Contents

CRYPTOGRAPHY & INFORMATION SECURITY

UNIT 1

MATHEMATICAL BACKGROUND FOR CRYPTOGRAPHY – ABSTRACT ALGEBRA, NUMBER THEORY, MODULAR INVERSE, EXTENDED EUCLID ALGORITHM, FERMAT'S LITTLE THEOREM, EULER PHI-FUNCTION, EULER'S THEOREM

Q.1. Define abstract algebra.

Ans. A non-empty set G equipped with one or more binary operations is said to be an ***algebraic structure.*** Suppose * is a binary operation on G. Then (G, *) is an algebraic structure. (N, +), (I, +), (I, –), (R, +, ·) are all algebraic structure. Here (R, +, .) is an algebraic structure equipped with two operations. Some algebraic structures are group, ring and field.

Q.2. Write properties of an algebraic system.

Ans. By a property of an algebraic system, we mean a property possessed by any of its operations. Important properties of an algebraic system are –

(i) Associative and Commutative Laws – An operation * on a set S is said to be associative or to satisfy the associative law if, for any elements a, b, c in S, we have

$$(a * b) * c = a * (b * c)$$

An operation * on a set S is said to be ***commutative*** or satisfy the commutative law if

$$a * b = b * a$$

For any elements, a, b in S.

(ii) Identity Element and Inverses – Consider an operation * on a set S. An element e in S is called an identity element for * if, for any element a in S,

$$a * e = e * a = a$$

Generally, an element e is called a left identity or a right identity according as e * a = a or a * e = a where a is any element in S.

Suppose an operation * on a set S does have an identity element e. The inverse of an element in S is an element b such that

$$a * b = b * a = e$$

(iii) Cancelation Laws – An operation * on a set S is said to satisfy the left cancelation law if

$$a * b = a * c \text{ implies } b = c$$

and is said to satisfy the right cancelation law if

$$b * a = c * a \text{ implies } b = c$$

Q.3. What is group ?

Ans. A system consisting of a non-empty set G of elements a, b, c etc., with an operation is said to be a ***group*** provided the following postulates are satisfied –

(i) Closure Property – For all $a, b \in G \Rightarrow a.b \in G$

i.e., G is closed under the operation '.'.

(ii) Associativity – $(a.b).c = a.(b.c), \forall\, a, b, c \in G$

i.e., the binary operation '.' over G is associative.

(iii) Existence of Identity – There exists an unique element in G, such that, $e.a = a = a.e$, for every $a \in G$. This element e is called the ***identity***.

(iv) Existence of Inverse – For each $a \in G$, there exists an element $a^{-1} \in G$, such that $a.a^{-1} = e = a^{-1}.\,a$.

The element a^{-1} is called the inverse of a.

Abelian or Commutative Group – A group G is said to be ***abelian*** or ***commutative*** if in addition to the above four postulates the following postulate is also satisfied.

(v) Commutativity – $a.b = b.a$, for every $a, b \in G$.

Q.4. Define the ring with example.

Ans. **Definition 1** – An algebraic structure $(R, +, \cdot)$ consisting of a non-empty set R and two binary operation, called addition (+) and multiplication (·) is called a ***ring*** provided the following postulates are satisfied –

R_1 – The system (R, +) is an abelian group. So we have the following properties –

(i) Closure Property – The set R is closed with respect to the composition +,

i.e., $a \in R, b \in R \Rightarrow a + b \in R, \forall\, a, b \in R$

(ii) Associativity – Associative law holds good in the set R for the composition +

i.e., $(a + b) + c = a + (b + c), \forall\, a, b, c \in R$

(iii) Existence of Identity (or Zero) – There exists an unique $0 \in R$ (called zero element) such that

$$a + 0 = a = 0 + a, \forall\, a \in R$$

(iv) Existence of Inverse (or negative) – For each $a \in R$, there exists an element $-a \in R$, such that $a + (-a) = 0 = (-a) + a$.

(v) Commutative of Addition – Commutative law holds good in the set R for the composition +

i.e., $a + b = b + a, \forall\, a, b \in R$

R_2 – The set R is closed with respect to the multiplication composition.

i.e., $a.b \in R, \forall\, a, b \in R$.

R_3 – Multiplication composition is associative i.e., $(a.b).c = a.(b.c), \forall\, a, b, c \in R$.

R_4 – The multiplication composition is right and left distributive with respect to addition.

i.e., $a.(b + c) = a.b + a.c., \forall\, a, b, c \in R$ (left distributive law)

and $(b + c).a = b.a + c.a$ (right distributive law).

Definition 2 – An algebraic (or mathematical) system (R, *, o) consisting of a non-empty set R any two binary operations * and o defined on R such that

(i) (R, *) is an abelian group;

(ii) (R, o) is a semigroup and

(iii) the operation o is distributive over the operation * is said to be the ***ring***.

Q.5. Write short note on fields.

Ans. A ring R with at least two elements is called a ***field*** if,

(i) it is commutative

(ii) it has unity

(iii) it is such that each non-zero element possesses multiplicative inverse.

Example – The ring of rational numbers $(Q, +, \cdot)$ is a field since it is a commutative ring with unity and each non-zero element is inversible.

Skew Field – A ring R with at least two elements is called a ***division ring*** or a ***skew field*** if it (i) has unity, (ii) is such that each non-zero element possesses multiplicative inverse.

Q.6. Define number theory.

Ans. We will be outlining several topics from number theory which we will need in order to explore the mathematics behind the cryptography.

Definition (i) – Suppose we have two integers a and b with $a \neq 0$. If an integer c exists so that we can write $b = ac$ then we say that a divides b, or that a is a divisor of b. Often we will use the notation $a \mid b$ which is read as "a divides b".

Suppose we have two integers a and b with a common divisor $d \neq 0$. That is, $d \mid a$ and $d \mid b$, then we will have $d \mid (ra + sb)$ for any integers r and s.

Proof. Because d is a divisor of both a and b then we can write $a = dj$ and $b = dk$ for some integers j and k. Then $ra + sb = r(dj) + s(dk) = d(rj + sk)$. Since $(rj + sk)$ is an integer then it follows that $d \mid (ra + sb)$.

Proposition – Given two non-negative integers a and b, with $a \neq 0$, there exists a pair of unique integers q and r with $0 \leq r < a$ such that $b = aq + r$. We call q the quotient and r the remainder when b is divided by a.

Finding such a quotient and remainder is what we find when performing long division. After doing long division it is sometimes common to represent the ratio of b divided by a as $\frac{b}{a} = q + \frac{r}{a}$, but we can see that this statement is equivalent to the statement $b = aq + r$ given in the proposition above.

Definition (ii) – The greatest common divisor of two non-zero integers a and b is the largest integer c such that c divides both a and b. This is denoted by gcd $(a, b) = c$ or sometimes by $(a, b) = c$, however we will use the former notation in this text. If the greatest common divisor of a and b is 1 then we say that a and b are relatively prime.

There is a very useful procedure for computing the greatest common divisor of two positive integers. It is known as the Euclidean algorithm. Let us suppose we have two positive integers a and b, with $a \leq b$ and let $d = \gcd(a, b)$. By proposition we can write $b = aq_1 + r1$ with $0 \leq r_1 < a$. Then because $d \mid a$ and $d \mid b$ it follows that $d \mid r_1$, since we can write $r_1 = b - aq_1$. So then d is common divisor of a and r_1. Continuing the process, next we write $a = r_1q_2 + r_2$. Now, because $d \mid a$ and $d \mid r_1$ we have $d \mid r_2$ because we can write $r_2 = a - r_1q_2$. So similarly, we can conclude that d is also a common divisor of r_1 and r_2. Iterating this process we continue until we obtain an $r_{k+1} = 0$. Then we have that $d = r_k$. We can see that the process will terminate because the remainders are getting smaller with each iteration, but remain non-negative by definition, so eventually we must reach a remainder of zero.

It should be clear that r_k is a common divisor of a and b, but we will omit the proof that r_k is actually the greatest common divisor. Let us do a few examples to better understand this algorithm.

Example – Find gcd(522, 213). First divide 522 by 213.

$$522 = 213(2) + 96$$

Next, divide 213 by the remainder 96 and continue this process.

$$213 = 96(2) + 21$$
$$96 = 21(4) + 12$$
$$21 = 12(1) + 9$$
$$12 = 9(1) + 3$$
$$9 = 3(3) + 0.$$

So gcd(522, 213) = 3.

Definition (iii) – We say that a number p is prime if it is an integer greater than 1, whose only positive divisors are 1 and itself. An integer greater than 1 which is not prime is said to be composite.

Theorem – The fundamental theorem of arithmetic – Given an integer greater than 1 we can write that integer as a unique product of primes (up to reordering of the factors).

Example – $10 = 2.5$, since 2 and 5 are prime. $7800 = 2^3.3.5^2.13$ because 2, 3, 5, and 13 are prime. $23 = 23$ since 23 is a prime number.

Definition (iv) – For a positive integer m, which we will call our modulus, we say that two integers a and b are congruent modulo m if $m \mid (a - b)$ or equivalently if a and b have the same remainder when divided by m. Symbolically this is written as $a \equiv b \pmod m$ which is read as "a is congruent to be mode m".

Example – 23 is congruent to 3 modulo 10 since $10 \mid (23 - 3) = 20$. Also we have $59 \equiv -6 \pmod{13}$ because $13 \mid (59 - (-6)) = 65$. We find though, $7 \not\equiv 3 \pmod 5$ since $5 \mid (7 - 3) = 4$.

Q.7. Define greatest common divisor.

Ans. One integer often needed in cryptography is the greatest common divisor of two positive integers. Two positive integers may have many common divisors but only one greatest common divisor, the common divisors of 12 and 140 are 1, 2 and 4. However, the greatest common divisor is 4 as shown in fig. 1.1.

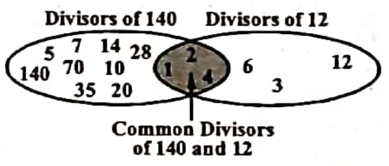


Fig. 1.1 Common Divisors of Two Integers

Q.8. Describe modulo arithmetic with its properties.

Ans. Let d be an integer and let n be a positive integer. Let q and r be the quotient and remainder obtained from dividing d by n. The relationship between d, n, q and r is

$$d = n * q + r, \qquad 0 \le r < n$$

where r is a non-negative integer less than n. d and n are the dividend and the divisor, respectively. We can say "d is equal to r modulo n" if the remainder obtained from dividing d by n is r. This is expressed as

$$r \equiv d (\text{mod } n)$$

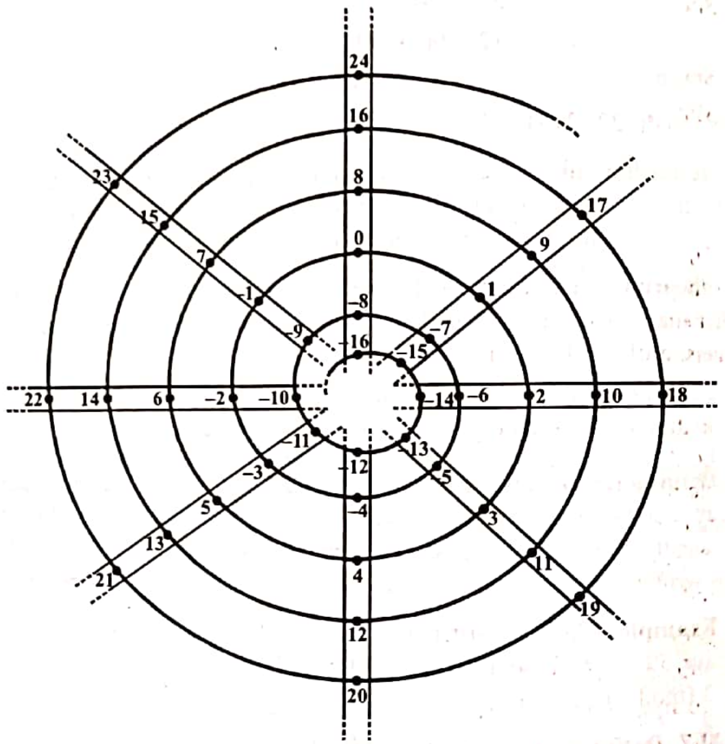


Fig. 1.2 Equivalence Classes Modulo 8

Any two numbers in the set {...., – 37, – 27, – 17, – 7, 3, 13, 23, 33, 43,...} are said to be congruent modulo 10 and the set itself is referred to as a congruence class. It is helpful to visualize the "modulo n relationship" using fig. 1.2. The integers are laid out along a spiral with n integers on a single "circle". Starting with 0, we encounter the positive integers in sequence as we traverse the spiral clockwise, while the negative numbers are encountered as we traverse the spiral in the anti-clockwise direction. The set of elements along a given radius constitute one of the congruence classes modulo n. There

are n congruence classes mod n. It is convenient to represent a class by the smallest non-negative integer in that class.

Two distinct integers, a and b, that are congruent modulo n map to the same radius in the spiral. Counting from a to b involves one or more revolutions. It follows that – if two integers are congruent modulo n, then they differ by an integral multiple of n. Algebraically, if

$$a \text{ mod } n = r \quad \text{and} \quad b \text{ mod } n = r$$

then $\quad a = n*q_1 + r \quad \text{and} \quad b = n*q_2 + r$

where q_1 and q_2 are integers.

Subtracting, we get

$$a - b = n(q_1 - q_2)$$

Since q_1 and q_2 are integers, a and b differ by an integral multipe of n.

Many useful properties of modulo arithmetic are as follows –

(i) (a + b) mod n = ((a mod n) + (b mod n)) mod n

(ii) (a – b) mod n = ((a mod n) – (b mod n)) mod n

(iii) (a * b) mod n = ((a mod n) * (b mod n)) mod n

These properties are useful in cryptography. In cryptography, we often have to perform computations such as multiplying a large number of term itself being a very large number. For example, we may have to multiply 50 integers, each about 1000 digits long.

$$(a_1 * a_2 * a_3 ..... * a_{50}) \text{ mod } n$$

In the worst case, the size of a_1*a_2 will be 2000 digits, the size of $a_1*a_2*a_3$ will be 3000 digits. Property (iii), however, tells us that we could "reduce modulo n" each intermediate product before mutiplying by the next term. For example, we could

Compute the product a_1*a_2

reduce, i.e., compute $b = (a_1*a_2)$ mod n

compute the product $b*a_3$

reduce, i.e., compute $(b*a_3)$ mod n

but, we are restricting the size of each intermediate result. In particular, if n is roughly 1000 digits, then the length of the intermediate results after a multiplication and a reduction is no more than 2000 and 1000 digits, respectively.

Q.9. What is the difference between modular arithmetic and ordinary arithmetic ? List three classes of polynomial arithmetic. (R.G.P.V., June 2013)

Ans. In ordinary arithmetic, the division relationship (a = q × n + r) has two inputs a and n and two outputs (q and r). In modular arithmetic, we are interested in only one of the outputs, the remainder r. It means that we want to know what is the value of r when we divide a by n and the division relation is a mod n = r.

Polynomial Arithmetic Classes – There are three classes of polynomial arithmetic –

(i) Ordinary polynomial arithmetic, using the basic rules of algebra.

(ii) Polynomial arithmetic in which the arithmetic on the coefficients is performed modulo p. That is, the coefficients are in Z_p.

(iii) Polynomial arithmetic in which the coefficients are in Z_p, and the polynomials are defined modulo *a* polynomial m(x) whose highest power is some integer n.

Q.10. Define the term modular inverse.

Ans. **Definition (i)** – Given an integer a and a positive integer n, satisfying gcd(a, n) = 1, we define the multiplicative inverse of a modulo n to be an integer d such that ad ≡ 1 (mod n). This d is sometimes represented symbolically by $d = a^{-1}$.

The Euclidean algorithm which we described earlier can provide a convenient way of finding multiplicative inverses modulo n. The way we can do this is by first using the algorithm to show that gcd(a, n) = 1. We then work backwards through the equations that were found in order to represent 1 = ad + nc for some integers d and c. We then will have that the multiplicative inverse of a modulo n is d. This is because if we consider the equation 1 = ad + nc modulo n then we see –

$$1 \equiv ad + nc \equiv ad + 0 \equiv ad \pmod{n}.$$

Let us do a few examples to see how this works.

Example – Find the multiplicative inverse of 9 modulo 32. First let us perform the Euclidean algorithm to show that gcd (32, 9) = 1; this is seen in the left hand column below. At each step we will also solve for the remainder in the equation, which can be seen in the right hand column below. These remainder equations are then labeled in a reverse ordering for later reference.

$$32 = 9(3) + 5 \quad \rightarrow \quad 5 = 32 - 9(3) \qquad \text{...(iii)}$$
$$9 = 5(1) + 4 \quad \rightarrow \quad 4 = 9 - 5(1) \qquad \text{...(ii)}$$
$$5 = 4(1) + 1 \quad \rightarrow \quad 1 = 5 - 4(1) \qquad \text{...(i)}$$

Now we work backwards through these equations. First we use the last equation (i) which states 1 = [5 – 4(1)]. Next we use the second to last equation (ii) to substitute 4 = [9 – 5(1)] into our previous expression. We then distribute and group our terms to obtain an expression of 1 = 9j + 5k for some integers j and k. Lastly we will replace 5 = [32 – 9(3)] (iii) and again group our terms to obtain the desired equation in terms of 32 and 9.

$$1 = [5 - 4(1)] \qquad \text{...(i)}$$
$$= 5 - [9 - 5(1)] \qquad \text{...(ii)}$$
$$= 5 - 9 + 5 \text{ distribute}$$
$$= 9(-1) + 5(2) \text{ group terms}$$

$$= 9(-1) + 2[32 - 9(3)] \qquad \text{...(iii)}$$
$$= 9(-1) + 32(2) + 9(-6) \text{ distribute}$$
$$= 32(2) + 9(-7) \text{ group terms.}$$

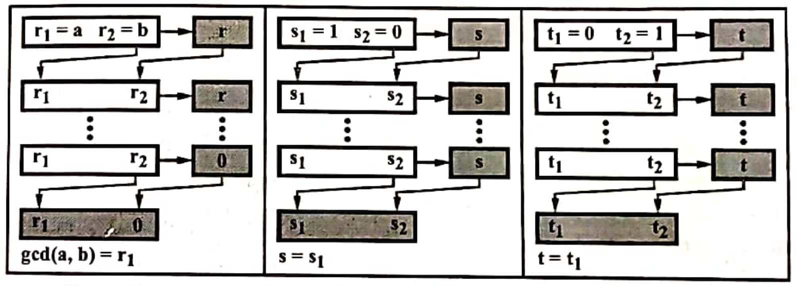
Thus we see that $9^{-1} \equiv -7 \equiv 25 \pmod{32}$. We can confirm this by checking that $9(-7) \equiv -63 \equiv 1 \pmod{32}$.

Q.11. Discuss the extended Euclidean algorithm.

Ans. Given two integers a and b, we often need to find two integers, s and t, such that

$$s \times a + t \times b = \gcd(a, b)$$

The extended Euclidean algorithm can calculate the gcd(a, b) and at the same time calculate the value of s and t. The algorithm and process is shown in fig. 1.3. The extended Euclidean algorithm uses the same number of steps as the Euclidean algorithm. However, in each step, we use three sets of calculations and exchanges instead of one. The algorithm uses three sets of variables, r's, s's and t's.



(a) Process

```
$r_1 \leftarrow a$;  $r_2 \leftarrow b$;
$s_1 \leftarrow 1$;  $s_2 \leftarrow 0$;        (Initialization)
$t_1 \leftarrow 0$;  $t_2 \leftarrow 1$;
while ($r_2 > 0$)
{
  $q \leftarrow r_1 / r_2$;
    $r \leftarrow r_1 - q \times r_2$;
    $r_1 \leftarrow r_2$; $r_2 \leftarrow r$;      (Updating r's)
    $s \leftarrow s_1 - q \times s_2$;
    $s_1 \leftarrow s_2$; $s_2 \leftarrow s$;      (Updating s's)
    $t \leftarrow t_1 - q \times t_2$;
    $t_1 \leftarrow t_2$; $t_2 \leftarrow t$;      (Updating t's)
}
gcd (a, b) $\leftarrow r_1$;  $s \leftarrow s_1$;  $t \leftarrow t_1$
```

(b) Algorithm

Fig. 1.3 Extended Euclidean Algorithm

In each step, r_1, r_2 and r have the same values in the Euclidean algorithm. The variable r_1 and r_2 are initialized to the values of a and b, respectively. The variables s_1 and s_2 are initialized to 1 and 0 respectively. The variables t_1 and t_2 are initialized to 0 and 1, respectively. The calculations of r, s and t are similar, with one warning. Although r is the remainder of dividing r_1 by r_2, there is no such relationship between the other two sets. There is only one quotient, q, which is calculated as r_1/r_2 and used for the other two calculations.

Q.12. Explain Euclidean algorithm for finding the greatest common divisor.

Ans. Finding the greatest common divisor (gcd) of two positive integers by listing all common divisor is not practical when the two integers are large. Fortunately, more than 2000 years ago a mathematician named Euclid developed an algorithm that can find the greatest common divisor of two positive integers. This algorithm is based on the following two facts –

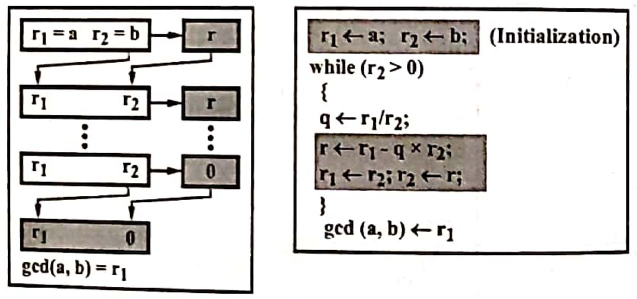
(i) gcd (a, 0) = a

(ii) gcd (a, b) = gcd (b, r) where r is the remainder of dividing a by b

The first fact tells us that if the second integers is 0, then the greatest common divisor is the first integer. The second fact allows us to change the value of a, b until b becomes 0. For example, to calculate the gcd (36, 10), we can use the second fact several times and the first fact once, as shown below.

gcd(36, 10) = gcd(10, 6) = gcd (6, 4) = gcd(4, 2) = gcd(2, 0) = 2

In other words, gcd(36, 10) = 2, gcd(10, 6) = 2, and so on. This means that instead of calculating gcd(36, 10), we can find gcd (2, 0). Fig. 1.4 shows how we use the above two facts to calculate gcd(a, b).



```
r1 ← a;  r2 ← b;
while (r2 > 0)
{
  q ← r1/r2;
  r ← r1 - q × r2;
  r1 ← r2; r2 ← r;
}
gcd (a, b) ← r1
```

(a) Process *(b) Algorithm*

Fig. 1.4 Euclidean Algorithm

We use two variables, r_1 and r_2, to hold the changing values during the process of reduction. They are initialized to a and b. In each step, we calculate the remainder of r_1 divided by r_2 and store the result in the variable r. We then replace r_1 by r_2 and r_2 by r. The steps are continued until r_2 becomes 0. At this moment, we stop. The gcd (a, b) is r_1.

Q.13. What is Euler's totient ? *(R.G.P.V., June 2016)*

Ans. For calculating the inverse modulo n, there is another method, but it is not always possible to use it. *The reduced set of residues* mod n is the subset of the complete set of residues which is relatively prime to n. For instance, the reduced set of residues mod 12 is {1, 5, 7, 11}. When n is prime, then the reduced set of residues mod n is the set of all numbers from 1 to n – 1. The number 0 is not piece of the reduced set of residues for any n not equal to 1.

The ***Euler totient*** function is also known as ***Euler phi*** function. The Euler totient function is written as $\phi(n)$. $\phi(n)$ is the number of elements in the reduced set of residues modulo n. In other words, $\phi(n)$ is the number of positive integers less than n that are relatively prime to n (for any n greater than 1). When n is prime, then $\phi(n) = n - 1$. When n = pq, where p and q are prime, then $\phi(n) = (p - 1)(q - 1)$. These numbers appear in some public key algorithms; this is why. According to ***Euler's generalization of Fermat's little theorem,*** when gcd (a, n) = 1, then

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

Now it is simple to calculate a^{-1} mod n –

$$x = a^{\phi(n) - 1} \bmod n$$

For example, what is the inverse of 5, modulo 7 ? Since 7 is prime, $\phi(7) = 7 - 1 = 6$. Hence, the inverse of 5, modulo 7, is

$$5^{6-1} \bmod 7 = 5^5 \bmod 7 = 3$$

Both techniques for computing inverses can be extended to solve for x in the general problem (if gcd(a, n) = 1) –

$$(a * x) \bmod n = b$$

Using Euler's generalization to solve –

$$x = (b * a^{\phi(n) - 1}) \bmod n$$

Using Euclid's algorithm to solve –

$$x = (b * (a^{-1} \bmod n)) \bmod n$$

Normally, Euclid's algorithm is faster as compared to Euler's generalization for computing inverses, especially for numbers in the 500 bit range. If gcd(a, n) ≠ 1, all is not lost. In this normal condition, (a * x) mod n = b, may have a lot of solutions or no solution.

Q.14. Define the Euler Phi-function.

Ans. The Euler ϕ-function, $\phi(n)$, is defined to be the number of positive integers less than or equal to n which are relatively prime to n.

Examples – Consider n = 10. We see that the only positive integers $k \le 10$ such that gcd(k, 10) = 1 are k = 1, 3, 7, 9. Thus we have $\phi(10) = 4$.

Consider n = 7. Then all positive integers strictly less than 7 are relatively prime to 7, since 7 is a prime number. So $\phi(n) = 7 - 1 = 6$.

Another example is $\phi(1) = 1$ because the only positive integer $k \le 1$ is $k = 1$ and $\gcd(1, 1) = 1$.

Proposition (i) – $\phi(n)$ is a multiplicative function. This means that if $\gcd(a, b) = 1$ then $\phi(ab) = \phi(a)\,\phi(b)$.

We can find a formula for computing $\phi(n)$, given the prime factorization of n. We do this by first considering cases where n is a prime power.

Consider $n = p$ for any prime p. Then we see that all positive integers. $k < p$ satisfy $\gcd(k, p) = 1$. There are $p - 1$ such k's thus $\phi(p) = p - 1$.

Consider $n = p^j$ where p is any prime and j is any positive integer.

The only integers which will have a common (non-trivial) factor with p^j are multiples of p. The multiples of p less than or equal to p^j are – p, 2p, 3p, ..., $(p^{j-1})p = p^j$. So we see that there are p^{j-1} such multiples. We can then conclude that the number of positive integers less than or equal to p^j which are relatively prime to p^j will be $\phi(p^j) = p^j - p^{j-1} = p^j\left(1-\frac{1}{p}\right)$.

We can now conclude the following proposition –

Proposition (ii) – For a positive integer n where $n = p_1^{a_1} p_2^{a_2} \ldots p_k^{a_k}$ we have

$$\phi(n) = n\prod_{i=1}^{k}\left(1-\frac{1}{p_i}\right)$$

Proof. Since $\phi(n)$ is multiplicative then we have

$$\phi(n) = \phi(p_1^{a_1})\,\phi(p_2^{a_2})\ldots\phi(p_k^{a_k})$$

$$= p_1^{a_1}\left(1-\frac{1}{p_1}\right)p_2^{a_2}\left(1-\frac{1}{p_2}\right)\ldots p_k^{a_k}\left(1-\frac{1}{p_k}\right)$$

$$= p_1^{a_1}p_2^{a_2}\ldots p_k^{a_k}\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\ldots\left(1-\frac{1}{p_k}\right)$$

$$= n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\ldots\left(1-\frac{1}{p_k}\right)$$

$$= n\prod_{i=1}^{k}\left(1-\frac{1}{p_i}\right).$$

An alternative way of representing this product is to write $\phi(n) = n\prod_{p|n}\left(1-\frac{1}{p}\right)$, where it is understood that this will mean to index the product over all prime divisors of n.

Q.15. Explain the Euler's theorem.

Ans. For two positive integers n and m which are relatively prime we have that .

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

where $\phi(n)$ is the Euler ϕ-function.

Examples – Let n = 10. We saw earlier than $\phi(10) = 4$. Consider m = 3. We have that $\gcd(3, 10) = 1$ and we can confirm that $3^4 \equiv 81 \equiv 1 \pmod{10}$.

Let n = 7 and m = 2. Then $\phi(7) = 7 - 1 = 6$, and we have $\gcd(2, 7) = 1$. We can confirm that $2^6 \equiv (2^3)^2 \equiv (8)^2 \equiv (1)^2 \equiv 1 \pmod{7}$.

It is interesting to point out that Euler's theorem is a generalization of Fermat's Little theorem, which states that for any prime p and integer a we have $a^p \equiv a \pmod{p}$. Restricting a to be not divisible by p then makes Fermat's Little theorem equivalent to $a^{p-1} \equiv 1 \pmod{p}$ which is the same as Euler's theorem in the case that n is a prime p. Also, for those readers familiar with groups we can see that Euler's theorem is a specific case of the fact that in a finite group, the order of an element of that group divides the order of the group.

NUMERICAL PROBLEMS

Prob.1. Show that the set $G = \{a + b\sqrt{2} : \forall a, b \in Q\}$ is a group with respect to addition.

Or

Show that the algebraic structure $\left(\{a+b\sqrt{2}: a, b \in I\}, +\right)$ forms a group.

(R.G.P.V., Dec. 2016)

Sol. (i) Closure Property – Let x, y be any two elements of G.

Then, $x = a + b\sqrt{2}$ and $y = c + d\sqrt{2}$, where, $\forall$ a, b, c, d $\in$Q

Now, $x + y = \left(a+b\sqrt{2}\right)+\left(c+d\sqrt{2}\right) = (a + c) + (b + d)\sqrt{2}$

Since (a + c) and (b + d) are the elements of Q

$\therefore$ $(a + c) + (b + d)\sqrt{2} \in G \Rightarrow x + y \in G, \ \forall\ x, y \in G$

$\therefore$ G is closed with respect to addition.

(ii) Associativity – The elements of G are all real numbers and the addition of real numbers is associative.

(iii) Existence of Identity – Let $a + b\sqrt{2} \in G$, where $\forall$ a , b$\in$Q. Then we have,

$$\left(a+b\sqrt{2}\right)+\left(0+0\sqrt{2}\right) = (a + 0) + (b + 0)\sqrt{2} = a + b\sqrt{2}$$

$\therefore$ $0 + 0\sqrt{2}$ is the additive identity of $a + b\sqrt{2}$.

(iv) Existence of Inverse – Let $a+b\sqrt{2} \in G$, where, $\forall$ a,b$\in$Q. Then we have, $a+b\sqrt{2} \in G \Rightarrow (-a)+(-b)\sqrt{2} \in G$

Since $a, b \in Q \Rightarrow -a, -b \in Q$

Now, $[(-a)+(-b)\sqrt{2}] + (a+b\sqrt{2}) = [(-a)+a] + [(-b)+b]\sqrt{2}$

$= 0 + 0\sqrt{2}$ = identity

$\therefore$ $(-a)+(-b)\sqrt{2}$ is the additive inverse of $a+b\sqrt{2}$

Hence, G is a group with respect to addition. **Proved**

Prob.2. Find the multiplicative inverse of 726 modulo 1549.

Sol. Recall earlier we used the Euclidean algorithm to find that gcd (1549, 726) = 1 by finding –

$1549 = 726(2) + 97 \rightarrow \quad 97 = 1549 - 726(2)$ ...(v)

$726 = 97(7) + 47 \rightarrow \quad 47 = 726 - 97(7)$ ...(iv)

$97 = 47(2) + 3 \rightarrow \quad 3 = 97 - 47(2)$ ...(iii)

$47 = 3(15) + 2 \rightarrow \quad 2 = 47 - 3(15)$ ...(ii)

$3 = 2(1) + 1 \rightarrow \quad 1 = 3 - 2(1)$ ...(i)

Working backwards through these equations we obtain –

$1 = [3 - 2]$ ...(i)

$= 3 - [47 - 3(15)]$ ...(ii)

$= -47 + 3(16)$

$= -47 + 16[97 - 47(2)]$ ...(iii)

$= 97(16) - 47(33)$

$= 97(16) - 33[726 - 97(7)]$ ...(iv)

$= -726(33) + 97(247)$

$= -726(33) + 247[1549 - 726(2)]$ ...(v)

$= 726(-527) + 1549(247)$.

Thus $726^{-1} \equiv -527 \equiv 1022 \pmod{1549}$

Prob.3. Explain Euclidean algorithm. Solve the following using this algorithm –

(i) Determine gcd (24140, 16762)

(ii) Determine gcd (4655, 12075)

(R.G.P.V., Dec. 2011)

Sol. **Euclidean Algorithm** – Refer to Q.12.

(i) gcd(24140, 16762)

We apply the Euclidean algorithm using a table.

q	*r₁*	*r₂*	*r*
1	24140	16762	7378
2	16762	7378	2006
3	7378	2006	1360
1	2006	1360	646
2	1360	646	68
9	646	68	34
2	68	34	0
	34	0	

Thus, we have gcd(24140, 16762) = **34** **Ans.**

(ii) gcd (4655, 12075)

We apply the Euclidean algorithm using a table.

q	*r₁*	*r₂*	*r*
0	4655	12075	4655
2	12075	4655	2765
1	4655	2765	1890
1	2765	1890	875
2	1890	875	140
6	875	140	35
4	140	35	0
	35	0	

Thus, we have gcd(4655, 12075) = **35** **Ans.**

Prob.4. Explain Euclidean algorithm and solve the following using above algorithm –

(i) Determine gcd (1970, 1066)

(ii) Determine gcd (24140, 16762)

(R.G.P.V., June 2012)

Sol. **Euclidean Algorithm** – Refer to Q.12.

(i) gcd(1970, 1066)

We apply Euclidean algorithm using a table –

q	*r₁*	*r₂*	*r*
1	1970	1066	904
1	1066	904	162
5	904	162	94
1	162	94	68

1	94	68	26
2	68	26	16
1	26	16	10
1	16	10	6
1	10	6	4
1	6	4	2
2	4	2	0
	2	0	

Thus, gcd(1970, 1066) = **2** Ans.

(ii) Refer to Prob.3 (i).

Prob.5. Find gcd(1549, 726) using the Euclidean algorithm.

Sol. We find,

$$1549 = 726(2) + 97$$
$$726 = 97(7) + 47$$
$$97 = 47(2) + 3$$
$$47 = 3(15) + 2$$
$$3 = 2(1) + 1.$$

The next remainder we obtain will be zero, so gcd(1549, 726) = 1.

INTRODUCTION TO CRYPTOGRAPHY – PRINCIPLES OF CRYPTOGRAPHY, CLASSICAL CRYPTOSYSTEM, CRYPTANALYSIS ON SUBSTITUTION CIPHER (FREQUENCY ANALYSIS), PLAY FAIR CIPHER, BLOCK CIPHER

Q.16. What do you understand by cryptography ?

Ans. Cryptography is the art and science of achieving security by encoding messages to make them non-readable. Fig. 1.5 shows the conceptual view of cryptography.

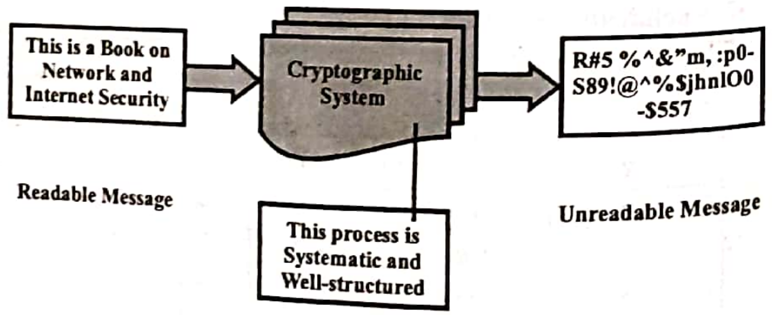

This is a Book on Network and Internet Security
Cryptographic System
R#5 %^&"m, ;p0- S89!@^%$jhnlO0 -$557
Readable Message
This process is Systematic and Well-structured
Unreadable Message

Fig. 1.5 Cryptographic System

Q.17. What are three basic operations in cryptography ? ***(R.G.P.V., June 2011)***

Ans. Cryptographic systems are characterized along three independent dimensions –

(i) The Type of Operations Used for Transforming Plaintext to Ciphertext – All encryption algorithms are based on two general principles – substitution, in which each element in the plaintext (bit, letter, group of bits or letters) is mapped into another element, and transposition, in which elements in the plaintext are rearranged. The fundamental requirement is that no information be lost (that is, that all operations are reversible). Most systems, referred to as product systems, involve multiple stages of substitutions and transpositions.

(ii) The Number of Keys Used – If both sender and receiver use the same key, the system is referred to as symmetric, single-key, secret-key, or conventional encryption. If the sender and receiver each uses a different key, the system is referred to as asymmetric, two-key, or public-key encryption.

(iii) The Way in which the Plaintext is Processed – A block cipher processes the input one block of elements at a time, producing an output block for each intput block. A stream cipher processes the input elements continuously, producing output one element at a time, as it goes along.

Q.18. What is the difference between traditional cryptography and modern cryptography ? Explain the significance of work factor in cryptography. ***(R.G.P.V., Dec. 2004)***

Ans. In the last few decades, traditional cryptographic algorithms, being mathematical in nature, have become so advanced that they can only be handled by computers. This, in effect, means that the uncoded message (prior to encryption) is binary in form and can therefore be anything a picture, a text such as an e-mail or even a video.

As with most historical ciphers, the security of the message being sent relies on the algorithm itself remaining secret. This technique is known as a Restricted Algorithm. It has the following fundamental drawbacks.

(i) The algorithm obviously has to be restricted to only those people that you want to be able to decode your message. Therefore, a new algorithm must be invented for every discrete group of users.

(ii) A large or changing group of users cannot utilise them, as every time one user leaves the group, everyone must change the algorithm.

(iii) If the algorithm is compromised in any way, a new algorithm must be implemented.

Because of these drawbacks, restricted algorithms (traditional cryptography) are no longer popular and have given way to key-based algorithms.

Practically, all modern cryptographic systems make use of a key. Algorithms that use a key allow all details of the algorithm to be widely available. This is because all of the security lies in the key. With a key-based algorithm, the plaintext is encrypted and decrypted by the algorithm which uses a certain key, and the resulting ciphertext is dependent on the key, and not the algorithm. This means that an eavesdropper can have a complete copy of the algorithm in use but without the specific key used to encrypt that message, it is useless.

Modern cryptography has become so complex and effective that now it is not only used for military but has many commercial uses and applications.

The real secrecy now is in the key, and its length is a major design issue. Consider a simple combination lock. The general principle is that you enter digits in sequence. Everyone knows this but the key is secret. A key length of two digits means that there are 100 possibilities. A key length of three digits means 1000 possibilities and a key length of six digits means a million. The longer the key, the higher the ***work factor*** the cryptanalyst has to deal with. The work factor for breaking the system by exhaustive search of the key space is exponential in the key length. This is the significance of work factor in cryptography. Work factor provides an indication of processing complexity required for cryptanalysis. It is a measure of time needed to perform the attack.

Q.19. How many keys are required for two people to communicate via a cipher ? ***(R.G.P.V., June 2013)***

Or

How many keys are required for two parties to communicate via a cipher ? Why ? ***(R.G.P.V., June 2017)***

Ans. The encryption and decryption algorithms are known as ciphers. A key is a set of values that the cipher, as an algorithm, operates on. In symmetric key cryptography, the secret key must be shared between two persons. In asymmetric-key cryptography, the secret is personal, i.e. unshared. Each one creates and keeps his or her own secret.

In a community of m people m(m – 1)/2 shared secrets are required for symmetric-key cryptography, and m personal secrets are required in asymmetric-key cryptography. It means that for two people only one key is required for symmetric-key cryptography and two keys are required in asymmetric-key cryptography.

Q.20. Describe conventional encryption model. What are the requirements for secure use of conventional encryption ? ***(R.G.P.V., June 2009)***

Or

What do you understand by conventional encryption model ? Discuss in detail. ***(R.G.P.V., Dec. 2005)***

Ans. Symmetric encryption is also referred to as ***conventional encryption*** or ***single-key encryption***. It was the only type of encryption in use prior to the development of public-key encryption. It remains by far the most widely used of the two types of encryption.

A symmetric encryption scheme has five ingredients as shown in fig. 1.6.

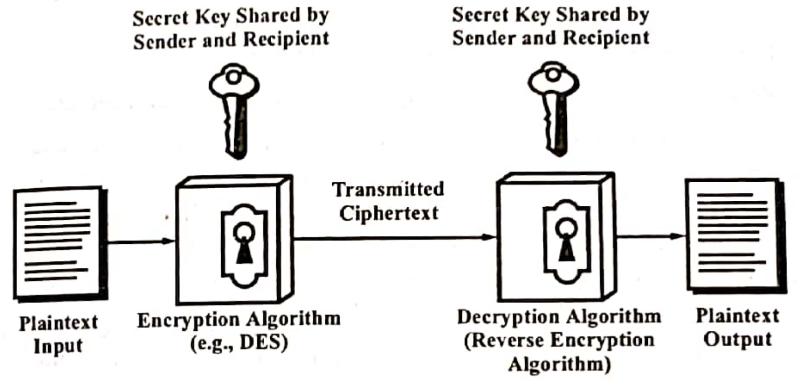

Secret Key Shared by Sender and Recipient
Secret Key Shared by Sender and Recipient
Transmitted Ciphertext
Plaintext Input
Encryption Algorithm (e.g., DES)
Decryption Algorithm (Reverse Encryption Algorithm)
Plaintext Output

Fig. 1.6 Simplified Model of Conventional Encryption

(i) ***Plaintext*** – This is the original intelligible message or data that is fed into the algorithm as input.

(ii) ***Encryption Algorithm*** – The encryption algorithm performs various substitutions and transformations on the plaintext to convert it into ciphertext.

(iii) ***Secret Key*** – The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext. The algorithm will produce a different output depending on the specific key being used at the time. The exact substitutions and transformations performed by the algorithm depend on the key.

(iv) ***Ciphertext*** – This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts. The ciphertext is an apparently random stream of data and, as it stands, is unintelligible.

(v) ***Decryption Algorithm*** – This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key as the input and produces the original plaintext.

There are two requirements for secure use of conventional encryption –

(i) We need a strong encryption algorithm. At a minimum, we would like the algorithm to be such that an opponent who knows the algorithm and has access to one or more ciphertexts would be unable to decipher the ciphertext or figure out the key. Usually, this requirement is stated in a stronger form. The opponent should be unable to decrypt ciphertext or discover the key even if he or she is in possession of a number of ciphertexts together with the plaintext that produced each ciphertext.

(ii) Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure. If someone can discover the key and knows the algorithm, all information using this key is readable.

Q.21. Explain the basic principle of cryptography.

Ans. The following are some important principles of cryptography –

(i) Encryption – Encryption is one of the important principles of cryptography. This principle indicates that a message or information must be encrypted to become unreadable so that the privacy of individuals is protected. This principle also shows that the recipient of information must decrypt the received information by using a special digital key.

(ii) Authentication – One of the important principles of cryptography is identifying the origin of the information. When the source of information is identified it is easy to communicate securely. Authentication is only possible by providing a special key exchange to be used accordingly by the sender to prove his/her identity.

(iii) Integrity – Integrity of information sent to the receiver is very important. This principle indicates that cryptography ensures the integrity of data by providing codes and digital keys to ensure that what we receive is genuine and from the intended person. The receiver is assured that the information received has not been modified or compromised during the process of transmission. For example, a cryptographic hash is utilized to ensure the integrity of the information.

(iv) Non-Repudiation – This principle ensures that the sender of the information cannot deny the fact that he/she never sent the information. This principle uses digital signatures to prevent the sender from denying the origin of the data.

Q.22. Explain some classical cryptosystem.

Ans. Some classical cryptosystems are as follows –

(i) Affine and Caesar Cryptosystem – In the affine cryptosystem AFFINE a message symbol i (a residue class modulo m represented in the positive residue system) is encrypted in the following way –

$$e_{k_1}(i) = (ai + b, \text{ mod } M).$$

Here a and b are integers and $\bar{a}$ has an inverse class $\bar{c}$ modulo M, in other words gcd(a, M) = 1. The encrypting key k_1 is formed by the pair (a, b) and the decrypting key k_2 by the pair (c, b) (usually represented in the positive residue system). The decrypting function is

$$d_{k_2}(j) = (c(j - b), \text{ mod } M).$$

So the length of the message block is one. Hence affine encrypting is also suitable for stream encryption. When choosing a and b from the positive residue system the number of possible values of a is $\phi(M)$ and all in all there are $\phi(M)$ M different encrypting keys. The number of encrypting keys is thus quite small. Some values –

$$\phi(10) = 4,\ \phi(26) = 12,\ \phi(29) = 28,\ \phi(40) = 16$$

The special case where a = 1 is known as the Caesar cryptosystem CAESAR. A more general cryptosystem, where

$$e_{k_1}(i) = (p(i), \text{ mod } M)$$

and p is a polynomial with integral coefficients, is not really much more useful as there are still very few keys.

(ii) Hill Cryptosystem – In Hill's cryptosystem HILL we use the same encoding of symbols as residue classes modulo M as in AFFINE. However, now the block is formed of d residue classes considered as a d-vector. Hill's original d was 2. The encrypting key is a d × d matrix **H** that has an inverse matrix modulo M. This inverse matrix $\mathbf{H}^{-1} = \mathbf{K}$ modulo M is the decrypting key.

A message block

$$\mathbf{i} = (i_1, \ldots\ldots, i_d)$$

is encrypted as

$$e_{\mathbf{H}}(\mathbf{i}) = (\mathbf{iH}, \text{ mod } M)$$

and decrypted similarly as

$$e_{\mathbf{K}}(\mathbf{j}) = (\mathbf{jK}, \text{ mod } M)$$

Here we calculate modulo M in the positive residue system.

There are as many encrypting keys as there are invertible d × d, matrices modulo M. This number is quite hard to compute. However, usually there is a relatively large number of keys if d is large.

A special case of HILL is PERMUTATION or the so-called, permutation encryption. Here **H** is a permutation matrix, in other words, a matrix that has exactly one element equal to one in every row and in every column all other elements being zeros. Note that in this case $\mathbf{H}^{-1} = \mathbf{H}^T$, or that **H** is an orthogonal matrix. In permutation encrypting the symbols of the message block are permutated using the constant permutation given by **H**.

A more general cryptosystem is AFFINE-HILL or the affine Hill cryptosystem. Comparing with HILL, now the encrypting key k_1 is a pair (**H**, **b**), where **b** is a fixed d-vector modulo M, and the decrypting key k_2 is the corresponding pair (**K**, **b**). In this case

$$e_{k_1}(\mathbf{i}) = (\mathbf{iH} + \mathbf{b}, \text{ mod } M)$$

and

$$e_{k_2}(\mathbf{j}) = ((\mathbf{j} - \mathbf{b})\mathbf{K}, \text{ mod } M).$$

From this we obtain a special case, the so-called Vigenere encryption VIGENERE by choosing $\mathbf{H} = \mathbf{I}_d$(d × d identity matrix). (This choice of **H** is not suitable for HILL!) In Vigenere's encryption we add in the message block symbol by symbol a keyword of length d modulo M.

Other generalizations of HILL are the so-called rotor cryptosystems, that are realized using mechanical and electro-mechanical devices. The most familiar example is the famous ENIGMA machine used by Germans in the Second World War.

(iii) One-time-pad Cryptosystem – Message symbols are often encoded binary numbers of a certain maximum length, for example ASCII encoding or UNICODE encoding. Hence we may assume that the message is a bit vector of length M. If the maximum length of the message is known in advance and encrypting is needed just once then we may choose a random bit vector **b** (or vector modulo 2) of length M as the key, the so-called one-time-pad, which we add to the message modulo 2 during the encryption. The encrypted message vector obtained as result is also random and a possible eavesdropper won't get anything out of it without the key. During the decrypting we correspondingly add the same vector **b** to the encrypted message, since $2\mathbf{b} \equiv \mathbf{0}$ mod 2. In this way we get the so-called one-time-pad cryptosystem ONE-TIME-PAD.

Q.23. What do you understand by plaintext ?

Ans. The data that you want to keep secret is called plaintext (some call it clear text). Any communication in the language that we speak – that is the human language, takes the form of plaintext or clear text. That is, a message in plaintext can be understand by anybody knowing the language as long as the message is not codified in any manner. For example, when we speak in our daily life, we use plaintext because we do not want to hide anything. Suppose I say "Hello Bob", it is a plaintext because both Bob and I know its

meaning. We also use plaintext during electronic conversations. For example, when we send an email to someone, we compose the email message using English language. Any person who reads this email would know that I have written. As before, this is simply because I am not using any codified language here.

Q.24. Differentiate between private and public keys.

Ans. Public-key cryptography requires each user to have two keys; a ***public-key,*** used by the entire world for encrypting messages to be sent to that user, and a ***private-key,*** which the user needs for decrypting messages.

Q.25. What is substitution cipher or technique ? Discuss briefly the various substitution cipher.

Ans. A substitution cipher or technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols. If the plaintext is viewed as a sequence of bits, then substitution invovles replacing plaintext bit patterns with ciphertext bit patterns.

(i) Caesar Cipher – The earliest known use of a substitution cipher, and the simplest, was by Julius Caesar. The Caesar cipher involves replacing each letter of the alphabet with the letter standing three places further down the alphabet. For example

plain : meet me after the toga party
cipher : PHHW PH DIWHU WKH WRJD SDUWB

Note that the alphabet is wrapped around, so that the letter following Z is A. We can define the transformation by listing all possibilities, as follows –

plain :	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
cipher :	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Let us assign a numerical equivalent to each letter –

a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12

n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

Then the algorithm can be expressed as follows. For each plaintext letter p, substitute the ciphertext letter C.

$$C = E(p) = (p + 3) \text{ mod } (26)$$

A shift may be of any amount, so that the general Caesar algorithm is

$$C = E(p) = (p + k) \bmod (26)$$

where k takes on a value in the range 1 to 25. The decryption algorithm is simply

$$p = D(C) = (C - k) \bmod (26)$$

(ii) Monoalphabetic Ciphers – If, instead, the "cipher" line can be any permutation of the 26 alphabetic characters, then there are 26! or greater than 4×10^{26} possible keys. This is 10 orders of magnitude greater than the key space for DES and would seem to eliminate brute-force techniques for cryptanalysis. Such an approach is referred to as a ***monoalphabetic substitution cipher***, because a single cipher alphabet (mapping from plain alphabet to cipher alphabet) is used per message.

(iii) Playfair Cipher – The best-known multiple-letter encryption cipher is the Playfair, which treats diagrams in the plaintext as single units and translates these units into ciphertext diagrams.

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

The Playfair algorithm is based on the use of a 5 × 5 matrix of letters constructed using a keyword. Here is an example, solved by Lord Peter Wimsey in Dorothy Sayers's *Have His Carcase*.

In this case, the keyword is *monarchy*. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter. Plaintext is encrypted two letters at a time, according to the following rules –

(a) Repeating plaintext letters that would fall in the same pair are separated with a filler letter, such as x, so that balloon would be treated as ba lx lo on.

(b) Plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.

(c) Plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the row circularly following the last. For example, mu is encrypted as CM.

(d) Otherwise, each plaintext letter is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM (or JM, as the encipherer wishes).

(iv) Hill Cipher – Another interesting multiletter cipher is the Hill cipher, developed by the mathematician Lester Hill in 1929. The encryption

algorithm takes m successive plaintext letters and substitutes for them m ciphertext letters. The substitution is determined by m linear equations in which each character is assigned a numerical value (a = 0, b = 1,..., z = 25). For m = 3, the system can be described as follows –

$$c_1 = (k_{11}p_1 + k_{12}p_2 + k_{13}p_3) \bmod 26$$
$$c_2 = (k_{21}p_1 + k_{22}p_2 + k_{23}p_3) \bmod 26$$
$$c_3 = (k_{31}p_1 + k_{32}p_2 + k_{33}p_3) \bmod 26$$

This can be expressed in term of column vectors and matrices –

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \bmod 26$$

or

$$\mathbf{C} = \mathbf{KP} \bmod 26$$

where **C** and **P** are column vectors of length 3, representing the plaintext and ciphertext, and **K** is a 3 × 3 matrix, representing the encryption key. Operations are performed mod 26.

For example, consider the plaintext "paymoremoney", and use the encryption key

$$\mathbf{K} = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$$

The first three letters of the plaintext are represented by the vector (15 0 24). Then **K**(15 0 24) = (375 819 486) mod 26 = (11 13 18) = LNS. Continuing in this fashion, the ciphertext for the entire plaintext is LNSHDLEWMTRW.

Decryption requires using the inverse of the matrix **K**. The inverse $\mathbf{K}^{-1}$ of a matrix **K** is defined by the equation $\mathbf{KK}^{-1} = \mathbf{K}^{-1}\mathbf{K} = \mathbf{I}$, where **I** is the matrix that is all zeros except for ones along the main diagonal from upper left to lower right. The inverse of a matrix does not always exist, but when it does, it satisfies the preceding equation. In this case, the inverse is

$$\mathbf{K}^{-1} = \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix}$$

This is demonstrated as follows –

$$\begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix} = \begin{pmatrix} 443 & 442 & 442 \\ 858 & 495 & 780 \\ 494 & 52 & 365 \end{pmatrix} \bmod 26 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$K = \begin{pmatrix} 15 & 2 \\ 16 & 5 \end{pmatrix} \begin{pmatrix} 9 & 2 \\ 1 & 15 \end{pmatrix} = \begin{pmatrix} 137 & 60 \\ 149 & 107 \end{pmatrix} \bmod 26 = \begin{pmatrix} 7 & 8 \\ 19 & 3 \end{pmatrix}$$

This result is verified by testing the remaining plaintext-ciphertext pair.

(v) Polyalphabetic Ciphers – One way to change the frequencies and destroy common sequences is to use different monoalphabetic substitutions as one proceeds through the plaintext message. Such an approach is known as ***polyalphabetic substitution cipher***. Like the monoalphabetic cipher, it replaces each character with another. The difference is that a given plaintext character is not always replaced with the same ciphertext one. We can choose a replacement depending not only on the actual plaintext character but on its position in the message as well.

An example of a polyalphabetic cipher is a ***Vigenere cipher***. It uses a two dimensional array of characters (its encryption key) in which each row contains the letters of the alphabet. Fig. 1.7 shows the key for vigenère cipher.

```
row 0 : A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
row 1 : B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
row 2 : C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
row 3 : D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
            .                            .
            .                            .
row 24 : Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
row 25 : Z A B C D E F G H I J K L M N O P Q R S T U V W X Y
```

Fig. 1.7 Key for Vigenere Cipher

The first row (row 0) contains the letters written from A to Z. The second row has the letters written from B to Z followed by A at the end. Each subsequent row is formed by moving each letter from the previous row left one position, with the leftmost character being moved to the right most position.

To replace a letter, let i be its relative position (first position is relative position zero) in the message and j its relative position in the alphabet. Let V be the array. Replace the letter with the one in V [(i mod 26), j)].

For example, suppose the word THE appears three times in a message, beginning in positions 25, 54, and 104. Table 1.1 shows the required calculations and the substituted ciphertext letters. The three occurrences of the word THE encrypt to SHF, VKI and TIG.

Table 1.1 Letters Substitutions Using Vigenere Cipher

Plaintext Letter	*i = (Relative Position in Message)*	*i MOD 26*	*j Relative Position in Alphabet*	*Ciphertext Letter*
T	25	25	19	S
H	26	0	7	H
E	27	1	4	F
T	54	2	19	V
H	55	3	7	K
E	56	4	4	I
T	104	0	19	T
H	105	1	7	I
E	106	2	4	G

The Vigenere cipher seems to solve the repetition problems, but in fact it has only reduced them. Repetitions and patterns still occur. For example, the two encrypted words SHF and TIG may seem dissimilar, but they are not. The letters in TIG are the alphabetic successors to the letters is SHF. To a cryptanalyst trying to break a code, it is a very big clue to the encryption method.

In fact, all of the encrypted forms of THE share a similarity. Consider two consecutive letters occupying the same relative positions, in each of two encrypted versions of THE. The difference of their ASCII codes is the same. For example, the difference between the ASCII codes of S and H, T and I, V and K are all 11. The common differences occur because each row of the matrix in fig. 1.7 is essentially in alphabetic order. The only exception is the transition point form Z to A.

This problem can be fixed by making each row a random permutation of the alphabet, but even that will not be enough, at least for long text. Since there are 26 rows, there are essentially 26 ways to encrypt a letter or a word. In a long message common words may appear several hundred times with only 26 different ways to encrypt the word, repetitions will still occur. They will just be a little harder to find.

Again you might respond by using even more rows in the matrix, each with a unique permutation of the alphabet, to provide more ways to encrypt words thus reducing the repetitions. Indeed, 26! unique permutations of the alphabet provide many alternatives. In the extreme case we could use a number of rows equal to the length of the message. Here each row is used just once during encryption, thus avoiding any repetition. Now the problem is that the encryption key is longer than the message communicating it to authorized receivers and storing it securely become problems.

Q.26. What is plaintext ? Why is monoalphabetic cipher difficult to crack ? ***(R.G.P.V., June 2014)***

Ans. Refer to Q.23 and Q.25 (ii).

Q.27. Briefly define the playfair cipher with taking a suitable example. ***(R.G.P.V., May 2019)***

Or

Explain playfair cipher with suitable example. ***(R.G.P.V., June 2017)***

Ans. Refer to Q.25 (iii).

Q.28. Define playfair cipher and polyalphabetic cipher with suitable example. ***(R.G.P.V., June 2016)***

Ans. Refer to Q.25 (iii) and (v).

Q.29. Write a short note on transposition techniques. ***(R.G.P.V., Dec. 2003, June 2004)***

Or

Write short note on transposition cipher.(R.G.P.V., June 2007, Dec. 2007)

Or

Explain the two fundamental building blocks of all cryptographic techniques.

Ans. Transposition cipher is a technique in which a kind of mapping is achieved by performing some sort of permutation on the plaintext letters.

The simplest such cipher is the rail fence technique, in which the plaintext is written down as a sequence of diagonals and then read off as a sequence of rows. For example, to encipher the message "meet me after the toga party" with a rail fence of depth 2, we write as follows –

m		e		m		a		t		r		h		t		g		p		r		y					
	e		t		e		f		e		t		e		o		a		a		t						

The encrypted message is

MEMATRHTGPRYETEFETEOAAT

This sort of thing would be trivial to cryptanalyze. A more complex scheme is to write the message in rectangle, row by row, and read the message off, column by column, but permute the order of the columns. The order of the columns then becomes the key to the algorithm. For example,

Key :	4	3	1	2	5	6	7
Plaintext :	a	t	t	a	c	k	p
	o	s	t	p	o	n	e
	d	u	n	t	i	l	t
	w	o	a	m	x	y	z

Ciphertext TTNAAPTMT SUOAOOWCOIXKNLYPETZ

A pure transposition cipher is easily recognized because it has the same latter frequencies as the original plaintext. For the type of columnar transposition just shown, cryptanalysis is fairly straightforward and involves laying out the ciphertext in a matrix and playing around with column positions. Digram and trigram frequency tables can be useful.

The transposition cipher can be made more secure by performing more than one stage of transposition. The result is a more complex permutation that is not easily reconstructed. Thus, if the foregoing message is reencrypted using the same algorithm.

Key :	4	3	1	2	5	6	7
Input :	t	t	n	a	a	p	t
	m	t	s	u	o	a	o
	d	w	c	o	i	x	k
	n	l	y	p	e	t	z

Output : NSCYAUOPTTWLTMDNAOIEPAXTTOKZ

To visualize the result of this double transposition, designate the letters in the original plaintext message by the numbers designating their position. Thus, with 28 letters in the message, the original sequence of letters is

01	02	03	04	05	06	07	08	09	10	11	12	13	14
15	16	17	18	19	20	21	22	23	24	25	26	27	28

After the first transposition we have

03	10	17	24	04	11	18	25	02	09	16	23	01	08
15	22	05	12	19	26	06	13	20	27	07	14	21	28

which has a somewhat regular structure. But after the second transposition, we have

17	09	05	27	24	16	12	07	10	02	22	20	03	25
15	13	04	23	19	14	11	01	26	21	18	08	06	28

This is a much less structured permutation and is much more difficult to cryptanalyze.

Q.30. What are classical encryption techniques ? Write in brief. ***(R.G.P.V., June 2006)***

Or

Explain the classical encryption techniques. ***(R.G.P.V., Dec. 2005)***

Or

Explain two primary ways by which we can convert plaintext message into a ciphertext. ***(R.G.P.V., Dec. 2009)***

Ans. The classical encryption techniques are –

(i) ***Substitution Techniques*** – Refer to Q.25.

(ii) ***Transposition Techniques*** – Refer to Q.29.

Q.31. Differentiate substitution and transposition ciphers with suitable examples. ***(R.G.P.V., May 2018)***

Ans. The difference between substitution and transposition ciphers are as follows –

S.No.	*Basis for Comparison*	*Substitution Cipher*	*Transposition Cipher*
(i)	Basic	Replaces the plaintext characters with other characters, numbers and symbols.	Rearranges the position of the characters of the plaintext.
(ii)	Forms	Monoalphabetic and polyalphabetic substitution cipher.	Keyless and keyed transpositional cipher.
(iii)	Alterations	The identity of the character is changed while its position remains unchanged.	The position of the character is changed in spite of its identity.
(iv)	Demerit	The letter with the low frequency can discern the plaintext.	Keys near to the correct key can disclose the plaintext.
(v)	Example	Caesar cipher	Reil fence cipher

Q.32. Write short note on block cipher.

Ans. In block ciphers, the plaintext is split into fixed size chunks called *blocks* and each block is encrypted separately. Typically all blocks in the plaintext are encrypted using the same key. Block ciphers include DES, AES, RSA and ECC. Block sizes used in secret key cryptography are usually smaller – 64 bits in DES and 128 bits in AES. The block size in RSA is much larger – 768 or more bits, while the block size in ECC is about 200 bits. If two blocks of plaintext within a message are identical, their corresponding ciphertexts are identical.

Q.33. Write short note on confusion and diffusion.

Or

Differentiate between diffusion and confusion. ***(R.G.P.V., Dec. 2011, June 2012)***

Ans. The idea of confusion and diffusion in the operation of a cipher was first proposed by Claude Shannon in 1949.

Confusion is the property of a cipher whereby it provides no clue regarding the relationship between the ciphertext and the key. This relationship is obfuscated to the point where given a plaintext p, a sequence of keys k_1, k_2,..., k_i, and the corresponding ciphertexts, $E_{k1}(p)$, $E_{k2}(p)$,...., $E_{k(i)}(p)$, it is near impossible to deduce the value of a new, arbitrarily chosen key, k_j, used to create the ciphertext, $E_{kj}(p)$. Confusion reigns supreme with a cipher if, for any plaintext, p, if even a single bit in a key, k, is changed to produce k, then roughly the half bits in the ciphertexts $E_k(p)$ and $E_{k'}(p)$ are different. Moreover, the positions of the flipped bits in the block are random.

Diffusion is concerned with the relationship between the plaintext and the corresponding ciphertext. Diffusion holds if the statistics of a block of the plaintext is irretrievably dissipated or scattered across the block of its ciphertext. Thus, changing a single bit in a block of the plaintext will have the effect of changing each bit of the block of ciphertext with probability 0.5. If this were not the case i.e., if the probability were closer to 0 or 1, the cryptanalyst could derive clues that might help in mounting a known plaintext attack.

Practically, a strong substitution function enhances confusion while transposition is used to enhance diffusion. To get the benefit of both confusion and diffusion, both substitutions and transpositions are combined to create product ciphers, which is the principal design template for contemporary symmetric block ciphers.

Q.34. Write any two difference between diffusion and confusion. ***(R.G.P.V., June 2016)***

Ans. Refer to Q.33.

NUMERICAL PROBLEMS

Prob.6. Encrypt the message "money helps to build infrastructure" using the Hill cipher with the key $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$. ***Show your calculations and result.*** ***(R.G.P.V., Dec. 2005)***

Sol. According to Hill cipher

$$C = KP \bmod 26$$

where, C and P are column vectors of length 2, representing the ciphertext and plaintext, and k is a 2 × 2 matrix, representing the encryption key. Operations are performed mod 26.

Now, in the above problem, since the encryption key is a 2 × 2 matrix, therefore, grouping the given text in groups of two characters we get

mo ne yh e*l* ps to bu i*l* di nf ra st ru ct ur ex, where x is an additional character inserted to complete the grouping.

Now, converting the given characters to numeric values on a scale of 1 to 26.

mo	ne	yh	el	ps	to	bu	il	di	nf	ra	st	ru	ct	ur	ex
↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓	↓↓
13 15	14 5	25 8	5 12	16 19	20 15	2 21	9 12	4 9	14 6	18 1	19 20	18 21	3 20	21 18	5 24

Now as per the given formula, proceeding with each pair we get the desired result.

First of all take mo $\Rightarrow$ 13, 15

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} m = 13 \\ o = 15 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 177 \\ 170 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 21 \\ 14 \end{pmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get

21 = u, 14 = n.

Thus, mo becomes un. Similarly, processed for other pairs.

Now taking ne

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 14 \\ 5 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 146 \\ 105 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 16 \\ 1 \end{pmatrix} = \begin{pmatrix} p \\ a \end{pmatrix}$$

Again taking yh

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 25 \\ 8 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 257 \\ 181 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 23 \\ 25 \end{pmatrix} = \begin{pmatrix} w \\ y \end{pmatrix}$$

Again taking el

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 5 \\ 12 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 93 \\ 109 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 15 \\ 5 \end{pmatrix} = \begin{pmatrix} o \\ e \end{pmatrix}$$

Again taking ps

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 16 \\ 19 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 220 \\ 213 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 12 \\ 5 \end{pmatrix} = \begin{pmatrix} l \\ e \end{pmatrix}$$

Again taking to

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 20 \\ 15 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 240 \\ 205 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 6 \\ 23 \end{pmatrix} = \begin{pmatrix} f \\ w \end{pmatrix}$$

Again taking bu

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 2 \\ 21 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 102 \\ 157 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 24 \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ a \end{pmatrix}$$

Again taking il

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 9 \\ 12 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 129 \\ 129 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 25 \\ 25 \end{pmatrix} = \begin{pmatrix} y \\ y \end{pmatrix}$$

Again taking di

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 4 \\ 9 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 72 \\ 83 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 20 \\ 5 \end{pmatrix} = \begin{pmatrix} t \\ e \end{pmatrix}$$

Again taking nf

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 14 \\ 6 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 150 \\ 112 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 20 \\ 8 \end{pmatrix} = \begin{pmatrix} t \\ h \end{pmatrix}$$

Again taking ra

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 18 \\ 1 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 166 \\ 97 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 10 \\ 19 \end{pmatrix} = \begin{pmatrix} j \\ s \end{pmatrix}$$

Again taking st

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 19 \\ 20 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 251 \\ 235 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 17 \\ 1 \end{pmatrix} = \begin{pmatrix} q \\ a \end{pmatrix}$$

Again taking ru

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 18 \\ 21 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 246 \\ 237 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 12 \\ 3 \end{pmatrix} = \begin{pmatrix} l \\ c \end{pmatrix}$$

Again taking ct

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 3 \\ 20 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 107 \\ 155 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 3 \\ 25 \end{pmatrix} = \begin{pmatrix} c \\ y \end{pmatrix}$$

Again taking ur

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 21 \\ 18 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 261 \\ 231 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 1 \\ 23 \end{pmatrix} = \begin{pmatrix} a \\ w \end{pmatrix}$$

Again taking ex

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 5 \\ 24 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 141 \\ 193 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 11 \\ 11 \end{pmatrix} = \begin{pmatrix} k \\ k \end{pmatrix}$$

Now arranging the derived ciphertext as per the given plaintext.

"Money helps to build infrastructur e<u>x</u>".

unpaw yoele fw xayyt ethjsqalccyawk<u>k</u>

removing the extra letter x in the plaintext and corresponding derived ciphertext k we get the encrypted message as follows –

"unpaw yoele fw xayyt ethjsqalccyawk".

Prob. 7. Encrypt the message "Cryptography" using the Hill cipher with key $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$. ***Show your calculation.*** ***(R.G.P.V., June 2016)***

Sol. According to Hill cipher

$$C = KP \text{ mod } 26$$

By grouping the given text we get,

"C	r	y	p	t	o	g	r	a	p	h	y"
3	18	25	16	20	15	7	18	1	16	8	25

Now as per the given formula, proceeding with each pair we get the desired result.

First of all take "Cr" ⇒ 3, 18

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} c = 3 \\ r = 18 \end{pmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 9\times3+4\times18 \\ 5\times3+7\times18 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 27+72 \\ 15+126 \end{bmatrix} = \begin{bmatrix} 99 \\ 141 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 21 \\ 11 \end{bmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get 21 = u, 11 = k.

Thus, "cr" becomes "uk". Similarly, processed for other pairs. Taking "yp",

$$C = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}\begin{bmatrix} y = 25 \\ p = 16 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 9\times25+4\times16 \\ 5\times25+7\times16 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 225+64 \\ 125+112 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 289 \\ 237 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get 3 = c, 3 = c.

Thus, "yp" becomes "cc". Similarly, processed for other pairs. Taking "to",

$$C = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}\begin{bmatrix} t = 20 \\ o = 15 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 9 \times 20 + 4 \times 15 \\ 5 \times 20 + 7 \times 15 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 180 + 60 \\ 100 + 105 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 240 \\ 205 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 6 \\ 23 \end{bmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get 6 = f, 23 = w.

Thus, "to" becomes "fw". Similarly, processed for other pairs. Taking "gr",

$$C = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}\begin{bmatrix} g = 7 \\ r = 18 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 9 \times 7 + 4 \times 18 \\ 5 \times 7 + 7 \times 18 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 135 \\ 161 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get 5 = e, 5 = e.

Thus, "gr" becomes "ee". Similarly, processed for other pairs. Taking "ap",

$$C = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}\begin{bmatrix} a = 1 \\ p = 16 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 9 \times 1 + 4 \times 16 \\ 5 \times 1 + 7 \times 16 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 73 \\ 117 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 21 \\ 13 \end{bmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get 21 = u, 13 = m.

Thus, "ap" becomes "um". Similarly, processed for other pairs. Taking "hy",

$$C = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}\begin{bmatrix} n = 8 \\ y = 25 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 172 \\ 215 \end{bmatrix} \text{mod } 26$$

$$= \begin{bmatrix} 16 \\ 7 \end{bmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get 16 = p, 7 = g.

Thus, "hy" becomes "pg".

The derived ciphertext as per the given plaintext is "ukccfweeumpg".

Prob.8. Encrypt "meet me" using Hill cipher with key $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$***. Also decrypt the same.*** ***(R.G.P.V., June 2010)***

Sol. According to Hill cipher

$$C = KP \text{ mod } 26$$

By grouping the given text we get,

me et me

Now converting the given characters to numeric values on a scale of 1 to 26.

$$\begin{tikzcd}[column sep=tiny] & me \arrow[dl] \arrow[dr] & & & et \arrow[dl] \arrow[dr] & & & me \arrow[dl] \arrow[dr] & \\ 13 & & 5 & 5 & & 20 & 13 & & 5 \end{tikzcd}$$

Now as per the given formula, proceeding with each pair we get the desired result.

First of all take me ⇒ 13, 5

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} m = 13 \\ e = 5 \end{pmatrix} \text{mod } 26$$

$$= \begin{pmatrix} 137 \\ 100 \end{pmatrix} \text{mod } 26 = \begin{pmatrix} 7 \\ 22 \end{pmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get

$$7 = g,\ 22 = v$$

Thus, me becomes gv. Similarly, processed for other pairs.

Taking et,

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} e=5 \\ t=20 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 125 \\ 165 \end{pmatrix} \bmod 26 = \begin{pmatrix} 21 \\ 9 \end{pmatrix} = \begin{pmatrix} u \\ i \end{pmatrix}$$

Taking me,

$$C = \begin{pmatrix} g \\ v \end{pmatrix}$$

The derived ciphertext as per the given plaintext is "gvui gv".

Prob.9. Encrypt the message "meet at the airport" using the Hill cipher with the key $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$. ***Show your calculation and the result.***

(R.G.P.V., May 2019)

Sol. According to Hill cipher

$$C = KP \bmod 26$$

Now, converting the given characters to numeric values on a scale of 1 to 26.

me		et		at		th		ea		ir		po		rt	
13	5	5	20	1	20	20	8	5	1	9	18	16	15	18	20

Now as per the given formula, proceeding with each pair we get the desired result.

First of all take me ⇒ 13, 5

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} m=13 \\ e=5 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times13+4\times5 \\ 5\times13+7\times5 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 117+20 \\ 65+35 \end{pmatrix} = \begin{pmatrix} 137 \\ 100 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 7 \\ 22 \end{pmatrix}$$

Converting these values back to the alphabets to get the ciphertext, we get

$$7 = g,\ 22 = v$$

Thus, me becomes gv. Similarly, processed for other pairs.

Now taking et

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 5 \\ 20 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times5+4\times20 \\ 5\times5+7\times20 \end{pmatrix} = \begin{pmatrix} 45+80 \\ 25+140 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 125 \\ 165 \end{pmatrix} \bmod 26 = \begin{pmatrix} 21 \\ 9 \end{pmatrix} = \begin{pmatrix} u \\ i \end{pmatrix}$$

Again taking at

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 1 \\ 20 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times1+4\times20 \\ 5\times1+7\times20 \end{pmatrix} \bmod 26 = \begin{pmatrix} 9+80 \\ 5+140 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 89 \\ 145 \end{pmatrix} \bmod 26 = \begin{pmatrix} 11 \\ 15 \end{pmatrix} = \begin{pmatrix} k \\ o \end{pmatrix}$$

Again taking th

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 20 \\ 8 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times20+4\times8 \\ 5\times20+7\times8 \end{pmatrix} \bmod 26 = \begin{pmatrix} 180+32 \\ 100+56 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 212 \\ 156 \end{pmatrix} \bmod 26 = \begin{pmatrix} 4 \\ 0 \end{pmatrix} = \begin{pmatrix} d \\ z \end{pmatrix}$$

Again taking ea

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 5 \\ 1 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times5+4\times1 \\ 5\times5+7\times1 \end{pmatrix} \bmod 26 = \begin{pmatrix} 45+4 \\ 25+7 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 49 \\ 32 \end{pmatrix} \bmod 26 = \begin{pmatrix} 23 \\ 6 \end{pmatrix} = \begin{pmatrix} w \\ f \end{pmatrix}$$

Again taking ir

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 9 \\ 18 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times9+4\times18 \\ 5\times9+7\times18 \end{pmatrix} \bmod 26 = \begin{pmatrix} 81+72 \\ 45+126 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 153 \\ 171 \end{pmatrix} \bmod 26 = \begin{pmatrix} 23 \\ 15 \end{pmatrix} = \begin{pmatrix} w \\ o \end{pmatrix}$$

Again taking po

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 16 \\ 15 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times16+4\times15 \\ 5\times16+7\times15 \end{pmatrix} \bmod 26 = \begin{pmatrix} 144+60 \\ 80+105 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 204 \\ 185 \end{pmatrix} \bmod 26 = \begin{pmatrix} 22 \\ 3 \end{pmatrix} = \begin{pmatrix} v \\ c \end{pmatrix}$$

Again taking rt

$$C = \begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}\begin{pmatrix} 18 \\ 20 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 9\times18+4\times20 \\ 5\times18+7\times20 \end{pmatrix} \bmod 26 = \begin{pmatrix} 162+80 \\ 90+140 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 242 \\ 230 \end{pmatrix} \bmod 26 = \begin{pmatrix} 8 \\ 22 \end{pmatrix} = \begin{pmatrix} h \\ v \end{pmatrix}$$

The derived ciphertext as per the given plaintext is "gvui ko dzw fwovchv".

Prob.10. Write a program that can encrypt and decrypt using general Caesar cipher also known as additive cipher. ***(R.G.P.V., June 2013)***

Sol. This program translates a line of text into its Caesar cipher form –

```
import acm.program.*;
public class CaesarCipher extends ConsoleProgram {
    public void run( ) {
    println("This program uses a Caesar cipher for encryption.");
    int key = readInt("Enter encryption key:");
    String plaintext = readLine("Plaintext:");
```

```
        String ciphertext = encryptCaesar(plaintext, key);
        println("Ciphertext:" + ciphertext);
        }
    /*
    * Encrypts a string by adding the value of key to each character.
    * The first line makes sure that key is always positive by converting
    * negative keys to the equivalent positive shift.
    */
    private String encryptCaesar(String str, int key) {
        if(key < 0){
        key = 26 – (–key%26);
        }
        String result = " ";
        for(int i = 0; i < str.length( ); i++)
        {
            char ch = encryptCharacter(str.charAt(i), key);
            result + = ch;
        }
        return result;
        }
    /* Encrypts a single character using the key given. This method
    * assumes the key is non-negative. Non-letter characters are
    * returned unchanged.
    */
    private char encryptCharacter(char ch, int key)
    {
        if(Character.isLetter(ch))
        {
            ch = (char) ('A' + (Character.toUpperCase(ch)
                                      – 'A' + key)% 26);
        }
        return ch;
    }
}
```

DATA ENCRYPTION STANDARD (DES), TRIPLE DES, MODES OF OPERATION, STREAM CIPHER

Q.35. With the help of a block diagram explain DES encryption algorithm. *(R.G.P.V., June 2006, 2015)*

Or

Explain the data encryption standard. *(R.G.P.V., June 2005, Dec. 2005)*

Or

Write short note on DES. *(R.G.P.V., Dec. 2007)*

Or

Explain DES algorithm with the help of diagrams. *(R.G.P.V., June 2012)*

Ans. Data Encryption Standard (DES), is the name of the Federal Information Processing Standard (FIPS) 46-3, which describes the Data Encryption Algorithm (DEA) created by IBM, DES came about due to a request by the US National Bureau of Standards (NSB) requesting proposals for a standard cryptographic algorithm that satisfied the following criteria –

(i) Provides a high level of security.

(ii) The security depends on keys, not the secrecy of the algorithm.

(iii) The security is capable of being evaluated.

(iv) The algorithm is completely specified and easy to understand.

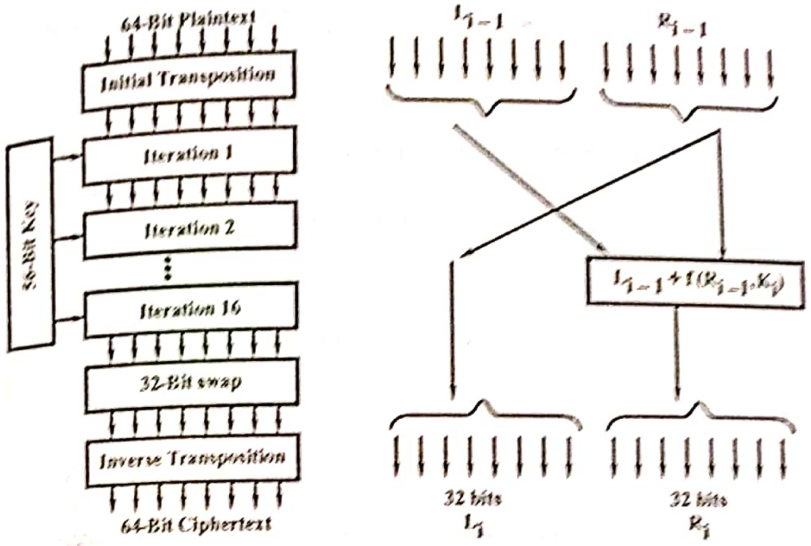
(v) It is efficient to use and adaptable.

(vi) Must be available to all users.

(vii) Must be exportable.

DEA, is an improvement of the 'Algorithm Lucifer' (IBM, 1970). DES is the best known and most widely used symmetric algorithm in the world. It was adopted in 1977 as a standard by US Government for all commercial and unclassified information. It is no longer secure in its original form, but in a modified form it is still useful. The DES works as follows –

The general outline of DES is given in fig. 1.8 (a). Plaintext is encrypted in blocks of 64 bits, giving 64 bits of ciphertext. The algorithm has 19 distinct stages. It is parameterized by a 56-bit key. The first stage is a key-independent transposition on the 64-bit plaintext. The exact inverse of this transposition is the last stage. The second last stage exchanges the leftmost 32 bits with the rightmost 32 bits. The rest of 16 stages are functionally identical but are parameterized by different functions of the key. The algorithm has been designed to allow decryption to be done with the same key as encryption. Thus its a symmetric-key algorithm. The steps are just run in the reverse order.



(a) General Outline *(b) Detail of One Iteration*

Fig. 1.8 Data Encryption Standard (DES)

The operation of one of these intermediate stages is illustrated in fig. 1.8 (b). Each stage takes two 32-bit inputs and produces two 32-bit outputs. The left output is simply a copy of the right input. The right ouput is the bitwise XOR of the left input and a function of the right input and the key for this stage K_i. All the complexity is due to this function.

This function constitutes four steps, carried out in sequence. First, a 48-bit number, E, is constructed by expanding the 32-bit R_{i-1} according to a fixed transposition and duplication rule. Second, E and K_i are XORed together. Then this output is partitioned into eight groups of 6 bits each, each of which is fed into a different S-box. Each of the 64 possible inputs to an S-box is mapped onto a 4-bit output. Finally, these 8 × 4 bits are passed through a P-box.

A different key is used in each of the 16 iterations. The algorithm starts after a 56-bit transposition is applied to the key. Just before each iteration, the key is partitioned into two 28-bit units, each of which is rotated left by a number of bits dependent on the iteration number. K_i is derived form this rotated key by applying yet another 56-bit transposition to it. A different 48-bit subset of the 56 bits is extracted and permuted on each round.

A technique that is sometimes used to make DES stronger is called ***whitening***. It consists of XORing a random 64-bit key with each plaintext block before feeding it into DES and then XORing a second 64-bit key with the resulting ciphertext before transmitting it. Whitening can easily be removed by running the reverse operations (if the receiver has the two whitening keys). Since this technique effectively adds more bits to the key length, it makes exhaustive search of the key space much more time consuming. It is noted that the same whitening key is used for each block (i.e., there is only one whitening key).

Q.36. What is the most popular symmetric encryption system used over the Web ? ***(R.G.P.V., June 2011)***

Ans. The most popular symmetric encryption used over the Web is the data encryption standard (DES).

DES – Refer to Q.35.

Q.37. During encrypting a message using DES in ciphertext block chaining mode, one bit of ciphertext in block C_i is accidentally transformed from a 0 to a 1 during transmission. How much plaintext will be grabled as a result ? ***(R.G.P.V., Dec. 2006)***

Ans. In ciphertext block chaining (CBC) mode, a single bit error in ciphertext block C_i during transmission may create error in most bits in plaintext block P_i during decryption. However, this single error toggles only one bit in plaintext P_{i+1} (the bit in the same location). Plaintext blocks P_{i+2} to P_N are not affected by this single bit error. A single bit error in ciphertext is self-recovered.

Q.38. While DES keys are 64 bits long, but its effective key length is only 56 bits, why ?

Ans. DES uses keys that are 64 bits long, but because eight of those bits are only parity bits and used to ensure that the key itself does not contain undiscovered errors, the effective length of the key is only 56 bits. While DES is fast, it has been broken using commercial grade computers in a reasonable, period of time that can be as short as three days. As computing power increased, it became clear that DES was less secure than it once was. To respond to this vulnerability, a new encryption algorithm known as Triple DES was developed.

Q.39. Write short note on strength of DES. ***(R.G.P.V., May 2018)***

Ans. As we know that DES has been adopted as a federal standard, there have been lingering concerns about the level of security provided by DES. These concerns, by and large, fall into two areas – key size and the nature of the algorithm.

The Use of 56-Bit Keys – With a key length of 56-bits, there are 2^{56} possible keys. Thus, brute-force attack appears impractical. However the assumption of one encryption per microsecond is overly conservative. As far back as 1977, Diffie and Hellman postulated that the technology existed to build a parallel machine with 1 million encryption devices, each of which could perform one encryption per μs.This would bring the average search time to 10 hours. But is expensive.

DES finally proved insecure in July 1998, when the Electronic Frontier Foundation announced that it has broken a DES encryption using a special purpose 'DES Cracker' machine that was built for less than $250,000. The attack took less than 3 days.

It is important to note that there is more to a key-search attack than simply running through all possible keys. Unless known plaintext is provided, the analyst must be able to recognize plaintext as plaintext. If the message is just plaintext in English then the result pops out easily, although English identification task have to be automated. If the text message has been compressed before encryption, then recognition is more difficult. If the numerical message has been compressed then the problem becomes even more difficult to automate. Thus, to supplement the brute-force attack some degree of knowledge about the expected plaintext is needed.

Fortunately, there are a number of alternatives to DES, the most important of which are AES and triple DES.

Nature of DES Algorithm – Another concern is the possibility that cryptanalysis is possible by exploiting the characteristics of the DES algorithm. The focus of concern has been on the eight substitution tables or S-boxes, that are used in each iteration. Because the design criteria for these boxes and indeed for entire algorithm, were not made public, there is a suspicion that the boxes were constructed in such a way that cryptanalysis is possible for an opponent who knows the weaknesses in the S-boxes. This assertion is tantalizing and over the years a number of regularities and unexpected behaviors of the S-boxes have been discovered. Despite this no one has so far succeeded in discovering the supposed fatal weaknesses in the S-boxes.

Timing Attacks – A timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts. A timing attack exploits the fact that an encryption or decryption algorithm often takes sightly different amounts of time on different inputs. This is a long way from knowing the actual key, but it is an intriguing first step. The authors conclude that DES appears to be fairly resistant to a successful timing attack but suggest some avenues to explore. Although this is an interesting line of attack, it so far appears unlikely that this technique will ever be successful against DES or more powerful symmetric ciphers such as AES and triple DES.

Q.40. Why is the middle portion of 3DES a decryption rather than an encryption ? ***(R.G.P.V., June 2017)***

Ans. For use in financial applications, 3DES was first standardized in ANSI standard X9.17 in 1985. 3DES was incorporated as part of the data encryption standard in 1999, with the publication of FIPS PUB 46-3. 3DES uses 3 keys. 3DES uses three executions of the DES algorithm. An encrypt-decrypt-encrypt sequence (see fig. 1.9) is followed by the function, is given below –

$$C = E(K_3, D(K_2, E(K_1, P)))$$

where, C = Ciphertext

P = Plaintext

E[K, X] = Encryption of X using key K

D[K, Y] = Decryption of Y using key K.

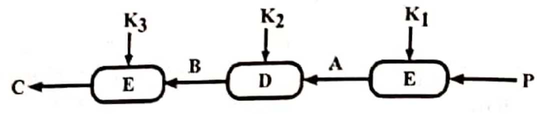


Fig. 1.9 Encryption

Decryption is easily the same operation with the keys reversed as shown in fig. 1.10.

$$P = D(K_1, E(K_2, D(K_3, C)))$$

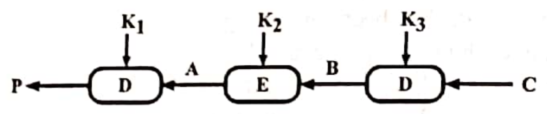


Fig. 1.10 Decryption

There is no cryptographic significance to the use of decryption for the second stage of 3DES encryption. Its only profit is that, it permits users of 3DES to decrypt data encrypted by users of the older single DES.

$$C = E(K_1, D(K_1, E(K_1, P))) = E[K, P]$$

3DES has an effective key length of 168 bits with three different keys. FIPS 46-3 also permits for the use of two keys, with $K_1 = K_3$, this gives for a key length of 112 bits. FIPS 46-3 has the guidelines for 3DES, which are given below –

(i) 3DES is the FIPS approved symmetric encryption algorithm of selection.

(ii) The original DES which uses a single 56-bit key, is allowed under the standard for legacy systems only. 3DES should be supported by new procurements.

(iii) With legacy DES systems, government organizations are encouraged to transition to 3DES.

(iv) It is anticipated that 3DES and the advanced encryption standard will coexist like FIPS-approved algorithms, permitting for a gradual transition to AES.

It is simple to see that 3DES is a formidable algorithm. Since the underlying cryptographic algorithm is DEA, 3DES may claim the same resistance to cryptanalysis based on the algorithm as is claimed for DEA. Again, brute force attacks are effectively not possible with a 168-bit key length.

AES is intended to replace 3DES, but this process will take a several years. It is anticipated by NIST that 3DES will remain an approved algorithm for the foreseeable future.

Q.41. Explain function of single round performed in each round of DES. ***(R.G.P.V., Dec. 2011)***

Ans. The internal structure of a single round is shown in fig. 1.11. The left and right halves of each 64-bit intermediate value are considered as separate 32-bit quantities, labeled L(left) and R(right). The overall processing at each round is shown by the following formulas –

$$L_i = R_i - 1$$

$$R_i = L_i - 1 \oplus F(R_{i-1}, K_i)$$

The R input is 32 bits. The round key K_i is 48 bits. This R input is first expanded to 48 bits by using a table that defines a permutation plus an expansion that involves duplication of 16 of the R bits (table 1.2). The resulting 48 bits are XORed with K_i. This 48-bit output passes through a substitution function and providing a 32-bit output. This 32-bit output is permuted as defined by table 1.3.

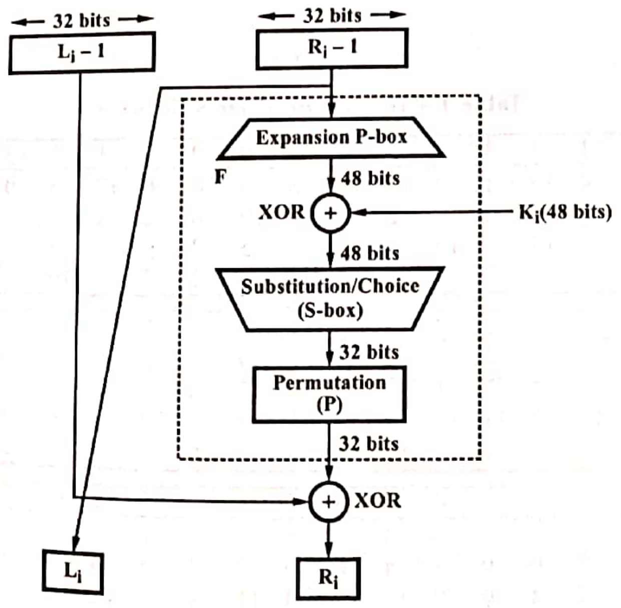


Fig. 1.11 DES Function

Table 1.2 Expansion Permutation (E)

32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

Table 1.3 Permutation Function P

16	7	20	21	29	12	28	17
01	15	23	26	5	18	31	10
02	8	24	14	32	27	3	9
19	13	30	06	22	11	4	25

Fig. 1.12 shows the role of the S-boxes. The substitution uses 8 S-boxes, each with a 6-bit input and a 4-bit output. These transformations are defined in table 1.4, which is interpreted as follows. The combination of bits 1 and 6 of the input to box S_i forms a 2 bit binary number to select one of four substitutions defined by the four rows in the table for S_i. The combination of bits 2 through 5 selects one of the sixteen columns. The decimal value in the cell selected by the row and column is then converted to its 4-bit representation to produce the output.

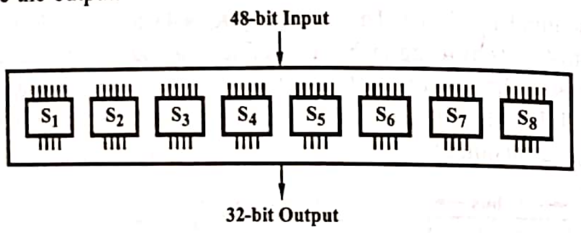


Fig. 1.12

Table 1.4 Definition of DES S-Boxes

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

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

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

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

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

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

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

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

Q.42. What is the purpose of the S-boxes in DES ? (R.G.P.V., June 2016)

Ans. Refer to Q.41.

Q.43. Which parameters and design choices determine the actual algorithm of a Feistel cipher ? What is the purpose of the S-boxes in DES ? (R.G.P.V., June 2013)

Ans. The choice of the following parameters and design features determines the exact realization of Fiestel cipher –

(i) Block Size – If the block size is large, then the high security is required. This also reduces encryption/decryption speed.

(ii) Key Size – If the key size is large, then the high security is required. The key size of 64 bits or less is now absolete, and 128 bits has become a common size.

(iii) Number of Rounds – A single round provides inadequate security but that multiple rounds provide increasing security. A typical size is 16 rounds.

(iv) Subkey Generation Algorithm – In this algorithm, greater complexity should lead to greater difficulty of cryptanalysis.

(v) Round Function – Again, greater complexity in this algorithm means greater resistance to cryptanalysis.

Purpose of the S-boxes in DES – Refer to Q.41.

Q.44. What is the purpose of the S-boxes in DES ? Explain the avalanche effect. ***(R.G.P.V., May 2018)***

Ans. **Purpose of the S-boxes in DES** – Refer to Q.41.

Avalanche Effect – A mathematical function which takes a message string of any length (pre-string) and backs a smaller fixed-length string (hash value) is known as one-way hash function. These functions are developed in such a way that not only is it complex to deduce the message from its hashed version as well as that even provided that all hashes are a certain length, it is very difficult to search two messages which hash to the same value. In fact to determine two messages with the same hash from a 128-bit hash function, 2^{64} hashes would have to be tried. In other words, the hash value of a file is a small unique "fingerprint". Even a slight change in an input string must cause the hash value to vary drastically. Even when one bit is flipped in the input string, then at least half of the bits will flip as a result in the hash value. This effect is known as *avalanche effect.*

Q.45. Explain the avalanche effect. ***(R.G.P.V., June 2016)***

Ans. Refer to Q.44.

Q.46. What are the block cipher modes of operation ? Explain each in short. ***(R.G.P.V., Dec. 2006)***

Or

What are the block cipher modes of operation ?
(R.G.P.V., June 2008, Dec. 2008)

Ans. The DES algorithm is a basic building block for providing data security. To apply DES in various applications four "modes of operation" have been defined. These four modes are intended to cover virtually all the possible applications of encryption for which DES could be used. As new applications and requirements have appeared, NIST has introduced a new mode. These five modes are intended for use with any symmetric block cipher, including triple DES and AES. All these modes are summarized in table 1.5.

Table 1.5 Block Cipher Modes of Operation

Mode	*Description*	*Typical Application*
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	• Secure transmission of single values (e.g., an encryption key)
Cipher Block Changing (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	• General-purpose block-oriented transmission. • Authentication.
Cipher Feedback (CFB)	Input is processed J bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	• General-purpose stream-oriented transmission. • Authentication.
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	• Stream-oriented transmission over noisy channel (e.g., satellite communication).
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	• General-purpose block-oriented transmission. • Useful for high-speed requirements.

Q.47. What is an initialization vector (IV) ? What is its significance ? ***(R.G.P.V., June 2014)***

Ans. An initialization vector is usually a random b-bit string, where b is the block size. It has no special meaning. The IV is used to make each message unique. The likelihood of IV repeating in two different message is quite rare because it is randomly generated. As a result, IV helps in making the cipher text somewhat unique or at least quite different from all the other cipher texts in a different message. It is not mandatory to keep IV secret, it can be known to everybody. This seems slightly concerning and confusing. However, if we relook at the operation of CBC, we will realize that IV is simply one of the two inputs to the first encryption step. The output of step 1 is cipher text block 1, which is also one of the two inputs to the second encryption step. In other words, cipher text block 1 is also an IV for step 2. Similarly, cipher text block 2 is also an IV for step 3 and so on. Since all these cipher text blocks will be sent to the receiver, we are actually anyway sending all IVs for step 2 onwards. Thus, there is no special reason why the IV for step 1 should be kept secret.

Q.48. Compare cipher block chaining with cipher feedback mode in terms of encryption operation needed to transmit a large file. Which one is better and why ? ***(R.G.P.V., Dec. 2004)***

Ans. In ***Cipher Block Chaining*** mode, the input to the encryption algorithm is the XOR of the current plaintext block and the preceding ciphertext block; the same key is used for each block. In effect, we have chained together the processing of the sequence of plaintext blocks. The input to the encryption function for each plaintext block bears no fixed relationship to the plaintext block. Therefore, repeating patterns of 64-bits are not exposed. Encryption in CBC mode is,

$$C_j = E_K[C_{j-1} \oplus P_j]$$

Whereas, in Cipher Feedback mode as with CBC, the units of plaintext are chained together, so that the ciphertext of any plaintext unit is a function of all the preceding plaintext. Thus, rather then units of 64 bits, the plaintext is divided into segments of s bits.

The input to the encryption function is a 64-bit shift register that is initially set to some initialization vector (IV). The leftmost s bits of the output of the encryption function are XORed with the first segment of plaintext P_1 to produce the first unit of ciphertext C_1, which is then transmitted. In addition, the contents of the shift register are shifted left by s bits and C_1 is placed in the rightmost s bits of the shift register.

Let, $S_s(X)$ be the most significant s bits of X. Then,

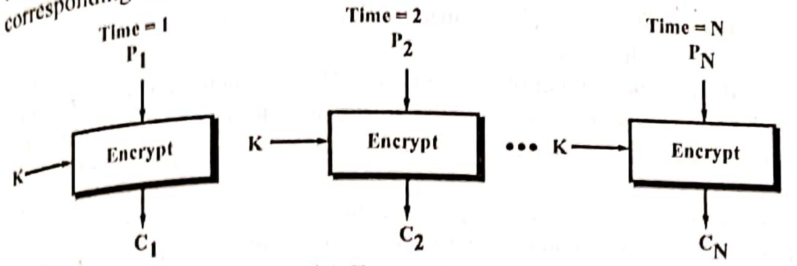
$$C_1 = P_1 \oplus S_s(E_K(IV)).$$

This process continues until all plaintext units have been encrypted.

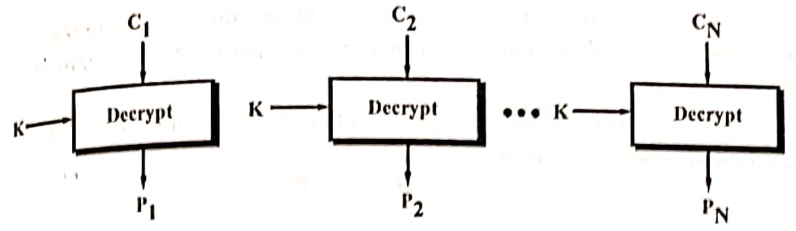
Because of the chaining mechanism of cipher block chaining (CBC) mode, it is an appropriate or a better mode than cipher feedback mode (CFB) to encrypt a large file and transmit it. Generally CBC is best suited for encrypting messages of length greater than 64 bits.

Q.49. Briefly explain electronic codebook mode.

Ans. The ***electronic codebook (ECB) mode*** is the simplest mode, in which plaintext is handled 64 bits at a time and each block of plaintext is encrypted using the same key as shown in fig. 1.13. The term *codebook* is used because, for a given key, there is a unique ciphertext for every 64-bit block of plaintext. Therefore, we can imagine a gigantic codebook in which there is an entry for every possible 64-bit plaintext pattern showing its corresponding ciphertext.



(a) Encryption



(b) Decryption

Fig. 1.13 Electronic Codebook (ECB) Mode

For a message longer than 64-bits, the procedure is simply to break the message into 64-bit blocks, padding the last block if necessary. Decryption is performed one block at a time, always using the same key. Fig. 1.13 shows the plaintext (padded as necessary) consists of a sequence of 64-bit blocks, $P_1, P_2, \ldots, P_N$ and the corresponding sequence of ciphertext blocks, $C_1, C_2, \ldots, C_N$.

The ECB method is ideal for a short amount of data, such as an encryption key. Thus, if you want to transmit a DES key securely, ECB is the appropriate mode to use.

The most significant characteristic of ECB is that the same 64-bit block of plaintext, if it appears more than once in the message always produces the same ciphertext.

For lengthy message, the ECB mode may not be secure. If the message is highly structured, it may be possible for a cryptanalyst to exploit these regularities. For example, if it is known that the message always starts out with certain predefined fields, then the cryptanalyst may have a number of known plaintext-ciphertext pairs to work with. If the message has repetitive elements, with a period of repetition a multiple of 64-bits, then these elements can be identified by the analyst. This may help in the analysis or may provide an opportunity for substituting or rearranging blocks.

involved with key distribution centers for symmetric encryption. In fact, some form of protocol is needed, generally involving a central agent, and the procedures involved are no simpler nor any more efficient than those required for symmetric encryption.

Q.8. Explain public-key cryptosystems.

Or

What are principal elements of public key cryptosystem ? What are the roles of the public key and private key ? *(R.G.P.V., Dec. 2006, 2008)*

Ans. A public-key encryption scheme has six ingredients as shown in fig. 2.4.

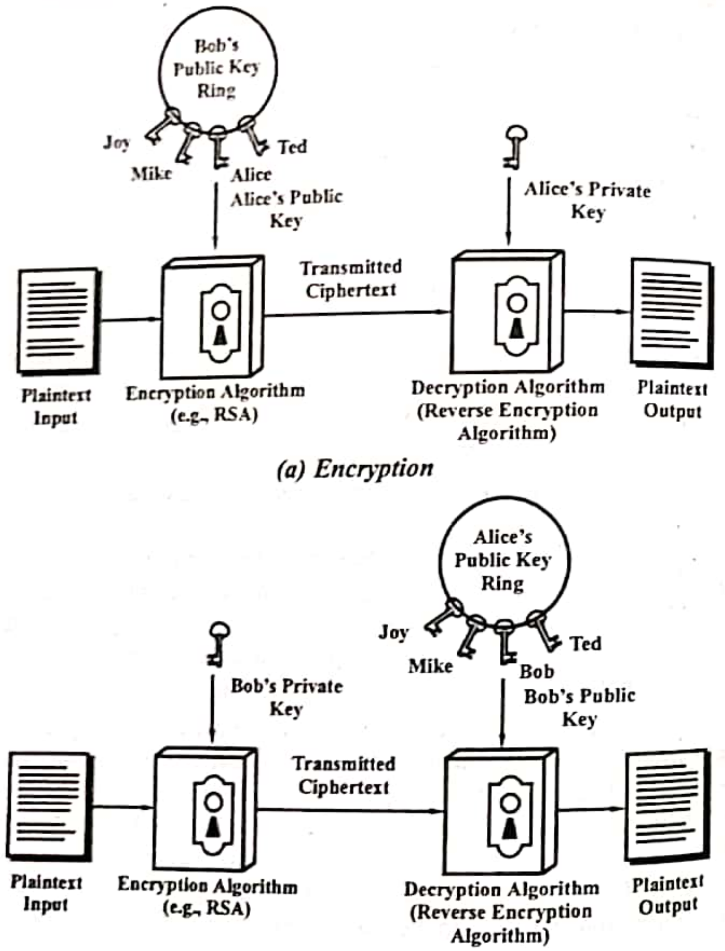


Fig. 2.4 Public-key Cryptography

(i) Plaintext – This is the readable message or data that is fed into the algorithm as input.

(ii) Encryption Algorithm – The encryption algorithm performs various transformation on the plaintext.

(iii) Public and Private Key – This is a pair of keys, one is used for encryption, and the other is used for decryption. The exact transformations performed by the encryption algorithm depend on the public or private key that is provided as input.

(iv) Ciphertext – This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.

(v) Decryption Algorithm – This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

The essential steps of the algorithm are as follows –

(i) Each user generates a pair of keys to be used for the encryption and decryption of messages.

(ii) Each user places one of the two keys in a public register or other accessible file. This key is known as ***public key***. The companion key is kept private. This key is known as ***private key***. As suggested in fig. 2.4, each user maintains a collection of public keys obtained from others.

(iii) In the figure, if Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.

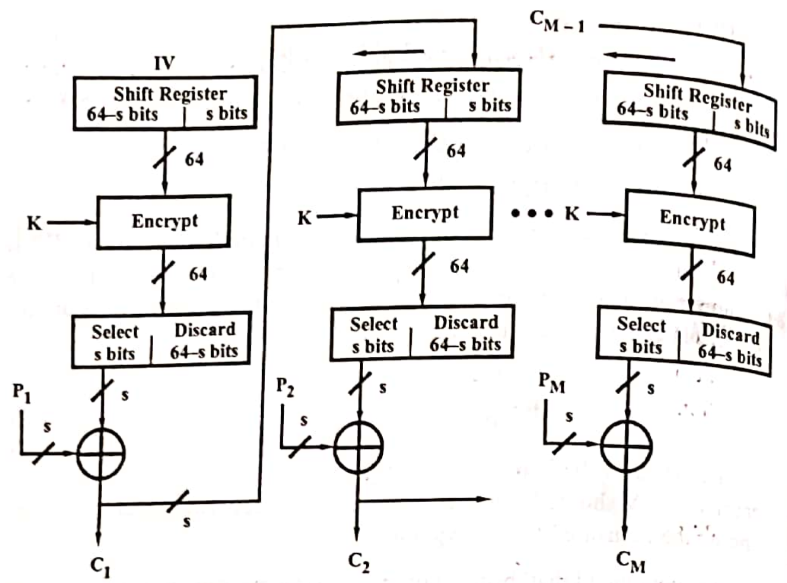
(iv) When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows her private key.

With this approach, all users have access to public keys, and private keys are generated locally by each user and therefore need never be distributed. As long as a system controls its private key, its incoming communication is secure. At any time a system can change its private key and publish the companion public key to replace its old public key.

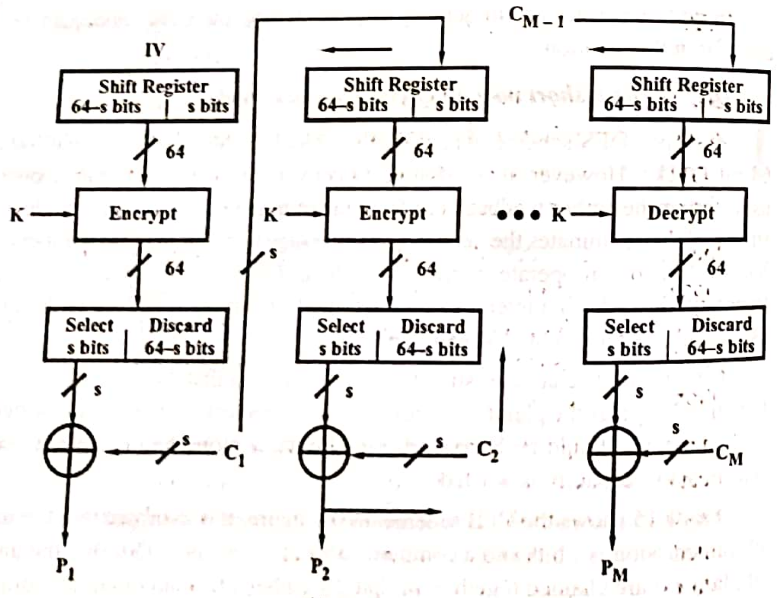
To discriminate between, conventional and public-key encryption, we will generally refer to the key used in symmetric encryption as a secret key. The two keys used for public-key encryption are referred to as the public key and private key. Invariably the private key is kept secret, but it is referred to as a private key rather than a secret key to avoid confusion with symmetric encryption.

Q.9. What are the principles of the public-key cryptosystems ?

Ans. The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption.



(a) Encryption



(b) Decryption

Fig. 1.15 s-bit Cipher Feedback (CFB) Mode

The input to the encryption function is a 64-bit shift register that is initially set to some initialization vector (IV). The leftmost (most significant) s bits of the output are XORed with the first segment of plaintext P_1 to produce the first unit of ciphertext C_1, which is then transmitted.

In addition, the contents of the shift register are shifted left by s bits and C_1 is placed in the rightmost (least significant) s bits of the shift register. This process continues until all plaintext units have been encrypted.

The same scheme is used for decryption, except that the received ciphertext unit is XORed with the output of the encryption function to produce the plaintext unit. It is noted that the encryption function that is used, not the decryption function. Let $S_s(X)$ be defined as the most significant s bits of X. Then

$$C_1 = P_1 \oplus S_s\,[E_k[IV]]$$

Therefore,

$$P_1 = C_1 \oplus S_s[E_K[IV]]$$

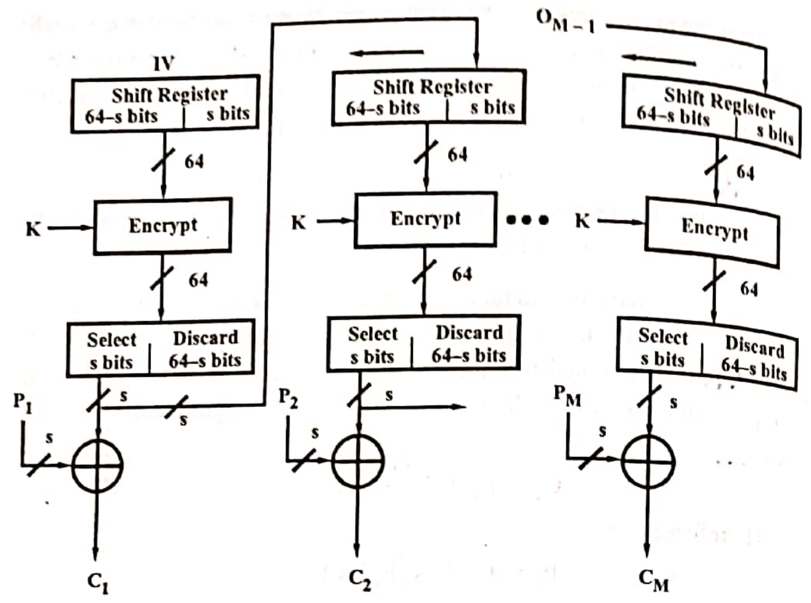
The same reasoning holds for subsequent steps in the process.

Q.52. Compare output feedback mode with cipher feedback mode.
(R.G.P.V., June 2015)

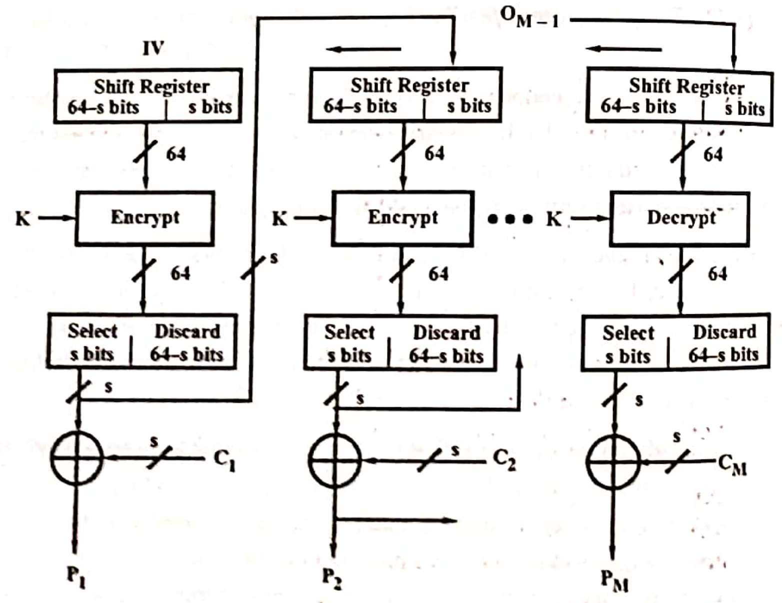
Ans. The output feedback (OFB) mode is similar in structure to that of CFB as shown in fig. 1.16. It is clearly seen from the figure that, the output of the encryption function is fed back to the shift register in OFB, whereas in CFB, the ciphertext unit is fed back to the shift register.

One advantage of the OFB method is that bit errors in transmission do not propagate. For example, if a bit error occurs in C_1, only the recovered value of P_1 is affected, subsequent plaintext units are not corrupted. With CFB, C_1 also serves as input to the shift register and therefore causes additional corruption downstream.

The disadvantage of OFB is that it is more vulnerable to a message stream modification attack than is CFB. Consider that complementing a bit in the ciphertext complements the corresponding bit in the recovered plaintext. Thus, controlled changes to the recovered plaintext can be made. This may make it possible for an opponent, by making the necessary changes to the checksum portion of the message as well as the data portion, to alter the ciphertext in such a way that it is not detected by an error-correcting code.



(a) Encryption

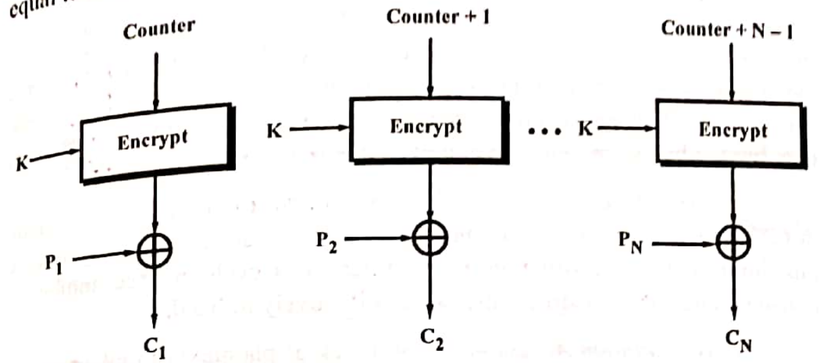


(b) Decryption

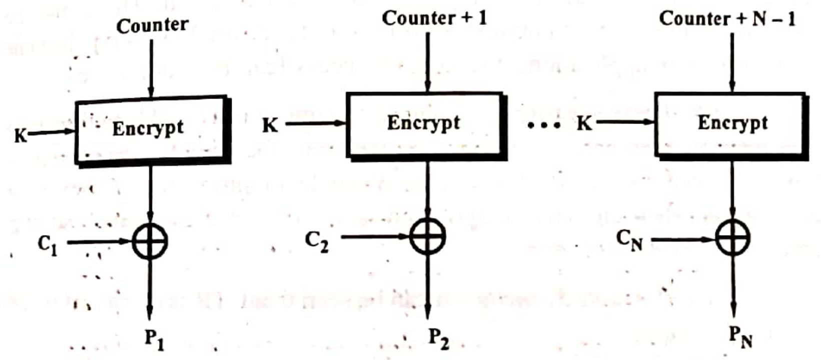
Fig. 1.16 s-bit Output Feedback (OFB) Mode

Q.53. What is counter mode ? List various advantages of CTR mode over other modes of operation. ***(R.G.P.V., Dec. 2009)***

Ans. Although interest in the counter mode (CTR) has increased recently, with applications to ATM (asynchronous transfer mode), network security and IP security. The scheme is illustrated in fig. 1.17. In this scheme, a counter, equal to the plaintext block size is used.



(a) Encryption



(b) Decryption

Fig. 1.17 Counter (CTR) Mode

The only requirement is that the counter value must be different for each plaintext block that is encrypted. The counter is initialized to some value and then incremented by 1 for each subsequent block (modulo 2^b, where b is the block size). For encryption, the counter is encrypted and then XORed with the plaintext block to produce the ciphertext block.

There is no chaining. For decryption, the same sequence of counter values is used, with each encrypted counter XORed with a ciphertext block to recover the corresponding plaintext block.

CTR mode has the following advantages –

(i) Simplicity – Unlike ECB and CBC modes, CTR mode requires only the implementation of the encryption algorithm and not the decryption algorithm.

(ii) Hardware Efficiency – Unlike the three chaining mode, in CTR mode encryption or decryption can be done in parallel on multiple blocks of plaintext or ciphertext. For the chaining modes, the algorithm must complete the computation on one block before beginning on the next block. This limits the maximum throughput of the algorithm to the reciprocal of the time for one execution of block encryption or decryption. In CTR mode, the throughput is only limited by the amount of parallelism that is achieved.

(iii) Software Efficiency – Similarly, because of the parallel execution in CTR mode, processors that support parallel features, such as aggressive pipelining, multiple instruction dispatch per clock cycle, a large number of registers, and SIMD instructions, can be effectively utilized.

(iv) Random Access – The ith block of plaintext of ciphertext can be processed in random-access fashion. With the chaining mode, block C_i cannot be computed until i – 1 prior block are computed. There may be applications in which a ciphertext is stored and it is desired to decrypt just one block. For such applications, the random access feature is attractive.

(v) Preprocessing – If sufficient memory is available and security is maintained, preprocessing can be used to prepare the output of the encryption boxes that feed into the XOR functions. When the plaintext or ciphertext input is presented, then the only computation is a series of XORs. Such strategy greatly enhances throughput.

(vi) Provable Security – It can be seen that CTR is at least as secure as the other modes.

Q.54. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption ? (R.G.P.V., June 2005)

Ans. The DES algorithm is a basic building block for providing data security. To apply DES in a variety of applications, four "modes of operation" have been defined. These four modes are intended to cover virtually all the possible applications of encryption for which DES could be used. As new applications and requirements have appeared, NIST has expanded the list of recommended modes to five in special publication 800-38A. These modes are intended for use with any symmetric block cipher, including triple DES and AES.

Some block cipher modes of operation only use encryption while others use both encryption and decryption to furnish different types of data security requirements. Various block cipher modes of operation are intended to support various types of applications. For detailed working of various types of applications. For detailed working of various block cipher modes of operation, refer to Q.49, Q.50, Q.51, Q.52 and Q.53 also.

Q.55. Explain such block cipher modes of operation which use any encryption, and why they use only encryption. Draw complete and clear diagrams of each. (R.G.P.V., Dec. 2011)

Or

Explain such block cipher modes of operation which use encryption and decryption. Draw complete and clear diagrams of each. (R.G.P.V., June 2012)

Ans. Refer to Q.49, Q.50, Q.51, Q.52 and Q.53.

Q.56. What are the advantages and disadvantages of DES ?

Ans. The advantages of DES are as follows –

(i) The security factors with respect to the fact that solving the discrete logarithm is very challenging

(ii) The shared key (i.e. the secret) is never itself transmitted over the channel.

The disadvantages of DES are as follows –

(i) The fact that there are expensive exponential operations involved, and the algorithm cannot be used to encrypt messages – it can be used for establishing a secret key only.

(ii) There is also a lack of authentication.

(iii) There is no identity of the parties involved in the exchange.

(iv) It is easily susceptible to man-in-the-middle attacks. A third party C, can exchange keys with both A and B, and can listen to the communication between A and B.

(v) The algorithm is computationally intensive. Each multiplication varies as the square of n, which must be very large. The number of multiplications required by the exponentiation increases with increasing values of the exponent, x or y in this case.

(vi) The computational nature of the algorithm could be used in a denial-of-service attack very easily.

Q.57. Write short note on stream cipher.

Ans. Stream ciphers typically operate on bits. The one-time pad is an example of a stream cipher. Practical stream ciphers typically generate a pseudo-random keystream as a function of a fixed length key and a per-message bit string. The key is known to both the sender and the receiver. The per-message string could be a message sequence number. Alternatively, it could be a random number generated by the sender and transmitted to the receiver along with the encrypted message. The ciphertext is itself obtained by performing an ⊕ operation between the plaintext and the keystream. An example of a stream cipher is RC4 used in the wireless LAN protocol, IEEE 802.11. Stream ciphers are usually faster than block ciphers and use less complicated circuits. However, RC4 and some other stream ciphers have been shown to be vulnerable to attack. Their use has not been a widespread as block ciphers.

Q.58. Differentiate between block cipher and stream cipher.
(R.G.P.V., Dec. 2011, June 2012)

Ans. Refer to Q.32 and Q.57.

●●

UNIT 2

ADVANCED ENCRYPTION STANDARD (AES), INTRODUCTION TO PUBLIC KEY CRYPTOSYSTEM, DISCRETE LOGARITHMIC PROBLEM, DIFFIE-HELLMAN KEY EXCHANGE COMPUTATIONAL & DECISIONAL DIFFIE-HELLMAN PROBLEM

Q.1. What is the advanced encryption standard (AES) ?

Ans. Advanced encryption standard (AES) is a fast symmetric cryptosystem for mass encryption. It was developed through competition, and is based on the ***RIJNDAEL*** system, published in 1999 by Joan Daemen and Vincent Rijmen from Belgium. AES replaced the old DES system published in 1975.

AES works on bit symbols, so the residue classes (bits) 0 and 1 of Z_2 can be considered as plaintext and cryptotext symbols. The workings of ***RIJNDAEL*** can be described using the field F_{2^8} and its polynomial ring $F_{2^8}(z)$. To avoid confusion we use z as the dummy variable in the polynomial ring and x as the dummy variable for polynomials in Z_2 needed in defining and representing the field F_{2^8}. Furthermore, we denote addition and multiplication in F_{2^8} by ⊕ and ⊙, the identity element is denoted by **1** and the zero element by **0**. Note that because $1 = -1$ in Z_2, the additional inverse of an element in $Z_2(x)$, F_{2^8} and in $F_{2^8}(z)$ is the element itself. So subtraction ⊖ is the same as addition ⊕, in this case.

Construction – In the ***RIJNDAEL*** system the length l_B of the plaintext block and the length l_K of the key are independently either 128, 192 or 256 bits. Dividing by 32 we get the numbers.

$$N_B = \frac{l_B}{32} \text{ and } N_K = \frac{l_K}{32}.$$

Bits are handled as bytes of 8 bits. An 8-bit byte $b_7b_6 \ldots\ldots b_0$ can be considered as an element of the finite field F_{2^8}.

The key is usually expressed as a $4 \times N_K$ matrix whose elements are bytes. If the key is, byte by byte,

$$\mathbf{k} = k_{00}k_{10}k_{20}k_{30}k_{01}k_{11}k_{21}\ldots\ldots k_{3,N_K-1}$$

then the corresponding matrix is

$$K = \begin{pmatrix} k_{00} & k_{01} & k_{02} & \cdots & k_{0,N_K-1} \\ k_{10} & k_{11} & k_{12} & \cdots & k_{1,N_K-1} \\ k_{20} & k_{21} & k_{22} & \cdots & k_{2,N_K-1} \\ k_{30} & k_{31} & k_{32} & \cdots & k_{3,N_K-1} \end{pmatrix}$$

Note how the elements of the matrix are indexed starting from zero. Similarly, if the input block (plaintext block) is, byte by byte,

$$a = a_{00}a_{10}a_{20}a_{30}a_{01}a_{11}a_{21}.....a_{3,N_B-1}$$

then the corresponding matrix is

$$A = \begin{pmatrix} a_{00} & a_{01} & a_{02} & \cdots & a_{0,N_B-1} \\ a_{10} & a_{11} & a_{12} & \cdots & a_{1,N_B-1} \\ a_{20} & a_{21} & a_{22} & \cdots & a_{2,N_B-1} \\ a_{30} & a_{31} & a_{32} & \cdots & a_{3,N_B-1} \end{pmatrix}$$

During encryption we are dealing with a bit sequence of length l_B, the so-called state. Like the block, it is also expressed byte by byte in the form of a $4 \times N_B$ matrix.

$$S = \begin{pmatrix} s_{00} & s_{01} & s_{02} & \cdots & s_{0,N_B-1} \\ s_{10} & s_{11} & s_{12} & \cdots & s_{1,N_B-1} \\ s_{20} & s_{21} & s_{22} & \cdots & s_{2,N_B-1} \\ s_{30} & s_{31} & s_{32} & \cdots & s_{3,N_B-1} \end{pmatrix}$$

Elements of the matrices K, A and S are bytes of 8 bits, which can be interpreted as elements of the field F_{2^8}. In this way these matrices are matrices over this field. Another way to interpret the matrices is to consider their columns as sequences of elements of the field F_{2^8} of length 4. These can be interpreted further, from top to bottom, as coefficients of polynomials with maximum degree 3 from the polynomial ring $F_{2^8}(z)$. So, the state S mentioned above would thus correspond to the polynomial sequence.

$$s_{00} \oplus s_{10}z + s_{20}z^2 \oplus s_{30}z^3, s_{01} \oplus s_{11}z \oplus s_{21}z^2 \oplus s_{31}z^3, ...,$$
$$s_{0,N_{B-1}} \oplus s_{1,N_{B-1}} z \oplus s_{2,N_{B-1}} z^2 \oplus s_{3,N_{B-1}} z^3.$$

For the representation to be unique, a given fixed irreducible polynomial of degree 8 from $Z_2(x)$ must be used in the construction of F_{2^8}. In *RIJNDAEL* it is the so-called *RIJNDAEL* polynomial.

$$p(x) = 1 + x + x^3 + x^4 + x^8.$$

Q.2. Define the term rounds in RIJNDAEL.

Ans. There is a certain number N_R of so-called ***rounds*** in ***RIJNDAEL.*** The number of rounds is given by the following table –

N_R	$N_B = 4$	$N_B = 6$	$N_B = 8$
$N_K = 4$	10	12	14
$N_K = 6$	12	12	14
$N_K = 8$	14	14	14

The i^{th} round receives as its input the current state S and its own so-called round key $\mathbf{R_i}$. In particular, we need the initial round key $\mathbf{R_0}$. In each round, except for the last one, we go through the following sequence of operations –

$S \leftarrow$ SubBytes(S)

$S \leftarrow$ ShiftRows(S)

$S \leftarrow$ MixColumns(S)

$S \leftarrow$ AddRoundKey(S, $\mathbf{R_i}$)

The last round is the same except that we drop MixColumns.

The encrypting key is expanded first and then used to distribute round keys to all rounds. This and the different operations in rounds are discussed one by one in the following sections. Encrypting itself then consists of the following steps –

(i) Initialize the state – $S \leftarrow$ AddRoundKey(A, $\mathbf{R_0}$)

(ii) $N_R - 1$ "usual" rounds.

(iii) The last round.

When decrypting we go through the inverse steps in reverse order.

Q.3. Define the following terms –
(i) SubBytes (transforming bytes) ***(ii) ShiftRows***
(iii) MixColumns ***(iv) AddRoundKey.***

Ans. (i) SubBytes (Transforming Bytes) – In this operation each byte s_{ij} of the state is transformed in the following way –

(a) Interpret s_{ij} as an element of the field F_{2^8} and compute its inverse s_{ij}^{-1}. It is agreed here that the inverse of the zero element is the element itself.

(b) Expand s_{ij}^{-1} in eight bits $b_7b_6b_5b_4b_3b_2b_1b_0$, denote

$$b(x) = b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7$$

[a polynomial in $Z_2(x)$]

and compute

$$b'(x) \equiv b(x)\,(1 + x + x^2 + x^3 + x^4) + (1 + x + x^5 + x^6) \bmod 1 +$$

The result

$$b'(x) = b'_0 + b'_1x + b'_2x^2 + b'_3x^3 + b'_4x^4 + b'_5x^5 + b'_6x^6 + b'_7x^7$$

is interpreted as a byte $b'_7b'_6b'_5b'_4b'_3b'_2b'_1b'_0$ or as an element of F_{2^8}. By the way, division by $1 + x^8$ in $Z_2(x)$ is easy since

$$x^k \equiv x^{(k \bmod 8)} \bmod 1 + x^8.$$

The operation in #2 may also be done by using matrices. We then apply an affine transformation in Z_2 –

$$\begin{pmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Byte transformation is done in reverse order during the decryption. Because in $Z_2(x)$

$$1 = \gcd(1 + x + x^2 + x^3 + x^4, 1 + x^8)$$

(easy to verify using the Euclidean algorithm), the polynomial $1 + x + x^2 + x^3 + x^4$ has an inverse modulo $1 + x^8$ and the occurring 8×8 matrix is invertible modulo 2. This inverse is $x + x^3 + x^6$.

Transforming the byte is in all a nonlinear transformation, which can be given in one table, the so-called RIJNDAEL S-box. This table can be found for example in MOLLIN and STINSON.

(ii) ShiftRows – In this operation the elements of the rows of the matrix representation of the state are shifted left cyclically in the following way –

shift	row 0	row 1	row 2	row 3
$N_B = 4$	no shift	1 element	2 elements	3 elements
$N_B = 6$	no shift	1 element	2 elements	3 elements
$N_B = 8$	no shift	1 element	3 elements	4 elements

While decrypting rows are correspondingly shifted right cyclically.

(iii) MixColumns – In this transformation columns of the state matrix are interpreted as polynomials of maximum degree 3 in the polynomial ring $F_{2^8}(z)$. Each column (polynomial) is multiplied by the fixed polynomial.

$$c(z) = c_0 \oplus c_1z \oplus c_2z^2 \oplus c_3z^3 \in F_2(z)$$

modulo $1 \oplus z^4$ where

$$c_0 = x,\ c_1 = c_2 = 1 \text{ and } c_3 = 1 + x$$

Dividing by the polynomial $1 \oplus z^4$ in $F_{2^8}(z)$ is especially easy since

$$z^k \equiv z^{(k, \bmod 4)} \bmod 1 \oplus z^4$$

Alternatively the operation can be considered as a linear transformation of F_{2^8} –

$$\begin{pmatrix} s'_{0i} \\ s'_{1i} \\ s'_{2i} \\ s'_{3i} \end{pmatrix} = \begin{pmatrix} c_0 & c_3 & c_2 & c_1 \\ c_1 & c_0 & c_3 & c_2 \\ c_2 & c_1 & c_0 & c_3 \\ c_3 & c_2 & c_1 & c_0 \end{pmatrix} \begin{pmatrix} s_{0i} \\ s_{1i} \\ s_{2i} \\ s_{3i} \end{pmatrix}$$

When decrypting we divide by the polynomial c(z) modulo $1 \oplus z^4$. Although $1 \oplus z^4$ is not an irreducible polynomial of $F_{2^8}(z)^1$, c(z) has an inverse modulo $1 \oplus z^4$, because

$$1 = \gcd(c(z), 1 \oplus z^4)$$

The inverse is obtained using the Euclidean algorithm (hard to compute!) and it is

$$d(z) = d_0 \oplus d_1z \oplus d_2z^2 \oplus d_3z^3$$

where $d_0 = x + x^2 + x^3$, $d_1 = 1 + x^3$, $d_2 = 1 + x^2 + x^3$ and $d_3 = 1 + x + x^3$.

So, when decrypting the column (polynomial) is multiplied by d(z) modulo $1 \oplus z^4$ and the operation is thus no more complicated than when encrypting. In matrix form in F_{2^8}.

$$\begin{pmatrix} s_{0i} \\ s_{1i} \\ s_{2i} \\ s_{3i} \end{pmatrix} = \begin{pmatrix} d_0 & d_3 & d_2 & d_1 \\ d_1 & d_0 & d_3 & d_2 \\ d_2 & d_1 & d_0 & d_3 \\ d_3 & d_2 & d_1 & d_0 \end{pmatrix} \begin{pmatrix} s'_{0i} \\ s'_{1i} \\ s'_{2i} \\ s'_{3i} \end{pmatrix}$$

(iv) AddRoundKey – The round key is as long as the state. In this operation the round key is added to the state byte by byte modulo 2. The inverse operation is the same.

Q.4. Describe the operating mode of AES.

Ans. The usual way of using AES is to encrypt one long message block at a time with the same key, the so-called *ECB mode* (electronic codebook).

Another way, the so-called *CBC mode* (cipher block chaining), is to always form a sum of a message block w_i and the preceding cryptoblock c_{i-1} bit by bit modulo 2, i.e. $w_i \oplus c_{i-1}$, and encrypt it, using the same key k all the time. In the beginning we need an initial (crypto) block. Schematically *CBC mode* is the following operation –

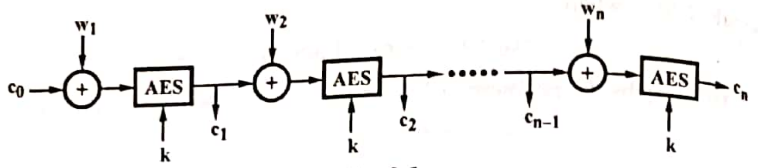


Fig. 2.1

A change in a message block causes changes in the following cryptoblocks in *CBC mode*. This way *CBC mode* can be used for authentication or the so-called MAC (message authentication code) in the following way. The initial block can e.g. be formed of just 0-bits. The sender has a message that is formed of message blocks w_1, ..., w_n and he/she computes, using CBC mode, the corresponding cryptoblocks c_1, ..., c_n applying a secret key k. The sender sends the message blocks and c_n to the receiver. The receiver also has the key k and he/she can check whether the c_n is valid by using the key.

In the so-called *OFB mode* (output feedback) *AES* is used to transform the key in a procedure similar to *ONE-TIME-PAD* encrypting. Starting from a certain "initial key" k_0 we get a key stream k_1, ..., k_n by encrypting this key over the over using *AES*, k_1 is obtained by encrypting k_0. Again, when encrypting we use the same secret key k all the time. Schematically –

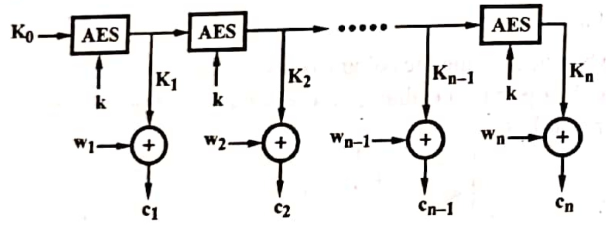


Fig. 2.2

OFB mode gives rise to a variant, the so-called *CFB mode* (cipher feedback), where the key k_i of the key stream is formed by encrypting the preceding cryptoblock. Again k_1 is obtained by encrypting the initial block c_0.

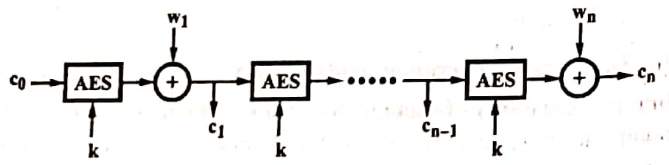


Fig. 2.3

This variant can be used for authentication much as the *CBC-mode*, which it also otherwise resembles.

There are also other modes, for example the so-called *CTR mode* (counter mode).

Q.5. What do you understand by public key cryptography ?

Ans. The development of public key cryptography is the greatest and perhaps the only true revolution in the entire history of cryptography. Public-key cryptography provides a radical departure from all that has gone before. It is based on mathematical functions rather than substitution or permutation.

Public-key cryptography was developed by Diffie and Hellman. This technique is also known as Asymmetric Encryption. The concept is simple. There are two keys, one is held privately and the other one is made public. What one key can lock, the other key can unlock. The use of two keys has profound consequences in the areas of confidentiality, key distribution and authentication.

Q.6. What is PKCS ?

Ans. The public-key cryptography standards (PKCS) are specifications produced by RSA Laboratories in cooperation with secure systems developers worldwide for the purpose of accelerating the deployment of public-key cryptography. First published in 1991 as a result of meetings with a small group of early adopters of public-key technology, the PKCS documents have become widely referenced and implemented. Contributions from the PKCS series have become part of many formal and defacto standards, including ANSI X9 documents, PKIX, SET, S/MIME and SSL.

Q.7. What are the misconceptions concerning public-key encryption ? Justify all of them. ***(R.G.P.V., Dec. 2006)***

Ans. The several common misconceptions concerning public-key encryption are mentioned below –

(i) One such misconception is that public-key encryption is more secure from cryptanalysis than is symmetric encryption. Such a claim was made, for example, in a famous article in ***Scientific American*** by Gardner [GARD 77]. In fact, the security of any encryption scheme depends on the length of the key and the computational work involved in breaking a cipher. There is nothing in principle about either symmetric or public-key encryption that makes one superior to another from the point of view of resisting cryptanalysis.

(ii) A second misconception is that public-key encryption is a general-purpose technique that has made symmetric encryption obsolete. On the contrary, because of the computational overhead of current public-key encryption schemes, there seems no foreseeable likelihood that symmetric encryption will be abandoned. As one of the inventors of public-key encryption has put it [DIFF 88], "the restriction of public-key cryptography to key management and signature applications is almost universally accepted".

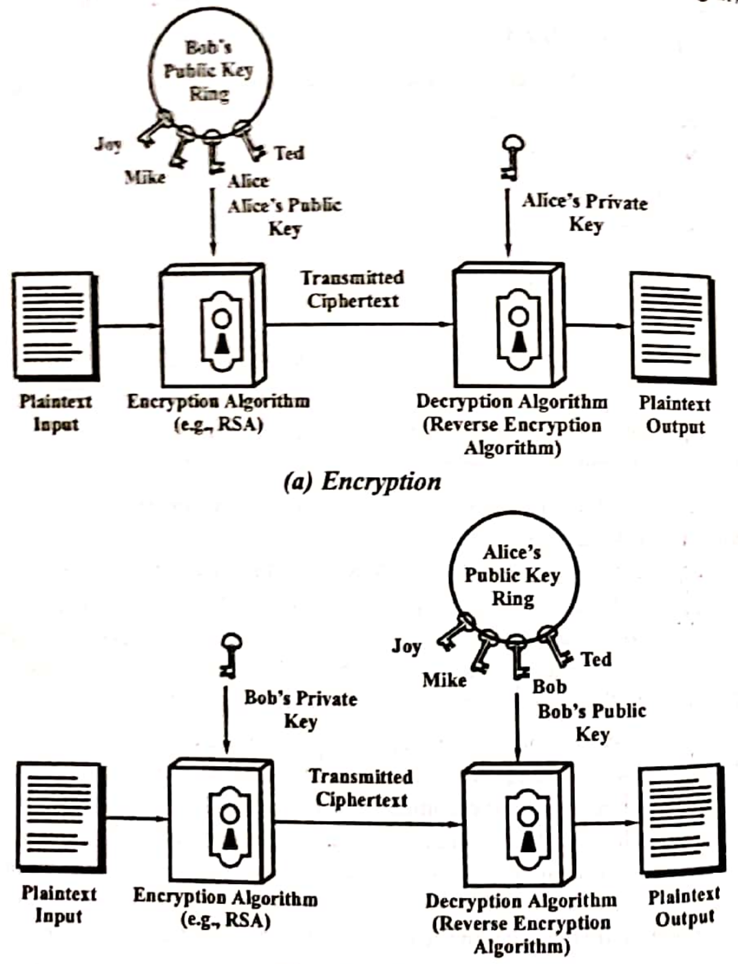
(iii) Finally, there is a feeling that key distribution is trivial whe using public-key encryption, compared to the rather cumbersome handshakin involved with key distribution centers for symmetric encryption. In fact, some form of protocol is needed, generally involving a central agent, and the procedures involved are no simpler nor any more efficient than those required for symmetric encryption.

Q.8. Explain public-key cryptosystems.

Or

What are principal elements of public key cryptosystem ? What are the roles of the public key and private key ? **(R.G.P.V., Dec. 2006, 2008)**

Ans. A public-key encryption scheme has six ingredients as shown in fig. 2.4.



(b) Authentication

Fig. 2.4 Public-key Cryptography

(i) ***Plaintext*** – This is the readable message or data that is fed into the algorithm as input.

(ii) ***Encryption Algorithm*** – The encryption algorithm performs various transformation on the plaintext.

(iii) ***Public and Private Key*** – This is a pair of keys, one is used for encryption, and the other is used for decryption. The exact transformations performed by the encryption algorithm depend on the public or private key that is provided as input.

(iv) ***Ciphertext*** – This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.

(v) ***Decryption Algorithm*** – This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

The essential steps of the algorithm are as follows –

(i) Each user generates a pair of keys to be used for the encryption and decryption of messages.

(ii) Each user places one of the two keys in a public register or other accessible file. This key is known as ***public key***. The companion key is kept private. This key is known as ***private key***. As suggested in fig. 2.4, each user maintains a collection of public keys obtained from others.

(iii) In the figure, if Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.

(iv) When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows her private key.

With this approach, all users have access to public keys, and private keys are generated locally by each user and therefore need never be distributed. As long as a system controls its private key, its incoming communication is secure. At any time a system can change its private key and publish the companion public key to replace its old public key.

To discriminate between, conventional and public-key encryption, we will generally refer to the key used in symmetric encryption as a secret key. The two keys used for public-key encryption are referred to as the public key and private key. Invariably the private key is kept secret, but it is referred to as a private key rather than a secret key to avoid confusion with symmetric encryption.

Q.9. What are the principles of the public-key cryptosystems ?

Ans. The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption.

The first problem is that of key distribution. It requires either

(i) that two communicants already share a key, which somehow has been distributed to them, or

(ii) the use of a key distribution center.

Whitfield Diffie and Martin Hellman reasoned that this second requirement negated the very essence of cryptography the ability to maintain total secrecy over our own communication.

The second problem that Diffie pondered, and one that was apparently unrelated to the first was that of "digital signatures". If the use of cryptography was to become widespread, not just in military situations but for commercial and private purposes, then electronic messages and documents would need the equivalent of signatures used in paper documents. That is, could a method be devised that would stipulate, to the satisfaction of all parties, that a digital message had been sent by a particular person ? This is a somewhat broader requirement than that of authentication , and its characteristics and ramifications.

Diffie and Hellman achieved an astounding break through in 1976 by coming up with a method (i.e. public-key encryption algorithm) that addressed both problems and that was radically different from all previous approaches to cryptography.

Q.10. Write the difference between conventional encryption and public-key encryption. ***(R.G.P.V., June 2015)***

Or

Compare conventional encryption and public-key encryption. ***(R.G.P.V., Dec. 2006)***

Or

What is the fundamental difference between symmetric and asymmetric encryption ? ***(R.G.P.V., June 2011)***

Or

Distinguish between symmetric and asymmetric key cryptography. ***(R.G.P.V., June 2014)***

Ans. The major differences between conventional and public-key encryption are as follows –

S.No.	*Conventional Encryption*	*Public-key Encryption*
(i)	The same algorithm with the same key is used for encryption and decryption.	One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption.
(ii)	The sender and receiver must share the algorithm and the key.	The sender and receiver must each have one of the matched pair of keys (not the same one).
(iii)	The key must be kept secret.	One of the two keys must be kept secret.
(iv)	It must be impossible or at least impractical to decipher a message if no other information is available.	It must be impossible or at least impractical to decipher a message if no other information is available.
(v)	Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Q.11. What is public-key cryptography ? Explain. Bring out the difference between conventional encryption and public-key encryption. ***(R.G.P.V., June 2010)***

Ans. **Public-key Cryptography** – Refer to Q.5.

Difference – Refer to Q.10.

Q.12. What are the principles of the public-key cryptosystems ? Differentiate conventional encryption and public-key encryption. ***(R.G.P.V., June 2009)***

Ans. **Principles** – Refer to Q.9.

Difference – Refer to Q.10.

Q.13. What are the three broad categories of applications of public-key cryptosystems ? ***(R.G.P.V., June 2013, 2016, May 2019)***

Ans. Public-key systems are characterized by the use of a cryptographic type of algorithm with two keys, one held private and one available publicly. Depending on the application, the sender uses either the sender's private key or the receiver's public key, or both, to perform some type of cryptographic function. In broad terms, we can classify the use of public-key cryptosystems into three categories –

(i) Encryption/Decryption – The sender encrypts a message with the recipient's public key.

(ii) Digital Signature– The sender "signs" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message.

(iii) Key Exchange – Two sides cooperate to exchange a session key. Several different approaches are possible, involving the private key of one or both parties.

Some algorithms are suitable for all three applications, whereas others can be used only for one or two of these applications. Table 2.1 indicates the applications supported by the algorithms given in the table.

Table 2.1 Applications for Public-key Cryptosystems

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Q.14. What requirements must a public-key cryptosystem fulfil to be a secure algorithm ?

Or

What requirements must a public-key cryptosystem fulfil to be a secure algorithm ? Briefly explain each of them with example. (R.G.P.V., June 2008)

Ans. The cryptosystem depends on a cryptographic algorithm based on two related keys. Diffie and Hellman postulated this system without demonstrating that such algorithm exist. However, they did lay out the conditions that such algorithm must fulfil –

(i) It is computationally easy for a recipient B to generate a pair of key (public key KU_b and private key KR_b).

(ii) It is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding ciphertext –

$$C = E_{KU_b}(M)$$

(iii) It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message –

$$M = D_{KR_b}(C) = D_{KR_b}\left[E_{KU_b}(M)\right]$$

(iv) It is computationally infeasible for an opponent, knowing the public key, KU_b, to determine the private key, KR_b.

(v) It is computationally infeasible for an opponent, knowing the public key, KU_b, and a ciphertext, C, to recover the original message, M.

A sixth requirement can be added that, although useful, is not necessary for all public-key applications.

(vi) The encryption and decryption functions can be applied in either order –

$$M = E_{KU_b}\left[D_{KR_b}(M)\right] = D_{KU_b}\left[E_{KR_b}(M)\right]$$

These are the formidable requirements, as evidenced by the fact that only two such algorithms (RSA and Elliptic curve cryptography) have received widespread acceptance.

Q.15. How does asymmetric key encryption ensure "Non-repudiation" ? Explain with an example. (R.G.P.V., Dec. 2007)

Ans. Let us assume that the public-key encryption and decryption algorithms have the property that E(D(P)) = P in addition to the usual property that D(E(P)) = P. Assuming that this is the case, Alice can send a signed plaintext message, P, to Bob by transmitting $E_B(D_A(P))$. Note carefully that Alice knows her own (private) decryption key, D_A, as well as Bob's public key, E_B, so constructing this message is something Alice can do.

When Bob receives the message, he transforms it using his private key, as usual, yielding $D_A(P)$, as shown in fig. 2.5. He stores this text in a safe place and then decrypts it using E_A to get the original plaintext.

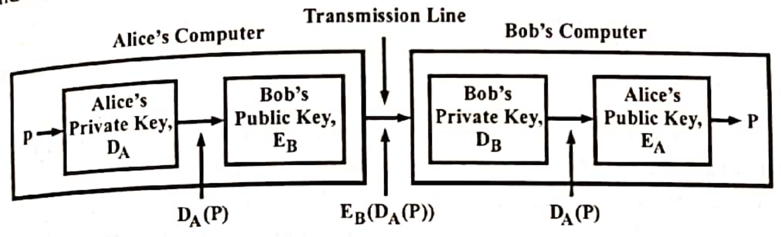


Fig. 2.5 Digital Signatures Using Public-key Cryptography

To see how the signature property works, suppose that Alice subsequently denies having sent the message P to Bob. When the case comes up in court, Bob can produce both P and $D_A(P)$. The judge can easily verify that Bob indeed has a valid message encrypted by D_A by simply applying E_A to it. Since Bob does not know what Alice's private key is, the only way Bob could have acquired a message encrypted by it is if Alice did indeed sent it.

Q.16. What are six components of public key infrastructure (PKI) ? (R.G.P.V., June 2011)

Ans. Following are the components of public key infrastructure –

(i) Certification Authority – The certification authority (CA) takes responsibility for identifying the correctness of the identity of the person asking for a certificate to be issued, and ensures that the information contained within the certificate is correct and digitally signs it.

(ii) Revocation – In revocation, system depends upon publishing certificates so that people are able to communicate with each other, there has to be a system for letting people know when certificates are no longer valid.

(iii) Registration Authority – A CA may use a third-party registration authority (RA) to perform the necessary checks on the person on company requesting the certificate to ensure that they are who they say they are. That

RA may appear to the certificate requestor as a CA, but they do not actually sign the certificate that is issued.

(iv) Certificate Publishing Methods – One of the fundamentals of PKI systems is the need to publish certificates so that users can find them. This can be performed in two ways. One is to publish certificates in the equivalent of an electronic telephone directory. The other is to send your certificate out of those people you think might need it by one means or another.

(v) Certificate Management System – This term refers to the management system by which certificates are published, temporarily or permanently suspended, renewed or revoked. Certificate management systems do not normally delete certificates because it may be necessary to prove their status at a point in time, perhaps for legal reasons. A CA will run certificate management systems to be able to keep track of their responsibilities and liabilities.

(vi) PKI Aware Applications – This term usually refers to applications that have had a particular CA software supplier's toolkit added to them so that they are able to use the suppliers CA and certificates to implement PKI functions. The term does not mean that the applications have any knowledge built into them about what the security requirements really are, or which PKI services are relevant to delivering them. These issues are quite separate from having PKI services available.

Q.17. What drawbacks to symmetric and asymmetric encryption are resolved by using a hybrid method like Diffie-Hellman ? (R.G.P.V., June 2011)

Ans. **Problems with Symmetric Encryption** – The biggest problem with symmetric encryption is that a single key must be shared in pairs of each sender and receiver. In a distributed environment with large numbers of combination pairs involved in many-to-one communication topology, it is hard for the one recipient to keep so many keys in order to support all communication.

Besides the key distribution problem above, the size of the communication space creates problems. The secret-key cryptography, if strictly used, needs billions of secret keys pairs to be created, shared, and stored because of the massive potential number of individuals who can carry on communication in a many-to-one, one-to-many, and many-to-many topologies supported by the Internet for example. Large numbers of potential correspondents in the many-to-one, one-to-many, and many-to-many communication topologies may cause symmetric encryption to fail due to its requirement of prior relationships with the parties to establish the communication protocols such as the setting up of and acquisition of the secret key.

The following additional problems are also observable besides the problems discussed above and a result of them –

(i) The secret kay may not be changed frequently enough to ensure confidentiality.

(ii) The integrity of data can be compromised because the receiver cannot verify that the message has not been changed before receipt.

(iii) The method does not provide a way to ensure secrecy even if the encryption process is compromised.

(iv) It is possible for the sender to repudiate the message because there are no mechanisms for the receiver to make sure that the message has been sent by the claimed sender.

Problems with Public Key Encryption (Asymmetric Encryption) – The biggest problem with public key encryption is speed. Public key algorithms are slower than symmetric algorithms. This is because public key calculations take longer than symmetric key calculations since they involve the use of exponentiation of very large numbers.

Besides speed, public key encryption algorithms have a potential to suffer from the man-in-the-middle attack.

Q.18. Briefly explain public-key cryptanalysis.

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

Another form of attack is to find some way to compute the private key given the public key. It has not been proven that this form of attack is infeasible for a particular public key algorithm. Thus, any given algorithm including the widely used RSA algorithm, is suspect.

Finally, there is a form of attack that is peculiar to public-key systems. In essence, this is a probable message attack. For example, that a message were to be sent that consisted solely of a 56-bit DES key. An opponent could encrypt all possible keys using the public key and could decipher any message by matching the transmitted ciphertext. Thus no matter how large the key size of the public-key approach, the attack is reduced to a brute-force attack on a 56-bit key. This attack can be thwarted by appending some random bits to such simple messages.

Q.19. Discuss about the discrete logarithm problem.

Ans. There is a famous problem in mathematics known as the discrete logarithm problem (DLP) which has been very well used in the world of cryptography. The DLP has the potential of being a very difficult problem to solve and so cryptographers have created ciphers in which cracking the system would require solving the DLP.

Given a positive integer modulus n, and two positive integers s and $t = s^m$, both reduced modulo n, find m. We call the smallest positive integer m such that $t \equiv s^m \pmod n$ the discrete logarithm base s of t modulo n.

Example – Find the discrete logarithm base 5 of 2 modulo 7. That is, we want to find the smallest integer m so that $5^m \equiv 2 \pmod 7$. With a small amount of trial and error we can find that $5^4 \equiv 2 \pmod 7$ so $m = 4$.

When working in the standard real numbers, solving logarithms is a very well understood problem. We can use series to accurately solve for or give good approximations for real valued logarithms, and so for a computer this would be considered an "easy" problem. When working in the finite group of $(Z/pZ)^*$ for an odd prime p, the discrete log problem (DLP) can be a very difficult problem to solve. In particular if we choose $s = g$ where g is a primitive root modulo p then solving the DLP becomes extremely difficult, especially as p becomes very large. The intuition behind why this particular problem is so hard is that since g is primitive root modulo p then, by its definition, every integer a which is relatively prime to p can be expressed as $g^k \equiv a \pmod p$ for some positive integer $k \in (1, \phi(p))$ where $\phi(p) = p - 1$. Since p is a large prime then all integers 1, ..., p – 1 are relatively prime to p. Thus for a random exponent $k \in (1, p-1)$, g^k has an equal probability of being equivalent modulo p to any $a \in (1, p-1)$. As p then becomes very large, the probability of choosing the correct exponent, k, for which $g^k \equiv t \pmod p$ for a given integer t is $\frac{1}{p-1}$ which is extremely small.

To date, there are no known "fast" algorithms which can solve this DLP. Because of this difficulty, cryptographers have developed ciphers which are based upon the DLP. That is, they have developed systems in which, in order to crack the system, one would need to be able to solve the DLP.

Q.20. Describe Diffie-Hellman key exchange algorithm.
(R.G.P.V., Dec. 2004, June 2015)

Or

Briefly discuss Diffie-Hellman key exchange scheme.
(R.G.P.V., May/June 2006, May 2018)

Or

What is the purpose of Diffie-Hellman public key technique ? Briefly describe its algorithm. ***(R.G.P.V., June 2008, Dec. 2008)***

Or

Explain Diffie-Hellman key exchange algorithm. ***(R.G.P.V., June 2010)***

Or

Briefly explain Diffie-Hellman key exchange.
(R.G.P.V., Dec. 2003, June 2004)

Or

Explain Diffie Hellman key exchange algorithm using flowchart and an example. ***(R.G.P.V., Dec. 2011)***

Or

Write short note on Diffie Hellman key exchange. (R.G.P.V., June 2017)

Ans. Diffie-Hellman key exchange is a public-key algorithm. The purpose of the algorithm is to enable two users to exchange a key securely that can then be used for subsequent encryption of messages. The algorithm is limited to the exchange of keys. The Diffie-Hellman algorithm depends for its effectiveness on the difficulty of computing discrete logarithms.

We can define the Diffie-Hellman key exchange algorithm as follows –

(i) ***Global Public Elements*** – For this scheme there are two publicly known numbers a prime number q and an integer α that is a primitive root of q and $\alpha < q$.

Suppose users A and B wish to exchange a key.

(ii) ***User A Key Generation*** – User A selects a random integer $X_A < q$ as his private key and calculate his public key Y_A where $Y_A = \alpha^{X_a} \bmod q$.

(iii) ***User B Key Generation*** – User B also independently selects a random integer $X_B < q$ as his private key and computes public key Y_B where $Y_B = \alpha^{X_B} \bmod q$.

Now both A and B exchange their public keys and each compute the common secret key.

(iv) ***Generation of Secret Key by User A*** – User A computes the key as $K = (Y_B)^{X_A} \bmod q$.

(v) ***Generation of Secret Key by User B*** – User B computes the key as $K = (Y_A)^{X_B} \bmod q$. The two calculations (in step (iv) and (v)) produce the identical results as –

$$K = (Y_B)^{X_A} \bmod q \qquad \text{(secret key of A)}$$
$$= \left(\alpha^{X_B} \bmod q\right)^{X_A} \bmod q$$

$$= (\alpha^{X_B})^{X_A} \bmod q$$
$$= \alpha^{X_B X_A} \bmod q$$
$$= (\alpha^{X_A})^{X_B} \bmod q$$
$$= (\alpha^{X_A} \bmod q)^{X_B} \bmod q$$
$$= (Y_A)^{X_B} \bmod q \qquad \text{(secret key of B)}$$

Thus two sides have exchanged the secret key.

The security of the Diffie-Hellman key exchange lies in the fact that, while it is relatively easy to calculate exponentials modulo prime, it is very difficult to calculate discrete logarithms.

Fig. 2.6 shows a simple protocol that makes use of the Diffie-Hellman calculation.

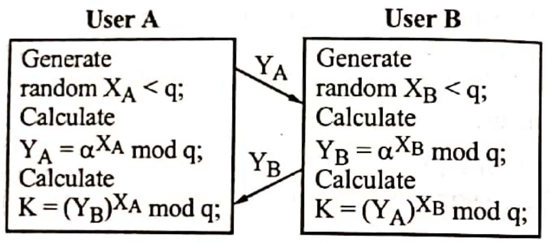

User A
User B
Generate random $X_A < q$; Calculate $Y_A = \alpha^{X_A} \bmod q$; Calculate $K = (Y_B)^{X_A} \bmod q$;
Y_A
Y_B
Generate random $X_B < q$; Calculate $Y_B = \alpha^{X_B} \bmod q$; Calculate $K = (Y_A)^{X_B} \bmod q$;

Fig. 2.6

Example –

(i) Take prime number $q = 353$

and its primitive root $\alpha = 3$

(ii) Select $X_A = 97$

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

(iii) Select $X_B = 233$

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

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

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

$= 248^{97} \bmod 353 = 160$

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

$= 40^{233} \bmod 353 = 160$

As a result, the two sides have exchanged the secret key.

Q.21. Write short note on meet-in-the-middle attacks.

(R.G.P.V., June 2012)

Ans. This protocol has a weakness. Eve does not have to find the value of x and y to attack the protocol. She can fool Alice and Bob by creating two keys – one between herself and Alice, and another between herself and Bob. Fig. 2.7 shows this situation.

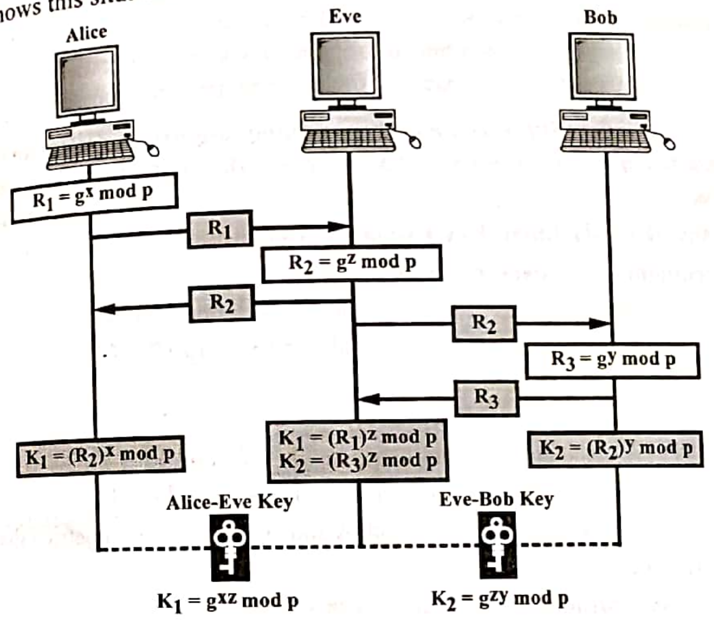


Fig. 2.7 Man-in-the-middle Attack

The following can happen –

(i) Alice chooses x, calculates $R_1 = g^x \bmod p$, and sends R_1 to Bob.

(ii) Eve the intruder, intercepts R_1. She chooses z, calculates $R_2 = g^z \bmod p$, and sends R_2 to both Alice and Bob.

(iii) Bob chooses y, calculates $R_3 = g^y \bmod p$, and sends R_3 to Alice. R_3 is intercepted by Eve and never reaches Alice.

(iv) Alice and Eve calculate $K_1 = g^{xz} \bmod p$, which becomes a shared key between Alice and Eve. Alice, however, thinks that it is a key shared between Bob and herself.

(v) Eve and Bob calculate $K_2 = g^{zy} \bmod p$, which becomes a shared key between Eve and Bob. Bob, however, thinks that it is a key shared between Alice and himself.

In other words, two keys, instead of one, are created – one between Alice and Eve, and another between Eve and Bob. When Alice sends data to Bob encrypted with K_1 (shared by Alice and Eve), it can be deciphered and read by Eve. We can send the message to Bob encrypted by K_2 (shared key between Eve and Bob); or she can even change the message or send a totally new message. Bob is fooled into believing that the message has come from Alice. A similar scenario can happen to Alice in the other direction.

This situation is known as a man-in-the-middle attack because Eve comes in between the intercepts R_1, sent by Alice to Bob, and R_3, sent by Bob to Alice. It is also known as a bucket brigade attack since it resembles a short line of volunteers passing a bucket of water from person to person.

Q.22. Explain Diffie-Hellman key exchange algorithm. Calculate secret shared key if h = 17, g = 13, x = 3 and y = 7. Also explain man-in-middle attack. ***(R.G.P.V., Dec. 2009)***

Ans. **Diffie-Hellman Key Exchange Algorithm** – Refer to Q.20.

Problem – The steps are as follows –

(i) A chooses x = 3 and calculates = 13^3 mod 17 = 4

(ii) B chooses y = 7 and calculates = 13^7 mod 17 = 4

(iii) A sends the number 4 to B.

(iv) B sends the number 4 to A.

(v) A calculates the symmetric key k = 4^3 mod 17 = 13

(vi) B calculates the symmetric key k = 4^7mod 17 = 13

The value of k is the same for both A and B.Thus, the symmetric (shared) key is 13.

Man-in-middle Attack – Refer the ans. of Q.21.

Q.23. Why is Diffie-Hellman not resilient to a man-in-the middle attack ? ***(R.G.P.V., June 2007)***

Ans. A weakness of Diffie-Hellman is that although two individuals can agree on a shared secret key, there is no authentication, which means that Alice might be establishing a secret key with a bad guy.

Q.24. Discuss the variations of computational Diffie-Hellman problem.

Ans. Let p be a large prime number such that the discrete logarithm problem defined in Z^*_p is hard. Let G ∈ Z^*_p be a cyclic group of prime order q and g is assumed to be a generator of G. We assume that G is prime order, and security parameters p, q are defined as the fixed form p = 2q + 1 and ord(g) = q. A remarkable computational problem has been defined on this kind of set by Diffie and Hellman. More precisely, Diffie-Hellman assumption (CDH assumption) is referred to as the following statement –

Computational Diffie-Hellman problem (CDH) – On input g, g^x, g^y, computing g^{xy}.

An algorithm that solves the computational Diffie-Hellman problem is a probabilistic polynomial time Turing machine, on input g, g^x, g^y, outputs g^{xy} with non-negligible probability. Computational Diffie-Hellman assumption means that there is no such a probabilistic polynomial time Turing machine. This assumption is believed to be true for many cyclic groups, such as the prime sub-group of the multiplicative group of finite fields.

We are considering useful variations of Diffie-Hellman problem like square computational and decisional Diffie-Hellman problem, inverse computational and decisional Diffie-Hellman problem and divisible computational Diffie-Hellman problem.

Q.25. Write short note on the square computational Diffie-Hellman problem.

Ans. Square computational Diffie-Hellman problem (SCDH), on input g, g^x, computing g^{x^2}.

An algorithm that solves the square computational Diffie-Hellman problem is a probabilistic polynomial time Turing machine, on input g, g^x, outputs g^{x^2} with non-negligible probability. Square computational Diffie-Hellman assumption means that there is no such a probabilistic polynomial time Turing machine. Fortunately, we are able to argue that the SCDH assumption and CDH assumption are equivalent.

$$\text{SCDH} \Leftarrow \text{CDH}$$

Proof – Given an oracle A_1, on input g, g^x, g^y, outputs g^{xy}, we want to show that there exists an algorithm A_2, on input g^x, outputs g^{x^2}. Given a random value $u := g^r$, we choose $t_1, t_2 \in Z_q$ at random, and compute $u_1 = u^{t_1} = g^{rt_1}$, and $u^2 = u^{t_2} = g^{rt_2}$. Therefore we are able to compute $v = A_1(u_1, u_2) = g^{r^2 t_1 t_2}$ with non-negligible probability. It follows that g^{r^2} can be computed from v, t_1, t_2 immediately with same advantage.

$$\text{CDH} \Leftarrow \text{SCDH}$$

Proof – Given an oracle A_2, on input g, g^x, outputs g^{x^2}, we want to show that there exists an algorithm A_1, on input g, g^x, g^y, outputs g^{xy}. Now given g^x, we choose $s_1, s_2, t_1, t_2 \in Z_q$ at random and compute $v_1 := A_2(g^{xs_1}) = g^{(xs_1)^2}$, $v_2 := A_2((g^y)^{s_2}) = g^{(ys_2)^2}$. Finally, we compute $v_3 := A_2(g^{xs_1t_1 + ys_2t_2}) = g^{(xs_1t_1 + ys_2t_2)^2}$.

Since s_1, s_2, t_1, t_2 are known already, it follows that g^{xy} can be computed from $v_1, v_2, v_3, s_1, s_2, t_1, t_2$ immediately with same advantage.

Q.26. Discuss about the inverse computational Diffie-Hellman problem.

Ans. Inverse computational Diffie-Hellman problem (InvCDH), on input g, g^x, outputs $g^{x^{-1}}$.

An algorithm that solves the inverse computational Diffie-Hellman problem is a probabilistic polynomial time Turing machine, on input g, g^x, outputs $g^{x^{-1}}$ with non-negligible probability. Inverse computational Diffie-Hellman assumption means that there is no such a probabilistic polynomial time Turing machine. Fortunately, we are able to argue that the SCDH assumption and InvCDH assumption are also equivalent.

InvCDH $\Leftarrow$ SCDH

Proof – Given an oracle A_2, on input g, g^x, outputs g^{x^2}, we want to show that there exists an algorithm A_3, on input g^x, outputs $g^{x^{-1}}$. Given a random value g^r, we set $h_1 \leftarrow g^r$ and $h_2 \leftarrow g$. Finally, we view (h_1, h_2) as an input to the oracle A_2 to obtain $A_2(h_1, h_2) = g^{r^{-2}}$. It follows that $g^{r^{-1}}$ can be computed from A_2 immediately with same advantage.

SCDH $\Leftarrow$ InvCDH

Proof – Given an oracle A_3, on input g, g^x, outputs $g^{x^{-1}}$, we want to show that there exists an algorithm A_2, on input g, g^x, outputs g^{x^2}. Now given g, g^r, we set $h_1 \leftarrow g^r$ and $h^2 \leftarrow g$. Finally, we view (h_1, h_2) as an input to the oracle A_3 to obtain $A_3(h_1, h_2) = A_3(g^r, (g^r)^{r^{-1}})$. It follows that g^{r^2} can be computed from A_3 with the same advantage.

Q.27. Discuss on the divisible computational Diffie-Hellman problem.

Ans. Divisible computation Diffie-Hellman problem (DCDH problem), on random input g, g^x, g^y, computing $g^{y/x}$. We refer this oracle to as divisional computation Diffie-Hellman problem.

An algorithm that solves the divisible computational Diffie-Hellman problem is a probabilistic polynomial time Turing machine, on input g, g^x, g^y, outputs $g^{x/y}$ with non-negligible probability. Divisible computation Diffie-Hellman assumption means that there is no such a probabilistic polynomial time Turing machine. As desired, we are able to show that divisible computational Diffie-Hellman assumption is equivalent to computational Diffie-Hellman assumption –

CDH $\Leftarrow$ DCDH

Proof – Suppose we are given an divisible computation Diffie-Hellman oracle denoted by A_4, on input g, g^x, g^y, outputs $g^{y/x}$. We want to show that there exists an algorithm A_1, on input g, g^x, g^y, outputs g^{xy}. Given g, g^x, g^y, we choose $s_1, s_2, t_1, t_2 \in Z_q$ at random, and compute $v_1 := A_4(g, (g^x)^{s_1}, g^{s_2}) = g^{xs_1/s_2}$, $v_2 := A_4(g, g^{t_1}, (g^y)^{t_2} = g^{t_1/(yt_2)}$. Finally, we compute $v := A_3(v_1, v_2) = g^{(xys_1t_2)/(s_2t_1)}$. Since s_1, s_2, t_1, t_2 are known already, it follows that g^{xy} can be computed from v, s_1, s_2, t_1, t_2 immediately with same advantage.

DCDH $\Leftarrow$ CDH

Proof – Suppose we are given an computational Diffie-Hellman oracle A_1, on input g, g^x, g^y, it outputs g^{xy}. We want to show that there exists an algorithm A_4, on input g, g^x, g^y, outputs $g^{y/x}$. Suppose we are given a triple g, g^x, g^y now. By assumption, we are given a computational Diffie-Hellman oracle A_1, consequently, we are able to construct an InvCDH oracle A_3. Viewing g, g^y as input to A_3 to obtain $v := g^{y^{-1}}$. Finally, one views g, g^x, v as input to A_1 to obtain $g^{x/y}$.

We prove the fact that if the underlying group with prime order q, all variations of computational Diffie-Hellman problem are equivalent, i.e., CDH $\Leftrightarrow$ SCDH $\Leftrightarrow$ InvCDH $\Leftrightarrow$ DCDH.

Q.28. Explain the variation of decisional Diffie-Hellman problem.

Ans. Decisional Diffie-Hellman assumption (DDH) – Let G be a large cyclic group of prime order q defined in Q.24. We consider the following two distributions –

(i) Given a Diffie-Hellman quadruple g, g^x, g^y and g^{xy}, where x, y $\in Z_q$, are random strings chosen uniformly at random.

(ii) Given a random quadruple g, g^x, g^y and g^r, where x, y, r $\in Z_q$, are random strings chosen uniformly at random.

An algorithm that solves the Decisional Diffie-Hellman problem is a statistical test that can efficiently distinguish these two distributions. Decisional Diffie-Hellman assumption means that there is no such a polynomial statistical test. This assumption is believed to be true for many cyclic groups, such as the prime sub-group of the multiplicative group of finite fields.

Square decisional Diffie-Hellman assumption (SDDH) – Let G be a large cyclic group of prime order q defined in Q.24. We consider the following two distributions –

(i) Given a square Diffie-Hellman triple g, g^x and g^{x^2}, where x $\in Z_q$, is a random string chosen uniformly at random.

(ii) Given a random triple g, g^x and g^r, where x, r $\in Z_q$, are two random strings chosen uniformly at random.

An algorithm that solves the square decisional Diffie-Hellman problem (SDDH for short) is a statistical test that can efficiently distinguish these two distributions. Square decisional Diffie-Hellman assumption means that there is no such a polynomial statistical test.

Inverse decisional Diffie-Hellman assumption (InvDDH) – Let G be a large cyclic group of prime order q defined in Q.24. We consider the following two distributions –

(i) Given a inverse Diffie-Hellman triple g, g^x and $g^{x^{-1}}$, where x $\in Z_q$, is a random string chosen uniformly at random.

(ii) Given a random triple g, g^x and g^r, where x, r $\in Z_q$, are random strings chosen uniformly at random.

An algorithm that solves the inverse decisional Diffie-Hellman problem (InvDDH for short) is a statistical test that can efficiently distinguish these two distributions. Inverse decisional Diffie-Hellman assumption means that there is no such a polynomial statistical test.

Divisible decision Diffie-Hellman assumption (DDDH) – Let G be a large cyclic group of prime order q defined in Q.24. We consider the following two distributions –

(i) Given a divisible Diffie-Hellman quadruple g, g^x, g^y and $g^{x/y}$, where x, y ∈ Z_q, are random strings chosen uniformly at random.

(ii) Given a random quadruple g, g^x and g^y and g^r, where x, y, r ∈ Z_q, are random strings chosen uniformly at random.

An algorithm that solves the divisible decision Diffie-Hellman problem (DDDH for short) is a statistical test that can efficiently distinguish these two distributions. Divisive decision Diffie-Hellman assumption means that there is no such a polynomial statistical test.

NUMERICAL PROBLEMS

Prob.1. For a Diffie-Hellman scheme with a common prime q = 11 and a primitive root α = 2 –

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

(ii) If user A has public key Y_A = 9, what is A's private key X_A?

(iii) If user B has public key Y_B = 3, what is shared secret key K, shared with A? ***(R.G.P.V., June 2010)***

Sol. (i) We know that, a primitive root of a prime number p is one whose powers generate all the integers from 1 to P – 1. That is if 2 (a) is primitive root of prime number 11 (p), then the numbers 2 mod 11, 2^2 mod 11, ..., 2^{11-1} mod 11 are distinct and consist of the integers from 1 through 10.

(ii) Here q = 11 and α = 2

$Y_A = 9, \quad X_A = ?$

According to Diffie-Hellman scheme,

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

$$9 = 2^{X_A} \bmod 11$$

Solving the above equation, we get

$$9 = 2^6 \bmod 11$$

$$\therefore \quad X_A = 6$$

Thus A's private key X_A = 6.

(iii) Here q = 11 and α = 2,

$Y_B = 3, \quad k = ?, \quad X_A = 6$ (from above)

According to Diffie-Hellman scheme.

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

$$= (3)^6 \bmod 11$$

$$= (729) \bmod 11 = 3$$

Thus, the shared secret key k = 3. **Ans.**

Prob.2. Consider a Diffie-Hellman scheme with a common prime q = 11 and a primitive root α = 2.

(i) If user A has public key Y_A = 9, what is A's private key X_A?

(ii) If user B has public key Y_B = 3, what is the shared secret key? ***(R.G.P.V., June 2005)***

Sol. Refer to Prob.1 (ii) and (iii).

Prob.3. Briefly explain Diffie-Hellman key exchange. The Diffie-Hellman key exchange is being used to establish a secret key between 'A' and 'B'. 'A' sends B (719, 3, 191), B responds with (543). A's secret number, x, is 16. What is the secret key? ***(R.G.P.V., Dec. 2006)***

Sol. **Diffie-Hellman Key Exchange Algorithm** – Refer to Q.20.

In the given problem,

α = 3, q = 191

$X_A = 16 \qquad X_B$ = Not known

$Y_A = 719 \qquad Y_B = 543$

A can calculate the secret key as below,

$$k = (Y_B)^{X_A} \bmod q = (543)^{16} \bmod 191$$

Now, according to modular arithmetic we can proceed as follows –

$k = [(543)^8 \bmod 191 \times (543)^8 \bmod 191] \bmod 191$

Now $(543)^8 \bmod 191 = [(543)^4 \bmod 191 \times (543)^4 \bmod 191] \bmod 191$

Now $(543)^4 \bmod 191 = [(543)^2 \bmod 191 \times (543)^2 \bmod 191] \bmod 191$

Now $(543)^2 \bmod 191 = (294849) \bmod 191 = 136$

Thus $(543)^4 \bmod 191 = (136 \times 136) \bmod 191$

$= (18496) \bmod 191 = 160$

Now $(543)^8 \bmod 191 = (160 \times 160) \bmod 191$

$= (25600) \bmod 191 = 6$

Now $(543)^{16} \bmod 191 = (6 \times 6) \bmod 191 = 36 \bmod 191 = 36$

Thus, the secret key k = 36 **Ans.**

Prob.4. Users A and B use the Diffie-Hellman key exchange technique a common prime $q = 71$ and a primitive root $\alpha = 7$

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

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

(R.G.P.V., June 2016)

Sol. Given that, $q = 71$, $\alpha = 7$, $X_A = 5$, $X_B = 12$.

(i) A computes $Y_A = 7^5 \bmod 71$

(A's public key) $Y_A = \mathbf{51}$ **Ans.**

(ii) B computes $Y_B = 7^{12} \bmod 71$

(B's public key) $Y_B = \mathbf{3}$ **Ans.**

RSA ASSUMPTIONS & CRYPTOSYSTEM, RSA SIGNATURES & SCHNORR IDENTIFICATION SCHEMES, PRIMALITY TESTING

Q.29. Explain the RSA algorithm with an example.

Or

Write short note on RSA encryption algorithm.

(R.G.P.V., June 2006, May 2018)

Or

Write down RSA algorithm. Explain with the help of an example.

(R.G.P.V., Dec. 2006)

Or

Write short note on RSA algorithm. ***(R.G.P.V., June 2008)***

Or

Write a short note on RSA. ***(R.G.P.V., June 2015)***

Ans. RSA, named after its three creators – Rivest, Shamir and Adleman, was the first effective public key algorithm and for years have withstood intense scrutiny by cryptanalysis all over the world.

RSA relies on the fact that it is easy to multiply two large prime numbers together, but extremely hard (i.e. time consuming) to factor them back from the result.

The RSA scheme is a block cipher in which the plaintext and ciphertext are integers between 0 and $n - 1$ for some n, whose typical size is 1024 bits.

The RSA algorithm is as follows –

(i) Key Generation –

(a) Select two very large prime numbers, normally of equal length. Let them be denoted by p and q respectively but $p \neq q$.

(b) Calculate, $n = p \times q$

(c) Calculate, the Euler totient function $\phi(n)$ where $\phi(n) = (p-1)(q-1)$

(d) Select integer e such that

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

$1 < e < \phi(n)$

(e) Calculate d, such that

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

or

$$ed = 1 \bmod \phi(n)$$

or

(f) Now, the public key, $KU = \{e, n\}$

and, the private key, $KR = \{d, n\}$

(ii) Encryption – Let the plaintext be M such that $M < n$ then,

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

(iii) Decryption – Given, ciphertext C then,

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

Example – Key generation –

(a) Select two prime numbers

$p = 17$ and $q = 11$

(Usually large values must be selected)

(b) Calculate $n = pq = 17 \times 11 = 187$

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

(d) Select e, such that e is relatively prime to $\phi(n) = 160$ and less than $\phi(n)$, we choose $e = 7$.

(e) Determine d such that

$de = 1 \bmod 160$ and $d < 160$.

The correct value is $d = 23$

(f) The resulting keys are –

Public key, $KU = \{e, n\} = \{7, 187\}$

and Private key, $KR = \{d, n\} = \{23, 187\}$

Encryption –

Let plaintext, $M = 88$ then

Ciphertext, $C = 88^7 \bmod 187 = 11$

Decryption –

To decrypt calculate, $M = 11^{23} \bmod 187 = 88$

Q.30. Discuss about the security of RSA.

Ans. Three possible approaches to attacking the RSA algorithm are as follows –

(i) ***Brute Force*** – This involves trying all possible private keys.

(ii) ***Mathematical Attacks*** – There are several approaches, all equivalent in effect to factoring the product of two primes.

(iii) ***Timing Attacks*** – These depend on the running time of the decryption algorithm.

The defence against the brute force attack is the same for RSA as for other cryptosystems i.e., use a large key space. Thus, the larger the number of bits in e and d, the better. But, the problem is larger the size of the key, the slower the system will run.

The Factoring Problem – We can identify the three approaches to attacking RSA mathematically –

(i) Factor n into its two prime factors. This enables calculation of $\phi(n)$, which in turn enables determination of $d = e^{-1} \bmod \phi(n)$.

(ii) Determine ϕ (n) directly, without first determining p and q. Again, this enables determination of $d = e^{-1} \bmod \phi(n)$.

(iii) Determine d directly, without first determining $\phi(n)$.

Thus, it is clear that the security of RSA depends on the ability of the hacker to factorize numbers.

In August 1999, a specific assessment of the security of 512-bit RSA showed that one may be factored for less than $ 1,000,000 in cost and eight months of effort. Thus, we need to be careful in choosing key size for RSA. For near future, a key size in the range of 1024 to 2048 bits seems reasonable. For instance an Intel Paragon, which can achieve 50,000 MIPS would take a million years to factor a 2048-bit key using current techniques.

Timing Attacks – Paul Kocher, a cryptographic consultant, demonstrated that a snooper can determine a private key by keeping track of how long a computer takes to decipher messages. Timing attacks are applicable not just to RSA, but to other public-key cryptography systems. This attack is alarming for two reasons – It comes from a completely unexpected direction and it is a ciphertext only attack.

A timing attack is somewhat analogous to a burglar guessing the combination of a safe by observing how long it takes for someone to turn the dial from number to number.

Although the timing attack is a serious threat, there are some counter measures that can be used –

(i) ***Constant Exponentiation Time*** – Ensure that all exponentiations take the same amount of time before returning a result. This is a simple fix but does degrade performance.

(ii) ***Random Delay*** – Better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack. But Kocher points out that if defenders don't add enough noise, attackers could still succeed by collecting additional measurements to compensate for the random delays.

(iii) ***Blinding*** – Multiply the ciphertext by a random number before performing exponentiation. This process prevents the attacker from knowing what ciphertext bits are being processed inside the computer and therefore prevents the bit-by-bit analysis essential to the timing attack.

RSA Data Security incorporates a blinding feature into some of its products but with 2 to 10% performance penalty.

Q.31. Explain factoring problem in RSA. ***(R.G.P.V., June 2016)***

Ans. Refer to Q.30.

Q.32. Explain why the security of RSA depends on the difficulty of factoring large numbers ? ***(R.G.P.V., Dec. 2008)***

Ans. Refer to Q.30.

Q.33. What are the performance factor of RSA algorithm ?

Ans. The performance factor of RSA algorithm are –

(i) ***Time Complexity*** – Both encryption and decryption involve repeated multiplications of b-bit numbers. Unoptimized multiplication of two b-bit numbers and reduction modulo n, both take $O(b^2)$ time. The encryption key is usually a small integer. So encryption involves a small, constant number of modulo n multiplications. Hence, the time complexity of encryption is $O(b^2)$.

Decryption, on the other hand, involves raising a b-bit number to the power of d. A naive implementation of decryption thus involves d multiplications. Since d is of the same order as n, the complexity of a decryption operation is $O(nb^2)$.

(ii) ***Speeding Up RSA*** – Decryption of ciphertext c is speed up by computing c, c^2, c^4, c^8 etc., up to a maximum of b terms. Each element in the series is the square of the preceding element. Then we multiply elements in this series whose positions correspond to 1's in the binary representation of the decryption key d. Of course, each multiplication is a modulo n multiplication so the intermediate products are never more than b bits wide. This approach, which first computes squares followed by products, is referred to as "square and multiply".

In general, decryption involves b-1 square operations and at most b-1 multiplications. Also, each square operation and multiplication is followed by a reduction modulo n. Hence the time for decryption is $O(b^3)$ – this is considerably slower than $O(b^2)$ time necessary for encryption.

The choice of key size represents a trade-off between security and performance. A large key size provides greater security, but the times for both encryption and decryption increase. The asymptotic complexities tell us that doubling the key size increases the time for encryption by, roughly, a factor of 4, while the time for decryption increases by a factor of 8.

(iii) Software Performance – The Java programming language has a number of APIs of relevance to cryptography. These include APIs for key generation and encryption/decryption, message digests and digital signatures. These are contained in the Java security package and its various subpackages. Java also permits the import of classes created by various third parties that implement cryptographic algorithms. An example of third party is Bouncy Castle. Bouncy Castle cryptographic APIs are available for use in both Java and C++ programs. An example of the use of the Java APIs for key generation, encryption and decryption is shown in fig. 2.8.

```
KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA", "BC");
kpg.initialize(1024);
KeyPair kp = kpg.generateKeyPair( );
Cipher C = Cipher.getInstance ("RSA/ECB/PKCS1Padding", "BC");

String plainText = "Hello World";
c.init(Cipher.ENCRYPT_MODE, kp.getPublic( ));
byte[ ] encryptedText = c.doFinal(plainText.getBytes( ));

c.init(Cipher.DECRYPT_MODE, kp.getPrivate( ));
byte[ ] decryptedText = c.doFinal(encryptedText);
String recoveredText = new String (decryptedText);
```

Fig. 2.8 Illustrating Java APIs for RSA Encryption/Decryption

Q.34. Write short note on RSA signatures.

Ans. A signature system is obtained from RSA by defining –

$$k_s = (n, b) \text{ and } k_v = (n, a),$$

and

$$s_{k_s}(w) = (w,(w^b, \text{mod } n)) \text{ and } v_{k_v}(w, u)$$

$$= \begin{cases} \text{CORRECT, if } w \equiv u^a \text{ mod } n \\ \text{FALSE otherwise} \end{cases}$$

Apparently faking this signature in one way or another is equivalent to breaking RSA. An outside party can however choose a signature u by taking w = (u^a, mod n) as the message. Such a message does not contain any information, though. Even this does not work if an one-way hash function h is used. In that case k_v = (n, a, h) and

$$s_{k_s}(w) = (w,(h(w)^b, \text{mod } n)) \text{ and } v_{k_v}(w, u)$$

$$= \begin{cases} \text{CORRECT, if } h(w) \equiv u^a \text{ mod } n \\ \text{FALSE otherwise} \end{cases}$$

RSA can also be used to get a so-called blind signature. If A wishes to sign a message w of B, without knowing its content, the procedure is the following–

(i) B chooses a random number l such that gcd(l, n) = 1, computes the number t = (l^aw, mod n) and sends it to A.

(ii) A computes the signature u' = (t^b, mod n) as if the message would be t, and sends it to B.

(iii) B computes the number u = (l^{-1}u', mod n).

Because A does not know the number l, he/she does not get any information about the message w. On the other hand, u is the correct signature of the message w, since

$$l^{-1}u' \equiv l^{-1}t^b \equiv l^{-1}l^{ab}w^b \equiv l^{-1}lw^b \equiv w^b \text{ mod } n.$$

Q.35. Define the RSA digital signature scheme and compare it to the RSA cryptosystem.

Ans. When the concept of RSA is used for signing and verifying a message it is called RSA digital signature scheme. The digital signature scheme changes the roles of the private and public keys. The private and public keys of the sender are used. The sender uses her own private key to sign the document, the receiver uses the sender's public key to verify it. If we compare the scheme with the conventional way of signing, we see that the private key plays the role of the sender's own signature, the sender's public key plays the role of the copy of the signature that is available to public. Obviously John cannot use

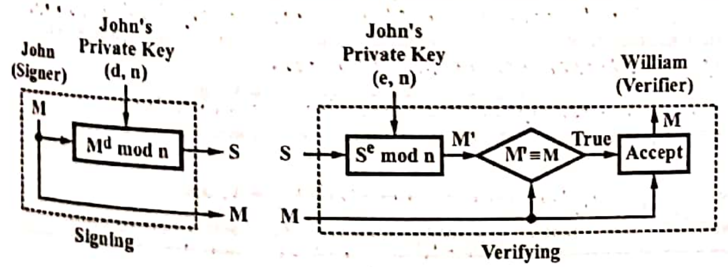


Fig. 2.9

William's public key to sign the message because then any other person could do the same. The signing and verifying sites use the same function but with different parameters. The verifier compares the message and the output of the function for congruence. The message is accepted, if the result is true. The RSA digital signature scheme is shown in fig. 2.9.

Key Generation – Key generation is exactly the same as in the RSA cryptosystem. John selects two primes p and q and calculates $n = p \times q$. John calculates $\phi(n) = (p - 1)(q - 1)$. Then, he selects e the public exponent and calculates d the private exponent such that $e \times d = 1 \bmod \phi(n)$. John keeps d; he publicly announces n and e.

Signing – John creates a signature out of the message using her private exponent, $S = M^d \bmod n$ and sends the message and the signature to William.

Verifying – William receives M and S. William applies John's public exponent to the signature to create a copy of the message $M' = S^e \bmod n$. William compares the value of M' with the value of M. William accepts the message if the two values are congruent.

Q.36. Explain the term Schnorr's identification scheme.

Ans. Let p and q be two primes such that $q \mid p - 1$ and $|q| = n$. Let $g \neq 1$ be an element of order q in Z_p^*. Let G_q be the subgroup generated by g. The integers p, q, g are known and can be common to a group of users.

An identity consists of a private/public key pair. The private key w is a random non-negative integer less than q. The public key is computed as $y = g^{-w} \bmod p$. The protocol is described in fig. 2.10.

Schnorr

Common Input – p, q, g, y. A security parameter t.
Secret Input for the Prover – $w \in Z_q$ such that $y = g^{-w} \bmod p$.

(i) Commitment by Prover. Prover picks $r \in_R Z_q$ and sends $x = g^r \bmod p$ to the Verifier.

$$\text{Prover} \xrightarrow{x = g^r} \text{Verifier}$$

(ii) Challenge from Verifier. Verifier picks a number $e \in_R [1 \ldots 2^t]$ and sends it to the Prover.

$$\text{Prover} \xrightarrow{e} \text{Verifier}$$

(iii) Response from Prover. Prover computes $s = r + w.e \bmod q$ and sends it to the Verifier.

$$\text{Prover} \xrightarrow{s = r + w.e} \text{Verifier}$$

The Verifier checks that $x = g^s.y^e \bmod p$ and accepts if and only if equality holds.

Fig. 2.10 Schnorr's Protocol

It is well known that Schnorr is an honest-verifier zero-knowledge proof of knowledge of w, the discrete logarithm of y. The protocol is only honest-verifier ZK, because if a dishonest verifier chooses the challenge e in a non-random way (particularly dependent on the first message x) we are not able to simulate the interaction.

It is shown that the Schnorr scheme is secure against impersonation, under concurrent attacks, under the assumption that discrete logarithm is secure under one more inversion in the underlying group.

Q.37. Explain the batching Schnorr identification scheme.

Ans. A naive generalization of Schnorr's scheme would be to do the simultaneous authentication of d identities by composing d rounds in parallel. In other words the prover would send over d commitments and the verifier would reply with d challenges – one per identity. Note that this scheme has a communication and computation cost that is d times the cost of Schnorr's original scheme. A possible improvement would be to use the same challenge for all rounds, and apply batch verification techniques to the last verification step. Even with these improvements, the communication and computation cost of the whole scheme would still be higher by a factor of d (the prover would still have to send and compute d commitments).

We propose a more efficient scheme where the prover sends one commitment and the verifier sends one challenge across all identities. The prover's response is generalized from a degree one polynomial to a degree d polynomial formed from the d secret keys. We are able to show that the resulting scheme is sound and further that it is secure against impersonation under concurrent attacks. We present two theorems that demonstrate that the new scheme is an honest-verifier zero knowledge proof of knowledge and also a secure identification against impersonation under concurrent attacks.

The parameters are very similar to Schnorr. Let p and q be two primes such that $q \mid p - 1$. Let $g \neq 1$ be an element of order q in Z_p^*. The integers p, q, g are public and can be common to a group of users.

We have d identities, each consisting of a private/public key pair indexed by i. The private keys w_i are non-negative integers less than q, chosen uniformly at random. The public keys are computed as $y_i = g^{-w_i} \bmod p$.

The Prover initiates the protocol by sending over the list of public keys y_i for which it claims to possess the corresponding private keys w_i. The protocol is described in fig. 2.11.

Batch-Schnorr

Common Input – $p, q, g, y_1, \text{----} y_d$. A security parameter t.

Secret Input for the Prover – $w_i \in Z_q$ such that $y_i = g^{-w_i} \bmod p$.

(i) **Commitment by Prover.** Prover picks $r \in_R Z_q$ and sends $x = g^r \bmod p$ to the Verifier.

$$\text{Prover} \xrightarrow{x = g^r} \text{Verifier}$$

(ii) **Challenge from verifier.** Verifier picks a number $e \in_R [1 \ldots 2^{(t + \log d)}]$ and sends it to the Prover.

$$\text{Prover} \xrightarrow{e} \text{Verifier}$$

(iii) **Response from Prover.** Prover computes $s = r + \Sigma_i w_i . e^i \bmod q$ and sends it to the Verifier.

$$\text{Prover} \xrightarrow{s = r + \Sigma_i w_i . e^i} \text{Verifier}$$

The Verifier checks that $x = g^y . \Pi_i y_i^{e^i} \bmod p$ and accepts if and only if equality holds.

Fig. 2.11 Batch Version of Schnorr's Protocol

Q.38. Write short note on primality testing.

Ans. A primality test is simply an algorithm that tests, either probabilistically or deterministically, whether or not a given input number is prime. A general primality test does not provide us with a prime factorization of a number not found to be prime, but simply labels it as composite. In cryptography, for example, we often need the generation of large primes and one technique for this is to pick a random number of requisite size and determine if it's prime. The larger the number, the greater will be the time required to test this and this is what prompts us to search for efficient primality tests that are polynomial in complexity. Note that the desired complexity is logarithmic in the number itself and hence polynomial in its bit-size as a number n requires O(log n) bits for its binary representation.

There are some primality tests which conclusively determine whether a number is prime or composite and are therefore deterministic, while others, such as the Fermat and the Miller-Rabin tests, despite correctly classifying all prime numbers, may allow some some composites to filter through, incorrectly labelling them as primes or probably primes, and this makes these tests probabilistic. There are usually four criteria which we look for in an efficient primality testing algorithm, it must be general, unconditional, deterministic and polynomial in complexity.

NUMERICAL PROBLEMS

Prob.5. Perform encryption and decryption using the RSA algorithm for the following data –

$p = 3$, $q = 11$ (p and q are prime numbers)

$l = 7$, $m = 5$ (l – encryption key, m – message)

(R.G.P.V., Dec. 2005)

Sol. Here $p = 3,\ q = 11,\ e = l = 7,\ m = 5,\ d = ?$

Here $n = p \times q = 3 \times 11 = 33$

$$\phi(n) = (p-1)(q-1) = 2 \times 10 = 20$$

First of all find d

$$\because \quad ed \equiv 1 \bmod \phi(n)$$
$$7 \times d \equiv 1 \bmod 20$$
$$\therefore \quad d = 3$$

To encrypt the message $m = 5$ we use

$$C = m^e \,(\text{mod } n) = 5^7 \bmod 33$$
$$= 78125 \bmod 33 = \mathbf{14}$$

After encryption $m = 5$ becomes $C = 14$.

Now performing the decryption we get

$$C = 14,$$

To get back the plain text M, we use

$$M = C^d \,(\text{mod } n) = 14^3 \bmod 33$$
$$= 2744 \bmod 33 = 5$$

which is same as the given M.

Hence, the process is correct.

Prob.6. In RSA encryption method, if the prime numbers p and q are 3 and 7 respectively, the encryption exponent e is 11, find the following –

(i) The least positive decryption exponent d

(ii) Public and private key

(iii) Ciphertext when the plaintext P is encrypted using the public key.

(R.G.P.V., Dec. 2007)

Sol. Given, two prime numbers $p = 3$, $q = 7$ and $e = 11$

Then,

$$n = pq = 3 \times 7 = 21$$
$$\phi(n) = (p-1)(q-1) = 2 \times 6 = 12$$

(i) $de \bmod \phi(n) = 1$

$d \times 11 \bmod 12 = 1$ and $d < 12$

The correct value of d is

$$11 \times 11 = 121 = 10 \times 12 + 1$$

Thus, $d = 11$

(ii) Public key KU = {e, n} = {11, 21}

Private key KR = {d, n} = {11, 21}

(iii) Ciphertext $C = p^e (\text{mod } n) = p^{11} (\text{mod } 21)$

Prob.7. Explain the RSA algorithm. Using the RSA public key cryptography with z = 1, y = 2, x = 3...., a = 26 and p = 5, q = 7, and d = 5 find e and encrypt 'fedcba'. ***(R.G.P.V., June 2007)***

***Sol.* RSA Algorithm** – Refer to Q.29.

Given, p = 5, q = 7, and d = 5

Then

(i) $n = pq = 5 \times 7 = 35$

(ii) $\phi(n) = (p - 1)(q - 1) = 4 \times 6 = 24$

(iii) Determine e such that $d \times e = 1 \text{ mod } \phi(n)$

i.e., $5 \times e = 1 \text{ mod } 24$ and $e < 24$

The correct value of e is 5.

The resulting keys are public key KU = {5, 35} and private key, KR = {5, 35}.

Fig. 2.12 shows the encryption of the plaintext 'fedcba'.

Plaintext (P)			Ciphertext (C)		After Decryption	
Symbolic	Numeric	P^5	P^5 (mod 35)	C^5	C^5 (mod 35)	Symbolic
f	6	7776	6	7776	6	f
e	5	3125	10	100000	5	e
d	4	1024	9	59049	4	d
c	3	243	33	39135393	3	c
b	2	32	32	33554432	2	b
a	1	1	1	1	1	a

Sender's Computation

Receiver's Computation

Fig. 2.12

Prob.8. What are the main features of RSA algorithms ? If p = 7 and q = 17, then calculate value of e and d and also encrypt SIR. ***(R.G.P.V., Dec. 2009)***

***Sol.* RSA Algorithm** – Refer to Q.29.

Given, p = 7 and q = 17

$$n = p \times q = 7 \times 17 = 119$$

$$\phi(n) = (p - 1)(q - 1) = 6 \times 16 = 96$$

Now we select e such that e is relatively prime to $\phi(n) = 96$ and less than $\phi(n)$. We choose e = 5

Now $de = 1 \text{ mod } 96$

$5d = 1 \text{ mod } 96$

Then $d = 77$ since $5 \times 77 = 385 = 4 \times 96 + 1$

The ciphertext, C, for a plaintext message, P, is given by $e = P^5 \text{ mod } 119$. The ciphertext is decrypted by the receiver according to the rule $P = C^{77} (\text{mod } 119)$.

Fig. 2.13 shows the encryption of the plaintext "SIR".

Plaintext			*Ciphertext (C)*		*After Decryption*	
Symbolic	Numeric	P^5	P^5 (mod 119)	C^{77}	C^{77} (mod 119)	Symbolic
S	19	2476099	66	66^{77}	19	S
I	9	59049	25	25^{77}	9	I
R	18	1889568	86	86^{77}	18	R

Sender's computation

Receiver's computation

Fig. 2.13 An Example of RSA Algorithm

Prob.9. What do you mean by RSA algorithms ? In a public-key system using RSA, you intercept the ciphertext C = 11 sent to a user whose public-key is e = 7, n = 33. What is the plaintext M ? ***(R.G.P.V., June 2009)***

***Sol.* RSA Algorithm** – Refer to Q.29.

Given, C = 11, e = 7, and n = 33.

As we know that $n = pq$

$33 = pq$ then p and q are 3 and 11

$\phi(n) = (p - 1)(q - 1) = 2 \times 10 = 20$

$de = 1 \text{ mod } \phi(n)$

$d7 = 1 \text{ mod } 20$

$d = 3$

Now the plaintext M for the ciphertext C = 11 is

$M = C^d \text{ mod } n = 11^3 \text{ mod } 33$

$M = 1331 \text{ mod } 33 = \mathbf{11}$ **Ans.**

Prob.10. Explain RSA algorithm and using this algorithm encrypt the following –

(i) p = 3, q = 11, e = 7, M = 5 (ii) p = 7, q = 11, e = 17, M = 8 ***(R.G.P.V., June 2012)***

***Sol.* RSA Algorithm** – Refer to Q.29.

(i) Refer to Prob.5.

(ii) Here p = 7, q = 11, e = 17, M = 8.

$n = p \times q = 7 \times 11 = 77$

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

$ed = 1 \text{ mod } \phi(n)$

$17 \times d = 1 \text{ mod } 60$

$d = 17^{-1} \text{ mod } 60$

$d = 53$

Now, encrypt the message M = 8

$C = M^e \bmod n$

$= 8^{17} \bmod 77$

$= ((8^4 \bmod 77) \times (8^4 \bmod 77) \times (8^4 \times \bmod 77)$

$(8^4 \bmod 77)(8^1 \bmod 77)) \bmod 77$

$= (15 \times 15 \times 15 \times 15 \times 8) \bmod 77$

$= 405000 \bmod 77 = \mathbf{57}$

Ans.

Prob.11. Perform encryption and decryption using the RSA algorithm for the following –

(i) p = 17, q = 31, e = 7, m = 2 (ii) p = 11, q = 13, e = 11, m = 7.

(R.G.P.V., Dec. 2008)

Sol. (i) Here p = 17, q = 31, e = 7, m = 2

Here $n = p \times q = 17 \times 31 = 527$

We know that $\phi(n) = (p-1)(q-1) = 16 \times 30 = 480$

Then, $ed = 1 \bmod \phi(n)$

$7 \times d = 1 \bmod 480$

Then 7d must be 481, 961, 1441, 1921, etc.

Dividing each of these in turn by 7 to see which is divisible by 7, we get

$$\frac{2401}{7} = 343$$

Thus, d = 343

Now,

Supertext $C = m^e \bmod n$

$= 2^7 \bmod 527$

$= 128 \bmod 527 = 128$

And, 343

Plaintext $m = 128 \bmod 527 = \mathbf{2}$ **Ans.**

(ii) Here p = 11, q = 13, e = 11, m = 7

$n = p \times q$

$= 11 \times 13 = 143$

We know that

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

$= 10 \times 12 = 120$

Then $ed = 1 \bmod \phi(n)$

$11d = 1 \bmod 120$

$d = 11$

To encrypt the message m = 7.

We use following formula

$C = m^e(\bmod n) = (7)^{11} \bmod 143 = 106$

After encryption m = 7 becomes C = 106

Now performing the decryption to get back the plaintext m, we use

$m = C^d(\bmod n) = (106)^{11} \bmod 143$

$m = [(106)^4 \bmod 143 \times (106)^4 \bmod 143 \times (106)^2 \bmod 143 \times 106] \bmod 143$

$m = [3 \times 3 \times 82 \times 106] \bmod 143$

$m = \mathbf{7}$ **Ans.**

which is same as the given m.

Prob.12. Perform the encryption and decryption using RSA algorithm –

(i) p = 3, q = 11, e = 7, m = 5 (ii) p = 11, q = 13, e = 17, m = 8

(R.G.P.V., May 2019)

Sol. (i) Refer to Prob.5.

(ii) Here p = 11, q = 13, e = 17, m = 8

$n = p \times q = 11 \times 13 = 143$

$\phi(n) = (p-1) \times (q-1) = 10 \times 12 = 120$

Then $ed = 1 \bmod \phi(n)$

$17 \times d = 1 \bmod 120$

Then 17d must be 121, 241, 361, 481, etc.

Dividing each of these in turn by 17 to see which is divisible by 17, we get

$$\frac{1921}{17} = 113$$

Thus, d = 113

Now, encrypt the message m = 8

$C = m^e \bmod n$

$= 8^{17} \bmod 143$

$= [(8^4 \bmod 143) \times (8^4 \bmod 143) \times (8^4 \bmod 143)$

$\times (8^4 \bmod 143) \times (8^1 \bmod 143)] \bmod 143$

$= (92 \times 92 \times 92 \times 92 \times 8) \bmod 143$

$= 573114368 \bmod 143 = \mathbf{112}$ **Ans.**

ELLIPTIC CURVE OVER THE REALS, ELLIPTIC CURVE MODULO A PRIME, CHINESE REMAINDER THEOREM

Q.39. What is elliptic curve ? What is sum of three points on an elliptic curve that lie on a straight line ? (R.G.P.V., Dec. 2003, June 2004)

Or

What is an elliptic curve and what is zero point on elliptic curve ?

(R.G.P.V., June 2013

Ans. An elliptic curve is a set of points on the coordinate plane satisfying an equation of the form

$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

where a, b, c, d and e are real numbers, and x and y take on values in the real numbers. For our purpose, it is sufficient to limit ourselves to equation of the form

$$y^2 = x^3 + ax + b \quad ...(i)$$

Such equation is said to be cubic, or of degree 3. Also included in the definition of an elliptic curve is a single element denoted by O and called the *point at infinity* or the *zero point*. To plot such a curve, we have to compute

$$y = \sqrt{x^3 + ax + b}$$

For given values of a and b, the plot consists of positive and negative values of y for each value of x. Thus, each curve is symmetric about y = 0.

Now, consider the set of points E (a, b) consisting of all of the points (x, y) that satisfy equation (i) together with the element O. Using a different value of the pair (a, b) results in a different set E (a, b).

A group can be defined based on the set E (a, b) provided that $x^3 + ax + b$ has no repeated factors. This is equivalent to the condition

$$4a^3 + 27b^2 \neq 0 \quad ...(ii)$$

Now we define an operation, called addition and denoted by +, for the set E(a, b) where a and b satisfy equation (ii). The rules for addition can be stated as follows – if three points on an elliptic curve lie on a straight line, their sum is O. By this definition, the rules of addition can be defined over an elliptic curve as follows –

(i) O serves as the additive identity, Thus O = – O. For any point P on the elliptic curve, P + O = P. In what follows, we assume P ≠ O and Q ≠ O.

(ii) The negative of a point P is the point with the same x coordinate but the negative of the y coordinate [i.e., if P = (x, y), then – P = (x, – y)]. It is noted that P + (– P) = P – P = O.

(iii) To add two points P and Q with different x coordinates, draw a straight line between them and find the third point of intersection R i.e., a unique point of intersection (unless the line is tangent to the curve at either P or Q, in which case we consider R = P or R = Q, respectively). To form a group structure, we need to define addition on these three points as follows - P + Q = – R. i.e., we define P + Q to be the mirror image (with respect to the x axis) of the third point of intersection.

(iv) The geometric interpretation of the preceding item also applies to two points, P and – P, with the same x coordinate. The points are joined by a vertical line, which can be viewed as also intersecting the curve at the infinity point. Therefore, we have P + (– P) = O, consistant with rule (ii).

(v) To double a point Q, draw the tangent line and find the other point of intersection S. Then Q + Q = 2Q = – S.

With the list of these rules it can be shown that the set E (a, b) is an abelian group.

Q.40. Explain elliptic curve cryptography. ***(R.G.P.V., Dec. 2005, June 2007)***

Or

Discuss in brief elliptical curve cryptography. ***(R.G.P.V., June 2006)***

Or

Describe elliptic curve cryptography. ***(R.G.P.V., Dec. 2004)***

Or

Write short note on elliptic curve cryptography. ***(R.G.P.V., Dec. 2007)***

Or

What is an elliptic curve cryptography ? ***(R.G.P.V., Dec. 2006, 2008)***

Or

Explain elliptic curve cryptography with suitable example. ***(R.G.P.V., June 2016)***

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

Consider the equation Q = KP, where Q, P ∈ E_p (a, b) and K < P. It is relatively easy to calculate Q given K and P, but it is relatively hard to determine K given Q and P. This is called the discrete logarithm problem for elliptic curves.

ECC Key Exchange – Key exchange using elliptic curve can be done in the following manner. First pick a large integer q, which is either a prime number p or an integer of the form 2^m and elliptic curve parameters a and b for equation

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

or

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

This defines the elliptic group of points E_q (a, b). Next, pick a base point $G = (x_1, y_1)$ in E_p (a, b) whose order is a very large value n. The order n of a point G on an elliptic curve is the smallest positive integer n such that nG = 0. E_q (a, b) and G are parameters of the cryptosystem known to all participants.

A key exchange between user A and user B can be accomplished as follows (fig. 2.14).

(i) A selects an integer n_A less than n. This is A's private key. A then generates a public key $P_A = n_A \times G$; the public key is a point in E_q (a, b).

(ii) B similarly selects a private key n_B and computes a public key P_B.

(iii) A generates the secret key $K = n_A \times P_B$.

B generates the secret key $K = n_B \times P_A$.

Global Public Elements

Eq (a, b)	Elliptic Curve with Parameters a, b and q, where q is a prime or an Integer of the form 2^m
G	Point on Elliptic Curve whose Order is Large Value n

User A Key Generation

Select Private n_A	$n_A < n$
Calculate Public P_A	$P_A = n_A \times G$

User B Key Generation

Select Private n_B	$n_B < n$
Calculate Public P_B	$P_B = n_B \times G$

Generation of Secret Key by User A

$K = n_A \times P_B$

Generation of Secret Key by User B

$K = n_B \times P_A$

Fig. 2.14 ECC Key Exchange

The two calculations in step (iii) produce the same result because

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

To break this scheme, an attacker would need to be able to compute K given G and KG, which is assumed hard.

Example –

Take p = 211.

E_p (0, – 4), which is equivalent to the curve $y^2 = x^3 - 4$, and G = (2, 2).

One can calculate 240 G = 0

A's private key, $n_A = 121$

So A's public key, $P_A = 121\,(2, 2)$

$= (115, 48)$

B's private key, $n_B = 203$

So B's public key, $P_B = 203\,(2, 2) = (130, 203)$

The shared secret key is,

121 (130, 203) = 203 (115, 48) = (161, 69).

Q.41. Explain elliptic curve cryptography and its applications.

(R.G.P.V., June 2017)

***Ans.* Elliptic Curve Cryptography –** Refer to Q.40.

Applications – There are several applications of elliptic curve cryptography –

(i) Discrete Logarithm on EC – Modular exponentiation – computing g^k mod p (provided the prime p, a generator, g of Z_p^* and k) was relatively simple. It needs O(log k) multiplications/squarings using the "Square and Multiply" technique. On the other hand, computing k provided g, p, and g^k mod p is infeasible for large p (100's of digits). The discrete logarithm problem is likewise infeasible for a group of points on carefully selected elliptic curves. Analogous to computing modular exponentiation in Z_p^*, computing kG, provided an integer k, and the EC parameters is relatively simple. The parameters include the coefficients in the EC equation and the value of a point G, that is a generator of points on the EC. The operation,

$$kG = G + G + \ldots\ G \quad k \text{ times}$$

is known as scalar multiplication. Analogous to the "Square and Multiply" technique employed for modular exponentiation, scalar multiplication may be speeded up by using a "Double and Add" technique. This involves computing G, 2G, 4G, ... and then adding the suitable terms from the series.

(ii) Diffie-Hellman Key Exchange on EC Groups – For an EC specified over F(p), a six-tuple is used to identify (a) the EC and (b) the subgroup of points on the EC over which the discrete logarithm difficulty is infeasible.

$$\langle p, a, b, G, n, h\rangle$$

(a) p represents a prime number. p is the order of the field F(p). a and b are the coefficients of the EC equation.

(b) G represents a generator of a large subgroup of the points on the EC. The order of G is a prime number, n. The last parameter, h, in the six-tuple is the "cofactor" equal to $\dfrac{\#EC(F_p)}{n}$.$\#EC(F_p)$ is the number of points on the EC.

Using the group of points on an EC, the Diffie-Hellman key exchange protocol is described.

Consider that A and B require to agree on a fresh session key. Also consider that both A and B have already agreed to employ the same EC with parameters, ⟨p, a, b, G, n, h⟩.

Then, A and B proceed to complete the following steps as given below –

(a) A selects a random integer x, computes xG and transmits this to B.

(b) B selects a random integer y, computes yG and transmi this to A.

(c) B computes y(xG) = xyG on receipt of message of A.

(d) A computes x(yG) = xyG on receipt of message of B.

Now, both A and B share a common secret xyG. xyG is a point on the provided EC. Note that an eavesdropper who sees the "partial secrets" xG and yG will not be capable to deduce x, y, or xyG due to the infeasibility of the discrete logarithm difficulty on well selected elliptic curves and more specifically, the intractability of the computational Diffie-Hellman difficulty on ECs.

(iii) Encryption on EC Groups – El Gamal encryption over Z_p^* to encryption over a group of points on the elliptic curve, is defined as follows. a large subgroup of prime order is selected, n of the points on the EC. Let ⟨G⟩ show this subgroup.

(a) Let the integer, a be private key of A.

(b) Then the public key of A is $\alpha = aG$.

To encrypt a message to A, B performs the tasks as given below –

(a) B selects a random number, r, $1 \le r \le n - 1$ and calculate rG.

(b) B calculates $M + r\alpha$. Note that the message has been shown like a point, M, in ⟨G⟩.

(c) The encrypted text is the pair $\langle rG, M + r\alpha \rangle$ which is transmitted to A.

To decrypt the message received from B, A performs the tasks as given below–

(a) A extracts rG, the first part of the encrypted message and uses her private key, a, to compute $a(rG) = r(aG) = r\alpha$.

(b) Then A extracts $M + r\alpha$ from the encrypted message and subtracts out $r\alpha$ to achieve $M + r\alpha - r\alpha = M$.

Modified El Gamal Encryption –

Step (i) of El Gamal encryption remains the same. Although, step (ii) involves the following computation given below –

$$[r\alpha]_x \times m \bmod p$$

where $[r\alpha]_x$ represents the x-coordinate of $r\alpha$.

The ciphertext is the pair $\langle rG, [r\alpha]_x \times m \bmod p \rangle$.

To decrypt the ciphertext, A uses her private key, a as before to calculate $a(rG) = r(aG) = r\alpha$. The x-coordinate of the point $r\alpha$ is extracted by her and then performs the following computation with the second part of the ciphertext, are given below –

$$([r\alpha]_x)^{-1} \bmod p) \times ([r\alpha]_x \times m \bmod p) \bmod p$$
$$= m \bmod p$$

(iv) EC-based Digital Signatures – This signature algorithm is known as EC-DSA.

Assume that ⟨p, a, b, G, n, h⟩ show an EC. G represents a generator of a large prime subgroup of the EC.

Consider the integer, a be private key of A and $\alpha = aG$ be her public key.

The following task is performed by A to sign a message, m

(a) A selects a random number, r, $1 < r < n - 1$ and calculates rG.

(b) She calculates the hash of the message, h(m).

(c) Then, she calculates the pair (S_1, S_2) where

$$S_1 = [rG]_x \bmod n \text{ and}$$
$$S_2 = r^{-1}(h(m) + a \times S_1) \bmod n$$

We next derive the expression to verify A's signature, (S_1, S_2), on a message, m. By definition,

$$S_2 = r^{-1}(h(m) + a \times S_1) \bmod n$$

Hence, $r = S_2^{-1}h(m) + S_2^{-1}\ S_1\ a + kn$ (where k is an integer)

Using both sides of the above equation like scalar multipliers of point G, we obtain

$$rG = S_2^{-1}h(m)\,G + S_2^{-1}S_1(aG) + k(nG)$$

Now, $aG = \alpha$ and $nG = O$ since the order of G is n.

Hence, $rG = S_2^{-1}h(m)\,G + (S_2^{-1}S_1)\alpha$

Taking the x-coordinate of the points on the R.H.S. and L.H.S., equation is achieved which is used to verify the digital signature with the support of public key of signer's, α.

$$S_1 \overset{?}{=} [S_2^{-1}h(m)G + (S_2^{-1}S_1)\alpha]_x$$

Q.42. Give the main differences between RSA algorithm and elliptic curve cryptography (ECC). ***(R.G.P.V., June 2014)***

Ans. Refer to Q.29 and Q.40.

Q.43. Explain elliptic curve encryption/decryption with the help of an example. Also, discuss the security of ECC.

Or

Briefly explain elliptic curve encryption/decryption using suitable examples. ***(R.G.P.V., Dec. 2009)***

Ans. The first task in this approach is to encode the plaintext message m to be sent as an x – y, point P_m. It is the point P_m that will be encrypted as a ciphertext and subsequently decrypted. It is noted that we cannot simply encode the message as the x or y coordinate of a point, because not all such coordinates are in E_q (a, b).

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

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

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

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

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

A has masked the message P_m by adding kP_B to it. Only A knows the value of k, so even though P_B is a public key, no one can remove the mask kP_B. However, A also includes a "clue," which is enough to remove the mask if one knows the private key n_B. For an attacker to recover the message, the attacker would have to compute k given G and kG, which is assumed hard.

For example, take P = 751 and $E_p = (-1, 188)$ which is equivalent to the curve $y^2 = x^3 - x + 188$, and G (0, 376). Suppose that A wishes to send a message to B that is encoded in the elliptic point P_m (562, 201) and that A selects the random number k = 386. B's public key is $P_B = (201, 5)$. We have 386 (0, 376) = (676, 558), and (562, 201) + 386 (201, 5) = (385, 328). Thus A sends the ciphertext {(676, 558), (385, 328)}.

Security of Elliptic Curve Cryptography – The security of ECC depends on how difficult it is to determine k given kP and P. This is referred to as the elliptic curve logarithm problem. The fastest known technique for taking the elliptic curve logarithm is known as the pollard rho method. It is clear that, considerably a smaller key size can be used for ECC compared to RSA. Furthermore, for equal key lengths, the computational efforts required for ECC and RSA is comparable. Thus, there is a computational advantage to using ECC with a shorter key length than a comparably secure RSA.

Q.44. What are the hardware and software requirements ? Classify them into various cryptographic services. ***(R.G.P.V., June 2013)***

Ans. **Hardware and Software Requirements** – There are two clear portions of the application – the client-side and the server-side.

(i) Client-side Requirements – The client-side requirements include a browser-based workstation that has Internet Explorer browser installed. A specific browser is required due to use of the services of Microsoft's MS-CAPI cryptographic toolkit on the client-side. Because MS-CAPI is installed automatically as a part of Internet Explorer, we will need that the user must have it installed. Thus, the user can use another browser for actual surfing transactions. That is, the user's workstation should still have Internet Explorer installed.

(ii) Server-side Requirements – They include the presence of a cryptographic toolkit. Even though JCA/JCE, MS-CAPI etc. can suffice, a specialist toolkit like the one from RSA, entrust or baltimore is preferred to use. No other special requirements exist on the server-side.

Classification of Requirements – These requirements are classified into appropriate categories as depicted in fig. 2.15.

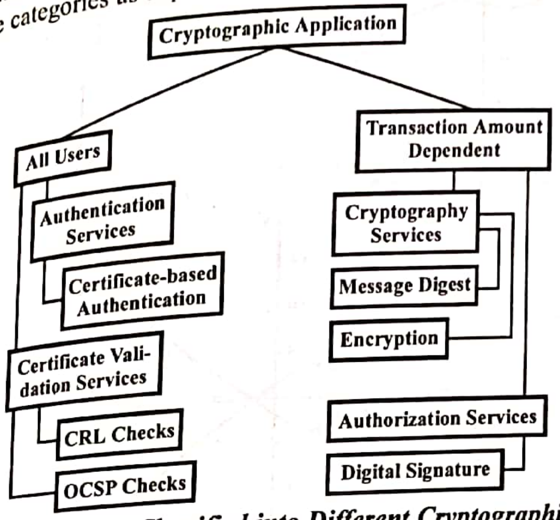


Fig. 2.15 Requirements Classified into Different Cryptographic Services

Q.45. Discuss the elliptic curve over the reals.

Ans. As in the case of any group, we need to define –

(i) The elements of the group

(ii) The group operation

(iii) The group identity

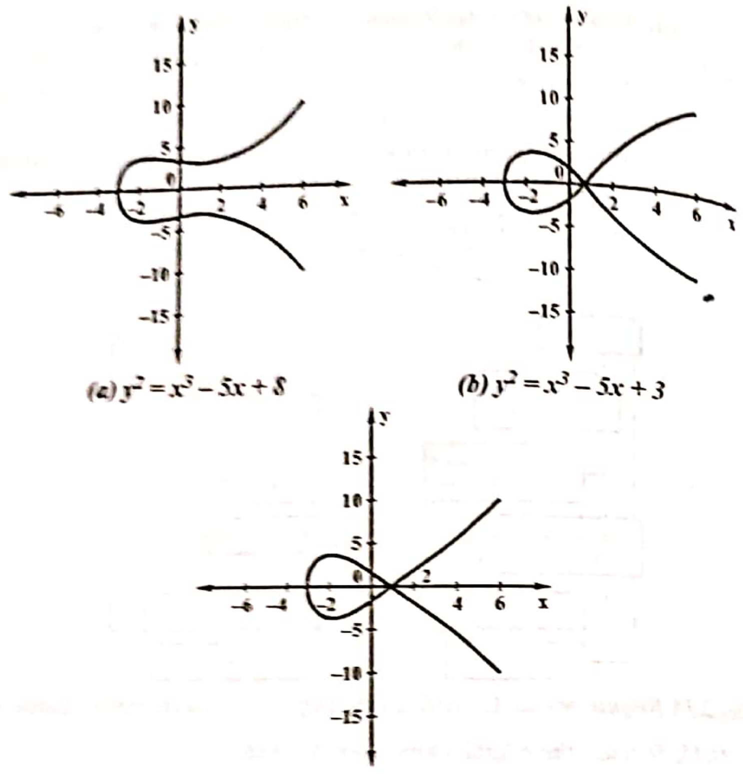
(iv) The inverse of each group element.

The elements of our groups of interest are points whose (x, y) coordinates are real numbers satisfying.

$$y^2 = x^3 + ax + b \qquad ...(i)$$

Here a, b are real numbers to ensure that the curve does not intersect itself, the rests of the RHS polynomial in the above equation should be distinct. This constraint translates into the inequality, $4a^3 + 27b^2 \neq 0$. We use the notation EC(a, b) to refer to an elliptic curve where a and b are the coefficients in equation (i).

Example – Fig. 2.16 (a) shows the elliptic curve, EC(–5, 8), which is in a single piece fig. 2.16 (b) shows EC(–5, 3), which comprises two disjoint pieces. Fig. 2.16 (c) shows EC(–3, 2), which is self-intersecting and violates the condition $4a^3 + 27b^2 \neq 0$.



(a) $y^2 = x^3 - 5x + 8$ *(b) $y^2 = x^3 - 5x + 3$*

(c) $y^2 = x^3 - 3x + 2$

Fig. 2.16 Elliptic Curve Over Reals

Q.46. Describe the elliptic curve modulo a prime.

Ans. The equation of elliplic curve EC over F(p), where p is prime, is identical to the one for ECs over reals. Note that the coordinates of the points and the coefficients in the equation are elements of F(p). The algebraic expressions for point negation, point addition and point doubling are each identical to those derived for ECs over reals except that all operations are performed modulo p. However, unlike in the case for reals, there is no obvious geometrical interpretation for point addition or doubling.

Example – Let the EC, $y^2 = x^3 + 2x + 4$ over F_{13}.

The 17 points on the elliptic curve (EC) including the point at infinity as follows –

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

The relationship between p and the number of points on an EC defined over F_p, it turns out that the number of points is O(p). In fact, Hasse's theorem established tight bounds on, #EC(Fp) the number of points on the EC.

$$p + 1 - 2\sqrt{p} \le \#\text{EC}(Fp) \le p + 1 + 2\sqrt{p}$$

Q.47. Define the Chinese remainder theorem.

Ans. If factors of the modulus m are known, i.e. we can write

$$m = m_1 m_2 \ldots\ldots m_k,$$

the congruences $x \equiv y \bmod m_i (i = 1, 2, \ldots\ldots, k)$ naturally follow from $x \equiv y$ mod m. If the modulus is a large number, it may often be easier to compute using these smaller moduli. This can be done very generally, if the factors m_1, m_2, ..., m_k are pairwise coprime, in other words, if $\gcd(m_i, m_j) = 1$ when $i \ne j$ –

Theorem (Chinese remainder theorem) – If the number $y_1, y_2, \ldots\ldots, y_k$ are given and the moduli $m_1, m_2, \ldots\ldots, m_k$ are pairwise coprime then there is a unique integer x modulo $m_1 m_2 \ldots\ldots m_k$ that satisfies the k congruences.

$$x \equiv y_i \bmod m_i \; (i = 1, 2, \ldots\ldots, k).$$

Proof – Denote $M = m_1 m_2 \ldots\ldots m_k$ and $M_i = M/m_i$ $(i = 1, 2, \ldots\ldots, k)$. Since the m_i's are pairwise coprime, $\gcd(M_1, M_2, \ldots\ldots, M_k) = 1$ and $\gcd(m_i, M_i) = 1$ (i = 1, 2, ..., k). The following procedure produces a solution x(if there is one!), and also shows that the solution is unique modulo M.

NUMERICAL PROBLEMS

Prob.13. On the elliptic curve over the real numbers $y^2 = x^3 - 36x$, let P = (– 3.5, 9.5) and Q = (– 2.5, 8.5). Find P + Q and 2P.

(R.G.P.V., June 2008)

Sol. The sum R = P + Q can be expressed as follows –

$$x_R = \Delta^2 - x_P - x_Q$$

and

$$y_R = - y_P + \Delta(x_P - x_R)$$

where $\Delta = (y_Q - y_P)/(x_Q - x_P)$

Thus,

$$P = (-3.5, 9.5)$$

and

$$Q = (-2.5, 8.5)$$

$$x_R = \left(\frac{8.5 - 9.5}{-2.5 + 3.5}\right)^2 - (-3.5) - (-2.5)$$

$$= \left(\frac{-1.0}{1.0}\right)^2 + 3.5 + 2.5$$

$$= 1 + 3.5 + 2.5 = 7$$

and $$y_R = -(9.5) + \left(\frac{8.5 - 9.5}{-2.5 + 3.5}\right)(-3.5 - 7)$$

$$= -(9.5) - 1\,(-10.5)$$

$$= -9.5 + 10.5$$

$$= 1$$

We also need to be able to add a point to itself, i.e., $P + P = 2P = R$. When $y_P \neq 0$ the expression are

$$x_R = \left(\frac{3x_P^2 + a}{2y_P}\right)^2 - 2x_P$$

$$y_R = \left(\frac{3x_P^2 + a}{2y_P}\right)(x_P - x_R) - y_P$$

Thus, $$x_R = \left(\frac{3(-3.5)^2 + (-36)}{2 \times 9.5}\right) - 2(-3.5)$$

$$= \left(\frac{3 \times (12.25) - 36}{2 \times 9.5}\right) + 7.0$$

$$= \left(\frac{36.75 - 36}{2 \times 9.5}\right) + 7.0 = \frac{0.75}{2 \times 9.5} + 7.0 = 7.04$$

and $$y_R = \left(\frac{3(-3.5)^2 + (-36)}{2 \times 9.5}\right)(-3.5 - 7.04) - 9.5$$

$$= (0.039)\,(-10.54) - 9.5$$

$$= -0.41 - 9.5 = -\mathbf{9.91}$$

Ans.

●●

UNIT 3

MESSAGE AUTHENTICATION, DIGITAL SIGNATURE, KEY MANAGEMENT, KEY EXCHANGE, HASH FUNCTION, UNIVERSAL HASHING, CRYPTOGRAPHIC HASH FUNCTION, MD, SECURE HASH ALGORITHM (SHA), DIGITAL SIGNATURE STANDARD (DSS)

Q.1. What is message authentication ? Give the name of two levels of functionality that comprise a message authentication or digital signature mechanism.

Ans. Message authentication is a procedure to verify that received messages come from the alleged source and have not been altered. Message authentication may also verify sequencing and timeliness.

Any message authentication or digital signature mechanism can be viewed as having fundamentally two levels. At the lower level, there must be some sort of function that produces an authenticator – a value to be used to authenticate a message. This lower-level function is then used as primitive in a higher-level authentication protocol that enables a receiver to verify the authenticity of a message.

There are many types of functions that may be used to produce an authenticator. These may be grouped into three catagories as follows –

(i) ***Message Encryption*** – The ciphertext of the entire message serves as its authenticator.

(ii) ***Message Authentication Code (MAC)*** – A public function of the message and a secret key that produces a fixed-length value that serves as the authenticator.

(iii) ***Hash Function*** – A public function that maps a message of any length into a fixed-length hash value, which serves as the authenticator.

Q.2. Why are message authentication codes derived from a cryptographic hash function being preferred over authentication code derived from symmetric cipher ?

Ans. A message authentication code (MAC) based on the use of a symmetric block cipher is known as Data Authentication Algorithm. This has traditionally been the most common approach to constructing a MAC. In recent years, there has been increased interest in developing a MAC derived from a cryptographic hash function. The motivations for this interest are as follows –

(i) Cryptographic hash functions such as MD5 or SHA-1 generally execute faster in software than symmetric block ciphers such as DES.

(ii) Library code for cryptographic hash functions is widely available.

(iii) There are no export restrictions from the U.S. or other countries for cryptographic hash functions, whereas symmetric block ciphers, even when used for MACs are restricted.

Q.3. Explain the message authentication codes.

Ans. An alternative technique of authentication involves the use of a secret key to generate a small fixed-size block of data, known as a cryptographic checksum or message authentication code (MAC) that is appended to the message. Consider two communicating parties, say A and B, share a common secret key K. When A has a message to send to B, it calculates the MAC as a function of the message and the key –

$$\text{MAC} = C_K(M)$$

where M = Input message

C = MAC function

K = Shared secret key

MAC = Message authentication code.

The message plus MAC are transmitted to the intended recipient. The recipient performs the same calculation on the received message, using the same secret key, to generate a new MAC. The received MAC is compared to the calculated MAC as shown in fig. 3.1 (a). If we suppose that only the receiver and the sender know the identity of the secret key, and if the received MAC matches the calculated MAC, then

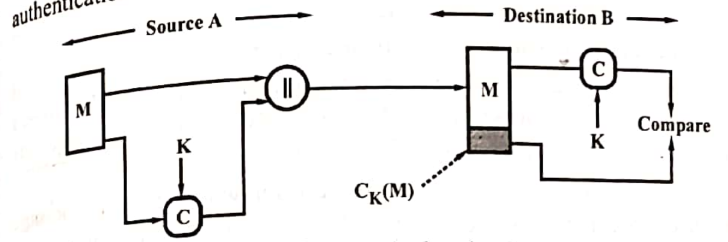
(i) The receiver is assured that the message has not been altered. If an attacker alters the message but does not alter the MAC, then the receiver's calculation of the MAC will differ from the received MAC. Because the attacker is assumed not to know the secret key, the attacker cannot alter the MAC to correspond to the alterations in the message.

(ii) The receiver is assured that the message is from the alleged sender. Because no one else knows the secret key, no one else could prepare a message with a proper MAC.

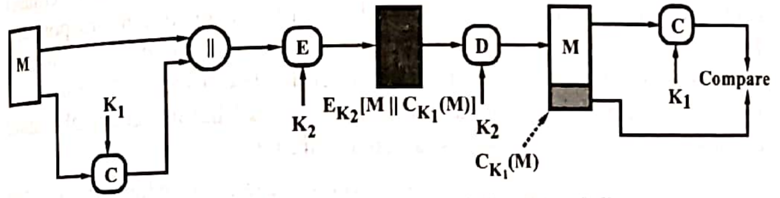
(iii) If the message includes a sequence number (such as is used with HDLC, X.25, and TCP), then the receiver can be assured of the proper sequence because an attacker cannot successfully alter the sequence number.

A MAC function is similar to encryption. One difference is that the MAC algorithm need not be reversible, as it must for decryption. In general, the MAC function is a many-to-one function. The domain of the function consists of messages of some arbitrary length, whereas the range consists of all possible MACs and all possible keys. If an n-bit MAC is used, then there are 2^n possible MACs, whereas there are N possible messages with $N >> 2^n$. Furthermore, with a k-bit key, there are 2^k possible keys.

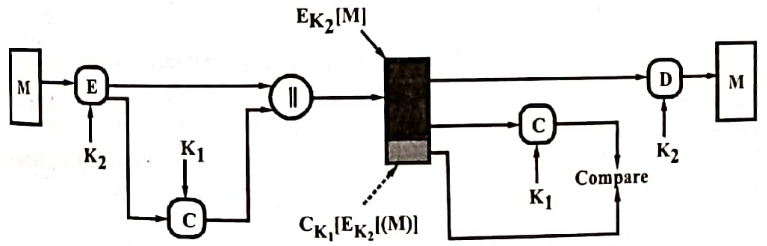
It is pointed that because of the mathematical properties of the authentication function, it is less vulnerable to being broken than encryption.



(a) Message Authentication



(b) Message Authentication and Confidentiality; Authentication Tied to Plaintext



(c) Message Authentication and Confidentiality; Authentication Tied to Ciphertext

Fig. 3.1 Basic Uses of Message Authentication Code (MAC)

The process illustrated in fig. 3.1(a) provides authentication but not confidentiality, because the message as a whole is transmitted in the clear.

Confidentiality can be provided by performing message encryption either after fig. 3.1(b) or before fig. 3.1(c) the MAC algorithm. In both these cases, two separate keys are needed, each of which is shared by the sender and the receiver. In the first case, the MAC is calculated with the message as input and is then concatenated to the message. The entire block is then encrypted. In the second case, the message is encrypted first. Then the MAC is calculated using the resulting ciphertext and is concatenated to the ciphertext to form the transmitted block. Typically, it is preferable to tie the authentication directly to the plaintext, so the method of fig. 3.1(b) is used.

Finally, it is noted that the MAC does not provide a digital signature because both sender and receiver share the same key.

Table 3.1 summarizes the confidentiality and authentication implications of the approaches illustrated in fig. 3.1.

Because symmetric encryption will provide authentication and because it is widely used with readily available products, why not simply use this instead of a separate message authentication code? [DAVI 89] suggests three situations in which a message authentication code is used –

(i) There are a number of applications in which the same message is broadcast to a number of destinations. Examples are notification to users that the network is now unavailable or an alarm signal in a military control center. It is cheaper and more reliable to have only one destination responsible for monitoring authenticity. Thus, the message must be broadcast in plaintext with an associated message authentication code. The responsible system has the secret key and performs authentication. If a violation occurs, the other destination systems are alerted by a general alarm.

(ii) Another possible scenario is an exchange in which one side has a heavy load and cannot afford the time to decrypt all incoming messages. Authentication is carried out on a selective basis, message being chosen at random for checking.

(iii) Authentication of a computer program in plaintext is an attractive service. The computer program can be executed without having to decrypt it every time, which would be wasteful of processor resources. However, if a message authentication code were attached to the program, it could be checked whenever assurance was required of the integrity of the program.

There other rationales may be added, as follows –

(iv) For some applications, it may not be of concern to keep messages secret, but it is important to authenticate messages. An example is the Simple Network Management Protocol version 3 (SNMP v3), which separates the functions of confidentiality and authentication. For this application, it is usually important for a managed system to authenticate incoming SNMP messages, particularly if the message contains a command to charge parameters at the managed system. On the other hand, it may not be necessary to conceal the SNMP traffic.

(v) Separation of authentication and confidentiality functions affords architectural flexibility. For example, it may be desired to perform authentication at the application level but to provide confidentiality at a lower level, such as the transport layer.

(vi) A user may wish to prolong the period of protection beyond the time of reception and yet allow processing of message contents. With message encryption, the protection is lost when the message is decrypted, so the message is protected against fraudulent modifications only in transit but not within the target system.

Table 3.1 Basic Uses of Message Authentication Code C

$A \rightarrow B : M \| C_K(M)$ • Provides Authentication – Only A and B share K **(a) Message Authentication**
$A \rightarrow B : E_{K_2}[M \| C_{K_1}(M)]$ • Provides authentication – Only A and B share K_1 • Provides confidentiality – Only A and B share K_2 **(b) Message Authentication and Confidentiality : Authentication Tied to Plaintext**
$A \rightarrow B : E_{K_2}[M] \| C_{K_1}(E_{K_2}[M])$ • Provides authentication – Using K_1 • Provides confidentiality – Using K_2 **(c) Message Authentication and Confidentiality : Authentication Tied to Ciphertext**

Finally, note that the MAC does not provide a digital signature because both sender and receiver share the same key.

***Q.4. What types of attacks are addressed by message authentication ?* (R.G.P.V., June 2017)**

Ans. In communication across a network the following attacks can be identified –

(i) ***Disclosure*** – Release of message contents to any person or process not possessing the appropriate cryptographic key.

(ii) Traffic Analysis – Discovery of the pattern of traffic between parties. In a connection-oriented application, the frequency and duration of connections could be determined. In either a connection-oriented or connectionless environment, the number and length of messages between parties could be determined.

(iii) Masquerade – Insertion of messages into the network from a fraudulent source. This includes the creation of messages by an opponent that are purported to come from an authorized entity. Also included acknowledgements of message recipient.

(iv) Content Modification – Changes to the contents of a message, including insertion, deletion, transposition and modification.

(v) Sequence Modification – Any modification to a sequence of messages between parties including insertion, deletion and reordering.

(vi) Timing Modification – Delay or replay of messages in a connection-oriented application, an entire session or sequence of messages could be a replay of some previous valid session, or individual messages in the sequence could be delayed or replayed. In a connectionless application, an individual message, such as datagram could be delayed or replayed.

(vii) Source Repudiation – Denial of transmission of message by source.

(viii) Destination Repudiation – Denial of receipt of message by destination.

Thus, some counter-measures are required to deal with these attacks which are generally regarded as message authentication.

Q.5. Explain digital signature with the help of an example. Give its properties and requirement in brief. ***(R.G.P.V., June 2007)***

Or

Explain digital signature. ***(R.G.P.V., Dec. 2004, 2005)***

Or

Write short note on digital signatures. ***(R.G.P.V., June 2006)***

Ans. Message authentication protects two parties who exchange messages from any third party. However, it does not protect the two parties against each other. Several forms of dispute between the two are possible.

For example, suppose that Vishal sends an authenticated message to Gourav, using one of the schemes of message authentication code. Then the following disputes can arise –

(i) Gourav may forge a different message and claim that it came from Vishal. Gourav would simply have to create a message and append an authentication code using the key that Vishal and Gourav share.

(ii) Vishal can deny sending the message. Because it is possible for Gourav to forge a message, there is no way to prove that Vishal did in fact send the message.

We take an example of the first scenario. An electronic funds transfer takes place, and the receiver increases the amount of funds transferred and claims that the larger amount had arrived from the sender. An example of the second scenario is that an electronic mail message contains instructions to a stockbroker for a transaction that subsequently turns out badly. The sender pretends that the message was never sent.

When 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. Then it must have the following properties –

(i) It must verify the author and the date and time of the signature.

(ii) It must authenticate the contents at the time of the signature.

(iii) It must be verifiable by third parties, to resolve disputes.

Thus the digital signature function includes the authentication function.

On the basis of these properties, we can formulate the following requirements for a digital signature –

(i) The signature must be a bit pattern that depends on the message being signed.

(ii) The signature must use some information unique to the sender, to prevent both forgery and denial.

(iii) It must be relatively easy to produce the digital signature.

(iv) It must be relatively easy to recognize and verify the digital signature.

(v) It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message.

(vi) It must be practical to retain a copy of the digital signature in storage.

Q.6. What is the important aspect that establishes trust in digital signatures ? ***(R.G.P.V., June 2014)***

Ans. Refer to Q.5.

Q.7. What property does a digital signature provide that an HMAC does not ? Discuss. ***(R.G.P.V., June 2007, Dec. 2007)***

Ans. HMAC does not provide nonrepudiation. Digital signature provides for nonrepudiation. If the sender denies sending the message, her private key corresponding to her public key can be tested on the original plaintext. If the result of decryption matches the original message then we know the sender sent the message.

Q.8. Explain arbitrated and direct approach for digital signature function. ***(R.G.P.V., Dec. 2004)***

Or

What are some threats associated with a direct digital signature scheme ? Describe in detail. ***(R.G.P.V., June 2005)***

Or

Explain direct digital signature. What are some threats associated with a direct digital signature scheme ? ***(R.G.P.V., Dec. 2009)***

Or

Explain digital signature with arbitrated and direct approaches. ***(R.G.P.V., May 2018)***

Ans. **Direct Digital Signature** – The direct digital signature involves only the communicating parties (source, destination). Assume that the destination knows the public key of the source. A digital signature may be formed by encrypting the entire message with the sender's private key or by encrypting a hash code of the message with the sender's private key.

Confidentiality can be provided by further encrypting the entire message plus signature with either the receiver's public key or a shared secret key (symmetric encryption). It is noted that it is important to perform the signature function first and then an outer confidentiality function. In case of any dispute, some third party must view the message and its signature. If the signature is calculated on an encrypted message, then the third party also needs access to the decryption key to read the original message. However, if the signature is the inner operation, then the recipient can store, the plaintext message and its signature for later use in dispute resolution.

All direct schemes share a common weakness. The validity of the scheme depends on the security of the sender's private key. If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and that someone else forget his or her signature. Administrative controls relating to the security of private keys can be employed to thwart or at least weaken this ploy, but the thwart is still there, at least to some degree. One example is to require every signed message to include a timestamp (date and time) and to require prompt reporting of compromised keys to a central authority.

Another threat is that some private key might actually be stolen from X at time T. The opponent can then send a message signed with X's signature and stamped with a time before or equal to T.

Arbitrated Digital Signature – The problem associated with direct digital signature can be addressed by using an arbiter. As with direct signature schemes, there are various arbitrated signature schemes. Generally, they all operate as follows. Every signed message from a sender X to a receiver Y goes first to an arbiter A, who subjects the message and its signature to a number of tests to check its origin and content. The message is then dated and sent to Y with an indication that it has been verified to the satisfaction of the arbiter. The presence of A solves the problem faced by direct signature scheme – that X might disown the message.

The arbiter plays a sensitive and crucial role in this sort of scheme, and all parties must have a great deal of trust that the arbitration mechanism is working properly. The use of a trusted system might satisfy this requirement. Table 3.2 gives several examples of arbitrated digital signatures.

Table 3.2 Arbitrated Digital Signature Techniques

(a) Conventional Encryption, Arbiter Sees Message
(1) $X \rightarrow A: M \| E_{K_{xa}}[ID_X \| H(M)]$
(2) $A \rightarrow Y: E_{K_{ay}}[ID_X \| M \| E_{K_{xa}}[ID_X \| H(M)] \| T]$
(b) Conventional Encryption, Arbiter Does Not See Message
(1) $X \rightarrow A: ID_X \| E_{K_{xy}}[M] \| E_{K_{xa}}[ID_X \| H(E_{K_{xy}}[M])]$
(2) $A \rightarrow Y: E_{K_{ay}}[ID_X \| E_{K_{xy}}[M] \| E_{K_{xa}}[ID_X \| H(E_{K_{xy}}[M]) \| T]$
(c) Public-key Encryption, Arbiter Does Not See Message
(1) $X \rightarrow A: ID_X \| E_{KR_x}[ID_X \| E_{KU_y}(E_{KR_x}[M])]$
(2) $A \rightarrow Y: E_{KR_a}[ID_X \| E_{KU_y}[E_{KR_x}[M]] \| T]$

Notation – X = Sender M = Message
Y = Recipient T = Timestamp
A = Arbiter

In the first scenario, both parties must have a high degree of trust in A –

(i) X must trust A not to reveal K_{xa} and not to generate false signatures of the form $E_{K_{xa}}[ID_X \| H(M)]$.

(ii) Y must trust A to send $E_{K_{ay}}[ID_x \| M \| E_{K_{xa}}[ID_x \| H(M)] \| T]$ only if the hash value is correct and the signature was generated by X.

(iii) Both sides must trust A to resolve disputes fairly.

If the arbiter live up to this trust, then X is assured that no one can forge his signature and Y is assured that X cannot disavow his signature. This scenario also implies that A is able to read messages from X to Y and, indeed that any eavesdropper.

Table 3.2 (b) shows a scenario that provides the arbitration as before but also assures confidentiality. Although unable to read the message, the arbiter is still in a position to prevent fraud on the part of either X or Y. A remaining problem, one shared with the first scenario, is that the arbiter could form an alliance with the sender to deny a signed message or with the receiver to forge the sender's signature.

All the problems just discussed can be resolved by going to a public key scheme, which is shown in table 3.2 (c). This scheme has a number of advantages over the preceding two schemes. First, no information is shared among the parties before communication, preventing alliances to defraud. Second, no incorrectly dated message can be sent, even if KR_x is compromised, assuming that KR_a is not compromised. Finally, the content of the message from X to Y is secret from A and anyone else. However, this final scheme involves encryption of the message twice with a public-key algorithm.

Q.9. Differentiate between direct digital signature and arbitrated digital signature. ***(R.G.P.V., June 2008)***

Or

Compare direct digital signature vs arbitrated digital signature. ***(R.G.P.V., Dec. 2008)***

Ans. A direct digital signature involves only the communication parties i.e., source and destination. A digital signature is formed by encrypting the entire message with the sender's private key. Confidentiality is provided by further encrypting the entire message plus signature with either the receiver's public key or a shared secret key.

On the other hand in arbitrated digital signature, each signed message from a sender X to a receiver Y goes first to an arbiter A, who subjects the message and its signature to a number of tests to check its origin and content. The message is then dated and sent to Y with an indication that it has been varified to the satisfaction of the arbiter.

Q.10. "Digital envelopes combine the best features of symmetric and asymmetric key cryptography." Explain it. Why ? ***(R.G.P.V., June 2014)***

Ans. Symmetric key cryptography and asymmetric key cryptography are combined to have a very efficient security solution. The way it works is as follows –

(i) Suppose A is the sender of a message then A's computer encrypts the original plain text message with the help of a standard symmetric key cryptography algorithm and generates cipher text message (CT). In this operation, the key used (K1) is called one time symmetric key because it is used only once.

(ii) A now takes the one time symmetric key of step (i) i.e. K1 and encrypts K1 with B's public key K2. This process is called as key wrapping of the symmetric key. B is the receiver of a message.

(iii) Now, A puts the cipher text (CT1) and the encrypted symmetric key together inside a digital envelope.

(iv) The sends the digital envelope to B.

(v) B receives the digital envelope that contains cipher text and the one time session key (K1) encrypted using B's private key (K2).

(vi) Now B uses the same asymmetric key algorithm as used by A and her private key (K3) to decrypt the logical box that has the symmetric key (K1), which was encrypted with B's public key (K2).

(vii) At last, B applies the same symmetric key algorithm as was used by A and the symmetric key K1 to decrypt the cipher text. Thus, generates the original plain text.

Q.11. Discuss the digital signature process.

Ans. Fig. 3.2 shows the digital signature process. The sender uses a ***signing algorithm*** to sign the message. The message and the signature are sent to the receiver. The receiver receives the message and the signature and applies the ***verifying algorithm*** to the combination. If the result is true, the message is accepted; otherwise, it is rejected.

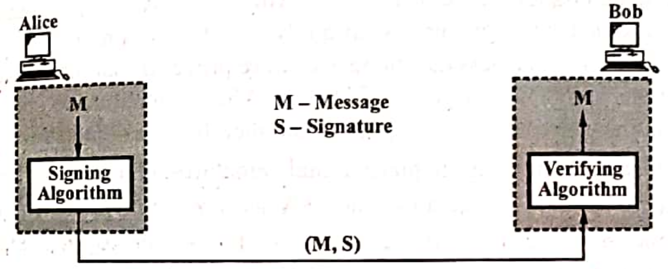


Fig. 3.2 Digital Signature Process

In a digital signature, the signer uses her private key, applied to a signing algorithm, to sign the document. The verifier, on the other hand, uses the public key of the signer, applied to the verifying algorithm, to verify the document.

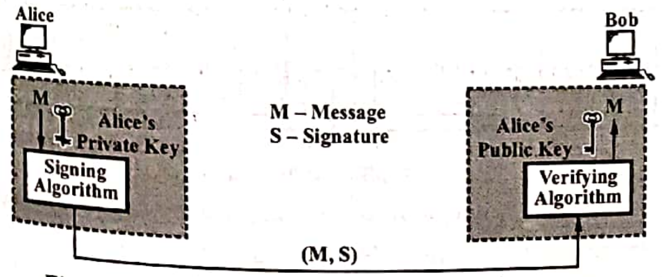


Fig. 3.3 Adding Key to the Digital Signature Process

We can add the private and public keys to fig. 3.2 to give a more complete concept of digital signature (see fig. 3.3). Note that when a document is signed, anyone, including Bob, can verify it because everyone has access to Alice's public key. Alice must not use her public key to sign the document because then anyone could forge her signature.

Q.12. Describe how digital signature can be used for ensuring message integrity in distributed system ? ***(R.G.P.V., Dec. 2007)***

Ans. Message integrity often goes beyond the actual transfer through a secure channel. Consider the situation in which Bob has just sold Alice a collector's item of some phonograph record for $500. The whole deal was done through e-mail. In the end, Alice sends Bob a message confirming that she will buy the record for $500. In addition to authentication, there are at least two issues that need to be taken care of regarding the integrity of the message.

(i) Alice needs to be assured that Bob will not maliciously change the $500 mentioned in her message into something higher, and claimed she promised more than $500.

(ii) Bob needs to be assured that Alice cannot deny ever having sent the message, for example, because she had second thoughts.

These two issues can be dealt with if Alice digitally signs the message in such a way that her signature is uniquely tied to its content. The unique association between a message and its signature prevents that modifications to the message will go unnoticed. In addition, if Alice's signature can be verified to be genuine, she cannot later repudiate the fact that she signed the message.

There are several ways to place digital signatures. One popular form is to use a public-key cryptosystem such as RSA, as shown in fig. 3.4. When Alice sends a message m to Bob, she encrypts it with her private key K_A^-, and sends it off to Bob. If she also wants to keep the message content a secret, she can use Bob's public key and send $K_B^+(m, K_A^-(m))$, which combines m and the version signed by Alice.

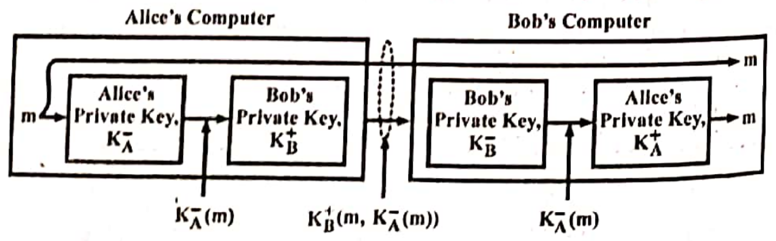

Alice's Computer
Bob's Computer
m
Alice's Private Key, K_A^-
Bob's Private Key, K_B^+
Bob's Private Key, K_B^-
Alice's Private Key, K_A^+
m
m
$K_A^-(m)$
$K_B^+(m, K_A^-(m))$
$K_A^-(m)$

Fig. 3.4 Digital Signing a Message Using Public-key Cryptography

When the message arrives at Bob, he can decrypt it using Alice's public key. If he can be assured that the public key is indeed owned by Alice, then decrypting the signed version of m and successfully comparing it to m can

mean only that it came from Alice. Alice is protected against any malicious modifications to m by Bob, because Bob will always have to prove that the modified version of m was also signed by Alice. In other words, the decrypted message alone essentially never counts as proof. It is also in Bob's own interest to keep the signed version of m to protect himself against repudiation by Alice.

There are a number of problems with this scheme, although the protocol in itself is correct. First the validity of Alice's signature holds only as long as Alice's private key remains a secret. If Alice wants to bail out of the deal even after sending Bob her confirmation, she could claim that her private key was stolen before the message was sent.

Another problem occurs when Alice decides to change her private key, as changing keys from time to time helps against intrusion. However, once Alice has changed her key, her statement sent to Bob becomes worthless. What may be needed in such cases is a central authority that keeps track of when keys are changed, in addition to using timestamps when signing messages.

Another problem with this scheme is that Alice encrypts the entire message with her private key. Such an encryption may be costly in terms of processing requirements and is actually unnecessary. A cheaper and elegent scheme is to use a ***message digest.***

A message digest is a fixed-length bit string h that has been computed from an arbitrary-length message m by means of a cryptographic hash function H. If m is changed to m', its hash H(m') will be different from h = H(m) so that it can easily be detected that a modification has taken place.

To digitally sign a message, Alice can first compute a message digest and subsequently encrypt the digest with her private key, as shown in fig. 3.5. The encrypted digest is sent along with the message to Bob.

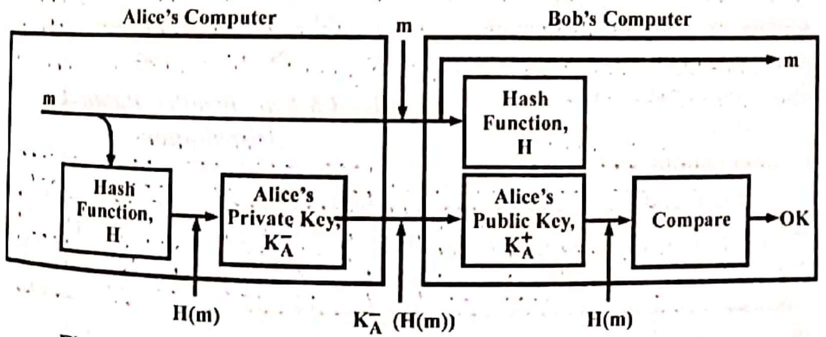


Fig. 3.5 Digitally Signing a Message Using a Message Digest

When Bob receives the message and its encrypted digest, he needs merely decrypt the digest with Alice's public key, and separately calculate the message digest. If the digest calculated from the received message and the decrypted digest match, Bob knows the message have been signed by Alice.

Q.13. What do you mean by key management ?

Ans. Key management is the hardest part of cryptography. It is the management of cryptographic keys in a cryptosystem. Key management is related to the generation, storage, distribution, and backup of keys. It includes cryptographic protocol design, key servers, user procedures and other relevant protocols. Key management concerns keys at the user level, either between users or systems. Successful key management is critical to the security of a cryptosystem.

Q.14. List four general categories of schemes for the distribution of public-keys. Describe each of them briefly.

Or

Define various methods for key management.

Ans. Various techniques have been proposed for the distribution of public-keys. All these proposals can be grouped into four catagories –

(i) Public announcement (ii) Publicly available directory
(iii) Public-key authority (iv) Public-key certificates.

(i) Public Announcement of Public-keys – The point of the public-key encryption is that the public-key is public. Thus, if there is some public-key algorithm, such as RSA, any participant can send his or her public-key to any other participant or broadcast the key to the community at large as shown in fig. 3.6. For example, many PGP (pretty good privacy) users, have adopted the practice of appending their public-key to messages that they send to public forums, such as USENET newsgroups and Internet mailing lists.

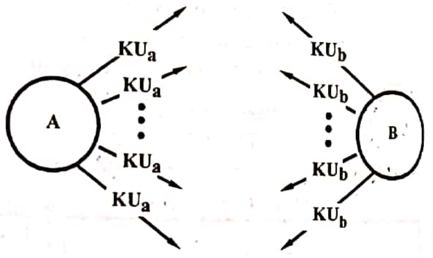


Fig. 3.6 Uncontrolled Public-key Distribution

Although this approach is convenient, it has a major weakness. Anyone can forge such a public announcement i.e, some user could pretend to be user A and send a public key to another participant or broadcast such a public-key. Until such time as user A discovers the forgery and alerts other participants, the forger is able to read all encrypted messages intended for A and can use the forged keys for authentication.

(ii) Publicly Available Directory – A higher 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 as shown in fig. 3.7. Such scheme has the following elements –

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

(b) 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.

(c) A participant may replace the existing key with a new key at any time.

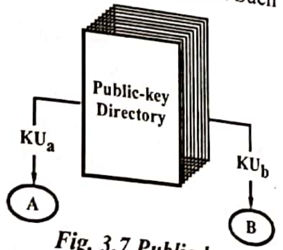


Fig. 3.7 Public-key Publication

(d) Periodically, the authority publishes the entire directory or updates to the directory.

(e) Participant could also access the directory electronically. For this purpose secure authenticated communication from the authority to the participant is mandatory.

This scheme is more secure then individual public announcements but still has vulnerabilities. If an opponent succeeds in obtaining or computing the private key of the directory authority, the opponent could authoritatively pass out public-keys and subsequently impersonate any participant. Another way to achieve the same end is for the opponent to tamper with the records kept by the authority.

(iii) Public-key Authority – Stronger security for public-key distribution can be achieved by providing tighter control over the distribution of public-keys from the directory. A typical scenario is illustrated in fig. 3.8. The scenario assumes that a central authority maintains a dynamic directory of public-keys of all participants. In addition, each participant reliably knows a public-key for the authority, with only the authority knowing the corresponding private key. The following steps occur in the scenario –

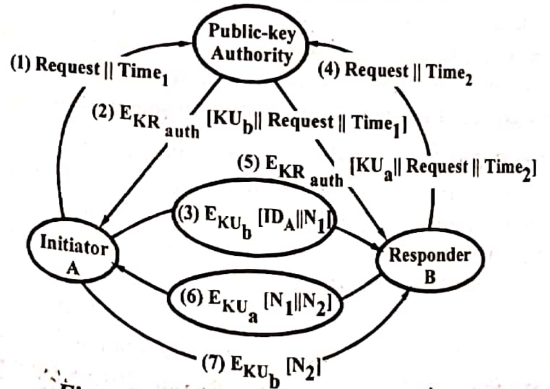


Fig. 3.8 Public-key Distribution Scenario

(a) **Message 1 –** A sends a timestamped message to the public-key authority containing a request for the current public key of B.

(b) **Message 2 –** The authority responds with a message that is encrypted using the authority's private key, KR_{auth}. Thus, A is able to decrypt the message using the authority's public key. Therefore, A is assured that the message originated with the authority. The message includes the following -

(1) B's public key KU_b, which A can use to encrypt messages destined for B.

(2) The original request, to enable A to match this response with the corresponding earlier request and to verify that the original request was not altered before reception by the authority.

(3) The original time stamp, so A can determine that this is not an old message from the authority containing a key other than B's current public-key.

(c) **Message 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 name (N_1) which is used to identify this transaction uniquely.

(d) **Message 4, 5 –** B retrieves A's public-key from the authority in the same manner as A retrieved B's public-key.

At this point public-keys have been securely delivered to A and B, and they may begin their protected exchange. However, two additional steps are desirable –

(e) **Message 6 –** B sends a message to A encrypted with KU_a and containing A's nonce (N_1) as well as a new nonce generated by B (N_2). Because only B could have decrypted message (3), the presence of N_1 in message (6) assures A that the correspondent is B.

(f) **Message 7 –** A returns N_2, encrypted using B's public-key, to assure B that its correspondent is A.

Thus, a total of 7 messages are required. However, the initial four messages need be used only infrequently because both A and B can save the other's public key for future use, a technique known as caching. Periodically, a user should request fresh copies of the public-keys of its correspondents to ensure currency.

(iv) Public-key Certificates – The scenario described above has some drawbacks. The public-key authority could be somewhat of a bottleneck in the system, for a user must appeal to the authority for a public-key for every other user that it wishes to contact. As before, the directory of names and public-keys maintained by the authority is vulnerable to tampering.

An alternative approach is to use *certificates* that can be used by participants to exchange keys without contacting a public-key authority, in a way, i.e., as reliable as if the keys were obtained directly from a public-key

authority. Each certificate contains a public-key and other information, is created by a certificate authority, and is given to the participant with the matching private key. A participant conveys its key information to another by transmitting its certificate. Other participants can verify that the certificate was created by the authority. This scheme has following requirements –

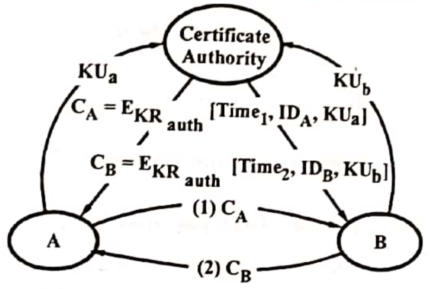
(a) Any participant can read a certificate to determine the name and public-key of the certificate's owner.

(b) Any participant can verify that the certificate originated from the certificate authority and is not counterfeit.

(c) Only the certificate authority can create and update certificates.

(d) In addition, any participant can verify the currency of the certificate.

A certificate scheme is illustrated in fig. 3.9. Each participant applies to the certificate authority, supplying a public key and requesting a certificate. Application must be in person or by some form of secure authenticated communication. For participant A, the authority provides a certificate of the form


Certificate Authority
KUa
KUb
CA = EKR auth [Time1, IDA, KUa]
CB = EKR auth [Time2, IDB, KUb]
(1) CA
(2) CB
A
B
Fig. 3.9 Exchange of Public-key Certificates

$$CA = E_{KR_{auth}}\ [T, IDA, KUa]$$

where KR_{auth} is the private key used by the authority. Then A may pass this certificate on to any other participant, who reads and verifies the certificate as follows –

$$D_{KU_{auth}}[C_A] = D_{KU_{auth}}\left[E_{KR_{auth}}\left[T, ID_A, KU_A\right]\right]$$
$$= (T, ID_A, KU_A)$$

The recipient uses the authority's public key, KU_{auth}, to decrypt the certificate. The elements ID_A and KU_A provide the recipient with the name and public key of the certificate's holder. The timestamp T validates the currency of the certificate. The timestamp counters the following scenario. A's private key is learned by an opponent. A generates a new private/public key pair and applies to the certificate authority for a new certificate. Meanwhile, the opponent replays the old certificate to B. If B then encrypts messages using the compromised old public key, the opponent can read those messages.

In this case, the compromise of a private key is comparable to the loss of a credit card. The owner cancels the credit card number but is a risk until all possible communicants are aware that the old credit card is obsolete. Thus, the timestamp serves as something like an expiration date. If a certificate is sufficiently old, it is assumed to be expired.

Q.15. Write short note on key exchange.

Ans. Key exchange protocols enable secure communication over an untrusted network by setting up shared keys between two or more parties. For example, SSL and TLS provide symmetric encryption keys for secure Internet transactions, IPSec protocols provide confidentiality and integrity at the IP layer, IEEE 802.11i provides data protection and integrity in wireless local area networks, and Kerberos provides authenticated client-server interaction in local area networks. While some of these protocols have been proved correct in the simplified symbolic Dolev-Yao model, most key exchange protocols in use today have not been proved secure in the complexity-theoretic model of modern cryptography.

Q.16. Define hash function. ***(R.G.P.V., Dec. 2007)***

Or

Write short note on hash functions. ***(R.G.P.V., Dec. 2004, 2008)***

Or

Write short note on hash value. ***(R.G.P.V., June 2012)***

Or

Explain hash function in detail. ***(R.G.P.V., June 2015)***

Ans. A hash function is a function, mathematical or otherwise, that takes a variable-length input string (called a ***pre-image***) and converts it to a fixed-length (generally smaller) output string (called a ***hash value***). A simple hash function would be a function that takes pre-image and returns a byte consisting of the XOR of all the input bytes.

A hash value h is generated by a function H of the form

$$h = H(M)$$

where M is a variable-length message and H(M) is the fixed-length hash value. The hash value is appended to the message at the source at a time when the message is assumed or known to be correct. The receiver authenticates that message by recomputing the hash value. Because the hash function itself is not considered to be secret, some means is required to protect the hash value.

Q.17. What is hash function and what can it be used for ? ***(R.G.P.V., June 2011)***

Or

What is hash function ? Give the basic uses of hash function. ***(R.G.P.V., May 2018)***

Ans. **Hash Function** – Refer to Q.16.

Uses of Hash Function –

(i) The symmetric encryption is used to encrypt the message plus concatenated hash code. This is similar in structure to the internal error control strategy. The same line of reasoning applies – the message must have come from X and has not been changed because only X and Y share the secret key. The hash code offers the structure or redundancy needed to get authentication. Confidentiality is also offered because encryption is applied to the entire message plus hash code.

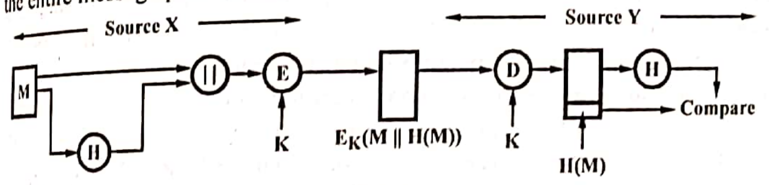
(ii) Using symmetric encryption, only the hash code is encrypted. This minimizes the processing burden for those applications that do not need confidentiality.

(iii) Using public-key encryption and using the sender's private key, only the hash code is encrypted. This provides authentication as shown in fig. 3.10 (b). Because only the sender could have produced the encrypted hash code, a digital signature is also provided.

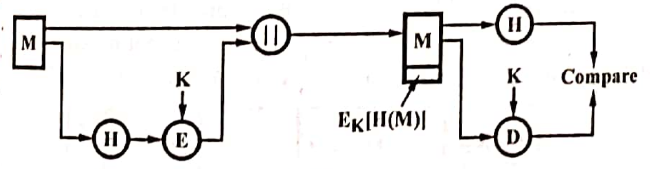
(iv) The message plus the public-key-encrypted hash code can be encrypted using a symmetric secret key to achieve confidentiality as well as a digital signature. This is a common technique.

(v) A hash function is used in this technique but there is no encryption for message authentication. In this technique, it is assumed that the two communicating parties share a common secret value S. X computes the hash value over the concatenation of M and S and appends the resulting hash value to M. Since Y keeps S, it can recompute the hash value to verify.

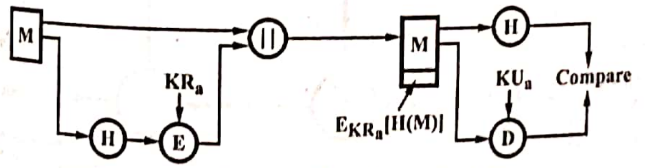
(vi) Confidentiality can be added to the approach of (v) by encrypting the entire message plus the hash code.



(a)



(b)



(c)

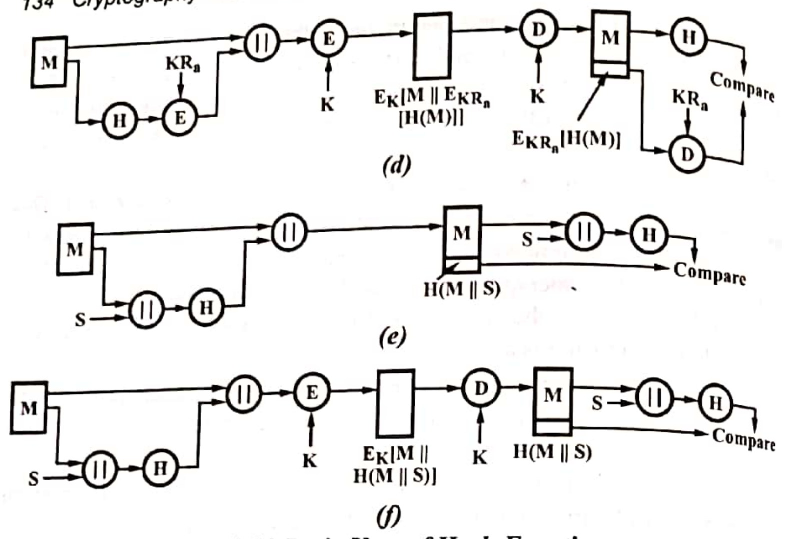


Fig. 3.10 Basic Uses of Hash Function

Q.18. What is the role of a compression function in a hash function?
(R.G.P.V., Dec. 2005)

Ans. Damgard and Merkle greatly influenced cryptographic hash function design by defining a hash function in terms of what is called a ***compression function.*** A compression function takes a fixed-length input and returns a shorter, fixed-length output. Given a compression function, a hash function can be defined by repeated applications of the compression function until the entire message has been processed. In this process, a message of arbitrary length is broken into blocks whose length is broken into blocks whose length depends on the compression function, and "padded" (for security reasons) so the size of the message is a multiple of the block size. The blocks are then processed sequentially, taking as input the result of the hash so far and the current message block, with the final output being the hash value for the message (see fig. 3.11).

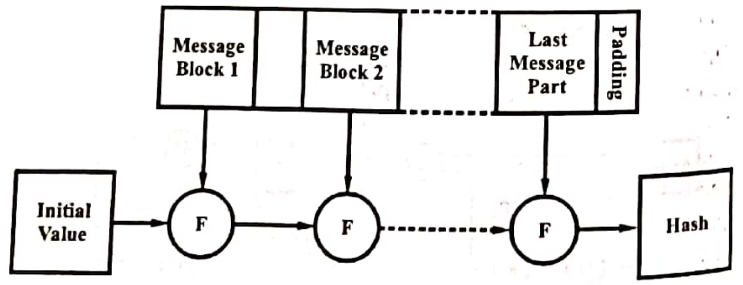


Fig. 3.11 Iterative Structure for Hash Functions; F is a Compression Function

The motivation for this iterative structure stems from the observation by Damgard and Markle that if the compression function is collision resistant, then so is the resultant iterated hash function. Therefore, the structure can be used to produce a secure hash function to operate on a message of any length. The problem of designing a secure hash function reduces to that of designing a collission-resistant compression function that operate on inputs of some fixed sizes.

Q.19. What are the various requirements for a hash function to be used for message authentication ?

Or

What are various requirements must be fulfilled by a hash function?
(R.G.P.V., Dec. 2011)

Or

What characteristics are needed in a secure hash function ?
(R.G.P.V., June 2012)

Ans. The purpose of a hash function is to produce a "fingerprint"of a file message, or other block of data. A hash function H must have the following properties to be useful for message authentication –

(i) H can be applied to a block of data of any size.

(ii) H produces a fixed-length output.

(iii) H (x) is relatively easy to compute for any given x, making both hardware and software implementations practical.

(iv) For any given value h, it is computationally infeasible to find x such that H(x) = h. This is sometimes referred to in the literature as the one-way property.

(v) For any given block x, it is computationally infeasible to find y≠x with H(y) = H(x). This is sometimes referred to as ***weak collision resistance.***

(vi) It is computationally infeasible to find any pair (x, y) such that H(x) = H(y), this is sometimes referred to as ***strong collision resistance,***

The first three properties are requirements for the practical application of hash function to message authentication.

The fourth property is the one-way property. It is easy to generate a code given a message but virtually impossible to generate a message given a code. This property is important when the authentication technique involves the use of a secret value. The secret value itself is not sent. However, if the hash function is not one way, an attacker can easily discover the secret value.

The fifth property guarantees that an alternative message hashing to the same value as a given message cannot be found. This prevents forgery when an encrypted hash code is used. For these cases, the opponent can read t message and therefore generate its hash code. However, because the oppone does not have the secret key, the opponent should not be able to alter the message without detection. If this property were not true, an attacker would be capable of the following sequence, i.e., first, intercept a message plus its encrypted hash code, second, generate an unencrypted hash code from the message, third, generate an alternate message with the same hash code.

The sixth property refers to how resistant the hash function is to a class of attack known as the birthday attack.

Q.20. Discuss the working principles of hash function.

Or

Define simple hash functions using bitwise XOR. (R.G.P.V., Dec. 2009)

Ans. All hash functions operate using the following general principles. The input (message, file etc.) 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 (XOR) of every block. This can be expressed as follows –

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

where, C_i = ith bit of the hash code, $1 \le i \le n$

m = Number of n-bit blocks in the input

b_{ij} = ith bit in jth block

$\oplus$ = XOR operation.

Fig. 3.12 shows this operation; it produces a simple parity for each bit position and is known as a longitudinal redundancy check. It is reasonably effective for random data as a data integrity check. Each n-bit hash value is equally likely. Thus, the probability that a data error will result in an unchanged hash value is 2^{-n}. With more predictably formatted data, the function is less effective. For example, in most normal text files, the high-order bit of each octet is always zero. So if a 128-bit hash value is used, instead of an effectiveness of 2^{-128}, the hash function on this type of data has an effectiveness of 2^{-112}.

	Bit 1	Bit 2	•••	Bit n
Block 1	b_{11}	b_{21}		b_{n1}
Block 2	b_{12}	b_{22}		b_{n2}
	⋮	⋮	⋮	⋮
Block m	b_{1m}	b_{2m}		b_{nm}
Hash Code	C_1	C_2		C_n

Fig. 3.12 Simple Hash Function Using Bitwise XOR

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 can be summarized as follows –

(i) Initially set the n-bit has value to zero.

(ii) 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.

This has the effect of "randomizing" the input more completely and overcoming any regularities that appear in the input.

Q.21. Write short note on birthday attack. (R.G.P.V., June 2008, June 2012)

Or

What do you mean by birthday attack ? (R.G.P.V., June 2009)

Ans. Suppose that a 64-bit hash code is used. One might think that this is quite secure. For instance, if an encrypted hash code C is transmitted with the corresponding unencrypted message M, then an opponent would need to find an M' such that H(M') = H(M) to substitute another message and fool the receiver. On average, the opponent would have to try about 2^{63} messages to find one that matches the hash code of the intercepted message.

However, a different type of attack is possible based on the birthday paradox. Yuval proposed the following strategy –

(i) The source, A, is prepared to "sign" a message by appending the appropriate m-bit hash code and encrypting that hash code with A's private key.

(ii) The opponent generates $2^{m/2}$ variations on the message, all of which convey essentially the same meaning. The opponent prepares an equal number of messages, all of which are variations on the fraudulent message to be substituted for the real one.

(iii) The two sets of messages are compared to find a pair of messages that produces the same hash code. The probability of success, by the birthday paradox, is greater than 0.5. If no match is found, additional valid and fraudulent messages are generated until a match is made.

(iv) The opponent offers the valid variation to A for signature. This signature can then be attached to the fraudulent variation for transmission to the intended recipient. Since the two variations have the same hash code, they will produce the same signature. The opponent is assured of success even though the encryption key is not known.

Thus, if a 64-bit hash code is used, the level of effort required is only on the order of 2^{32}.

The generation of many variations that convey the same meaning is not difficult. For example, the opponent could insert a number of "space-space-backspace" character pairs between words throughout the document. Variations could then be generated by substituting "space-backspace-space" in selected instances. Alternatively, the opponent could simply reword the message but retain the meaning. Fig. 3.13 shows an example.

Dear Anthony,

$\begin{Bmatrix}\text{This letter is}\\ \text{I am writing}\end{Bmatrix}$ to introduce $\begin{Bmatrix}\text{you to}\\ \text{to you}\end{Bmatrix}$ $\begin{Bmatrix}\text{Mr.}\\ \text{--}\end{Bmatrix}$ Alfred $\begin{Bmatrix}\text{P.}\\ \text{--}\end{Bmatrix}$

Barton, the $\begin{Bmatrix}\text{new}\\ \text{newly appointed}\end{Bmatrix}$ $\begin{Bmatrix}\text{chief}\\ \text{senior}\end{Bmatrix}$ jewellery buyer for $\begin{Bmatrix}\text{our}\\ \text{the}\end{Bmatrix}$

Northern $\begin{Bmatrix}\text{European}\\ \text{Europe}\end{Bmatrix}$ $\begin{Bmatrix}\text{area}\\ \text{division}\end{Bmatrix}$. He $\begin{Bmatrix}\text{will take}\\ \text{has taken}\end{Bmatrix}$ over $\begin{Bmatrix}\text{the}\\ \text{--}\end{Bmatrix}$

responsibility for $\begin{Bmatrix}\text{all}\\ \text{the whole of}\end{Bmatrix}$ our interests in $\begin{Bmatrix}\text{watches and jewellery}\\ \text{jewellery and watches}\end{Bmatrix}$

in the $\begin{Bmatrix}\text{area}\\ \text{region}\end{Bmatrix}$. Please $\begin{Bmatrix}\text{afford}\\ \text{give}\end{Bmatrix}$ him $\begin{Bmatrix}\text{every}\\ \text{all the}\end{Bmatrix}$ help he $\begin{Bmatrix}\text{may need}\\ \text{needs}\end{Bmatrix}$

to $\begin{Bmatrix}\text{seek out}\\ \text{find}\end{Bmatrix}$ the most $\begin{Bmatrix}\text{modern}\\ \text{up to date}\end{Bmatrix}$ lines for the $\begin{Bmatrix}\text{top}\\ \text{high}\end{Bmatrix}$ end of the

market. He is $\begin{Bmatrix}\text{empowered}\\ \text{authorized}\end{Bmatrix}$ to receive on our behalf $\begin{Bmatrix}\text{samples}\\ \text{specimens}\end{Bmatrix}$ of the

$\begin{Bmatrix}\text{latest}\\ \text{newest}\end{Bmatrix}$ $\begin{Bmatrix}\text{watch and jewellery}\\ \text{jewellery and watch}\end{Bmatrix}$ products, $\begin{Bmatrix}\text{up}\\ \text{subject}\end{Bmatrix}$ to a $\begin{Bmatrix}\text{limit}\\ \text{maximum}\end{Bmatrix}$

of ten thousand dollars. He will $\begin{Bmatrix}\text{carry}\\ \text{hold}\end{Bmatrix}$ a signed copy of this $\begin{Bmatrix}\text{letter}\\ \text{document}\end{Bmatrix}$

as proof of identity. An order with his signature, which is $\begin{Bmatrix}\text{appended}\\ \text{attached}\end{Bmatrix}$

$\begin{Bmatrix}\text{authorizes}\\ \text{allows}\end{Bmatrix}$ you to charge the cost to this company at the $\begin{Bmatrix}\text{above}\\ \text{head office}\end{Bmatrix}$

address. We $\begin{Bmatrix}\text{fully}\\ \text{--}\end{Bmatrix}$ expect that our $\begin{Bmatrix}\text{level}\\ \text{volume}\end{Bmatrix}$ of orders will increase in

the $\begin{Bmatrix}\text{following}\\ \text{next}\end{Bmatrix}$ year and $\begin{Bmatrix}\text{trust}\\ \text{hope}\end{Bmatrix}$ that the new appointment will $\begin{Bmatrix}\text{be}\\ \text{prove}\end{Bmatrix}$

$\begin{Bmatrix}\text{advantageous}\\ \text{an advantage}\end{Bmatrix}$ to both our companies.

Fig. 3.13 A Letter in 2^{27} Variations

Q.22. Write short note on universal hashing.

***Ans.* Definition** – A randomized algorithm H for constructing hash functions $h : U \rightarrow \{1, ..., M\}$ is universal if for all $x \neq y$ in U, we have

$$\Pr_{h \leftarrow H}[h(x) = h(y)] \leq \frac{1}{M}$$

We also say that a set H of hash functions is a universal hash function family if the procedure "choose $h \in H$ at random" is universal. (Here we are identifying the set of functions with the uniform distribution over the set).

Theorem – If H is universal, then for any set $S \subseteq U$ of size N, for any $x \in U$ (e.g., that we might want to lookup), if we construct h at random according to H, the expected number of collisions between x and other elements in S is at most N/M.

Proof – Each $y \in S$ $(y \neq x)$ has at most a 1/M chance of colliding with x by the definition of "universal". So,

(i) Let $C_{xy} = 1$ if x and y collide and 0 otherwise.

(ii) Let C_x denote the total number of collisions for x. So, $C_x = \Sigma_{y \in S, y \neq x} C_{xy}$.

(iii) We know $\mathbf{E}|C_{xy}| = \Pr(\text{x and y collide}) \leq 1/M$.

(iv) So, by linearity of expectation, $E(C_x) = \Sigma_y E(C_{xy}) < N/M$.

Q.23. Write short note on cryptographic hash function.

Ans. The term hash function has been used in computer science from quite some time and it refers to a function that compresses a string of arbitrary input to a string of fixed length. However if it satisfies some additional requirements, then it can be used for cryptographic applications and then known as cryptographic hash functions.

Cryptographic hash functions are one of the most important tool in the field of cryptography and are used to achieve a number of security goals like authenticity, digital signatures, pseudo number generation, digital steganography, digital time stamping etc.

Q.24. What is message digest (MD) ? Explain.

Ans. A message digest (MD) is a fingerprint or the summary of a message. It is similar to the concepts of *Longitudinal Redundancy Check (LRC)* or *Cyclic Redundancy Check (CRC)*. That is, it is used to verify the *integrity* of the data (i.e., to ensure that a message has not been tampered with after it leaves the sender but before it reaches the receiver). Let us understand this with the help of an LRC example.

Fig. 3.14 shows an example of LRC calculation at the sender's end. As shown, a block of bits is organized in the form of a list (as rows) in the *Longitudinal Redundancy Check (LRC)*. Here, for instance, if we want to send 32 bits, we arrange them into a list of four (horizontal) rows. Then we count how many 1 bits occur in each of the 8 (vertical) columns. [If the number of 1s in the column is odd, then we say that the column has *odd parity* (indicated by a 1 bit in the shaded LRC row); otherwise if the number of 1s in the column is even, we call it as *even parity* (indicated by a 0 bit in the shaded LRC row)]. For instance, in the first column, we have two 1s, indicating an even parity, and therefore, we have a 0 in the shaded LRC row for the first column. Similarly, for the last column, we have three 1s, indicating an odd parity, and therefore, we have a 1 in the shaded LRC row for the last column. Thus, the parity bit for each column is calculated and a new row of eight parity bits is created. These become the parity bits for the whole block. Thus, the LRC is actually a *fingerprint* of the original message.

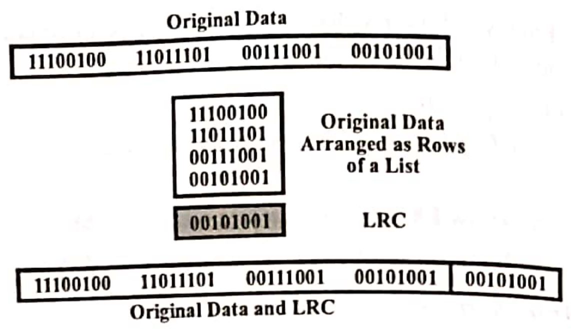


Fig. 3.14 Longitudinal Redundancy Check (LRC)

The data along with the LRC is then sent to the receiver. The receiver separates the data block from the LRC block (shown shaded). It performs its own LRC on the data block alone. It then compares its LRC values with the ones received from the sender. If the two LRC values match, then the receiver has a reasonable confidence that the message sent by the sender has not been changed, while in transit.

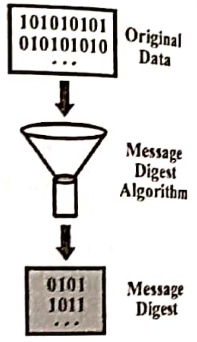


Fig. 3.15 Message Digest Concept

We perform a hashing operation (or a message digest algorithm) over a block of data to produce its hash or message digest, which is smaller in size than the original message. This concept is shown in fig. 3.15.

Actually, the message digests are not so small and straightforward to compute. Message digests usually consist of 128 or more bits. This means that the chance of any two message digests being the same is anything between 0 to at least 2^{128}. The message digest length is chosen to be so long with a purpose. This ensures that the scope for two message digests being the same.

Q.25. Explain Secure Hash Algorithm.

Ans. The Secure Hash Algorithm (SHA) was developed by the National Institute of Standards and Technology (NIST) and published as a federal information processing standard (FIPS 180) in 1993. A revised version was issued as FIPS 180-1 in 1995 and is referred to as SHA-1. The actual standards document is entitled Secure Hash Standard.

This standard specifies a SHA, which is necessary to ensure the security of the Digital Signature Algorithm (DSA). When a message of any length $< 2^{64}$ bits is input, the SHA produces a 160-bit output called a message digest. The message digest is then input to the DSA, which computes the signature for the message. Signing the message digest rather than the message often improves the efficiency of the process, because the message digest is usually much smaller than the message. The same message digest should be obtained by the verifier of the signature when the received version of the message is used as input to SHA. The SHA is called secure because it is designed to be computationally infeasible to recover a message corresponding to a given message digest, or to find two different messages which produce the same message digest. Any change to a message in transit will, with a very high probability, result in a different message digest, and the signature will fail to verify. The SHA is based on principles similar to those used by Rivest when designing MD4 and is closely modelled after the algorithm. SHA produces 160-bit hash, longer than MD5.

Q.26. Explain secure hash algorithm (SHA-1).

Ans. The SHA-1 algorithm takes as input a message with a maximum length of less than 2^{64} bits and produces as output a 160-bit message digest. The input is processed in 512-bit blocks.

The processing of a message consists of the following steps –

*(i) **Append Padding Bits*** – The message is padded so that its length is congruent to 448 modulo 512 (length $\equiv$ 448 mod 512). Padding is always added, even if the message is already of the desired length. Thus, the number of padding bits is in the range of 1 to 512. The padding consists of a single 1-bit followed by the necessary number of 0-bits.

*(ii) **Append Length*** – A block of 64 bits is appended to the message. This block is treated as an unsigned 64-bit integer (MSB first) and contains the length of the original message (before the padding).

(iii) Initialize MD Buffer – A 160-bit buffer is used to hold intermediate and final results of the hash function. The buffer can be represented as five 32-bit registers (A, B, C, D, E). These registers are initialized to the following 32-bit integers (hexadecimal values) –

A = 67452301

B = EFCDAB89

C = 98BADCFE

D = 10325476

E = C3D2E1F0

It is noted that the first four values are the same as those used in MD5. However, in the case of SHA-1, these values are stored in big-endian format, which is the MSB of a word in the low-address byte position. As 32-bit strings, the initialization values, in hexadecimal, appear as follows –

word A : 67 45 23 01

word B : EF CD AB 89

word C : 98 BA DC FE

word D : 10 32 54 76

word E : C3 D2 E1 F0

(iv) Process Message in 512-bit (16-word) Blocks – The heart of the algorithm is a module that consists of four rounds of processing of 20 steps each. The logic is shown in fig. 3.16. The four rounds has a similar structure, but each uses a different primitive logical function, which we refer to as f_1, f_2, f_3, and f_4.

Each round takes as input the current 512-bit block being processed (Y_q) and the 160-bit buffer value ABCDE and updates the contents of the buffer. Each round also makes use of an additive constant K_t, where $0 \le t \le 79$ indicates one of the 80 steps across five rounds. In fact, only four distinct constants are used. The values, in hexadecimal and decimal, are as follows –

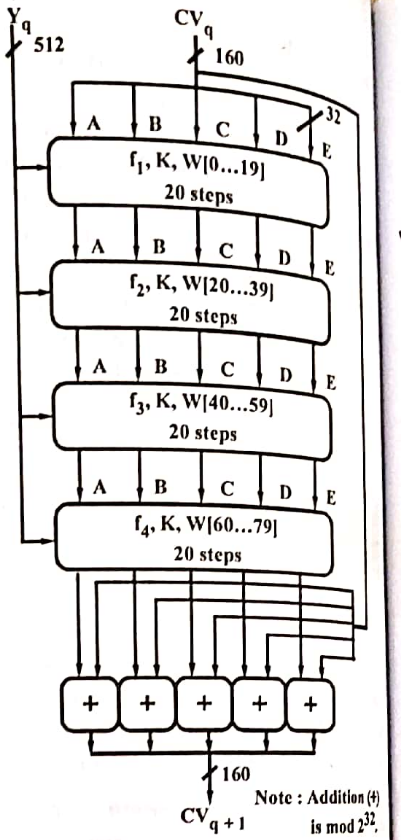


Fig. 3.16 SHA-1 Processing of a Single 512-bit Block (SHA-1 Compression Function)

Step Number	*Hexadecimal*	*Take Integer Part of*
$0 \le t \le 19$	K_t = 5A827999	$\lceil 2^{30} \times \sqrt{2} \rceil$
$20 \le t \le 39$	K_t = 6ED9EBA1	$\lceil 2^{30} \times \sqrt{3} \rceil$
$40 \le t \le 59$	K_t = 8F1BBCDC	$\lceil 2^{30} \times \sqrt{5} \rceil$
$60 \le t \le 79$	K_t = CA62C1D6	$\lceil 2^{30} \times \sqrt{10} \rceil$

The output of the fourth round (80th step) is added to the input to the first round (CV_q) to produce CV_{q+1}. The addition is done independently for each of the five words in the buffer with each of the corresponding words in CV_q, using addition modulo 2^{32}.

(v) Output – After all L 512-bit blocks have been processed, the output from the L^{th} stage is the 160-bit message digest.

The behaviour of SHA-1 can be summarized as follows –

$$CV_0 = IV$$
$$CV_{q+1} = \text{SUM}_{32}\ (CV_q,\ \text{ABCDE}_q)$$
$$MD = CV_L$$

where, IV = Initial value of the ABCDE buffer, defined in step (iii).

$ABCDE_q$ = The output of the last round of the q^{th} message block.

L = The number of blocks in the message (including padding and length fields)

SUM_{32} = Addition modulo 2^{32} performed separately on each word of the pair of inputs.

MD = Final message digest value.

Q.27. What do you understand by DSS ? With DSS because the value of K is generated for each signature, even if the message is signed twice the signatures will differ. This is not true with RSA signatures what is the practical implication of this difference ? What are some threats associated with DSS ? (R.G.P.V., Dec. 2003, June 2004)

Or

Write short note on digital signature standard. (R.G.P.V., Dec. 2007)

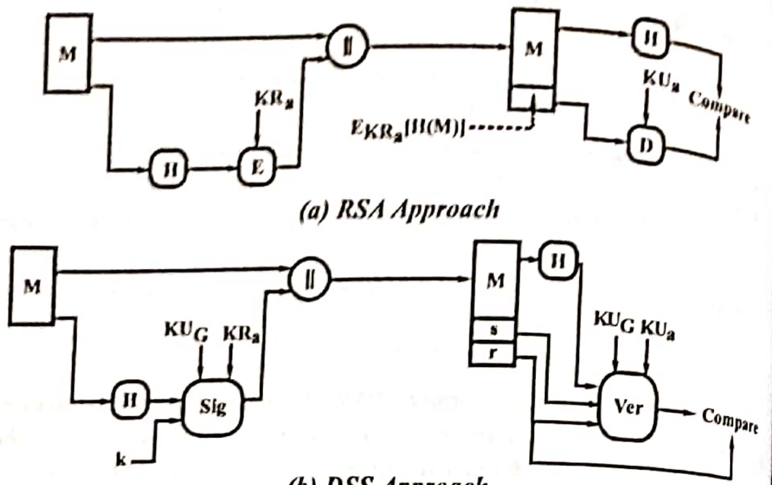
Or

Explain digital signature standards in brief. (R.G.P.V., June 2013)

Ans. The National Institute of Standards and Technology (NIST) has published Federal Information Processing Standard FIPS 186, known as the Digital Signature Standard (DSS). The DSS makes use of the Secure Hash Algorithm (SHA) and presents a new digital signature technique the Digital Signature Algorithm (DSA). The DSS was originally proposed in 1991 and revised in 1993 in response to public feedback concerning the security of the scheme. An expanded version of the standard was issued as FIP 186-2 in 2000. This latest version also incorporates digital signature algorithms based on RSA and on elliptic curve cryptography.

The DSS uses an algorithm which is designed to provide only the digital signature function. Unlike RSA, it cannot be used for encryption or key exchange. Neverthless it is a public-key technique.

The DSS approach for generating digital signatures to that used with RSA is shown in fig. 3.17. In the RSA approach, the message to be signed is input to a hash function that produces 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. The recipient takes the message and produces a hash code. The recipient also decrypts the signature using the sender's public-key.



(a) RSA Approach

(b) DSS Approach

Fig. 3.17 Two Approaches to Digital Signatures

If the calculated hash code matches the decrypted signature, the signature is accepted as valid. Because only the sender knows the private key, only the sender could have produced a valid signature.

The DSS approach also makes use of a hash function. The hash code is provided as input to a signature function along with random number k generated for this particular signature. The signature function also depends on the senders private key (KRa) and a set of parameters known to a group of communicating principles. We can consider this set to constitute a global public key (KU_G). 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. The verification function also depends on the global public key as well as the sender's public key (KVa), which is paired with the sender's private key. The output of the verification function is a value that is equal to the signature component r if the signature is valid. The signature function is such that only the sender with knowledge of the private key, could have produced the valid signature.

Threats associated with DSS is that it is criticized for being –

(i) Too secret

(ii) Too new (yet not thoroughly analyzed)

(iii) Too slow (10 to 40 times slower than RSA)

(iv) Too insecure (fixed 512-bit key).

In a subsequent revision, the fourth point was rendered moot when keys upto 1024 bits were allowed.

Q.28. Explain DSA.

Ans. The National Institute of Standards and Technology (NIST), proposed the Digital Signature Algorithm (DSA) for use in their Digital Signature Standard (DSS) to generate and verify a digital value called a *signature.*

DSA is a variant of the Schnorr and ElGamal signature algorithms. The algorithm uses the following parameters –

p = A prime number L bits long, when L ranges from 512 to 1024 and is a multiple of 64.

q = A 160-bit prime factor of p – 1.

$g = h^{(p-1)/q} \bmod p$, where h is any number less than p – 1 such that $h^{(p-1)/q} \bmod p$ is greater than 1.

x = A number less than q.

$y = g^x \bmod p$.

The algorithm also makes use of a one way hash function, H(m).

The standard specifies the secure has algorithm (SHA).

The first three parameters p, q and g are public and can be common across a network of users. The private key is x, the public key is y.

To sign a message m –

(i) Alice generates a random number, k, less than q.

(ii) Alice generates –

$r = (g^k \bmod p) \bmod q.$

$s = (k^{-1} (H(m) + xr)) \bmod q.$

The parameters r and s are her signature, she sends these to Bob.

(iii) Bob verifies the signature by computing –

$W = s^{-1} \bmod q$

$u_1 = (H(m) * W) \bmod q$

$u_2 = (rw) \bmod q$

$v = ((g^{u_1} * y^{u_2}) \bmod p) \bmod q.$

If v = r, then the signature is verified.

Table 3.3 provides a summary.

Table 3.3 DSA Signatures

Public Key –	
p	512-bit to 1024-bit prime (can be shared among a group of users)
q	160-bit prime factor of p – 1 (can be shared among a group of users)
g	$= h^{(p-1)/q} \bmod p$, where h is less than p – 1 and $h^{(p-1)/q} \bmod p > 1$ (can be shared among a group of users)
$y = g^x \bmod p$ (a p-bit number)	

Private Key –

x $< q$(a 160-bit number)

Signing –

k choose at random, less than q

r (signature) = $(g^k \bmod p) \bmod q$

s (signature) = $(k^{-1}(H(m) + xr)) \bmod q$

Verifying –

$W = s^{-1} \bmod q$

$u_1 = (H(m) * w) \bmod q$

$u_2 = (rw) \bmod q$

$v = \left(\left(g^{u_1} * y^{u_2}\right) \bmod p\right) \bmod q$

If $v = r$, then the signature is verified.

Q.29. DSA specifies that if the signature generation process results in a value of S = 0, a new value of K should be generated and the signature should be recalculated, why ? ***(R.G.P.V., June 2005)***

Ans. In DSA algorithm, at the signing end we compute

$r = (g^k \bmod p) \bmod q$

$s = [k^{-1}(H(M) + xr)] \bmod q$

signature = (r, s)

At the receiving end, for verification we compute

$w = (s')^{-1} \bmod q$

$u_1 = [H(M')w] \bmod q$

$u_2 = (r')\, w \bmod q$

$v = [(g^{u_1} y^{u_2}) \bmod p) \bmod q$

TEST : $v = r'$

If during the signing process, the value of s comes out to be zero, then at the receiving end, for verification. We compute $w = (s')^{-1}$ which results in $w = 1/0 = \infty$ (infinite quantity) from which the computation of u_1, u_2 and hence v is not possible and the scheme fails.

Thus, if the signature generation process results in a value of $s = 0$, a new value of k should be generated, so that a new s results with a non-zero value and the computation of w, u_1 u_2 and v becomes possible.

Q.30. What is the difference between digital signatures and digital certificates ? ***(R.G.P.V., June 2011)***

Ans. In digital signature, a hash function such as SHA-I or MD5 generates a unique fingerprint of the portion of data to be signed. This fingerprint is of a much smaller size than the message, it is irreversible and any change to the data will cause a mismatch with the fingerprint (also called digest) with his private key, obtaining what is called the digital signature. The signature is appended to the message, and both are sent to the recipient. The latter then extracts the message itself and uses the same hash function to obtain his or her version of the fingerprint. The user extracts the encrypted digest and use the sender's public key to decrypt it. Both results are compared, and the match will confirm that the message was received as sent. Digital signature provide authentication, privacy, non repudiation and integrity in the virtual world. We need digital signatures for secure messaging, online banking applications, online workflow applications, e-tendering, supply chain management, etc.

Digital certificates are digital documents attesting to the binding of a public key to an individual or specific entity. They allow verification of the claim that a specific public key does in fact belong to a specific individual. Digital certificates help prevent someone from using a phone key to impersonate someone else. Digital certificates are the main structured piece commonly exchanged and processed throughout a PKI. Using the digital signature, an authority server will add its weight to the authenticity of an identity by signing the related information and the user's public key into a certificate. The result is in essence saying "I guarantee that this particular public key is associated with this particular user, trust me!" If that trust is not broken or weak, the certificate will be the preferred way to make one's public key available for correspondents (and others) and have it irrefutably linked to the identity information contained therein.

Q.31. What is idea behind certification authority hierarchy ? ***(R.G.P.V., June 2014)***

Ans. Suppose John has received Mike's digital certificate and he wants to verify it. It means that John needs to de-sign the certificate using the certification authority public key. But how does John knows what's the certification authority public key. One possibility is that the certification authority of John and Mike is same. In such a case, there is no problem. However, this cannot always be guaranteed. In such a case, how can John obtain the public key of the certification authority. To resolve such problems, a certification authority hierarchy is created. Certification authority hierarchy is also known as the chain of trust. Fig. 3.18 shows that all the certification authorities are grouped into multiple levels of a certification authority hierarchy. The certification authority hierarchy starts with the root certification authority. The root certification authority contains one or more second level certification authorities. Each of the second level certification authorities contain one or more third level certification authorities, which in turn can have lower level certification authorities and so on.

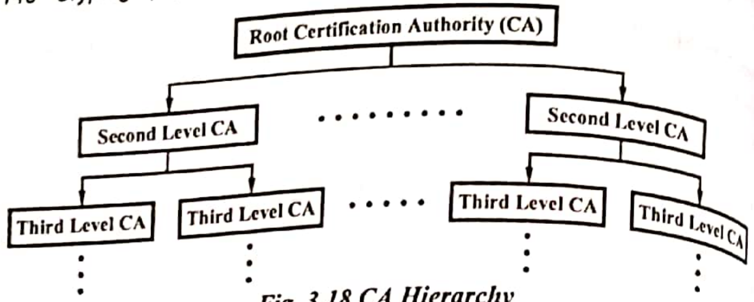


Fig. 3.18 CA Hierarchy

CRYPTANALYSIS – TIME-MEMORY TRADE-OFF ATTACK, DIFFERENTIAL CRYPTANALYSIS, SECURE CHANNEL AND AUTHENTICATION SYSTEM LIKE KERBEROS

Q.32. Define the term cryptanalysis. Explain linear and differential cryptanalysis. ***(R.G.P.V., May 2018)***

Ans. **Cryptanalysis** – Cryptanalysis is the technique of decoding messages from a non-readable format back to readable format without knowing how they were initially converted from readable format to non-readable format. In other words, it is like breaking a code. This is shown in fig. 3.19.

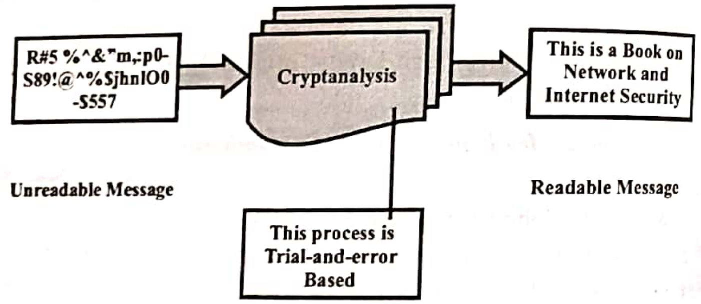

R#5 %^&"m,:p0-S89!@^%$jhnlO0-$557
Cryptanalysis
This is a Book on Network and Internet Security
Unreadable Message
Readable Message
This process is Trial-and-error Based

Fig. 3.19 Cryptanalysis

(i) Linear Cryptanalysis – This attack is based on determining linear approximations to define the transformations done in data encryption standard. For differential cryptanalysis, this procedure may search a data encryption standard key provided 2^{47} known plaintexts than 2^{47} selected plaintexts. However, this is a minor improvement, because it can be more simple to acquire known plaintext as compared to selected plaintext, it still leaves linear cryptanalysis infeasible as an attack on data encryption standard. To validate the linear cryptanalytic method, small task has been performed by other groups. The principle on which linear cryptanalysis is based as follows – For a cipher with n-bit plaintext and ciphertext blocks and an m-bit key, let the plaintext block be labeled P[1], ..., P[n], the ciphertext block C[1], ..., C[n], and the key K[1], ..., K[m]. Then specify

$$A[i, j, ..., k] = A[i] \oplus A[j] \oplus ..... \oplus A[k]$$

The aim of linear cryptanalysis is to search an effective linear equation of the form –

$$P[\alpha_1, \alpha_2, ..., \alpha_a] \oplus C[\beta_1, \beta_2, ..., \beta_b] = K[\gamma_1, \gamma_2, ... \gamma_c]$$

(where x = 0 or 1; $1 \leq a, b \leq n$, $1 \leq c \leq m$, and where the α, β and γ denotes fixed, unique bit locations) which catches with probability $p \neq 0.5$. The further p is from 0.5, the more effective the equation. Once a proposed relation is determined, the procedure is to calculate the results of the left-hand side of the preceding equations for a large number of plaintext-ciphertext pairs. When the result is zero more than half the time, consider $K[\gamma_1, \gamma_2, ... \gamma_c] = 0$. When it is one most of the time, consider $K[\gamma_1, \gamma_2, ... \gamma_c] = 1$. This provides us a linear equation on the key bits. Try to achieve more this type of relations so that we can solve for the key bits. The problem can be approached one round of the cipher at a time, with the results combined because we are dealing with linear equations.

(ii) Differential Cryptanalysis – Differential cryptanalysis is one of the most significant advances in cryptanalysis in recent years. Until 1990, differential cryptanalysis was not reported in the open literature. The first published effort appears to have been the cryptanalysis of a block cipher known as FEAL by Murphy [MURP90]. This was followed by a multiple papers by Biham and Shamir, who demonstrated this form of attack on a variety of encryption algorithms and hash functions. Their results are described in BIHA93. For this method, the most publicized results have been those that have application to DES. Differential cryptanalysis is the first published attack which is capable of decomposing DES in less than 2^{55} complexity. The technique, as reported in BIHA93 can successfully cryptanalyze DES with an effort on the order of 2^{47}, requiring 2^{47} selected plaintexts. However, 2^{47} is certainly significantly less than 2^{55}. The requirement to search 2^{47} selected plaintexts makes this attack of only theoretical interest. However, differential cryptanalysis is a powerful tool. It does not do very well against data encryption standard. According to a member of the IBM team that designed DES, the cause is that differential cryptanalysis was known to the team as early as 1974. The requirement to strengthen data encryption standard (DES) against attacks using differential cryptanalysis played a large part in the design of the S-boxes and the permutation P. As evidence of the impact of these changes, assume these comparable results described in BIHA93. Only 256 selected plaintexts are required by differential cryptanalysis of an eight-round LUCIFER algorithm, while, an attack on an eight-round version of DES needs 2^{14} selected plaintexts.

Differential Cryptanalysis Attack – The differential cryptanalysis attack is complex. Start with a change in notation for data encryption standard. Assume the real plaintext block m to consist of two halves m_0, m_1. Each round of DES maps the right-hand input into the left-hand output, and sets the right-hand output to be a function of the left-hand input and the subkey for this round. Hence, only one new 32-bit block is created at each round. When each new block $m_i (2 \leq i \leq 17)$ is labeled, then the intermediate message halves are related as given below –

$$m_{i+1} = m_{i-1} \oplus f(m_i, K_i), \quad i = 1, 2, \ldots, 16.$$

In differential cryptanalysis, begin with two messages m and m', with a known XOR difference $\Delta m = m \oplus m'$, and assume the difference between the intermediate message halves –

$$\Delta m_i = m_i \oplus m'_i$$

Then

$$\begin{aligned} \Delta m_{i+1} &= m_{i+1} \oplus m'_{i+1} \\ &= [m_{i-1} \oplus f(m_i, K_i)] \oplus [m'_{i-1} \oplus f(m'_i, K_i)] \\ &= \Delta m_{i-1} \oplus [f(m_i, K_i) \oplus f(m'_i, K_i)] \end{aligned}$$

Now, assume that a several pairs of inputs to f with the same difference provide the same output difference when the same subkey is employed. To put this more precisely, let us say that X can cause Y with probability p, when for a fraction p of the pairs in which the input XOR is X, the output XOR equals Y. Suppose that, there are a number of values of X which have high probability of causing a specific output difference. Hence, when Δm_{i-1} and Δm_i are known with high probability, then Δm_{i-1} is known with high probability. In addition, when a number of such differences are determined, it is feasible to determine the subkey employed in the function f. The whole procedure of differential cryptanalysis is based on these considerations for a single round. The procedure is to start with two plaintext messages m and m' with a provided difference and trace through a probable pattern of differences after each round to provide a probable difference for the ciphertext. For two 32-bit halves $(\Delta m_{17} \| \Delta m_{16})$, there are two probable differences. Next, m and m' are submitted for encryption to determine the actual difference under the unknown key and compare the result to the probable difference. When there is a match,

$$E_K(m) \oplus E_K(m') = (\Delta m_{17} \| \Delta m_{16})$$

then suspected that all the probable patterns at all the intermediate rounds are correct. Some deductions about the key bits can be made with that assumption. To determine all the key bits, this method must be repeated a lot of times.

Q.33. Discuss the various types of attacks.

Or

Discuss the various types of cryptanalysis attacks. (R.G.P.V., June 2015)

Ans. There are four common types of cryptanalysis attacks, as shown in fig. 3.20.

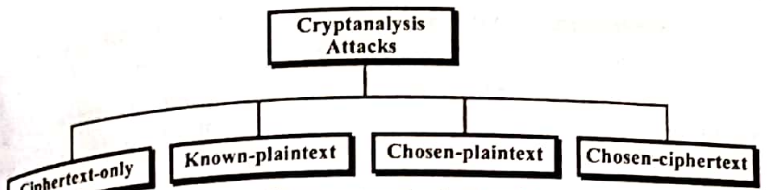


Fig. 3.20 Cryptanalysis Attacks

(i) ***Ciphertext-only Attack*** – In a ***ciphertext-only attack***, Eve has access to only some ciphertext. She tries to find the corresponding key and the plaintext. The assumption is that Eve knows the algorithm and can intercept the ciphertext. The ciphertext-only attack is the most probable one because Eve needs only the ciphertext for this attack. To thwart the decryption of a message by an adversary, a cipher must be very resisting to this type of attack. Fig. 3.21 shows the process.

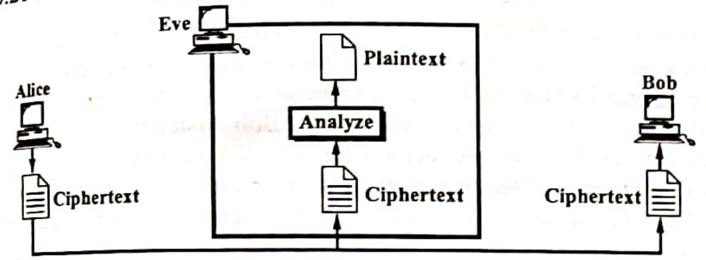


Fig. 3.21 Ciphertext-only Attack

Various methods can be used in ciphertext-only attack. We mention some common ones here.

(a) Brute-force Attack – In the ***brute-force method*** or ***exhaustive-key-search method,*** Eve tries to use all possible keys. We assume that Eve knows the algorithm and knows the key domain (the list of all possible keys). Using the intercepted cipher, Eve decrypts the ciphertext with every possible key until the plaintext makes sense. Using brute-force attack was a difficult task in the past; it is easier today using a computer. To prevent this type of attack, the number of possible keys must be very large.

(b) Statistical Attack – The cryptanalyst can benefit from some inherent characteristics of the plaintext language to launch a ***statistical attack.*** For example, we know that the letter E is the most-frequently used letter in English text. The cryptanalyst finds the mostly-used character in the ciphertext and assumes that the corresponding plaintext character is E. After finding a few pairs, the analyst can find the key and use it to decrypt the message. To prevent this type of attack, the cipher should hide the characteristics of the language.

(c) Pattern Attack – Some ciphers may hide the characteristics of the language, but may create some patterns in the ciphertext. A cryptanalyst may use a ***pattern attack*** to break the cipher. Therefore, it is important to use ciphers that make the ciphertext look as random as possible.

(ii) Known-plaintext Attack – In a ***known-plaintext attack***, Eve has access to some plaintext/ciphertext pairs in addition to the intercepted ciphertext that she wants to break, as shown in fig. 3.22.

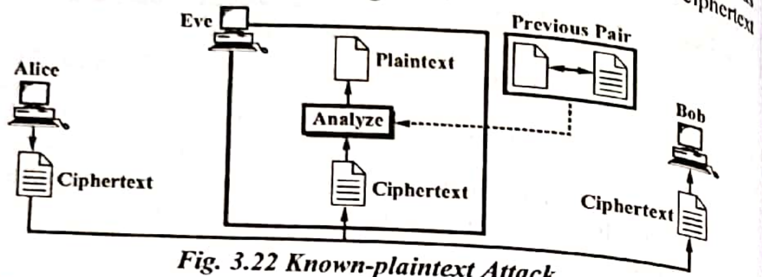


Fig. 3.22 Known-plaintext Attack

The plaintext/ciphertext pairs have been collected earlier. For example, Alice has sent a secret message to Bob, but she has later made the contents of the message public. Eve has kept both the ciphertext and the plaintext to use them to break the next secret message from Alice to Bob, assuming that Alice has not changed her key. Eve uses the relationship between the previous pair to analyze the current ciphertext. The same methods used in a ciphertext-only attack can be applied here. This attack is easier to implement because Eve has more information to use for analysis. However, it is less likely to happen because Alice may have changed her key or may have not disclosed the contents of any previous messages.

(iii) Chosen-plaintext Attack – The ***chosen-plaintext attack*** is similar to the known-plaintext attack, but the plaintext/ciphertext pairs have been chosen by the attacker herself. Fig. 3.23 shows the process.

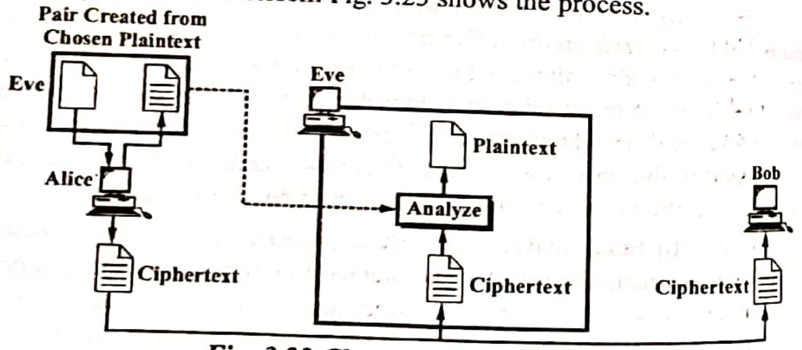


Fig. 3.23 Chosen-plaintext Attack

This can happen, for example, if Eve has access to Alice's computer. She can choose some plaintext and intercept the created ciphertext. Of course, she does not have the key because the key is normally embedded in the software used by the sender. This type of attack is much easier to implement, but it is much less likely to happen.

(iv) Chosen-ciphertext Attack – The ***chosen-ciphertext attack*** is similar to the chosen-plaintext attack, except that Eve chooses some ciphertext and decrypts it to form a ciphertext/plaintext pair. This can happen if Eve has access to Bob's computer. Fig. 3.24 shows the process.

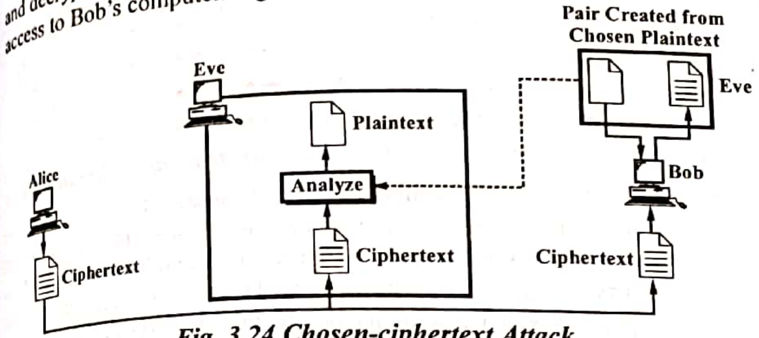


Fig. 3.24 Chosen-ciphertext Attack

Q.34. Write short note on brute-force attacks.
(R.G.P.V., Dec. 2003, June 2004, May 2018)

Ans. Refer to Q.33 (i) (a).

Q.35. List and briefly define types of cryptanalytic attacks based on what is known to the attacker. ***(R.G.P.V., May 2019)***

Ans. Refer to Q.33.

Q.36. Brief overview of time-memory tradeoff (TMTO) attacks.

Ans. Time-memory Tradeoff (TMTO) attacks on stream ciphers are a serious security threat and the resistance to this class of attacks is an important criterion in the design of a modern stream cipher. TMTO attacks are especially effective against stream ciphers where a variant of the TMTO attack can make use of multiple data to reduce the off-line and the on-line time complexities of the attack (given a fixed amount of memory).

We start with the basic TMTO attack of Hellman. Let $f : \{0, 1, ..., N-1\} \to \{0, 1, ..., N-1\}$ be the function the attacker tries to invert.

The pre-processing phase of the attack consists of constructing several tables. To construct each table, the attacker chooses m random starting points, and from each starting point x she computes the chain SP = x, f(x), $f^2(x) = f(f(x)), ..., f^t(x) = EP$, as shown in fig. 3.25 (where f is the function the attacker tries to invert). The pairs (EP, SP) are stored in a table. The attacker constructs l such tables $T_0, ..., T_{l-1}$, each for a different function f_i that is usually a slight modification of the original f (e.g., a permutation of the bits). In the on-line phase of the attack, the attacker is given z = f(y) and has to find y. For all $0 \le i \le l-1$, she applies f_i repeatedly to $f_i(y)$ (that can be easily computed given

$f_i(y)$) to get the sequence $f_i(y), f_i^2(y), \ldots\ldots, f_i^t(y)$, for each new value the attacker checks whether the obtained value is an end point in the table T_i. If a value appears as an end point in the table, the attacker takes the corresponding starting point and applies f_i sequentially, until $f_i(y)$ is reached. The point encountered just before $f_i(y)$ is indeed y.

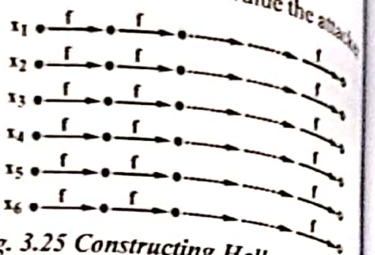


Fig. 3.25 Constructing Hellman's Table

The time complexity of the attack is t^2 applications of f and t^2 database accesses, and the memory required for the attack is mt. Since the tables should cover most of the N possible states, we have $N = mt^2$, and hence $N^2 = TM^2$ is the tradeoff curve obtained for this attack. The time complexity of the preprocessing phase is N, but as we noted before, this phase is usually neglected in the analysis of TMTO attacks.

Q.37. What was Kerberos designed for ? Explain the architecture of Kerberos. ***(R.G.P.V., Dec. 2003, June 2004, May 2011)***

Or

What are Kerberos ? Write the working principle of Kerberos. ***(R.G.P.V., May/June 2006)***

Or

What are Kerberos designed for ? Explain operation of Kerberos. ***(R.G.P.V., Dec. 2009)***

Or

What is Kerberos ? How does Kerberos work ? ***(R.G.P.V., June 2013, 2014)***

Ans. An authentication service used by many real systems is *Kerberos*, which is based on a variant of Needham-Schroeder. It is named for a multiheaded dog in Greek Mythology that used to guard the entrance to Hades (presumably to keep undesirables out). Kerberos was designed at M.I.T. to allow workstation users to access network resources in a secure way. Its biggest difference with Needham-Schroeder is its assumption that all clocks are fairly-well synchronized.

The problem that Kerberos addresses is – Assume an open distributed environment in which users at workstations wish to access services on servers distributed throughout the network. We would like for servers to be able to restrict access to authorized users and to be able to authenticate requests for service. In this environment, workstation cannot be trusted to identify its users correctly to network services. In particular, the following three threats exist –

(i) A user may gain access to a particular workstation and pretend to be another user operating from that workstation.

(ii) A user may alter the network address of a workstation so that the requests sent from the altered workstation appear to come from the impersonated workstation.

(iii) A user may eavesdrop on exchanges and use a replay attack to gain entrance to a server or to disrupt operations.

In any of these cases, an unauthorized user may be able to gain access to services and data that he or she is not authorized to access. Rather than building in elaborate authentication protocol at each server, Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users. Unlike most other authentication schemes, Kerberos relies exclusively on symmetric encryption, making no use of public key encryption.

Two versions of kerberos are in common use. Version-4 is still widely used. Version-5 corrects some of the security deficiencies of version-4 and has been issued as a proposed Internet Standard (RFC 1510). Fig. 3.26 shows the overview of Kerberos.

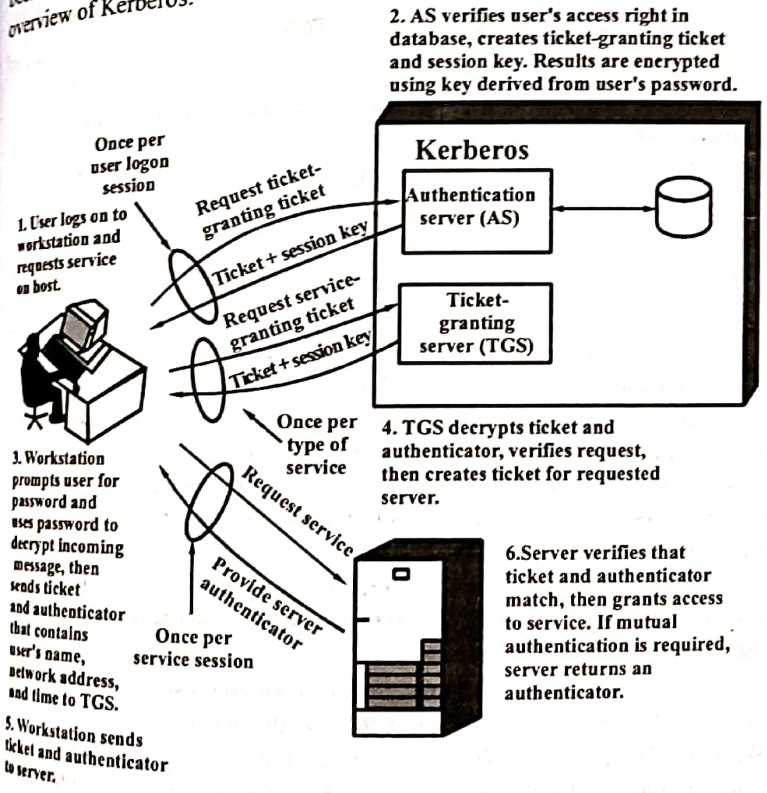

2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are enerypted using key derived from user's password.
Kerberos
Authentication server (AS)
Ticket-granting server (TGS)
Once per user logon session
1. User logs on to workstation and requests service on host.
Request ticket-granting ticket
Ticket + session key
Request service-granting ticket
Ticket + session key
Once per type of service
4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.
3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.
Request service
Provide server authenticator
Once per service session
6.Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.
5. Workstation sends ticket and authenticator to server.

Fig. 3.26 Overview of Kerberos

Kerberos involves three servers in addition to Alice (a client workstation) as shown in fig. 3.27.

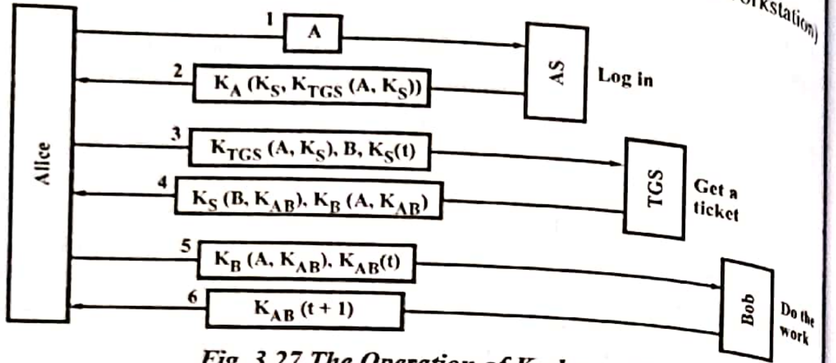

Alice
1
A
AS
Log in
2
$K_A (K_S, K_{TGS} (A, K_S))$
3
$K_{TGS} (A, K_S), B, K_S(t)$
TGS
Get a ticket
4
$K_S (B, K_{AB}), K_B (A, K_{AB})$
5
$K_B (A, K_{AB}), K_{AB}(t)$
6
$K_{AB} (t + 1)$
Bob
Do the work

Fig. 3.27 The Operation of Kerberos V4

(i) ***Authentication Server (AS)*** – The AS verifies users during login.

(ii) ***Ticket-generating Server (TGS)*** – The TGS issues "proof of identity tickets".

(iii) ***Bob the Server*** – The server actually does the work that Alice wants to perform.

AS is similar to a KDC in that it shares a secret password with every user. The TGS's job is to issue tickets that can convince the real servers that the bearer of a TGS ticket really is who he or she claims to be.

To start a session, Alice sits down at an arbitrary public workstation and types her name. The workstation sends her name to the AS in plaintext as shown in fig. 3.27. A session key and a ticket, K_{TGS} (A,K_s), comes back from the AS, intended for the TGS. These items are packaged together and encrypted using Alice's secret key, so that only Alice can decrypt them. Only when message 2 arrives does the workstation asks for Alice's password. The password is then used to generate K_A in order to decrypt message 2 and obtain the session key and TGS ticket inside it. At this point, the workstation overwrites Alice's password to make sure that it is only inside the workstation for a few milliseconds at most. If Trudy tries logging in as Alice, the password she types will be wrong and the workstation will detect this because the standard part of message 2 will be incorrect.

After she logs in, Alice may tell the workstation that she wants to contact Bob the file server. The workstation then sends message 3 to the TGS asking for a ticket to use with Bob. The key element in this request is K_{TGS} (A, K_s), which is encrypted with the TGS's secret key and used as proof that the sender really is Alice. The TGS responds by creating a session key, K_{AB}, for Alice to use with Bob. Two versions of it are sent back. The first is encrypted with only K_s, so Alice can read it. The second is encrypted with Bob's key, K_B, so Bob can read it.

Trudy can copy message 3 and try to use it again, but she will be failed by the encrypted timestamp, t, sent along with it. Trudy cannot replace the timestamp with a more recent one, because she does not know K_s, the session key Alice uses to talk to the TGS. Even if Trudy replays message 3 quickly, all she will get is another copy of message 4 which she could not decrypt the first time and will not be able to decrypt the second time either.

Now, Alice can send K_{AB} to Bob to establish a session with him. This exchange is also timestamped. The response is proof to Alice that she is actually talking to Bob, not to Trudy.

After this series of exchange, Alice can communicate with Bob under cover of K_{AB}. If she later decides she needs to talk to another server, Carol, she just repeats message 3 to the TGS, only now specifying C instead of B. The TGS will promptly respond with a ticket encrypted with K_c that Alice can send to Carol and that Carol will accept as proof that it came from Alice.

Now, Alice can access servers all over the network in a secure way and her password never has to go over the network. In fact, it only had to be in her own workstation for a few milliseconds. However, it is noted that each server does its own authorization. When Alice presents her ticket to Bob, this merely proves to Bob who sent it. Precisely What Alice is allowed to do is up to Bob.

Since the Kerberos designers did not expect the entire world to trust a single authentication server, they made provision for having multiple ***realms,*** each with its own AS and TGS. To get a ticket for a server in a distant realm, Alice would ask her own TGS for a ticket accepted by the TGS in the distant realm. If the distant TGS has registered with the local TGS, the local TGS will give Alice a ticket valid at the distant TGS. She can then do business over there, such as getting tickets for servers in that realm. However, it is also noted that for parties in two realms to do business, each one must trust the other's TGS.

Q.38. Explain the concept of Kerberos. How is it useful ?

(R.G.P.V., May 2019)

Ans. Refer to Q.37.

Q.39. What entities constitute a full Kerberos environment and what is a realm ?

(R.G.P.V., June 2008, Dec. 2008)

Or

What entities contitutes a full service Kerberos environments ?

(R.G.P.V., Dec. 2011, June 2017)

Ans. A full-service Kerberos environment consisting of a Kerberos server, a number of clients, and a number of application servers requires the following –

(i) The Kerberos server must have the user ID (UID) and hashed password of all participating users in its database. All users are registered with the Kerberos server.

(ii) The Kerberos server must share a secret key with each server. All servers are registered with the Kerberos server.

Such an environment is referred to as a **realm**.

Q.40. In the context of Kerberos, what is realm ? ***(R.G.P.V., June 2016)***

Ans. Refer to Q.39.

Q.41. What do you mean by Kerberos ? Compare it with digital signature. ***(R.G.P.V., June 2009)***

Ans. **Kerberos** – Refer to Q.37.

Comparison between Kerberos and Digital Signature –

S.No.	*Kerberos*	*Digital Signature*
(i)	It is an authentication protocol.	It is also an authentication method.
(ii)	It is based on symmetric cryptography.	We cannot use a secret (symmetric) key to both sign and verify a signature.
(iii)	Its operation is as follows – A client requests a ticket for a ticket-granting service from Kerberos. This ticket is sent to the client, encrypted in the client's secret key. To use a particular server, the client requests a ticket for that server from the ticket-granting service. The client then presents this ticket to the server along with an authenticator. If there is nothing wrong with the client's credentials, the server lets the client have access to the service.	Its operation is as follows – The sender uses a signing algorithm to sign the message. The message and the signature are sent to the receiver. The receiver receives the message and the signature and applies the verifying algorithm to the combination.

Q.42. Briefly describe the motivation for Kerberos scheme.

Or

Justify suitability of Kerberos for online real time applications. ***(R.G.P.V., Dec. 2009)***

Ans. If a set of users is provided with dedicated personal computers that have no network connections, then a user's resources and files can be protected by physically securing each personal computer. When these users instead are served by a centralized time sharing system, the time-sharing operating system must provide the security. The operating system can enforce access control policies based on user identity and use the logon procedure to indentify users.

Today, neither of these scenarios is typical. More common is a distributed architecture consisting of dedicated user workstations (clients) and distributed or centralized servers. In this environment, three approaches to security can be envisioned –

(i) Rely on each individual client workstation to assure the identity of its user or users and rely on each server to enforce a security policy based on user identification (ID).

(ii) Require that client systems authenticate themseleves to servers, but trust the client system concerning the identity of its user.

(iii) Require the user to prove identity for each service invoked. Also require that servers prove their identity to clients.

In a small closed environment, in which all systems are owned, and operated by a single organization, the first or perhaps the second strategy may suffice. But in a more open environment, in which network connections to other machines are supported, the third approach is needed to protect user information and resources housed at the server. This third approach is supported by Kerberos. Kerberos assumes a distributed client/server architecture and employs one or more Kerberos servers to provide an authentication service.

Q.43. What are the requirements of Kerberos ?

Or

What four requirements were defined for Kerberos ? Explain. ***(R.G.P.V., Dec. 2011)***

Ans. The following are the requirements for Kerberos –

(i) ***Secure*** – A network eavesdropper should not be able to obtain the necessary information to impersonate a user. More generally, Kerberos should be strong enough that a potential opponent does not find it to be the weak link.

(ii) ***Reliable*** – For all services that rely on Kerberos for access control, lack of availability of the Kerberos service means lack of availability of the supported services. Hence, Kerberos should be highly reliable and should employ a distributed server architecture, with one system able to back up another.

(iii) ***Transparent*** – Ideally, the user should not be aware that authentication is taking place, beyond the requirement to enter a password.

(iv) ***Scalable*** – The sysem should be capable of supporting large numbers of clients and servers. This suggests a modular distributed architecture.

To support these requirements, the overall scheme of Kerberos is that of a trusted third-party authentication service that uses a protocol based on that proposed by Needham-Schroeder. It is trusted in the sense that clients and servers trust Kerberos to mediate their mutual authentication.

Q.44. Differentiate between Kerberos version 4 and version 5. What are the shortcomings of Kerberos version 4 ? How are they overcome in Kerberos version 5 ?

Or

What are the principal differences between version 4 and version 5 of Kerberos ? ***(R.G.P.V., Dec. 2006, June 2012)***

Or

Differentiate Kerberos version 4 and 5. ***(R.G.P.V., June 2009)***

Or

Give the differences between version 4 and version 5 of Kerberos. ***(R.G.P.V., June 2010)***

***Ans.* Differences between Versions 4 and 5** – Version 5 is intended to address the limitations of version 4 in two areas – environmental shortcomings and technical deficiencies.

Version 4 of Kerberos was developed for use within the project Athena environment and accordingly, did not fully address the need to be of general purpose. This led to the following ***environmental shortcomings.***

(i) Encryption System Dependence – Version 4 requires the use of DES. Export restriction of DES as well as doubts about the strength of DES are thus of concern. In version 5, ciphertext is tagged with an encryption type identifier so that any encryption technique may be used. Encryption keys are tagged with a type and a length, allowing the same key to be used in different algorithms and allowing the specification of different variations on a given algorithm.

(ii) Internet Protocol Dependence – Version 4 requires the use of Internet Protocol (IP) addresses. Other address types, such as the ISO network address, are not accommodated. Version 5 network addresses are tagged with type and length, allowing any network address type to be used.

(iii) Message Byte Ordering – In version 4, the sender of a message employs a byte ordering of its own choosing and tags the message to indicate least significant byte in lowest address or most significant byte in lowest address. This technique works but does not follow established conventions. In version 5, all message structures are defined using Abstract Syntax Notation One (ASN.1) and Basic Encoding Rules (BER), which provide an unambiguous byte ordering.

(iv) Ticket Lifetime – Lifetime values in version 4 are encoded in an 8-bit quantity in units of five minutes. Thus, the maximum lifetime that can be expressed is $2^8 \times 5 = 1280$ minutes, or a little over 21 hours. This may be inadequate for some applications. In version 5, tickets include an explicit start time and end time, allowing tickets with arbitrary lifetimes.

(v) Authentication Forwarding – Version 4 does not allow credentials issued to one client to be forwarded to some other host and used by some other client. This capability would enable a client to access a server and have that server access another server on behalf of the client. Version 5 provides this capability.

(vi) Interrealm Authentication – In version 4, interoperability among N realms requires on the order of N^2 Kerberos-to-Kerberos relationships. Version 5 supports a method that requires fewer relationships.

Apart from these environmental deficiencies, there are technical deficiencies in the version 4 protocol itself. The deficiencies are listed as follows –

(i) Double Encryption – Often tickets provided to clients are encrypted twice, once with the secret key of the target server and then again with a secret key known to the client. The second encryption is not necessary and is computationally wasteful.

(ii) PCBC Encryption – Encryption in version 4 makes use of a nonstandard mode of DES known as propagating block chaining (PCBC). It has been demonstrated that this mode is vulnerable to an attack involving the interchange of ciphertext blocks. PCBC was intended to provide an integrity check as part of the encryption operation. Version 5 provides explicit integrity mechanisms allowing the standard CBC mode to be used for encryption.

(iii) Session Keys – Each ticket includes a session key that is used by the client to encrypt the authenticator sent to the service associated with that ticket. In addition, the session key may subsequently be used by the client and the server to protect messages passed during that session. However, because the same ticket may be used repeatedly to gain service from a particular server, there is the risk that an opponent will replay messages from on old session to the client or the server. In version 5, it is possible for a client and server to negotiate a subsession key, which is to be used only for that one connection. A new access by the client world results in the use of a new subsession key.

(iv) Password Attacks – Both versions are vulnerable to a password attack. The message from the AS to the client includes material encrypted with a key based on the client's password. An opponent can capture this message and attempt to decrypt it by trying various passwords. If the result of a test decryption is of the proper form, then the opponent has discovered the client's password and may subsequently use it to gain authentication credentials from Kerberos. Version 5 does provide a mechanism known as preauthentication, which should make password attacks more difficult, but it does not prevent them.

Q.45. Write any two difference between Kerberos 4 and Kerberos 5. ***(R.G.P.V., June 2016)***

Ans. Refer to Q.44.

Q.46. What are the advantages of Kerberos ?

Ans. The advantages of Kerberos are as follows –

(i) A user's password is not sent on the wire (either in plaintext or ciphertext) during session initiation.

(ii) Kerberos provides cryptographic protection against spoofing. Each service access request is mediated by the TGS, which knows that the identity of the user/client is authenticated by the Kerberos AS and processes the user/client request encrypted with the Client/TGS session key.

(iii) As each ticket has a limited validity period, long-term cryptanalytic attacks cannot be launched.

(iv) Kerberos assumes that the clocks across all the clients and the servers are synchronized. A host responds back only if the request messages have timestamp value close to the current time at the host.

(v) Kerberos provides mutual authentication. The TGS and SS can respectively get access to the Client/TGS session key and the Client/Server session key only after they can decrypt the messages containing these keys with their appropriate secret keys. The client uses this approach to indirectly authenticate the servers.

Q.47. What are the weaknesses of Kerberos ?

Ans. The weaknesses of Kerberos are as follows –

(i) Kerberos requires continuous availability of a trusted ticket-granting server for all access control and authentication checks.

(ii) Authenticity of servers requires a trusted relationship between the TGS and every service server.

(iii) Timely transactions are required to reduce chances of a user with genuine ticket being denied service.

(iv) Password guessing could still work to get the valid secret key for a user. The whole system is still dependent on the user password.

(v) Kerberos does not scale well as the number of service servers is increased. The TGS has to maintain a trustworthy relationship and maintain the secret key for each SS. Adding backup service servers further complicates the situation.

(vi) Network services cannot be accessed without obtaining Kerberos authentication. All applications run by the users in the network need to go through Kerberos authentication.

●●

UNIT 4

INFORMATION SECURITY – THREATS IN NETWORKS, NETWORK SECURITY CONTROLS, ARCHITECTURE, WIRELESS SECURITY, HONEY POTS, TRAFFIC FLOW SECURITY

Q.1. Describe the critical characteristics of information. How are they used in computer security ? ***(R.G.P.V., June 2017)***

Or

Explain the basic principles of information security. ***(R.G.P.V., June 2008, 2015)***

Or

What are the key principles of security ? ***(R.G.P.V., Dec. 2008, 2009)***

Or

Explain the following –
(i) Confidentiality (ii) Integrity (iii) Availability.
(R.G.P.V., June 2011)

Ans. Network security problems can be divided roughly into four closely intertwined areas – secrecy, authentication, non-repudiation and integrity control. Secrecy, also called confidentiality, has to do with keeping information out of the hands of unauthorized users. Authentication deals with determining whom you are talking to before revealing sensitive information or entering into a business deal. Non-repudiation deals with signatures – How do you prove that your customer really placed an electronic order for ten million left-handed doohickeys at 89 cents each when he later claims the price was 69 cents ? Or maybe he claims he never placed any order. Finally how can you be sure that a message you received was really the one sent and not something that a malicious adversary modified in transit or concocted ?

These are the four chief issues (principles) of security. There are two more, access control and availability.

The different security principles are discussed below –

(i) ***Confidentiality*** – The principle of confidentiality specifies that only the sender and the intended recipient(s) should be able to access the conte of a message. Confidentiality gets compromised if an unauthorized person is able to access a message. Example of compromising the confidentiality of a message is shown in fig. 4.1. In this figure, the user of A sends a message to user B. Another user C gets access to this message, which is not desired, and therefore, defeats the purpose of confidentiality. Example of this could be a confidential email message sent by A to B, which is accessed by C without the permission or knowledge of A and B. This type of attack is known as ***interception.***

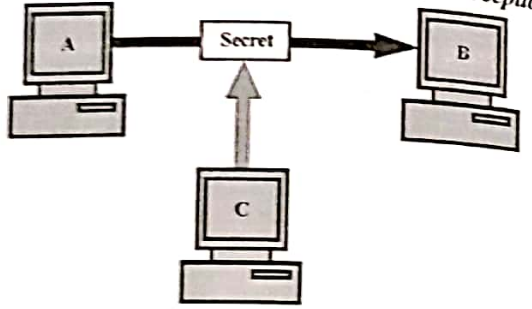


Fig. 4.1 Loss of Confidentiality

Interception causes loss of message confidentiality.

(ii) Authentication – Authentication mechanisms help to establish proof of identities. The authentication process ensures that the origin of a electronic message or document is correctly identified. For instance, suppose that user C sends an electronic document over the Internet to user B. However, the difficulty is that user C has posed as user A when he sent this document to user B. How would user B know that the message has come from user C, who is posing as user A ? A real life example of this could be the case of a user C, posing as user A sending a funds transfer request (from A's account to C's account) to bank B. The bank might happily transfer the funds from A's account to C's account – after all, it would think that user A has requested for the funds transfer. This concept is shown in fig. 4.2. This type of attack is known as ***fabrication.***

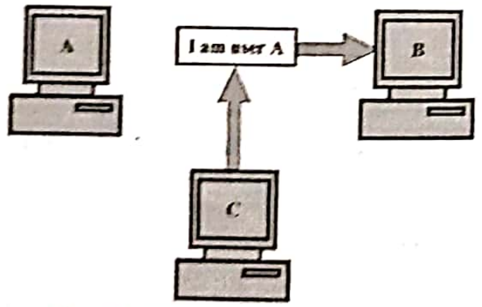


Fig. 4.2 Absence of Authentication

Fabrication is possible in absence of proper authentication mechanisms

(iii) Integrity – When the data of a message are changed after the sender sends it, but before it reaches the intended recipient, we say that the integrity of the message is lost. For example, suppose you write a check for $1000 to pay for the goods bought from the US. However, when you see your next account statement, you are started to see that the check resulted in a payment of $10000. This is the case for loss of message integrity. Conceptually, this is shown in fig. 4.3. Here, user C tampers with a message originally sent by user A, which is actually destined for user B. User C somehow manages to access it, change its contents, and send the changed message to user B. User B has no way of knowing that the contents of the message were changed after user A had sent it. User A also does not know about this change. This type of attack is known as ***modification.***

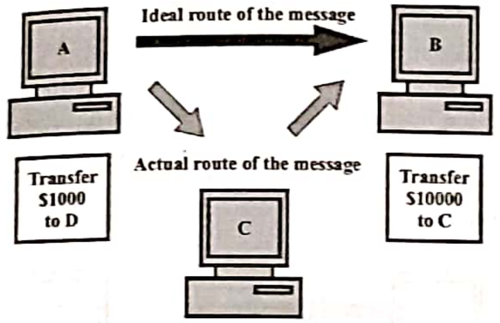


Fig. 4.3 Loss of Integrity

Modification causes loss of message integrity.

(iv) Non-repudiation – There may situations where a user sends a message, and later on refuses that he had sent that message. For example, user A could send a funds transfer request to bank B over the Internet. After the bank performs the funds transfer as per A's instructions, a could claim that he never sent the funds transfer instruction to the bank. Thus, A repudiates, or denies, his funds transfer instruction. The principle of non-repudiation defeats such possibilities of denying instructions, once sent. This is shown in fig. 4.4.

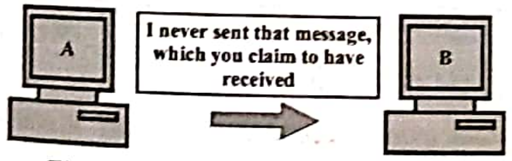


Fig. 4.4 Establishing Non-repudiation

Non-repudiation does not allow the sender of a message to refute the claim of not sending that message.

(v) Access Control – The principle of access control determines who should be able to access what. For example, we can specify that user A can view the records in a database, but cannot update them. However, user B might be allowed to make updates as well. An access control mechanism can be used to ensure this. Access control is related to two areas, role management and rule management. Role management concentrates on the user side (which user can do what), whereas rule management focuses on the resources side (which resource is accessible, and under what circumstances). Based on the decisions taken here, an access control matrix is prepared, which lists the users against a list of items they can access (e.g., it can say that user A can write to file X, but can only update files Y and Z). An ***Access Control List (ACL)*** is a subset of an access control matrix.

Access control specifies and controls who can access what.

(vi) Availability – The principle of availability says that resources (i.e., information) should be available to authorized parties at all times. For example, due to the intentional actions of another unauthorized user C, an authorized user A may not be able to contact a server computer B, as shown in fig. 4.5. This would defeat the principle of availability. This type of attack is known as ***interruption.***

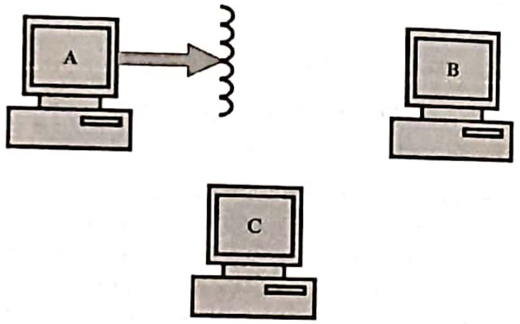


Fig. 4.5 Attack on Availability

Interruption puts the availability of resources in danger.

Q.2. Why is confidentiality an important principle of security ?
(R.G.P.V., June 2014)

Ans. Refer to Q.1 (i).

Q.3. Define the terms integrity, confidentiality, denial of service and authentication. ***(R.G.P.V., May 2018)***

***Ans.* Integrity** – Refer to Q.1 (iii).

Confidentiality – Refer to Q.1 (i).

Denial of Service – DoS may slow down or totally interrupt the service of a system. Several strategies can be used by attackers to achieve this. He may send too many bogus requests to a server that the server crashes due to the heavy load. The attacker may intercept and delete a server's response to a client, resulting in the client to believe that the server is not responding. The attacker may also intercept requests from the clients, leading the clients to send requests several times and overload the system.

Authentication – Refer to Q.1 (ii).

Q.4. What is network security ?

Ans. Security is a continuous process of protecting an object from attack. That object may be a person, an organization like a business, or property like a computer system or a file. When we consider a computer system, for example, its security involves the security of all its resources like its physical hardware components like readers, printers, the CPU, the monitors, and others. In addition to its physical resources, it also stores non-physical resources like data and information that need to be protected. In a distributed computer system like a network, the protection covers physical and non-physical resources that make up the network including communication channels and connectors like modems, bridges, switches, and servers, as well as the files stored on those servers. Therefore, in each of these cases, security means preventing unauthorized access, use, alteration, and theft or physical damage to these resources.

Q.5. Explain various types of software threats in detail.
(R.G.P.V., June 2010, 2015)

Or

What are the types of malware (malicious software) ? Briefly explain each of them. ***(R.G.P.V., June 2005, Dec. 2005, 2006, 2008)***

Ans. Fig. 4.6 shows an overall taxonomy of software threats, or malicious programs. These threats can be divided into two categories – those that need a host program, and those that are independent. The former are essentially fragments of programs that cannot exist independently of some actual application program, utility, or system program. The latter are self-contained programs that can be scheduled and run by the operating system.

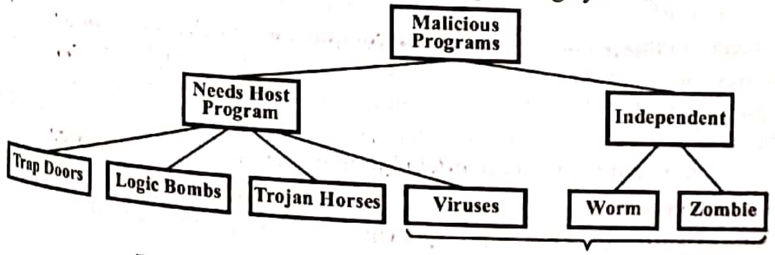


Fig 4.6 Taxonomy of Malicious Programs

Trap Doors – A trap door is a secret entry point into a program that allows someone that is aware of the trapdoor to gain access without going through the usual security access procedures. Trap doors have been used legitimately for many years by programmers to debug and test programs. This is done when the programmer is developing an application that has an authentication procedure, or a long setup, requiring the user to enter many different values to run the application. To debug the program, the developer may wish to gain special privileges or to avoid all the necessary setup and authentication. The programmer may also want to ensure that there is a method of activating the program should something be wrong with the authentication procedure that is being built into the application. The trap door is code that recognizes some special sequence of input or is triggered by being run from a certain user ID or by an unlikely sequence of events.

Trap doors become threats when they are used by unscrupulous programmers to gain unauthorized access. It is difficult to implement operating system controls for trap doors.

Logic Bomb – The logic bomb is code embedded in some legitimate program that is set to explode when certain conditions are met. Examples of conditions that can be used as triggers for a logic bomb are the presence or absence of certain files, a particular day of the week or date, or a particular user running the application. When triggered, a bomb may alter or delete data or entire files, cause a machine halt, or do some other damage.

Trojan Horses – A Trojan horse is a program or command procedure containing hidden code that, when invoked, performs some unwanted or harmful function.

Trojan horse programs can be used to accomplish functions indirectly that an unauthorized user could not accomplish directly. For example, to gain access to the files of another user on a shared system, a user could create a Trojan horse program that, when executed, changed the invoking user's file permissions so that the files are readable by any user. The author could then induce users to run the program by placing it in a common directory and naming it such that it appears to be a useful utility.

Zombie – A zombie is a program that secretly takes over another Internet-attached computer and then uses that computer to launch attacks that are difficult to trace to the zombie's creator. Zombies are used in denial-of-service attacks, against targeted Web sites. The zombie is planted on hundreds of computers belonging to unsuspecting third parties, and then used to overwhelm the target Web site by launching an overwhelming onslaught of Internet traffic.

Virus – A virus is a program that can infect other programs by modifying them. The modification includes a copy of the virus program, which can then go to infect other programs.

Biological viruses are tiny scraps of genetic code – DNA or RNA – that can take over the machinery of a living cell and trick it into making thousands of flawless replicas of the original virus. Like its biological counterpart, a computer virus carries in its instructional code the recipe for making perfect copies of itself. Lodged in a host computer, the typical virus takes temporary control of the computer's disk operating system. Then, whenever the infected computer comes into contact with an uninfected piece of software, a fresh copy of the virus passes into the new program. Thus, the infection can be spread from computer to computer by unsuspecting users who either swap disks or send programs to one another over a network. In a network environment, the ability to access applications and system services on other computers provides a perfect culture for the spread of a virus.

Worm – A worm is a program that can replicate itself and send copies from computer to computer across network connections. Upon arrival, the worm may be activated to replicate and propagate again. In addition to propagation, the worm performs some unwanted functions. An e-mail virus has some of the characteristics of a worm, because it propagates itself from system to system. However, we can still classify it as a virus because it needs a human to move it forward.

Q.6. What is worm ? What is the significant differences between a worm and a virus ? ***(R.G.P.V., June 2014)***

Or

Differentiate between viruses and worms. ***(R.G.P.V., June 2016)***

Ans. Refer to Q.5.

Q.7. What is the difference between passive and active security threats? ***(R.G.P.V., June 2013)***

Or

Why are some attacks called as passive ? Why are other attacks called active ? ***(R.G.P.V., June 2014)***

Ans. **Passive Attacks** – A passive attack attempts to learn or make use of information from the system but does not affect system resources. It means that the goal of the opponent is to obtain information. Passive attacks are those that threaten confidentiality – snooping and traffic analysis. The revealing of the information may harm the sender or receiver of the message, but the system is not affected. That is why, it is difficult to find this kind of attack until the sender or receiver finds out about the leaking of confidential information.

Active Attacks – An active attack may change system resources or affect their operation. Active attacks are those that threaten integrity and availability. Active attacks are easier to detect compared to prevent, because an attacker can launch them in a number of ways.

Q.8. What are the differences between threat and attack ?

***Ans.* Threat** – A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability.

Attack – An assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt to evade security services and violate the security policy of a system.

Q.9. Describe the model of network security.

Ans. Fig. 4.7 shows the model for network security. A message is to be sent from one party to another across some sort of Internet. The two parties involved in this transaction, must cooperate for the exchange to occur. A logical information channel is established by defining a route through the Internet from source to destination and by the cooperative use of communication protocols by the two principals (parties).

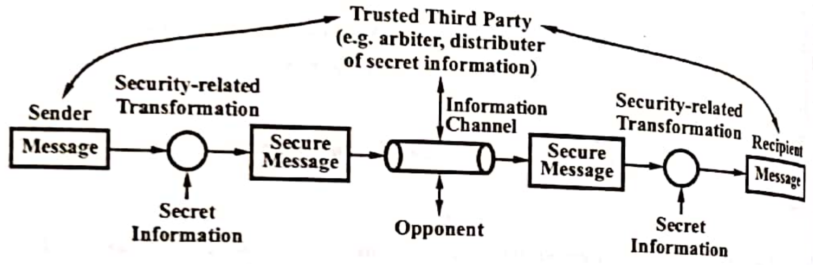


Fig. 4.7 Network Security Model

Security aspects are important when it is necessary or desirable to protect the information transmission from an opponent who may present a threat to confidentiality, authenticity, and so on. The techniques for providing security have two components –

(i) A security-related transformation on the information to be transferred. Examples are the encryption of the message, and the addition of a code based on the contents of the message.

(ii) Some secret information shared by the two principals and, it is expected, unknown to the opponent. For example, an encryption key used in conjunction with the transformation to scramble the message before transmission and unscramble it on reception.

A trusted third party may be required to get secure transmission. For instance, a third party may be responsible for distributing the secret information to the two principals while keeping it from any opponent. Or a third party may be required to arbitrate disputes between the two principals concerning the authenticity of a message transmission.

According to this general model, there are four basic tasks in designing a particular security service –

(i) Design an algorithm for performing the security-related transformation. The algorithm should be such that an opponent cannot defeat its purpose.

(ii) Generate the secret information to be used with the algorithm.

(iii) Find methods for the distribution and sharing of the secret information.

(iv) Specify a protocol to be used by the two principals that makes use of the security algorithm and the secret information to get a particular security service.

Q.10. What are various security mechanism to achieve security goals ?

Ans. Some security mechanism to achieve security goals are –

(i) Encipherment – Encipherment can provide confidentiality. It can also be used to complement other mechanisms to provide other services. Today, two techniques are used for enciphering – cryptography and steganography.

(ii) Data Integrity – The data integrity mechanism appends to the data a short checkvalue that has been created by a specific process from the data itself. The data and the checkvalue are received by the receiver. From the received data the receiver creates a new checkvalue and compares the newly created checkvalue with the one received. The integrity of data has been preserved if the two checkvalues are same.

(iii) Digital Signature – A digital signature is a means by which the sender can electronically sign the data and the receiver can electronically verify the signature. The sender uses a process that involves showing the she owns a private key related to the public key that she has announced publicly. The receiver uses the sender's public key to prove that the message is indeed signed by the sender who claims to have sent the message.

(iv) Authentication Exchange – In authentication exchange, two entities exchange some messages to prove their identity to each other. For example, one entity can prove that she knows a secret that only she is supposed to know.

(v) Routing Control – Routing control means selecting and continuously changing different available routes between the sender and the receiver to prevent the opponent from eaves dropping on a particular route.

(vi) Notarization – Notarization means selecting a third trusted party to control the communication between two entities. This can be done to prevent repudiation. The receiver can involve a trusted party to store the sender request in order to prevent the sender from later denying that she has made such a request.

(vii) Access Control – Access control uses methods to prove that a user has access right to the data or resources owned by a system.

Q.11. With a proper diagram, bring out the taxonomy of security goals and the categorization of various security attacks while realizing these goals.

***Ans.* Security Goals** – Fig. 4.8 shows three security goals – confidentiality, integrity and availability.

(i) Confidentiality – Confidentiality is the most common aspect of information security. It is required to protect our confidential information. An organization requires to protect against those malicious actions that endanger the confidentiality of its information. In the military concealment of sensitive information is very crucial.

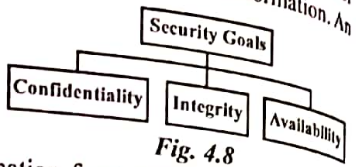


Fig. 4.8

In industry, concealment of information from competitors is necessary the operation of the organization. In banking, customers' accounts require to be kept secret.

(ii) Integrity – Information changes constantly. In banking, when a customer deposits or withdraws money, the balance of his account requires to be changed. Integrity requires that changes need to be done only by authorized entities and through authorized mechanisms. It is not necessary that the integrity violation is the result of a malicious act. An interruption in the system, like a power failure, may also create undesired changes in some information.

(iii) Availability – It is required that the information created and stored by an organization be available to authorized person. Information changes constantly, which means it must be accessible to authorized person. The unavailability of information is just as dangerous for an organization as the lack of confidentiality or integrity.

Attacks – Three security goals – confidentiality, integrity, and availability – can be threatened by security attacks. Fig. 4.9 shows the taxonomy of security attacks.

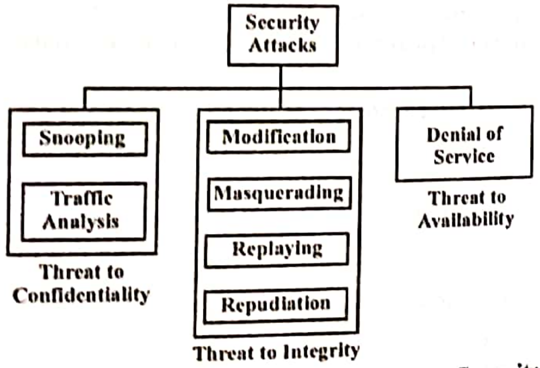


Fig. 4.9 Taxonomy of Attacks with Relation to Security Goals

(i) Attacks Threatening Confidentiality – There are two types of attacks which threaten the confidentiality of information – snooping and traffic analysis.

(a) Snooping – Snooping means to unauthorized access to or interception of data. For instance, a file transferred through the Internet may have confidential information. An unauthorized person may intercept the transmission and use the contents for his own benefit. The data can be made nonintelligible to the intercepter by using encipherment techniques to prevent snooping.

(b) Traffic Analysis – Encipherment of data may make it nonintelligible for the intercepter, however, he can get some other type information by monitoring online traffic. For example, he can get the electronic address of the sender or the receiver. He can collect pairs of requests and responses to help his guess the nature of transaction.

(ii) Attacks Threatening Integrity – The integrity of data can be threatened by the following attacks –

(a) Modification – The attacker alters the information to make it beneficial to himself after intercepting or accessing information. For example, a customer sends a message to a bank to perform some transaction. The attacker intercepts the message and alters the mode of transaction to benefit himself.

(b) Masquerading – It occurs when the attacker impersonates somebody else. For example, an attacker may steal the bank card and PIN of a bank customer and pretend that he is that customer. Also the attacker pretends instead to be the receiver entity. For example, a user attempts to contact a bank, but another site pretends that it is the bank and get some important information from the user.

(c) Replaying – The attacker gets a copy of a message sent by a user and later attempts to replay it. For instance, a person sends a request to his bank to ask for payment to the attacker, who has done a job for him. The attacker intercepts the message and sends it again to get another payment from the bank.

(d) Repudiation – It is performed by one of the two parties involved in the communication – the sender or the receiver. The sender of the message may later deny that he has sent the message. The receiver of the message may later deny that he has received the message.

A bank customer asking him bank to send some money to a third party but later denying that he has made such a request is an example of denial by the sender. When a person buys a product from a manufacturer and pays for it electronically, but the manufacturer later denies having received the payment and asks to be paid is an example of denial by the receiver.

(iii) Attacks Threatening Availability –

Denial of Service (DoS) – DoS may slow down or totally interrupt the service of a system. Several strategies can be used by attackers to achieve this. He may send too many bogus requests to a server that the server crashes due to the heavy load. The attacker may intercept and delete a server's response to a client, resulting in the client to believe that the server is not responding. The attacker may also intercept requests from the clients, leading the clients to send requests several times and overload the system.

Q.12. Explain the concept of honeypot. ***(R.G.P.V., Dec. 2003, June 2004)***

Or

Write short note on honeypot.

(R.G.P.V., Dec. 2006, June 2008, Dec. 2008)

Or

What is honey-pot ? ***(R.G.P.V., June 2009)***

Ans. A relatively recent innovation in intrusion detection technology is the honeypot. Honeypots are decoy systems that are designed to lure a potential attacker away from critical systems. Honeypots are designed to –

(i) Divert an attacker from accessing critical systems.

(ii) Collect information about the attackers activity.

(iii) Encourage the attacker to stay on the system long enough for administrators to respond.

These systems are filled with fabricated information designed to appear valuable but that a legitimate user of the system would not access. Thus, any access to the honeypot is suspect. The system is instrumented with sensitive monitors and event loggers that detect these accesses and collect information about the attacker's activities. Because any attack against the honeypot is made to seem successful, administrators have time to mobilize and log and track the attacker without ever exposing productive systems.

Initial efforts involved a single honeypot computer with IP addresses designed to attract hackers. More recent research has focused on building entire honeypot networks that emulate an enterprise, possibly with actual or simulated traffic and data. Once hackers are within the network, administrators can observe their behaviour in detail and figure out defenses.

Q.13. Write short note on traffic flow security.

Ans. Traffic-flow security is the use of measures that conceal the presence and properties of valid messages on a network to prevent traffic analysis. This can be done by operational procedures or by the protection resulting from features inherent in some cryptographic equipment. Techniques used include–

(i) Changing radio callsigns frequently

(ii) Encryption of a message's sending and receiving addresses (codress messages)

(iii) Causing the circuit to appear busy at all times or much of the time by sending dummy traffic

(iv) Sending a continuous encrypted signal, whether or not traffic is being transmitted. This is also called masking or link encryption.

Traffic-flow security is one aspect of communications security.

FIREWALLS – DESIGN AND TYPES OF FIREWALLS, PERSONAL FIREWALLS, IDS

Q.14. What are the firewalls ? How Firewall guards corporate networks? ***(R.G.P.V., June 2013)***

Or

Write short note on firewalls. ***(R.G.P.V., June 2017)***

Ans. A ***firewall*** defines a single choke point that keeps unauthorized users out of the protected network, prohibits potentially vulnerable services from entering or leaving the network, and provides protection from various kinds of IP spoofing and routing attacks. The use of a single choke point simplifies security management because security capabilities are consolidated on a single system or set of systems.

A firewall works just as a sentry. The implementation of a firewall guards a corporate network by standing between the network and the outside world. All traffic between the network and the Internet must pass through the firewall. It depends on firewall that the traffic should be allowed to flow or not. Fig. 4.10 shows this.

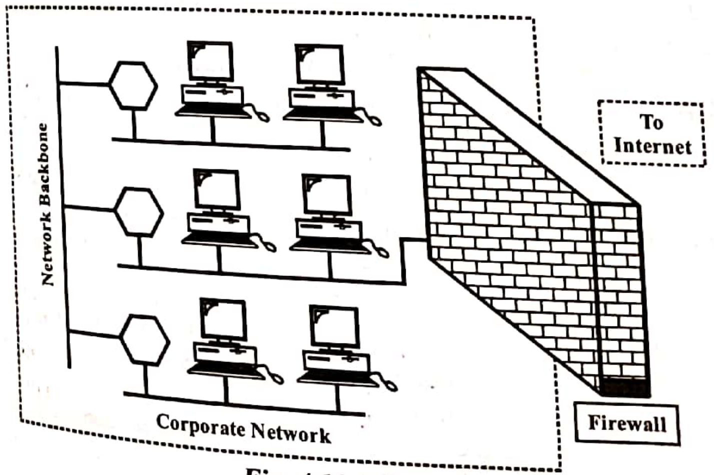


Fig. 4.10 Firewall

aracteristics of a good firewall implementation.
(R.G.P.V., June 2013)

Ans. The following capabilities (or characteristics) are within the scope of a firewall –

(i) A firewall defines a single choke point that keeps unauthorized users out of the protected network, prohibits potentially vulnerable services from entering or leaving the network, and provides protection from various kinds of IP spoofing and routing attacks. The use of a single choke point simplifies security management because security capabilities are consolidated on a single system or set of systems.

(ii) A firewall provides a location for monitoring security-related events. Audits and alarms can be implemented on the firewall system.

(iii) A firewall is a convenient platform for several Internet functions that are not security related. These include a network address translator, which maps local addresses to Internet addresses, and a network management function that audits or logs Internet usage.

(iv) A firewall can serve as the platform for IPSec. Using the tunnel mode capability, the firewall can be used to implement virtual private networks.

Q.16. Discuss the firewall design principles.
(R.G.P.V., June 2006, Dec. 2008)

Or

Write short note on firewall design principles.
(R.G.P.V., June 2005, Dec. 2005)

Ans. Following are the design goals or design principles of firewall –

(i) All traffic from inside to outside, and vice versa, must pass through the firewall. This is achieved by physical blocking all access to the local network except via the firewall.

(ii) Only authorized traffic will be allowed to pass. Various types of firewalls are used, which implement various types of security policies.

(iii) The firewall itself is immune to penetration. This implies that use of a trusted system with a secure operating system.

Q.17. What are the two main attacks on corporate networks ?
(R.G.P.V., June 2014)

Ans. The two main attacks on corporate networks are as follows –

(i) Most corporations contain huge amount of valuable and confidential data in their networks. Leaking of this critical information to competitors can be a great setback.

(ii) There is a great danger of the outside elements entering a corporate network to create havoc.

Q.18. What are the limitations of a firewall ? ***(R.G.P.V., June 2014)***

Ans. Firewalls have the following limitations –

(i) The firewall does not protect against internal threats, such as a disgruntled employee or an employee who unwillingly cooperates with an external attacker.

(ii) The firewall cannot protect against the transfer of virus-infected programs or files. Because of the variety of operating systems and applications supported inside the perimeter, it would be impractical and perhaps impossible for the firewall to scan all incoming files, e-mail, and messages for viruses.

(iii) The firewall cannot protect against attacks that bypass the firewall. Internal systems may have dial-out capability to connect to an ISP. An internal LAN may support a modem pool that provides dial-in capability for travelling employees and telecommuters.

Q.19. What are the various functions of firewall ?

Or

Explain the functionality of firewalls. ***(R.G.P.V., June 2011)***

Ans. The main functions of a firewall are as follows –

(i) Access Control – A firewall filter incoming as well as outgoing packets. A firewall is said to be configured with a ruleset based on which it decides which packets are to be allowed and which are to be dropped.

(ii) Address/Port Translation – Network Address Translation (NAT) was initially devised to alleviate the serious shortage of IP address by providing a set of private addresses that could be used by system administrators on their internal networks but that are globally invalid. Publicly accessible machines within an organization, such as Web servers, may or may not have public Internet addresses. However, in the latter case, it is possible to conceal the addressing schema of these machines from the outside world through the use of NAT. Through NAT, internal machines, though not visible on the Internet, can establish a connection with external machines on the Internet. NATing is often done by firewalls.

(iii) Logging – A sound security architecture will ensure that each incoming or outgoing packet encounters at least one firewall. The firewall can log all anomalous packets or flows for later study. These logs are very useful for studying attempts at intrusion together with various worm and DDoS attacks.

(iv) Authentication, Caching, etc. – Some types of firewalls perform authentication of external machines attempting to establish a connection with an internal machine. A special type of firewall called a Web proxy authenticates internal users attempting to access an external service. Such a firewall is also used to cache frequently requested webpages. This results in decreased response time to the client while saving communication bandwidth.

Q.20. What is the use of firewall ? Explain firewall design principles.
(R.G.P.V., May 2018)

Ans. **Use of Firewall** – Refer to Q.19.

Firewall Design Principles – Refer to Q.16.

Q.21. List four techniques used by firewalls to control access and enforce a security policy.
(R.G.P.V., Dec. 2011)

Ans. The four techniques that firewalls use to control access and enforce the site's security policy are as follows –

(i) Service Control – The firewall may filter traffic on the basis of IP addresses, TCP, UDP, port numbers, and DNS and FTP protocols in addition to providing proxy software that receives and interprets each service request before passing it on.

(ii) Direction Control – Where permission for traffic flow is determined from the direction of the requests.

(iii) User Control – Where access is granted based on which user is attempting to access the internal protected network; may also be used on incoming traffic.

(iv) Behaviour Control – In which access is granted based on how particular services are used, for example, filtering email to eliminate spam.

Q.22. Discuss policies and access control lists for access to various types of services.

Ans. High-level policies for access to various types of services are formulated within an organization or campus. Examples of these include the following –

(i) All received e-mail should be filtered for spam and viruses.

(ii) All HTTP requests by external clients for access to authorized pages of the organization's website should be permitted.

(iii) The organization's employees should be allowed to remotely log into authorized internal machines. However, all such communication should be authenticated and encrypted.

(iv) DNS queries made by external clients should be allowed provided they pertain to addresses of the organization's publicly accessible services such as the Web server or the external e-mail server. However, queries related to the IP addresses of internal machines should not be entertained.

(v) Only two types of outgoing traffic are permitted. First, all e-mail from within the organization to the outside world are permitted. Second, requests emanating from within the organization for external Webpages are permitted. However, requests for pages from certain "inappropriate" Websites should be denied.

High-level policies are translated into a set of rules that comprise an Access Control List. A rule specifies the action to be taken as a function of –

(i) The packet's source IP address and port number

(ii) The packet's destination IP address and port number

(iii) The transport protocol in use

(iv) The packet's direction – incoming or outgoing.

Policies can, in general, be either permissive or restrictive. A permissive policy permit all packets except those that are explicitly forbidden. A restrictive policy drop all packets except those that are explicitly permitted. The ACL in table 4.1 implements a restrictive policy – the default action is Deny as expressed in rules 5 and 8. The rules are scanned top to bottom. As soon as a rule is found that matches the packet's attributes (IP addresses, port numbers, etc.) the action in that rule is taken and no further rules are processed for that packet.

Table 4.1 Example Access Control List

S. No.	*In-bound (I) or Out-bound (O)*	*Transport Protocol*	*Src. IP Addr.*	*Src. Port*	*Dest. IP Addr.*	*Dest. Port*	*Action*	*Comment*
(i)	I	TCP	Any	Any	MS	25	Permit	Allow incoming e-mail
(ii)	I	TCP	Any	Any	WS	80	Permit	Allow requests for organization's webpages
(iii)	I	UDP	Any	Any	NS	53	Permit	Allow DNS queries
(iv)	I	IPSec	Any	Any	*	*	Permit	Allow incoming VPN traffic
(v)	I	Any	Any	Any	Any	Any	Deny	Forbid all other incoming traffic
(vi)	O	TCP	Any	Any	Any	25	Permit	Allow outgoing e-mail
(vii)	O	TCP	Any	Any	*	80	Permit	Allow requests for external webpages
(viii)	O	Any	Any	Any	Any	Any	Deny	Forbid all other outgoing traffc

Q.23. Why access control is more important in security ?
(R.G.P.V., June 2016)

Ans. Access control is a security technique that can be used to regulate who or what can view or use resources in a computing environment.

There are two main types of access control – physical and logical. Physical access control limits access to campuses, buildings, rooms and physical IT assets.

Logical access control limits connections to computer networks, system files and data.

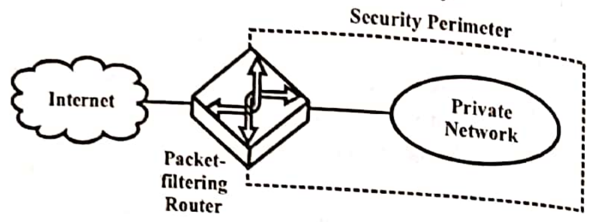
Q.24. Explain various types of firewalls.

Or

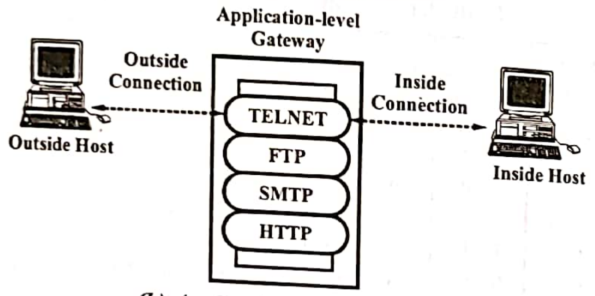
What are some weaknesses of a packet-filtering router ? What is an application level gateway and circuit level gateway ?

(R.G.P.V., Dec. 2003, June 2004, 2009)

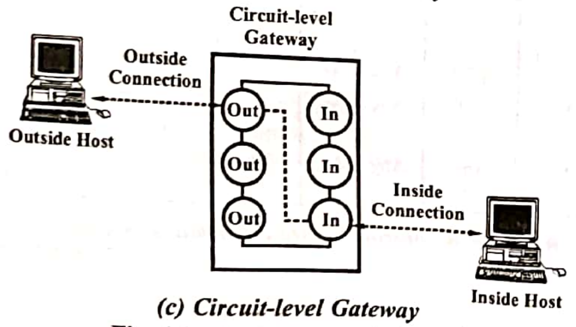
Ans. Fig. 4.11 shows the three common types of firewalls – packet filters, application-level gateways, and circuit-level gateways.



(a) Packet-filtering Router



(b) Application-level Gateway



(c) Circuit-level Gateway

Fig. 4.11 Three Types of Firewalls

(i) ***Packet-filtering Router*** – A packet filtering router shown in fig. 4.11 (a), applies a set of rules to each incoming IP packet and then forwards or discards the packet. The router is typically configured to filter packets going in both directions (from and to the internal network). Filtering rules are based on information contained in a network packet – source IP address, destination IP address, source and destination transport-level address, IP protocol field and the interface.

The packet filter is typically setup as a list of rules based on matches to fields in the IP or TCP header. If there is a match to one of the rules, that rule is invoked to determine whether to forward or discard the packet. If there is no match to any rule, then a default action is taken. Two default policies are possible–

(a) **Default = discard** – That which is not expressly permitted is prohibited.

(b) **Default = forward** – That which is not expressly prohibited is permitted.

The default discard scheme is the more conservative. Initially, everything is blocked, and services must be added on a case-by-case basis. This policy is more visible to users, who are more likely to see the firewall as a hindrance. The default forward scheme increases ease of use for end users but provides reduced security. In essence, the security administrator must react to each new security threat as it becomes known.

One advantage of a packet-filtering router is its simplicity. Also, packet filters are transparent to users and are very fast.

Packet-filtering router also has some weakness. Some of the weaknesses of packet-filtering router are as follows –

(a) Because packet filter firewalls do not examine upper-layer data, they cannot prevent attacks that employ application-specific vulnerabilities or functions. For example, a packet filter firewall cannot block specific application commands so that all functions available within that application will be permitted.

(b) Because of the limited information available to the firewall, the logging functionality present in packet filter firewalls is limited. Packet filter logs normally contain the same information used to make access control decisions.

(c) Because of the lack of upper-layer functionality of the firewall, most packet filter firewalls do not support advanced user authentication policies.

(d) They are generally vulnerable to attacks and exploits that take advantage of problems within the TCP/IP specification and protocol stack, such as network layer address spoofing.

Many packet filter firewalls cannot detect a network packet in which the OSI Layer 3 addressing information has been modified. Spoofing attacks are generally employed by intruders to bypass the security controls implemented in a firewall platform.

(e) Finally, due to small number of variables used in access control decisions, packet filter firewalls are susceptible to security breaches caused by improper configurations. In other words, it is easy to accidentally configure a packet filter firewall to allow traffic types, sources and destinations that should be denied based on an organization's information security scheme.

(ii) Application-level Gateway – An application-level gateway, also known as a *proxy server*, acts as a relay of application-level traffic as shown in fig. 4.11 (b). The user contacts the gateway using a TCP/IP application, such as Telnet or FTP, and the gateway asks the user for the name of the remote host to be accessed. When the user responds and inputs a valid user ID and authentication information, the gateway contacts the application on the remote host and relays TCP segments containing the application data between the two endpoints. If the gateway does not implement the proxy code for a specific application, the service is not supported and cannot be forwarded across the firewall. Further, the gateway can be configured to support only specific features of an application that the network administrator considers acceptable while denying all other features.

Application-level gateways tend to be more secure than packet filters. Rather than trying to deal with the numerous possible combinations that are to be allowed and forbidden at the TCP and IP level, the application level gateway need only scrutinize a few allowable applications. In addition, it is easy to log and audit all incoming traffic at the application level.

A prime disadvantage of the application-level gateway is the additional processing overhead on each connection. In effect, there are two spliced connections between the end users, with the gateway at the splice point, and the gateway must examine and forward all traffic in both directions.

(iii) Circuit-level Gateway – A third type of the firewall is the circuit-level gateway shown in fig. 4.11 (c). It can be a stand-alone system or it can be a specialized function performed by an application-level gateway for certain applications. A circuit-level gateway does not allow an end-to-end TCP connection. Rather, the gateway sets up two TCP connections, one between itself and a TCP user on an inner host and other between itself and a TCP user on an outside host. Once the two connections are established, the gateway typically relays TCP segments form one connection to other without examining the contents. The security function consists of determining which connections will be allowed.

A typical use of circuit-level gateway is a situation in which the system administrator trusts the internal users. The gateway can be configured to support application-level or proxy service on inbound connections and circuit-level functions for outbound connections. In this configuration, the gateway can incur the processing overhead of examining incoming application data for forbidden functions but does not incur that overhead on outgoing data.

Q.25. What is firewall and its types ? ***(R.G.P.V., May 2019)***

Ans. Refer to Q.14 and Q.24.

Q.26. What is firewall ? List the type of firewalls and explain. Draw a schematic diagram of a packet filtering router used as a firewalls. ***(R.G.P.V., June 2016)***

Ans. Refer to Q.14 and Q.24.

Q.27. Write a short note on packet filters. ***(R.G.P.V., June 2015)***

Ans. Refer to Q.24 (i).

Q.28. What is the role of application level gateway ? ***(R.G.P.V., June 2011)***

Or

Write short note on application level gateway. ***(R.G.P.V., Dec. 2011)***

Or

Explain the working of application level gateway. ***(R.G.P.V., June 2012)***

Ans. Refer to Q.24 (ii)

Q.29. How is a circuit gateway different from application gateway. ***(R.G.P.V., June 2013)***

Ans. Refer to Q.24 (ii) and (iii).

Q.30. What are the weaknesses of a packet filtering router ? Discuss its solution. ***(R.G.P.V., June 2012)***

Ans. **Weakness of a Packet Filtering Router** – Refer to Q.24 (i).

Some of the attacks on packet-filtering routers and the corresponding countermeasures are as follows –

(i) IP Address Spoofing – An intruder can send packets from the outside with a source IP address field containing an address of an internal user. This attack can be defeated by discarding packets with an inside source address if the packet arrives on an external interface.

(ii) Source Routing Attacks – The source station can specify the route that a packet should take as it moves along the Internet, in the hopes that this will bypass security measures that do not analyze the source routing information. This attack can be defeated by discarding all packets that use this option.

(iii) Tiny Fragment Attacks – The IP fragmentation option is used by the intruder to create small fragments and force the TCP header information into a separate packet fragment. The attacker hopes that only the first fragment is checked by the filtering router, and the remaining fragments are not checked. This attack can be defeated by discarding all packets where the protocol type is TCP and the IP fragment offset is 1.

Q.31. What is bastion host ? List some common characteristics of a bastion host.

Or

Explain bastion host. *(R.G.P.V., Dec. 2009)*

Ans. A bastion host is a system identified by the firewall administrator as a critical strong point in the network's security. Typically, it serves as a platform for an application-level or circuit-level gateway. Common characteristics of a bastion host are as follows –

(i) The bastion host hardware platform executes secure version of its operating system, making it a trusted system.

(ii) Only the services that the network administrator considers essential are installed on the bastion host. These include proxy applications such as Telnet, DNS, FTP, SMTP, and user authentication.

(iii) The bastion host may require additional authentication before a user is allowed access to the proxy services. In addition, each proxy service may require its own authentication before granting user access.

(iv) Each proxy is configured to support only a subset of the standard application's command set.

(v) Each proxy is configured to allow access only to specific host systems. This means that the limited command/feature set may be applied only to a subset of systems on the protected network.

(vi) Each proxy maintains detailed audit information by logging all traffic, each connection, and the duration of each connection. The audit log is an essential tool for discovering and terminating intruder attacks.

(vii) Each proxy module is a very small software package specifically designed for network security. Because of its relative simplicity, it is easier to check such modules for security flaws.

(viii) Each proxy is independent of other proxies on the bastion host. If there is a problem with the operation of any proxy, or if a future vulnerability is discovered, it can be uninstalled without affecting the operation of the other proxy applications. Also, if the user population requires support for a new service, the network administrator can easily install the required proxy on the bastion host.

(ix) A proxy generally performs no disk access other than to read its initial configuration file. This makes it difficult for an intruder to install Trojan horse sniffers or other dangerous files on the bastion host.

(x) Each proxy runs as a nonprivileged user in a private and secured directory on the bastion host.

Q.32. Explain various firewall configurations.

Or

Explain the different types of firewalls configuration. What are the differences among the three configuration ? *(R.G.P.V., June 2007)*

Ans. Fig. 4.12 shows three common firewall configurations.

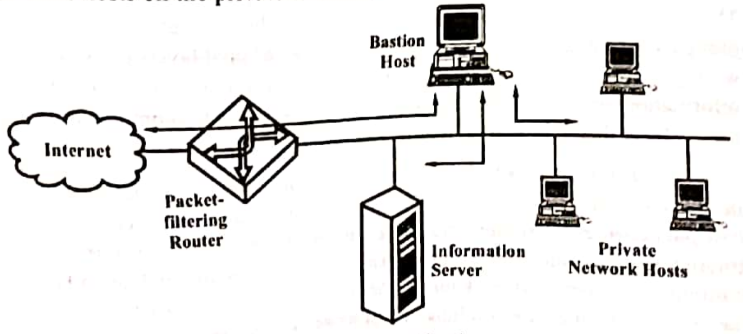
(i) ***Screened Host Firewall System (Single-homed Bastion Host)*** – This configuration is shown in fig. 4.12 (a). In this configuration, the firewall consists of two systems – a packet filtering router and a bastion host. Typically, the router is configured so that –

(a) For traffic from the Internet, only IP packets designed for the bastion host are allowed in.

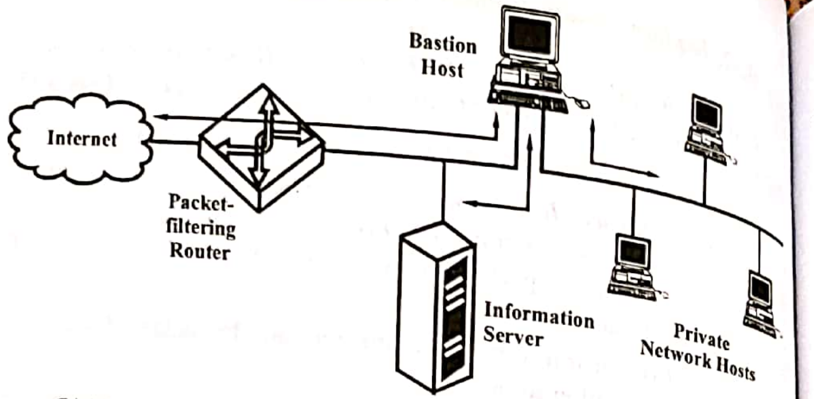
(b) For traffic from the internal network, only IP packets from the bastion host are allowed out.

The bastion host performs authentication and proxy functions. This configuration has greater security than simply a packet-filtering router or an application-level gateway alone, for two reasons. First this configuration implements both packet-level and application-level filtering, allowing for considerable flexibility in defining security scheme. Second, an intruder must generally penetrate two separate systems before the security of the internal network is compromised.

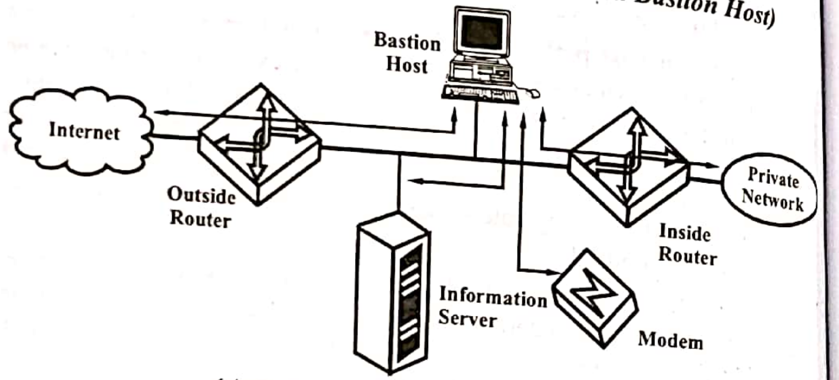
This configuration also affords flexibility in providing direct Internet access. For example, the internal network may include a public information server, such as a Web server, for which a high level of security is not required. In that case, the router can be configured to allow direct traffic between the information server and the Internet. If the packet filtering router is completely compromised, traffic could flow directly through the router between the Internet and other hosts on the private network.



(a) Screened Host Firewall System (Single-homed Bastion Host)



(b) Screened Host Firewall System (Dual-homed Bastion Host)



(c) Screened-subnet Firewall System

Fig. 4.12 Firewall Configurations

(ii) Screened Host Firewall System (Dual-homed Bastion Host) – This configuration is shown in fig. 4.12 (b). The configuration physically prevents such a security breach. The advantages of dual layers of security that were present in the previous configuration are present here as well. Again, an information server or other hosts can be allowed direct communication with the router if this is in accord with the security scheme.

(iii) Screened Subnet Firewall System – This configuration is shown in fig. 4.12 (c). This is the most secure configuration. In this configuration, two packet-filtering routers are used, one between the bastion host and the Internet and the other between the bastion host and the internal network. This configuration creates an isolated subnetwork, which may consist of simply the bastion host but may also include one or more information servers and modems for dial-in capability. Typically, both the Internet and the internal network have

access to hosts on the screened subnet, but traffic across the screened subnet is blocked. This configuration has following advantages –

(a) There are now three levels of defense to thwart intruders.

(b) The outside router advertises only the existence of the screened subnet to the Internet. Therefore, the internal network is invisible to the Internet.

(c) Similarly, the inside router advertises only the existence of the screened subnet to the internal network. Therefore, the systems on the inside network cannot construct direct routes to the Internet.

Q.33. How is screened host firewall, dual-homed bastion different from screened host firewall, single homed bastion ? ***(R.G.P.V., June 2014)***

Ans. Refer to Q.32.

Q.34. Write short note on personal firewalls.

Ans. Personal firewalls are very much needed for standalone machines connected to the Internet through various means like dial-ups, cable modems or DSL connections. Having a separate firewall computer to protect a single computer system can be too expensive and complex. A personal firewall is an application program running on a specific computer system. The firewall screens the incoming and outgoing traffic for the computer system and blocks the unwanted traffic from entering or leaving the system. The user could configure the personal firewall to accept traffic only from certain sites or not from specific sites and to generate logs of the past activities. The personal firewall can also be configured to function as a virus scanner so that any incoming data to the system will be first scanned for any potential virus infection.

Q.35. Explain intrusion detection system (IDS). ***(R.G.P.V., June 2011)***

Or

Write a short note on intrusion detection. ***(R.G.P.V., June 2015)***

Or

Write a short note on IDS. ***(R.G.P.V., June 2016)***

Or

Write a short note on intrusion detection system. ***(R.G.P.V., May 2019)***

Ans. An ***intrusion detection system (IDS)*** is a system used to detect unauthorized intrusions into computer systems and networks. Intrusion detection as a technology is not new, it has been used for generations to defend valuable resources. Kings, emperors, and nobles who had wealth used it in rather an interesting way. They built castles and palaces on tops of mountains and sharp cliffs with observation towers to provide them with a clear overview of the lands below where they could detect any attempted intrusion ahead of time to defend themselves.

Over the years, intrusion detection has been used by individuals and companies in a number of ways including erecting ways and fences around valuable resources with sentry boxes to watch the activities surrounding the premises of the resource. Individuals have used dogs, flood lights, electronic fences, and closed circuit television and other watchful gadgets to be able to detect intrusions.

As technology has developed, a new industry based on intrusion detection has sprung up. Security firms are cropping up everywhere to offer individual and property security–to be a watchful eye so that the property owner can sleep or take a vacation in peace. These new systems have been made to configure changes, compare user actions against known attack scenarios, and be able to predict changes in activities that indicate and can lead to suspicious activities.

Q.36. Discuss the concept of encrypted tunnels.

Or

Write a short note on encrypted tunnel. ***(R.G.P.V., June 2016)***

Ans. A tunnel is a point-to-point connection in which the actual communication occurs across a network. Suppose the only reason you have hooked into the Internet is to connect disconnected pieces of your own network to each other. Instead of the example in fig. 4.13, you might have bought dedicated links between G_1, G_2 and G_3 and trusted those links as part of your corporate network because you owned them. But it's likely to be cheaper to have the Gs pass data across the Internet. Your corporate data crossing over the Internet by configuring G_1, G_2 and G_3 with security information about each other. All information between them is cryptographically protected. The most common protocol for encrypted tunnels is IPsec.

The mechanics of the tunnel, from the IP point of view, is that when A sends a packet to C, A will launch it with an IP header that has source = A, distination = C. When G_1 sends it across the tunnel, it puts it into another envelope i.e., it adds an additional IP header, treating the inner header as data. The outer IP header will contain source = G_1 and destination = G_2. And all the contents will be encrypted and integrity protected, so it is safe to traverse the Internet. G_1, G_2 and G_3 use the Internet like some sort of insecure wire. You might want your users to be able to access the corporate network from across the Internet as well. Suppose B is some sort of workstation that can attach to the Internet in any location. To do this, B would create a tunnel with one of the Gs. This configuration is often referred to as VPN (Virtual Private Network).

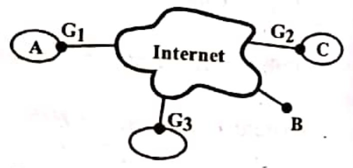


Fig. 4.13 Connecting a Private Network over a Public Internet

Q.37. Specify and explain classes of intruders. ***(R.G.P.V., June 2012)***

Ans. Three classes of intruders are as follows –

(i) ***Masquerader*** – An individual who does not have the authority to use the computer, but penetrates into a system to access a legitimate user's account.

(ii) ***Misfeasor*** – A legitimate user who accesses some applications, data or resources for which such access is not authorized, or who has access to some applications, data or resources but misuses his/her privileges.

(iii) ***Clandestine User*** – A user who seizes supervisory control of the system and tries this control to avoid auditing and access controls or to suppress audit collection.

Q.38. List and briefly define three classes of intruders. What is honey pot ? ***(R.G.P.V., June 2017)***

Ans. **Three Classes of Intruders** – Refer to Q.37.

Honey Pot – Refer to Q.12.

Q.39. Write the basic principle of intrusion detection system. ***(R.G.P.V., June 2006)***

Or

Explain benefits that can be provided by an intrusion detection. ***(R.G.P.V., June 2008, Dec. 2008)***

Or

What are three benefits that can be provided by an intrusion detection system ? ***(R.G.P.V., June 2012)***

Ans. Intrusion detection is learning of an attack either before or after its success. This interest is motivated by a number of considerations, including the following benefits –

(i) If an intrusion is detected quickly enough, the intruder can be identified and ejected from the system before any damage is done or any data are compromised. Even if the detection is not sufficiently timely to preempt the intruder, the sooner that the intrusion is detected, the less the amount of damage and the more quickly that recovery can be achieved.

(ii) Intrusion detection enables the collection of information about intrusion techniques that can be used to strengthen the intrusion prevention facility.

(iii) An effective intrusion detection system can serve as a deterrent, so acting to prevent intrusions.

Q.40. Define virus, intruders, worms. Also write the basic principle of intrusion detection system. ***(R.G.P.V., May 2018)***

Ans. **Virus** – Refer to Q.5.

Intruders – One of the two most popular threats to security is the intruder (the other is viruses), generally referred to as a *hacker* or *cracker*. An individual who gains, or attempts to gain, unauthorized access to a computer system or to gain unauthorized privileges on that system, is known as intruder.

Also refer to Q.37.

Worms – Refer to Q.5.

Basic Principle of Intrusion Detection System – Refer to Q.39.

Q.41. Why would leased line as a better approach than VPN ?

(R.G.P.V., June 2013)

Ans. There are some circumstances when VPN is not the best alternative. These are mission critical situation when you have to have a guaranteed bandwidth all day long. This cannot be achieved presently by an IP connection but can be achieved using leased lines. The second case is when the remote access user is using only local calls to access the remote access server. In this case, it is also more cost efficient to use dial-in or leased lines solutions.

Q.42. What are the two approaches to intrusion detection ?

Ans. Following two approaches has been identified for intrusion detection –

(i) Statistical Anomaly Detection – It involves the collection of data relating to the behaviour of legitimate users over a period of time. Then statistical tests are applied to observed behaviour to determine with a high level of confidence whether that behaviour is not legitimate user behaviour.

(a) Threshold Detection – This approach involves defining thresholds, independent of user, for the frequency of occurrence of various events.

(b) Profile Based – A profile of the activity of each user is developed and used to detect changes in the behaviour of individual accounts.

(ii) Rule-Based Detection – It involves an attempt to define a set of rules that can be used to decide that a given behaviour is that of an intruder.

(a) Anomaly Detection – Rules are developed to detect deviation from previous usage patterns.

(b) Penetration Identification – An expert system approach that searches for suspicious behaviour.

In terms of the types of attackers discussed earlier, statistical anomaly detection is effective against masqueraders, who are unlikely to mimic the behaviour patterns of the accounts they appropriate. On the other hand, such techniques may be unable to deal with misfeasors. For such attacks, rule-based approaches may be able to recognize events and sequences that, in context, reveal penetration. In practice, a system may exhibit a combination of both approaches to be effective against a broad range of attacks.

Q.43. What is an audit record ? What are its types ? Describe the fields of audit record.

Ans. An audit record is the fundamental tool for intrusion detection. Some record of ongoing activity by users must be maintained as input to an intrusion detection system.

There are two types of audit records –

(i) Native Audit Records – Virtually all multiuser operating systems include accounting software that collects information on user activity. The advantage of using this information is that no additional collection software is needed. The disadvantage is that the native audit records may not contain the needed information or may not contain it in a convenient form.

(ii) Detection-specific Audit Records – A collection facility can be implemented that generates audit records containing only that information required by the intrusion detection system. One advantage of such an approach is that it could be made vendor independent and ported to a variety of systems. The disadvantage is the extra overhead involved in having, in effect, two accounting packages running on a machine.

Each audit record contains the following fields –

(i) Subject – Initiators of actions. A subject is typically a terminal user but might also be a process acting on behalf of users or groups of users. All activity arises through commands issued by subjects.

(ii) Action – Operation performed by the subject on or with an object. For example, login, read, perform I/O, execute.

(iii) Object – Receptors of actions. Examples include files, programs, messages, records, terminals, printers and user or program created structures.

(iv) Exception Condition – Denotes which, if any, exception condition is raised on return.

(v) Resource-usage – A list of quantitative elements in which each element gives the amount used of some resource.

(vi) Time-stamp – Unique time-and-date stamp identifying when the action took place.

Q.44. Explain rule-based intrusion detection method.

Ans. Rule-based techniques detect intrusion by observing events in the system and applying a set of rules that lead to a decision regarding whether a given pattern of activity is or is not suspicious. In very general terms, we can characterize all approaches as focusing on either anomaly detection or penetration identification, although there is some overlap in these approaches.

Rule-based anomaly detection is similar in terms of its approach and strength to statistical anomaly detection. With this approach, historical audit records are analyzed to identify usage patterns and to generate automatically rules that describes those patterns. Rules may represent past behaviour patterns of users, programs, privileges, time slots, terminals and so on. Current behaviour is then observed and each transaction is matched against the set of rules to determine if it conforms to any historically observed pattern of behaviour. Rule-based anomaly detection does not require knowledge of security vulnerabilities within the system. For this approach to be effective, a large database of rules will be needed.

Rule-based penetration identification takes a very different approach to intrusion detection, one based on expert system technology. The key feature of such systems is the use of rules for identifying known penetrations or penetrations that would exploit known weaknesses. Rules can also be defined that identify suspicious behaviour, even when the behaviour is within the bounds of established patterns of usage. Typically the rules used in these systems are specific to the machine and operating system. Also, such rules are generated by "experts" rather than by means of an automated analysis of audit records. The normal procedure is to interview system administrators and security analysts to collect a suite of known penetration scenarios and key events that threaten the security of the target system. Thus, the strength of the approach depends on the skill of those involved in setting up the rules.

Q.45. Discuss intrusion detection exchange format.

Ans. To facilitate the development of distributed intrusion detection systems that can function across a wide range of platforms and environments, standards are needed to support interoperability. Such standards are the focus of the IETF Intrusion Detection Working Group. The purpose of the working group is to define data formats and exchange procedures for sharing information of interest to intrusion detection and response systems and to management systems that may need to interact with them. The outputs of this working group include the following –

(i) A requirements document, which describe the high-level functional requirements for communication between intrusion detection systems and requirements for communication between intrusion detection systems and with management systems, including the rationale for those requirements. Scenarios will be used to illustrate the requirements.

(ii) A common intrusion language specification, which describes data formats that satisfy the requirements.

(iii) A framework document, which identifies existing protocols best used for communication between intrusion detection systems and describes how the devised data formats relate to them.

Q.46. Explain statistical anomaly detection method for intrusion detection.

Ans. Statistical anomaly detection techniques fall into two broad categories – threshold detection and profile-based systems. Threshold detection involves counting the number of occurrences of a specific event type over an interval of time. If the count surpasses what is considered a reasonable number that one might expect to occur, then intrusion is assumed.

Threshold analysis, by itself, is a crude and ineffective detector of even moderately sophisticated attacks. Both the threshold and the time interval must be determined. Because of the variability across users, such thresholds are likely to generate either a lot of false positives or a lot of false negatives. However, simple threshold detectors may be useful in conjunction with more sophisticated techniques.

Profile-based anomaly detection focuses on characterizing the past behaviour of individual users or related groups of users and then detecting significant deviations. A profile may consist of a set of parameters, so that deviation on just a single parameter may not be sufficient in itself to signal an alert.

The foundation of this approach is an analysis of audit records. The audit records provide input to the intrusion detection function in two ways. First, the designer must decide on a number of quantitative metrics that can be used to measure user behaviour. An analysis of audit records over a period of time can be used to determine the activity profile of the average user. Thus, the audit records serve to define typical behaviour. Second, current audit records are the input used to detect intrusion. That is, the intrusion detection model analyzes incoming audit records to determine deviation from average behaviour.

Examples of metrics that are useful for profile-based intrusion detection are the following –

(i) Counter – A nonnegative integer that may be incremented but not decremented until it is reset by management action. Typically, a count of certain event types is kept over a particular period of time. Examples include the number of logins by a single user during an hour, the number of times a given command is executed during a single user session and the number of password failures during a minute.

(ii) Gauge – A nonnegative integer that may be incremented or decremented. Typically, a gauge is used to measure the current value of some entity. Examples include the number of logical connections assigned to a user application and the number of outgoing messages queued for a user process.

(iii) Interval Timer – The length of time between two related events. An example is the length of time between successive logins to an account.

(iv) Resource Utilization – Quantity of resources consumed during a specified period. Examples include the number of pages printed during a user session and total time consumed by a program execution.

The main advantage of the use of statistical profiles is that a prior knowledge of security flaws is not required. The detector program learns what is "normal behaviour" and then looks for deviations. The approach is not based on system dependent characteristics and vulnerabilities. Thus, it should be readily portable among a variety of systems.

Q.47. What metrices are useful for profile based intrusion detection? What are benefits that can be provided by an intrusion detection system?

(R.G.P.V., Dec. 2011)

Ans. Refer to Q.46 and Q.39.

Q.48. Explain the term base-rate fallacy. Explain statistical anomaly detection method for intrusion detection.

(R.G.P.V., June 2010)

Ans. **Base-rate Fallacy** – To be of practical use, an intrusion detection system should detect a substantial percentage of intrusions while keeping the false alarm rate at an acceptable level. If only a modest percentage of actual intrusions are detected, the system provides a false sense of security. On the other hand, if the system frequently triggers an alert when there is no intrusion (a false alarm) then either system managers will begin to ignore the alarms, or much time will be wasted analyzing the false alarms.

Unfortunately, because of the nature of the probabilities involved, it is very difficult to meet the standard of high rate of detections with a low rate of false alarms. In general, if the actual numbers of intrusions is low compared to the number of lagitimate users of a system, then the false alarm rate will be high unless the test is extermely discriminating. A study of existing intrusion detection systems, indicated that current systems have not overcome the problem of the base-rate fallacy.

Statistical Anomaly Detection Method – Refer to Q.46.

Q.49. Differentiate between the following –

(i) Statistical anomaly detection and rule based intrusion detection

(ii) Rule-based anomaly detection and rule based penetration identification.

(R.G.P.V., Dec. 2011)

Ans. (i) Refer to Q.46 and Q.44.

(ii) Refer to Q.44.

E-MAIL SECURITY – SERVICES SECURITY FOR E-MAIL ATTACKS THROUGH E-MAILS, PRIVACY AUTHENTICATION OF SOURCE MESSAGE, PRETTY GOOD PRIVACY (PGP), S-MIME

Q.50. Define e-mail security in detail. Why e-mail security is important?

(R.G.P.V., June 2016)

Ans. E-mail security defines multiple methods for keeping sensitive information in e-mail communication and accounts secure against unauthorized access, loss, or compromise. E-mail is a popular medium for the spread of malware, spam, and phishing attacks, using deceptive messages to entice recipients to divulge sensitive information, open attachments or click on hyperlinks that install malware on the victim's device. E-mail is also a common entry vector for attackers looking to gain a foothold in an enterprise network and breach valuable company data. E-mail security is necessary for both individual and business email accounts.

Importance of E-mail Security – There are several importance of e-mail security –

(i) Avoid Business Risks – These days there is so much at stake, so no wants to send unencrypted e-mails. Without encryption any stranger can have access to the information which is contained in your mail. Your competitors can use such information against you. Therefore to avoid business as well as other kinds of risks it is advisable that you should go in for e-mail encryption.

(ii) Protection of Confidential Information – E-mail encryption protects confidential information such as your credit card number, banking account number, social security number etc. In case your mail is not encrypted some wrong elements can make use of your personal information for their ulterior motives. Can you imagine that the messages which you sent can be read or even altered in transit? Even the username as well as password which you type can be stolen without much difficulty. So to avoid leakage on such vital information e-mail encryption is important.

(iii) To Nullify Message Replay Possibilities – You already know that the message you sent can be modified, but then there is one more thing which is possible with the messages you send. Messages can be saved, altered, and then re-sent later on. One can get an authentic message first, and then receive fake messages which appear to be official later on. The recipient cannot tell whether the e-mail message which has been sent to him is altered. In case the message was just deleted they will not even know that it had ever been sent.

(iv) Avoidance of Identity Theft – If any person gets hold of your username as well as password which you use to get to your e-mail servers, he or she can read the e-mails which you send and also send false e-mail messages on your behalf. This is referred to as identity theft and can be avoided if you go in for e-mail encryption.

(v) Repudiation of Messages Sent – Owing to the fact that it is easy to forge regular e-mail messages, you can never really prove that a so and so person has sent you a particular message. This connotes that even if a person actually sent you a particular message, he can deny sending it. This has serious implications as regards to making use of e-mail for the purpose of contracts, electronic commerce, business communications, etc.

(vi) Unprotected Backups – The messages which you send are stored on SMTP (Simple Mail Transfer Protocol) servers – outgoing mail servers. The backups of server disks encompass text copies of your messages. These backups can be present for years. So anyone who has access to the backup files can read your messages and use the information to your disadvantage even while you are thinking that you have deleted the message.

Q.51. Write a short note on e-mail security. ***(R.G.P.V., May 2019)***

Ans. Refer to Q.50.

Q.52. What security protocols are used to protect e-mail ? ***(R.G.P.V., June 2011)***

Ans. The three main e-mail security protocols are privacy enhanced mail (PEM), pretty good privacy (PGP), and secure MIME (S/MIME).

(i) PEM – The privacy enhanced mail (PEM) is an e-mail security standard adopted by the Internet Architecture Board (IAB) to offer secure electronic mail over the Internet. It was initially designed by the Internet Research Task Force (IRTF) and Privacy Security Research Group (PSRG). They then handed over the PEM standard to the Internet Engineering Task Force (IETF) PEM working group. The PEM protocols provide for encryption, authentication, message integrity, and key management.

PEM is an inclusive standard. The PEM procedures and protocols are intended to be compatible with a wide range of key management approaches, including both symmetric and public key schemes to encrypt data-encrypting keys. Symmetric cryptography is used for message text encryption. Cryptographic hash algorithms are used for message integrity. Other documents support key-management mechanisms using public-key certificates; algorithms, modes, and associated identifiers; and paper and electronic format details and procedures for the key-management infrastructure to support these services. PEM provides three privacy enhancement services – message integrity, authentication and confidentiality.

(ii) PGP – Pretty good privacy (PGP) is another secure mail protocol. It is a freeware electronic-mail security program, originally designed by Phil Zimmermann. PGP was invented to provide e-mail with privacy, integrity and authentication. PGP can be used to create a secure e-mail message or to store a file securely for future retrieval. It uses IDEA for data encryption, RSA for key management and digital signatures, and MD5 as a one-way hash function. It is effective, easy to use, and free. PGP is based on the public-key method.

PGP is not just for mail. It performs encryption and integrity protection on files. Mail is not treated any differently from ordinary files. Some wishing to send a secure mail message could first transform the file to be mailed using PGP, and then mail the transformed file using a traditional mailer. Similarly, if one were to receive a PGP-encrypted mail message, one could treat the received message as a file and feed it to PGP to process. This is a bit convenient. So PGP source code comes with modifications for a number of common mail systems, enabling people to integrate PGP into their mail systems.

(iii) S/MIME – Another security service designed for electronic mail is Secure/Multipurpose Internet Mail Extension (S/MIME). S/MIME is a security enhancement to the MIME Internet e-mail format standard, based on technology from RSA data security. Although both PGP and S/MIME are on an IETF standards track, it appears likely that S/MIME will emerge as the industry standard for commercial and organizational use, while PGP will remain the choice for personal e-mail security for many users. S/MIME adds some new content types to include security services to the MIME. All of these new types include the parameter "application/pkcs7-mime", in which "pkcs" defines public key cryptography specification.

Q.53. Describe the kind of security services for electronic mail.

Ans. Most electronic mail systems do not provide most of these features. Even those designed specifically for security often only provide for some of these features.

(i) Privacy – The ability to keep anyone but the intended recipient from reading the message.

(ii) Authentication – Reassurance to the recipient of the identity of the sender.

(iii) Integrity – Reassurance to the recipient that the message has not been altered since it was transmitted by the sender.

(iv) Non-repudiation – The ability of the recipient to prove to a third party that the sender really did send the message. This feature is also sometimes called ***third party authentication.*** The term *non-repudiation* means that the sender cannot later deny sending the message.

(v) Proof of Submission – Verification given to the sender that the message was handed to the mail delivery system. With certified postal mail you just receive proof that you sent something to a particular address on a particular date, but with electronic mail it is possible to have the mail system verify acceptance of the contents of a particular message, perhaps by signing the message digest of the contents of the message.

(vi) Proof of Delivery – Verification that the recipient received the message. Postal mail has a similar feature (return receipt requested), but again it only verifies that something was delivered on a particular date to the recipient. With electronic mail it is possible to verify the contents of a particular message.

(vii) Message Flow Confidentiality – An extension of privacy such that Carol not only cannot know the content of the message Alice sent Bob, but cannot even determine whether Alice sent Bob a message.

(viii) Anonymity – The ability to send a message so that the recipient can't find out the identity of the sender.

(ix) Containment – The ability of the network to keep certain security levels of information from leaking out of a particular region.

(x) Audit – The ability of the network to record events that might have some security relevance, such as that Alice sent a message to Bob on a particular date.

(xi) Accounting – The ability of the mail system to maintain system usage statistics. In addition to providing clues for system resource management, this information allows the mail system to charge its clients according to their usage. For example, the system might charge by number of messages sent, as long as the system itself authenticates the source of each message to ensure that the proper party is billed.

(xii) Self Destruct – It is an option allowing a sender to specify that a message should be destroyed after delivery to the recipient. This allows Alice to send a message to Bob that Bob cannot forward or store. The mail system will decrypt and display the message, but then delete it. This can be implemented by marking the message as a *self destruct* message, and having the mail program at the destination cooperate by deleting the message immediately after displaying it.

(xiii) Message Sequence Integrity – Reassurance that an entire sequence of messages arrived in the order transmitted, without any loss.

Q.54. Draw generic transmission diagram in PGP and explain in brief.
(R.G.P.V., June 2012)

Ans. Fig. 4.14 shows the generic transmission diagram in PGP. On transmission, if necessary, a signature is generated using a hash code of the

uncompressed plaintext. Then the plaintext, as well as signature if present, is compressed. Next, if confidentiality is needed, the block is encrypted and prepended with the public-key encrypted symmetric encryption key. At last, the entire block is converted to radix-64 format.

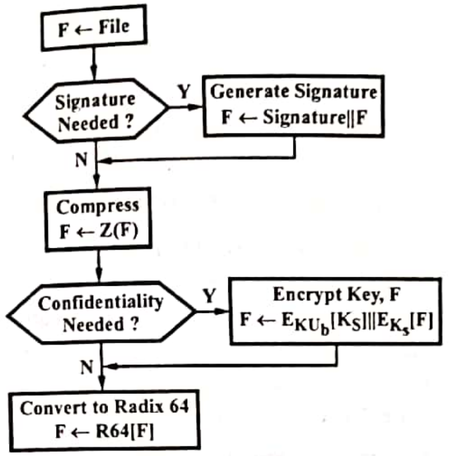

F ← File
Signature Needed ?
Y
Generate Signature F ← Signature||F
N
Compress F ← Z(F)
Confidentiality Needed ?
Y
Encrypt Key, F F ← $E_{KU_b}[K_S]||E_{K_s}[F]$
N
Convert to Radix 64 F ← R64[F]

Fig. 4.14 Generic Transmission Diagram

Q.55. Explain the steps involved in key generation in PGP.
(R.G.P.V., Dec. 2011)

Ans. The steps involved in key generation in PGP are as follows –

(i) The sender generates a message. For this message, a random 128-bit number is used as a session key.

(ii) The encryption of the message is done using CAST-128 with the session key.

(iii) The session key is encrypted with RSA, using the recipient's public key, and is prepended to the message.

(iv) The receiver uses RSA with its private key to decrypt and recover the session key.

(v) The session key is used to decrypt the message.

Q.56. Why does PGP generate a signature before applying compression ?
(R.G.P.V., June 2017)

Ans. The signature is generated before compression for two reasons –

(i) It is preferable to sign an uncompressed message. So that one can store only the uncompressed message together with the signature for future verification. If one signed a compressed document, then it would be necessary either to store a compressed version of the message for later verification or recompress the message when verification is required.

(ii) Even when one were willing to generate dynamically a recompressed message for verification, PGP's compression algorithm shows a problem. The algorithm is not deterministic. Various implementations of the algorithm get distinct tradeoffs in running speed versus compression ratio and, as a result, generate distinct compressed forms. Although, these distinct compression algorithms are interoperable because any version of the algorithm may correctly decompress the output of any other version. Applying the hash function and signature after compression would constrain all PGP implementations to the same version of the compression algorithm.

Q.57. What are the capabilities of S/MIME ?

Ans. S/MIME is capable of providing privacy in e-mails by enveloped messages, and authenticity by signed messages. Signed messages also ensure non-repudiation of sender. Message integrity can be verified using message digest. The use of signed receipts is claimed to provide non-repudiation of the recipient, though this claim is not beyond question. However, "strong fairness" as mentioned in section 2 is yet to achieve.

For instance, Alice may want to send a message to Bob, which Bob needs to verify for authenticity. But Alice wants to make sure that Bob will not be able to prove to a third person that the message originated from Alice.

S/MIME supports both 'opaque' and 'clear' (multipart) signing format. In case of clear signed format, only the digital signature is encoded using radix-64. Consequently, the recipients without S/MIME capability can still view the message, although they cannot verify the signature.

Q.58. What are the limitations of S/MIME ?

Ans. Limitations of S/MIME are as follows –

(i) Non-repudiation of recipient is claimed to have achieved by the use of signed receipt.

(ii) Multi-recipient message support in S/MIME is not efficient enough.

(iii) Partial content signature is not supported by S/MIME.

(iv) E-mail header protection provided by S/MIME is not sufficient.

(v) Possible use of bogus name for sender is not prevented in S/MIME, due to the fact that class-1 certificates do not contain validated names, and e-mail clients allow any name for sender while sending an e-mail message.

(vi) E-mail storage in encrypted form with the original encryption key is a design flaw of all current S/MIME client implementations.

IP SECURITY – OVERVIEW OF IPSEC, IP & IP VERSION 6 AUTHENTICATION, ENCAPSULATION SECURITY PAYLOAD (ESP), INTERNET KEY EXCHANGE (IKE)

Q.59. Write a short note on IP security.
(R.G.P.V., Dec. 2004, May/June 2006, 2007)

Or

Write an essay on IP security. **(R.G.P.V., Dec. 2005)**

Or

What do you mean by IP security ? **(R.G.P.V., June 2009)**

Or

Explain IP security. **(R.G.P.V., Dec. 2006, June 2015)**

Or

Discuss IP security in detail. **(R.G.P.V., June 2016)**

Ans. In 1994, the Internet Architecture Board (IAB) issued a report entitled ***Security in Internet Architecture.*** The report stated the general consensus that the Internet needs more and better security, and it identified key areas for security mechanisms. Among these were the need to secure the network infrastructure from unauthorized monitoring and control of network traffic and the need to secure end-user-to-end-user traffic using authentication and encryption mechanisms.

These concerns are fully justified. As confirmation, the 2001 annual report from the Computer Emergency Response Team (CERT) lists over 52,000 reported security incidents. The more severe types of attacks are IP spoofing, in which intruders create packets with false IP addresses and exploit applications that use authentication based on IP; and various forms of eavesdropping and packet sniffing, in which attackers read transmitted information, including logon information and database contents.

In response to these issues, the IAB included authentication and encryption as necessary security features in the next-generation IP, which has been issued as IPv6. These security capabilities were designed to be usable both with the current IPv4 and the future IPv6. It means that vendors can begin offering these features now, and many vendors do now have some IPSec capability in their products.

Q.60. What are the various applications of IPSec ?

Ans. IPSec provides the capability of secure communication across a LAN, across private and public WANs, and across the Internet. Following are the examples of its uses –

(i) ***Secure Remote Access Over the Internet*** – An end user whose system is equipped with IP security protocols can make a local call to an Internet service provider (ISP) and gain secure access to a company network. This reduces the cost of toll charges for traveling employees and telecommuters.

(ii) Secure Branch Office Connectivity Over the Internet – A company can build a secure virtual private network over the Internet or over a public WAN. This enables a business to rely heavily on the Internet and reduce its need for private networks, saving costs and network management overhead.

(iii) Enhancing Electronic Commerce Security – Even though some Web and electronic commerce applications have built-in security protocols, the use of IPSec enhances that security.

(iv) Establishing Extranet and Intranet Connectivity with Partners – IPSec can be used to secure communication with other organizations, ensuring authentication and confidentiality and providing a key exchange mechanism.

Q.61. What are the benefits of IPSec ? ***(R.G.P.V., Dec. 2006)***

Ans. The various benefits of IPSec are given as follows –

(i) When IPSec is implemented in a firewall or router, it provides strong security that can be applied to all traffic crossing the perimeter.

(ii) IPSec in a firewall is resistant to bypass if all traffic from the outside must use IP, and firewall is the only means of entrance from the Internet into the organization.

(iii) IPSec is below the transport layer (TCP, UDP) and so is transparent to applications. There is no need to change software on a user or server system when IPSec is implemented in the firewall or router. Even if IPSec is implemented in end systems, upper-layer software, including applications, is not affected.

(iv) IPSec can be transparent to end users. There is no need to train users on security mechanisms, issue keying material on a per-user basis, or revoke keying material when users leave the organization.

(v) IPSec can provide security for individual users if needed. This is usual for offsite workers and for setting up a secure virtual sub-network within an organization for sensitive applications.

Q.62. What comprises the basic IPSec architecture ? Describe briefly IP security documents.

Ans. IPSec is being developed by the Internet Engineering Task Force (IETF) IPSec Working Group. The full set of specifications for IPSec is not finished in writing but they are nearing completion and the basic RFC's are complete.

IPSec specification has become quite complex. To get a feel for the overall architecture we have to through an insight into the following –

(i) IPSec documents (ii) IPSec services (iii) Security associations.

IPSec Documents – The IPSec specification consists of numerous documents. The most important of these are RFCs 2401, 2402, 2406 and 2408.

- **RFC 2401** – An overview of security architecture.
- **RFC 2402** – Description of packet authentication extension to IPv4 and IPv6.
- **RFC 2406** – Description of a packet encryption extension to IPv4 and IPv6.
- **RFC 2408** – Specification of key management capabilities.

In addition to these four RFC's a number of additions drafts have been published by IPSec Protocol Working Group and are divided into seven categories as (shown in fig. 4.15) follows –

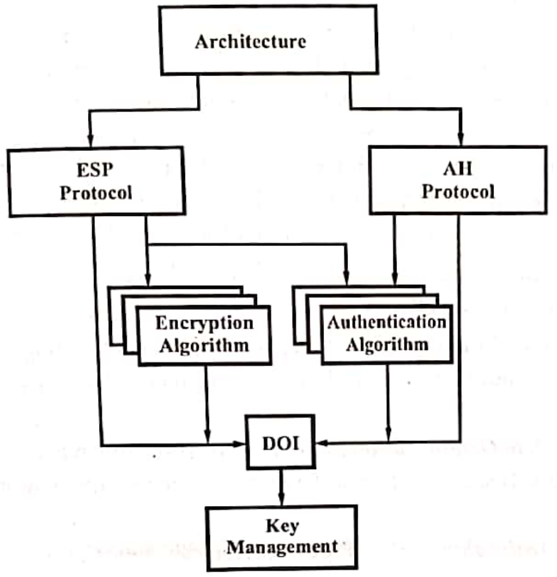


Fig. 4.15 IPSec Documents Overview

(i) Architecture – Covers the general concepts, security requirements, definitions and mechanisms defining IPSec technology.

(ii) Encapsulating Security Payload (ESP) – Covers the packet format and general issues related to the use of the ESP for packet encryption and optionally, authentication.

(iii) Authentication Header (AH) – Covers the packet format and general issues related to the use of AH for packet authentication.

(iv) Encryption Algorithm – A set of documents that describes how various encryption algorithms are used for ESP.

(v) Authentication Algorithm – A set of documents that describes how various authentication algorithms are used for AH and for authentication option of ESP.

(vi) Key Management – Documents that describe key management schemes.

(vii) Domain of Interpretation (DOI) – Contains values needed for the other documents to relate to each other. These include identifiers for approved encryption and authentication algorithms as well as operational parameters such as key lifetime.

Q.63. Discuss about the IP and IP version 6.

Ans. IP security, or IPsec, is a framework of open standards developed by the Internet Engineering Task Force (IETF) that provide security for transmission of sensitive information over unprotected networks such as the Internet. IPsec acts at the network layer, protecting and authenticating IP packets between participating IPsec devices (peers), such as Cisco routers. IPsec provides the following optional network security services. In general, local security policy will dictate the use of one or more of these services –

(i) Data Confidentiality – The IPsec sender can encrypt packets before sending them across a network.

(ii) Data Integrity – The IPsec receiver can authenticate packets sent by the IPsec sender to ensure that the data has not been altered during transmission.

(iii) Data Origin Authentication – The IPsec receiver can authenticate the source of the IPsec packets sent. This service depends upon the data integrity service.

(iv) Antireplay – The IPsec receiver can detect and reject replayed packets.

With IPsec, data can be sent across a public network without observation, modification, or spoofing. IPsec functionality is similar in both IPv6 and IPv4; however, site-to-site tunnel mode only is supported in IPv6.

In IPv6, IPsec is implemented using the AH authentication header and the ESP extension header. The authentication header provides integrity and authentication of the source. It also provides optional protection against replayed packets. The authentication header protects the integrity of most of the IP header fields and authenticates the source through a signature-based algorithm. The ESP header provides confidentiality, authentication of the source, connectionless integrity of the inner packet, antireplay, and limited traffic flow confidentiality.

Q.64. What services are provided by IPSec ? ***(R.G.P.V., June 2012)***

Ans. IPSec provides security services at the IP layer by enabling a system to select required security protocols, determine the algorithms to use for the services, and put in place any cryptographic keys required to provide the requested services. IPSec uses two protocols to provide security – first an authentication protocol, designated by the header of the protocol, Authentication Header (AH), and second a combined encryption/authentication protocol designated by the format of the packet for that protocol, Encapsulating Security Payload (ESP). The services are as follows –

(i) Access control

(ii) Connectionless integrity

(iii) Data origin authentication

(iv) Rejection of replayed packets (a form of partial sequence integrity)

(v) Confidentiality (encryption)

(vi) Limited traffic flow confidentiality.

Table 4.2 IPSec Services

S.No.	*Services*	*AH*	*ESP (encryption only)*	*ESP (encryption plus authentication)*
(i)	Access control	√	√	√
(ii)	Connectionless integrity	√		√
(iii)	Data origin authentication	√		√
(iv)	Rejection of replayed packets	√	√	√
(v)	Confidentiality		√	√
(vi)	Limited traffic flow confidentiality		√	√

Table 4.2 shows which services are provided by the AH and ESP protocols. For ESP, there are two cases – with and without the authentication option. Both AH and ESP are vehicles for access control, based on the distribution of cryptographic keys and the management of traffic flows relative to these security protocols.

Q.65. What is security association ?

Ans. Security association (SA) is a concept that appears in both the authentication and confidentiality mechanisms for IP. An association is a one-way relationship between a sender and a receiver that affords security services to the traffic carried on it. If a peer relationship is needed, for two way secure exchange, then two security associations are required. Security services are afforded to an SA for the use of AH or ESP, but not both.

Q.66. Explain encapsulating security payload (ESP) protocol.

Ans. The Encapsulating Security Payload protocol provides confidentiality services, as well as confidentiality of message contents and limited traffic flow confidentiality. As an optional feature, ESP can also provide the same authentication services as AH.

Fig. 4.16 shows the format of an ESP packet. It contains the following fields –

(i) Security Parameters Index (32 bits) – Identifies a security association.

(ii) Sequence Number (32 bits) – A monotonically increasing counter value. This provides an anti-replay function, as discussed for AH.

(iii) Payload Data (Variable) – This is a transport-level segment (transport mode) or IP packet (tunnel mode) that is protected by encryption.

(iv) Padding (0-255 bytes) – The purpose of this field is to expand the plaintext for required length.

(v) Pad Length (8 bits) – Indicates the number of pad bytes immediately preceding this field.

(vi) Next Header (8 bits) – Identifies the type of data contained in the payload data field by identifying the first header in that payload (for example, an extension header in IPv6, or an upper-layer protocol such as TCP).

(vii) Authentication Data (Variable) – A variable-length field (must be an integral number of 32-bit words) that contains the Integrity Check Value computed over the ESP packet minus the Authentication Data field.

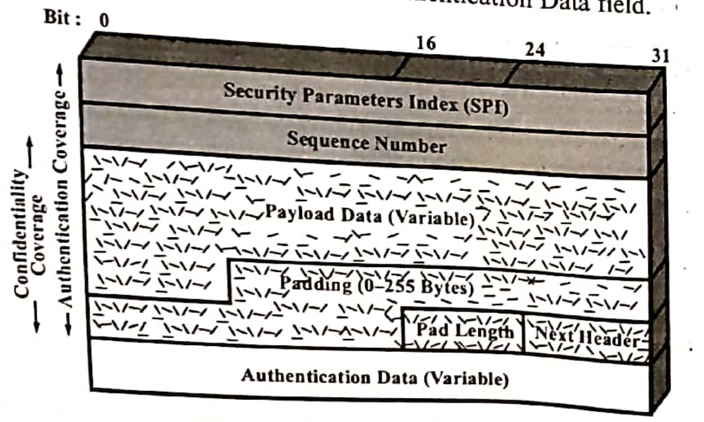


Fig. 4.16 IPSec ESP Format

Q.67. What parameters identify an SA and what parameters characterize the nature of a particular SA ?

Ans. A security association is uniquely identified by three parameters –

(i) Security Protocol Identifier – This indicates whether the association is an AH or ESP security association.

(ii) Security Parameters Index (SPI) – A bit string is assigned to this SA and having local significance only. The SPI is carried in AH and ESP headers to enable the receiving system to select the SA under which a received packet will be processed.

(iii) IP Destination Address – Currently, only unicast addresses are allowed. This is the address of the destination endpoint of the SA, which may be an end user system or a network system such as a firewall or router.

Hence, in any IP packet (an IPv4 datagram or an IPv6 packet), the security association is uniquely identified by the destination address in the IPv4 or IPv6 header and the SPI in the enclosed extension header (AH or ESP).

A security association is normally defined by the following parameters –

(i) AH Information – Authentication algorithm, keys, key lifetimes, and related parameters being used with AH (required for AH implementations).

(ii) ESP Information – Encryption and authentication algorithm, keys, initialization values, key lifetimes, and related parameters being used with ESP (required for ESP implementations).

(iii) Sequence Number Counter – A 32-bit value used to generate the sequence number field in AH or ESP headers (required for all implementations).

(iv) Sequence Counter Overflow – A flag indicating whether overflow of the Sequence Number Counter should generate an auditable event and prevent further transmission of packets on this SA (required for all implementations).

(v) Anti-replay Window – Used to determine whether an inbound AH or ESP packet is a replay (required for all implementations).

(vi) Lifetime of this Security Association – A time interval or byte count after which an SA must be replaced with a new SA and new SPI or terminated, plus an indication of which of these actions should occur (required for all implementations).

(vii) Path MTU – Any observed path maximum transmission unit (maximum size of a packet that can be transmitted without fragmentation) and aging variables (required for all implementations).

(viii) IPSec Protocol Mode – Tunnel, transport, or wildcard (required for all implementations).

The key mechanism that is used to distribute keys is coupled to the authentication and privacy mechanisms only by way of the Security Parameters Index. Hence, authentication and privacy have been specified independent of any specific key management mechanism.

Q.68. Give a overview of transport mode and tunnel mode.
(R.G.P.V., June 2015)

Or

IP sec can be used in two modes. What are they ? (R.G.P.V., June 2011)

Ans. The two modes of use supported by AH and ESP are transport and tunnel mode –

(i) Transport Mode – This mode provides protection primarily for upper-layer protocols. That is, transport mode protection extends as to the payload of an IP packet. For example, a TCP or UDP segment or an ICMP packet, all of which operate directly above IP in a host protocol stack. Transport mode typically used for end-to-end communication between two hosts (For example, a client and a server, or two workstations). When a host runs AH or ESP over IPv4, the payload is the data that normally follow the IP header. For IPv6, the payload is the data that normally follow both the IP header and any IPv6 extensions headers that are present, with the possible exception of the destination options header, which may be included in the protection.

ESP in transport mode encrypts and optionally authenticates the IP payload but not the IP header. AH in transport mode authenticates the IP payload and selected portions of the IP header.

(ii) Tunnel Mode – This mode provides protection to the entire IP packet. To achieve this, after the AH or ESP fields are added to the IP packet, the entire packet plus security fields is treated as the payload of new "outer" IP packet with a new outer IP header. The entire original, or inner, packet travels through a "tunnel" from one point of an IP network to another. No routers along the way are able to examine the inner IP header. Because the original packet is encapsulated, the new, larger packet may have totally different source and destination addresses, adding to the security. This mode is used when one or both ends of an SA is a security gateway, such as a firewall or router that implements IPSec. With this mode, a number of hosts on networks behind firewalls may engage in secure communications without implementing IPSec.The unprotected packets generated by such hosts are tunneled through external networks by tunnel mode SAs set up by the IPSec software in the firewall or secure router at the boundary of the local network.

ESP in tunnel mode encrypts and optionally authenticates the entire inner IP packet, including the inner IP header. AH in tunnel mode authenticates the entire inner IP packet and selected portions of the outer IP header.

Table 4.3 summarizes the transport and tunnel modes functionality.

Table 4.3 Tunnel Mode and Transport Mode Functionality

	Transport Mode SA	***Tunnel Mode SA***
AH	Authenticates IP payload and selected portions of IP header and IPv6 extension headers.	Authenticates entire inner IP packet (inner header plus IP payload) plus selected portions of outer IP header and outer IPv6 extension headers.
ESP	Encrypts IP payload and any IPv6 extension headers following the ESP headers.	Encrypts inner IP packet.
ESP with Authentication	Encrypts IP payload and any IPv6 extension headers following the ESP header. Authenticates IP payload but not IP header.	Encrypts inner IP packet. Authenticates inner IP packet.

Q.69. What is the difference between transport mode and tunnel mode ?
(R.G.P.V., June 2012, 2017)

Ans. Refer to Q.68.

Q.70. Discuss how ESP operates in transport and tunnel modes.
(R.G.P.V., June 2010)

Ans. **ESP Transport Mode** – ESP transport mode is used to encrypt and optionally authenticate the data carried by IP (for example, a TCP segment), as shown in fig. 4.17. For this mode using IPv4, the ESP header is inserted into the IP packet immediately prior to the transport-layer header (for example, TCP, UDP, ICMP) and an ESP trailer (Padding, Pad Length, and Next Header fields) is placed after the IP packet. If authentication is selected, the ESP Authentication Data field is added after the ESP trailer. The entire transport-level segment plus the ESP trailer are encrypted. Authentication covers all of the ciphertext plus the ESP header.

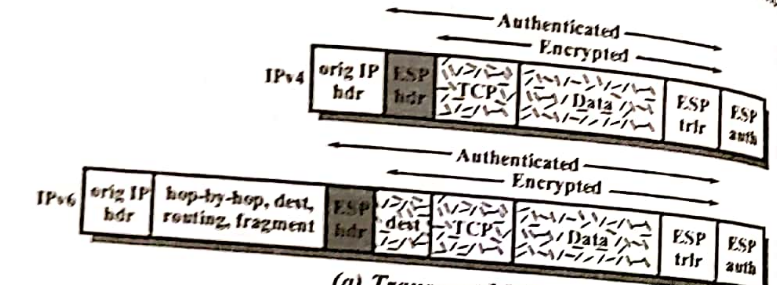
Transport mode operation may be summarized as follows –

(i) At the source, the block of data consisting of the ESP trailer plus the entire transport-layer segment is encrypted and the plaintext of this block is replaced with its ciphertext to form the IP packet for transmission. Authentication is added if this option is selected.

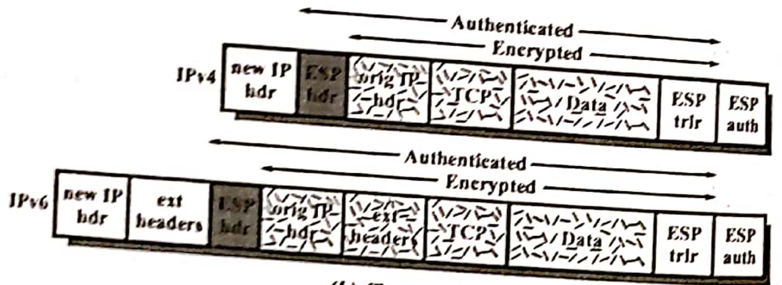
(ii) The packet is then routed to the destination. Each intermediate router needs to examine and process the IP header plus any plaintext IP extension headers but does not need to examine the ciphertext.

(iii) The destination node examines and processes the IP header plus any plaintext IP extension headers. Then, on the basis of the SPI in the ESP header, the destination node decrypts the remainder of the packet to recover the plaintext transport-layer segment.

Transport mode operation provides confidentiality for any application that uses it, thus avoiding the need to implement confidentiality in every individual application. This mode of operation is efficient, adding little to the total length of the IP packet. One limitation to this mode is that it is possible to do traffic analysis on the transmitted packets.



(a) Transport Mode



(b) Tunnel Mode

Fig. 4.17 Scope of ESP Encryption and Authentication

ESP Tunnel Mode – Tunnel mode ESP is used to encrypt an entire IP packet (fig. 4.17). For this mode, the ESP header is prefixed to the packet and then the packet plus the ESP trailer is encrypted. This method can be used to counter traffic analysis.

Because the IP header contains the destination address and possibly source routing directives and hop-by-hop option information, it is not possible to transmit the encrypted IP packet prefixed by the ESP header. Intermediate

routers would be unable to process such a packet. Therefore, it is necessary to encapsulate the entire block with a new IP header that will contain sufficient information for routine but not for traffic analysis.

Whereas the transport mode is suitable for protecting connections between hosts that support the ESP feature, the tunnel mode is useful in a configuration that includes a firewall or other sort of security gateway that protects a trusted network from external networks. In this latter case, encryption occurs only between an external host and the security gateway or between two security gateways. This relieves hosts on the internal network of the processing burden of encryption and simplifies the key distribution task by reducing the number of needed keys. Further, it thwarts traffic analysis based on ultimate destination.

Consider a case in which an external host wishes to communicate with a host on an internal network protected by a firewall, and in which ESP is implemented in the external host and the firewalls. The following steps are required for transfer of a transport-layer segment from the external host to the internal host –

(i) The source prepares an inner IP packet with a destination address of the target internal host. This packet is prefixed by an ESP header, then the packet and ESP trailer are encrypted and authentication data may be added. The resulting block is encapsulated with a new IP header whose destination address is the firewall, this forms the outer IP packet.

(ii) The outer packet is routed to the destination firewall. Each intermediate router needs to examine and process the outer IP header plus any outer IP extension headers but does not need to examine the ciphertext.

(iii) The destination firewall examines and processes the outer IP header plus any outer IP extension headers. Then, on the basis of the SPI in the ESP header, the destination node decrypts the remainder of the packet to recover the plaintext inner IP packet. This packet is then transmitted in the internal network.

(iv) The inner packet is routed through zero or more routers in the internal network to the destination host.

Q.71. How IPSec can be used to create VPN ? (R.G.P.V., Dec. 2007)

Ans. Fig. 4.18 shows the way in which the IPSec ESP service can be used. It shows how tunnel mode operation can be used to set up a virtual private network. In this example an organization has four private networks interconnected across the Internet. Hosts on the internal networks use the Internet for transport of data but do not interact with other Internet-based hosts. By terminating the tunnels at the security gateway to each internal network, the configuration allows the hosts to avoid implementing the security capability.

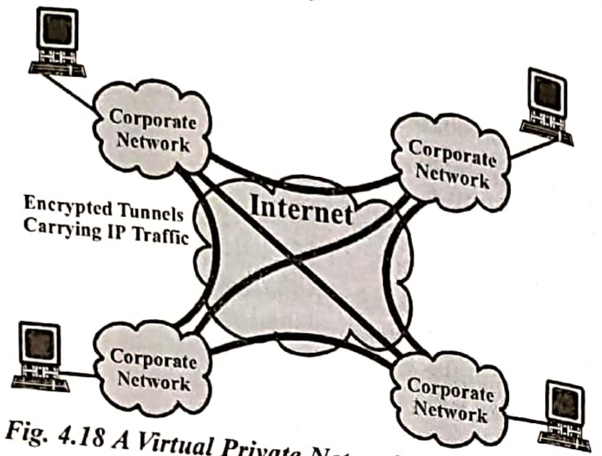


Fig. 4.18 A Virtual Private Network via Tunnel Mode

Q.72. What are the roles of the Oakley key determination protocol and ISAKMP in IPSec ?

Ans. The key management portion of the IPSec involves the determination and distribution of secret keys. A typical requirement is four keys for communication between two applications i.e., transmit and receive pairs for both AH and ESP. The IPSec architecture document mandates support for two types of key management –

(i) Automated – An automated system enables the on-demand creation of keys for SAs and facilitates the use of keys in a large distributed system with an evolving configuration.

(ii) Manual – A system administrator manually configures each system with its own keys and with the keys of other communicating systems. This is practical for small, relatively static environments.

The default automated key management protocol for IPSec is referred to as ISAKMP/Oakley and consists of the following elements –

(i) Oakley Key Determination Protocol – Oakley is a key exchange protocol based on the Diffie-Hellman key exchange algorithm but providing added security. Oakley is generic in that it does not dictate specific formats.

(ii) Internet Security Association and Key Management Protocol (ISAKMP) – This protocol provides a framework for Internet key management and provides the specific protocol support, including formats, for negotiation of security attributes.

ISAKMP by itself does not dictate a specific key exchange algorithm. Rather, it consists of a set of message types that enable the use of a variety of key exchange algorithms. Oakley is the specific key exchange algorithm mandated for use with the initial version of ISAKMP.

Q.73. Write short note on Internet key exchange (IKE) protocol.
Or
Explain the Internet key exchange protocol. ***(R.G.P.V., May 2019)***

Ans. The IKE is a protocol designed to create both inbound and outbound Security Association (SA). When a peer needs to send an IP packet, it consults the Security Policy Database (SPDB) to see if there is an SA for that type of traffic. If there is no SA, IKE is called to establish one. The main goal of IKE is to establish an SA between two parties that wish to communicate securely, using IPSec. IKE is an application-layer protocol using the connectionless UDP protocol on port 500. IKE is a complex protocol based on three other protocols. The *Internet Security Association and Key Management Protocol (ISAKMP)* is a protocol designed by the National Security Agency (NSA) that actually implements the exchanges defined in IKE. It defines several packets, protocols and parameters that allow the IKE exchanges to take place in standardized, formatted message to create SAs. The *Oakley* protocol was developed by Hilarie Orman. It is a key creation protocol based on the Diffie-Hellman key-exchange method, but with some improvements. Oakley is a free-formatted protocol in the sense that it does not define the format of the message to be exchanged. *SKEME*, designed by Hugo Krawcyzk, is another protocol for key exchange. It uses public-key encryption for entity authentication in a key-exchange protocol.

IKE is comprised of two phases. In the first phase, an "IKE SA" is established. This creates a secure channel upon which the communicating parties can then establish multiple "IPSec SA" instances over time. Setting up an IKE SA is analogous to setting up a session in the SSL protocol, while setting up in IPSec SA is analogous to setting up an SSL connection. It is good security practice to periodically change cryptographic keys used by two communicating parties. In phase 1, longer term keys are derived. Phase 1 occurs rarely and is more computationally intensive compared to phase 2. In phase 2, shorter terms keys are derived for use between two parties. This key is a function of the long term keys computed in phase 1 together with nonces exchanged in phase 2.

WEB SECURITY – SSL/TLS, BASIC PROTOCOLS OF SECURITY, ENCODING – SECURE ELECTRONIC TRANSACTION (SET)

Q.74. Write a short note on Web security. ***(R.G.P.V., Dec. 2004)***

Ans. The World Wide Web (WWW) is fundamentally a client/server application running over the Internet and TCP/IP intranets. But web presents new challenges not generally appreciated in the context of computer and network security which are given as follows –

(i) The Internet is two way. Unlike traditional publishing environments, even electronic publishing systems involving teletext, voice response, or fax-back, the Web is vulnerable to attacks on the Web servers over the Internet.

(ii) The Web is increasingly serving as a highly visible outlet for corporate and product information and as the platform for business transactions. Reputations can be damaged and money can be lost if the Web servers are subverted.

(iii) Although Web browsers are very easy to use, Web servers are relatively easy to configure and manage, and Web content is increasingly easy to develop, the underlying software is extraordinarily complex. This complex software may hide many potential security flaws. This short history of the web is filled with examples of new and upgraded systems, properly installed, that are vulnerable to a variety of security attacks.

(iv) A Web server can be exploited as a launching pad into the corporation's or agency's entire computer complex. Once the Web server is subverted, an attacker may be able to gain access to data and systems not part of the Web itself but connected to the server at the local site.

(v) Casual and untrained users are common clients for Web based services. Such users are not necessarily aware of the security risks that exist and do not have the tools or knowledge to take effective countermeasures.

Virtually, all business, most government agencies and many individuals now have Web sites. As a result, businesses are enthusiastic about setting up facilities on the Web for electronic commerce. But the reality is that the Internet and the Web are extremely vulnerable to compromises various sorts. As businesses wake up to this reality, the demand for secure Web services grows.

Q.75. Discuss the following Web security problem –
- ***(i) Spoofing a site to a user***
- ***(ii) Merchants unclear on the concept***
- ***(iii) Getting impersonated by a subsequent user***
- ***(iv) Cross-site scripting***
- ***(v) Poisoning cookies.***

Or

What are Web security problems ? Explain. ***(R.G.P.V., June 2011)***

Or

Write a short note on Web security problem. ***(R.G.P.V., June 2015)***

Ans. The various Web security problems are as follows –

(i) Spoofing a Site to a User – Suppose you want to trick a user into thinking he is visiting, say, his stock broker site, but instead he is visiting EmbezzlersInc. Let's say that EmbezzlersInc manages to convince the manager of the domains under country code tv to give it an innocuous sounding name like gg.tv. If EmbezzlersInc can trick you into thinking they are your stock

broker, they can connect to the site you want to talk to and act as a man-in-the-middle, learning your password and then being able to make stock trades with your money. One would hope that using SSL would make this impossible.

(ii) Merchants Unclear on the Concept – Suppose the user is very careful, so the user is talking to the server and the user session is cryptographically protected. After the user does his SSL-protected transaction, where the credit card is cryptographically protected while on the wire, it is not uncommon for the merchant to subsequently email a confirmation to the user, in the clear, with all the details of the transaction, including the credit card number!

(iii) Getting Impersonated by a Subsequent User – The authorization feature of HTTP lets a Web server signal the browser that it should prompt the user for a username and password and then the browser can calculate the proper authorization information for the server. It is natural for the browser to store the user's name and password so that on subsequent visits to that same site, the browser will not need to prompt the user for name and password again. The browser can complete the authentication on the user's behalf. If a user is surfing the Web from a public workstation and then walks away from the machine without logging out of the browser, the next user who walks up to that machine will be assumed to be the authenticated previous user. For this reason, it is important for cookies and authorization information to only be cached for a short time and certainly deleted when the user exits from the browser.

Some browser vendors have thought up a really "helpful" feature that gives us the chills. When a server requests authorization information, causing the client machine to prompt the user for username and password, the browser asks the user Alice whether she would like the browser to remember this information so that she would not be bothered in the future. It means that every subsequent user will be automatically authorized as Alice on any site that Alice authenticated. If Alice realizes her horrible error after answering "Yes" and wants to take it back, there is no easy way. You might think that she can cause the prompt to appear and this time overwrite her real information with bogus information in order to erase the dangerous information from being stored for the next user. But the browser will never prompt again ! This is such a bad feature that is should certainly be removed from browsers. But until it is, servers should not use the authorization feature of HTTP and instead accomplish the same thing by presenting the user with a login page at the server site and giving the client machine a cookie proving the user is authorized. And the server should specify that the cookie be deleted after the user exits the browser.

(iv) Cross-site Scripting – One of the types of content that can be embedded in a Web page is active content i.e., a program e.g., <SCRIPT> script commands</SCRIPT>. Types of active content are Java, Javascript and Active X.

It is theoretically possible for someone to post a message on a bulletin board with embedded active content and anyone who displayed the message would inadvertently run his script. The script commands run in a sandbox on the client machine, but the client machine would allow the script to transmit anything it wants to the server. So the program can do anything on the server that the user duped into running the script is allowed to do. For instance, if the server stores the user's email, the script might send a command to delete all the user's email or send email viruses to everyone he knows. The way this vulnerability was "fixed" was that servers that display such things check to make sure they are not displaying anything dangerous. Figuring out whether something is dangerous is tricky and if they do not get it right, there will be a loophole to exploit.

Cross-site scripting is a security vulnerability in which one site, A, can create a program that they can trick you into running on another site B. Even though it's probably not a real vulnerability, it is somewhat interesting to understand. Assume that the design of site B's Web page is such that a client can issue commands by doing a GET of a URL, where the instructions for what to do are included in the portion of the URL to the right of the single slash. For instance,

http://www.respectablestockbroker.com/transfer&$50000.&main account&account#14567.

Assume that the user has visited respectablestockbroker recently, so has a cookie from them identifying her. When she visits site A's Web page and hits the embedded URL above, this will cause her to visit site B with that URL and her cookie. The cookie will identify her to B, so B will know what her account number is. That example does not involve code. Here's an example, again convoluted and unlikely, that does involve code. Suppose site B's Web site allow you to specify a greeting that it will display when you enter the Web site and the greeting is passed to it in the URL. So, if you entered B using

http://www.respectablestockbroker.com/hello there!

it would display hello there. But suppose instead of a text string, A puts an HTML file that specifies active content e.g., <SCRIPT>script commands </SCRIPT>. When the user's browser does a GET of that URL, site B repeats the string back to the user's browser. Since site B is trusted, the browser will be willing to run the program.

(v) ***Poisoning Cookies*** – Some servers used cookies without any cryptographic protection. It was easy to look at their contents and modify them. For instance –

(a) If the cookie contained a user ID, user A could modify the user ID in the cookie so that the Web site would assume user A was really user B. This would enable A to impersonate B and have access, say, to his bank account.

(b) An on-line merchant stored the contents of the user's shopping cart, including the prices of items, in the cookie. When the user checked out, the prices charged were those stored in the cookie. So it was easy to modify the prices in the cookie so that instead of paying $60 for an item, the user would pay 35 cents.

Q.76. Write a short notes on web security and cookies. ***(R.G.P.V., May 2019)***

Or

Write short note on cookies. ***(R.G.P.V., Dec. 2011, June 2015)***

Or

Discuss the concept of a cookie. ***(R.G.P.V., June 2014)***

***Ans.* Web Security** – Refer to Q.74.

Cookies – If the client is browsing content that requires authentication and access control or is accumulating information such as items in a virtual shopping basket to be purchased when the user is finished browsing the on-line catalog, the information for that session really needs to be kept somewhere. But HTTP is stateless. Each request/response interaction is allowed to take place over a fresh TCP connection. The cookie mechanism enables the server to maintain context across many request/response interactions. A cookie is a data structure created by the server and stored at the client.

Q.77. Explain secure socket layer. ***(R.G.P.V., June 2015)***

Or

Explain architecture of Secure Socket Layer (SSL).
(R.G.P.V., June 2004, Dec. 2009, June 2010, May 2018)

Or

Write a short note on socket secure layer. ***(R.G.P.V., May 2019)***

Ans. The secure socket layer (SSL) is a mechanism invented by Netscape, Inc. to provide secure communication between a browser and a server.

When a browser encrypts information, the process is entirely hidden from a user. With SSL, a browser can encrypt a message so the contents remain private. Moreover, the entire process is automated; the browser performs the encryption without requiring the user to act.

SSL is designed to make use of TCP to provide a reliable end-to-end secure service. SSL is not a single protocol but rather two layers of protocols, as illustrated in fig. 4.19.

The SSL Record Protocol provides basic security services to various higher-layer protocols. In particular the Hyptertext Transfer Protocol (HTTP), which provides the transfer service for Web client/server interaction, can operate on top of SSL. Three higher-layer protocols are defined as part of SSL – the Handshake Protocol, the Change Cipher Spec Protocol and the Alert Protocol. These SSL-specific protocols are used in the management of SSL exchanges.

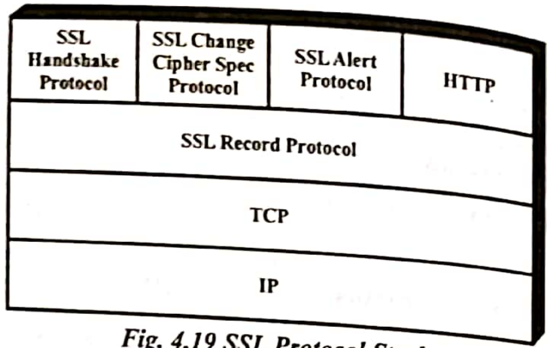


Fig. 4.19 SSL Protocol Stack

Two important SSL concepts are the SSL session and the SSL connection, which are defined in the specification as follows –

(i) Connection – A connection is a transport that provides a suitable type of service. For SSL, such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.

(ii) Session – An SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol. Sessions define a set of cryptographic security parameters, which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.

Between any pair of parties there may be multiple secure connections. Theoretically there may also be multiple simultaneous sessions between parties but this feature is not used in practice.

Actually, there are a number of states associated with each session. Once a session is established there is current operating state for both read and write. In addition, during the Handshake Protocol, pending read and write states are created. Upon successful conclusion of the Handshake Protocol, the pending states become the current states.

A session state is defined by the following parameters –

(i) Session Identifier – An arbitrary byte sequence chosen by the server to identify an active or resumable session state.

(ii) Peer Certificate – An X509.V3 certificate of the peer. This element of the state may be null.

(iii) Compression Method – The algorithm used to compress data prior to encryption.

(iv) Cipherspec – Specifies the bulk data encryption algorithm and a hash algorithm used for MAC calculation. It also defines cryptographic attributes such as the hash size.

(v) Master Secret – 48-byte secret shared between the client and server.

(vi) Isresumable – A flag indicating whether the session key can be used to initiate new connections.

A connection state is defined by the following parameters –

(i) Server and Client Random – Byte sequences that are chosen by the server and client for each connection.

(ii) Server Write MAC Secret – The secret key used in MAC operations on data sent by the server.

(iii) Client Write MAC Secret – The secret key used in MAC operations on data sent by the client.

(iv) Server Write Key – The conventional encryption key for data encrypted by the server and decrypted by the client.

(v) Client Write Key – The conventional encryption key for data encrypted by the client and decrypted by the server.

(vi) Initialization Vectors – When a block cipher in CBC mode is used, an initialization vector (IV) is maintained for each key. This field is first initialized by the SSL Handshake Protocol. Thereafter the final ciphertext block from each record is preserved for use as the IV with the following record.

(vii) Sequence Numbers – Each party maintains seperate sequence numbers for transmitted and received messages for each connection. When a party sends or receives a change cipher spec message, the appropriate sequence number is set to zero. Sequence numbers may not exceed $2^{64} - 1$.

Q.78. List and briefly define the parameters that define an SSL session state. ***(R.G.P.V., June 2012)***

Ans. Refer to Q.77.

Q.79. Explain the SSL record protocol. ***(R.G.P.V., June 2010)***

Or

What are the services provided by the SSL record protocol ? ***(R.G.P.V., Dec. 2011)***

Ans. The SSL Record Protocol provides two services for SSL connections –

(i) Confidentiality – The Handshake Protocol defines a shared secret key that is used for conventional encryption of SSL payloads.

(ii) Message Integrity – The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC).

Fig. 4.20 indicates the overall operation of the SSL Record Protocol. The record protocol takes an application message to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC, ypt, adds a header, and transmits the resulting unit in a TCP segment. Received data are decrypted, verified, decompressed, and reassembled and then delivered to higher-level users.

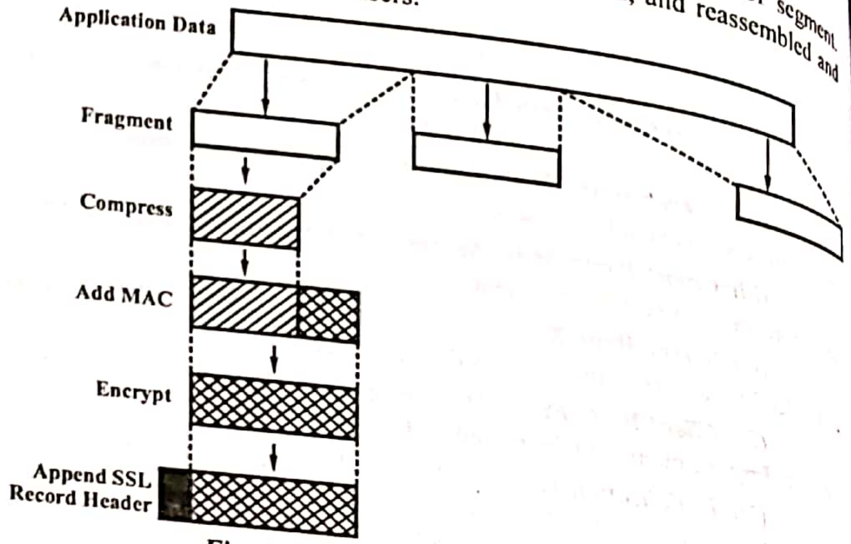


Fig. 4.20 SSL Record Protocol Operation

The first step is ***fragmentation.*** Each upper-layer message is fragmented into blocks of 2^{14} bytes (16384 bytes) or less. Next, ***compression*** is optionally applied. Compression must be lossless and may not increase the content length by more than 1024 bytes. In SLLv3, no compression algorithm is specified, so the default compression algorithm is null.

The next step in processing is to compute a ***message authentication code*** over the compressed data. For this purpose, a shared secret key is used. The calculation is defined as –

hash (MAC_write_secret || pad-2 || hash (MAC_write_secret || pad-1 || seq-num || SSL compressed.type || SSL compressed.length || SSL compressed.fragment))

where,

|| = concatenation

MAC_write_secret = Shared secret key

hash = cryptographic hash algorithm, either MD5 or SHA-1

pad-1 = the byte 0 × 36 (0011 0110) repeated 48 times (384 bits) for MD5 and 40 times (320 bits) for SHA-1.

pad-2 = the byte 0 × 5C (0101 1100) repeated 48 times for MD5 and 40 times for SHA-1.

seq_num = the sequence number for this message

SSL compressed.type = the higher-level protocol used to process this fragment.

SSL compressed.length = the length of the compressed fragment.

SSL compressed.fragment = the compressed fragment (if compression is not used, the plaintext fragment).

The difference from the HMAC algorithm is that the two pads are concatenated in SSLv3 and are XORed in HMAC.

Next, the compressed message plus the MAC are ***encrypted*** using symmetric encryption. Encryption may not increase the contact length by more than 1024 bytes, so that the total length may not exceed 2^{14} + 2048. The following encryption algorithms are permitted –

Block Cipher		*Stream Cipher*	
Algorithm	*Key Size*	*Algorithm*	*Key Size*
IDEA	128	RC4-40	40
RC2-40	40	RC4-128	128
DES-40	40		
DES	56		
3DES	168		
Fortezza	80		

Fortezza can be used in a smart card encryption scheme.

The final step of SSL Record Protocol processing is to prepend a header, consisting of the following fields –

(i) Content Type (8 bits) – The higher layer protocol used to process the enclosed fragment.

(ii) Major Version (8 bits) – Indicates major version of SSL in use. For SSLv3, the value is 3.

(iii) Minor Version (8 bits) – Indicates minor version in use. For SSLv3, the value is 0.

(iv) Compressed Length (16 bits) – The length in bytes of the plaintext fragment (or compressed fragment if compression is used). The maximum vlaue is 2^{14} + 2048.

The content types that have been defined are change-cipher-aspect, alert, hand-shake, and application-data. The first three are the SSL-specific protocols. Fig. 4.21 illustrates the SSL record format.

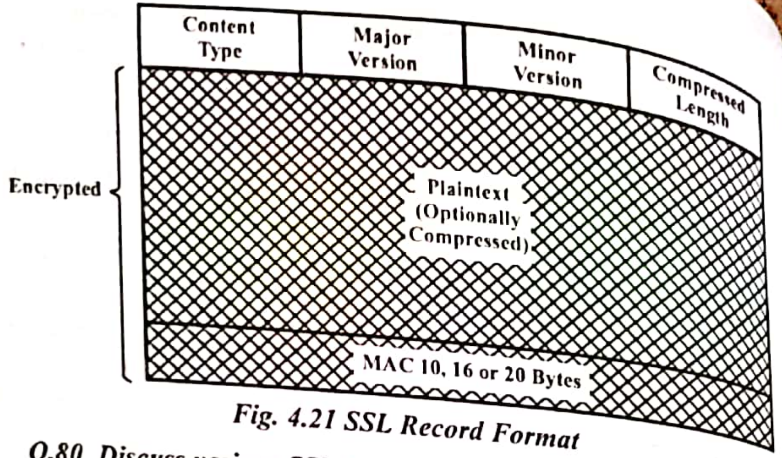


Fig. 4.21 SSL Record Format

Q.80. Discuss various SSL-specific protocols.

Or

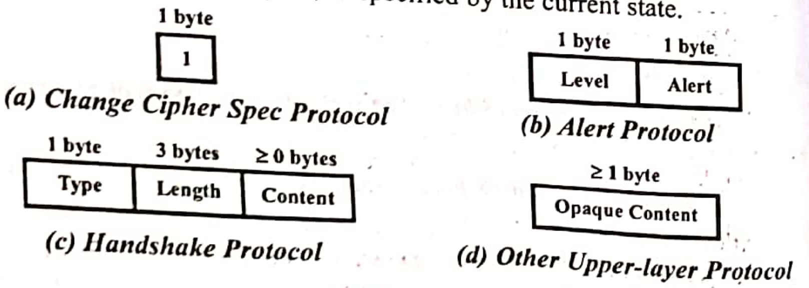
What are the protocols that SSL comprised of ? Explain.

(R.G.P.V., Dec. 2011)

Ans. The SSL-specific protocols are discussed below –

(i) Change Cipher Spec Protocol – The Change Cipher Spec protocol is one of the three SSL-specific protocols that use the SSL Record Protocol, and it is the simplest. This protocol consists of a single message [fig. 4.22 (a)], which consists of a single byte with the value 1. The sole purpose of this message is to cause the pending state to be copied into the current state, which updates the cipher suite to be used on this connection.

(ii) Alert Protocol – The alert protocol is used to convey SSL-related alerts to the peer entity. As with other applications that use SSL, alert messages are compressed and encrypted, as specified by the current state.



(a) Change Cipher Spec Protocol

(b) Alert Protocol

(c) Handshake Protocol

(d) Other Upper-layer Protocol (e.g. HTTP)

Fig. 4.22 SSL Record Protocol Payload

Each message in this protocol consists of two bytes [fig. 4.22 (b)]. The first byte takes the value warning (1) or fatal (2) to convey the severity of the message. If the level is fatal, SSL immediately terminates the connection. Other

connections on the same session may continue, but no new connections on this session may be established. The second byte contains a code that indicates the specific alert. First, we list those alerts that are always fatal.

(a) unexpected_message	(b) bad_record_mac
(c) decompression_failure	(d) handshake_failure
(e) illegal_parameter	

The remainder of the alerts are the following –

(a) close_notify	(b) no._certificate
(c) bad_certificate	(d) unsupported_certificate
(e) certificate_revoked	(f) certificate_expired
(g) certificate_unknown.	

(iii) Handshake Protocol – This protocol allows the server and client to authenticate each other and to negotiate an encryption and MAC algorithm and cryptographic keys to be used to protect data sent in an SSL record. The Handshake Protocol is used before any application data is transmitted.

The Handshake Protocol consists of a series of messages exchanged by client and server. All of those have the format shown in fig. 4.22 (c). Each message has three fields –

(a) Type (1 byte) – Indicates one of 10 messages. Table 4.4 lists the defined message types.

(b) Length (3 bytes) – The length of the message in bytes.

(c) Content (≥ 1 byte) – The parameters associated with this message, these listed in table 4.4.

Table 4.4 SSL Handshake Protocol Message Types

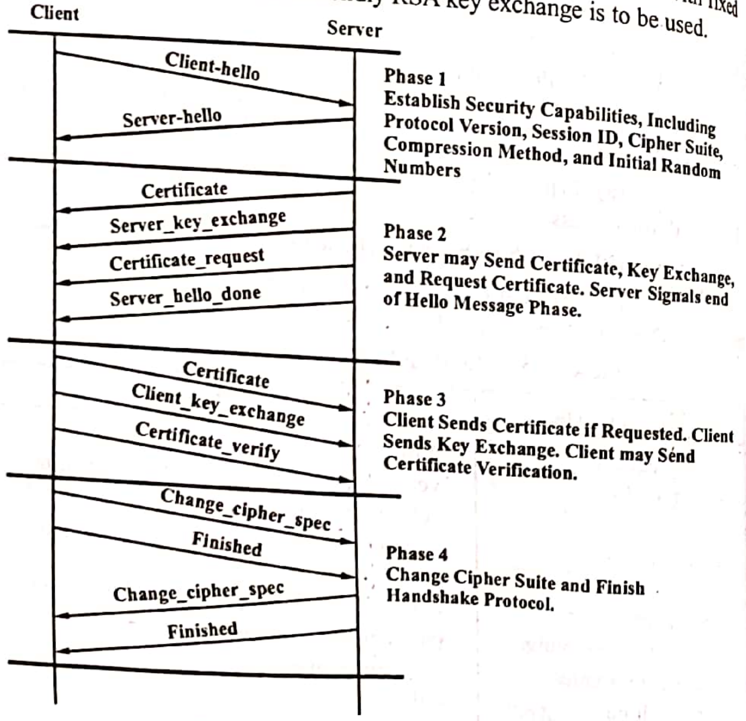
Message Type	*Parameters*
hello_request	null
client_hello	version, random, session id, cipher suite, compression method.
server_hello	version, random, session id, cipher suite, compression method.
certificate	chain of X.509v3 certificates
server_key_exchange	parameters, signature.
certificate_request	type, authorities
server_done	null
certificate_verify	signature
client_key_exchange	parameters, signature
finished.	hash value

Fig. 4.23 shows the initial exchange needed to establish a logical connection between client and server. The exchange can be viewed as having four phases –

Phase 1. Establish Security Capabilities – This phase is used to initiate a logical connection and to establish the security capabilities that will be associated with it. The exchange is initiated by the client, which sends a ***client-hello message***. After sending the client-hello message, the client waits for the ***server-hello message***.

Phase 2. Server Authentication and Key Exchange – The server begins this phase by sending its certificate, if it needs to be authenticated, the message contains one or a chain of X.509 certificates. The ***certificate message*** is required for any agreed on key exchange method except anonymous Diffie-Hellman.

Next, a ***server_key_exchange message*** may be sent if it is required. It is not required in two instances – First, the server has sent a certificate with fixed Diffie–Hellman parameters, or secondly RSA key exchange is to be used.



Client
Server
Client-hello
Server-hello
Phase 1
Establish Security Capabilities, Including Protocol Version, Session ID, Cipher Suite, Compression Method, and Initial Random Numbers
Certificate
Server_key_exchange
Certificate_request
Server_hello_done
Phase 2
Server may Send Certificate, Key Exchange, and Request Certificate. Server Signals end of Hello Message Phase.
Certificate
Client_key_exchange
Certificate_verify
Phase 3
Client Sends Certificate if Requested. Client Sends Key Exchange. Client may Send Certificate Verification.
Change_cipher_spec
Finished
Change_cipher_spec
Finished
Phase 4
Change Cipher Suite and Finish Handshake Protocol.

Fig. 4.23 Handshake Protocol Action

Phase 3. Client Authentication and Key Exchange – Upon receipt of the server_done message, the client should verify that the server provided a valid certificate if required and check that the server_hello parameters are acceptable. If all is satisfactory, the client sends one or more messages back to the server.

If the server has requested a certificate, the client begins this phase by sending a ***certificate message***. If no suitable certificate is available, the client sends a no_certificate alert instead.

Next is the ***client_key_exchange message***, which must be set in this phase. The content of the message depends on the type of key exchange.

Phase 4. Finish – This phase completes the setting up of a secure connection. The client sends a ***change_cipher_spec*** message and copies the pending Cipher Spec into the current CipherSpec. The client then immediately sends the ***finished message*** under the new algorithms, keys, and secrets. The finished message verifies that the key exchange and authentication processes were successful.

Q.81. Why is the SSL layer positioned between the application layer and the transport layer ? ***(R.G.P.V., June 2014)***

Ans. The application layer of the sending computer prepares the data to be sent to the receiving computer. The application layer data is passed to the SSL layer which performs encryption on the received data and also adds its own encryption header known as SSL header. Thereafter SSL layer data passed to the transport layer which adds its own header and passes it on to the Internet layer and so on. Finally, the data is sent in the form of voltage pulses across the transmission medium when the data reaches the physical layer. The process happens similar to how it happens in the case of a normal TCP/IP connection at the receiver's end until it reaches the SSL layer. The SSL layer removes the SSL header, decrypts the encrypted data and gives the plain text back to the application layer. Thus, only the application layer data is encrypted by SSL. If SSL has to encrypt all the headers, it must be positioned below the data link layer. That would serve no purpose at all. In fact, it would lead to problems. If SSL encrypted all the lower layer headers, even the IP and physical address of the computers would be encrypted and become unreadable. Thus, the address to deliver the packets would be unknown. That's why SSL layer positioned between the application layer and the transport layer.

Q.82. What is TLS ? Explain the differences between SSL and TLS protocols.

Or

What do you mean by transport layer security (TLS) ? Explain. ***(R.G.P.V., June 2011)***

Ans. Transport Layer Security (TLS) is an IETF standardization initiative whose goal is to produce an Internet standard version of SSL. TLS is defined as a proposed Internet standard in RFC 2246. RFC 2246 is similar to SSLv3. The differences between SSL and TLS protocols are –

(i) ***Version Number*** – The TLS record format is same as that of the SSL record format and the fields in the header have the same meanings. The one difference is in version values. For the current version of TLS, the major version is 3 and minor version is 1.

(ii) ***Cipher Suite*** – Another major difference between SSL and TLS is the lack of support for the Fortezza method. TLS does not support Fortezza for key exchange or for encryption/decryption.

(iii) ***Message Authentication Code*** – There are two differences between the SSLv3 and TLS MAC schemes – the actual algorithm and the scope of the MAC calculation. TLS makes use of the HMAC algorithm defined in RFC 2104. HMAC is defined as –

$$HMAC_K(M) = H[(K^+ \oplus opad) \| H[(K^+ \oplus ipad) \| M]]$$

where H is embedded hash function, M is message input to HMAC, K^+ is secret key padded with zeros on the left so that the result is equal to block length of the hash code, opad is 01011100 (5C in hexadecimal) repeated 64 times (512 bits) and ipad is 00110110 (36 in hexadecimal) repeated 64 times (512 bits).

SSLv3 uses the same algorithm, except that the padding bytes are concatenated with the secret key rather than being XORed with the secret key padded to the block length.

For TLS, the MAC calculation encompasses the fields indicated in the following expression –

HMAC_hash (MAC_write_secret, seq_num||TLSCompressed.type||
TLSCompressed.version||TLSCompressed.length||
TLSCompressed.fragment)

The MAC calculation covers all of the fields covered by the SSLv3 calculation, plus the field TLSCompressed.version, which the version of the protocol being employed.

(iv) ***Pseudorandom Function*** – It is referred to as PRF to expand secrets into blocks of data for purposes of key generation or validation. The objective is to make use of a relatively small shared secret value but to generate longer blocks of data in a way that is secure from the kinds of attacks made on hash functions and MACs. The PRF is based on the following data expansion function (fig. 4.24).

P_hash(secret, seed) = HMAC_hash(secret, A(1)||seed)||
HMAC_hash(secret, A(2)||seed)||
HMAC_hash(secret, A(3)||seed)||...

where A( ) is defined as

A(o) = seed

A(i) = HMAC_hash(secret, A(i – 1))

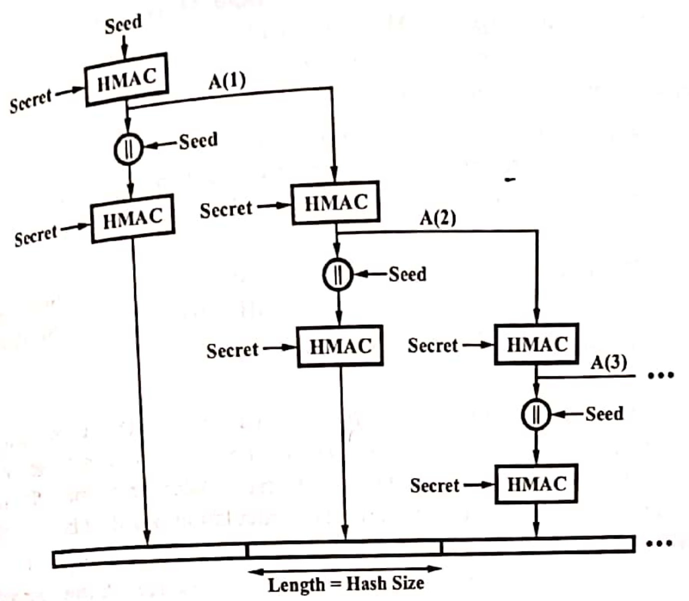


Fig. 4.24 TLS Function P_hash(secret, seed)

The data expansion function makes use of the HMAC algorithm, with either MD5 or SHA-1 as the underlying hash function. As can be seen, P_hash can be iterated as many times as necessary to produce the required quantity of data.

To make PRF as secure as possible, it uses two hash algorithms in a way that should guarantee its security if either algorithm remains secure. PRF is defined as –

PRF(secret, label, seed) = P_MD5(S1, label||seed) ⊕ P_SHA-1 (S2, label||seed)

PRF takes as input a secret value, an identifying label and a seed value and produces an output of arbitrary length. The output is created by splitting the secret value into two halves and performing P_hash on each half, using MD5 on one half and SHA-1 on the other half. The two results are XORed to produce the output; for this purpose, P_MD5 will generally have to be iterated more times than P_SHA-1 to produce an equal amount of data for input to the XOR function.

(v) ***Client Certificate Types*** – TLS defines the following certificate types to be requested in a certificate_request message – rsa_sign, dss_sign, rsa_fixed_dh and dss_fixed_dh. These are all defined in SSLv3. In addition, SSLv3 includes rsa_ephemeral_dh, dss_ephemeral_dh and fortezza_kea. Ephemeral Diffie-Hellman involves signing the Diffie-Hellman parameters with either RSA or DSS; for TLS, the rsa_sign and dss_sign types are used for that function; a separate signing type is not needed to sign Diffie-Hellman parameters. TLS does not include the Fortezza scheme.

(vi) Certificate_verify and Finished Messages – In the TLS certificate_verify message, the MD5 and SHA-1 hashes are calculated only over handshake_messages. For SSLv3 the hash calculation also include the master secret and pads. These extra fields were felt to add no additional security.

As with the finished message in SSLv3, the finished message in TLS is a hash based on the shared master_secret, the previous handshake messages and a label that identifies client or server. The calculation is somewhat different. For TLS, we have

PRF(master_secret, finished_label, MD5(handshake_messages)||
SHA-1(handshake_messages))

where finished_label is the string "client finished" for the client and "server finished" for the server.

(vii) Cryptographic Computations – The pre_master_secret for TLS is calculated in the same way as in SSLv3. As in SSLv3, the master_secret in TLS is calculated as a hash function of the pre_master_secret and the two hello random numbers. The form of the TLS calculation is different from that of SSLv3 and is defined as follows –

master_secret = PRF(pre_master_secret, "master secret",
ClientHello.random||ServerHello.random)

the algorithm is performed until 48 bytes of pseudorandom output are produced. The calculation of the key block material is defined as follows –

key_block = PRF(master_secret, "key expansion", securityParameters.
server_random||securityParameters.client_random)

until enough output has been generated. As with SSLv3, the key_block is a function of the master_secret and the client and server random numbers, but for the TLS actual algorithm is different.

(viii) Padding – In SSL, the padding added prior to encryption of user data is the minimum amount required so that the total size of the data to be encrypted is a multiple of the cipher's block length. In TLS, the padding can be any amount that results in a total that is a multiple of the cipher's block length, upto a maximum of 255 bytes.

Q.83. Explain secure socket layer and transport layer security. ***(R.G.P.V., June 2016)***

Ans. **Secure Socket Layer** – Refer to Q.77.

Transport Layer Security – Refer to Q.82.

Q.84. Discuss various alert codes of TLS. ***(R.G.P.V., June 2015)***

Ans. TLS support all of the alert codes defined in SSLv3 except no_certificate. A number of additional codes are defined in TLS; of these, the following are always fatal –

(i) Decryption_failed – A ciphertext decrypted in an invalid way; either it was not an even multiple of the block length or its padding values, when checked, were incorrect.

(ii) Record_overflow – A TLS record was received with a payload whose length exceeds $2^{14} + 2048$ bytes or the ciphertext decrypted to a length of greater than $2^{14} + 1024$ bytes.

(iii) Unknown_ca – A valid certificate chain or partial chain was received, but the certificate was not accepted because the CA certificate could not be located or could not be matched with a known, trusted CA.

(iv) Access_denied – A valid certificate was received, but when access control was applied, the sender decided not to proceed with the negotiation.

(v) Decode_error – A message could not be decoded because a field was out of its specified range or the length of the message was incorrect.

(vi) Export_restriction – A negotiation not in compliance with export restrictions on key length was detected.

(vii) Protocol_version – The protocol version the client attempted to negotiate is recognized but not supported.

(viii) Insufficient_security – Returned instead of handshake_failure when a negotiation has failed specifically because the server requires ciphers more secure than those supported by the client.

(ix) Internal-error – An internal error unrelated to the peer or the correctness of the protocol makes it impossible to continue.

The remainder of the new alerts are the following –

(i) Decrypt_error – A handshake cryptographic operation failed, including being unable to verify a signature, decrypt a key exchange or validate a finished message.

(ii) User_canceled – This handshake is being canceled for some reason unrelated to a protocol failure.

(iii) No_renegotiation – Sent by a client in response to a hello request or by the server in response to a client hello after initial handshaking. Either of these messages would normally result in renegotiation, but this alert indicates that the sender is not able to renegotiate. This message is always a warning.

Q.85. What is secure electronic transaction (SET) ? ***(R.G.P.V., Dec. 2003, June 2004)***

Or

Explain secure electronic transaction (SET). ***(R.G.P.V., June 2009)***

Or

What do you mean by secure electronic transaction ? Explain in brief. ***(R.G.P.V., Dec. 2009)***

Ans. Secure electronic transaction (SET) is an open encryption and security specification designed to protect credit card transactions on the Internet. The SET standards were developed collectively by a number of companies, including IBM, Microsoft, Netscape, RSA, Terisa, Visa International, and Verisign. Trials and tests were carried out as early as 1996 and the first set of SET-compliant products were available in 1998. SET is not itself a payment system. Rather it is a set of security protocols and formats that enable users to employ the existing credit card payment infrastructure on an open network, such as the Internet, in a secure fashion. SET is made up of three services –

(i) The provision of a secure channel between all parties involved in a transaction.

(ii) The provision of a trust relationship based on X.509 certificates.

(iii) The provision of privacy by making information available to any party in a transaction only where and when it is needed.

Q.86. What security protocols are predominantly used in web-based electronic commerce ? ***(R.G.P.V., June 2011)***

Ans. The predominantly used security protocols in web-based electronic commerce are secure electronic transaction (SET), and 3-D secure.

SET – Refer to Q.85.

3-D Secure – 3D secure protocol was developed by Visa. The difference between SET and 3D secure protocol is that any cardholder who wants to participate in a payment transaction involving the usage of the 3D secure protocol has to enroll on the issuer bank's enrolment server. That is, the user must enroll with the issuer bank's enrolment server before a cardholder makes a card payment. During actual 3D secure transaction, when the merchant receives a payment instruction from the cardholder, the merchant forwards this request to the issuer bank through the Visa network. The issuer bank needs the cardholder to give the user id and password that were created during enrolment process. The cardholder gives these details which are verified by issuer bank. Only after the user is authenticated successfully, the issuer bank informs the merchant that it can accept the card payment instruction.

●●

UNIT 5

CRYPTOGRAPHY AND INFORMATION SECURITY TOOLS – SPOOFING TOOLS LIKE ARPING ETC., FOOT PRINTING TOOLS (EX-NSLOOKUP, DIG, WHOIS, ETC.), VULNERABILITIES SCANNING TOOLS (i.e. ANGRY IP, HPing2, IP SCANNER, GLOBAL NETWORK INVENTORY SCANNER, NET TOOLS SUITE PACK) NETBIOS ENUMERATION USING NET VIEW TOOL

Q.1. What is spoofing ? Also write its' different form.

Ans. The term spoofing applies to actions that make an electronic transmission appear to originate from somewhere that it does not spoofing can be used to steal sensitive information such as the social engineering attacks based on e-mail and web spoofs.

There are different forms of spoofing. The first is IP spoofing in which unauthorized computer access is gained when an intruder sends messages including an IP address that indicates, by modifying packet headers, that the message is from a trusted host. Web spoofing creates a "shadow copy" of the entire Web. This copy is sent through the victim's computer while tracking all of the users activities on "the Web" including passwords, account numbers, or any other information that the victim may enter. A third example of spoofing is e-mail spoofing. E-mail spoofing occurs when a user receives an e-mail that appears to have originated from a source different from the source that it was actually from. The arping tool is used in the Linux platform to send ARP request messages to a destination host in a LAN. It is used to test whether an IP address is in use or not.

Q.2. Write short note on IP spoofing. ***(R.G.P.V., May 2018)***

Ans. The most common type of spoofing that you are likely to encounter is IP spoofing, used primarily to spoof the source address of e-mail. In this case, an e-mail message looks like it comes from one address, when in fact , comes from somewhere else instead. The intent is to trick the user into thinking the e-mail comes from a trusted source so that the user will open the e-mail and act on it in some way.

E-mail spoofing can be used to –

(i) Deliver a phishing message (one that cons the user into divulging confidential information). Replying to the e-mail won't work properly, but clicking on links in the e-mail will take the user to a spoofed Web site.

(ii) Deliver a malware payload, such as a virus, worm, or Trojan horse. The malware may come as an attachment that must be downloaded (and perhaps executed) or may be coded into the e-mail so that all the user needs to do is open the e-mail. (The malware installs itself when the e-mail is opened).

One of the more annoying e-mail spoofing tricks is to use the contents of the e-mail address book on a compromised machine as the sources of spoofed e-mail. If you happen to receive an e-mail message, then you know that your address has been harvested and used in that way.

Q.3. What do you understand by ARP spoofing ? Explain.

Ans. ARP stands for address resolution protocol. ARP is used to map IP addresses to hardware addresses. A table, usually called the ARP cache, is used to maintain a correlation between each MAC address and its corresponding IP address. ARP provides the protocol rules for making this correlation and providing address resolution in both directions. When an incoming packet sent to a host machine on a network arrives at a router, it asks the ARP program to find a MAC address that matches the IP address. The ARP program looks in the ARP cache and, if it finds the address, provides it so that the packet can be converted to the right packet length and format and sent to the machine. If no entry is found for the IP address, ARP broadcasts a request packet in a special format to all the machines on the network to determine if any machine knows who has that IP address. A machine that recognizes the IP address as its own returns a reply so indicating. ARP updates the ARP cache for future reference and then sends the packet to the MAC address that replied.

One might deduct that this addressing scheme could also be spoofed to provide a host with incorrect information "ARP spoofing involves constructing forged ARP request and reply packets. By sending forged ARP replies, a target computer could be convinced to send frames destined for computer A to instead go to computer B." This referred to as ARP poisoning. There are currently programs that automate the process of ARP poisoning – ARPoison, Ettercap, and Parasite. All three have the capability to provide spoofed ARP packets and therefore redirect transmission, intercept packets, and/or perform some type of man in the middle attack. Either enabling MAC binding at a switch or implementing static ARP tables achieves prevention of ARP spoofing. MAC binding makes it so that once an address is assigned to an adapter; it cannot be changed without authorization. Static ARP management is only realistically achieved in a very small network. In a large dynamic network, it would be impossible to manage the task of keeping the entries updated. ARPWATCH, for UNIX based systems, monitors changes to the ARP cache and alerts administrator as to the changes.

Q.4. Write short note on footprinting.

Ans. Footprinting is the first and most convenient way that hackers use to gather information about computer systems and the companies they belong to. The purpose of footprinting to learn as much as you can about a system, it's remote access capabilities, its ports and services, and the aspects of its *security*.

In order to perform a successful hack on a system, it is best to know as much as you can, if not everything, about that system. While there is nary a company in the world that is not aware of hackers, most companies are now hiring hackers to protect their systems. And since footprinting can be used to attack a system, it can also be used to protect it. If you can find anything out about a system, the company that owns that system, with the right personnel, can find out anything they want about you.

Footprinting is necessary for one basic reason – it gives you a picture of what the hacker sees. And if you know what the hacker sees, you know what potential security exposures you have in your environment. And when you know what exposures you have, you know how to prevent exploitation.

Hackers are very good at one thing – getting inside your head, and you do not even know it. They are systematic and methodical in gathering all pieces of information related to the technologies used in your environment. Without a sound methodology for performing this type of reconnaissance yourself, you are likely to miss key pieces of information related to a specific technology or organization – but trust me, the hacker won't.

Be forewarned, however, footprinting is often the most arduous task of trying to determine the security posture of an entity; and it tends to be the most boring for freshly minted security professionals eager to cut their teeth on some test hacking. However, footprinting is one of the most important steps and it must be performed accurately and in a controlled fashion.

Some of the common techniques used for information gathering in footprinting phase include the following –

(i) DNS enumeration and identify types of DNS records

(ii) Nslookup and DNSstuff

(iii) Whois and AfriNIC Lookups and analyzing Whois output

(iv) Finding the address range of the network

(v) Using traceroute

(vi) E-mail tracking

(vii) Web spiders.

Q.5. Discuss about the active and passive footprinting.

Ans. **Active Footprinting** – In active footprinting the hackers directly interacts with the system or application to gather the information about the system. In the case of active footprinting there is a high possibility that the target system saves the information such as IP address.

Passive Footprinting – In this technique the hackers can collect system or application information without interacting with the system directly. Here, the search engines or public records help the hacker in collecting the information from the system.

Q.6. Explain the footprinting tools.

Ans. The footprinting tools are as follows –

(i) Nslookup – Nslookup means name server lookup. To execute queries, nslookup uses the operating system's local domain name system (DNS) resolver library. Nslookup operates in interactive or non-interactive mode. When used interactively by invoking it without arguments or when the first argument is – (minus sign) and the second argument is host name or IP address, the user issues parameter configurations or requests when presented with the nslookup prompt (>). When no arguments are given, then the command queries to default server. The – (minus sign) invokes subcommands which are specified on command line and should precede nslookup commands. In non-interactive mode, i.e. when first argument is name or internet address of the host being searched, parameters and the query are specified as command line arguments in the invocation of the program. The non-interactive mode searches the information for specified host using default name server.

One of the powerful tools queries DNS servers for record information. It's included in UNIX, Linux, and Windows operating systems. Nslookup is a network administration command-line tool available in many computer operating systems for querying the domain name system (DNS) to obtain domain name or IP address mapping or for any other specific DNS record.

(ii) WHOIS – WHOIS (pronounced "who is") is an Internet database that contains information on domain names including the name servers associated with the domain name, the domain registrar and the administrative, billing and technical contacts with postal and e-mail addresses.

The WHOIS is also a tool or an application which searches the domain name information contained in WHOIS database. It is generally used to check either the availability of a domain name or the ownership of a domain name. The tool requires you to enter a domain name such as sustech.edu (without the www prefix). If the domain is available you will be informed of the same, else, you would be displayed one or more details –

(a) The registrant information. Details of the person who registered the domain name including their postal and e-mail addresses and phone number.

(b) The contacts – Each domain name is associated with three contacts – Administrative, billing and technical. In most cases, all the three would belong to the same person (the registrant).

(c) The creation and expiration data of the domain name.

(d) The name servers associated with the domain name.

(iii) Dig – Dig (domain information groper), part of the popular DNS server BIND, is a command-line tool that can be used to query DNS servers. It is DNSSEC capable and can be used to verify the DNSSEC chain of trust from a top-down and a bottom-up perspective. However, we found that the current version queries all possible name servers for a TLD or authoritative zone for their A-record, even when glue records are known, when using the top-down approach, resulting in an infeasible amount of lookups. Hence, we only used Dig in a bottom-up approach using a DNSSEC-capable resolver as performed in our second measurement scenario in "Bottom-up measurement scenario".

Q.7. What are the advantages of footprinting ?

Ans. The advantages of footprinting are as follows –

(i) Footprinting allows hackers to gather the basic security configurations of a target machine along with network route and data flow.

(ii) Once attacker finds the vulnerabilities he/she focuses towards a specific area of the target machine.

(iii) It allows the hacker to identify as to which attack is more handy to hack the target system.

Q.8. What is scanning ? Discuss various types of scanning in brief.

Ans. Scanning is a systematic process of sweeping through a collection of data looking for a specific pattern. In a network environment, the scanning process may involve a program that sweeps through thousands of IP addresses looking a particular IP address string or a string that represents a vulnerability or a string that represents a vulnerable port number.

Table 5.1 lists the three types of scanning.

Table 5.1 Types of Scanning

Scanning Type	*Purpose*
Port scanning	Determines open ports and services
Network scanning	IP addresses
Vulnerability scanning	Presence of known weaknesses

(i) Port Scanning – Port scanning is the process of identifying open and available TCP/IP ports on a system. Port-scanning tools enable a hacker to learn about the services available on a given system. Each service or application on a machine is associated with a *well-known* port number. For example, a port-scanning tool that identifies port 80 as open indicates a web server is running on that system. Hackers need to be familiar with well-known port numbers.

(ii) Network Scanning – Network scanning is a procedure for identifying active hosts on a network, either to attack them or as a network security assessment. Hosts are identified by their individual IP addresses. Network-scanning tools attempt to identify all the live or responding hosts on the network and their corresponding IP addresses.

(iii) Vulnerability Scanning – Vulnerability scanning is the process of proactively identifying the vulnerabilities of computer systems on a network. Generally, a vulnerability scanner first identifies the operating system and version number, including service packs that may be installed. Then, the vulnerability scanner identifies weaknesses or vulnerabilities in the operating system. During the later attack phase, a hacker can exploit those weaknesses in order to gain access to the system.

An intrusion detection system (IDS) or a sophisticated network security professional with the proper tools can detect active port-scanning activity. Scanning tools probe TCP/IP ports looking for open ports and IP addresses, and these probes can be recognized by most security intrusion detection tools. Network and vulnerability scanning can usually be detected as well, because the scanner must interact with the target system over the network.

Q.9. What are the forms of scanning ?

Ans. There are two forms of scanning – *pattern-based* and *heuristic scanning.*

(i) Pattern-based Scanning – In pattern-based, scanning all content coming into or leaving the network, an ISP gateway, or user PC is scanned and checked against a list of patterns, or definitions, supplied and kept up-to-date by the vendor. The technique involves simply comparing the contents, which can be done in many ways. Almost all antivirus software packages work this way. This approach can, however, be slow and resource intensive.

(ii) Heuristic Scanning – Heuristics scanning is performed by looking at a section of code and determining what it is doing, then deciding, whether the behaviour exhibited by the code is unwanted, harmful like a virus or otherwise malicious. This approach to scanning is difficult because it involves modeling the behaviour of code and comparing that abstract model to a rule set. The rule set is kept in a rule database on the machine and the database is updated by the vendor. This approach is time consuming because of the checking and cross-checking and it is also resource intensive, if not more than the previous one. Theoretically, heuristics has many advantages over pattern-based scanning including better efficiency and accuracy. It can, potentially, detect viruses that have not been written yet.

Q.10. Discuss about the vulnerabilities scanning tools.

Ans. Some important vulnerabilities scanning tools are as follows –

(i) Angry IP Scan – Angry IP scanner is a tool that scans network for open IP addresses designed for network administrator to check the network security. Angry IP scanner is a cross-platform port and IP scanner. The application is developed in Java, so it is cross platforms compatible with different OS. It is a great program for doing a network audit or for just finding out more information about your network. It can locate in any network device that responds to the scan. It can locate on any device in the network that has an IP address and that doesn't have any firewall. It performs basic host discovery and port scans on Windows. The size of its binary file is very small as compared to other scanners and other pieces of information about the target hosts that can be extended with plug-in.

Features of angry IP scanner tool are as follows –

(a) It is open source software, means free to use.

(b) The fastest IP scanner.

(c) Cross-platform tool (supporting Linux, Windows, Mac OS).

(d) Light weighted tool so its CPU utilization is less.

(e) No installation is required.

(f) It can get the host name.

(g) Design for multiple host.

(h) Number of routers per trip and distance between source and destination.

(i) Cross-platform application.

(ii) Ping – This tool is used to test network connectivity or reachability of a host on an IP network. Ping is a pioneering tool developed to check a computer or router, and Internet connectivity. The ping request is sent to a particular host or to a network using command prompt. As a reply it displays the response of the destination host and how long it takes to receive a reply. It uses the ICMP protocol, which has low priority and slower speed than regular network traffic.

(iii) Hping2 – It is used to send custom TCP/IP packets and display reply messages received from the target. It handles fragmentation and arbitrary packet size, and can also be used to transfer files. It performs firewall rule testing, port scanning, protocol based network performance testing, and path MTU discovery.

(iv) Hping3 – This tool works almost like Hping2 and can handle fragmentation with arbitrary packet size. It finds the sequence number for reply packets from the source port. It starts with a base source port number and increases this number as packets are sent. The default base source port is random. The source port number may be kept constant for each packet sent.

(v) IP Scanner – This tool used to test whether a specific host is reachable over a network. It is furthermore used to individual test the network interface card of the device, or for speed test. The work of ping is done by

sending ICMP – echo request packets to the target and listening for ICMP – echo response. The round trip time is measured by ping, it moreover records any packet lost and prints when finished a measurable rundown of the echo response packets retrieved, the minimum, mean, max and in some versions the standard deviation of the round trip time.

(vi) Global Network Inventory – It is a powerful and flexible software and hardware inventory system that can be used as an audit scanner in an agent-free and zero deployment environments. If used as an audit scanner, it only requires full administrator rights to the remote computers you wish to scan. Global network inventory can audit remote computers and even network appliances, including switches, network printers, document centers, etc.

Global network inventory agent can also be deployed to perform regular audits initiated through the domain login script when your users log on the network. In this scenario, global network inventory agent is exported to a shared network directory, and audit results are collected in audit repository directory as snap files and later merged into the main database.

Global network inventory key features are as follows –

(a) Scan computers by IP range, by domain, single computers, or computers, defined by the global network inventory host file.

(b) Reliable IP detection and identification of network appliances such as switches, network printers, document centers, and other devices.

(c) Scan only items that you need by customizing scan elements.

(d) View scan results, including historic results for all scans, individual machines, or selected number of addresses.

(e) Fully customizable layouts and colour schemes on all views and reports. Export data to HTML, XML, Microsoft Excel, and text formats.

(f) Customizable printing.

(g) Schedule inventory scans to run at specified time, hourly, daily, weekly, monthly, and annually. Ability to generate reports on schedule after every scan, daily, weekly, or monthly.

(h) Reports can be saved on disk, sent via e-mail, or both.

(i) The program comes with dozens of customizable reports. New reports can be easily added through the user interface.

(j) Audits can be performed by deploying a scan agent through the domain login script.

(k) Licenses are network based rather than user based. In addition, extra licenses to cover additional addresses can be purchased at any time if required.

(vii) Net Tools Suite Pack – Impressive suit that incorporates the following utilities – IP address scanner, IP calculator, IP converter, Port listener, Port scanner, Ping, NetStat, Trace route, TCP/IP configuration, Online checker, offline, Resolve domain name and IP, Time sync, Whois and MX lookup, Port connector, Analysis and protection connector, Net sender, E-mail seeker, Net pager, Active and passive port scanner, Spoofer, Hack trapper, HTTP flooder (DoS), Mass website visiter, Advanced ports, Trojan hunter multi IP, Ports connector, Advanced spoofer, Advanced anonymous e-mailer, Simple anonymous e-mailer, Anonymous e-mailer with attack support, Mass e-mailer, E-mail bomber, E-mail spoofer, Simple port scanner (fast), Advanced netstat monitoring, X Pinger, Web page scanner, Fast port scanner, Deep port scanner, Host scanner (UDP), Get header, Open port scanner, Multi port scanner, HTTP scanner, Multi ping for cisco routers, TCP packet sniffer, UDP flooder, Resolve and ping, Multi IP ping, File dependency sniffer, EXE-joiner, Encryptor, Advanced encryption, File difference engine, File comparison, Mass file renamer, Add bytes to an EXE, Variable encryption, Simple file encryption, From ASCII to binary, Enigma, Unmask password, Credit card number generator, HTTP server, eXtreme UDP flooder, Web server scanner, Forced reboot, Webpage info seeker, Bouncer, Advanced sniffer pack, IRC server, Connection test, Fake mail sender, Bandwidth monitor, Remote desktop scanner, MX query, Messenger packet sniffer, API spy, DHCP restart, File merger, E-mail extractor, Open FTP scanner.

Q.11. Explain the architecture of vulnerability scanners.

Ans. Vulnerability scanning means scanning of the systems, network devices and applications which works on front to external worlds or scanning the internally hosted system to find the security flaws on them. There are number of different approaches to understand the basic framework of vulnerability scanners. Vulnerability scanners have a database of already exposed vulnerabilities, with reference to known vulnerability, vulnerability scanner performs the security verification on remote host.

Vulnerability scanner is break down into four major modules, such as user interface, scan engine, scan databases, report generation module.

(i) User Interface – This is the part where user interact with scanner system to execute or configure their scan. This interface can be a graphical user interface (GUI) or a command line interface (CLI) or both.

(ii) Scan Engine – The scan engine part performance the security validation based on latest installed plug-ins and payloads. User can perform the single system scan or multiple host scan at a single time also.

(iii) Scan Database – The vulnerability database stores all the scan results previously performed. The scan database contains all the information related to port, packet type, services, a potential path to exploit, latest attack techniques etc. This may also contain the different techniques to patch the vulnerabilities and have detailed information of CVE-ID mapping (common vulnerability and exposures).

(iv) Report Module – The report module generate the different types of report such as a detailed report, a list of vulnerabilities, a graphical report with their recommendation to mitigate the detected vulnerabilities.

The architecture of vulnerability scanners is shown in fig. 5.1.

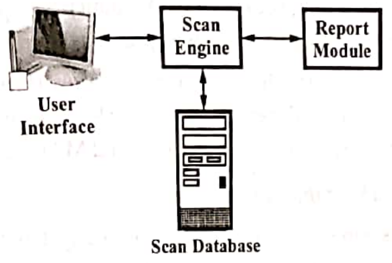


Fig. 5.1 Components of Vulnerability Scanner

Q.12. Define the term Nessus.

Ans. Nessus is one of the most popular vulnerability scanners. It is used for both authenticated and unauthenticated vulnerability scans. It is suitable for both internal and external network scans. It is also performed the scanning of web applications. The main advantage of this tool is to perform the multiple host scanning at once. The detected vulnerability is categories into four types based on their severity levels – High, Medium, Low and Informal.

A detail scan result is automatically saved as the scanning of desired host is completed. The results are expressed into two different forms – first is vulnerabilities by plug-ins and second is vulnerabilities by host. Firstly classifies the all detected vulnerabilities during scan, and then it shows the list of all hosts affected by these vulnerabilities. By using the detailed generated scan report, issues can be addressed easily. Then afterward finds the all host in scanning phase and their existed vulnerabilities.

This report will help the security administrator to address the distinct issues associated with individual host and overall networks. Its real time active scanning provides continuous network evaluation and bridges the security gaps. Nessus scan result can be exported in different formats which you desired like

PDF, HTM, and CSS etc. Nessus is works on the principle of client-server architecture. Each scan session is managed by client and scan test is done on the servers.

Q.13. *Write short note on enumeration.*

Ans. Enumeration occurs after scanning and is the process of gathering and compiling usernames, machine names, network resources, shares, and services. It also refers to actively querying or connecting to a target system to acquire this information.

The objective of enumeration is to identify a user account or system account for potential use in hacking the target system. It is not necessary to find a system administrator account, because most account privileges can be escalated to allow the account more access than was previously granted.

Many hacking tools are designed for scanning IP networks to locate NetBIOS name information. For each responding host, the tools list IP address, NetBIOS computer name, logged-in-username, and MAC address information.

Q.14. *Discuss about the null sessions.*

Ans. A null session occurs when we log in to a system with no username or password. NetBIOS null sessions are a vulnerability found in the common Internet file system (CIFS) or SMB, depending on the operating system – Microsoft Windows uses SMB, and Unix/Linux systems use CIFS – Once a hacker has made a NetBIOS connection using null session to a system, they can easily get a full dump of all usernames, groups, shares, permissions, policies, services and more using the NULL user account. The SMB and NetBIOS standards in Windows include APIs that return information about a system via TCP port 139.

One method of connecting a NetBIOS null session to a Windows system is to use the hidden inter process communication share (IPCS). This hidden share is accessible using the net use command. The net use command is a built-in Windows command that connects to a share on another computer. The empty quotation marks ("") indicate that we want to connect with no username and no password.

Once the net use command has been successfully completed, the hacker has a channel over which to use other hacking tools and techniques.

Q.15. *How can NetBIOS enumeration using NetView tool ?*

Ans. Many hacking tools are designed for scanning IP networks to locate NetBIOS name information. For each responding host, the tools list IP address, NetBIOS computer name, logged-in username, and MAC address information.

On a Windows 2000 domain, the built-in tool net view can be used for NetBIOS enumeration. To enumerate NetBIOS names using the net view command, enter the following at the command prompt –

```
net view/domain
nbstat – A IP address
```

The net view command is a great example of a built-in enumeration tool. Net view is an extraordinarily simple command-line utility that will list domains available on the network and then lay bare all machines in a domain. Here's how to enumerate domains on the network using net view –

```
C:\>net view/domain
Domain
-------------------------------------------------------------------
CORLEONE
BARZINI_DOMAIN
TATAGGLIA_DOMAIN
BRAZZI
The command completed successfully.
```

Supplying an argument to the /domain switch will list computers in a particular domain, as shown next –

```
C:\>net view /domain : corleone
Server Name              Remark
-------------------------------------------------------------------
\\VITO                   Make him an offer he can't refuse
\\MICHAEL                Nothing personal
\\SONNY                  Badda bing badda boom
\\FREDO                  I'm smart
\\CONNIE                 Don't forget the cannoli
```

For the command-line challenged, the network neighbourhood shows essentially the same information shown in these commands. However, because of the sluggishness of updates to the browse list, we think the command-line tools are snappier and more reliable.

Another great built-in tool is nbtstat, which calls up the NetBIOS name table from a remote system. The name table contains a great deal of information, as shown in the following example –

```
C:\>nbtstat – A 192.168.202.33
```

```
Local Area Connection :
Node IpAddress : [192.168.234.244] Scope Id: [ ]
      NetBIOS Remote Machine Name Table
      Name           Type     Status
--------------------------------------------------------
      CAESARS        <00>     UNIQUE    Registered
      VEGAS2         <00>     GROUP     Registered
      VGAS2          <1C>     GROUP     Registered
      CAESARS        <20>     UNIQUE    Registered
      VEGAS2         <1B>     UNIQUE    Registered
      VEGAS2         <1E>     GROUP     Registered
      VEGAS2         <1D>     UNIQUE    Registered
   ..__MSBROWSE__.   <01>     GROUP     Registered
MAC Address = 00 – 01 – 03 – 27 – 93 – 8F
```

Q.16. Discuss the various types of NetBIOS enumeration tools.

Ans. The various types of NetBIOS enumeration tools are as follows –

(i) DumpSec – It is a NetBIOS enumeration tool. It connects to the target system as a null user with the net use command. It then enumerates users, groups, NTFS permissions, and file ownership information.

(ii) Hyena – It is a tool that enumerates NetBIOS shares and additionally can exploit the null session vulnerability to connect to the target system and change the share path or edit the registry.

(iii) SMB Auditing – It is a password-auditing tool for the Windows and Server message block (SMB) platforms. Windows uses SMB to communicate between the client and server. The SMB auditing tool is able to identify usernames and crack passwords on Windows systems.

(iv) NetBIOS Auditing – It is another NetBIOS enumeration tool. It's used to perform various security checks on remote servers running NetBIOS file sharing services.

Q.17. What is SNMP enumeration ?

Ans. SNMP enumeration is the process of using SNMP to enumerate user accounts on a target system. SNMP employs two major types of software components for communication – the SNMP agent, which is located on the

networking device; and the SNMP management station, which communicates with the agent.

Almost all network infrastructure devices, such as routers and switches and including Windows systems, contain an SNMP agent to manage the system or device. The SNMP management station sends requests to agents, and the agents send back replies. The requests and replies refer to configuration variables accessible by agent software. Management stations can also send requests to set values for certain variables. Traps let the management station know that something significant has happened in the agent software such as a reboot or an interface failure. Management information base (MIB) is the database of configuration variables, which resides on the networking device.

SNMP has two passwords we can use to access and configure the SNMP agent from the management station. The first is called a read community string. This password lets us view the configuration of the device or system. The second is called the read/write community string, it's for changing or editing the configuration on the device. Generally, the default read community string is public and the default read/write community string is private. A common security loophole occurs when the community strings are left at the default settings – A hacker can use these default passwords to view or change the device configuration.

STEGANOGRAPHY, MERGE STREAMS, IMAGE HIDE, STEALTH FILES, BLINDSIDE USING – STOOLS, STEGHIDE, STEGANOS, STEGDETECT, STEGANALYSIS – STEGO WATCH – STEGO DETECTION TOOL, STEGSPY

Q.18. What is steganography ? How is it differs from cryptography ?
(R.G.P.V., May 2018)

Ans. A method which facilitates hiding of a message that is to be kept secret inside other messages is known as steganography. This outputs in the concealment of the secret message itself. Historically, the sender used procedures like invisible inks, minute variations between handwritten characters, pencil marks on handwritten characters, tiny pin punctures on specific characters, etc. Secret messages are hided by people within graphic images. For example, consider that user has a secret message to send. Another image file can be taken by user. User can replace the last two right most bits of each byte of that image with two bits of his secret message. The resulting image would not look too different as well as contain a secret message inside.

The opposite trick would be done by the receiver. Receiver would read the last two bits of each byte of the image file and recreate the secret message. Fig 5.2 represents this message.

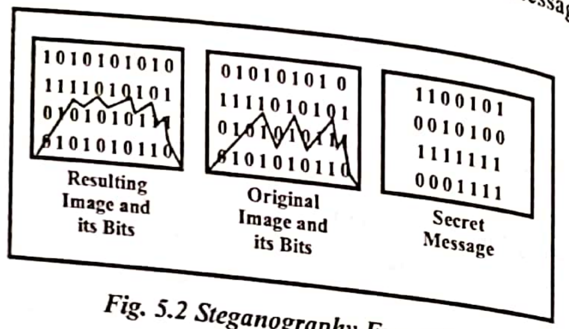


Fig. 5.2 Steganography Example

Difference between Steganography and Cryptography – Steganography is not actually a method of encrypting messages but hiding them within something else to enable them to pass undetected. Traditionally this was achieved with invisible ink, microfilm or taking the first letter from each word of a message. This is now achieved by hiding the message within a graphics or sound file. The methods of steganography conceal the existence of the message, whereas the methods of cryptography render the message unintelligible to outsiders by various transformations of the text.

Q.19. Discuss the various types of steganography.

Ans. Depending on the type of the cover object there are many suitable steganographic techniques which are in order to obtain security –

(i) Image Steganography – The process of concealing the secret message in an image file is known as image steganography. It has certain limitations like you cannot embed a large amount of data in an image because it may distort which may arise suspicion that the image might contain any information.

(ii) Video Steganography – The process of concealing the secret message in an video file is known as video steganography. Video steganography is far more safe and efficient as compared to that of the image steganography as you can embed large amount of data in audio and frames of the video.

(iii) Network Steganography – Network steganography method uses modification of a single network protocol. The protocol modification may be applied to the PDU (protocol data unit), time relations between exchanged PDUs, or both (hybrid methods). It is highly secure and robust.

(iv) Audio Steganography – In audio steganography audio is used as the cover to hide the secret information it is also very robust in nature but with limitation of the amount of data one can hide.

(v) Text Steganography – Secret data is hided in a text file. This method lacks robustness and is not that much efficient in hiding the data. It can be easily detected by the eyes of intruders.

Q.20. Describe the steganography measures.

Ans. The steganography measures are as follows –

(i) Imperceptibility – A steganographic process is imperceptible when human eye cannot distinguish between the cover image and the stego image.

(ii) Payload – It indicates the amount of secret information that can be embedded in the cover image. The embedding rate is given in absolute measurement such as the length of the secret message.

(iii) Statistical Attacks – The process of extracting the secret information from the stego object is known as statistical attack. The algo used for stegonagraphy must be robust to statistical attacks.

(iv) Security – Security of a steganographic system is defined in terms of undetectability, which is assured when the statistical tests cannot distinguish between the cover and the stego-image.

(v) Computational Cost – Data hiding and data retrieval are the two parameters used to figure computational cost of any steganography approach. Information concealing time alludes to the time required to implant information inside a cover video edge and information recovery alludes to extraction time of mystery message from the stego outline.

(vi) Perceptual Quality – Increasing the payload degrade the quality of the video so approach should be used such that the quality should remain intact to avoid it from getting in sight.

Q.21. What are the uses of steganography ?

Ans. The uses of steganography are as follows –

(i) Steganography can be a solution which makes it possible to send news and information without being censored and without the fear of the messages being intercepted and traced back to us.

(ii) It is also possible to simply use steganography to store information on a location. For example, several information sources like our private banking information, some military secrets, can be stored in a cover source. When we are required to unhide the secret information in our cover source, we can easily reveal our banking data and it will be impossible to prove the existence of the military secrets inside.

(iii) Steganography can also be used to implement watermarking. Although the concept of watermarking is not necessarily steganography, there are several steganographic techniques that are being used to store watermarks in data. The main difference is on intent, while the purpose of steganography is hiding information, watermarking is merely extending the cover source with extra information. Since people will not accept noticeable changes in images, audio or video files because of a watermark, steganographic methods can be used to hide this.

(iv) E-commerce allows for an interesting use of steganography. In current e-commerce transactions, most users are protected by a username and password, with no real method of verifying that the user is the actual card holder. Biometric finger print scanning, combined with unique session IDs embedded into the fingerprint images via steganography, allow for a very secure option to open e-commerce transaction verification.

(v) Paired with existing communication methods, steganography can be used to carry out hidden exchanges. Governments are interested in two types of hidden communications – those that support national security and those that do not. Digital steganography provides vast potential for both types. Businesses may have similar concerns regarding trade secrets or new product information.

(vi) The transportation of sensitive data is another key use of steganography. A potential problem with cryptography is that eavesdroppers know they have an encrypted message when they see one. Steganography allows to transport of sensitive data past eavesdroppers without them knowing any sensitive data has passed them. The idea of using steganography in data transportation can be applied to just about any data transportation method, from E-mail to images on Internet websites.

Q.22. What do you mean by merge streams ?

Ans. Merge streams shows you how to merge MS Word streams and MS Excel Workbook stream. It can hide MS Excel document inside MS Word document or vice a versa. If you wish to transparently hide some important documents inside old financial reports this is for you. It does not implement any crypto and is not secured enough, but is a smart trick.

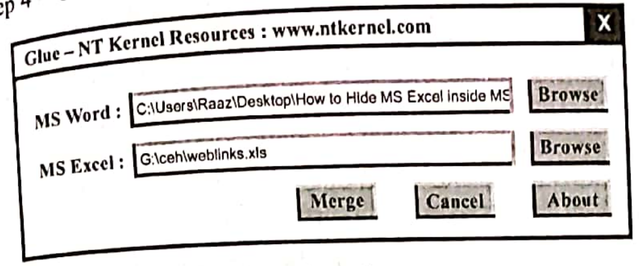
How to Hide

Step 1 – Open Merge Stream Software

Step 2 – Browse the MS Office File

Step 3 – Select the Excel which you want to hide inside MS Word

Step 4 – Click on Merge


Glue – NT Kernel Resources : www.ntkernel.com

MS Word : C:\Users\Raaz\Desktop\How to Hide MS Excel inside MS

MS Excel : G:\ceh\weblinks.xls

Fig. 5.3

Now, your file is been hidden and process completed.

How to Unhide

Right Click on Saved MS Office Document and on the option open with and then choose MS Office Excel.

Q.23. Discuss the steganography techniques for image hide.

Ans. Steganography techniques can be divided into following domains –

(i) Frequency Domain Technique – This is a more complex way of hiding information in an image various algorithms and transformations are used on the image to hide information in it frequency domain embedding can be termed as a domain of embedding techniques for which a number of algorithms have been suggested frequency domain are broadly classified into –

(a) Discrete Fourier Transformation Technique – The discrete Fourier transform to get frequency component for each pixel value. The discrete Fourier transform (DFT) of spatial value f(x, y) for the image of size M × N is defined in equation for frequency domain transformation.

(b) Discrete Cosine Transformation Technique – The discrete cosine transform (DCT) is a technique for converting a signal into elementary frequency components. It is widely used in image compression.

(c) Discrete Wavelet Transformation Technique – A discrete wavelet transform (DWT) is any wavelet transform for which the wavelets are discretely sampled.

(ii) Spatial Domain Methods – There are many versions of spatial steganography' a directly change some bits in the image pixel values in hiding data a directly change some bits in the image pixel values in hiding data. Spatial domain techniques are broadly classified into –

(a) Least Significant Bit – The least significant bit is the lowest bit in a series of numbers in binary the LSB is located at the far right of a string. For example, in the binary number – 10111001, the least significant bit is the far right 1. Here the secret information is stored in the LSB of the image.

(b) Pixel Value Differencing – The pixel-value differencing (PVD) scheme provides high imperceptibility to the stego image by selecting two consecutive pixels and designs a quantization range table to determine the payload by the difference value between the consecutive pixels.

(c) Edge Based Data Embedding Method – In ELSB, we use all the edge pixels in an image. Here, we first calculate the masked image by masking the two LSB bits in the cover image. Then we identify the edge pixels by using the Canny Edge detection method. After obtaining the edge pixels we hide the data in the LSB bits of the edge pixels only and send the stego object to the receiver.

(d) Random Pixel Embedding Method – Random pixels are used to embed and send the stego object to the receiver.

Q.24. Write short note on blindside.

Ans. Blindside is a steganography application that allows you to conceal a file, or set of files, within a single digital image. The resulting image appears identical to the human eye, but can typically contain around 50 kb of secret data. The concealed files can also be password protected. Blindside operates by creating slight colour inflections in an image which, although invisible to the human eye, can provide a great deal of space in which to store information. Blindside calculates the colour differentials between pixels and will modify the image only where it can be sure no one will notice. A proprietary cryptographic algorithm can also be used to scramble the data with a secret pass phrase.

Q.25. Define the following terms –

(i) S-Tools

(ii) Steghide

(iii) Steganos

(iv) Stegdetect

(v) StegoStick.

Ans. (i) S-Tools – The S-Tools spread a message over the whole carrier medium. S-Tools is a steganography tool that hides files in BMP, GIF, and WAV files. You open up a copy of S-Tools and drag pictures and sounds across to it. To hide files you just drag them over open sound/picture windows. You can hide multiple files in one sound/picture and your data is compressed before being encrypted then hidden. Multi-threaded operation means that you can have many hide/reveal operations going simultaneously without fear of them interferring with you or holding up your work. You can even close the original picture/sound with no ill effects to ongoing threads. Encryption services come courtesy of "cryptlib" by Peter Gutmann (and others).

(ii) Steghide – Steghide is a program able to hide data in various image and audio files like JPEG, BMP, WAV and AU developed by Stefan Hetzl. The colour respectively sample frequencies are not changed thus making the embedding resistant against first-order statistical tests. It can be used with batch files for hiding or revealing files.

(iii) Steganos – Steganos uses the carrier medium completely in every case. Steganos' advanced encryption options make hiding photos, documents and contact data easy an invaluable tool when you consider how many laptops are lost or stolen everyday. Hardware is replaceable. Your data is not.

Stego is a steganography tool that enables you to embed data in Macintosh PICT format files, without changing the appearance or size of the PICT file. Thus, Stego can be used as an "envelope" to hide a previously encrypted data file in a PICT file, making it much less likely to be detected.

(iv) Stegdetect – Stegdetect is a utility that analysis image files for steganographic content developed by Niels Provos. It runs statistical tests to determine if steganographic content is present, and also tries to find the system that has been used to embed the hidden information. This program is a direct compilation of the UNIX sources to a Windows binary. Out of the three other steganalysis tools tested this is the only one that can be executed from a batch file.

(v) StegoStick – This free and open source steganographic tool, lets you hide any file into any file. Is based on image, audio, video steganography that hides any file or message into an image (BMP, JPG, GIF), Audio/Video (MPG, WAV, etc.) or any other file format (PDF, EXE, CHM, etc.).

Q.26. Explain in detail about the steganalysis.

Ans. Steganalysis is "the process of detecting steganography by looking at variances between bit patterns and unusually large file sizes". It is the art of discovering and rendering useless convert messages. The goal of steganalysis is to identify suspected information streams, determine whether or not they have hidden messages encoded into them, and, if possible, recover the hidden information. Unlike cryptanalysis, where it is evident that intercepted encrypted data contains a message.

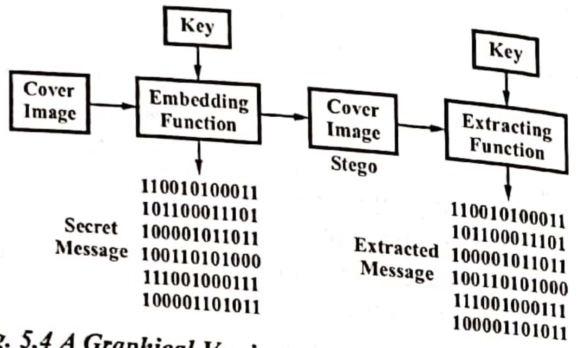


Fig. 5.4 A Graphical Version of the Steganographic System

Steganalysis generally starts with several suspect information streams but uncertainty whether any of these contain hidden message. The steganalyst starts by reducing the set of suspect information streams to a subset of most likely altered information streams. This is usually done with statistical analysis using advanced statistics techniques.

Steganalysis Techniques – Hiding information within an electronic medium cause alteration of the medium properties that can result in some form of degradation or unusual characteristics.

(i) Unusual Patterns – Unusual patterns in a stego image are suspecious. For example, there are some disk analysis utilities that can filter hidden information in unused partitions in storage devices. Filters can also be used to identify TCP/IP packets that contain hidden or invalid information in the packet headers. TCP/IP packets used to transport information across the Internet have unused or reserved space in the packet headers.

(ii) Visual Detection – Analyzing repetitive patterns may reveal the identification of a steganography tool or hidden information. To inspect these patterns an approach is to compare the original cover image with the stego image and note visible differences. This is called a known carrier attack. By comparing numerous images it is possible that patterns emerge as signatures to a steganography tool. Another visual clue to the presence of hidden information is padding or cropping of an image. With some stego tools if an image does not fit into a fixed size it is cropped or padded with black spaces.

There may also be a difference in the file size between the stego-image and the cover image. Another indicator is a large increase or decrease in the number of unique colours, or colours in a palette which increase incrementally rather than randomly.

(iii) Tools to Detect Steganography – The disabling or removal of hidden information in images is dependent on the image processing techniques. For example, with LSB methods of inserting data, simply compressing the image using lossy compression is enough to disable or remove the hidden message. There are several available steganographic detection tools such as Encase by Guidance Software Inc., ILook Investigator by electronic crimes program, Washington DC, various MD5 hashing utilities, etc.

Q.27. Write short note on StegSpy.

Ans. StegSpy is a program that is always in progress that allows identification of a "steganized" file, detecting steganography and the program used to hide the message developed by Spy-Hunter. Currently it identifies the following programs –

(i) Hiderman

(ii) JP hide and seek

(iii) Masker

(iv) JPegX

(v) Invisible secrets.

It is a window based application therefore it does not support batch file detection. It identifies the location of the hidden content as well.

TROJANS DETECTION TOOLS (I.E. NETSTAT, FPORT, TCPVIEW, CURRPORTS TOOLS, PROCESS VIEWER), LAN SCANNER TOOLS (I.E. LOOK@LAN, WIRESHARK, TCPDUMP)

Q.28. Write short note on Trojan horse.

Ans. A *Trojan* is a malicious program disguised as something benign. Trojans are often down-loaded along with another program or software package. Once installed on a system, they can cause data theft and loss, and system crashes or slowdowns; they can also be used as launching points for other attacks such as Distributed Denial of Service (DDOS). Many Trojans are used to manipulate files on the victim computer, manage processes, remotely run commands, intercept keystrokes, watch screen images, and restart or shut down infected hosts. Sophisticated Trojans can connect themselves to their originator or announce the Trojan infection on an Internet Relay Chat (IRC) channel. Table 5.2 lists some common Trojans and their default port numbers.

Table 5.2 Common Trojan Programs

Trojan	*Protocol*	*Port*
BackOrifice	UDP	31337 or 31338
Deep Throat	UDP	2140 and 3150
NetBus	TCP	12345 and 12346
Whack-a-mole	TCP	12361 and 12362
NetBus 2	TCP	20034
GirlFriend	TCP	21544
Masters Paradise	TCP	3129, 40421, 40422, 40423, and 40426

Trojans ride on the backs of other programs and are usually installed on a system without the user's knowledge. A Trojan can be sent to a victim system in many ways – as an Instant Messenger (IM) attachment, IRC, an e-mail attachment, or NerBIOS file sharing.

Q.29. How does a Trojan work ?

Ans. Trojans come in two parts, a Client part and a Server part. When the victim runs the server on its machine, the attacker will then use the Client to connect to the Server and start using the Trojan. TCP/IP protocol is the usual protocol type used for communications, but some functions of the Trojans use the UDP protocol as well. When the Server is being run on the victim's computer, it will (usually) try to hide somewhere on the computer, start listening on some port(s) for incoming connections from the attacker, modify the registry and/or use some other auto starting method.

It's necessary for the attacker to know the victim's IP address to connect to his/her machine. Many Trojans have features like mailing the victim's IP, as well as messaging the attacker via ICQ or IRC. This is used when the victim has dynamic IP which means every time you connect to the Internet you get a different IP (most of the dial-up users have this).

Most of the Trojans use Auto-Starting methods so even when you shut down your computer they're able to restart and again give the attacker access to your machine. New auto-starting methods and other tricks are discovered

all the time. The variety starts from "joining" the Trojan into some executable file you use very often like explorer.exe, for example, and goes to the known methods like modifying the system files or the Windows Registry. System files are located in the Windows directory.

Q.30. List the different types of Trojans.

Ans. Trojans can be created and used to perform different attacks. Some of the most common types of Trojans are –

(i) ***Remote Access Trojans (RATs)*** – Used to gain remote access to a system.

(ii) ***Data-sending Trojans*** – Used to find data on a system and deliver data to a hacker.

(iii) ***Destructive Trojans*** – Used to delete or corrupt files on a system.

(iv) ***Denial of Service Trojans*** – Used to launch a denial or service attack.

(v) ***Proxy Trojans*** – This is a type of trojan horse designed to use the victim's computer as a proxy server. This gives the attacker an opportunity to do everything from your computer, including the possibility of conducting credit card fraud and other illegal activities, or even to use your system to launch malicious attacks against other networks.

(vi) ***FTP Trojans*** – Allows the attacker to use someone else's computer as an FTP server. Installing this Trojan onto your computer would enable the intruder to download/upload files from his PC to yours, which could provide another avenue more installation of malware.

(vii) ***Security Software Disabler Trojans*** – Used to stop antivirus software.

Q.31. Discuss the various tools of Trojans.

Ans. A legitimate application that has been modified with malicious code. A Trojan horse is a social engineering technique. It masquerades as a legitimate download and injects the victim's host with an access point, or a client that can connect outbound to a server waiting remotely. They don't necessarily exploit a vulnerability unless privilege escalation is necessary. They provide a command environment for whoever connects to them that includes – File browsers, keyloggers, web cam viewer, and many additional tools.

Terms –

(i)	Wrapper or binder	– Application used to combine a malicious binary and a legitimate program.
(ii)	Rootkit	– Can be installed via Trojan, used to hide processes that create backdoor access.
(iii)	HTTP Trojan	– Reverses a connection outbound through an HTTP or SHTTP tunnel.
(iv)	Netcat	– Not really a Trojan, but often used in Trojan code to setup the listening socket.
(v)	Hoax	– Many legit tools are rumored to be Trojans but might not be.
(vi)	Keylogger	– Records the keystrokes on the install host and saves them in a log.

Famous Trojans Tools –

(i)	Tini	– Small 3 Kb file, uses port 7777.
(ii)	Loki	– Used ICMP as a tunneling protocol.
(iii)	Netbus	– One of the first RATs (Remote Authentication Trojan).
(iv)	Sub 7	– Written in Delphi, expanded on what Netbus had demonstrated.
(v)	Back orifice	– First modular malware, had the capabilities to be expanded on by outside authors.
(vi)	Beast	– All in one client/server binary.
(vii)	MoSucker	– Client could select the infection method for each binary.
(viii)	Nuclear RAT	– Reverse connecting Trojan.
(ix)	Monkey shell commands	– Provides a powerful shell environment that can reverse connections and encrypt.

Detecting Trojans Tools –

(i)	fport	– Command line tools for viewing open ports and connections.
(ii)	tcpview	– GUI tool for viewing open ports and connections.

(iii)	Process viewer	– GUI tool for showing open processes including child processes.
(iv)	Autoruns	– Lists all programs that will run on start up and where they are called from.
(v)	Hijack this	– Displays a list of unusual registry entries and files on the drive.
(vi)	Spybot S & D	– Originally volunteer supported scanning and detection tool.

Q.32. Write short notes on –
(i) NetSTAT
(ii) TCPView
(iii) CurrPorts.

Ans. (i) NetSTAT – It is a network based intrusion detection system that uses the state transition analysis technique. It operates on a high volume network. State transition analysis describes computer penetrations as a sequence of actions performed by an attacker to compromise a system. This system uses autonomous intrusion detection components known as probes. Probes are used to monitor network traffic and a filter module is used to select the messages that contribute assertions in a state transition scenario. If a single probe can detect all types of attacks, it does not interact with the analyzer. Otherwise, the analyzer decomposes an intrusion scenario into sub-scenarios so that each one can be detected by a single probe.

(ii) TCPView – TCPView is a Windows program that will show you detailed listings of all TCP and UDP endpoints on your system, including the local and remote addresses and state of TCP connections.

On Windows NT, 2000 and XP TCPView also reports the name of the process that owns the endpoint. TCPView provides a more informative and conveniently presented subset of the Netstat program that ships with Windows.

(iii) CurrPorts – CurrPorts displays the list of all currently opened TCP/IP and UDP ports on your local computer. For each port in the list, information about the process that opened the port is also displayed, including the process name, full path of the process, version information of the process (product name, file description, and so on), the time that the process was created, and the user that created it.

In addition, CurrPorts allows you to close unwanted TCP connections, kill the process that opened the ports, and save the TCP/UDP ports information to HTML file, XML file, or to tab-delimited text file.

CurrPorts also automatically mark with pink colour suspicious TCP/UDP ports owned by unidentified applications (Applications without version information and icons).

CurrPorts is a free software application from the Network Monitoring subcategory, part of the Network & Internet category. The app is currently available in English and it was last updated on 2020-07-28. The program can be installed on Win2000, Win7 ×32, Win7 ×64, Win98, WinVista, WinVista ×64, WinXP.

CurrPorts (version 2.62) has a file size of 97.91 KB and is available for download from our website. Just click the green Download button above to start. Until now the program was downloaded 420 times. We already checked that the download link to be safe, however for your own protection we recommended that you scan the downloaded software with your antivirus.

Q.33. Discuss the lan scanner tools.

Ans. Several tools exist that can monitor network traffic, usually such tools will put the network card of a computer into promiscuous mode, this enables the computer to listen to the entire traffic on that section of the network. Filtering of this packets can be done based on the IP related header data present in the packets, usually such filtering specifies simple criteria for the IP addresses and ports present in the packets. These passive network sniffing programs have been developed for either wired or wireless network measurement, the best-known are tcpdump and Wireshark.

(i) Look@Lan – Look@Lan is an advanced network monitor that allows you to monitor your net in few clicks. Extremely easy to use and very fast in discovering your network's active nodes. Full of relevant features such as auto-detect of network configuration, monitoring, reporting, trapping, statistics and graphs, network tree view, network log, proof single node scan, os detection. Main features of Look@Lan are as follows –

(a) Auto-detect of network settings

(b) Scanning of one or more scan-ranges

(c) Complete management of network profiles

(d) World's faster node discovery scan

(e) Automatic and manual network configuration

(f) Network statistics and graphs

(g) Profile export (text and HTML)

(h) Advanced trapping

(i) Network log

(j) Network tree view

(k) Proof single node scan

(*l*) Reporting.

(ii) Wireshark – This is a free and open-source packet analyzer and it is written in C. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named Ethereal, in May 2006 the project was renamed Wireshark due to trademark issues. Wireshark is very similar to tcpdump, but has a graphical front-end, plus some integrated sorting and filtering options.

Wireshark allows the user to put network interface controllers that support promiscuous mode into that mode, in order to see all traffic visible on that interface, not just traffic addressed to one of the interface's configured addresses and broadcast/multicast traffic. However, when capturing with a packet analyzer in promiscuous mode on a port on a network switch, not all of the traffic traveling through the switch will necessarily be sent to the port on which the capture is being done, so capturing in promiscuous mode will not necessarily be sufficient to see all traffic on the network. Port mirroring or various network taps extend capture to any point on net, simple passive taps are extremely resistant to malware tampering.

(iii) Tcpdump – This is a common packet analyzer that runs under the command line and parsing tool ported to several platforms. It allows the user to intercept and display TCP/IP and other packets being transmitted or received over a network to which the computer is attached. Tcpdump works by capturing and displaying packet headers and matching them against a set of criteria.

It runs on most UNIX-like operating systems – e.g. Linux, BSD, Solaris, Mac OS X, HP-UX and AIX amongst others making use of the libpcap library to capture packets.

DoS ATTACK UNDERSTANDING TOOLS – JOLT2, BUBONIC.C, LAND AND LATIERRA, TARGA, NEMESY BLAST, PANTHER2, CRAZY PINGER, SOME TROUBLE, UDP FLOOD, FSMAX

Q.34. What do you understand by DoS attacks ?

Ans. The common name employed for an attacker's interruption or disruption of the computing services of the victim is Denial of Service (DoS). A DoS attack attempts to prevent legitimate users from gaining access to network resources. It can take the form of flooding a network or server with traffic so that legitimate messages cannot get through or it can bring down a server. DoS attacks exhaust the computing power, memory capacity or communication bandwidth of their targets so they are rendered unavailable. One version of this attack causes website defacement. At various times, the websites of high-profile targets have been targeted. To prevent an alarm being raised, an advanced version of the DoS attack on a Web server, for example, does not necessarily paralyze it. Instead, it slows down the Web server so that its response time to requests from the outside world is unacceptably high.

Q.35. Write short note on DDoS.

Ans. A distributed DoS attack uses multiple source computers to disrupt its victims. This does not mean that the attack is coming from multiple attackers, however. The most typical architecture, in fact, is a single attacker or small group of attackers who trigger the attack by activating malware previously installed on computers throughout the world. A DDoS attack is harder to detect compared to a DoS attack. In a DDoS attack, the brain behind the attack (or controller) scans the Internet to find multiple vulnerable hosts called handlers and compromises them. Each handler, in turn, recruits many agents or zombies to launch the attack. Having multiple level of attackers means that more zombies can be co-opted thus amplifying the attack. The zombies are injected with the code that sends attack packets to the victim in a coordinated fashion to overwhelm it. Since the packets are coming from hundreds or thousands of machines, it is hard to distinguish these packets from packets coming from legitimate users. In addition, the source IP addresses are spoofed to obscure the source of the attacks.

Q.36. Discuss in brief the various types of DoS attacks.

Ans. Some of the DoS attacks are as follows –

(i) IP Spoofing – IP spoofing is forging of an IP packet address. In particular, a source address in the IP packet is forged. Since network routers use packet destination address to route packets in the network, the only time a source address is used is by the destination host to respond back to the source host. So forging the source IP address causes the responses to be misdirected, thus creating problems in the network. Many network attacks are a result of IP spoofing.

(ii) Smurf Attack – In this attack, the intruder sends a large number of spoofed ICMP Echo requests to broadcast IP addresses. Hosts on the broadcast multicast IP network, say, respond to these bogus requests with reply ICMP Echo. This may significantly multiply the reply ICMP Echos to the hosts with spoofed addresses.

(iii) Buffer Overflow Attack – In this attack, the attacker floods a carefully chosen field such as an address field with more characters than it can accommodate. These excessive characters, in malicious cases, are actually executable code, which the attacker can execute to cause havoc in the system, effectively giving the attacker control of the system. Since anyone with little knowledge of the system can use this type of attack, buffer overflow has become one of the most serious classes of security threats.

(iv) Ping of Death Attack – This vulnerability is used to hang remote systems so that no user could use its services. A system attacker sends IP packets that are larger than the 65,536 bytes allowed by the IP protocol. Many operating systems, including network operating systems, cannot handle these oversized packets, so they freeze and eventually crash.

(v) Teardrop Attack – The teardrop attack uses a program that causes fragmentation of a TCP packet. It exploits a reassembly and causes the victim system to crash or hang.

(vi) SYN Attack – The SYN attack exploits TCP/IPs three-way handshake. In a normal three-way handshake, the client sends a SYN packet to the host, the host replies to this packet with a SYN ACK packet. Then the client responds with a TCP ACK (acknowledgement).

Now, in a SYN attack too, several SYN packets are sent to the server but all these SYN packets have a bad source IP address. When the target systems receives these SYN packets with bad IP addresses, it tries to respond to each one of them with a SYN ACK packet. Now, the target system waits for a ACK message to come from the bad IP address. It queues up all these requests until it receives an ACK message. The requests are not removed unless and until the remote target system gets an ACK message. Hence, these requests take up or occupy valuable resources of the target machine.

To actually effect the target system, a large number of SYN bad IP packets have to be sent. Since these packets have a bad source IP, they queue up, use up resources and memory or the target system and eventually crash, hang or reboot the system.

A land attack is same as a SYN attack, the only difference being that instead of a bad IP address, the IP address of the target system is used.

(vii) SYN Flooding – A three-way handshake used by the TCP protocols to initiate a connection between two network elements. During the handshake, the port door is left half open. A SYN flooding attack is flooding the target system with so many connection requests coming from the spoofed source addresses that the victim server cannot complete because of the bogus source addresses. In the process all its memory gets hogged up and the victim is thus overwhelmed by these requests and can be brought down.

(viii) Sequence Number Sniffing – In this attack, the intruder takes advantage of the predictability of sequence numbers used in TCP implementations. The attacker then uses a sniffed text sequence number to establish legitimacy.

Q.37. Describe the various types of DoS attack tools.

Ans. The various types of DoS attack tools are as follows –

(i) Jolt – This DoS attack tool sends a large number of fragmented ICMP packets to a target machine running Windows 95 or NT in such a manner that the target machine fails to reassemble them for use, and as a result, it freezes up and cannot accept any input from the keyboard or mouse. However, this attack does not cause any significant damage to the victim system, and the machine can be recovered with a simple reboot.

Jolt 2 – This is a program that sends a large number of identical illegally fragmented ICMP echo or illegally fragmented UDP packet.

(ii) Burbonic – This DoS exploit attempts to victimize a Windows 2000 machine by sending a randomly large number of TCP packets with random settings with the purpose of increasing the load on the machines so that it leads to a crash.

(iii) Land and LaTierra – Land tool sends victim request by spoofing IP address of packet with IP address of victim. Since IP address of source and destination are same, system crashes as system starts flooding itself with packets.

LaTierra also works as Land tool but it sends TCP packets to more than one port number.

(iv) Targa – Targa is a collection of 16 different DoS attack programs. One can launch these attacks individually as well as in a group and can damage a network instantly.

(v) Nemsey – Nemsey is the DoS attack tool whose presence specifies the computer is insecure and infected with the malicious software. It is a GUI based attack tool that can deplete the bandwidth of the victim server. It does not generate the multiple sources and spoof the ip addresses. It attempts to launch an attack with a specified number of packets of specified sizes.

(vi) Blast – Blast is TCP services stress test tool but can also be used for launching DoS attack against unprotected server.

(vii) Blast20 – Blast20 is the DoS attack tool is called as the TCP service stress tool is able to identify the potential weaknesses in the network servers instantly.

It is command line based tool which has the ability to exhaust the resources of the victim server. The parameters required to launch attack are target IP address, start size and end size of the packet.

(viii) Panther – A UDP based DoS attack tool that can flood the specified IP at a particular port number. It takes IP address as the input parameter to launch the attack. This tool is the windows based. Panther has the ability to deplete the bandwidth of the victim server and can generate the traffic of UDP and TCP types. However, it is not so powerful attack tool.

Panther 2 – Panther 2 tool allows the attacker to perform a UDP based attack on a 28.8-56k connection. Also, this tool helps the attacker to crash the target server by flooding the server with too many connection requests.

(ix) Crazy Pinger – Crazy Pinger is the DoS attack tool which can launch attack by sending a large volume of ICMP packets to the victim machine or to the large remote network. Crazy Pinger is the GUI based attack tool that can spoof the ip addresses and can exhaust the resource and bandwidth. This kind of tool is easy to use and is effective over the multiple platforms.

(x) Some Trouble – This is a remote flooder. It is also a simple program with three remote function (a) mail bomb (b) Icq bomb (c) Netsend flood.

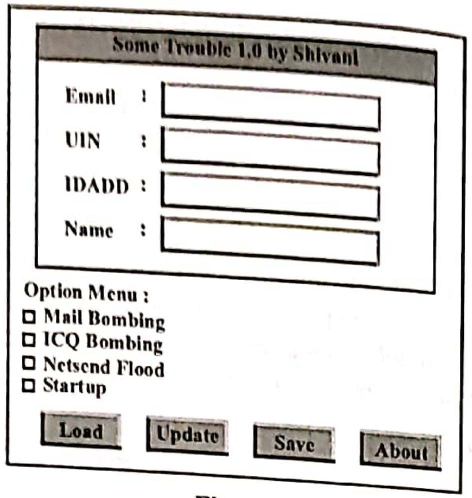


Fig. 5.5

(xi) UDP Flooder – UDP flooder is the port scanner and has the user friendly graphical user interface that can target the random ports and random packet size. It is the IRC-based attack tool which can also spoof the IP addresses of the source. It can deplete the bandwidth of the victim server in no time.

(xii) FSMax – FSMax is the DoS attack tool which can be used to test the stress of the network and to test the server to buffer overflows which may be exploited during attack, text file is accepted as the input which is executed through a sequence of tests based on the input. FSMax has the ability to exhaust the resources of the victim server.

●●●

RGPV

B.E. (Eighth Semester) EXAMINATION, June, 2011
(Information Technology Engg. Branch)
INFORMATION SECURITY
[IT - 801 (N)]

Note : Attempt any one question from each Unit.

Unit-I

1. (a) What are three basic operations in cryptography ? 10
(See Unit-I, Page 19, Q.17)

(b) What is hash function and what can it be used for ? 10
(See Unit-III, Page 132, Q.17)

Or

2. (a) How can a security framework assist in the design and implementation of a security infrastructure ? ** 10

(b) Explain the following – 10
(i) Confidentiality (ii) Integrity (iii) Availability
(See Unit-IV, Page 163, Q.1)

Unit-II

3. (a) What are six components of public key infrastructure (PKI) ? 10
(See Unit-II, Page 77, Q.16)

(b) What drawbacks to symmetric and asymmetric encryption are resolved by using a hybrid method like Diffie-Hellman ? 10
(See Unit-II, Page 78, Q.17)

Or

4. (a) What is the difference between digital signatures and digital certificates ? (See Unit-III, Page 146, Q.30) 10

(b) What is the fundamental difference between symmetric and asymmetric encryption ? (See Unit-II, Page 74, Q.10) 10

Unit-III

5. (a) IP sec can be used in two modes. What are they ? 10
(See Unit-IV, Page 208, Q.68)

(b) Using a modern pentium 4 computer, how long would it take to crack a cryptosystem that is based on a 32-bit key, 56-bit key, 64-bit key ? ** 10

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

Or

6. (a) Explain what is authentication and its types. ** 10

(b) What do you mean by transport layer security (TLS) ? Explain. 10
(See Unit-IV, Page 225, Q.82)

Unit-IV

7. (a) What security protocols are predominantly used in web-based electronic commerce ?
(See Unit-IV, Page 230, Q.86) 10

(b) Explain the terms phishing attacks, SQL injection attacks and format string attacks. ** 10

Or

8. (a) What security protocols are used to protect e-mail ? 10
(See Unit-IV, Page 196, Q.52)

(b) What is the most popular symmetric encryption system used over the web ?
(See Unit-I, Page 46, Q.36) 10

Unit-V

9. (a) What are web security problems ? Explain. (See Unit-IV, Page 214, Q.75) 10

(b) Explain intrusion detection system (IDS). (See Unit-IV, Page 187, Q.35) 10

Or

10. (a) What is the role of application level gateway ? 10
(See Unit-IV, Page 183, Q.28)

(b) Explain the functionality of firewalls. (See Unit-IV, Page 177, Q.19) 10

RGPV

B.E. (Eighth Semester) EXAMINATION, Dec., 2011
(Information Technology Engg. Branch)
INFORMATION SECURITY
(IT - 801)

Note : Attempt any one question from each Unit. Differentiate columnwise on the basis of properties, specified separately. Draw neat diagrams.

Unit-I

1. (a) Write the algorithm, draw the flowchart and also write program in C++ for Ceaser Cipher. ** 10

(b) Explain function of single round performed in each round of DES. 10
(See Unit-I, Page 49, Q.41)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

Or

2. (a) Differentiate between the following – 5 each

(i) Block Cipher and Stream Cipher (See Unit-I, Page 64, Q.58)

(ii) Diffusion and confusion (See Unit-I, Page 32, Q.33)

(b) Explain such block cipher modes of operation which use any encryption, and why they use only encryption. Draw complete and clear diagrams of each. (See Unit-I, Page 63, Q.55) 10

Unit-II

3. (a) Explain Euclidean algorithm. Solve the following using this algorithm –

(i) Determine gcd (24140, 16762) (ii) Determine gcd (4655, 12075)
(See Unit-I, Page 16 Prob.3) 10

(b) Explain Diffie-Hellman key exchange algorithm using flowchart and an example. (See Unit-II, Page 80, Q.20) 10

Or

4. (a) What are various requirements must be fulfilled by a Hash function ?
(See Unit-III, Page 135, Q.19) 10

(b) Find integer x such that – ** 10

(i) $5x \equiv 4 \pmod 3$ (ii) $7x \equiv 6 \pmod 5$

(iii) $9x \equiv 8 \pmod 7$ (iv) $3x \equiv 9 \pmod{10}$

Unit-III

5. (a) What four requirements were defined for Kerberos ? Explain. 10
(See Unit-III, Page 159, Q.43)

(b) What are the protocols that SSL comprised of ? Explain. 10
(See Unit-IV, Page 222, Q.80)

Or

6. (a) What are the services provided by the SSL record protocol ? 10
(See Unit-IV, Page 219, Q.79)

(b) What entities contitute a full service Kerberos environments ? 10
(See Unit-III, Page 157, Q.39)

Unit-IV

7. (a) Explain the steps involved in key generation in PGP. 10
(See Unit-IV, Page 199, Q.55)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

(b) What is role of compression and encryption in the operation of a virus? ** 10

Or

8. (a) How can we prevent CSSV attacks ? ** 10

(b) What are typical phases of operation of a virus or worm ? ** 10

Unit-V

9. (a) List four techniques used by firewalls to control access and enforce a security policy. (See Unit-IV, Page 178, Q.21) 10

(b) What metrices are useful for profile based intrusion detection ? What are benefits that can be provided by an intrusion detection system ? 10 (See Unit-IV, Page 194, Q.47)

Or

10.(a) Differentiate between the following – 10

(i) Statistical anomaly detection and rule based intrusion detection

(ii) Rule-based anomaly detection and rule based penetration identification (See Unit-IV, Page 194, Q.49)

(b) Write short notes on any two of the following – 10

(i) Application level gateway (See Unit-IV, Page 183, Q.28)

(ii) Cookies (See Unit-IV, Page 217, Q.76)

(iii) Secure HTTP

**

RGPV B.E. (Eighth Semester) EXAMINATION, June, 2012 (Information Technology Engg. Branch) INFORMATION SECURITY (IT - 801)

Note : Attempt any *one* question from each Unit. Draw neat diagrams. Differentiate columnwise on the basis of properties, specified separately.

Unit-I

1. (a) Write algorithm, draw flowchart and also write a program in C++ for one time pad cipher. 10

(b) Explain DES algorithm with the help of diagrams. 10 (See Unit-I, Page 44, Q.35)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

Or

2. (a) Explain such Block cipher modes of operation which use encryption and decryption. Draw complete and clear diagrams of each. 10 (See Unit-I, Page 63, Q.55)

(b) Differentiate between the following : 10

(i) Block cipher and stream cipher (See Unit-I, Page 64, Q.58)

(ii) Diffusion and confusion (See Unit-I, Page 32, Q.33)

Unit-II

3. (a) Write short notes on any *two* of the following : 10

(i) Hash value (See Unit-III, Page 132, Q.16)

(ii) Birthday attack (See Unit-III, Page 137, Q.21)

(iii) Meet-in-the-middle attacks (See Unit-II, Page 83, Q.21)

(b) Explain Euclidean algorithm and solve the following using above algorithm : 10

(i) determine gcd (1970, 1066) (ii) Determine gcd (24140, 16762) (See Unit-I, Page 17, Prob.4)

Or

4. (a) What characteristics are needed in a secure hash function ? 10 (See Unit-III, Page 135, Q.19)

(b) Explain RSA algorithm and using this algorithm encrypt the following :

(i) $P = 3, q = 11, e = 7, M = 5$ (ii) $P = 7, q = 11, e = 17, M = 8$ (See Unit-II, Page 101, Prob.10) 10

Unit-III

5. (a) What are the principal differences between version 4 and version 5 of kerberos ? (See Unit-III, Page 160, Q.44) 10

(b) List and briefly define the parameters that define an SSL session state. (See Unit-IV, Page 219, Q.78) 10

Or

6. (a) What is the difference between tunnel mode and transport mode ? 10 (See Unit-IV, Page 209, Q.69)

(b) What services are provided by IPsec ? (See Unit-IV, Page 205, Q.64) 10

Unit-IV

7. (a) Draw generic transmission diagram in PGP and explain in brief. 10 (See Unit-IV, Page 198, Q.54)

(b) How does a worm propagate ?

Or

** 10

8. (a) What sort of testing can be performed in order to guard against possible CSS attacks ? ** 10

(b) What is the role of compression in the operation of a virus ? ** 10

Unit-V

9. (a) What are *three* benefits that can be provided by an intrusion detection system ? (See Unit-IV, Page 189, Q.39) 10

(b) What are the weaknesses of a packet filtering router ? Discuss its solution. (See Unit-IV, Page 183, Q.30) 10

Or

10.(a) Explain the working of application level gateway. 10
(See Unit-IV, Page 183, Q.28)

(b) Specify and explain classes of intruders. (See Unit-IV, Page 189, Q.37) 10

RGPV

B.E. (Eighth Semester) EXAMINATION, June, 2013
(Information Technology Engg. Branch)
INFORMATION SECURITY
(IT - 801)

Note : All questions carry equal marks. Attempt any question from internal choice.

Unit-I

1. (a) What is the difference between passive and active security threats ? 10 (See Unit-IV, Page 169, Q.7)

(b) How many keys are required for two people to communicate via a cipher ? (See Unit-I, Page 20, Q.19) 10

Or

2. (a) Write a program that can encrypt and decrypt using general caesar cipher also known as additive cipher.(See Unit-I, Page 42, Prob.10) 10

(b) Which parameters and design choices determine the actual algorithm of a Feistel cipher ? What is the purpose of the s-boxes in DES ? 10 (See Unit-I, Page 51, Q.43)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

Unit-II

3. (a) What is the difference between modular arithmetic and ordinary arithmetic ? List three classes of polynomial arithmetic. 10 (See Unit-I, Page 9, Q.9)

(b) What are the broad categories of applications of public-key crypto-systems ? (See Unit-II, Page 75, Q.13) 10

Or

4. (a) What is an elliptic curve and what is zero point on elliptic curve ? 10 (See Unit-II, Page 103, Q.39)

(b) Explain digital signature standards in brief. (See Unit-III, Page 143, Q.27) 10

Unit-III

5. (a) What are the problems associated with clean text passwords ? 10 **

(b) How does one prevent the misuse of another user's certificate in certificate based authentication ? ** 10

Or

6. (a) Explain the security handshake pitfalls. ** 10

(b) What is Kerberos ? How does Kerberos work ? 10 (See Unit-III, Page 154, Q.37)

Unit-IV

7. (a) Explain SQL injection. Why is this for Web attack only ? ** 10

(b) How is a circuit gateway different from application gateway ? 10 (See Unit-IV, Page 183, Q.29)

Or

8. (a) What is phishing ? How to avoid phishing attacks ? ** 10

(b) What are the firewalls ? How firewall guards corporate networks ?10 (See Unit-IV, Page 175, Q.14)

Unit-V

9. (a) What are the hardware and software requirements ? Classify them into various cryptographic services. (See Unit-II, Page 110, Q.44) 10

(b) Give difference among viruses, worms and malwares. **10

Or

10.(a) Why would leased line as a better approach than VPN ? 10 (See Unit-IV, Page 190, Q.41)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

(b) List the characteristics of a good firewall implementation. 10

(See Unit-IV, Page 176, Q.15)

B.E. (Eighth Semester) EXAMINATION, June, 2014
(Information Technology Engg. Branch)
INFORMATION SECURITY
(IT - 801)

Note : (i) There are ten questions with internal choice.

(ii) Attempt any five from them.

(iii) Assume missing data (if any). All questions carry equal marks.

Unit-I

1. (a) Why is confidentiality an important principle of security ?

(See Unit-IV, Page 166, Q.2)

(b) What is a worm ? What is the significant differences between a worm and a virus ?

(See Unit-IV, Page 169, Q.6)

Or

2. (a) What is plaintext ? Why is monoalphabetic cipher difficult to crack ?

(See Unit-I, Page 30, Q.26)

(b) Distinguish between symmetric and asymmetric key cryptography ?

(See Unit-II, Page 74, Q.10)

Unit-II

3. (a) What is an initialization vector (IV) ? What is its significance ?

(See Unit-I, Page 53, Q.47)

(b) What is the important aspect that establishes trust in digital signatures?

(See Unit-III, Page 121, Q.6)

Or

4. (a) "Digital envelopes combine the best features of symmetric and asymmetric key cryptography." Explain it. Why ?

(See Unit-III, Page 124, Q.10)

(b) Give the main differences between RSA-algorithm and Elliptic Curve Cryptography (ECC) ? (See Unit-II, Page 109, Q.42)

Unit-III

5. (a) What is idea behind certification authority hierarchy ?

(See Unit-III, Page 147, Q.31)

Why is self signed certificate needed ? **

(b) Why is the SSL layer positioned between the application layer and the transport layer ? (See Unit-IV, Page 225, Q.81)

Or

6. (a) How does one prevent the misuse of another user's certificate in certificate-based authentication ? **

(b) What is kerberos? How does kerberos work ?

(See Unit-III, Page 154, Q.37)

Unit-IV

7. (a) Why are some attacks called as passive ? Why are other attacks called active ? (See Unit-IV, Page 169, Q.7)

(b) Think and write about offering phishing prevention techniques. Which one of them would be most effective and why. **

Or

8. (a) What are the two main attacks on corporate networks ?

(See Unit-IV, Page 176, Q.17)

(b) How can you describe SQL injection attacks ? What are the techniques to prevent them ? How can we over come it ? **

Unit-V

9. (a) What are the limitations of a firewall ? (See Unit-IV, Page 177, Q.18)

What is significance of tunnel mode ? **

(b) How is screened host firewall, dual-homed bastion different from screened host firewall, single homed bastion ? (See Unit-IV, Page 187, Q.33)

Or

10. (a) Discuss the concept of a cookie. (See Unit-IV, Page 217, Q.76)

How can cookies damage privacy?

(b) What is the role of audit records in intrusion detection ? Explain in detail ? **

B.E. (Eighth Semester) EXAMINATION, June, 2015
(Information Technology Engg. Branch)
INFORMATION SECURITY
(IT - 801)

Note : (i) Attempt all questions with carry equal marks.

(ii) Each unit have a internal choice.

Unit-I

1. (a) Compare output feedback mode with cipher feedback mode.

(See Unit-I, Page 59, Q.52)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

(b) Explain the basic principles of information security. (See Unit-IV, Page 163, Q.1)

Or

2. (a) With the help of a block diagram explain DES encryption algorithm. (See Unit-I, Page 44, Q.35)

(b) Discuss the various types of cryptanalysis attacks. (See Unit-III, Page 150, Q.33)

Unit-II

3. (a) Write the difference between conventional encryption and public key encryption. (See Unit-II, Page 74, Q.10)

(b) Write a short note on RSA. (See Unit-II, Page 90, Q.29)

Or

4. (a) Explain hash function in detail. (See Unit-III, Page 132, Q.16)

(b) Describe Diffie-Hellman key exchange algorithm. (See Unit-II, Page 80, Q.20)

Unit-III

5. (a) Give a overview of transport mode and tunnel mode. (See Unit-IV, Page 208, Q.68)

(b) Explain secure socket layer. (See Unit-IV, Page 217, Q.77)

Or

6. (a) Discuss various alert codes of TLS ? (See Unit-IV, Page 228, Q.84)

(b) Explain IP security. (See Unit-IV, Page 201, Q.59)

Unit-IV

7. Explain cross site scripting and phishing attacks ? How can you overcome to cross site scripting ? **

Or

8. Explain various types of software threats in detail. (See Unit-IV, Page 167, Q.5)

Unit-V

9. Write a short notes (any four) –

(a) Intrusion detection (See Unit-IV, Page 187, Q.35)

(b) Packet filters (See Unit-IV, Page 183, Q.27)

(c) URL **

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

(d) Web security problem (See Unit-IV, Page 214, Q.75)

(e) Cookies. (See Unit-IV, Page 217, Q.76)

RGPV

IT-801
B.E. (Eighth Semester) EXAMINATION, June 2016
INFORMATION SECURITY

Note : (i) Answer five questions. In each question part A, B, C is compulsory and D part has internal choice.

(ii) All parts of each questions are to be attempted at one place.

(ii) All questions carry equal marks, out of which part A and B (Max. 50 words) carry 2 marks, part C (Max. 100 words) carry 3 marks, part D (Max. 400 words) carry 7 marks.

(iv) Except numericals, Derivation, Design and Drawing etc.

1. (a) Write any two difference between diffusion and confusion. (See Unit-I, Page 33, Q.34)

(b) What is the purpose of the S-boxes in DES ? (See Unit-I, Page 51, Q.42)

(c) Explain the avalanche effect. (See Unit-I, Page 52, Q.45)

(d) Define playfair cipher and polyalphabetic cipher with suitable example. (See Unit-I, Page 30, Q.28)

Or

Encrypt the message "Cryptography" using the hill cipher with key $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$. Show your calculation. (See Unit-I, Page 37, Prob.7)

2. (a) What is Euler's totient ? (See Unit-I, Page 13, Q.13)

(b) What are the three broad categories of application of public key cryptosystems ? (See Unit-II, Page 75, Q.13)

(c) Explain factoring problem in RSA. (See Unit-II, Page 93, Q.31)

(d) Users A and B use the Diffie-Hellman key exchange technique a common prime $q = 71$ and a primitive root $\alpha = 7$.

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

(ii) If user B has private key $X_B = 12$, what is B's public key Y_B ? (See Unit-II, Page 90, Prob.4)

Or

Explain elliptic curve cryptography with suitable example.

(See Unit-II, Page 105, Q.40)

3. (a) In the context of Kerberos, What is realm ?

(See Unit-III, Page 158, Q.40)

(b) Write any two difference between Kerberos 4 and Kerberos 5 ?

(See Unit-III, Page 161, Q.45)

(c) What is chain of certificate ?

(d) Explain secure socket layer and transport layer security. **

(See Unit-IV, Page 228, Q.83)

Or

Discuss IP security in detail. (See Unit-IV, Page 201, Q.59)

4. (a) List different type of phishing attack.

(b) List different types of viruses. **

(c) Differentiate between viruses and worms. **

(See Unit-IV, Page 169, Q.6)

(d) Define following term –

(i) Format string

(ii) SQL injection attack. **

Or

Define E-mail security in detail. Why E-mail security is important ?

(See Unit-IV, Page 195, Q.50)

5. (a) Why access control is more important in security ?

(See Unit-IV, Page 179, Q.23)

(b) Define uniform resource locator. **

(c) Write difference between HTTP and HTTPS. **

(d) What is firewall ? List the type of firewalls and explain. Draw a schematic diagram of a packet filtering router used as a firewalls.

(See Unit-IV, Page 183, Q.26)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

Or

Write a short notes –

(i) Encrypted tunnel (See Unit-IV, Page 188, Q.36)

(ii) IDS (See Unit-IV, Page 187, Q.35)

RGPV

IT-801

B.E. (Eighth Semester) EXAMINATION, June 2017

INFORMATION SECURITY

Note : (i) Eight questions are there.

(ii) Attempt five questions.

(ii) All questions carry equal marks.

1. (a) Define security. What are multiple layers of security ? ** 7

(b) Describe the critical characteristics of information. How are they used in computer security ? (See Unit-IV, Page 163, Q.1) 7

2. (a) How many keys are required for two parties to communicate via a cipher ? Why ? (See Unit-I, Page 20, Q.19) 7

(b) Explain playfair cipher with suitable example. 7

(See Unit-I, Page 30, Q.27)

3. (a) Why is the middle portion of 3DES a decryption rather than an encryption ? (See Unit-I, Page 47, Q.40) 7

(b) Explain Elliptic curve cryptography and its applications. 7

(See Unit-II, Page 107, Q.41)

4. (a) What types of attacks are addressed by message authentication ? 7

(See Unit-III, Page 119, Q.4)

(b) What basic arithmetical and logical functions are used in whirlpool ? ** 7

5. (a) What entities constitutes a full-service Kerberos environment ? 7

(See Unit-III, Page 157, Q.39)

(b) Why does PGP generate a signature before applying compression ? 7

(See Unit-IV, Page 199, Q.56)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

6. (a) What is difference between transport mode and tunnel mode ? 7
(See Unit-IV, Page 209, Q.69)

(b) Give differences among Viruses, Worms and Malware. **7

7. (a) What is HTTP ? How HTTP differ from HTTPS ? **7

(b) List and briefly define three classes of Intruders. What is honey pot ? (See Unit-IV, Page 189, Q.38) 7

8. Write short notes on any two – 14

(a) Firewalls (See Unit-IV, Page 175, Q.14)

(b) Eavesdropping **

(c) Diffie-Hellman key exchange (See Unit-II, Page 80, Q.20)

(d) SQL Injection. **

IT-801 (GS)
B.E. (Eighth Semester) EXAMINATION, May 2018
Grading System (GS)
INFORMATION SECURITY

RGPV

Note : (i) Attempt any five questions.

(ii) All questions carry equal marks.

1. (a) Differentiate substitution and transposition ciphers with suitable examples. (See Unit-I, Page 32, Q.31)

(b) Define the term cryptanalysis. Explain linear and differential cryptanalysis. (See Unit-III, Page 148, Q.32)

2. (a) What is the purpose of the S-boxes in DES ? Explain the avalanche effect. (See Unit-I, Page 52, Q.44)

(b) What is steganography ? How is it differs from cryptography ?
(See Unit-V, Page 245, Q.18)

3. (a) Briefly discuss Diffie-Hellman key exchange scheme.
(See Unit-II, Page 80, Q.20)

(b) What is hash function ? Give the basic uses of hash function.
(See Unit-III, Page 132, Q.17)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

4. (a) Explain digital signature with arbitrated and direct approaches.
(See Unit-III, Page 122, Q.8)

(b) What was Kerberos designed for ? Explain the architecture of Kerberos. (See Unit-III, Page 154, Q.37)

5. Define following attacks in detail –

(a) SQL injection attack

(b) Phishing attack

(c) Ransomware attack. **

6. (a) Define virus, intruders, worms. Also write the basic principle of intrusion detection system. (See Unit-IV, Page 189, Q.40)

(b) What is the use of firewall ? Explain firewall design principles.
(See Unit-IV, Page 178, Q.20)

7. (a) Define the terms Integrity, Confidentiality, Denial of Service and Authentication. (See Unit-IV, Page 166, Q.3)

(b) Explain architecture of Secure Socket Layer.
(See Unit-IV, Page 217, Q.77)

8. Write short notes on any two –

(a) IP spoofing (See Unit-V, Page 231, Q.2)

(b) Brute force attack (See Unit-III, Page 153, Q.34)

(c) Strength of DES (See Unit-I, Page 46, Q.39)

(d) RSA encryption algorithm. (See Unit-II, Page 90, Q.29)

IT-8001 (CBGS)
B.E. VIII Semester
EXAMINATION, May 2019
Choice Based Grading System (CBGS)
INFORMATION SECURITY

RGPV

Note : (I) Attempt any five questions.

(II) All questions carry equal marks.

1. (a) List and briefly define types of cryptanalytic attacks based on what is known to the attacker ? (See Unit-III, Page 153, Q.35)

(b) Briefly define the playfair cipher with taking a suitable example.
(See Unit-I, Page 30, Q.27)

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus

2. Encrypt the message "meet at the airport" using the Hill cipher with the key $\begin{pmatrix} 9 & 4 \\ 5 & 7 \end{pmatrix}$. Show your calculation and the result.

(See Unit-I, Page 40, Prob.9)

3. (a) What is the primitive root of a number ?

[See Unit-II, Page 88, Prob. 1 (i)]

(b) What are the three board categories of applications of public-key-cryptosystems ? (See Unit-II, Page 75, Q.13)

4. Perform the encryption and decryption using RSA alogirthm

(i) $p = 3; q = 11; e = 7; m = 5$

(ii) $p = 11; q = 13; e = 17; m = 8$.

(See Unit-II, Page 103, Prob.12)

5. (a) Explain the concpet of kerberos ? How is it useful ?

(See Unit-III, Page 157, Q.38)

(b) Explain the internet key exchange protocol.

(See Unit-IV, Page 213, Q.73)

6. Explain the phishing and format string attack. Explain with tacking suitable example. **

7. (a) What is penetration testing ? **

(b) What is firewall and its types ? (See Unit-IV, Page 183, Q.25)

8. Write a short notes (any three) –

(i) Intrusion detection system (See Unit-IV, Page 187, Q.35)

(ii) Email security (See Unit-IV, Page 196, Q.51)

(iii) Socket secure layer (See Unit-IV, Page 217, Q.77)

(iv) Web Security and cookies. (See Unit-IV, Page 217, Q.76)

❖❖❖

**Now, according to new revised syllabus of R.G.P.V., it is not included in syllabus