UNIT V

5 Cryptographic Algorithm

Syllabus

Mathematical preliminaries : *Groups, Rings, Fields, Prime numbers, Symmetric key algorithms : Data Encryption Standards, Advanced Encryption Standard,* ***Public Key Encryption and Hash function :*** *RSA Digital signatures, Digital Certificates and Public Key Infrastructure: Private Key Management, Diffie Hellman key exchange, The PKIX Model*

Contents

5.1 Mathematical Preliminaries

5.1.1 Modular Arithmetic

- Much of modern number theory and many practical problems (including problems in cryptography), are concerned with *modular arithmetic*. In arithmetic modulo N, we are concerned with arithmetic on the integers, where we identify all numbers which differ by an exact multiple of N. That is,

  $x \equiv y \bmod N$ if $x = y + mN$ for some integer m.

- This identification divides all the integers into N equivalence classes. We usually denote these by their "simplest" members, that is, the numbers 0, 1, ... , N – 1.
- If a is an integer and n is a positive integer, define a mod n to be the remainder when a is divided by n. Then, $a = [a/n] \times n + (a \bmod n)$;
- Example : 11 mod 7 = 4; – 11 and 7 = 3.

Theorem : $\equiv$ **n** ,is an equivalence relation on the integers. An equivalence class consists of those integers which have the same remainder on division by n. The equivalence classes are also known as congruence classes modulo n. Rather than say the integers a and b are equivalent we say that they are congruent modulo n.

Definition :

The set of all integers congruent to a modulo n is called the residue class [a].

Example : Residue classes mod 3 :

$[0] = \{..., -6, -3, 0, 3, 6, ...\}$

$[1] = \{..., -5, -2, 1, 4, 7, ...\}$

$[2] = \{..., -4, -1, 2, 5, 8, ...\}$

- The modulo operator has the following properties :
  1. $a \equiv b \bmod n$ if $n \mid (a - b)$.
  2. $(a \bmod n) = (b \bmod n)$ implies $a \equiv b \bmod n$.
  3. $a \equiv b \bmod n$ implies $b \equiv a \bmod n$.
  4. $a \equiv b \bmod n$ and $b \equiv c \bmod n$ imply $a \equiv c \bmod n$.
- Properties of modular arithmetic operations :
  1. $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
  2. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
  3. $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

- Proof of property 1 :
  Define (a mod n) = r_a and (b mod n) = r_b. Then a = r_a + jn and b = r_b + kn for some integers j and k. Then,

$$\begin{aligned}(a + b) \bmod n &= (r_a + jn + r_b + kn) \bmod n \\ &= (r_a + r_b + (j + k)\, n) \bmod n \\ &= (r_a + r_b) \bmod n \\ &= [(a \bmod n) + (b \bmod n)] \bmod n\end{aligned}$$

- Examples for the above three properties
  11 mod 8 = 3; 15 mod 8 = 7
  [(11 mod 8) + (15 mod 8)] mod 8 = 10 mod 8 = 2
  (11 + 15) mod 8 = 26 mod 8 = 2
  [(11 mod 8) – (15 mod 8)] mod 8 = – 4 mod 8 = 4
  (11 – 15) mod 8 = – 4 mod 8 = 4
  [(11 mod 8) × (15 mod 8)] mod 8 = 21 mod 8 = 5
  (11 × 15) mod 8 = 165 mod 8 = 5
- Properties of modular arithmetic

Let, $Z_n = \{0, 1, 2, \ldots, (n - 1)\}$ be the **set of residues modulo n.**

Property		Expression
Commutative laws	1.	(w + x) mod n = (x + w) mod n
	2.	(w × x) mod n = (x × w) mod n
Associative laws	1.	[(w + x) + y] mod n = [w + (x + y)] mod n
	2.	[(w × x) × y] mod n = [w × (x × y)] mod n
Distributive law		[w × (x + y)] mod n = [(w × x) + (w × y)] mod n
Identities		(0 + w) mod n = w mod n
		(1 × w) mod n = w mod n
Additive inverse (– w)		For each $w \in Z_n$, there exists a z such that w + Z ≡ 0 mod n

- If (a + b) ≡ (a + c) mod n, then b ≡ c mod n (due to the existence of an additive inverse)

- If $(a \times b) \equiv (a \times c) \bmod n$, then $b \equiv c \bmod n$ (only if a is relatively prime to n; due to the possible absence of a multiplicative inverse).

e.g. $6 \times 3 = 18 \equiv 12 \bmod 8$ and

$6 \times 7 = 42 \equiv 2 \bmod 8$ but

$3 \neq 7 \bmod 8$ (6 is not relatively prime to 8)

- If n is prime then the property of multiplicative inverse holds (from a ring to a field).
- Following table provides modular addition and multiplication modulo 7.

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

(a) Addition modulo 7

*	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

(b) Multiplication modulo 7

w	– w	w^–1
0	0	---
1	6	1
2	5	4
3	4	5
4	3	2
5	2	3
6	1	6

(c) Additive and multiplicative inverses modulo 7

Table 5.1.1 Arithmetic modulo 7

5.1.2 Modular Exponentiation

- Modular exponentiation is a type of exponentiation performed over a modulus. Doing a modular exponentiation means calculating the remainder when dividing by a positive integer m (called the modulus) a positive integer b (called the base) raised to the e-th power (e is called the exponent).
- In other words, problems take the form where given base b, exponent e, and modulus m, one wishes to calculate c.
- Many public-key encryption algorithms use modular exponentiation - raising a number a (base) to some power b (exponent) mod p.
- $c = ab = a \cdot a \ldots a \bmod p$

Example 5.1.1 *To find* 11^{13} *mod 53*

Solution : $13 = 8 + 4 + 1$ so $11^{13} = 11^{8+4+1} = 11^8 * 11^4 * 11^1$

We can compute successive squares of 11 to obtain, $11, 11^2, 11^4, 11^8$ and then multiply together $11^1 * 11^4 * 11^8$ to get the answer 11^{13}.

Because we are working mod 53, we will "take mods" at every stage of the calculation.

Thus we have

$11 \bmod 53 = 11$

$11^2 = 121, \quad 121 \bmod 53 = 121 - 2*53 = 15$

$11^4 = (11^2)^2 = 15^2 \bmod 53 = 225 \bmod 53 = 225 - 4*53 = 13$

$11^8 = (11^4)^2 = 13^2 \bmod 53 = 169 \bmod 53 = 169 - 3*53 = 10$

Therefore $11^{13} \bmod 53 = 11 * 13 * 10 = 1430 \bmod 53 = 1430 = 26 * 53 + 52$

The answer is $11^{13} \bmod 53 = 52$.

5.1.3 Polynomial Arithmetic

- A polynomial is an expression of the form $a_n x^n + a_{n-1} x^{n-1} + \ldots\ldots + a_1 x + a_0$ for some non-negative integer n and where the coefficients $a_0, a_1, a_2, \ldots, a_n$ are drawn from some designated set S . The "S " is called the coefficient set.
- When $a_n \neq 0$, then it is called as polynomial of degree n. A zeroth-degree polynomial is called a constant polynomial. Polynomial arithmetic deals with the addition, subtraction, multiplication, and division of polynomials.
- If the value of $a_n = 1$ then the polynomial is said to be **monic**. If $n = 0$ then we simply have a constant known as a **constant polynomial**.
- A non-constant polynomial is irreducible, or prime, if it cannot be factorized as a product of polynomials of lower degree.

Addition of two polynomials	$f(x) = a_2x^2 + a_1x + a_0$ $g(x) = b_1x + b_0$ $f(x) + g(x) = a_2x^2 + (a_1 + b_1)x + (a_0 + b_0)$
Subtraction of two polynomials	$f(x) = a_2x^2 + a_1x + a_0$ $g(x) = b_3x^3 + b_0$ $f(x) - g(x) = -b_3x^3 + a_2x^2 + a_1x + (a_0 - b_0)$
Multiplication of two polynomials	$f(x) = a_2x^2 + a_1x + a_0$ $g(x) = b_1x + b_0$ $f(x) \times g(x) = a_2b_1x^3 + (a_2b_0 + a_1b_1)x^2 + (a_1b_0 + a_0b_1)x + a_0b_0$

5.1.4 Finite Fields

- A field is a set of elements on which two arithmetic operations i.e. addition and multiplication, have been defined and which has the properties of abstract algebra arithmetic, such as closure, associativity, commutativity, distributivity and having both additive and multiplicate inverses.
- A finite field is simply a field with a finite number of elements. It can be shown that the order of a finite field must be a power of a prime p^n, where n is a positive integer.

- Finite field of order p can be defined using arithmetic mod p.
- **Properties**

1. It can be shown that finite fields have order p^n, where p is a prime.
2. It can be shown that for each prime p and each positive integer 'n', there is, upto isomorphism, a unique finite field of order p^n.
3. Let $GF(p^n)$ represent a finite field of order p^n. GF stands for **Galois field.**

Construction of finite fields

- To construct $GF(p^n)$ first find an irreducible polynomial I of degree n, with coefficients in Z_p.
- Let $GF(p^n) = \{a_{n-1} X^{n-1} + a_{n-2}X^{n-2} + ... + a_1x + a_0 \mid a_i \varepsilon Z_p\}$
- (Note that here addition is done modulo Z_p while multiplication is done modulo I).
- Example GF (16) = $GF(2^4)$ we want polynomial of degree 4 with coefficients in $Z_2 = \{ax_3 + bx_2 + cx + d \mid a, b, c, d, \varepsilon Z_2\}$.
- Here addition is done as in Z_2 [x], while multiplication is done modulo $x^4 + x + 1$.

Properties of GF (p^n)

- It can be shown that for each positive integer n there exists an irreducible polynomial of degree n over GF(p) for any p.
- It can be shown that for each divisor m of n, $GF(p^n)$ has a unique sub field of order p^m. Moreover, these are the only subfields of $GF(p^n)$.

Primitive Element

- A nonzero element a ε GF(q) is called a **Primitive Element** if $h^1, h^2 ... h^{q-1}$, are precisely all the nonzero elements of GF(q) (i.e. the multiplicative order of a is (q – 1)).
  1. Generator of the multiplicative group of nonzero elements.
  2. Used to simplify multiplication.
- It can be shown that every $GF(p^n)$ contains a primitive element.

5.1.5 Groups

- A group G is a nonempty set together with a *binary operation* (*) such that the following three properties are satisfied :
  1. **Associativity :** (a*b)*c = a*(b*c). For all a, b, c ε G.

2. **Identity :** There is an element e ε G such that a*e = e*a. For all a ε G.
3. **Inverses :** For each element a ε G, there is an element b ε G such that a*b = b*a = e.

- Order of a Group G is the number of elements it contains (denoted |G|). Order of an element g ε G is the smallest positive integer n such that $g^n = e$ (denoted |g|). Here $g^n = g*g*...*g$ n (times). In a *finite* group, the order of each element of the group divides the order of the group.

Properties of Groups

- For all g ε G, $g^0 = e$.
- For all n, m ≥ 1, g ε G,
  1. $g^n = g^{n-1}*g$
  2. $g^n * g^m = g^{n+m}$
  3. $(g^n)^{-1} = g^{-n} = (g^{-1})^n$
  4. $(g^m)^n = g^{mn}$
- If G is a group and for all a, b, ε G we have a*b = b*a (commutativity) then G is called an **Abelian Group.**
- In an Abelian group G, for all a, b ε G, then $(a*b)^{-1} = b^{-1}*a^{-1} = a^{-1}*b^{-1}$

5.1.6 Ring with Unity

- A Ring R is a nonempty set with two binary operations, addition (denoted by a + b) and multiplication (denoted ab), such that for all a, b, c ε R :
  1. R is an abelian group under addition.
  2. a(bc) = (ab)c (associativity)
  3. a(b + c) = ab + ac and (b + c)a = bc + ca.
- A Unity in a ring is a nonzero element that is the identity under multiplication.
- A Commutative Ring R is ring such that for all a, b, c, ε R.
  1. a(b + c) = ab + ac = (b + c) a (commutativity)
- A Unit is a nonzero element of a Commutative Ring with Unity that has a multiplicative inverse.
- A Zero-Divisor is a nonzero element a ε R, R is a commutative ring, such that there is a nonzero element b ε R with ab = 0.

- An Integral Domain is a commutative Ring with unity and no zero-divisors.

Polynominal Rings

- A polynominal over a commutative ring R is an expression of the form

$$f(x) = a_n x^n + \ldots + a_1 x + a_0$$

where the coefficients a_i, $0 \le i \le n$, are elements of R and x is a variable with indeterminate meaning. The set of all such expressions is denoted by R{x}.

- The polynomial $0x^{m+n} + \ldots + 0x^{n+1} + a_n x^n + \ldots + a_1 x + a_0$ is regarded as the same polynomial as f(x). If $a_n \neq 0$, then n is called the degree of f(x), denoted by deg f(x). In this case $a_n = 1$ c (f(x)) is called the leading coefficient of f(x).

- Let $g(x) = b_m x^m + \ldots + b_1 x + b_0$ be a polynomial in R{x}. Addition of polynomials is defined by

$$f(x) + g(x) = b_m x^m + \ldots + b_{n+1} x^{n+1} + (a_n + b_n) x^n + \ldots + (a_1 + b_1) x + (a_0 + b_0)$$

where we assumed without loss of generality that $m \ge n$. The multiplication of polynomials is defined by

$$f(x)g(x) = c_{m+n} x^{m+n} + \ldots + c_2 x^2 + c_1 x + c_0, \text{ where } c_k = \sum_{i+j=k} a_i b_j.$$

5.1.7 Chinese Remainder Theorem

- Find a number x such that have remainders of 1 when divided by 3, 2 when divided by 5 and 3 when divided by 7. i.e.
  1. $x = 1 \bmod 3$
  2. $x = 2 \bmod 5$
  3. $x = 3 \bmod 7$
- Integers can be represented by their residues modulo a set of pair-wise relatively prime moduli. For example : In Z_{10}, integer 8 can be represented by the residues of the 2 relatively prime factors of 10 (2 and 5) as a tuple (0, 3).
- Let $M = m_1 \times m_2 \times m_3 \times \ldots \times m_k$, where m_i's are pairwise relatively prime, i.e. $\gcd(m_i, m_j) = 1, 1 \le i \ne j \le k$
- Assertion.
  1. $A \leftrightarrow (a_1, a_2, \ldots a_k)$ where $A \in Z_m$, $a_i \in Z_{mi}$ and $a_i = A \bmod m_i$ for $1 \le i \le k$

a) One to one correspondance (bijection) between Z_M and the Cartesian product $Z_{m1} \times Z_{m2} \times \ldots \times Z_{mk}$.

b) For every integer A such that $0 \le A < M$, there is a unique k-tuple $(a_1, a_2, \ldots a_k)$ with $0 \le a_i < m_i$.

c) For every such k-tuple $(a_1, a_2, \ldots a_k)$ there is a unque A in Z_M.

d) Transformation from A to $(a_1, a_2, \ldots a_k)$ is unique

e) Computing A from $(a_1, a_2, \ldots a_k)$ is done as follows :

1. Let $M_i = M/m_i$ for $1 \le i \le k$; i.e. $M_i = m_1 \times m_2 \times \ldots \times m_{i-1} \times \ldots \times m_k$
2. Note that $M_i \equiv 0 \pmod{m_j}$ for all $j \neq i$
3. Let $c_i = M_i \times (M_i^{-1} \bmod m_i)$ for $1 \le i \le k$
4. Then $A \equiv (a_1c_1 + a_2c_2 + \ldots + a_kc_k) \bmod M$
5. $\leftarrow a_i = A \bmod m_i$, since $c_j \equiv M_j \equiv 0 \pmod{m_i}$ if $j \neq i$ and $c_i \equiv 1 \pmod{m_i}$

- Operations performed on the elements of Z_M can be equivalently performed on the corresponding k-tuples by performing the operation independently in each co-ordinate position.

Example : $A \leftrightarrow (a_1, a_2, \ldots a_k)$ $B \leftrightarrow (b_1, b_2, \ldots b_k)$

$(A + B) \bmod M \leftrightarrow ((a_1 + b_1) \bmod m_1, \ldots (a_k + b_k) \bmod m_k)$

$(A - B) \bmod M \leftrightarrow ((a_1 - b_1) \bmod m_1, \ldots (a_k - b_k) \bmod m_k)$

$(A \times B) \bmod M \leftrightarrow ((a_1 \times b_1) \bmod m_1, \ldots (a_k \times b_k) \bmod m_k)$

- CRT provides a way to manipulate (potentially large) numbers mod M in terms of tuple of smaller numbers.

Chinese remainder theorem :

Suppose gcd (m, n) = 1. given a and b, there exists exactly one solution x (mod mn) to the simultaneous congruence under certain conditions.

$$x \equiv a \pmod{m}, \quad x \equiv b \pmod{n}.$$

Proof :

- There exist integers s, t such that ms + nt = 1. Then ms ≡ 1 (mod n) and nt ≡ 1 (mod m). Let x = bms + ant. Then x ≡ ant ≡ a (mod m) and x ≡ bms ≡ b (mod n), as desired.
- Suppose x_1 is another solution. Then $x \equiv x_1 \pmod{m}$ and $x \equiv x_1 \pmod{n}$, so $x\text{-}x_1$ is a multiple of both m and n.

Lemma :

Let m, n be integers with gcd (m, n) = 1. If an integer c is a multiple of both m and n, then c is a multiple of mn.

Proof :

Let c = mk = nl. Write ms + nt = 1 with integers s, t. Multiply by c to obtain c = cms + cnt = mnls + mnkt = mn (ls + kt).

- To finish the proof of the theorem, let $c = x - x_1$ in the lemma to find that $x - x_1$ is a multiple of mn. Therefore, $x \equiv x_1 \pmod{mn}$. This means that any two solutions x to the system of congruences are congruent mod mn, as claimed.

Solved Examples

Example 5.1.2 *Solve $x \equiv 3 \pmod 7$, $x \equiv 5 \pmod{15}$.*

Solution : $x \equiv 80 \pmod{105}$ **(Note :** $105 = 7 \cdot 15$). Since $80 \equiv 3 \pmod 7$ and $80 \equiv 5 \pmod{15}$, 80 is a solution. The theorem guarantees that such a solution exists, and says that it is uniquely determined mod the product mn, which is 105 in the present example.

How to solve :

- One way, which works with small numbers m and n, is to list the numbers congruent to b (mod n) until you find one that is congruent to a (mod m).
- For example, the numbers congruent to 5 (mod 15) are

  5, 20, 35, 50, 65, 80, 95, ...

  Mod 7, there are 5, 6, 0, 1, 2, 3, 4, ... since we want 3 (mod 7), we choose 80.
- For slightly larger numbers m and n, making a list would be inefficient. However, a similar idea works. The numbers congruent to b (mod n) are of the form $b + nk$ with k an integer, so we need to solve $b + nk \equiv a \pmod m$. This is the same as

  $$nk \equiv a - b \pmod m.$$

- Since gcd (m, n) = 1 by assumption, there is a multiplicative inverse i for n (mod m). Multiplication by 1 gives

  $$k \equiv (a - b)\, i \pmod m.$$

  Substituting back into $x = b + nk$, then reducing mod mn, gives the answer.

Example 5.1.3 *Solve $x \equiv 7 \pmod{12345}$, $x \equiv 3 \pmod{11111}$.*

Solution : First, we know from our calculations in section that the inverse of 11111 (mod 12345) is $i = 2471$.

Therefore $k \equiv 2471\ (7 - 3) \equiv 9884 \pmod{12345}$. This yields $x = 3 + 11111 \equiv 9884 \equiv 109821127 \pmod{(11111.12345)}$.

Example 5.1.4 *In a Chinese remainder theorem, let $n = 210$ and let $n_1 = 5$, $n_2 = 6$, $n_3 = 7$. Compute $f^{-1}(3, 5, 2)$, i.e. given $x_1 = 3$, $x_2 = 5$, $x_3 = 3$, compute x.*

Solution :

$$N_1 = n_2 \times n_3 = 42$$
$$N_2 = n_1 \times n_3 = 35$$
$$N_3 = n_1 \times n_2 = 30$$
$$v_1 \equiv (N_1)^{-1} \equiv 42^{-1} \equiv 2^{-1} \equiv 3 \pmod 5$$

$$v_2 \equiv (N_2)^{-1} \equiv 35^{-1} \equiv 5^{-1} \equiv 5 \pmod 6$$

$$v_3 \equiv (N_3)^{-1} \equiv 30^{-1} \equiv 2^{-1} \equiv 4 \pmod 7$$

$$x \equiv a_1 v_1 N_1 + a_2 v_2 N_2 + a_{133} N_3$$

$$\equiv 126 + 875 + 360$$

$$\equiv 1361$$

$$x \equiv 101 \pmod{210}$$

Example 5.1.5 *State Chinese Remainder theorem and find X for the given set of congruent equations using CRT.*

$X = 2 \pmod 3$

$X = 3 \pmod 5$

$X = 2 \pmod 7$.

Solution : The Chinese Remainder Theorem (CRT) tells us that since 3, 5 and 7 are co-prime in pairs then there is a unique solution modulo $3 \times 5 \times 7 = 105$.

$n_1 = 3, \quad n_2 = 5, \quad n_3 = 7$

$N = n_1 \times n_2 \times n_3 = 3 \times 5 \times 7 = 105.$

$c_1 = 2, \quad c_2 = 3, \quad c_3 = 2$

Now $N_1 = N/n_1$

$= 105/3$

$N_1 = 35$ and so $d_1 = 35^{-1} \pmod 3 = 2$,

$N_2 = N/n_2 = 105/5 = 21$ and so $d_2 = 21^{-1} \pmod 5 = 1$, and

$N_3 = N/n_3 = 105/7 = 15$ and so $d_3 = 15^{-1} \pmod 7 = 1$.

Hence $x = (2 \times 35 \times 2) + (3 \times 21 \times 1) + (2 \times 15 \times 1)$

$= 233$

$\equiv 233 \pmod{105} = 23$

The solution is x = 23. You can check that by noting that the relations

$23 = 7 \times 3 + 2 \equiv 2 \pmod 3$

$23 = 4 \times 5 + 3 \equiv 3 \pmod 5$

$23 = 3 \times 7 + 2 \equiv 2 \pmod 7$

are all satisfied for this value of x.

Example 5.1.6 *Determine the value of x using Chinese remainder theorem.*

$X = 1 \pmod 5$

$X = 6 \pmod 7$

$X = 8 \pmod{11}$

Solution : $X = 1(\text{mod } 5)\ X = 6(\text{mod } 7)\ X = 8(\text{mod } 11)$

$$M = 5 \cdot 7 \cdot 11 = 385$$

$$M_1 = 385/5 = 77$$

$$M_2 = 385/7 = 55$$

$$M_3 = 385/11 = 35$$

$$77x_1 = 1(\text{mod } 5)$$

$$55x_2 = 1(\text{mod } 7)$$

$$35x_3 = 1(\text{mod } 11)$$

$$x_1 = 2 \quad x_2 = 6 \quad x_3 = 3$$

$$x = (a_1M_1x_1 + a_2M_2x_2 + a_3M_3x_3)(\text{mod } n)$$

$$= (1 \cdot 77 \cdot 2 + 6 \cdot 55 \cdot 6 + 8 \cdot 35 \cdot 3)(\text{mod } 385)$$

$$x = 279$$

5.2 Simple DES

- Takes an 8-bit block plaintext, a 10-bit key and produces an 8-bit block of cipher-text.
- Decryption takes the 8-bit block of cipher-text, the same 10-bit key and produces the original 8-bit block of plaintext.
- It was designed as a test block cipher for learning about modern cryptanalytic techniques such as linear cryptanalysis, differential cryptanalysis and linear-differential cryptanalysis.
- The same key is used for encryption and decryption. Though, the schedule of addressing the key bits is altered so that the decryption is the reverse of encryption.
- An input block to be encrypted is subjected to an initial permutation IP. Then, it is applied to two rounds of key-dependent computation. Finally, it is applied to a permutation which is the inverse of the initial permutation.

plaintext = $b_1b_2b_3b_4b_5b_6b_7b_8$

key = $k_1k_2k_3k_4k_5k_6k_7k_8k_9k_{10}$

Subkey generation

- First, produce two subkeys K_1 and K_2:

$$K_1 = P8(LS_1(P10(key)))$$

$$K_2 = P8(LS_2(LS_1(P10(key))))$$

where P8, P10, LS_1 and LS_2 are bit substitution operators.

- For example, P10 takes 10 bits and returns the same 10 bits in a different order :

$$P10(k_1k_2k_3k_4k_5k_6k_7k_8k_9k_{10}) = k_3k_5k_2k_7k_4k_{10}k_1k_9k_8k_6$$

It's convenient to write such bit substitution operators in this notation :

P10 : (10 bits to 10 bits)

3	5	2	7	4	10	1	9	8	6

P8 : (10 bits to 8 bits)

6	3	7	4	8	5	10	9

LS_1 ("left shift 1 bit" on 5 bit words) : 10 bits to 10 bits

2	3	4	5	1	7	8	9	10	6

LS_2 ("left shift 2 bit" on 5 bit words) : 10 bits to 10 bits

3	4	5	1	2	8	9	10	6	7

Encryption

- The plain text is split into 8-bit blocks; each block is encrypted separately. Given a plaintext block, the cipher text is defined using the two subkeys K_1 and K_2, as follows:

$$Ciphertext = IP^{-1}(f_{K_2}(SW(f_{K_1}(IP(plaintext))))$$

where :

Initial Permutation (IP) : 8 bits to 8 bits

2	6	3	1	4	8	5	7

IP^{-1} (8 bits to 8 bits)

4	1	3	5	7	2	8	6

Switch (SW) : 8 bits to 8 bits

5	6	7	8	1	2	3	4

and $f_K( )$ is computed as follows.

We write exclusive-or (XOR) as +.

$$f_K(L, R) = (L + F_K(R), R)$$

$$F_K(R) = P4 (S0(lhs(EP(R)+K)) , S1(rhs(EP(R)+K)))$$

4 bits to 8 bits

4	1	2	3	2	3	4	1

P4 (4 bits to 4 bits)

2	4	3	1

lhs (8 bits to 4 bits)

1	2	3	4

rhs (8 bits to 4 bits)

5	6	7	8

$S0(b_1 b_2 b_3 b_4)$ = The $[b_1 b_4, b_2 b_3]$ cell from the "S-box" S0 below, and similarly for S1.

S0

	0	1	2	3
0	1	0	3	2
1	3	2	1	0
2	0	2	1	3
3	3	1	0	3

S1

	0	1	2	3
0	0	1	2	3
1	2	0	1	3
2	3	0	1	0
3	1	1	0	3

- **Algorithm :**

The block of 12 bits is written in the form L_0R_0, where L_0 consists of the first 6 bits and R_0 consists of the last 6 bits. The i^{th} round of the algorithm transforms an input L_{i-1} R_{i-1} to the output L_iR_i using an 8-bit K_i derived from K.

- Fig. 5.2.1 shows one round of a Feistel system.

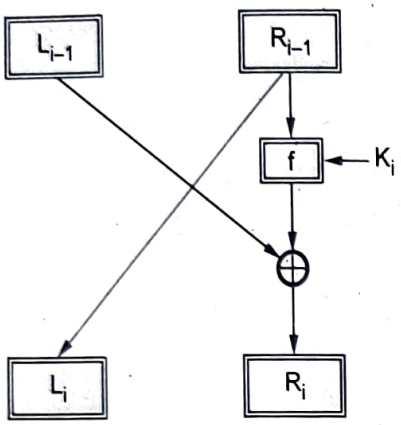


Fig. 5.2.1 One round of a Feistel system

- The output for the i^{th} round is found as follows :

$$L_i = R_{i-1} \text{ and } R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

- This operation is performed for a certain number of rounds, say n, and produces L_nR_n.
- The ciphertext will be R_nL_n.
- Encryption and decryption are done the same way except the keys are selected in the reverse order.
- The keys for encryption will be K_1, K_2. ..... K_n and for decryption will be K_n, ...K_{n-1} ..... K_1.
- **Function $f(R_{i-1}, K_i)$:** The function $f(R_{i-1}, K_i)$, depicted in the Fig. 5.2.2 below, is described in following steps.
  1. The 6-bits are expanded using the following expansion function. The expansion function takes 6-bit input and produces an 8-bit output. This output is the input for the two S-boxes.

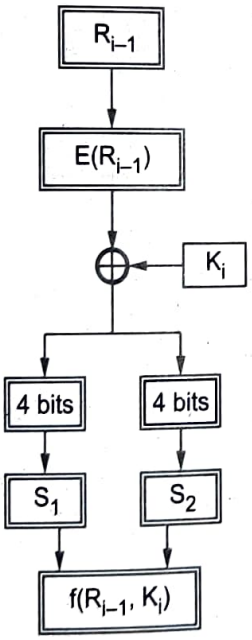


Fig. 5.2.2 The Function $f(R_{i-1}, K_i)$

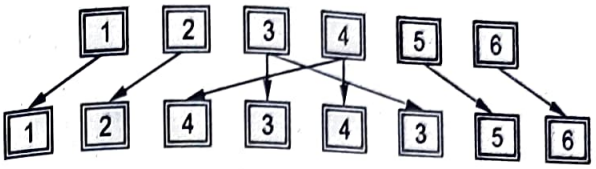


Fig. 5.2.3 The expansion function, $E(R_{i-1})$

2. The 8-bit output from the previous step is Exclusive-ORed with the key K_i
3. The 8-bit output is divided into two blocks. The first block consists of the first 4 bits and the last four bits make the second block. The first block is the input for the first S-box (S_1) and the second block is the input for the second S-box (S_2).
4. The S-boxes take 4-bits as input and produce 3-bits of output. The first bit of the input is used to select the row from the S-box, 0 for the first row and 1 for the second row. The last 3 bits are used to select the column.
5. The output from the S-boxes is combined to form a single block of 6-bits. These 6 bits will be the output of the function $f(R_{i-1}, K_i)$.

Example : Let the output from the expander function be 11010010.

Solution : 1101 will be the input for the S_1 box and 0010 will be the input for the S_2 box. The output from the S_1 box will be 111, the first of the input is 1 so select the second row and 101 will select the 6^{th} column. Similarly the output from the S_2 box will be 110. In above example we have the S_1 output 111 and S_2 output 110. So the output for the function

$f(R_{i-1}, K_i)$ will be 111110, the S_1 output followed by the S_2 output.

5.2.1 Data Encryption Standard (DES)

- DES Encryption standard (DES) is **a symmetric key block cipher** published by the National Institute of Standards and Technology (NIST).
- It encrypts data in 64-bit block.
- DES is symmetric key algorithm : The same algorithm and key is used for both encryption and decryption.
- Key size is 56-bit.
- The encryption process is made of two permutations i.e. P-boxes, which is called initial and final permutation.
- DES uses both transposition and substitution and for that reason is sometimes referred to as a **product cipher**. Its input, output and key are each 64-bits long. The sets of 64-bits are referred to as **blocks**.

- The cipher consists of 16 rounds or iterations. Each rounds uses a separate key of 48-bits.
- Fig. 5.2.4 shows DES encryption algorithm. First, the 64-bit plaintext passes through an Initial Permutation (IP) that rearranges the bits to produce the permuted input.

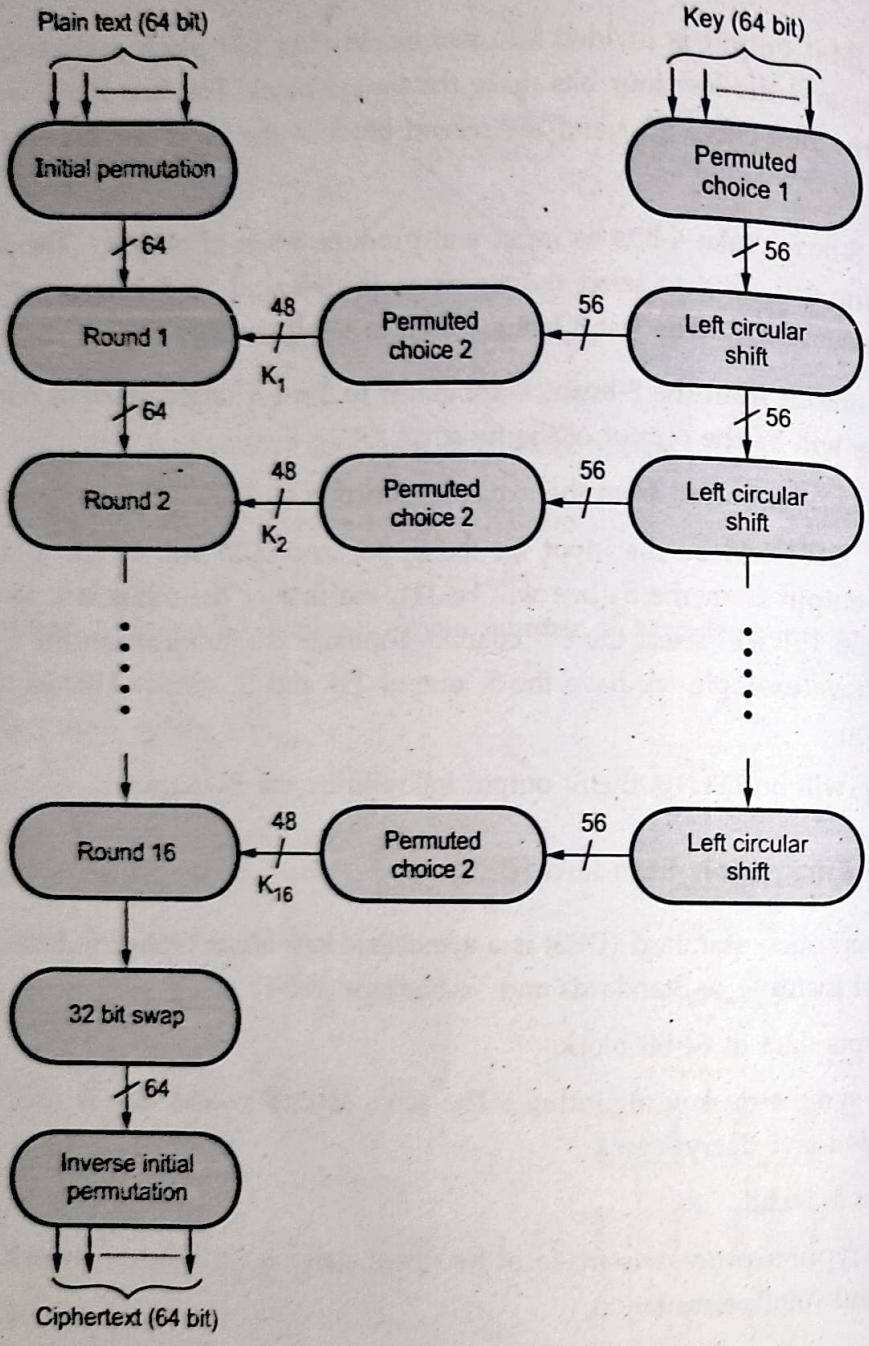


Fig. 5.2.4 DES encryption algorithm

- Then there is a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions.
- The output of the sixteenth round consists of 64-bits that are a function of the input plaintext and the key.
- The left and right halves of the output are swapped to produce the pre-output. At last, the pre-output is passed through a permutation (IP^{-1}) that is the inverse of the initial permutation function, to produce the 64-bit ciphertext.

Initial permutation

- Table shows the initial permutation and its inverse. The input to a table consist of 64-bits numbered from 1 to 64.
- The 64 entries in the permutation table contain a permutation of the numbers from 1 to 64. Each entry in the permutation table indicates the positon of a numbered input bit in the output, which also consists of 64-bits.

Initial Permutation (IP) table

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Inverse Initial Permutation (IP^{-1})

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

5.2.2 Single Round DES

- Fig. 5.2.5 shows single round of DES algorithm. The left and right halves of each 64-bit intermediate value are treated as separate 32-bit quantities, labeled L and R.

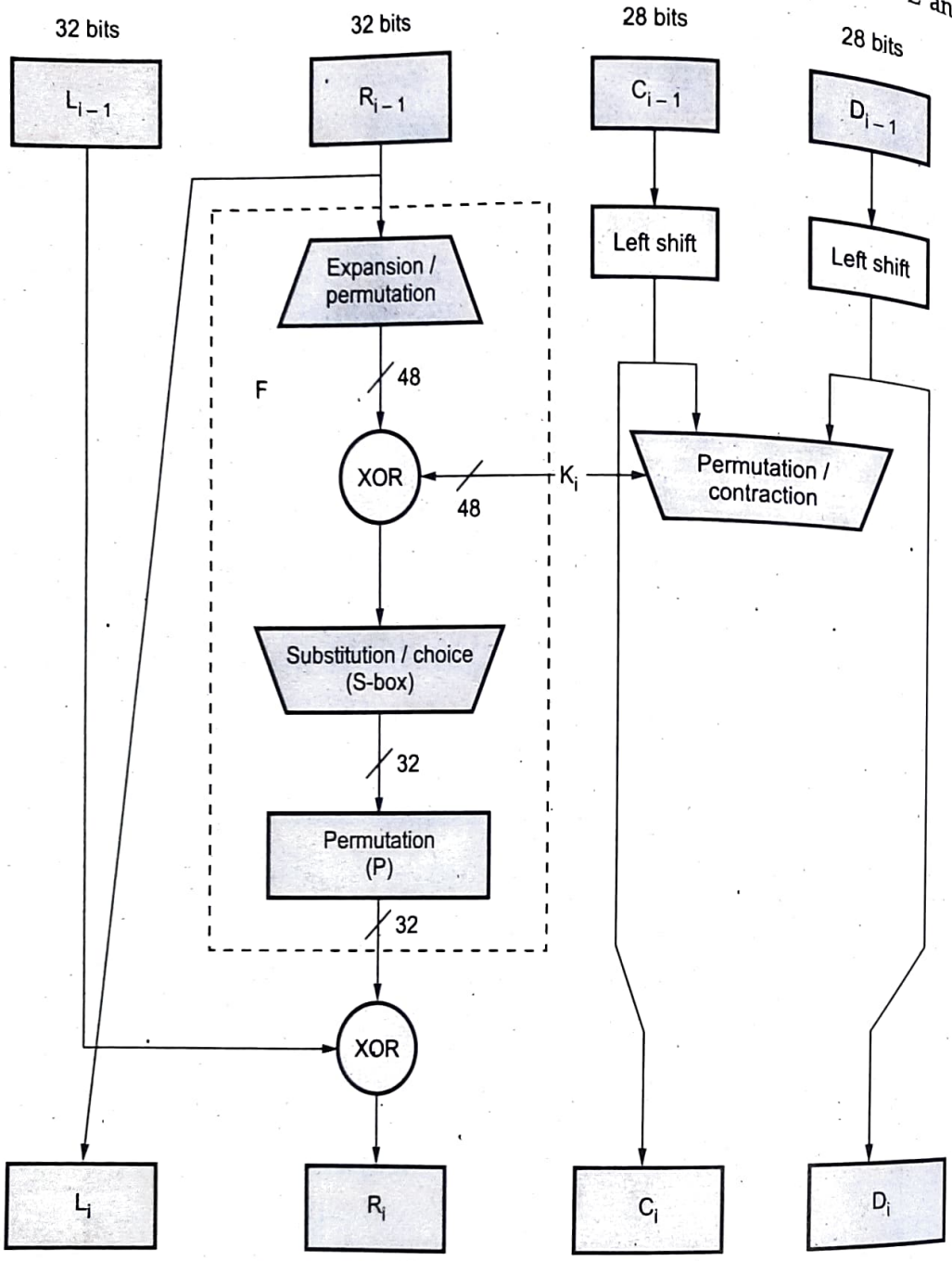


Fig. 5.2.5 Single round of DES algorithm

- The overall processing at each round can be summarised in the following formulae :

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \text{ X } F(R_{i-1}; K_i)$$

- The left output (L_i) is simply copy of the right input (R_{i-1}). The right output (R_i) is the XOR of left input (L_{i-1}) and right input (R_{i-1}) and key for this stage is K_i. In this stage, the substitution and permutation both functions are used.
- Fig. 5.2.6 shows role of S-boxes in the function F. It consists of set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output.

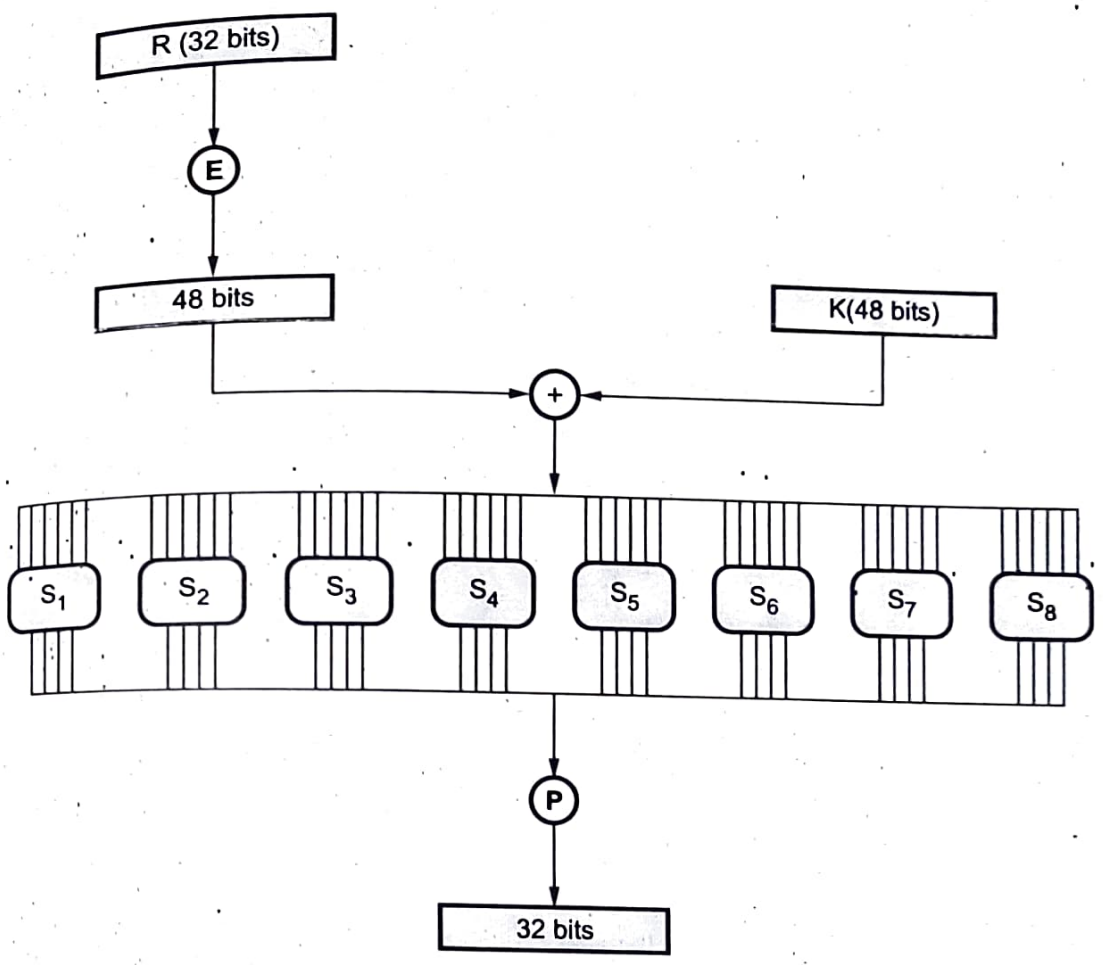


Fig. 5.2.6 S-boxes in the function (F)

- The 48 bit input block is divided into 8 subblocks and each subblock is given to a S-box. The S-box transforms the 6 bit input into a 4 bit output.
- First and last bits of the input to box S_i form a 2-bit binary number to select one of four substitutions defined by the four rows in the table for S_i. Two bits can store any decimal number between 0 and 3. This specifies the row number. The middle four bits select one of the sixteen columns.

- Following table gives the S-box value for DES

S_1	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

- Fig. 5.2.7 shows the selection of an entry in a S-box based n the 6-bit input. For example, in S_2, for input 101101, the row is 11 and the column is 0110. The value in row 3, column 6 which select row 3 and column 6 of S_2 box. The output is 4.

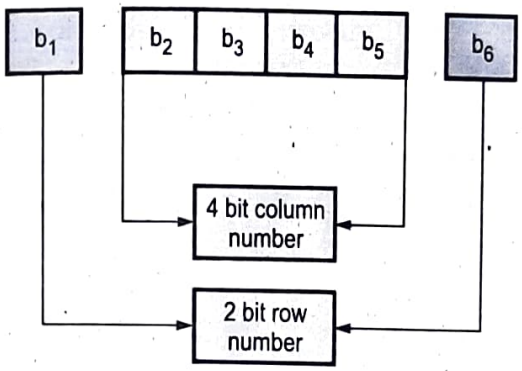


Fig. 5.2.7 Selecting entry in S-box

5.2.3 Key Generation

- 64-bit key is used as input to the algorithm. The initial 64-bit key is transformed into a 56-bit key by discarding every 8^{th} bit of the initial key.
- From 56-bit key, a different 48-bit subkey is generated during each round using a process called as key transformation.
- The resulting 56-bit key is then treated as two 28-bit quantities, labeled C_0 and D_0. At each round, C_{i-1} and D_{i-1} are separately subjected to a circular left shift, or rotation, of 1 or 2-bits.
- These shifted values serve as input to the next round. They also serve as input to Permuted choice Two, which produces a 48-bit output that serves as input to the function $F(R_{i-1}, K_i)$.

5.2.4 DES Encryption

- A block to be enciphered is subjected to an initial permutation IP, then to a complex key-dependent computation and finally to a permutation which is inverse of the initial permutation IP.
- The key-dependent computation can be simply defined in terms of a function *f*, called the cipher function, and a function KS, called the key schedule.
- Given two blocks L and R of bits, LR denotes the block consisting of the bits of L followed by the bits of R.

1. **Initial permutation** : The 64-bits of the input block to be enciphered are first subjected to the permutation, called the initial permutation.
2. **Key dependent computation** : The computation which uses the permuted input block as its input to produce the pre-output block consists. Cipher function *f* which operates on two blocks, one of 32-bits and one of 48-bits, and produces a block of 32-bits. Let the 64 bits of the input block in an iteration consist of a 32-bit block L followed by a 32-bit block R. Using the notation defined in the introduction the input block is then LR. Let K be a block of 48 bits chosen from the 64-bit key. Then the output L' R' of an iteration with input LR is defined by :

$$\left.\begin{aligned} L' &= R \\ R' &= L \,(+)\, f(R,K) \end{aligned}\right\} \quad \ldots (5.2.1)$$

where (+) denotes bit-by-bit addition modulo 2.

As before, let the permuted input block be LR. Finally, let L_0 and R_0 be respectively L and R and let L_n and R_n be respectively L' and R' of equation (5.2.1) hence L and R are respectively L_{n-1} and R_{n-1} and K is K_n i.e. when *n* is in the range from 1 to 16,

$$\text{Then } L_n = R_{n-1}$$

$$R_n = L_{n-1} \,(+)\, f(R_{n-1}, K_n)T$$

The pre-output block is then $R_{16}L_{16}$.

3. **Key schedule** : Key generation techniques is shown in the Fig. 5.2.8.
(See Fig. 5.2.8 on next page).

The input of the first iteration of the calculation is the permuted input block. If L' R' is the output of the 16^{th} iteration then R'L' is the pre-output block. At each iteration a different block K of key bits is chosen from the 64-bit key designated by KEY. Let KS be a function which takes a integer *n* in the range from 1 to 16 and a 64-bit block KEY as input and yields as output a 48-bit block K_n which is a permuted selection of bits from KEY i.e.

- $K_n = KS(n, KEY)$

with K_n determined by the bits in 48 distinct bit positions of KEY. KS is called the key schedule.

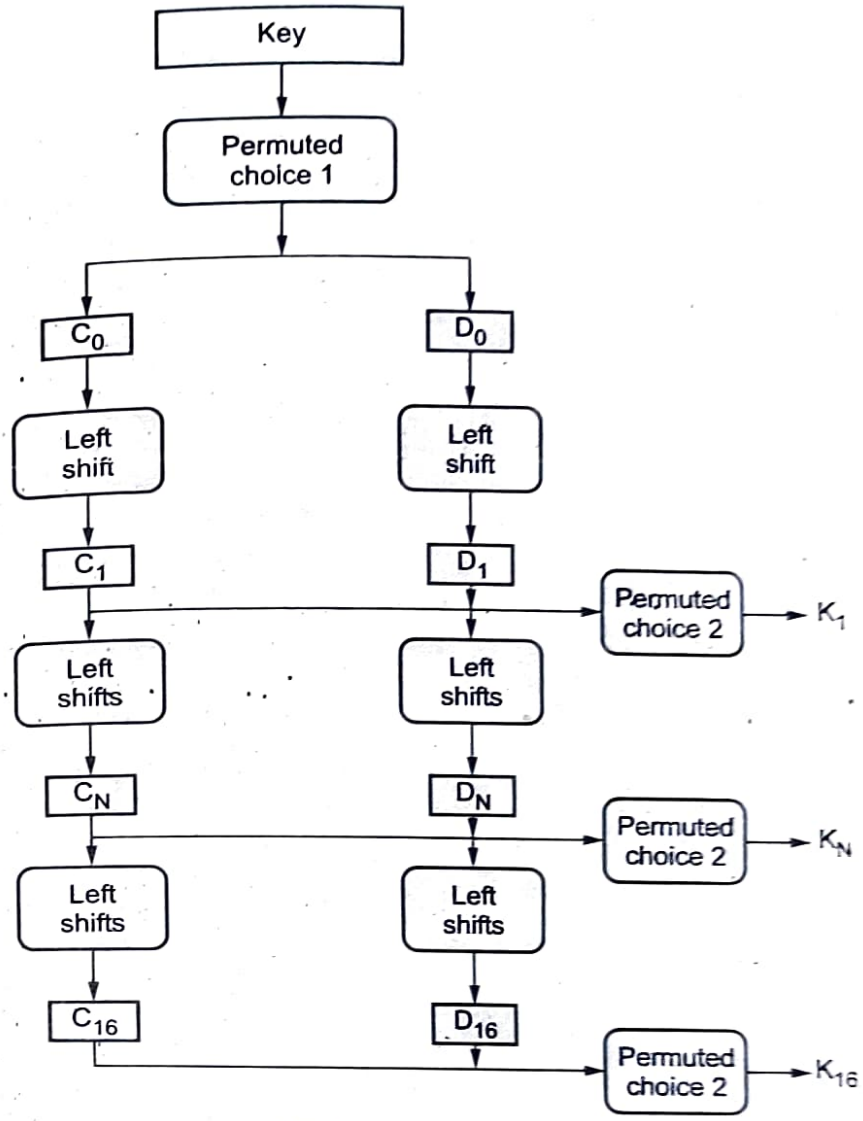


Fig. 5.2.8 Key generation techniques

5.2.5 DES Decryption

- The permutation IP^{-1} applied to the pre-output block is the inverse of the initial permutation IP applied to the input. Consequently, to decipher it is only necessary to apply the very same algorithm to an enciphered message block, taking care that at each iteration of the computation the same block of key bits K is used during decipherment as was used during the encipherment of the block only in a reverse order.
- For the decipherment calculation with $R_{10}L_{10}$ as the permuted input, K_{10} is used in the first iteration, K_{10} in the second, and so on, with K_1 used in the 16^{th} iteration.

5.2.6 DES Weak Keys

- With many block ciphers there are some keys that should be avoided, because of reduced cipher complexity.
- These keys are such that the same sub-key is generated in more than one round, and they include :
  1. **Weak keys :** The same sub-key is generated for every round and DES has 4 weak keys.
  2. **Semi-weak keys :** Only two sub-keys are generated on alternate rounds and DES has 12 of these (in 6 pairs).
  3. **Demi-semi weak keys :** Have four sub-keys generated.
- None of these cause a problem since they are a tiny fraction of all available keys however they MUST be avoided by any key generation program.

5.2.7 Avalanche Effect in DES

- Even a slight change in an input string should cause the hash value to change drastically. Even if 1 bit is flipped in the input string, at least half of the bits in the hash value will flip as a result. This is called an avalanche effect.
- A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext.
- In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext.
- This is referred to as the avalanche effect. If the change were small, this might provide a way to reduce the size of the plaintext or key space to be searched.

5.2.8 Advantages of DES

1. As 56-bit keys are used there are 70 quadrillion possible key values and hence a specific key cannot be identified easily.
2. As the length of the key is increased the security provided by the algorithm also increases.
3. The security of the DES algorithm resides in the key.

5.2.9 Disadvantages of DES

1. As it is a symmetric algorithm both sender and receiver must have same key, there is a possibility that the key is intercepted.
2. The design of S boxes makes it susceptible to linear cryptanalysis attack.

3. It is susceptible to differential cryptanalysis attack and brute force attack taking advantage of which DES crackers have been designed.
4. It has certain weak keys which generate the same key for all cycles of the algorithm like when all key bits are either 0s or 1s or if one half of the key bits are 0s or 1s. They are 0000000 0000000, 0000000 ffffffff, ffffffff 0000000, ffffffff ffffffff.
5. Some initial keys produce only two subkeys while some produce only four. They are called possible weak keys.

Possible techniques for improving DES

- Multiple enciphering with DES
- Extending DES to 128-bit data paths and 112-bit keys
- Extending the key expansion calculation.

5.2.10 S-Box Design Criteria

The criteria for the **S-boxes** are as follows :

1. No output bit of any S-box should be too close a linear function of the input bits.
2. Each row of an S-box should include all 16 possible output bit combinations.
3. If two inputs to an S-box differ in exactly one bit, the outputs must differ in at least two bits.
4. If two inputs to an S-box differ in the two middle bits exactly, the outputs must differ in at least two bits.
5. If two inputs to an S-box differ in their first two bits and are identical in their last two bits, the two outputs must not be the same.
6. For any non zero 6-bit difference betwen inputs, no more than 8 of the 32 pairs of inputs exhibiting that difference may result in the same output difference.

Criteria for **permutation P** are as follows.

1. The four output bits from each S-box at round i are distributed so that two of them affect middle bits of round (i + 1) and the other two affect end bits.
2. The four output bits from each S-box affect six different S-boxes on the next round, and no two affect the same S-box.
3. For two S-boxes j, k, if an output bit from S_j affects a middlle bits of S_{tock} on the next round, then an output bit from S_k cannot affect a middle bit of S_j.

5.2.11 Double DES

- Using two encryption stages and two keys.
  A) The plain text to ciphertext is as follows,

  $C = E_{K_2}(E_{K_1}(P))$ where k_1 and k_2 are the key.

  B) Ciphertext to plain text is as follows,

  $P = D_{K_1}(D_{K_2}(C))$
- Double DES suffers from Meet-in-the-Middle Attack.
- Meet-in-the-Middle Attack is as follows,
  1. Assume $C = E_{K_2}(E_{K_1}(P))$
  2. Given the plaintext P and ciphertext C
  3. Encrypt P using all possible keys K_1
  4. Decrypt C using all possible keys K_2

Fig. 5.2.9 shows the meet-in-the-middle attack for double DES.

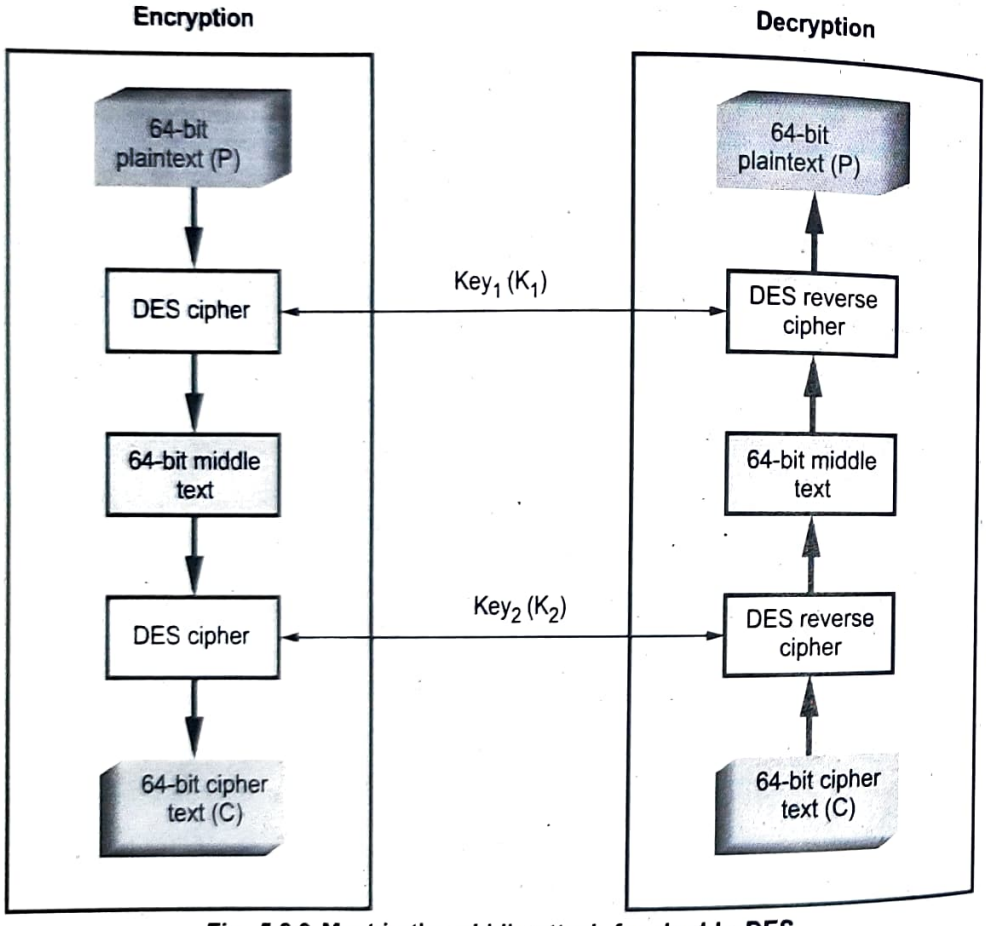


Fig. 5.2.9 Meet-in-the-middle attack for double DES

5.2.12 Triple DES

- Triple DES is simply another mode of DES operation. It takes three 64-bit keys, for an overall key length of 192 bits.
- The procedure for encryption is exactly the same as regular DES, but it is repeated three times. Hence the name triple DES.
- Triple DES uses 2 or 3 keys.
- The data is encrypted with the first key (K_1), decrypted with the second key (K_2) and finally encrypted again with the third key (K_3).
- Triple DES with three keys is used quite extensively in many products including PGP and S/MIME.
- Brute force search impossible on Triple DES.
- Meet-in-middle attacks need 256 Plaintext-Ciphertext pairs per key.
- Cipher text is produced as $C = E_{K_3}[D_{K_2}[E_{K_1}[P]]]$.
- Fig. 5.2.10 shows the 3DES method with three key.
- Triple DES runs three times slower than standard DES, but is much more secure if used properly.
- The procedure for decrypting something is the same as the procedure for encryption, except it is executed in reverse.
- Like DES, data is encrypted and decrypted in 64-bit chunks.

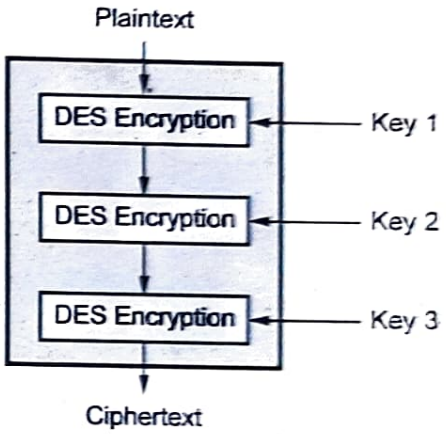


Fig. 5.2.10 3DES with three key method

- There are some weak keys that one should be aware of : If all three keys, the first and second keys, or the second and third keys are the same, then the encryption procedure is essentially the same as standard DES. This situation is to be avoided because it is the same as using a really slow version of regular DES.
- The input key for DES is 64-bits long; the actual key used by DES is only 56-bits in length.
- The least significant (right-most) bit in each byte is a parity bit, and should be set so that there are always an odd number of 1s in every byte.
- These parity bits are ignored, so only the seven most significant bits of each byte are used, resulting in a key length of 56-bits.

- This means that the effective key strength for Triple DES is actually 168-bits because each of the three keys contains 8 parity bits that are not used during the encryption process.

5.2.13 Triple DES with Two Keys

- In triple DES with two keys there are only two keys K_1 used by first and third stage and K_2 used in second stage.
- First the plain text is encrypted with key K_1 then the output of step one is decrypted with K_2 and final the output second step is encrypted again with key K_1.
- The function follows an encrypt-decrypt-encrypt (EDE) sequence :

$$C = E(K_1, D(K_2, E(K_1, P)))$$

$$P = D(K_1, E(K_2, D(K_1, C)))$$

- Fig. 5.2.11 shows 3DES with two keys.

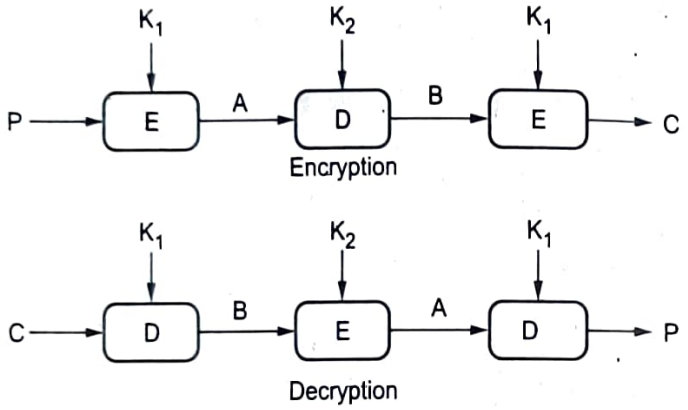


Fig. 5.2.11

- There is no cryptographic significance to the use of decryption for the second stage. Its only advantage is that it allows users of 3DES to decrypt data encrypted by users of the older single DES :

$$C = E(K_1, D(K_1, E(K_1, P))) = E(K_1, P)$$

$$P = D(K_1, E(K_1, D(K_1, C))) = D(K_1, C)$$

5.3 Block Cipher Design Principles

- A block cipher operates on blocks of data.
- Algorithm breaks the plaintext into blocks and operates on each block independently.
- Usually 2^n is the size of each block.
- Security of block ciphers depends on the design of the encryption function.
- Software implementations of block ciphers run faster than software implementation of the stream ciphers.
- Errors in transmitting one block generally do not affect other blocks.
- Each block is enciphered independently, using the same key, identical plaintext blocks produce identical ciphertext blocks.
- Suppose that plaintext is 227 bytes long and the cipher you are using operates on 16-byte blocks.
- Algorithm grabs the first 16-bytes of data, encrypts them using the key table.
- Algorithm produces 16-bytes of ciphertext.
- After first block, algorithm takes next block.
- The key table does not change from block to block.

Plaintext = 227 bytes

Block size = 16 bytes = $\frac{227}{16}$ = 14 blocks plus 3 bytes

- Algorithm encrypts 14 blocks and 3 bytes remain.
- For encrypting last 3 bytes data padding is used.
- Extra bytes are added to make the last block size to 16 bytes.
- Whoever decrypts the ciphertext must be able to recognize the padding.
- One problem with block ciphers is that if the same block of plaintext appears in two places, it encrypts to the same ciphertext.
- To avoid having these kinds of copies in the ciphertext, feedback modes are used.
- Cipher block chaining does not require the extra information to occupy bit spaces, so every bit in the block is part of the message.
- Before a plaintext block is enciphered, that block is XOR'ed with preceding ciphertext block.
- In addition to the key, this technique requires an initialization vector to XOR the initial plaintext block.

- For decrypting the data, copy a block of ciphertext, decrypt it and XOR the result with the preceding block of ciphertext.
- Taking E_K to be the encipherment algorithm with key K and I to be the initialization vector, the cipher block chaining technique is

$$C_o = E_K(m_o \oplus I)$$

$$C_i = E_K(m_i \oplus C_{i-1}) \quad \text{for } i > 0$$

5.3.1 Advantages and Disadvantage of Block Cipher

Advantages :

1. High diffusion
2. Immunity to insertation of symbols.

Disadvantages :

1. Slowness of encryption
2. Error propagation.

5.4 Stream Cipher

- Stream cipher algorithms are designed to accept a crypto key and a stream of plaintext to produce a stream of ciphertext.
- Fig. 5.4.1 shows the stream cipher.
- Stream cipher is similar to a one time pad.
- A stream cipher encrypts smaller block of data, typically bits or bytes.
- A key stream generator outputs a stream of bits $K_1, K_2, K_3 \ldots\ldots K_i$.

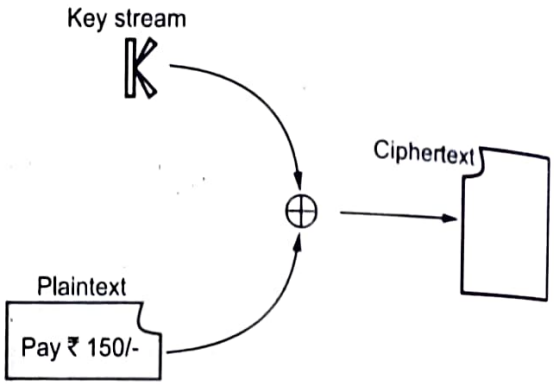


Fig. 5.4.1 Stream cipher

- This key stream is XORed with a stream of plaintext bits $P_1, P_2, P_3 \ldots\ldots P_i$ to produce the stream of ciphertext bits.

$$C_i = P_i \oplus K_i$$

- At the description end, the ciphertext bits are XORed with an identical key stream to recover the plaintext bits.

$$P_i = C_i \oplus K_i$$

- The system security depends entirely on the insides of the keystream generator.

5.4.1 Advantages and Disadvantages of Stream Cipher

Advantages :

1. Speed of transformation
2. Low error propagation.

Disadvantages :

1. Low diffusion
2. Susceptibility to malicious insertation and modifications.

5.4.2 Comparison between Stream and Block Cipher

Sr. No.	Stream cipher	Block cipher
1.	Stream ciphers operate on smaller units of plaintext.	Block ciphers operate on larger block of data.
2.	Faster than block cipher.	Slower than stream cipher.
3.	Stream cipher processes the input element continuously producing output one element at a time.	Block cipher processes the input one block of element at a time, producing an output block for each input block.
4.	Requires less code.	Requires more code.
5.	Only one time of key use.	Reuse of key is possible.
6.	Ex. - One time pad	Ex. - DES
7.	Application - SSL (secure connections on the web.)	Application - Database, file encryption.
8.	Stream cipher is more suitable for hardware implementation.	Easier to implement in software.

5.4.3 Confusion and Diffusion

Diffusion

- Diffusion is making output dependent on previous input (plain/cipher-text). Ideally, each output bit is influenced by every previous input bit.
- These are measures to thwart cryptanalysis based on statistical analysis. In diffusion, the statistical structure of the plaintext is dissipated into long range statistics of the cipher-text.

- This is achieved by having each plaintext letter affect the value of many cipher-text digits, which is equivalent to saying that each cipher-text digit is affected by many plaintext digits.
- The letter frequencies in the cipher-text will be more nearly equal than in the plaintext.

Confusion

- In Shannon's original definitions, confusion makes the relation between the key and the cipher-text as complex as possible. Confusion is making the output dependent on the key. Ideally, every key bit influences every output bit. Confusion tries to hide the connection between the cipher-text and the secret key.
- Confusion seeks to make the relationship between the statistics of the cipher-text and the value of the encryption key as complex as possible. This is achieved by the use of a complex substitution algorithm. These operations became the cornerstone of modern block cipher design.

5.4.4 Difference between Diffusion and Confusion

No.	Diffusion	Confusion
1.	Diffusion hides the relation between the ciphertext and the plaintext.	Confusion hides the relation between the ciphertext and key.
2.	If a single symbol in the plaintext is changed, several or all symbols in the ciphertext will also be changed.	If a single bit in the key is changed, most or all bits in the ciphertext will also be changed.
3.	In diffusion, the statistical structure of the plain text is dissipated into long-range statistics of the cipher text. This is achieved by permutation.	In confusion, the relationship between the statistics of the cipher text and the value of the encryption key is made complex. It is achieved by substitution.

5.5 Block Cipher Modes of Operation

Different types of cipher block modes are discussed here.

1. Electronic Code Book (ECB)

- A block of plaintext encrypts into a block of Ciphertext. Block size is 64-bits.
- Each block is encrypted independently.
- Plaintext patterns are not concealed since identical blocks of plaintext give identical blocks of ciphertext.
- It is not necessary to encrypt the file linearly.

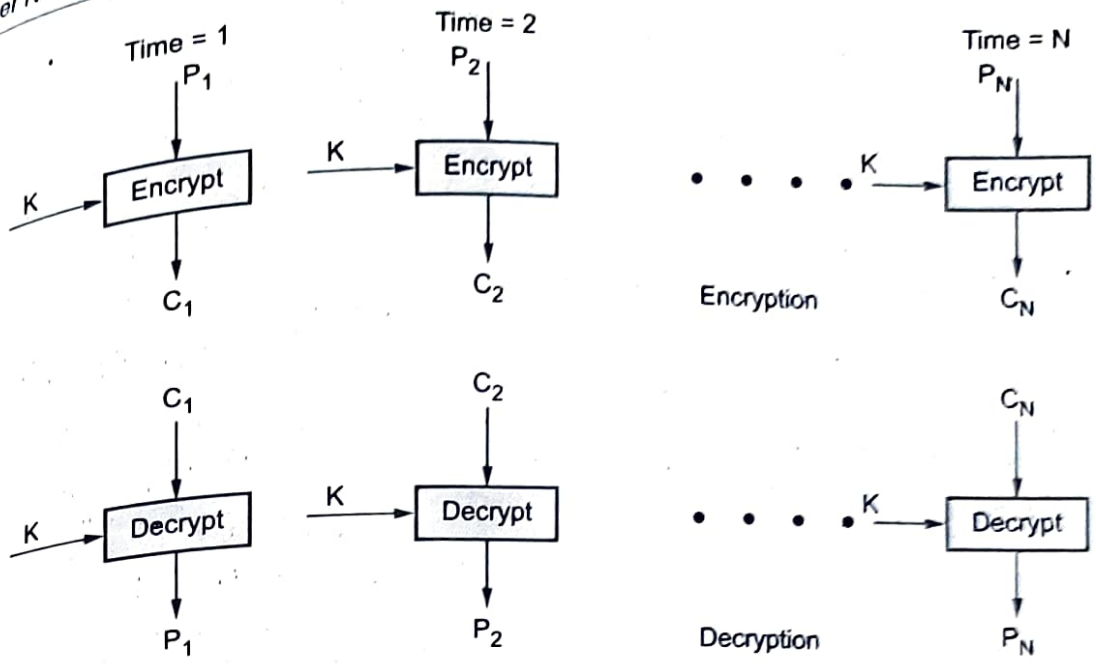


Fig. 5.5.1 ECB mode

- User can encrypt the 10 blocks in the middle first, then the blocks at the end, and finally the blocks in the beginning.
- Because of this, encrypted files are accessed randomly like a data base.
- It is very easy to parallelize the process.
- Pad the last block with some regular pattern i.e. zeros, ones to make it a complete block.
- End of file character is used to denote the final plaintext byte before padding.
- ECB method is ideal for a short amount of data, such as an encryption key.
- For lengthy messages, the ECB mode may not be secure.
- Used in secure transmission of single values i.e. an encryption key.
- ECB has security problems that limit its usability.
- Patterns in the plaintext can yield patterns in the ciphertext.
- It is also easy to modify a ciphertext message by adding, removing or switching encrypted blocks.
- Synchronization error is unrecoverable.

2. Cipher Block Chaining Mode (CBC)

- The plaintext is XORed with the previous ciphertext block before it is encrypted.
- The CBC mode is iterative mode.

- After a plaintext block is encrypted, the resulting ciphertext is also stored in a feedback register.
- Before the next plaintext block is encrypted, it is XORed with the feedback register to become the next input to the encrypting routine.
- The encryption of each block depends on all the previous blocks.
- A ciphertext block is decrypted normally and also saved in a feedback register.
- After the next block is decrypted, it is XORed with the results of the feedback register.
- Mathematically it is

$$C_i = E_k(P_i \oplus C_{i-1})$$

$$P_i = C_{i-1} \oplus D_k(C_i)$$

- It hides patterns in the plaintext.

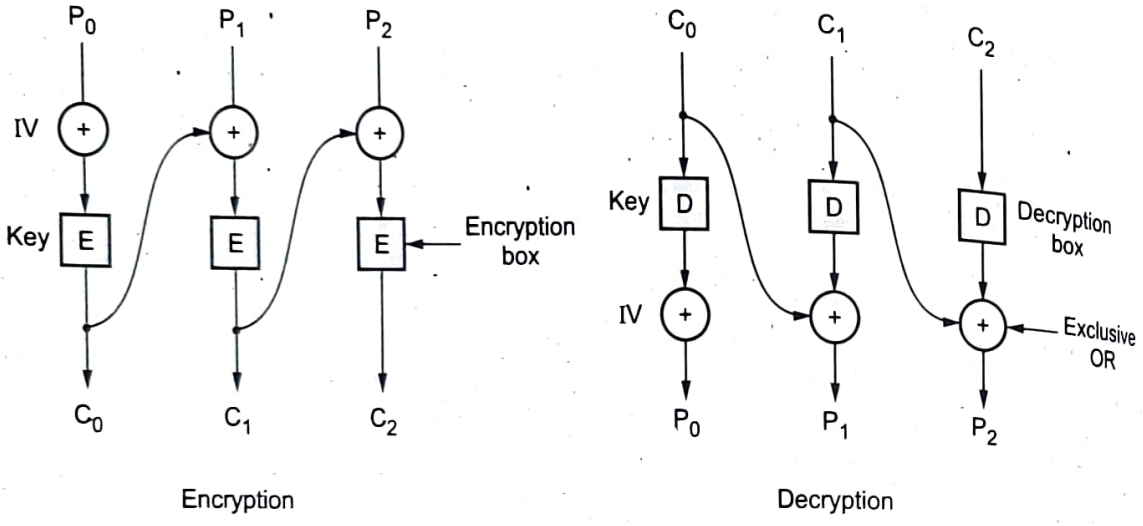


Fig. 5.5.2 CBC

- In order to guarantee that there is always some random looking ciphertext to apply to the actual plaintext, the process is started with a block of random bits called the Initialization Vector (IV).
- When used in networking messages, most CBC implementations add the IV to the beginning of the message in plaintext.
- A single bit error in a plaintext block will affect that ciphertext block and all subsequent ciphertext blocks.
- CBC mode is self recovering.
- Two blocks are affected by an error, but the system recovers and continues to work correctly for all subsequent blocks, Synchronization error is unrecoverable.

- Encryption is not parallelizable.
- Decryption is parallelizable and has a random access property.

3. Cipher Feedback Mode (CFB)

- Data is encrypted in units that are smaller than a defined block size.
- It is possible to convert the DES into stream cipher using cipher feedback mode.
- Fig. 5.5.3 shows encryption and decryption process.

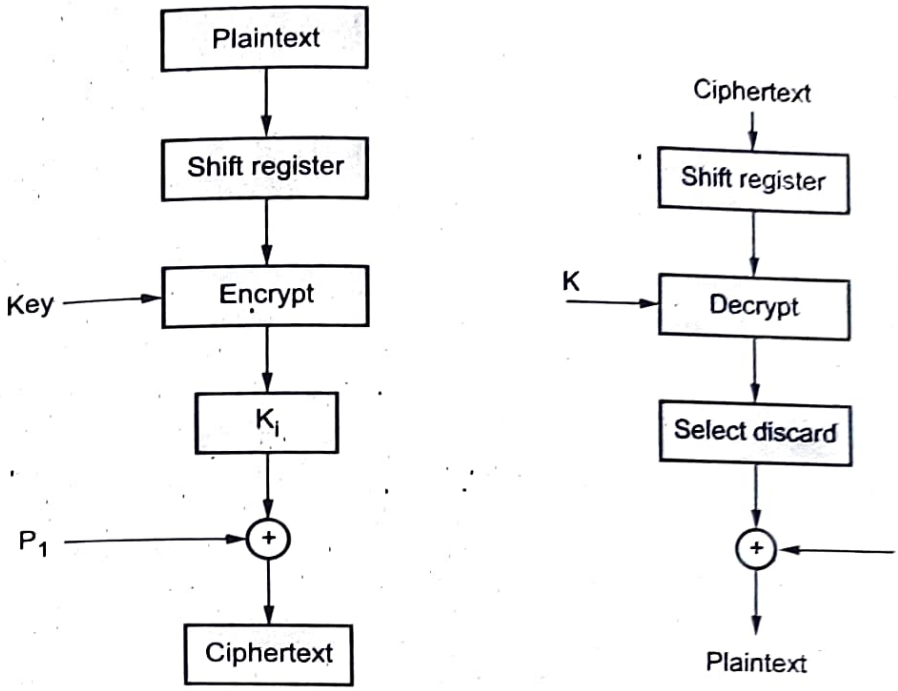


Fig. 5.5.3 CFB Modes

- More than one message can be encrypted with the same key, provided that a different initialization vector is used.
- CFB speed is the same as the block cipher.
- Encryption is not parallelizable, decryption is parallelizable and has a random access property.
- CFB is self recovering with respect to synchronization errors as well.

Advantages

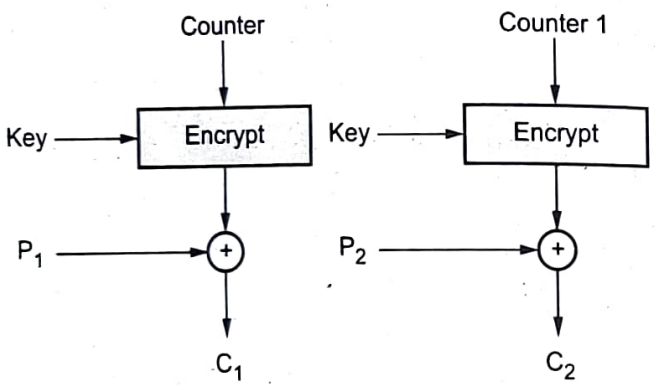
1. Simplicity
2. Need not be used on a byte boundary.
3. Input to the block cipher is randomized.
4. Ciphertext size is the same size as the plaintext size.

Disadvantages

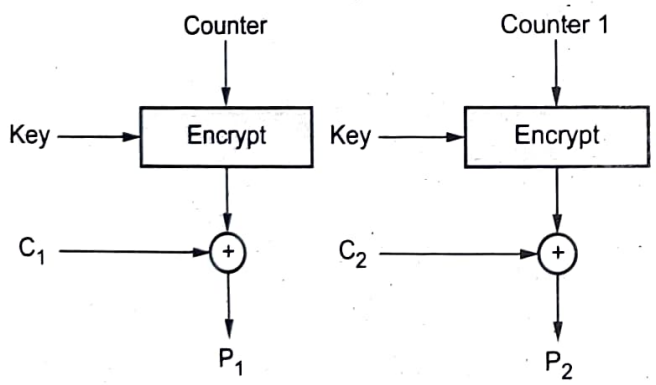
1. Encryption is not parallelizable.
2. Plaintext is somewhat difficult to manipulate.

4. Counter Mode

- Block ciphers in counter mode use sequence numbers as the input to the algorithm.
- More than one message can be encrypted with the same key, provided that a different initialise vector is used.
- Plaintext is very easy to manipulate, any change in ciphertext directly affects the plaintext.
- Synchronization error is unrecoverable.
- A ciphertext error affects only the corresponding bit of plaintext.
- **Encryption :** The counter is encrypted and then XORed with the plaintext block to produce the ciphertext block.
- Fig. 5.5.4 shows encryption and decryption.



(a) Encryption



(b) Decryption

Fig. 5.5.4 Counter mode

Advantages

1. Simple to implement.
2. It provides confidentiality.
3. Random access of block is possible.
4. Efficiency is same as block cipher.

5.6 Advanced Encryption Standards (AES)

- Advanced Encryption Standard (AES) is a symmetric key block cipher published by the NIST in December 2001.

5.6.1 Evaluation Criteria for AES

- NIST evaluation criteria for AES are
  1. Security 2. Cost 3. Algorithm and implementation characteristics.

1. Security

- This refers to the effort required to cryptanalyse an algorithm. Following parameters are also consider for evaluation.
  a. **Actual security** compared to other submitted algorithms.
  b. **Randomness :** The extent to which the algorithm output is indistinguishable from a random permutation on the input block.
  c. **Soundness** of the mathematical basis for the algorithm's security.
  d. Other security factors raised by the public during the evaluation process.

2. Cost

a. **Licensing requirements :** When the AES is issued, the algorithm specified in the AES shall be available on a worldwide, non-exclusive, royalty free basis.

b. **Computational efficiency :** The evaluation of computational efficiency will be applicable to both hardware and software implementations.

c. **Memory requirements :** The memory requirement for implementing the algorithm in hardware and software will be considered.

3. Algorithm and Implementation Characteristics

This category includes a variety of considerations, including flexibility, suitability for a variety of hardware and software implementations; and simplicity, which will make an analysis of security more straight forward.

The following criteria were used in the final evaluation :

1. **General security :** NIST relied on the public security analysis conducted by the cryptographic community.
2. **Software implementations :** It includes execution speed, performs across a variety of platforms and variation of speed with key size.
3. Restricted space environments.
4. Hardware implementations.
5. Attacks on implementations.
6. Encryption versus decryptions.
7. Key agility.
8. Other versatility and flexibility.
9. Potential for instruction level parallelism.

5.6.2 AES Cipher

- AES is a non-Feistel cipher that encrypts and decrypts a data block of 128-bits.
- The key size can be 128,192 or 256-bits. It depends on number of rounds.
- The number of rounds : 10 rounds for 128-bits,12 rounds for 192-bits, and 14 rounds for 256-bits.

Characteristics

1. Resistance against all known attacks.
2. Speed and code compactness on a wide range of platforms.
3. Design simplicity.

- For 128-bits AES, each round contains four steps :
  - i. Byte substitution
  - ii. Row shift
  - iii. Column mixing
  - iv. Round key addition
- The input to the encryption and decryption algorithms is a single 128-bit block. The block is represented as a row of matrix of 16 bytes.
- Fig. 5.6.1 shows the overall structure of AES.

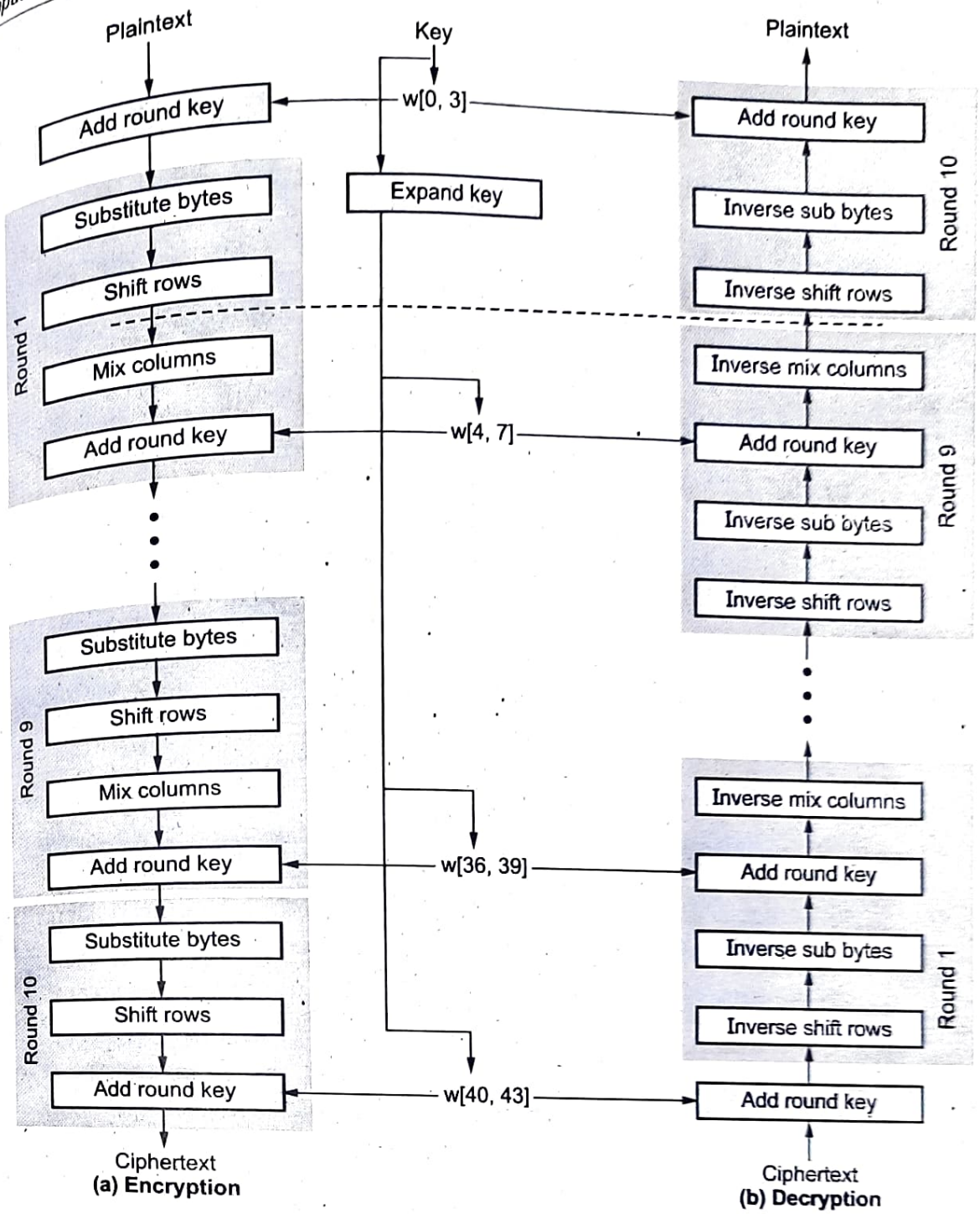


Fig. 5.6.1 AES encryption and decryption

- AES use several rounds in which each round is made of several stages. Data block is transformed from one stage to another.
- Data block is referred to as **state**. Block is copied into state array which is modified at each stage of encryption or decryption. After the final stage, state is copied to an output matrix.

Comments about the AES structure

1. AES structure is not a Feistel structure.
2. The key that is provided as input is expanded into an array of forty-four 32-bit words, w(i).
3. Four different stages are used, one of permutation and three of substitution.
4. For both encryption and decryption, the cipher begins with an AddRoundkey stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages.
5. Only the AddRoundkey stage make use of the key.
6. The AddRoundkey stage is, in effect, a form of Vernam Cipher and by itself would not be formidable.
7. Each stage is easily reversible.
8. The decryption algorithm makes use of the expanded key in reverse order.
9. Once it is established that all four stages are reversible, it is easy to verify that decryption does recover the plaintext.
10. The final round of both encryption and decryption consists of only three stages.

5.6.3 Applications of AES

1. AES can be used anywhere Symmetric Key cryptography is needed.
2. There is no particular list of applications of AES, but many banking systems use AES-128 and AES-256 to secure online banking or internet banking.

5.6.4 Comparison between AES and DES

Sr. No.	Parameters	AES	DES
1	Block size	128-bits	64-bits
2	Key length	128, 192, 256-bits	56-bits (effective length)
3	Encryption primitives	Substitution, shift, bit mixing	Substitution, Permutation
4	Cryptographic primitives	Confusion, Diffusion	Confusion, Diffusion
5	Design rationale	Closed	Open

Example 5.6.1 *For each of the following elements of DES, indicate the comparable element in AES or explain why it is not needed in AES.*

a. XOR of subkey material with the input to the function f

b. XOR of the f function output with left side of the block

c. The f function d. Permutation P e. Swapping of halves of the block

Solution :

a. **XOR of subkey material with the input to the function f :** The similar element in AES for XOR of subkey with the input to the function (that passes different stages before XORing) is the added round key stage in all the 10 rounds.

b. **XOR of the f function output with left side of the block :** There is no similar element in AES for XOR the f function output with left half side of the block, this is because AES structure is not a feistel structure. The entire block is processed in parallel (No two haives are using one half to modify the other half).

c. **The f function :** There is no single element that is similar to f function, but the four stages (Substitution bytes, shift rows, mix columns, added roundly) in each round do the same as f function.

d. **Permutation P :** The similar element for **P** is the shift rows in each of the 10 rounds.

e. **Swapping of halves of the block :** No similar element in AES this is because that AES structure not a feistel structure and no need to swap halves since work in parallel (No half needs to modify the other half).

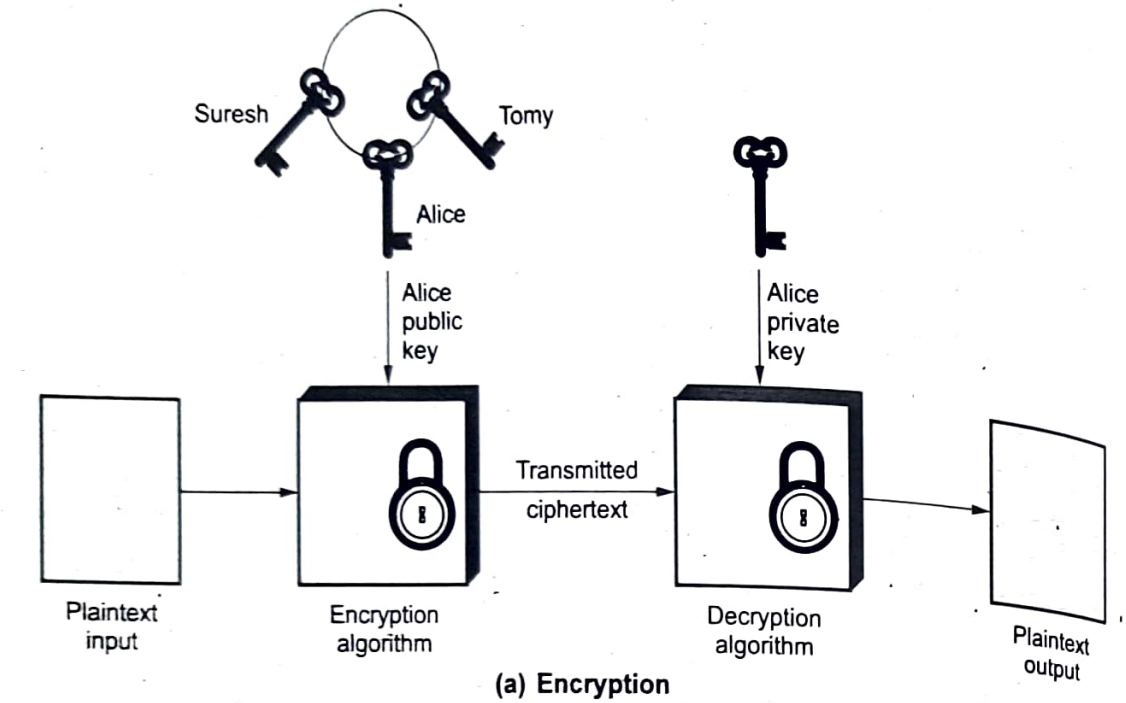
5.7 Public Key Encryption

- Public key cryptography, is a method of encrypting data with two different keys and making one of the keys, the public key, available for anyone to use. The other key is known as the private key.
- Data encrypted with the public key can only be decrypted with the private key, and data encrypted with the private key can only be decrypted with the public key. Public key encryption is also known as asymmetric encryption.
- In public key cryptography, there are two keys. Suppose Alice wishes to receive encrypted messages; she publishes one of the keys, the public key, and anyone, say Bob, can use it to encrypt a message and send it to her.
- When Alice gets the encrypted message, she uses the private key to decrypt it and read the original message. If Alice needs to reply to Bob, Bob will publish his own public key, and Alice can use it to encrypt her reply.
- These algorithms have the following important characteristic.

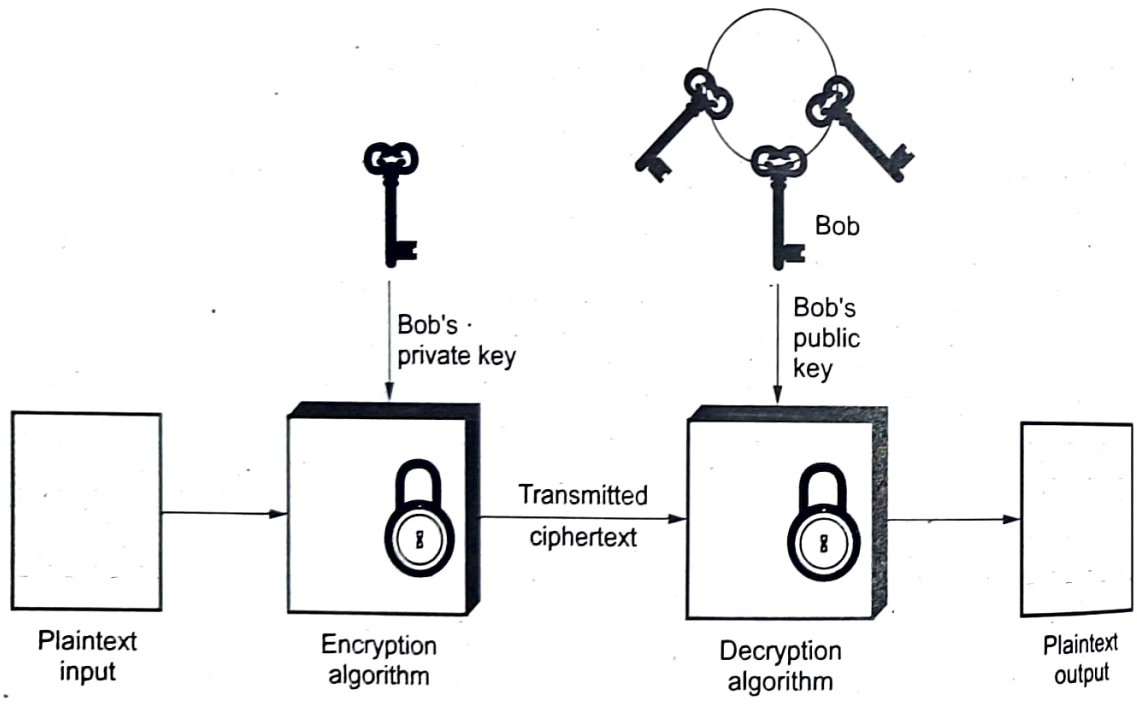
  1. It must be computationally easy to encipher or decipher a message given the appropriate key.

2. It must be computationally infeasible to derive the private key from the public key.
3. It must be computationally infeasible to determine the private key from a chosen plaintext attack.

- Fig. 5.7.1 shows public key cryptosystem.



(a) Encryption



(b) Authentication

Fig. 5.7.1 Public key cryptography

- Public key cryptographic algorithm has six elements as follow :

1. Plain text : This is a readable message which is given as input to the algorithm. In a public key algorithm, the plain text is encrypted in blocks.
2. Encryption algorithm : The encryption algorithm is implemented on the plain text which performs several transformations on plain text.
3. Public and private keys : These are the set of keys among which if one is used for encryption the other would be used for decryption. The transformation of plain text by encryption algorithm depends on the key chosen from the set to encrypt the plain text.
4. Cipher text : This is the output of encryption algorithm. The generated cipher text totally depends on the key selected from the set of the public and private key. Both of these keys, one at a time with plain text would produce different cipher texts.
5. Decryption algorithm : This would accept the output of the encryption algorithm i.e. the cipher text and will apply the related key to produce the original plain text.

- 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. This is the public key. The companion key is kept private.
3. If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.
4. Alice decrypts the message using her private key.

- The public key is accessed to all participants and private key is generated locally by each participant.

5.7.1 Requirement of Public Key Cryptography

1. It is computationally easy for a party B to generate a pair.
2. It is computationally easy for a sender A, to generate the corresponding ciphertext : $C = E(PU_b, M)$
3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message :

$$M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$$

4. It is computationally infeasible for an adversary, knowing the public key (PU_b) to determine the private key PR_b.
5. It is computationally infeasible for an adversary, knowing the public key (PU_b) and a ciphertext (C) to recover the original message (M).

5.7.2 Advantages and Disadvantages

- **Advantages of public key algorithm**
  1. Only the private key must be kept secret.
  2. The administration of keys on a network requires the presence of only a functional trusted TTP as opposed to an unconditionally trusted TTP.
  3. A private/public key pair remains unchanged for considerable long periods of time.
  4. There are many relatively efficient digital signature mechanisms as a result of asymmetric-key schemes.
  5. In a large network the number of keys necessary may be considerably smaller than in the symmetric-key scenario.
- **Disadvantages of public key algorithm**
  1. Slower throughput rates than the best known symmetric-key schemes.
  2. Large key size.
  3. No asymmetric-key scheme has been proven to be secure.
  4. Lack of extensive history.

5.7.3 Comparison between Public Key and Private Key

Public Key	Private Key
Public key encryption is also known as asymmetric key encryption.	Private key encryption is also known as symmetric key encryption.
One key for encryption and other key for decryption.	Same key is used for encryption and decryption.
Slower.	Very fast.
Key exchange is not a problem.	Key exchange is big problem.
Also called public key encryption.	Also called secret key encryption.
One of the two keys must be kept secret.	The key must be kept secret.
Public keys enable users to encrypt a message to other individuals on the system.	Private keys enable, user can decrypt a message secured by your public key.

5.8 RSA Algorithm

- RSA is a block cipher in which the plaintext and ciphertext are integers between 0 and $n - 1$ for some n.
- A typical size for n is 1024 bits.
- The RSA algorithm developed in 1977 by Rivest, Shamir, Adleman (RSA) at MIT.
- RSA algorithm is public key encryption type algorithm. In this algorithm, one user uses a public key and other user uses a secret (private key) key.
- In the RSA algorithm each station independently and randomly chooses two large primes p and q number, and multiplies them to produce $n = pq$ which is the modulus used in the arithmetic calculations of the algorithm.
- The details of the RSA algorithm are described as follows :
- **Key generation :**
  1) Pick two large prime numbers p and q, $p \neq q$;
  2) Calculate $n = p \times q$;
  3) Calculate $\phi(n) = (p - 1)(q - 1)$;
  4) Pick e, so that $\gcd(e, \phi(n)) = 1$, $1 < e < \phi(n)$;
  5) Calculate d, so that $d \cdot e \bmod \phi(n) = 1$, i.e. d is the multiplicative inverse of e in mod $\phi(n)$;
  6) Get public key as $K_U = \{e, n\}$;
  7) Get private key as $K_R = \{d, n\}$.
- **Encryption :**
  For plaintext block $P < n$, its ciphertext $C = P^e \bmod n$.
- **Decryption :**
  For ciphertext block C, its plaintext is $P = C^d \bmod n$.

Why RSA works :

- As we have seen from the RSA design, RSA algorithm uses modular exponentiation operation. For $n = p \cdot q$, e which is relatively prime to $\phi(n)$, has exponential inverse in mod n.
- Its exponential inverse d can be calculated as the multiplicative inverse of e in mod $\phi(n)$. The reason is illustrated as follows :
  Based on Euler's theorem, for y which satisfies $y \bmod \phi(n) = 1$, the following equation holds :

  $x^y \bmod n = x \bmod n$

AS d·e mod $\phi(n) = 1$, we have that $p^{ed} \equiv P \bmod n$. So the correctness of RSA cryptosystem is shown as follows :

- **Encryption :** $C = P^e \bmod n$;
- **Decryption :** $P = C^d \bmod n = (P^e)^d \bmod n = P^{ed} \bmod n = P \bmod n = P$.

Why RSA is secure :

- The premise behind RSA's security is the assumption that factoring a big number (n into p and q) is hard. And thus it is difficult to determine $\phi(n)$. Without the knowledge of $\phi(n)$ it would be hard to derive d based on the knowledge of e.

Advantages

1. RSA can be used both for encryption as well as for digital signatures.
2. Trapdoor in RSA is in knowing value of n but not knowing the primes that are factors of n.

Disadvantages

1. If any one of p, q, m, d is known, then the other values can be calculated. So secrecy is important.
2. To protect the encryption, the minimum number of bits in n should be 2048.

5.8.1 Attacks on RSA

Attacks on RSA algorithm are as follows :

1. **Brute force :** This involves trying all possible private keys.
2. **Mathematical attacks :** This involves the factoring the product of two primes.
3. **Timing attacks :** These depends on the running time of the description algorithm.
4. **Chosen ciphertext attacks :** This type of attack exploits properties of the RSA algorithm.

5.8.1.1 Computing $\phi(n)$

- Computing $\phi(n)$ is no easier that factoring n. For, if n and $\phi(n)$ are known, and n is the product of two primes p, q, then n can be easily factored, by solving the two equations.

$$n = pq \quad \ldots (5.8.1)$$

$$\phi(n) = (p - 1)(q - 1) \quad \ldots (5.8.2)$$

for the two unknowns p and q.

- If we substitute $q = n/p$ into the equation (5.8.2), we obtain a quadratic equation in the unknown value p :

$$p^2 - (n - \phi(n) + 1)\, p + n = 0 \quad \ldots (5.8.3)$$

- The two roots of equation (5.8.3) will be p and q, the factors of n. If a cryptanalyst can learn the value of $\phi(n)$ then he can factor 'n' and break the system.

5.8.1.2 Timing Attacks

- Kocher described a new attack on RSA in 1995.
- If the attacker Eve knows Alice's hardware in sufficient detail and is able to measure the decryption times for several known cipher-texts, she can deduce the decryption key (d) quickly. This attack can also be applied against the RSA signature scheme.
- In 2003, Boneh and Brumley demonstrated a more practical attack capable of recovering RSA factorizations over a network connection. This attack takes advantage of information leaked by the Chinese remainder theorem optimization used by many RSA implementations.
- One way to thwart these attacks is to ensure that the decryption operation takes a constant amount of time for every cipher-text. However, this approach can significantly reduce performance.
- There is simple counter-measures against timing attacks :
  1. **Constant exponentiation time :** Ensure that all exponentiations take the same time, but this will degrade performance.
  2. **Random delay :** Better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack.
  3. **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. RSA data security reports a 2 % to 10 % performance penalty for blinding.

5.8.1.3 Mathematical Attacks

- We can identify three approaches to attacking RSA mathematically :
  1. Factor n into two prime factors, this enables calculation of $\phi(n) = (p - 1)(q - 1)$, **which in turn, enables determination of** $d = e^{-1} \bmod \phi(n)$.
  2. Determine $\phi(n)$ directly, without first determining p and q.
  3. Determine d directly, without first determining $\phi(n)$

- Most discussions of cryptanalysis of RSA have focused on the task of factoring n into its two prime numbers. Determining $\phi(n)$ given n is equivalent to factoring n.
- With presently known algorithms, determining d given e and n appears to at least as time consuming as the factoring problem.

5.8.1.4 Adaptive Chosen Cipher-text Attacks

- In 1998, Daniel Bleichenbacher described the first practical adaptive chosen cipher-text attack, against RSA-encrypted messages using the PKCS#1 v1 padding scheme.
- Due to flaws with the PKCS#1 scheme, Bleichenbacher was able to mount a practical attack against RSA implementations of the Secure Socket Layer protocol and to recover session keys.
- As a result of this work, cryptographers now recommend the use of provably secure padding schemes such as Optimal Asymmetric Encryption padding and RSA laboratories has released new versions of PKCS#1 that are not vulnerable to these attacks.

Example 5.8.1 *For the given values p = 19, q = 23 and e = 3 find n, $\phi(n)$ and d using RSA algorithm.*

Solution :

$$n = p * q$$

$$n = 19 \times 23$$

$$\mathbf{n = 437}$$

$$\phi(n) = (p - 1) * (q - 1)$$

$$\phi(n) = 18 \times 22$$

$$\phi(n) = \mathbf{396}$$

$$e.d. = 1 \bmod \phi(n)$$

$$3d = 1 \bmod 396$$

$$d = \frac{1}{3}$$

Example 5.8.2 *Using the RSA algorithm, encrypt the following :*

i) p = 3, q = 11, e = 7, M = 12

ii) p = 7, q = 11, e = 17, M = 25

iii) Find the corresponding ds for i) and ii) and decrypt the ciphertext.

Solution : i) $n = p * q$

$$n = 3 * 11 = 33$$

$\phi(n) = (p - 1)(q - 1)$

$\phi(n) = 2 * 10 = 20$

$e \cdot d = 1 \bmod \phi(n)$

$7 \cdot d = 1 \bmod 20$

$d = 3$

Ciphertext (C) $= M^e \bmod n$

$= 12^7 \bmod 33$

$C = 12$

ii)

$n = p * q = 7 * 11 = 77$

$\phi(n) = (p - 1) * (q - 1) = 6 \times 10 = 60$

$e \cdot d = 1 \bmod \phi(n) \Rightarrow 17\ d = 1 \bmod 77$

$d = 3$

Ciphertext (C) $= M^e \bmod n$

$= 25^{17} \bmod 77 \Rightarrow 77 \Rightarrow c = 9$

$C = 12$

iii) **Decryption :**

$M = c^d \bmod n$

In case (i) $M = 12^3 \bmod 33 = 12$

In case (ii) $M = 9^{57} \bmod 77 = 25$

Example 5.8.3 *In RSA system the public key of a given user is e = 7 and n = 187*

i) What is the private key of this user ?

ii) If the intercepted ciphertext is c = 11 and sent to a user whose public key is e = 7 and n = 187. What is the plaintext ?

iii) What are the possible approaches to defeating the RSA algorithm ?

Solution : i) $n = p * q$

$n = 11 \times 17 \Rightarrow 187$

$\phi(n) = (p - 1)(q - 1)$

$= (17 - 1)(11 - 1) = 16 \times 10 = 160$

$e \cdot d = 1 \bmod \phi(n)$

$$7\,d = 1 \bmod 160$$

$$7 \times 23 = 1 \bmod 160$$

Public key PU $(e, n) = 7, 187$

Private key PR $(d, n) = 23, 187$

ii) $c = 11, \quad e = 7, \quad n = 187$

Plaintext $p = c^d \bmod n$

$$= 11^{23} \bmod 187$$

$$= 79720245 \bmod 187$$

$\therefore$ Plaintext = 88

Example 5.8.4 *Explain about the RSA algorithm with example as : $p = 11$, $q = 5$, $e = 3$ and $PT = 9$*

Solution : $p = 11, \quad q = 5$

$$n = p*q = 11 \times 5 = 55$$

$$\phi(n) = (p-1)*(q-1) = 10*4 = 40$$

$$e = 3 \text{ and } m = 9$$

$$\gcd(\phi(n), (e) = \gcd(40, 3) = 1$$

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

$$d \times e^{-1} (\bmod \phi(n)) = 1$$

$$3d \bmod 40 = 1$$

$$d = 27$$

public key $pu = \{e, n\} = \{3, 55\}$

private key $pr = \{d, n\} = \{27, 55\}$

Encryption : $C = M^e \bmod n = 9^3 \bmod 55 = 14$

decryption : $M = c^d \bmod n$

$$M = 14^{27} \bmod 55 = 9$$

Example 5.8.5 *In a public key system using RSA, the ciphertext intercepted is $C = 10$ which is sent to the user whose public key is $e = 5$, $n = 35$. What is the plaintext M ?*

Solution : Given data : C = 10, e = 5, n = 35

Find plaintext M

First calculate d : $e \cdot d = 1 \bmod \phi(n)$

$5d = 1 \bmod(24)$

$5 \times 5 \bmod (24) = 1$

$d = 5$

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

$= 10^5 \bmod (35)$

$= 100000 \bmod 35$

Plaintext M = 5

Example 5.8.6 *Perform encryption and decryption using the RSA algorithm for p = 3, q = 11, e = 7, M = 5.*

Solution : Given data : p = 3, q = 11, e = 7, M = 5

Calculate $n = p \times q = 3 \times 11$

$n = 33$

$\phi(n) = (p-1)(q-1) = 2 \times 10$

$\phi(n) = 20$

First calculate d : $e \cdot d = 1 \bmod \phi(n)$

$7d = 1 \bmod 20$

$7 \times 3 \bmod 20 = 1$

$d = 3$

To encrypt message m(<n), computers : $C = M^e \bmod n = 5^7 \bmod 33$

$= 78125 \bmod 33 = 14$

To decrypt received bit pattern, c, compute : $M = C^d$ mode $(n) = 14^3 \bmod(33)$

$= 2744 \bmod 33 = 5$

5.9 Digital Signatures

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

Requirements

- Message authentication protects two parties who exchange messages from any third party. However, it does not protect the two parties against each other.
- In situations where there is not complete trust between sender and receiver, something more than authentication is needed. The most attractive solution to this problem is the digital signature. The digital signature is analogous to the handwritten signature.
- It must have the following properties

  1. It must verify the author and the date and time of the signature.
  2. It must to authenticate the contents at the time of the signature.
  3. It must be verifiable by third parties, to resolve disputes.
- The digital signature function includes the authentication function. On the basis of these properties, we can formulate the following requirements for a digital signature.
- Must be a bit pattern depending on the message being signed.
- Signature must use some information unique to the sender to prevent forgery and denial.
- Computationally easy to produce a signature.
- Computationally easy to recognize and verify the signature.
- Computationally infeasible to forge a digital signature:

  a) either by constructing a new message for an existing digital signature.

  b) or by constructing a fraudulent digital signature for given message.
- Practical to retain a copy of the digital signature in storage.

Two general schemes for digital signatures

1) Direct
2) Arbitrated

5.9.1 Arbitrated Digital Signatures

Every signed message from A to B goes to an arbiter BB (Big Brother) that everybody trusts.

- *BB checks the signature and the timestamp, origin, content, etc.*
- *BB dates the message and sends it to B with an indication that it has been verified and it is legitimate.*

e.g. Every user shares a secret key with the arbiter

- A sends to BB in an encrypted form the plaintext P together with B's id, a timestamp and a random number RA.
- BB decrypts the message and thus makes sure it comes from A; it also checks the timestamp to protect against replays.
- BB then sends B the message P, A's id, the timestamp and the random number RA; he also sends a message encrypted with his own private key (that nobody knows) containing A's id, timestamp t and the plaintext P (or a hash).
- B cannot check the signature but trusts it because it comes from BB-he knows that because the entire communication was encrypted with KB.
- B will not accept the messages or messages containing the same RA to protect against replay.
- In case of dispute, B will show the signature he got from BB (only B may have produced it) and BB will decrypt it.

5.9.2 Direct Digital Signature

- This involves only the communicating parties and it is based on public keys.
- The sender knows the public key of the receiver.
- Digital signature : Encrypt the entire message (or just a hash code of the message) with the sender's private key.
- If confidentiality is required : Apply the receiver's public key or encrypt using a shared secret key.
- In case of a dispute the receiver B will produce the plaintext P and the signature E(KRA, P) - the judge will apply KUA and decrypt P and check the match : B does not know KRA and cannot have produced the signature himself.

Weaknesses

- The scheme only works as long as KRA remains secret : If it is disclosed (or A discloses it herself), then the argument of the judge does not hold : anybody can produce the signature.
- **Attack** : To deny the signature right after signing, simply claim that the private key has been lost-similar to claims of credit card misuse.
  i.e. If A changes her public-private keys (she can do that often) the judge will apply the wrong public key to check the signature.
- **Attack** : To deny the signature change your public-private key pair-this should not work if a PKI is used because they may keep trace of old public keys.
  i.e. A should protect her private key even after she changes the key.

- **Attack :** Eve could get hold of an old private key and sign a document with an old timestamp.

5.9.3 Digital Signature Standard

- The Digital Signature Standard (DSS) makes use of the Secure Hash Algorithm (SHA) and presents a new digital signature technique, the Digital Signature Algorithm (DSA). DSS cannot be used for encryption or key exchange. Fig. 5.9.1 shows the DSS approach.
- It uses a hash function. The hash code is provided as input to a signature function along with a random number K generated for this particular signature.
- The signature function also depends on the sender's private key (PR_a) and a set of parameters known to a group of communicating principles.
- The result is a signature consisting of two components, labeled s and r.
- At the receiving end, the hash code of the incoming message is generated. This plus the signature is input to a verification function.

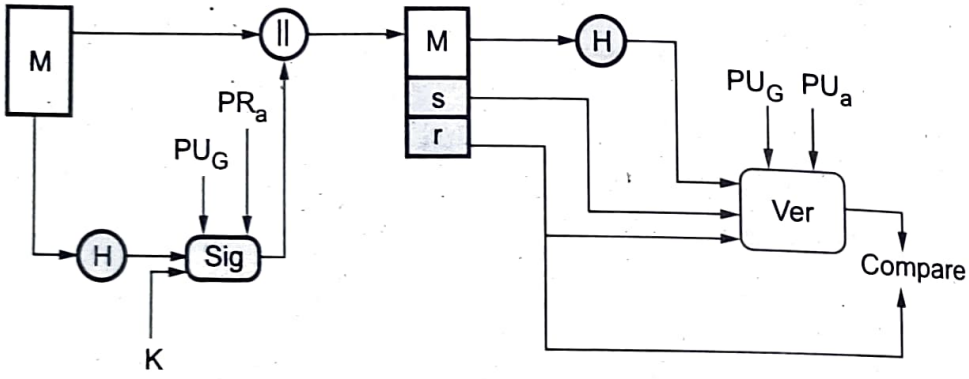


Fig. 5.9.1 DSS approach

- Fig. 5.9.2 shows the RSA approach. In the RSA approach, the message to be signed is input to a hash function that producs a secure hash code of fixed length. This hash code is then encrypted using the sender's private key to form the signature. Both the message and the signature are then transmitted.

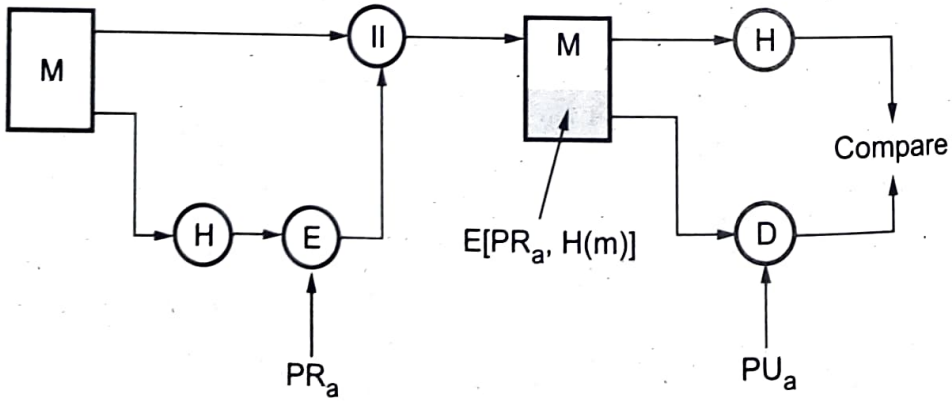
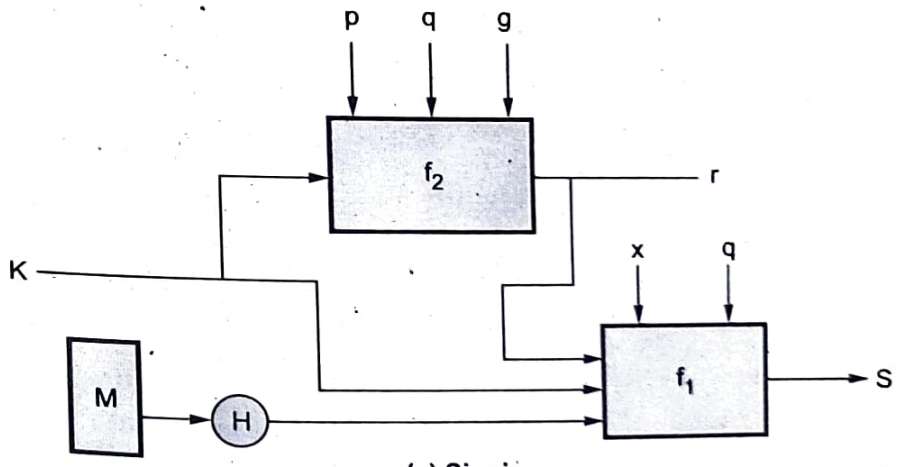


Fig. 5.9.2 RSA approach

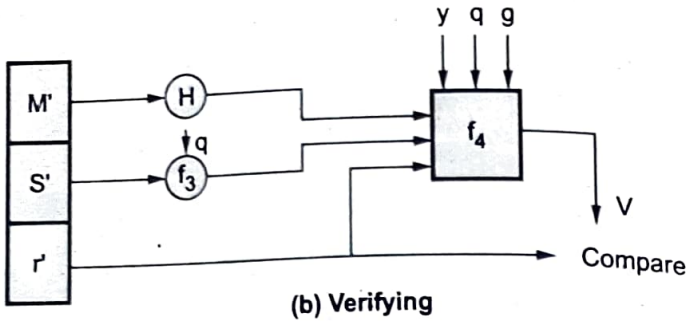
- The recipient takes the message and produces a hash code. The recipient also decrypts the signature using the sender's public key. If the calculated hash code matches the decrypted signature, the signature is accepted as valid.

5.9.4 Digital Signature Algorithm

- There are three parameters that are public and can be common to a group of users. **Prime number q** is chosen and it is **160-bit**. A **prime number p** is selected with a length between **512** and **1024 bits** such that q divides (P – 1).
- g is chosen to be of the form $h^{(P-1)/q}$ mod p where h is an integer between 1 and (P – 1)
- With these number, user selects a private key and generate a public key. The private key x must be a number from 1 to (q – 1) and should be chosen randomly or pseudorandomly.
- The public key is calculated from the private key as $y = g^x \bmod p$.
- To create a signature, a user calculates two quantities, **rands**, that are functions of
  i) Public key components (p, q, g)
  ii) User's private key (x)
  iii) Hash code of the message H(M)
  iv) An additional integer (K)
- **At the receiving end,** verification is performed. The receiver generates a quantity V that is a function of the public key components, the sender's public key and the hash code of the incoming message. If this quantity matches the r components of the signature, then the signature is validated.
- Fig. 5.9.3 shows the functions of signing and verifying.



(a) Signing



(b) Verifying

Fig. 5.9.3 Signing and verifying

5.10 Digital Certificate

- A data structure that securely binds an individual or entity to a public key used in cryptographic operations such as digital signatures or asymmetric encryption.
- To obtain digital certificate an organization must apply to a certification authority which is responsible for validating and ensuring the authenticity of requesting organization. The certificate will identify the name of the organization, a serial number, the validity date and the organization's public key where encryption to / from that organization is required.
- In addition, the digital certificate will also contain the digital signature of the certification authority to allow any recipient to confirm the authenticity of the digital certificate.
- A digital certificate is an ID that is carried with a file. To validate a signature, a certifying authority validates information about the software developers and then issues them digital certificates. The digital certificate contains information about the person to whom the certificate was issued, as well as information about the certifying authority that issued it. When a digital certificate is used to sign programs, ActiveX controls, and documents, this ID is stored with the signed item in a secure and verifiable form so that it can be displayed to a user to establish a trust relationship.
- A digital certificate allows unique identification of an entity; it is essentially an electronic ID card, issued by a trusted third party. Digital certificates form part of the ISO authentication framework, also known as the X.509 protocol. This framework provides for authentication across networks.
- A digital certificate serves two purposes : It establishes the owner's identity, and it makes the owner's public key available. A digital certificate is issued by a Certification Authority (CA). It is issued for only a limited time and when its expiry date has passed, it must be replaced.

- A digital certificate consists of :
  1. The public key of the person being certified
  2. The name and address of the person being certified, also known as the Distinguished Name (DN)
  3. The digital signature of the CA
  4. The issue date
  5. The expiry date
- The Distinguished Name is the name and address of a person or organization. You enter your Distinguished Name as part of requesting a certificate. The digitally-sighed certificate includes not only your own Distinguished Name, but the Distinguished Name of the CA, which allows verification of the CA.
- To communicate securely, the receiver in a transmission must trust the CA that issued the certificate that the sender is using. This means that when a sender signs a message, the receiver must have the corresponding CA's signer certificate and public key designated as a trusted root key. For example, your web browser has a default list of signer certificates for trusted CAs. If you want to trust certificates from another CA, you must receive a certificate from that CA and designate it as a trusted root key.
- If you send your digital certificate containing your public key to someone else, what keeps that person from misusing your digital certificate and posing as you? The answer is: your private key.
- A digital certificate alone is not proof of anyone's identity. The digital certificate allows verification only of the owner's identity, by providing the public key needed to check the owner's digital signature. Therefore, the digital certificate owner must protect the private key that belongs with the public key in the digital certificate. If the private key were stolen, anyone could pose as the legitimate owner of the digital certificate.

5.11 Diffie-Hillman Key Exchange Algorithm

- The Diffie-Hellman key agreement protocol was developed by Diffie and Hellman in 1976. This protocol allows two users to exchange a secret key over an insecure medium without any prior secrets.
- The protocol has two system parameters p and g. They are both public and may be used by all the users in a system.

- Parameter p is a prime number and parameter g is an integer less than p, with the following property :
  1. For every number n between 1 and p – 1 inclusive.
  2. There is a power k of g such that $n = g^k \bmod p$.
- The protocol depends on the discrete logarithm problem for its security. It assumes that it is computationally infeasible to calculate the shared secret key $k = g^{ab} \bmod p$ given the two public values $g^a \bmod p$ and $g^b \bmod p$ when the prime p is sufficiently large.
- The Diffie-Hellman key exchange is vulnerable to a man-in-the-middle attack. This vulnerability is present because Diffie-Hellman key exchange does not authenticate the participants. Possible solutions include the use of digital signatures and other protocol variants.
- Suppose Alice and Bob want to agree on a shared secret key using the Diffie-Hellman key agreement protocol. They proceed as follows :
  1. First, Alice generates a random private value a and Bob generates a random private value b.
  2. Both a and b are drawn from the set of integers. They derive their public values using parameters p and g and their private values.
  3. Alice's public value is $g^a \bmod p$ and Bob's public value is $g^b \bmod p$.
  4. They then exchange their public values.
  5. Finally, Alice computes $g^{ab} = (g^b)^a \bmod p$.
  6. Bob computes $g^{ba} = (g^a)^b \bmod p$.
  7. Since $g^{ab} = g^{ba} = k$, Alice and Bob now have a shared secret key k.

Algorithm :

- Select two numbers (1) prime number q (2) α an integer that is a primitive root of q.
- Suppose the users A and B wish to exchange a key.
  1. User A select a random integer $X_A < q$ and computes $Y_A = \alpha^{X_A} \bmod q$.
  2. User B selects a random integer $X_B < q$ and compute $Y_B = \alpha^{X_B} \bmod q$.
  3. Both side keeps the X value private and makes the Y value available publicly to the other side.
  4. User A computes the key as $K = (Y_B)^{X_B} \bmod q$.
  5. User B computes the key as $K = (Y_A)^{X_B} \bmod q$.

- Both side gets same results :

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

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

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

Example

- Key exchange is based on the use of the prime number and a primitive root of prime number.
- Prime number $q = 353$
  Primitive root $\alpha = 3$
- A and B select secret keys.
  $X_A = 97$ $X_B = 233$
- Calculates the public keys

A computes $Y_A = \alpha^{X_A} \bmod q$

$= (3)^{97} \bmod 353 = (1.9080 \times 10^{97}) \bmod 353 = 40$

B computes $Y_B = \alpha^{X_B} \bmod q$

$= (3)^{233} \bmod 353 = (1.4765 \times 10^{111}) \bmod 353 = 248$

- After they exchange public keys, each can compute the common *secret key*.

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

$= (1.8273 \times 10^{232}) \bmod 353 = 160$

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

$= (1.9053 \times 10^{373}) \bmod 353 = 160$

Problems

Example 5.11.1 *User A and B use the Diffie-Hellman key exchange technique with a common prime q = 71 and a primitive root α = 7.*

a) If user A has private key $X_A = 5$, what is A's public key Y_A ?

b) If user B has private key $X_B = 12$, what is B's public key Y_B ?

c) What is the shared secret key ?

Solution :

a) **A's public key Y_A**

$$Y_A = \alpha^{X_A} \bmod q = (7)^5 \bmod 71 = 16807 \bmod 71 = 51$$

b) B's public key Y_B

$$Y_B = \alpha^{X_B} \bmod q = (7)^{12} \bmod 71 = 13841287201 \bmod 71 = 4$$

c) Shared secret key

i) At user A $K = (Y_B)^{X_A} \bmod q$

$= (4)^5 \bmod 71 = 1024 \bmod 71$

$K = 30$

The man in middle attack can work against the Diffie-Hellman key exchange algorithm, causing it to fail.

Advantages

1. Any user can choose a random x and publish g^x in a public database such as a phone book.
2. Phone book must be maintained by a TTP.
3. Other users can look up the database and get the public key for the individual and use it to encrypt the message.
4. Ideal for use with emails.

Disadvantages

1. Does not protect against man-in-the-middle attacks.
2. Even can intercept all traffic between Alice and Bob and generate separate keys for communication with them.
3. If Allice sends an encrypted message for Bob with his public key, Eve simply forwards it.
4. For large prime p, (p – 1) is an even number and so Z_p^* will have an subgroup of order 2.

Example 5.11.2 *If generator g = 2 and n or P = 11, using Diffie-Hellman algorithm solve the following :*

i) Show that 2 is a primitive root of 11.

ii) If A has a public key '9' what is A's private key ?

iii) If B has a public key '3' what is B's private key ?

iv) Calculate the shared secret key.

Solution : i)

$2^1 \bmod 11 = 2$

$2^2 \bmod 11 = 4$

$2^3 \bmod 11 = 8$

$2^4 \bmod 11 = 5$

$2^5 \bmod 11 = 10$

$2^6 \bmod 11 = 9$

$2^7 \bmod 11 = 7$

$2^8 \bmod 11 = 3$

$2^9 \bmod 11 = 6$

Using 2 as integer, we get all the integer values between 1 to 11. So 2 is a primitive root of 11.

ii) Public key = 9

$2^6 \bmod 11 = 9$

$X_A = 6$

iii) $Y_B = (11)^6 \bmod 9$

$Y_B = 1$

iv) **Shared secret key :**

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

$$K = 3^6 \bmod 11$$

$$K = 3$$

5.12 Hash Function

- A hash function takes an input m, and computes a fixed size string known as a hash. Unlike a MAC, a hash code does not use a key but is a function only of the input message.
- **Definition :** A hash function is a computationally efficient function mapping binary strings of arbitrary length to binary strings of some fixed length, called hash-values.
- The data to be encoded is often called the "message", and the hash value is sometimes called the message digest or simply digests.

- A hash function maps a variable-length input into a fixed-length output. This hash function output can be treated as a fingerprint of the input data. A very simple example of hash function is modulo operation. Hash functions have been used in many fields of computer science such as hash table in data structure, checksum algorithms for error detection, digital signature in information security etc.
- The most common cryptographic uses of hash functions are with digital signatures and for data integrity.
- When hash functions are used to detect whether the message input has been altered, they are called Modification Detection Codes (MDC).
- There is another category of hash functions that involve a secret key and provide data origin authentication, as well as data integrity; these are called Message Authentication Codes (MACs).
- A hash value h is generated by a function H of the form.

$$h = H(M)$$

where M = Variable - Length message

$H(M)$ = Fixed - Length hash value.

- Hash code is also referred to as a message digest or hash value. A change to any bit or bits in the message results in a change to the hash code.
- Fig. 5.12.1 (a) shows the basic uses of hash function.

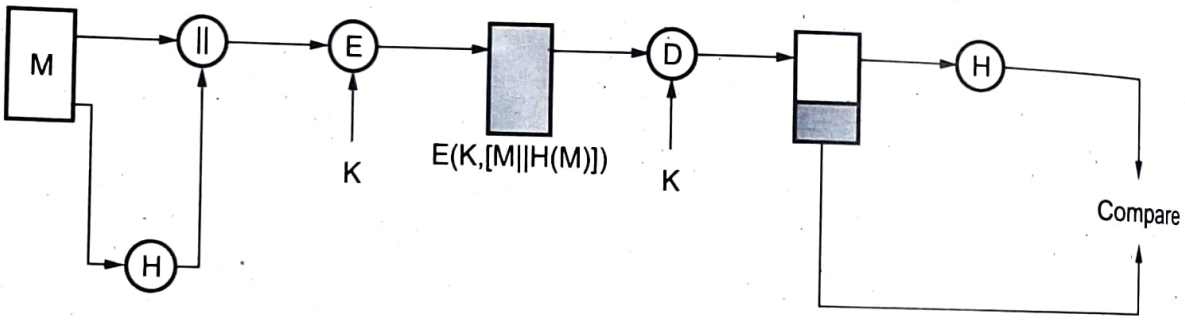


Fig. 5.12.1 (a) Encrypt message plus hash code

1. **Encrypt message plus hash code.**
   - Provide confidentiality : Only A and B share K.
   - Provides authentication : H(M) is cryptographically protected.
2. **Encrypt hash code - shared secret key**
   - Only the hash code is encrypted, using symmetric encryption.
   - Reduces the processing burden for those applications that do not require confidentiality.

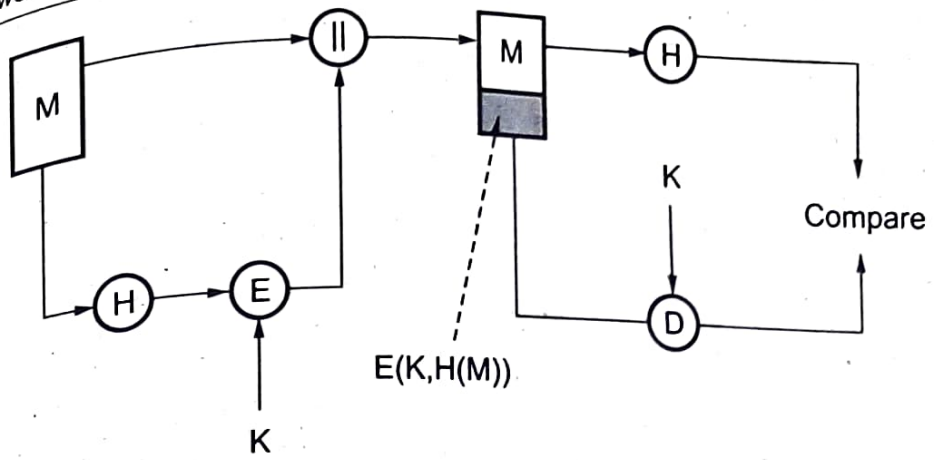


Fig. 5.12.1 (b) Encrypt hash code - shared secret key

3. **Encrypt hash code - sender's private key**

- It provides authentication and digital signature.

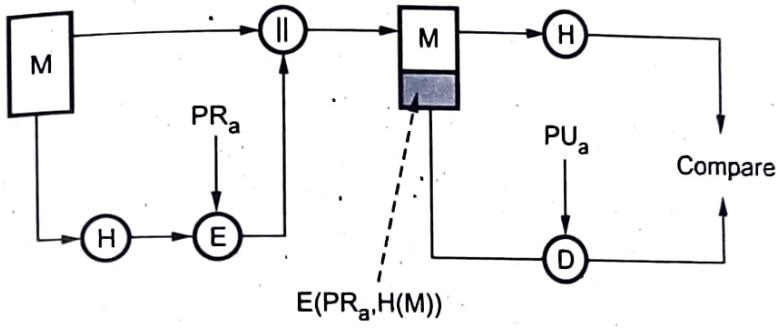


Fig. 5.12.1 (c) Encrypt hash code - sender's private key

5.12.1 Requirements for a Hash Functions

- The purpose of a hash function is to produce a fingerprint of a file, message or other block of data.

Properties

1. H can be applied to a block of data of any size.
2. H produces a fixed length output.
3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. For any given value h, it is computationally infeasible to find x such that $H(x) = h$. This is called one-way property.
5. For any given block x, it is computationally infeasible to find $y \neq x$ such that $H(y) = H(x)$. This is called as **weak collision** resistance.
6. It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$. This is called as **strong collision** resistance.

5.12.2 One-way Hash Function

- A one-way hash function is also known as a message digest, fingerprint or compression function. It is a mathematical function and takes a variable-length input string and converts it into a fixed-length binary sequence.
- One-way hash function is designed in such a way that it is hard to reverse the process. A good hash function also makes it hard to find two strings that would produce the same hash value. All modern hash algorithms produce hash values of 128 bits and higher.
- Even a slight change in an input string should cause the hash value to change drastically. Even if 1 bit is flipped in the input string, at least half of the bits in the hash value will flip as a result. This is called an **avalanche effect.**
- A common way for one-way hash functions to deal with the variable length input problem is called a compression function. Compression functions work by viewing the data being hashed as a sequence of n fixed-length blocks.
- To compute the hash value of a given block, the algorithm needs two things : The data in the block and an input seed.
- The input seed is set to some constant value, c, and the algorithm computes the hash value h_1 of the first block. Next, the hash value of the first block, h_1 is used as the seed for the second block.
- The function proceeds to compute the hash value of the second block based on the data in the second block and the hash value of the first block, h_1. So, the hash value for block n is related to the data in block n and the hash value h_{n-1} (for $n > 1$). The hash value of the entire input stream is the hash value of the last block.

5.12.3 Application of Hash Function

- A typical use of a cryptographic hash would be as follows :

1. Alice poses a tough math problem to Bob, and claims she has solved it. Bob would like to try it himself, but would yet like to be sure that Alice is not bluffing. Therefore, Alice writes down her solution, appends a random nonce, computes its hash and tells Bob the hash value. This way, when Bob comes up with the solution himself a few days later, Alice can prove that she had the solution earlier by revealing the nonce to Bob.
2. Second application of secure hashes is verification of message integrity. Determining whether any changes have been made to a message, for example, can be accomplished by comparing message digests calculated before, and after, transmission. A message digest can also serve as a means of reliably identifying a

file; several source code management systems, including Git, Mercurial and Monotone, use the sha1sum of various types of content (file content, directory trees, ancestry information, etc) to uniquely identify them.

3. A related application is password verification. Passwords are usually not stored in clear text, for obvious reasons, but instead in digest form. To authenticate a user, the password presented by the user is hashed and compared with the stored hash. This is sometimes referred to as one-way encryption.

4. Hash functions can also be used in the generation of pseudorandom bits. Hashes are used to identify files on peer-to-peer file sharing networks. For example, in an ed2k link, an MD4-variant hash is combined with the file size, providing sufficient information for locating file sources, downloading the file and verifying its contents. Magnet links are another example. Such file hashes are often the top hash of a hash list or a hash tree which allows for additional benefits.

5.12.4 Birthday Attack

- A birthday attack refers to a class of brute-force attacks.
- The attack is named after the statistical property of birthday duplication - you only need 23 people to have a larger than 50 % chance that they are born on the same day of the year.
- This is due to the fact that each time you adding one person to the set of people you are looking for duplicates in, you are looking for duplicates against all the people already in the set, not just one of them.
- The same technique can be used to look for conflicts in one-way functions. Instead of taking one output of the one-way function, you create or acquire a set of values (let us call this a) that have a some property and then create another set of other values that have different properties (let us call this b) and try to find any value that is in both a and b. This is a much smaller problem that finding a value that match a particular value in a.
- The properties in a and b might for instance be
  1. a contains secure hashes of an innocent message and b contains one of a less innocent message, so the attacker can substitute the messages at a later date.
  2. a is the password hashes of a system the attacker wants to get an account on, and b is a set of password hashes that the attacker knows the passwords for.
  3. a is the set of public keys from a Discrete Logarithms based cryptosystem where g and p are static, while b is the set of g^e mod p functions that the attacker knows e for.

- Birthday attacks are often used to find collisions of hash functions. To avoid this attack, the output length of the hash function used for a signature scheme can be chosen large enough so that the birthday attack becomes computationally infeasible.
- Resistance against this attack is why the Unix password hashes use a salt.

5.12.5 Attack on Collision Resistance

- Weak collision resistance : for any x, it is hard to find $x' \neq x$ such that $h(x)=h(x')$.
- Strong collision resistance : it is hard to find any x, x' for which $h(x) = h(x')$.
- It's easier to find collisions. Therefore strong collision resistance is a stronger assumption.
- Real world hash functions : MD5, SHA-1, SHA-256.
- The weak collision property refers guarantees that an alternative message yielding the same code cannot be found. This prevents forgery when an encrypted hash code is used.
- The strong collision property refers to how resistant the hash function is to a class of attacks known as the birthday attack.

5.12.6 Requirements and Security

- Attacks are of two types.
  1. Brute-force attack
  2. Cryptanalysis

Brute - force attacks

1. Hash functions

- The strength of a hash function against brute-force attacks depends solely on the length of the hash code produced by the algorithm.
- **Desirable properties**
  a. **One way :** For any given code h, it is computationally infeasible to find x such that $H(x) = h$.
  b. **Weak collision resistance :** For any given block x, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$.
  c. **Strong collision resistance :** It is computationally infeasible to find any pair (x,y) such that $H(x) = H(y)$.

- For a hash code of length n, the level of effort required, as we have seen is proportional to the following :

One way	2^n
Weak collision resistance	2^n
Strong collision resistance	$2^{n/2}$

2. Message authentication codes

- Given one or more text MAC pair $[x_i, C(K, x_i)]$ it is computationally infeasible to compute any text MAC pair $[x, C(K, x)]$ for any new input $x \neq x_i$.
- The attacker would like to come up with the valid MAC code for a given message x.
- There are two lines of attack possible. Attack the key space and attack the MAC value.
- If an attacker can determine the MAC key then it is possible to generate a valid MAC value for any input x.
- An attacker can also work on the MAC value without attempting to recover the key. Here, the objective is to generate a valid MAC value for a given message or to find a message that matches a given MAC value.
- The level of effort for brute-force attack on a MAC algorithm can be expressed as min $(2^k, 2^n)$.

Cryptanalysis

Hash functions

- The hash algorithm involves repeated use of a compression function (f), that takes two inputs and produces an n-bit output.
- Cryptanalysis of hash functions focuses on the internal structure of f and is based on attempts to find efficient techniques for producing collisions for a single execution of f.

5.12.7 Applications of Cryptographic Hash Functions

- Message Authentication and Digital Signatures are the two main application of the cryptographic hash function.

Message authentication

- Message authentication is an alternative technique which uses secret key. This technique assumes that two communicating parties, share a common secret key K. When A has a message to send to B, it calculates the MAC.

MAC = C(K, M)

where, M = Input message

C = MAC function

K = Shared secret key

MAC = Message authentication code.

- Calculated MAC and message are transmitted to the receiver. The receiver performs the same calculation on the received message.
- Received MAC is compared with the calculated MAC. If both are matches, then

1. The receiver is assured that the message has not been altered.
2. The receiver is assured that the message is from the alleged sender.
3. If the message includes a sequence number, then the receiver can be assumed of the proper sequence because an attacker cannot successfully alter the sequence number.

- Fig. 5.12.2 shows the message authentication.

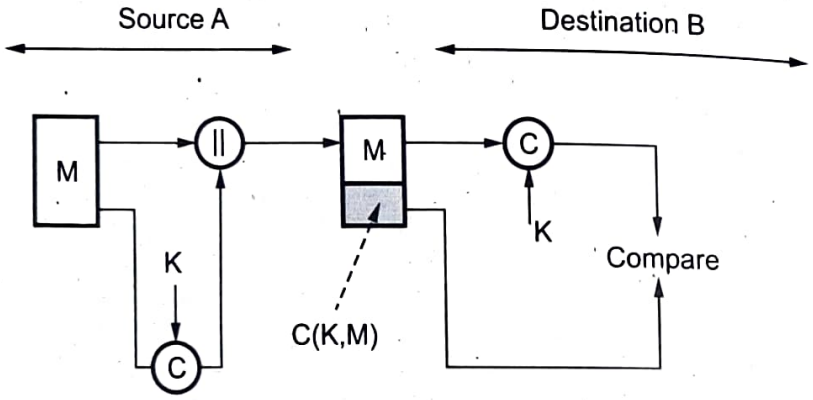


Fig. 5.12.2 Message authentication

- Above figure provides authentication but not confidentiality. Confidentiality can be provided by performing message encryption either after or before the MAC algorithm.
- Fig. 5.12.3 shows encryption after the MAC.

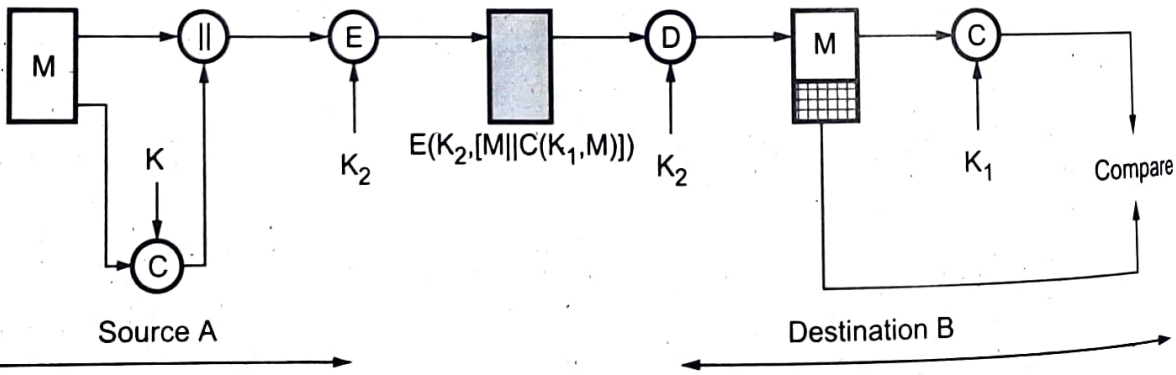


Fig. 5.12.3 Message authentication and confidentiality

- Two separate keys are needed, each of which is shared by the sender and the receiver. Here MAC is calculated with the message input and is then concatenated to the message. The entire block is then encrypted.
- When a hash function is used to provide message authentication, the hash function value is often referred to as a **message digest.**
- Message authentication is achieved using a Message Authentication Code (MAC), also known as a keyed hash function.

Digital Signatures

- In the digital signature, the hash value of a message is encrypted with a user's private key. Anyone who knows the user's public key can verify the integrity of the message that is associated with the digital signature.
- If confidentiality as well as a digital signature is desired, then the message plus the private-key-encrypted hash code can be encrypted using a symmetric secret key.

Other Applications

- One-way password file is created by hash function. Hash functions can be used for intrusion detection and virus detection.
- A cryptographic hash function can be used to construct a pseudorandom function (PRF) or a pseudorandom number generator. A common application for a hash-based PRF is for the generation of symmetric keys.

5.12.8 Simple Hash Functions

- For a hash function, the input is viewed as a sequence of n-bit blocks. The input is processed one block at a time in an iterative fashion to produce an n-bit hash function.
- One of the simplest hash functions is the bit-by-bit exclusive-OR of every block. This can be expressed as follows :

$$C_i = b_{i1} \oplus b_{i2} \oplus b_{i3} \oplus \ldots\ldots \oplus b_{im}$$

where C_i = i^{th} bit of the hash code, $1 \leq i \leq n$.

m = number of n-bit blocks in the input

b_{ij} = i^{th} bit in j^{th} block

$\oplus$ = XOR operation

- Fig. 5.12.4 shows two types of hash functions. (See Fig. 5.12.4 on next page)

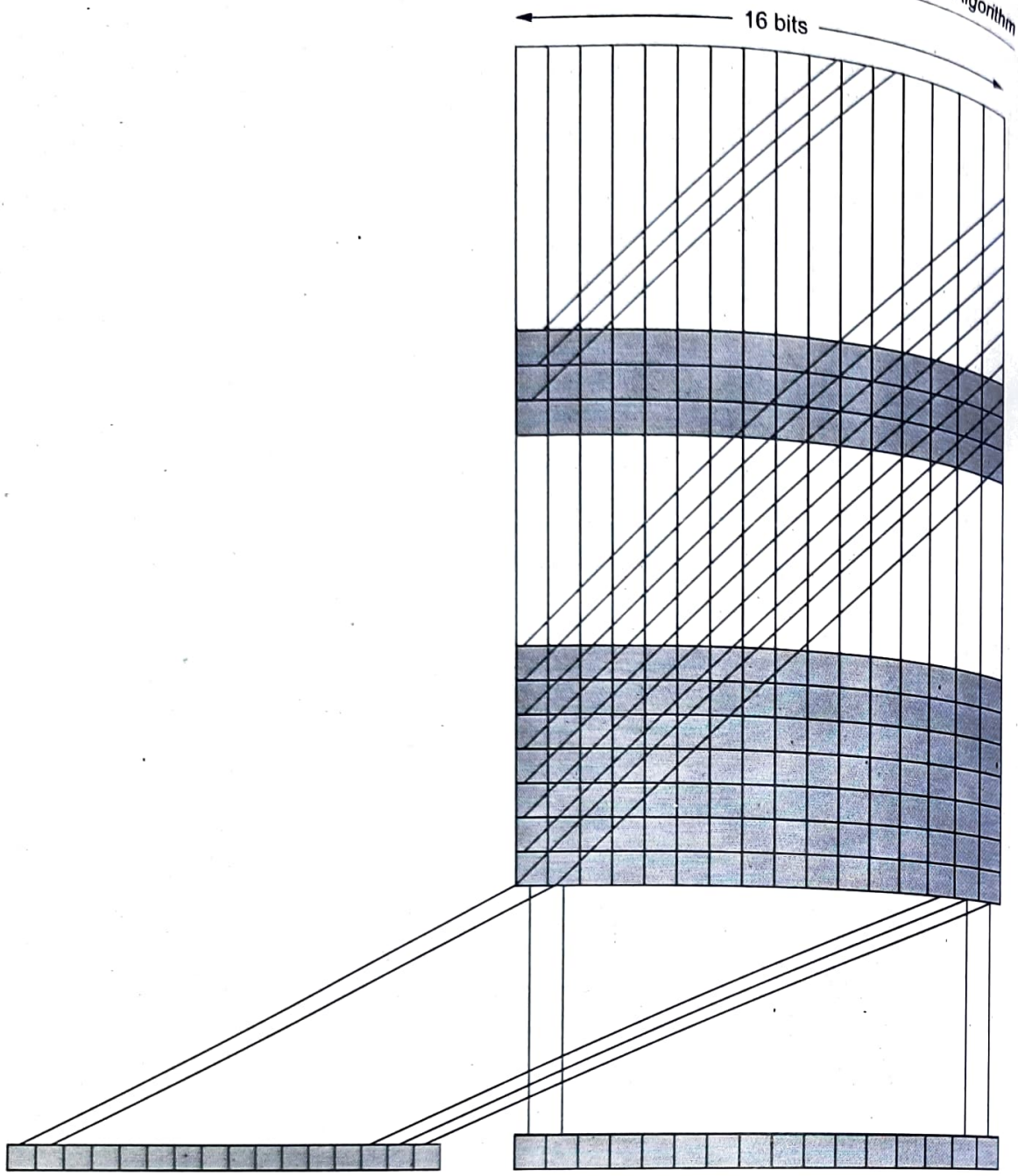


Fig. 5.12.4 Two simple hash functions

- A simple way to improve matters is to perform a one bit circular shift or rotation, on the hash value after each block is processed. The procedure is as follows :

1. Initially set the n-bit hash value to zero.
2. Process each successive n-bit block of data as follows.
   a. Rotate the current hash value to the left by one bit.
   b. XOR the block into the hash value.

5.12.9 Hash Functions based on Cipher Block Chaining

- Two major categories of hash functions are : **dedicated hash functions and block cipher-based hash functions.**
- Block cipher is a popular encryption-decryption primitive. To encrypt, the block cipher accepts a key K and a plaintext block x as input and produces a cipher text block $c = E(K, x)$, also written as $c = E_K(x)$.
- Given a message M consisting of a sequence of 64-bit blocks $P_1; P_2; : : : ; P_N$, define the hash code $h = H(M)$ as the block-by-block XOR of all blocks and append the hash code as the final block :

$$h = P_{N+1} = P_1 \oplus P_2 \oplus \ldots\ldots \oplus P_N$$

- Encrypt the entire message plus the hash code using CBC mode to produce the encrypted message $C_1; C_2; : : ; C_{N+1}$. There are several ways the ciphertext can be manipulated in such a way that it is not detectable by the hash code.
- By the definition of CBC :

$$CBC : C_j = E(K, [C_{j-1} \oplus P_j])$$

So we have;

$$P_1 = IV \oplus D(K, C_1)$$

$$P_i = C_{i-1} \oplus D(K, C_i)$$

$$P_{N+1} = C_N \oplus D(K, C_{N+1})$$

- But, P_{N+1} has the hash code.

$$P_{N+1} = P_1 \oplus P_2 \oplus \ldots\ldots \oplus P_N$$

$$= [IV \oplus D(K, C_1)] \oplus [C_1 \oplus D(K, C_2)] \oplus \ldots \oplus [Y_{C+1} \oplus D(K, C_N)]$$

- Because the terms in the preceding equation can be XOR'ed in any order, it follows that the hash code would not change if the ciphertext blocks were permuted.

5.13 Key Management

- The purpose of public key cryptography is
  1. The distribution of public keys.
  2. The use of public key encryption to distribute secret keys.

5.13.1 Distribution of Public Keys

- Different methods have been proposed for the distribution of public keys. There are

1. Public announcement.
2. Publicly available directory.
3. Public key authority.
4. Public key certificates.

1. Public announcement

- In public key algorithm, any participant can **send** his or her public key to any other participant or **broadcast** the key to the community at large.
- Fig. 5.13.1 shows the public key distribution.

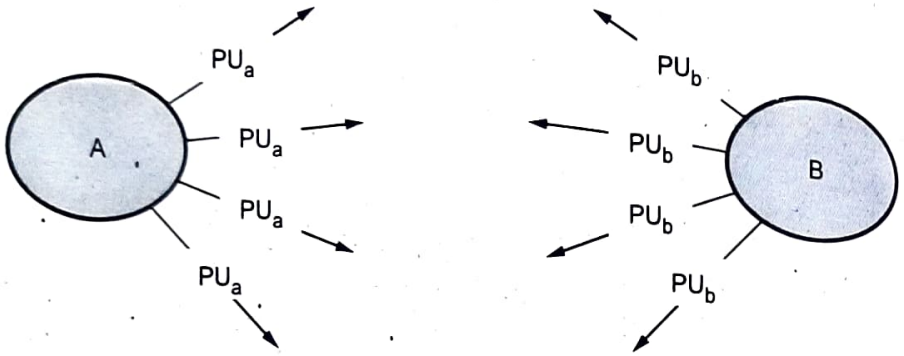


Fig. 5.13.1 Public key distribution

- Because of the growing popularity of PGP, which makes use of RSA, many PGP users have adopted the practice of appending their public key to messages that they send to public forums, such as USENET newgroups and Internet mailing lists.
- The disadvantage is that, 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.

2. Public available directory

- 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.
- Fig. 5.13.2 shows public key publication. (See Fig. 5.13.2 on next page.)
- Such a scheme would include the following elements :

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

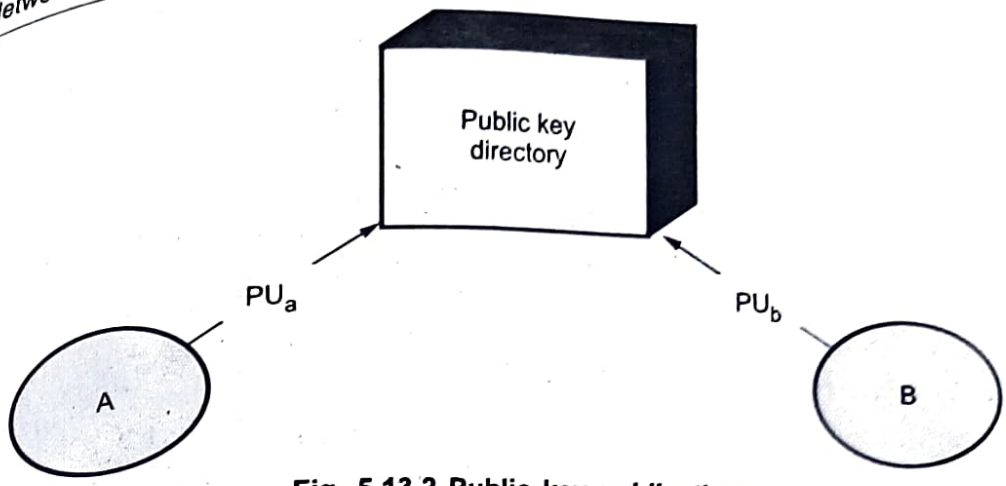


Fig. 5.13.2 Public key publication

2. Each participant registers a public key with the directory authority. Registration would have to be in person or by some form of secure authenticated communication.
3. A participant may replace the existing key with a new one at any time.
4. Participants could also access the directory electronically.

3. Public key authority

- Fig. 5.13.3 shows public key distribution scenario

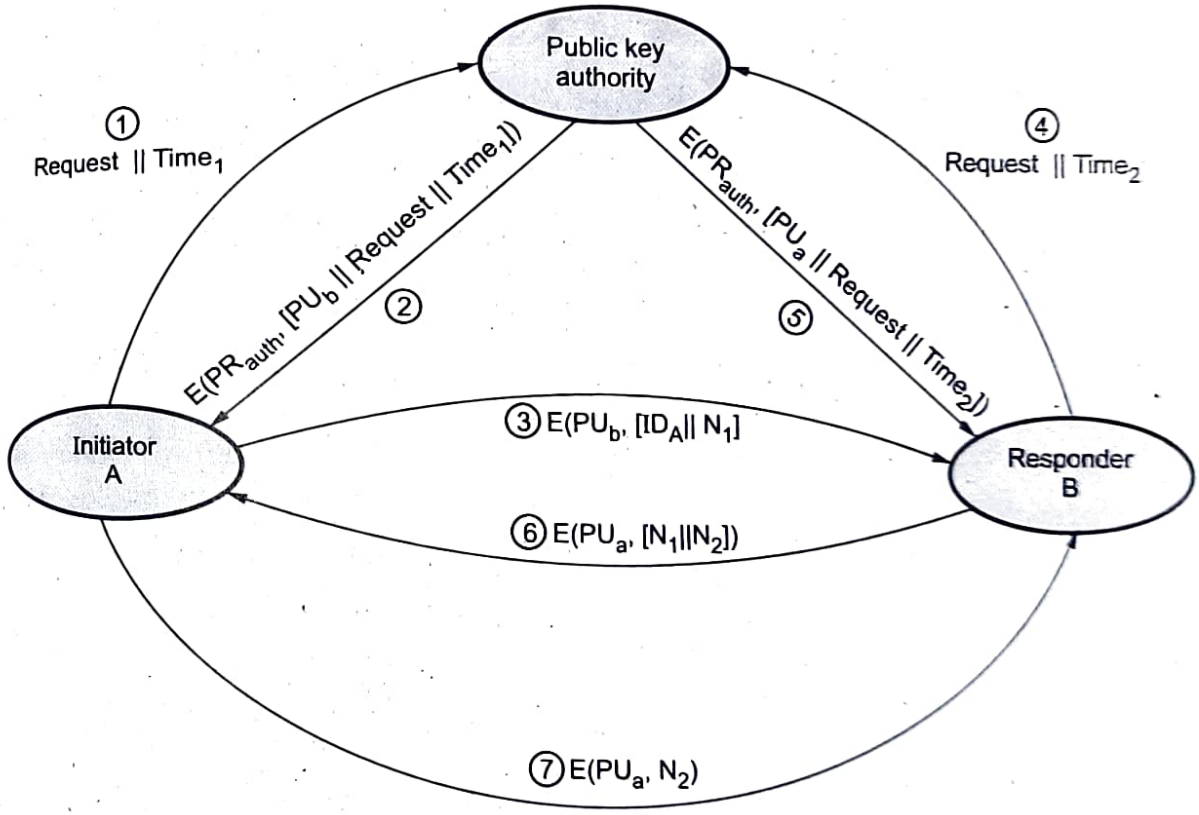


Fig. 5.13.3 Public key distribution scenario

- Following steps occur in public key distribution.

1. A sends a timestamped message to the public key authority containing a request for the current public key of B.
2. The authority responds with a message that is encrypted using the authority's private key, PR_{auth}. The message also contains B's public key (PU_b), original request and timestamp.
3. A stores B's public key and also uses it to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1) which is used to identify this transaction uniquely.
4. B retrieves A's public key from the authority in the same manner as A retrieved B's public key.
5. Public keys have been securely delivered to A and B and they may begin their protected exchange.
6. B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B(N_2).
7. A returns N_2, encrypted using B's public key, to assure B that its correspondent is A.

Drawback

Public key authority could be somewhat of a bottleneck in the system. The directory of name and public keys maintained by the authority is vulnerable to tampering.

4. Public key certificates

- Certificates can be used by participants to exchange keys without contacting a public key authority. Certificate consists of a public key plus an identifier of the key owner, with the whole block signed by a trusted third party.
- The third party is a certificate authority, such as 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.
- Requirements on this scheme :

1. Any participant can read a certificate to determine the name and public key of the certificate's owner.
2. Any participant can verify that the certificate originated from the certificate authority and is not counterfeit.
3. Only the certificate authority can create and update certificates.
4. Any participant can verify the currency of the certificate.

- A certificate scheme is illustrated in Fig. 5.13.4. Each participant applies to the certificate authority, supplying a public key and requesting a certificate.

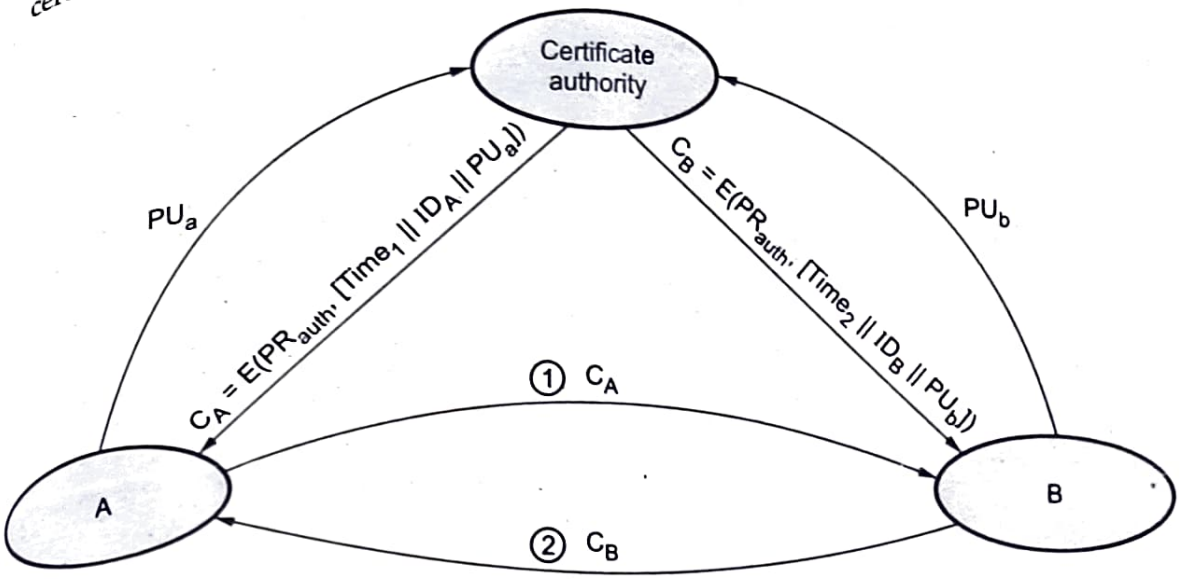


Fig. 5.13.4 Exchange of public key certificates

- For participant A, the authority provides a certificate of the form

$$C_A = E(PR_{auth}, [T || ID_A || PU_a])$$

- where PR_{auth} is the private key used by the authority and T is a timestamp.

5.13.2 Distribution of Secret Keys using Public Key Cryptography

- Public key encryption provides for the distribution of secret key to be used for conventional encryption.

Simple secret key distribution

If user A wishes to communicate with user B, the following procedure is employed :

1. User A generates a public/private key pair $\{PU_a, PR_a\}$ and transmits a message to user B consisting of PU_a and an identifier of A, ID_A.
2. User B generates a secret key (K_s) and transmits it to user A, encrypted with A's public key.
3. User A computes $D(PR_a, E(PU_a, K_s))$ to recover the secret key. Because only A can decrypt the message, only user A and user B know the identity of K_s.
4. User A discards PU_a and PR_a and user B discards PU_a.
5. Fig. 5.13.5 shows use of public key encryption.

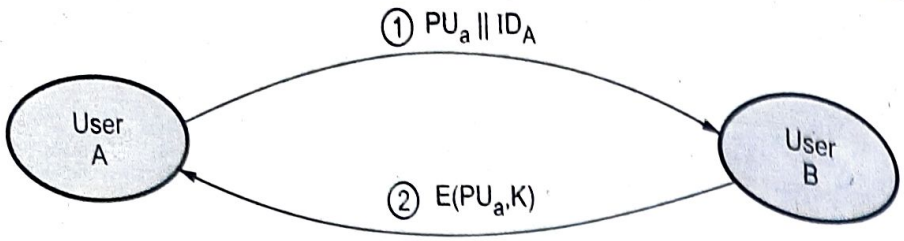


Fig. 5.13.5 Use of public key encryption

- User A and B can now securely communicate using conventional encryption and the session key K_s. At the completion of the exchange, both user A and B discard K_s.
- The protocol discussed above is insecure against an adversary who can intercept messages and then either relay the intercepted message or substitute another message. Such an attack is known as a **man in middle attack.**

Secret key distribution with confidentiality and authentication

- Fig. 5.13.6 shows the public key distribution of secret keys.

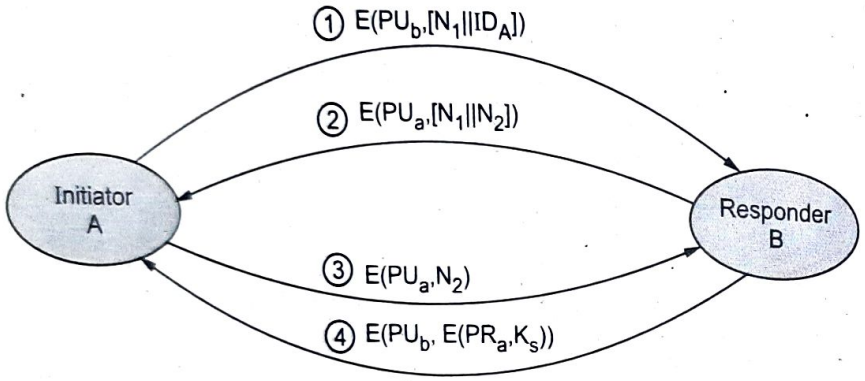


Fig. 5.13.6 Public key distribution of secret keys

- It provides protection against both passive and active attacks.

1. A uses B's public key to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1), which is used to identify this transaction uniquely.
2. B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B(N_2).
3. A returns N_2, encrypted using B's public key, to assure B that its correspondent is A.
4. A selects a secret key K_s and sends $M = E(PU_b, E(PR_a, K_s))$ to B.
5. B computes $D(PU_a, D(PR_b, M))$ to recover the secret key.

5.13.3 Key Distribution and Certification

- Management and handling of the pieces of secret information is generally referred to as **key management.**
- Activities of key management includes selection, exchange, storage, certification, revocation, changing, expiration and transmission of the key.
- Key management is the set of processes and mechanisms which support key establishment and maintenance of ongoing keying relationship between parties, including replacing older key with new keys.
- Two major issues in key management are :
  1. Key life time
  2. Key exposure

Key life time - limit of use which can be measured as a duration of time.

Issue related to key :

1. Users must be able to obtain securely a key pair suited to their efficiency and security needs.
2. Keys need to be valid only until a specified expiration date.
3. The expiration date must be chosen properly and publicized securely.
4. User must be able to store their private keys securely.
5. Certificates must be unforgettable, obtainable in a secure manner.

1. Public Key Infrastructure

- Public Key Infrastructure (PKI) is a well-known technology that can be used to establish identities, encrypt information and digitally sign documents.
- PKI identifies and manages relationships of parties in an electronic exchange, serving a wide array of security needs including access control, confidentiality, integrity, authentication and non-repudiation.
- PKI also uses unique Digital Certificates (DC) to secure eCommerce, email, data exchange and Virtual Private Networks (VPN) and intranets and is also used to verify the identity and privileges of each user.
- The Certificate Authority (CA) provides a full life-cycle management of public keys and certificates, including issuance, authentication, storage, retrieval, backup, recovery, updating and revocation to the PKI.
- All users of PKI must have a registered identity, which is stored in a digital certificate issued by a CA.

- Remote users and sites using public private keys and public key certificates can authenticate each other with a high degree of confidence.
- Authentication is dependent on three conditions :

1. It must be established that each party have a private key that has not been stolen or copied from the owner.
2. The certificate must be issued to the owner in accord with the stated policy of the certificate issuer.
3. The policies of the certificate issuer must be satisfactory to the parties so as to verify identity.

Benefits of PKI

1. Confidential communication : Only intended recipients can read files.
2. Data integrity : Guarantees files are unaltered during transmission.
3. Authentication : Ensures that parties involved are who they claim to be.
4. Non-repudiation : Prevents individuals from denying.

Limitation of PKI

The problems encountered deploying a PKI can be categorized as follows :

1. Public key infrastructure is new
2. Lack of standards
3. Shortage of trained personnel
4. Public key infrastructure is mostly about policies.

2. Certificate

- **Certificates** are digital documents that are used for secure authentication of communicating parties.
- A certificate binds identity information about an entity to the entity's public key for a certain validity period.
- A certificate is digitally signed by a Trusted Third Party (TTP) who has verified that the key pair actually belongs to the entity.
- Certificates can be thought of as analogous to passports that guarantee the identity of their bearers.
- **Authorities :** The trusted party who issues certificates to the identified end entities is called a **Certification Authority (CA).**
- Certification authorities can be thought of as being analogous to governments issuing passports for their citizens.

- A certification authority can be managed by an external certification service provider or the CA can belong to the same organization as the end entities.
- CAs can also issue certificates to other (sub) CAs. This leads to a tree-like **certification hierarchy.**
- The highest trusted CA in the tree is called a root CA.
- In large organizations, it may be appropriate to delegate the responsibility for issuing certificates to several different certificate authorities.
- For example, the number of certificates required may be too large for a single CA to maintain; different organizational units may have different policy requirements; or it may be important for a CA to be physically located in the same geographic area as the people to whom it is issuing certificates.
- The X.509 standard includes a model for setting up a hierarchy of the Certification Authority.
- Fig. 5.13.7 shows the hierarchy of certificate authorities.

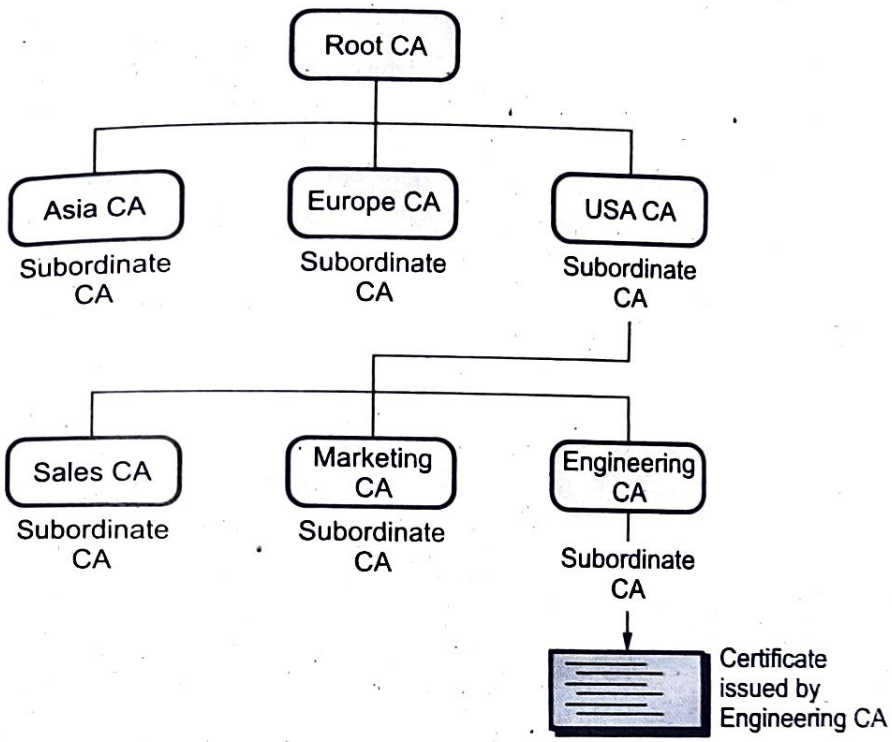


Fig. 5.13.7 Hierarchy of CA

- In the Fig. 5.13.7, the root CA is at the top of the hierarchy. The root CA's certificate is a self-signed certificate : that is, the certificate is digitally signed by the same entity.
- The CAs, that are directly subordinate to the root CA, have CA certificates signed by the root CA. CAs under the subordinate CAs in the hierarchy have their CA certificates signed by the higher-level subordinate CAs.

- Organizations have a great deal of flexibility in terms of the way they set up their CA hierarchies.
- **Certificate chains :** Certificate chain is series of certificates issued by successive CAs.
- In some cases, a CA can delegate the actual identification of end entities as well as some other administrative tasks to a separate entity, the **Registration Authority (RA).**

Verifying certificates

- When authentication is required, the entity presents a signatures it has generated from authentication data using its private key, and a certificate corresponding to that key.
- The receiving entity can verify the signature with the public key of the sender contained in the certificate.
- Next the receiving entity must verify the certificate itself by checking the validity time of the certificate and the signature of the CA in the certificate.
- If the CA is a sub CA, the receiving entity must also verify the signatures of all higher-level CAs up to the root CA.
- The list of certificates needed for verification is called a **certification path.**
- If all signatures are valid and the receiving entity trusts the root CA, the first entity will be authenticated successfully.
- If a private key of an end entity is compromised or the right to authenticate with a certificate is lost before its natural expiration date, the CA must revoke the certificate and inform all PKI users about this.
- The CA will periodically publish a **certificate revocation list (CRL).**
- The CRL is a list identifying the revoked certificates and it is signed by the CA.
- The end entities should check the latest CRL whenever they are verifying a validity of a certificate.

3. Key Length and Encryption Strength

- The strength of encryption depends on both the cipher used and the length of the key.
- Encryption strength is often described in terms of the size of the keys used to perform the encryption : in general, longer keys provide stronger encryption.
- Key length is measured in bits. For example, 128-bit keys for use with the RC4 symmetric-key cipher supported by SSL provide significantly better cryptographic protection than 40-bit keys for use with the same cipher.

- Roughly speaking, 128-bit RC4 encryption is 3×10^{26} times stronger than 40-bit RC4 encryption.
- Different ciphers may require different key lengths to achieve the same level of encryption strength.
- The RSA cipher used for public-key encryption, for example, can use only a subset of all possible values for a key of a given length, due to the nature of the mathematical problem on which it is based.
- Other ciphers, such as those used for symmetric key encryption, can use all possible values for a key of a given length, rather than a subset of those values.
- Thus a 128-bit key for use with a symmetric key encryption cipher would provide stronger encryption than a 128-bit key for use with the RSA public-key encryption cipher.

5.13.4 Key Distribution

- For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others. **Key distribution** refers to the means of delivering a key to two parties who wish to exchange data, without allowing others to see the key.
- For two parties A and B, key distribution can be achieved in a number of ways, as follows.
  1. User A can select a key and physically deliver it to user B.
  2. A third party can select the key and physically deliver it to user A and user B.
  3. If user A and user B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
  4. If user A and user B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to user A and user B.
- For manual delivery of key, **options 1 and 2** are used. These options are suitable for **link encryption.**
- Option 3 is suitable for link encryption or end-to-end encryption.
- For end-to-end encryption, some variation on option 4 has been widely adopted.
- The use of a key distribution center is based on the use of a hierarchy of keys. Minimum two levels of keys are used. Fig. 5.13.8 shows the use of a key hierarchy.

- Organizations have a great deal of flexibility in terms of the way they set up their CA hierarchies.
- **Certificate chains :** Certificate chain is series of certificates issued by successive CAs.
- In some cases, a CA can delegate the actual identification of end entities as well as some other administrative tasks to a separate entity, the **Registration Authority (RA).**

Verifying certificates

- When authentication is required, the entity presents a signatures it has generated from authentication data using its private key, and a certificate corresponding to that key.
- The receiving entity can verify the signature with the public key of the sender contained in the certificate.
- Next the receiving entity must verify the certificate itself by checking the validity time of the certificate and the signature of the CA in the certificate.
- If the CA is a sub CA, the receiving entity must also verify the signatures of all higher-level CAs up to the root CA.
- The list of certificates needed for verification is called a **certification path.**
- If all signatures are valid and the receiving entity trusts the root CA, the first entity will be authenticated successfully.
- If a private key of an end entity is compromised or the right to authenticate with a certificate is lost before its natural expiration date, the CA must revoke the certificate and inform all PKI users about this.
- The CA will periodically publish a **certificate revocation list (CRL).**
- The CRL is a list identifying the revoked certificates and it is signed by the CA.
- The end entities should check the latest CRL whenever they are verifying a validity of a certificate.

3. Key Length and Encryption Strength

- The strength of encryption depends on both the cipher used and the length of the key.
- Encryption strength is often described in terms of the size of the keys used to perform the encryption : in general, longer keys provide stronger encryption.
- Key length is measured in bits. For example, 128-bit keys for use with the RC4 symmetric-key cipher supported by SSL provide significantly better cryptographic protection than 40-bit keys for use with the same cipher.

- Roughly speaking, 128-bit RC4 encryption is 3×10^{26} times stronger than 40-bit RC4 encryption.
- Different ciphers may require different key lengths to achieve the same level of encryption strength.
- The RSA cipher used for public-key encryption, for example, can use only a subset of all possible values for a key of a given length, due to the nature of the mathematical problem on which it is based.
- Other ciphers, such as those used for symmetric key encryption, can use all possible values for a key of a given length, rather than a subset of those values.
- Thus a 128-bit key for use with a symmetric key encryption cipher would provide stronger encryption than a 128-bit key for use with the RSA public-key encryption cipher.

5.13.4 Key Distribution

- For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others. **Key distribution** refers to the means of delivering a key to two parties who wish to exchange data, without allowing others to see the key.
- For two parties A and B, key distribution can be achieved in a number of ways, as follows.
  1. User A can select a key and physically deliver it to user B.
  2. A third party can select the key and physically deliver it to user A and user B.
  3. If user A and user B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
  4. If user A and user B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to user A and user B.
- For manual delivery of key, **options 1 and 2** are used. These options are suitable for **link encryption.**
- Option 3 is suitable for link encryption or end-to-end encryption.
- For end-to-end encryption, some variation on option 4 has been widely adopted.
- The use of a key distribution center is based on the use of a hierarchy of keys. Minimum two levels of keys are used. Fig. 5.13.8 shows the use of a key hierarchy.

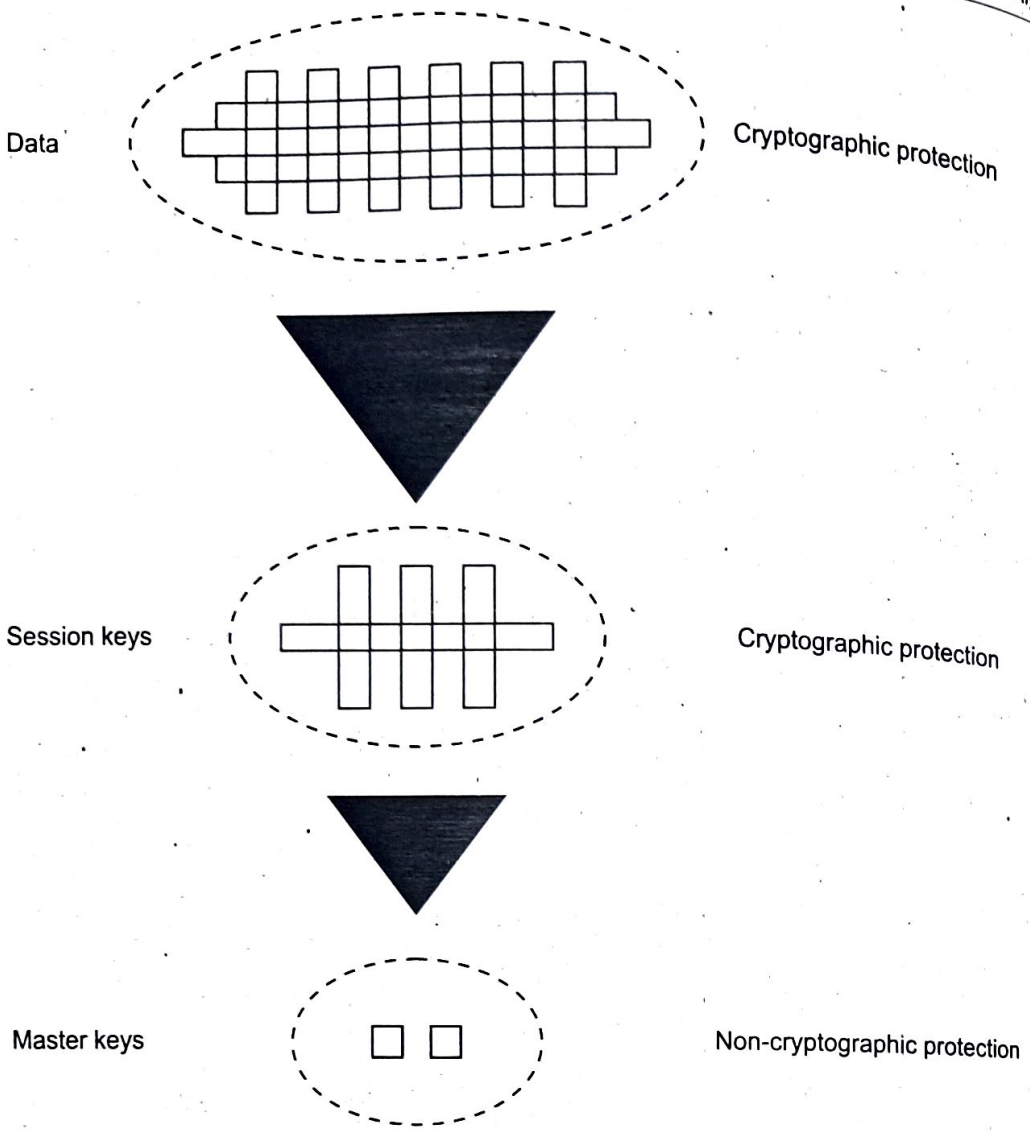


Fig. 5.13.8 Use of a key hierarchy

- Communication between end systems is encrypted using a temporary key, often referred to as a **session key**. The **session key** is used for the duration of a logical connection, such as a frame relay connection, or transport connection and then discarded.
- Session keys are transmitted in encrypted form, using a **master key** that is shared by the key distribution center and an end system or user. For each end user, there is a unique master key that it shares with the key distribution center.

A key distribution scenario

- User A wishes to establish a logical connection with user B and requires a one time session key to protect the data transmitted over the connection. User A has a master key (K_a), known only to itself and the KDC. User B shares the master key K_b with the KDC.

The following steps occur :

1. A issues a request to the KDC for a session key to protect a logical connection to B. The message includes the identity of A and B and a unique identifier (N_1) for this transaction.
2. KDC responds with a message encrypted using K_a.
3. A stores the session key for use in the upcoming session and forward to B the information that originated at the KDC for B :
4. User B sends a nonce N_2 to A.

- Fig. 5.13.9 shows the key distribution scenario.
- Steps 1, 2 are used for key distribution and steps 3, 4, 5 for authentication.

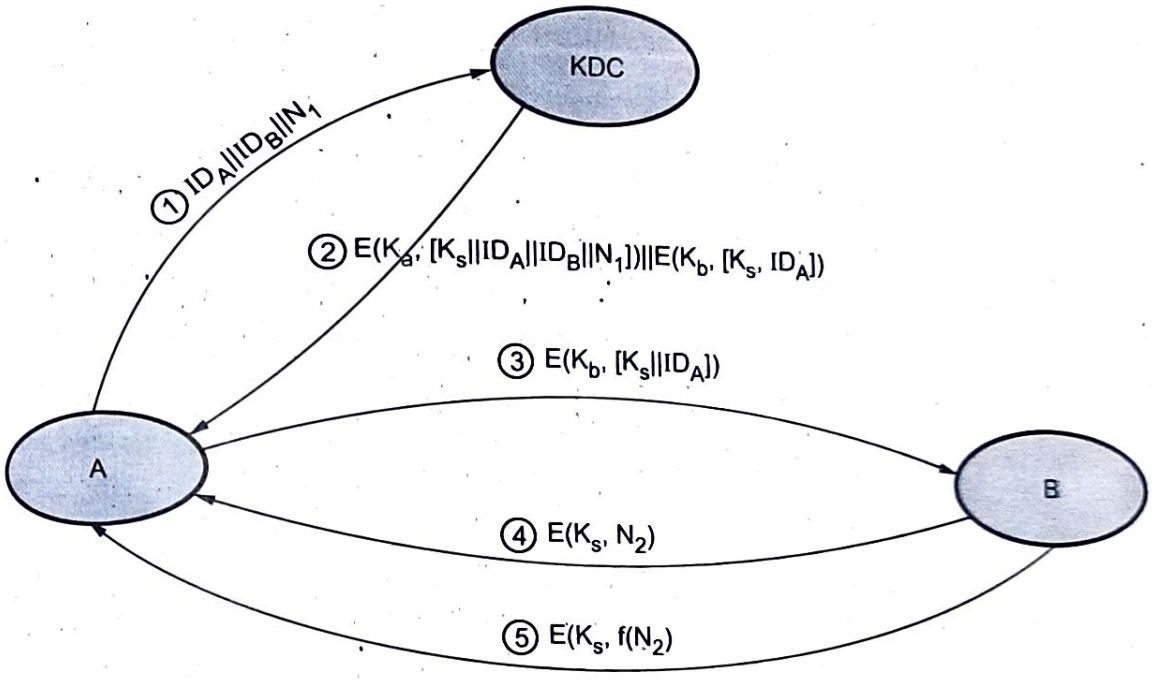


Fig. 5.13.9 Key distribution scenario

Session key lifetime

1. **For connection-oriented protocol**

- Use the same session key for the length of time that the connection is open. Use new session key for each new session.
- For long lifetime, change the session key periodically.

2. **For connectionless protocol**

- The most secure approach is to use a new session key for each exchange. For connectionless protocol, such as a transaction - oriented protocol, there is no explicit connection initiation or termination.

Transparent key control scheme

- Fig. 5.13.10 shows automatic key distribution for connection - oriented protocol.

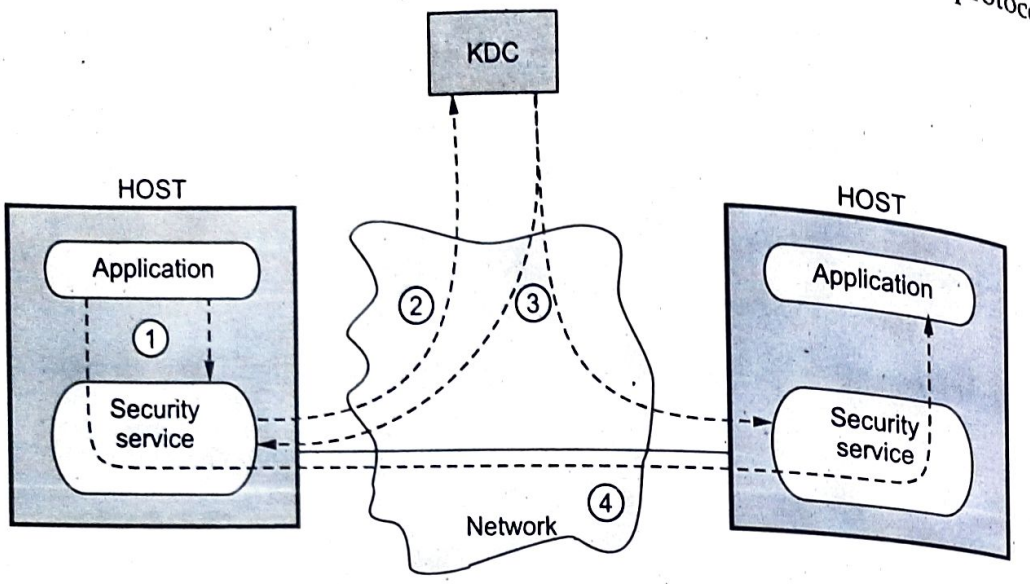


Fig. 5.13.10 Automatic key distribution for connection - oriented protocol

- Assume that communication make use of a connection-oriented end-to-end protocol, such as TCP.
- Following steps occurs :
  1. Host sends packet requesting connection.
  2. Session Security Module (SSM) saves that packet and applies to the KDC for permission to establish the connection.
  3. KDC distributes session key to both hosts.
  4. The requesting SSM can now release the connection request packet, and a connection is set up between the two end systems.

Decentralized key control

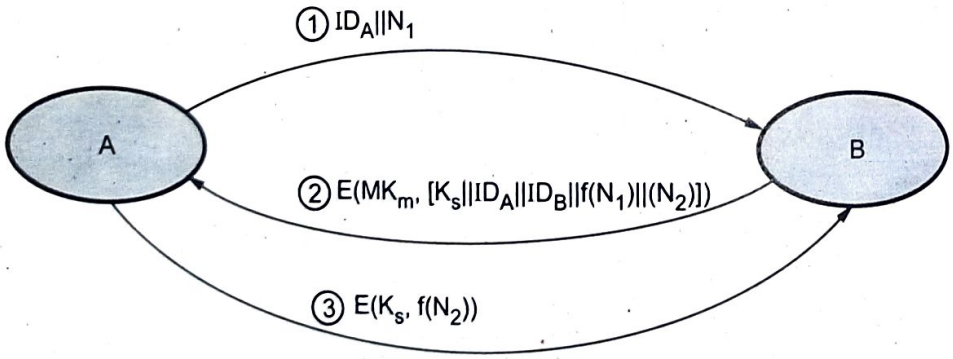


Fig. 5.13.11 Decentralized key distribution

- Decentralized approach requires that each end system be able to communicate in a secure manner with all potential parter end systems for purposes of session key distribution.
- A session key may be established with the following sequence of steps.

1. A issues a request to B for a session key and includes a none, N_1.
2. B responds with a message that is encryped using the shared master key.
3. Using the new session key, A returns f (N_2) to B.

5.14 PKIX Model

- Management and handling of the pieces of secret information is generally referred to as **key management.**
- Activities of key management includes selection, exchange, storage, certification, revocation, changing, expiration and transmission of the key.
- Key management is the set of processes and mechanisms which support key establishment and maintenance of ongoing keying relationship between parties, including replacing older key with new keys.
- Two major issues in key management are :
  1. Key life time
  2. Key exposure

Key life time - limit of use which can be measured as a duration of time.

Issue related to key :

1. Users must be able to obtain securely a key pair suited to their efficiency and security needs.
2. Keys need to be valid only until a specified expiration date.
3. The expiration date must be chosen properly and publicized securely.
4. User must be able to store their private keys securely.
5. Certificates must be unforgettable, obtainable in a secure manner.

1. Public Key Infrastructure

- Public Key Infrastructure (PKI) is a well-known technology that can be used to establish identities, encrypt information and digitally sign documents.
- PKI identifies and manages relationships of parties in an electronic exchange, serving a wide array of security needs including access control, confidentiality, integrity, authentication and non-repudiation.

- PKI also uses unique Digital Certificates (DC) to secure eCommerce, email, data exchange and Virtual Private Networks (VPN) and intranets and is also used to verify the identity and privileges of each user.
- The Certificate Authority (CA) provides a full life-cycle management of public keys and certificates, including issuance, authentication, storage, retrieval, backup, recovery, updating and revocation to the PKI.
- All users of PKI must have a registered identity, which is stored in a digital certificate issued by a CA.
- Remote users and sites using public private keys and public key certificates can authenticate each other with a high degree of confidence.
- Authentication is dependent on three conditions :

1. It must be established that each party have a private key that has not been stolen or copied from the owner.
2. The certificate must be issued to the owner in accord with the stated policy of the certificate issuer.
3. The policies of the certificate issuer must be satisfactory to the parties so as to verify identity.

Benefits of PKI

1. Confidential communication : Only intended recipients can read files.
2. Data integrity : Guarantees files are unaltered during transmission.
3. Authentication : Ensures that parties involved are who they claim to be.
4. Non-repudiation : Prevents individuals from denying.

Limitation of PKI

The problems encountered deploying a PKI can be categorized as follows :

1. Public key infrastructure is new
2. Lack of standards
3. Shortage of trained personnel
4. Public key infrastructure is mostly about policies.

2. Certificate

- **Certificates** are digital documents that are used for secure authentication of communicating parties.
- A certificate binds identity information about an entity to the entity's public key for a certain validity period.

- A certificate is digitally signed by a Trusted Third Party (TTP) who has verified that the key pair actually belongs to the entity.
- Certificates can be thought of as analogous to passports that guarantee the identity of their bearers.
- **Authorities :** The trusted party who issues certificates to the identified end entities is called a **Certification Authority (CA).**
- Certification authorities can be thought of as being analogous to governments issuing passports for their citizens.
- A certification authority can be managed by an external certification service provider or the CA can belong to the same organization as the end entities.
- CAs can also issue certificates to other (sub) CAs. This leads to a tree-like **certification hierarchy.**
- The highest trusted CA in the tree is called a root CA.
- In large organizations, it may be appropriate to delegate the responsibility for issuing certificates to several different certificate authorities.
- For example, the number of certificates required may be too large for a single CA to maintain; different organizational units may have different policy requirements; or it may be important for a CA to be physically located in the same geographic area as the people to whom it is issuing certificates.

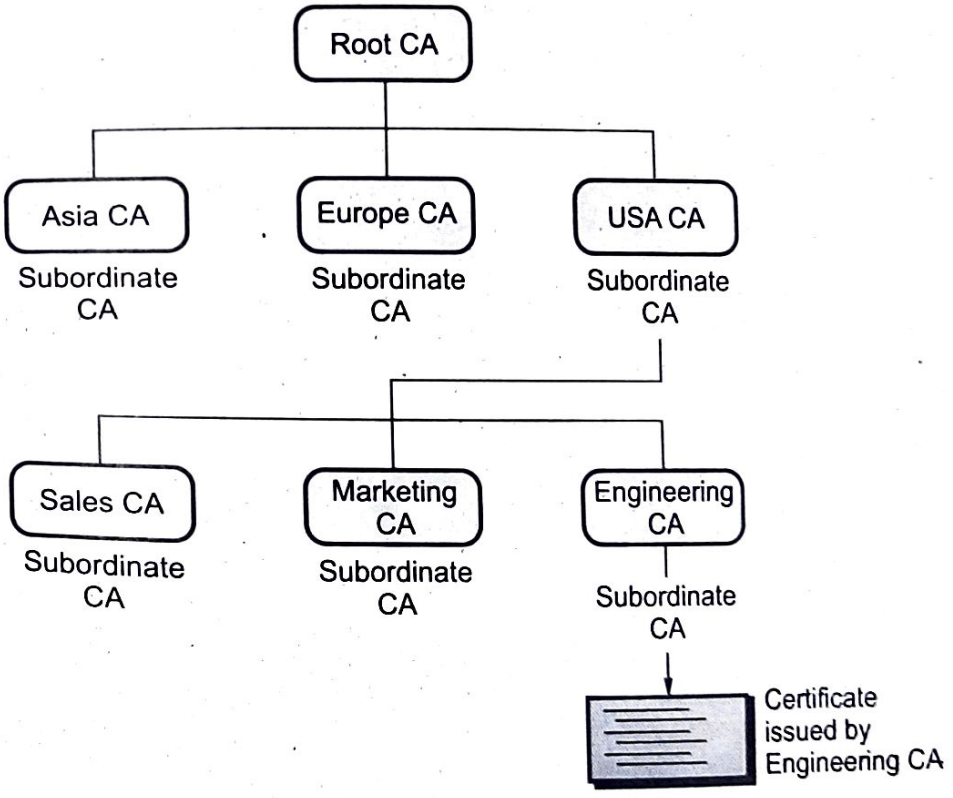


Fig. 5.14.1 Hierarchy of CA

- The X.509 standard includes a model for setting up a hierarchy of the Certification Authority.
- Fig. 5.14.1 shows the hierarchy of certificate authorities. (See Fig. 5.14.1 on previous page.)
- In the Fig. 5.14.1, the root CA is at the top of the hierarchy. The root CA's certificate is a self-signed certificate : that is, the certificate is digitally signed by the same entity – the root CA.
- The CAs, that are directly subordinate to the root CA, have CA certificates signed by the root CA. CAs under the subordinate CAs in the hierarchy have their CA certificates signed by the higher-level subordinate CAs.
- Organizations have a great deal of flexibility in terms of the way they set up their CA hierarchies.
- **Certificate chains** : Certificate chain is series of certificates issued by successive CAs.
- In some cases, a CA can delegate the actual identification of end entities as well as some other administrative tasks to a separate entity, the **Registration Authority (RA).**

Verifying certificates

- When authentication is required, the entity presents a signature it has generated from authentication data using its private key, and a certificate corresponding to that key.
- The receiving entity can verify the signature with the public key of the sender contained in the certificate.
- Next the receiving entity must verify the certificate itself by checking the validity time of the certificate and the signature of the CA in the certificate.
- If the CA is a sub CA, the receiving entity must also verify the signatures of all higher-level CAs up to the root CA.
- The list of certificates needed for verification is called a **certification path.**
- If all signatures are valid and the receiving entity trusts the root CA, the first entity will be authenticated successfully.
- If a private key of an end entity is compromised or the right to authenticate with a certificate is lost before its natural expiration date, the CA must revoke the certificate and inform all PKI users about this.
- The CA will periodically publish a **Certificate Revocation List (CRL).**
- The CRL is a list identifying the revoked certificates and it is signed by the CA.

- The end entities should check the latest CRL whenever they are verifying a validity of a certificate.

3. Key Length and Encryption Strength

- The strength of encryption depends on both the cipher used and the length of the key.
- Encryption strength is often described in terms of the size of the keys used to perform the encryption : in general, longer keys provide stronger encryption.
- Key length is measured in bits. For example, 128-bit keys for use with the RC4 symmetric-key cipher supported by SSL provide significantly better cryptographic protection than 40-bit keys for use with the same cipher.
- Roughly speaking, 128-bit RC4 encryption is 3×10^{26} times stronger than 40-bit RC4 encryption.
- Different ciphers may require different key lengths to achieve the same level of encryption strength.
- The RSA cipher used for public-key encryption, for example, can use only a subset of all possible values for a key of a given length, due to the nature of the mathematical problem on which it is based.
- Other ciphers, such as those used for symmetric key encryption, can use all possible values for a key of a given length, rather than a subset of those values.
- Thus a 128-bit key for use with a symmetric key encryption cipher would provide stronger encryption than a 128-bit key for use with the RSA public-key encryption cipher.

5.15 Short Answered Questions

Q.1 What is the difference between diffusion and confusion ?

Ans. : In **Diffusion** the statistical structure of the plaintext is dissipated into long range statistics of the cipher text. This is achieved by having each plaintext digit affect the value of many cipher text digits. **Confusion** seeks to make a relationship between the statistics of the cipher text and the value of the encryption key as complex as possible. Thus even if the attacker can get some handle on the statistics of the cipher text, the way in which the key was used to produce that cipher text is so complex as to make it difficult to deduce the key.

Q.2 What is a brute force attack ?

Ans. : A brute force attack consists of trying every possible code, combination or password until you find the right one.

Q.3 What is DES ?

Ans. : DES is a symmetric cipher defined in Federal Information Processing (FIPS) Standard Number 46 in 1977 as the federal government approved encryption algorithm for sensitive but non-classified information. DES utilizes a 56-bit key. This key size is vulnerable to a brute force attack using current technology.

Q.4 What are the ECB and CBC modes ?

Ans. : When we use a block cipher to encrypt a message of arbitrary length, we use techniques that are known as modes of operation for the block cipher. In ECB mode, each plaintext block is encrypted independently with the block cipher. ECB mode is as secure as the underlying block cipher. In CBC mode, each plaintext block is exclusive-ORed with previous ciphertext block, then encrypted.

Q.5 What are the CFB and OFB modes ?

Ans. : The Cipher Feedback (CFB) mode and the Output Feedback (OFB) mode are two more standard modes of operation for a block cipher. In **CFB** mode, the previous ciphertext block is encrypted and the output produced is combined with the plaintext block using exclusive-or to produce the current ciphertext block. It is possible to define CFB mode so that it uses feedback that is less than one full data block. **OFB** mode is similar to the CFB mode except that the quantity exclusive-ORed with each plaintext block is generated independently of both the plaintext and ciphertext. The encryption of a plaintext block is derived by takint the exclusive-OR of the plaintext block with the relevant data block.

Q.6 List out the ingredients of public key encryption scheme.

Ans. : Ingredients of public key encryptions are :

a) Plaintext b) Encryption algorithm

c) Public key d) Private key

e) Cipher-text f) Decryption algorithm

Q.7 What is the difference between statistical randomness and unpredictability ?

Ans. : In applications such as reciprocal authentication and session key generation the requirement is not so much that the sequence of numbers be statistically random but that the successive numbers of the sequence are unpredictable. With true random sequences each number is statistically independent of other numbers in the sequence and therefore unpredictable.

Q.8 What is the difference between a strong and a weak collision resistance ?

Ans. :

- For any given value h it is computationally infeasible to find y = x with H(y) = H(x). This is "weak collision resistance". It is a one-way property. It is easy to generate a code given a message, but almost impossible to do the reverse.

- It is computationally infeasible to find any pair (x, y) such that H(x) = H(y). This is "strong collision resistance". This guarantees that an alternative message hashing to the same value as a given message cannot be found. This prevents forgery.

Q.9 What are the different modes of operation in DES ?

Ans. : DES modes of operation :

1. Electronic Codebook Book (ECB) : Message is broken into independent blocks of 64 bits.
2. Cipher Block Chaining (CBC) : Message is broken in independent blocks of 64 bits, but next input depends on previous output.
3. Cipher FeedBack (CFB) : The message is XORed with the feedback of encrypting the previous block.
4. Output Feedback : The feedback is independent of the message

Q.10 What types of attacks are addressed by DES algorithm ?

Ans. : Timing attacks : Attacks actual implementation of cipher. Use knowledge of consequences of implementation to derive knowledge of some/all sub key bits. **Analytic attacks :** These utilize some deep structure of the cipher by gathering information about encryptions. It can eventually recover some/all of the sub-key bits and if necessary then exhaustively search for the rest.

Q.11 How do you determine that a given number is prime or not ?

Ans. : A positive integer is a prime if and only if it is exactly divisible by two integers, 1 and itself.

Q.12 Write down the purpose of the S-boxes in DES ?

Ans. : In S-box, each row defines a general reversible substitution. It consists of a set of 8 S-boxes, each of which accepts 6 bits as input and produces 4 bits as output.

Q.13 Define : Diffusion.

Ans. : Diffusion is the statistical structure of the plain text is dissipated into long-range statistics of the cipher text. This is possible by using permutation.

Q.14 Define : Replay attack.

Ans. : A replay attack is one in which an attacker obtains a copy of an authenticated packet and later transmits it to the intended destination. Each time a packet is send the sequence number is incremented.

Q.15 List out the parameters of AES.

Ans. : Parameters of AES are security, cost and algorithm and implementation characteristics.

Q.16 Distinguish between differential and linear cryptanalysis.

Ans. : In differential cryptanalysis, it breaks the DES in less 2^{55} complexities. In cryptanalysis, it finds the DES key given 2^{47} plaintexts.

Q.17 Write down the difference between the public key and private key cryptosystems.

Ans. : Private key cryptosystems : Same algorithm and same key is used for encryption and decryption. Sender and receiver must share the algorithm and key. Key must be kept secret.

Public-key cryptosystems : One algorithm is used for encryption and decryption with pair of keys. The sender and receiver must each have one of the matched pair of keys. One of two keys must be kept secret.

Q.18 What are the disadvantages of double DES ?

Ans. : Double DES suffer from Meet-in-the-middle attack.

Q.19 What is the disadvantages with ECB mode of operation ?

Ans. : Disadvantages :

a. Synchronization error is unrecoverable b. Not suitable for lengthy messages.

Q.20 What is weak collision resistance ? What is the use of it ?

Ans. : Weak collision-resistance : Given an x and h(x), it is infeasible to find x` such that h(x) = h(x`). This implies that given h(x), it is infeasible to find any x` such that h(x) = h(x`).

Q.21 What are the modes of DES ?

Ans. : Five standard modes of operation :

1. Electronic Code Book (ECB) 2. Cipher Block Chaining (CBC)
3. Cipher Feedback (CFB) 4. Output Feed (OFB) 5. Counter (CTR)

Q.22 List the uses of RC4.

Ans. : RC4 has become part of some commonly used encryption protocols and standards such as WEP, WPA, TLS, Kerberos and SASL mechanism Digest MD5.

Q.23 Why random numbers are used in network security ?

Ans. : Most encryption algorithms require source of random data. Random numbers are necessary not only for generating cryptographic keys but are also needed in steps of cryptographic algorithms or protocols.

Q.24 State whether symmetric and asymmetric cryptographic algorithms need key exchange.

Ans. : Both symmetric key and asymmetric key encryption uses a key to transform data along with mathematical algorithm that cannot be reversed called cryptographic hashing.

Q.25 Point out the types of cryptanalytic attacks.

Ans. : Types of cryptanalysis attacks: Cipher text only, Known plaintext, Chosen plaintext, Chosen cipher text.

Q.26 Is it possible to use the DES algorithm to generate message authentication code ? Justify.

Ans. : Yes. It can use any block cipher chaining mode and use final block as a MAC. Data Authentication Algorithm (DAA) is a widely used MAC based on DES-CBC. Encrypt message using DES in CBC mode and send just the final block as the MAC

Q.27 State few applications of RC4 algorithm.

Ans. : RC4 is used in SSL/TLS. It is also used in WEP, the IEEE 802.11 wireless networking security standard. It can also be found in a number of other applications including email encryption products.

Q.28 What is AES cipher ?

Ans. : Advanced Encryption Standard (AES) is a symmetric key block cipher. AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. The key size can be 128,192 or 256 bits. It depends on number of rounds. The number of rounds:10 rounds for 128 bits,12 rounds for 192 bits, and 14 rounds for 256 bits.

Q.29 Brief the strengths of triple DES.

Ans. : a) Strength for triple DES is actually 168 bits.
b) Brute force search impossible on triple DES.
c) It uses 2 or 3 keys.

Q.30 Give the five modes of operation of block cipher.

Ans. : Block cipher modes of operations are electronic code book, cipher block chaining mode, cipher feedback mode, counter mode and output feedback mode.

Q.31 List the parameters (block size, key size and no. of rounds) for the three AES versions.

Ans. :

AES versions	block size	key size	No. of rounds
AES-128	128 bits	128 bits	10
AES-192	128 bits	192 bits	12
AES-256	128 bits	126 bits	14

Q.16 Distinguish between differential and linear cryptanalysis.

Ans. : In differential cryptanalysis, it breaks the DES in less 2^{55} complexities. In cryptanalysis, it finds the DES key given 2^{47} plaintexts.

Q.17 Write down the difference between the public key and private key cryptosystems.

Ans. : Private key cryptosystems : Same algorithm and same key is used for encryption and decryption. Sender and receiver must share the algorithm and key. Key must be kept secret.

Public-key cryptosystems : One algorithm is used for encryption and decryption with pair of keys. The sender and receiver must each have one of the matched pair of keys. One of two keys must be kept secret.

Q.18 What are the disadvantages of double DES ?

Ans. : Double DES suffer from Meet-in-the-middle attack.

Q.19 What is the disadvantages with ECB mode of operation ?

Ans. : Disadvantages :

a. Synchronization error is unrecoverable b. Not suitable for lengthy messages.

Q.20 What is weak collision resistance ? What is the use of it ?

Ans. : Weak collision-resistance : Given an x and h(x), it is infeasible to find x` such that h(x) = h(x`). This implies that given h(x), it is infeasible to find any x` such that h(x) = h(x`).

Q.21 What are the modes of DES ?

Ans. : Five standard modes of operation :

1. Electronic Code Book (ECB) 2. Cipher Block Chaining (CBC)
3. Cipher Feedback (CFB) 4. Output Feed (OFB) 5. Counter (CTR)

Q.22 List the uses of RC4.

Ans. : RC4 has become part of some commonly used encryption protocols and standards such as WEP, WPA, TLS, Kerberos and SASL mechanism Digest MD5.

Q.23 Why random numbers are used in network security ?

Ans. : Most encryption algorithms require source of random data. Random numbers are necessary not only for generating cryptographic keys but are also needed in steps of cryptographic algorithms or protocols.

Q.24 State whether symmetric and asymmetric cryptographic algorithms need key exchange.

Ans. : Both symmetric key and asymmetric key encryption uses a key to transform data along with mathematical algorithm that cannot be reversed called cryptographic hashing.

Q.25 Point out the types of cryptanalytic attacks.

Ans. : Types of cryptanalysis attacks: Cipher text only, Known plaintext, Chosen plaintext, Chosen cipher text.

Q.26 Is it possible to use the DES algorithm to generate message authentication code ? Justify.

Ans. : Yes. It can use any block cipher chaining mode and use final block as a MAC. Data Authentication Algorithm (DAA) is a widely used MAC based on DES-CBC. Encrypt message using DES in CBC mode and send just the final block as the MAC

Q.27 State few applications of RC4 algorithm.

Ans. : RC4 is used in SSL/TLS. It is also used in WEP, the IEEE 802.11 wireless networking security standard. It can also be found in a number of other applications including email encryption products.

Q.28 What is AES cipher ?

Ans. : Advanced Encryption Standard (AES) is a symmetric key block cipher. AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. The key size can be 128,192 or 256 bits. It depends on number of rounds. The number of rounds:10 rounds for 128 bits,12 rounds for 192 bits, and 14 rounds for 256 bits.

Q.29 Brief the strengths of triple DES.

Ans. : a) Strength for triple DES is actually 168 bits.
b) Brute force search impossible on triple DES.
c) It uses 2 or 3 keys.

Q.30 Give the five modes of operation of block cipher.

Ans. : Block cipher modes of operations are electronic code book, cipher block chaining mode, cipher feedback mode, counter mode and output feedback mode.

Q.31 List the parameters (block size, key size and no. of rounds) for the three AES versions.

Ans. :

AES versions	block size	key size	No. of rounds
AES-128	128 bits	128 bits	10
AES-192	128 bits	192 bits	12
AES-256	128 bits	126 bits	14

Q.32 Compare DES and AES.

Ans. :

AES	DES
AES stands for Advanced Encryption Standard	DES stands for Data Encryption Standard
Key length can be of 128-bits, 192-bits and 256-bits.	Key length is 56 bits in DES.
Number of rounds depends on key length : 10(128-bits), 12(192-bits) or 14(256-bits)	DES involves 16 rounds of identical operations
The structure is based on substitution permutation network.	The structure is based in feistal network.
AES is more secure than the DES cipher and is the de facto world standard.	DES can be broken easily as it has known vulnerabilities. 3 DES(Triple DES) is a variation of DES which is secure than the usual DES.
The rounds in AES are : Byte Substitution, Shift Row, Mix Column and Key Addition.	The rounds in DES are : Expansion, XOR operation with round key.

Q.33 Why is trap door one way function used ?

Ans. : A trapdoor one way function is a function that is easy to compute in one direction, yet difficult to compute in the opposite direction (finding its inverse) without special information, called the "trapdoor". Trapdoor functions are widely used in cryptography.

Q.34 Define field and ring in number theory.

Ans. : Field - A field, denoted by F = <{...}, •, > is a commutative ring in which the second operation satisfies all five properties defined for the first operation except that the identity of the first operation has no inverse.

- A Field supports two pairs of operations : addition / subtraction and multiplication / division, except that the division by zero is not allowed.

Ring : A ring, R = <{...}, •, >, is an algebraic structure with two operations. The second operation must be distributed over the second.

5.16 Multiple Choice Questions

Q.1 Password based authentication can be divided into two broad categories : _____ and _____

a Fixed, Variable

b Time stamped, fixed

c Fixed, one-time

d None of the above

Q.2 **Challenge-response authentication can be done using _____.**

a symmetric-key ciphers
b asymmetric-key ciphers
c keyed-hash functions
d all of the above

Q.3 **_____ is a popular session key creator protocol that requires an authentication server and a ticket granting server.**

a KDC
b Kerberos
c CA
d none of the above

Q.4 **For each _____ the Kerberos Key Distribution Center (KDC) maintains a database of the realm's principal and the principal's associated "secret keys".**

a key
b realm
c document
d none of the mentioned

Q.5 **RSA is a _____ cipher in which the plaintext and ciphertext are integers between 0 and n – 1 for some n.**

a block
b stream
c private
d none

Q.6 **Public key encryption is also known as _____ key encryption.**

a symmetric
b private
c asymmetric
d none

Q.7 **One commonly used public-key cryptography method is the _____ algorithm.**

a RSS
b RAS
c RSA
d RAA

Q.8 **The _____ is the message after transformation.**

a ciphertext
b plaintext
c secret-text
d none

Q.9 **The _____ method provides a one-time session key for two parties.**

a RSA
b Diffie-Hellman
c DES
d AES

Answer Keys for Multiple Choice Questions :

Q.1	c	Q.2	d	Q.3	b
Q.4	b	Q.5	a	Q.6	c
Q.7	c	Q.8	a	Q.9	b

UNIT VI

6 Introduction to Cyber Security

Syllabus

Basic Cyber Security Concepts, Layers of security, Vulnerability, Threat, Harmful Acts-Malware, Phishing, MIM Attack, DOS Attack, SQL Injection, Internet Governance - Challenges and Constraints, Computer Criminals, Assets and Threat, Motive of Attackers, Software attacks, hardware attacks, Cyber Threats-Cyber Warfare, Cyber Crime, Cyber Stalking, Cyber Terrorism, Cyber Espionage, Comprehensive Cyber Security Policy.

Contents