example: if a and b are in F_P then $y^2 \bmod p = (x^3 + ax + b) \bmod P$ has an underlying field of F_P. The elliptic curve can be used to form a group if the term $x^3 + ax + b$ contains no repeating factors. An elliptic curve group over F_P consists of the points on the corresponding elliptic curve, together with a special point O called the point at infinity. There are finitely many points on an elliptic curve.

Example of an Elliptic Curve Group over F_P

Suppose, an elliptic curve over the field F_{13}. With $a = 1$ and $b = 0$, the elliptic curve equation is $y^2 = x^3 + x$. The point (3, 11) satisfies this equation since:

$$y^2 \bmod P = x^3 + x \bmod P$$

$$121 \bmod 13 = 27 + 3 \bmod 13$$

$$4 \bmod 13 = 30 \bmod 13$$

$$4 = 4$$

Here $P = 13$, therefore, there are 13 points which satisfy the given equation. These points are:

(0, 0), (2, 3), (2, 10), (3, 2), (3, 11), (6, 1), (6, 12), (7, 5), (7, 8), (9, 6), (9, 7), (11, 4), (11, 9)

If we observe the above points, for every value of x, there are two points. The graph is symmetric about $y = 6.5$. Over the field of F_{13}, the negative components in the y-values are taken modulo 13, resulting in a positive number as a difference from 13. Here $-P = (x_P, (-y_P \bmod 13))$.

8.4.5 Arithmetic in an Elliptic Curve Group over F_P

Elliptic curve groups over F_P and over real numbers have following difference:

1. There are finite numbers of points in elliptic curve groups over F_P. As some of the points are discrete, there is a problem of connecting these points to get a smooth curve.
2. It is difficult to apply geometric relationships. As a result, the geometry used in one group cannot be used for other groups. But, the algebraic rules of one group can be applied for other groups.
3. Due to use of real number, there is round off error in elliptic curves over real numbers. In the field of F_P there is no round-off error.

Adding Distinct Points P and Q

The negative of the point P is $-P$ where $x_P = x_P$ and $y_P = -y_P \bmod p$. If P and Q are distinct points such that P is not $-Q$, then

$$P + Q = R$$

where

$$m = (y_P - y_Q)/(x_P - x_Q) \bmod P$$

$$x_R = m^2 - x_P - x_Q \bmod p \text{ and } y_R = -y_P + m(x_P - x_R) \bmod p$$

Note that m is the slope of the line through P and Q.

Doubling the Point P

Suppose y_P is not 0,

$$2P = R$$

where

$$m = (3x_P^2 + a)/(2y_P) \bmod P$$

$$x_R = m^2 - 2x_P \bmod p \text{ and } y_R = -y_P + m(x_P - x_R) \bmod P$$

a is the parameter selected with the elliptic curve and m is the slope of the line PQ.

8.4.6 Elliptic Curve Groups over F_2n

The rules for arithmetic in F_2n can be defined by two ways:

1. Polynomial representation
2. Optimal normal basis representation.

With F_2n, an elliptic curve is formed by selecting a and b within F_2n (if $b \neq 0$). The elliptic curve equation for F_2n having a characteristic 2 is:

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

Elliptic curve equation over F_2n satisfies for all points (x, y). These points together with a point at infinity form the elliptic curve. On an elliptic curve, there are finitely many points. As these points are bits, addition is controlled by using XOR operation.

An Example of an Elliptic Curve Group over F_2n

The field F_24, defined by $f(x) = x^4 + x + 1$.

The element $g = (0010)$ is a primitive root for the field.

The powers of g are:

$g^0 = (0001)$ $g^1 = (0010)$ $g^2 = (0100)$ $g^3 = (1000)$ $g^4 = (0011)$ $g^5 = (0110)$

$g^6 = (1100)$ $g^7 = (1011)$ $g^8 = (0101)$ $g^9 = (1010)$ $g^{10} = (0111)$ $g^{11} = (1110)$

$g^{12} = (1111)$ $g^{13} = (1101)$ $g^{14} = (1001)$ $g^{15} = (0001)$

The large value of n generates the more efficient table which provides more security. For adequate security, $n = 160$. The pattern allows the use of primitive root notation (g^e) rather than bit string notation, as used in the following example.

$$f(x) = x^4 + x + 1.$$

Suppose the elliptic curve $y^2 + xy = x^3 + g^4x^2 + 1$.

Here $a = g^4$ and $b = g^0 = 1$. The point (g^5, g^3) satisfies this equation over F_2n:

$$y^2 + xy = x^3 + g^4x^2 + 1$$

$$(g^3)^2 + g^5g^3 = (g^5)^3 + g^4g^{10} + 1$$

$$g^6 + g^8 = g^{15} + g^{14} + 1$$

$(1100) + (0101) = (0001) + (1001) + (0001)$

$(1001) = (1001)$

The fifteen points which satisfy this equation are:

$(1, g^{13})$ (g^3, g^{13}) (g^5, g^{11}) (g^6, g^{14}) (g^9, g^{13}) (g^{10}, g^8) (g^{12}, g^{12})

$(1, g^6)$ (g^3, g^8) (g^5, g^3) (g^6, g^8) (g^9, g^{10}) (g^{10}, g) $(g^{12}, 0)$ $(0, 1)$

8.4.7 Arithmetic in an Elliptic Curve Group over F_2m

There is finite number of points for an elliptic curve group over F_2n without round off error. Use of binary arithmetic makes the method very efficient.

the key size for elliptic curve increases slowly as shown in Table 8.3. Hence, elliptic curve systems offer more security per bit increase in key size than either RSA or Diffie–Hellman algorithms.

Not only security, ECC is more attractive due to its computational efficiency than other algorithms such as RSA and Diffie–Hellman. ECC uses arithmetic which takes more computational time per bit as compared to RSA and Diffie–Hellman algorithm. But the security provided per bit by ECC is more than the extra time required for computation. Table 8.4 shows the ratio of computation of Diffie–Hellman to elliptic curve for different key sizes (in bits) listed in Table 8.3.

Table 8.4 Relative computation costs of Diffie–Hellman and elliptic curves

Security level (bits)	Ratio of DH cost: EC cost
80	3:1
112	6:1
128	10:1
192	32:1
256	64:1

If we use large key size for transferring the keys, the channel overhead is increased. So, EEC provides better solution as compared to RSA and Diffie–Hellman algorithms. There are number of elliptic curves standardised by NIST. Out of these, ten curves are for binary fields and five are for prime fields.

8.7 ZERO-KNOWLEDGE PROOF

A disadvantage of the above encryption algorithms is that when user A gives his secret key to user B, user B can thereafter impersonate user A. But, zero-knowledge (ZK) protocols allow user A to demonstrate knowledge of a secret key to user B without revealing any useful information about that secret key. Zero-knowledge proofs are probabilistic and based on interactive method. The example of zero-knowledge proof is RSA algorithm in which the user can prove that he knows the secret associated with his public key without revealing his private key. A protocol between two users in which one user is called prover and the other user is called the verifier. Prover tries to prove a certain fact to the verifier. This protocol is called *zero-knowledge protocol* or *zero-knowledge proof*. In cryptography, it is used for authentication.

Properties of zero-knowledge proof:

1. *Completeness:* If the fact is true, the honest verifier always accepts this fact and both the users follow the protocol.
2. *Soundness:* If the fact is false, the honest verifier always rejects this fact except with some small probability.
3. *Zero-knowledge:* If the fact is true, no cheating verifier learns anything other than this fact. This is Ormalised by showing that every cheating verifier has

Hamiltonian cycle in G onto ... P does not know which ... will be asked of the cycle. During each round, P does not know which ... in order to be able to answer both, H must be by V until after giving H. Therefore, in order to be able to answer both, H must be isomorphic to G and P must have a Hamiltonian cycle in H. Because only someone who knows a Hamiltonian cycle in G would always be able to answer both questions, V becomes convinced that P does know this information. This takes after a sufficient number of rounds.

However, P's answers do not reveal the original Hamiltonian cycle in G. Each round, V will learn only H's isomorphism to G or a Hamiltonian cycle in H. He would need both answers for a single H to discover the cycle in G, so the information remains unknown as long as P can generate a unique H every round. Because of the nature of the isomorphic graph and Hamiltonian cycle problems, V gains no information about the Hamiltonian cycle in G from the information revealed in each round.

If P does not know the information, she can guess which question V will ask and generate either a graph isomorphic to G or a Hamiltonian cycle for an unrelated graph, but since she does not know a Hamiltonian cycle for G, she cannot do both. With this guesswork, her chance of fooling V is $n/2$ where n is the number of rounds. For all realistic purposes, it is in feasibly difficult to defeat a zero-knowledge proof with a reasonable number of rounds in this way.

Attacks

To break zero-knowledge protocol, following attacks are tried against it.

1. *Impersonation:* One entity pretends to be another entity
2. *Replay:* Capture the information from a single previous protocol and use on the same or different verifier
3. *Interleaving:* A selective combination of information from one or more previous protocol
4. *Reflection:* Sending information from an ongoing protocol execution back to the originator
5. *Forced delay:* An adversary that intercepts a message and relays it later
6. *Chosen-text:* When an adversary chooses specific challenges in an attempt to gain information about the secret

SOLVED PROBLEMS

8.1 Users A and B use the Diffie–Hellman key exchange technique. They agree with a common prime $n = 41$ and a primitive root $g = 13$.
(a) If user A has private key $X_A = 27$, what is A's public key Y_A?
(b) If user B has private key $X_B = 18$, what is B's public key Y_B?
(c) What is the shared secret key?

Solution

User A		User B	
Private key	Calculation	Private key	Calculation
$X_A = 27$	For $n = 41$ and $g = 13$ $Y_A = g^{X_A} \bmod n$ $= 13^{27} \bmod 41$ $= 15$ $k_S = (Y_B)^{X_A} \bmod n$ $= (8)^{27} \bmod 41$ $= 2$	$X_B = 18$	For $n = 41$ and $g = 13$ $Y_B = g^{X_A} \bmod n$ $= 13^{18} \bmod 41$ $= 8$ $k_S = (Y_A)^{X_A} \bmod n$ $= (15)^{18} \bmod 41$ $= 2$

Therefore

(a) A's public key $Y_A = 15$

(b) B's public key $Y_B = 8$

(c) The shared secret key $k_s = 2$

8.2 Users A and B use the Diffie–Hellman key exchange technique. They agree with a common prime $n = 67$ and a primitive root $g = 5$.

(a) If user A has private key $X_A = 10$, what is A's public key Y_A?

(b) If user B has private key $X_B = 24$, what is B's public key Y_B?

(c) What is the shared secret key?

Solution

User A		User B	
Private key	Calculation	Private key	Calculation
$X_A = 10$	For $n = 67$ and $g = 5$ $Y_A = g^{X_A} \bmod n$ $= 40$ $K = (Y_B)^{X_A} \bmod n$ $= (25)^{10} \bmod 67$ $= 59$	$X_B = 24$	For $n = 67$ and $g = 5$ $Y_B = g^{X_B} \bmod n$ $= 25$ $K = (Y_A)^{X_B} \bmod n$ $= (40)^{24} \bmod 67$ $= 59$

Therefore,

(a) A's public key $Y_A = 40$

(b) B's public key $Y_B = 25$

(c) The shared secret key = 59

8.3 We use the Diffie–Hellman key exchange with private keys X_A and X_B and public keys $Y_A = g^{X_A} \bmod n$ and $Y_B = g^{X_B} \bmod n$. We assume $n = 71$ and $g = 7$.

(a) Give two possible pairs (X_A, X_B) such that the common key $k = 1$.

(b) An attacker knows that the product $Y_A * Y_B = 7 \bmod n$.

Give two possible pairs (X_A, X_B) that satisfy the attacker's knowledge.

Solution

(a) $k = (Y_B{}^{X_A}) \bmod n$

$Y_B = g^{X_B} \bmod n$

Therefore, $k = (g^{X_B})^{X_A} \bmod n$

$1 = (7^{X_B})^{X_A} \bmod 71$ (i)

Using Fermat's Little theorem $g^n = 1 \bmod n$

$$g^{n-1} \bmod n = 1$$

$$7^{(71-1)} \bmod 71 = 1 \quad \text{(ii)}$$

From equation (i) and (ii)

$$(7^{X_B})^{X_A} \bmod 71 = 7^{(71-1)} \bmod 71$$

Therefore, $X_A Y_A = 70.$

Therefore, the possible values of X_A and Y_A are 10 and 7 or 14 and 5

(b) $Y_A * Y_B = 7 \bmod n$

$Y_A * Y_B = g^{X_A} \bmod n * g^{X_B} \bmod n = 7 \bmod 71$

$7^{X_A} * 7^{X_B} \bmod 71 = 7 \bmod 71$

$7^{(X_A + X_B)} \bmod 71 = 7 \bmod 71$

Using Fermat's Little theorem $g^n = g \bmod n$

$$X_A + X_B = 71$$

We know that $Y_A * Y_B = 78 \bmod 71 = 7 \bmod 71.$

Factorise 78, we get (2, 39), (3, 26) and (6, 13) are the possible values of Y_A and Y_B.

$$6 = 7^{X_A} \bmod 71 \text{ and } 13 = 7^{X_B} \bmod 71$$

Solving we get $X_A = 39$ and $X_B = 32$

$$3 = 7^{X_A} \bmod 71 \text{ and } 26 = 7^{X_B} \bmod 71$$

We get, $X_A = 26$ and $X_B = 45$

SUMMARY

For public key cryptography, two important issues are: the distribution of public keys and the use of public key encryption for distribution of secret keys. In this chapter, we discuss different approaches for public key distribution. These include: the public announcement, publicly available directory, public key authority, and public key certificates.

Diffie–Hellman key exchange algorithm is used by two parties to generate a shared secret key. In Diffie–Hellman algorithm, there is no need of transferring the shared secret key for encryption. But it is suffered by man-in-the-middle attack. The Diffie–Hellman algorithm by itself does not provide authentication of the users. Elliptic curve cryptography is more secure [illegible] performance of elliptic curve cryptography is [illegible]