Lecture Notes

Sub: Error Control Coding and Cryptography Faculty: S Agrawal

1st Semester M.Tech, ETC (CSE)

Module-I: (10 Hours)

Reed Solomon Codes – Reed-Solomon Error Probability, Why R-S codes perform well against burst noise, R-S performance as a function of size, redundancy and code rate. Interleaving and Concatenated Codes- Block interleaving, Convolutional Interleaving, Concatenated Codes, Coding and Interleaving Applied to the Compact Disc, Digital Audio Systems- CIRC encoding, CIRC decoding, Interpolation and muting. Turbo Codes- Turbo code Concepts, log-likelihood Algebra

Module-II: (10 Hours)

Modulation & Coding Trade Offs: Goals of the Communications System Designer, Error Probability Plane, Nyquist Minimum Bandwidth, Shannon-Hartley Capacity Theorem, Bandwidth Efficiency Plane, Modulation and Coding Trade-Offs, Defining, Designing, and Evaluating Digital Communication Systems, Bandwidth Efficient modulation, Modulation and Coding for Bandlimited Channels, Trellis-Coded Modulation.

Module-III: (Selected portions from Text Book 3) (10 Hours)

Introduction to Security and Cryptographic Techniques: Introduction, Security Goals, Services and Mechanisms, Techniques (1.1-1.4), Traditional Symmetric Key Ciphers (3.1-3.4), Modern Symmetric Key Ciphers (5.1-5.2).

Brief idea about Data Encryption Standard (DES) (6.1-6.5), International Data Encryption Algorithm (DEA) and Advanced Encryption Standard (AES) (7.1-7.2), Encipherment using Modern Symmetric Key Ciphers (8.1-8.3), Asymmetric Key Cryptography (10.1-10.4).

Module-IV: (10 Hours)

Message Integrity(11.1), Message Authentication(11.3), Hash Function(12.1,12.2,12.4), Digital Signature(13.1-13.4), Entity Authentication(14.1-14.3,14.5), Key Management(15.1-15.5), Security in Email, PGP, S/MIME(16.1-16.3), Brief idea on Transport layer (17.1-17.2) and Network layer security(18.1-18.2), System security(19.4-19.8).

Text Books:

1. Digital Communication-Fundamental Application by Bernard Sklar, 2nd Edition of Pearson education Publication for Module-I and II.
2. B.Vucentic & J.Yuan, Turbo codes, Kluwer, 2000 for Module-I and II.
3. Cryptography and Network Security, B.A. Forouzan & D. Mukhopadhyay, (2/e), McGraw-Hill Publication, 2012. (Module III and IV).
4. S.Lin & D.J.Costello, Error Control Coding (2/e), Pearson, 2005.

Reference Books:

1. C.B.Schlegel & L.C.Perez, Trellis and Turbo Coding Wiley, 2004.
2. S. Gravano, Introduction to Error Control Codes, Oxford Pubs, 2001.
3. Information Theory, Coding and Cryptography by Ranjan Bose, TMH Publication.
4. Cryptography and Network Security" by A. Kahate, TMH Publication

THE WRITER IS NOT RESPONSIBLE FOR ANY LEGAL ISSUES ARISING OUT OF ANY COPYRIGHT DEMANDS AND/OR REPRINT ISSUES CONTAINED IN THIS MATERIAL.

THIS IS NOT MEANT FOR ANY COMMERCIAL PURPOSE. THIS IS SOLELY MEANT FOR PERSONAL REFERENCE OF STUDENTS FOLLOWING THE SYLLABUS PRESCRIBED BY THE UNIVERSITY.

READERS ARE REQUESTED TO SEND ANY TYPING ERRORS CONTAINED, HEREIN.

ACKNOWLEDGMENT

Different sources used in the preparation of this material are:

1. Digital Communications - Fundamentals and Applications - Bernard Sklar, 2nd Edition, Person Education Publication.
2. Information Theory, Coding & Cryptography - Ranjan Bose, TMH Publication.
3. Digital Communications – Simon Haykin, Wiley Edition.
4. Digital Communications - J.G.Proakis, 3rd Edition, McGrawHill Publications.

MODULE I: Reed Solomon Codes

INTRODUCTION

In 1960, Irving Reed and Gus Solomon published a paper in the Journal of the Society for Industrial and Applied Mathematics. The paper described a new class of error-correcting codes that are now called Reed-Solomon (R-S) codes. These codes have great power and utility, and are today found in many applications from compact disc players to deep-space applications.

Reed-Solomon codes are *nonbinary cyclic* codes with symbols made up of m-bit sequences, where m is any positive integer having a value greater than 2. R-S (n, k) codes on m-bit symbols exist for all n and k for which

$$0 < k < n < 2^m + 2 \qquad (1)$$

where k is the number of data symbols being encoded, and n is the total number of code symbols in the encoded block. For the most conventional R-S (n, k) code,

$$(n, k) = (2^m - 1, 2^m - 1 - 2t) \qquad (2)$$

where t is the symbol-error correcting capability of the code, and $n - k = 2t$ is the number of parity symbols. An extended R-S code can be made up with $n = 2^m$ or $n = 2^m + 1$, but not any further.

Reed-Solomon codes achieve the *largest possible* code minimum distance for any linear code with the same encoder input and output block lengths. For nonbinary codes, the distance between two code words is defined (analogous to Hamming distance) as the number of symbols in which the sequences differ. For Reed- Solomon codes, the code minimum distance is given by

$$d_{\text{min}} = n - k + 1 \qquad (3)$$

The code is capable of correcting any combination of t or fewer errors, where t can be expressed as

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor = \left\lfloor \frac{n-k}{2} \right\rfloor \qquad (4)$$

where $\lfloor x \rfloor$ means the largest integer not to exceed x. Equation (4) illustrates that for the case of R-S codes, correcting t symbol errors requires no more than $2t$ parity symbols. Equation (4) lends itself to the following intuitive reasoning. One can say that the decoder has $n - k$ redundant symbols to "spend," which is twice the amount of correctable errors. For each error, one redundant symbol is used to locate the error, and another redundant symbol is used to find its correct value.

The erasure-correcting capability, ρ, of the code is

$$\rho = d_{\text{min}} - 1 = n - k \qquad (5)$$

Simultaneous error-correction and erasure-correction capability can be expressed as follows:

$$2\alpha + \gamma < d_{\text{min}} < n - k \qquad (6)$$

where α is the number of symbol-error patterns that can be corrected and γ is the number of symbol erasure patterns that can be corrected. An advantage of nonbinary codes such as a Reed-Solomon code can be seen by the following comparison. Consider a binary $(n, k) = (7, 3)$ code. The entire n-tuple space contains $2^n = 2^7 = 128$ n-tuples, of which $2^k = 2^3 = 8$ (or 1/16 of the n-tuples) are codewords. Next, consider a *nonbinary* $(n, k) = (7, 3)$ code where each symbol is composed of $m = 3$ bits. The n-tuple space amounts to $2^{nm} = 2^{21} = 2{,}097{,}152$ n-tuples, of which $2^{km} = 2^9 = 512$ (or 1/4096 of the n-tuples) are codewords. When dealing with nonbinary symbols, each made up of m bits, only a small fraction (i.e., 2^{km} of the large number 2^{nm}) of possible n-tuples are codewords. This fraction decreases with increasing values of m. The important point here is that when a small fraction of the n-tuple space is used for codewords, a large d_{min} can be created.

Any linear code is capable of correcting $n - k$ symbol erasure patterns if the $n - k$ erased symbols all happen to lie on the parity symbols. However, R-S codes have the remarkable property that they are able to correct *any* set of $n - k$ symbol erasures within the block. R-S codes can be designed to have any redundancy. However, the complexity of a high-speed implementation increases with redundancy. Thus, the most attractive R-S codes have high code rates (low redundancy).

REED-SOLOMON ERROR PROBABILITY

The Reed-Solomon (R-S) codes are particularly useful for *burst-error correction*; that is, they are effective for channels that have memory. Also, they can be used efficiently on channels where the set of input symbols is large. An interesting feature of the R-S code is that as many as two information symbols can be added to an R-S code of length n without reducing its minimum distance. This extended R-S code has length $n + 2$ and the same number of parity check symbols as the original code. The R-S decoded symbol-error probability, P_E, in terms of the channel symbol-error probability, p, can be written as follows:

$$P_E \approx \frac{1}{2^m - 1} \sum_{j=t+1}^{2^m-1} j \binom{2^m - 1}{j} p^j (1-p)^{2^m-1-j} \tag{7}$$

where t is the symbol-error correcting capability of the code, and the symbols are made up of m bits each.

The bit-error probability can be upper bounded by the symbol-error probability for specific modulation types. For MFSK modulation with $M = 2^m$, the relationship between P_B and P_E is as follows:

$$\frac{P_B}{P_E} = \frac{2^{m-1}}{2^m - 1} \tag{8}$$

Figure 1 shows P_B versus the channel symbol-error probability p, plotted from Equations (7) and (8) for various (t-error-correcting 32-ary orthogonal Reed- Solomon codes with $n = 31$ (thirty-one 5-bit symbols per code block).

Figure 2 shows P_B versus E_b/N_0 for such a coded system using 32-ary MFSK modulation and noncoherent demodulation over an AWGN channel. For R-S codes, error probability is an exponentially decreasing function of block length, n, and decoding complexity is proportional to a small power of the block length. The R-S codes are sometimes used in a concatenated arrangement. In such a system, an inner convolutional decoder first provides some error control by operating on soft-decision demodulator outputs; the convolutional decoder then presents hard-decision data to the outer Reed-Solomon decoder, which further reduces the probability of error.

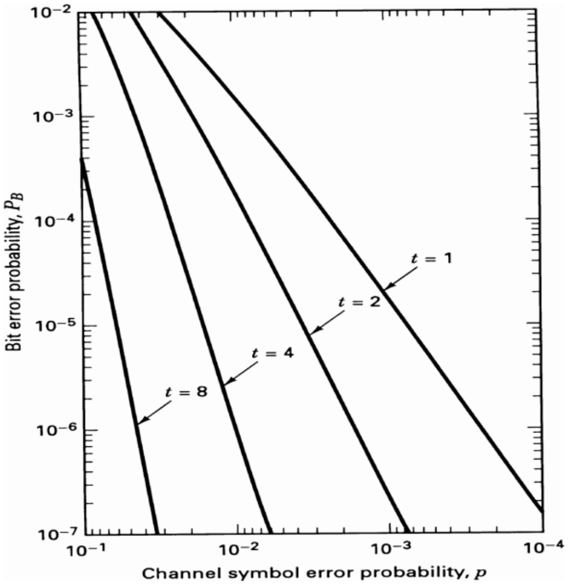


Figure 1

P_B versus p for 32-ary orthogonal signaling and n = 31, t-error correcting Reed-Solomon coding.

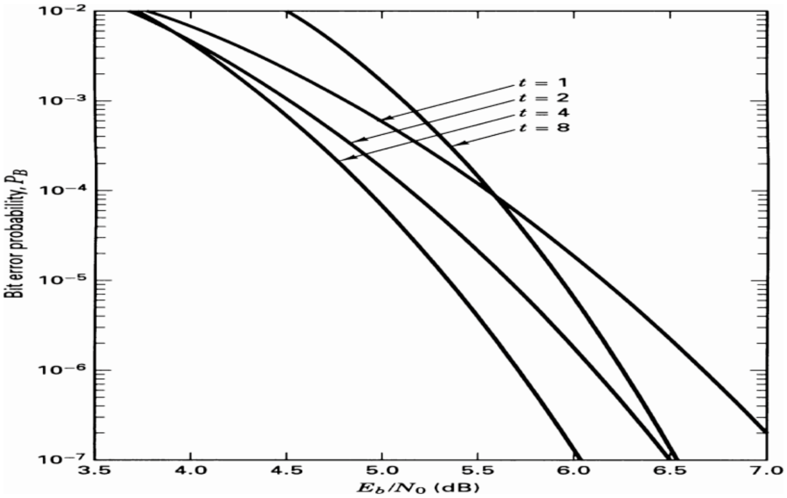


Figure 2

Bit-error probability versus E_b/N_0 performance of several n = 31, t-error correcting Reed-Solomon coding systems with 32-ary MPSK modulation over an AWGN channel.

WHY R-S CODES PERFORM WELL AGAINST BURST NOISE

Consider an (n, k) = (255, 247) R-S code, where each symbol is made up of m = 8 bits (such symbols are typically referred to as *bytes*). Since n - k = 8, Equation (4) indicates that this code can correct any four symbol errors in a block of 255. Imagine the presence of a noise burst, lasting for 25-bit durations and disturbing one block of data during transmission, as illustrated in Figure 3.

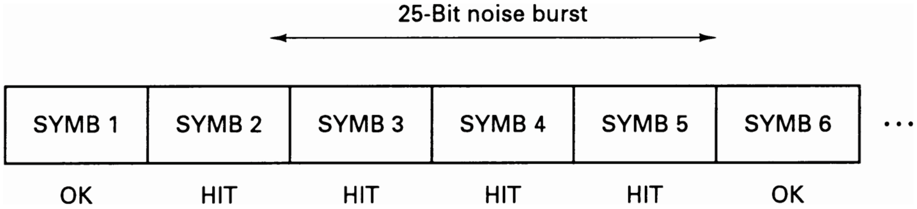


Figure 3

Data block disturbed by 25-bit noise burst.

In this example, notice that a burst of noise that lasts for a duration of 25 contiguous bits must disturb exactly four symbols. The R-S decoder for the (255, 247) code will correct *any* four-symbol errors without regard to the type of damage suffered by the symbol. In other words, when a decoder corrects a byte, it replaces the incorrect byte with the correct one, whether the error was caused by one bit being corrupted or all eight bits being corrupted. Thus if a symbol is wrong, it might as well be wrong in all of its bit positions. This gives an R-S code a tremendous burst-noise advantage over binary codes, even allowing for the interleaving of binary codes. In this example, if the 25-bit noise disturbance had occurred in a random fashion rather than as a contiguous burst, it should be clear that many more than four symbols would be affected (as many as 25 symbols might be disturbed). Of course, that would be beyond the capability of the (255, 247) code.

R-S PERFORMANCE AS A FUNCTION OF SIZE, REDUNDANCY, AND CODE RATE

For a code to successfully combat the effects of noise, the noise duration has to represent a relatively small percentage of the codeword. To ensure that this happens most of the time, the received noise should be averaged over a long period of time, reducing the effect of a freak streak of bad luck. Hence, error-correcting codes become more efficient (error performance improves) as the code block size increases, making R-S codes an attractive choice whenever long block lengths are desired. This is seen by the family of curves in Figure 4, where the rate of the code is held at a constant 7/8, while its block size increases from n = 32 symbols (with m = 5 bits per symbol) to n = 256 symbols (with m = 8 bits per symbol). Thus, the block size increases from 160 bits to 2048 bits.

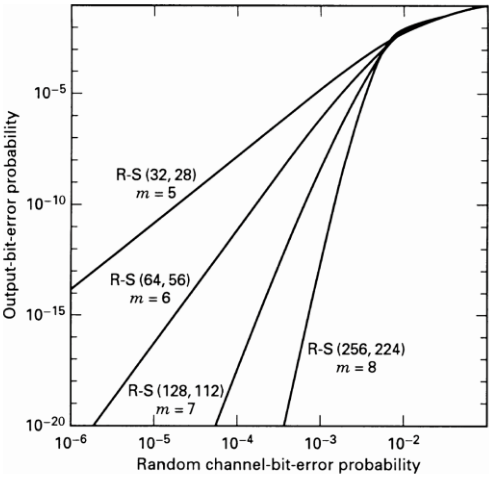


Figure 4
Reed-Solomon rate 7/8 decoder performance as a function of symbol size.

As the redundancy of an R-S code increases (lower code rate), its implementation grows in complexity (especially for high-speed devices). Also, the bandwidth expansion must grow for any real-time communications application. However, the benefit of increased redundancy, just like the benefit of increased symbol size, is the improvement in bit-error performance, as can be seen in Figure 5, where the code length n is held at a constant 64, while the number of data symbols decreases from k = 60 to k = 4 (redundancy increases from 4 symbols to 60 symbols).

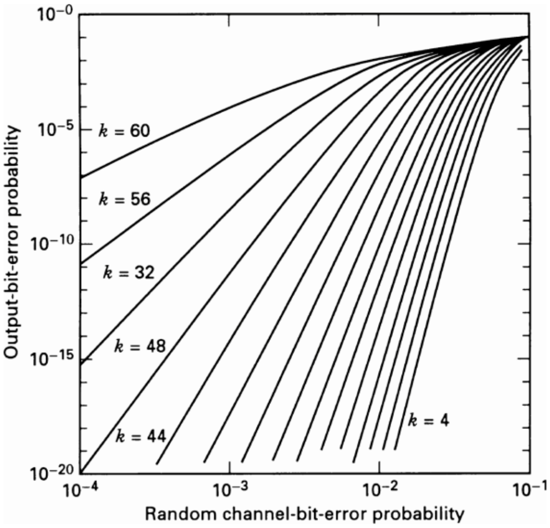


Figure 5
Reed-Solomon (64, *k*) decoder performance as a function of redundancy.

Figure 5 represents transfer functions (output bit-error probability versus input channel symbol-error probability) of hypothetical decoders. Because there is no system or channel in mind (only an output-versus-input of a decoder), you might get the idea that the improved error performance versus increased redundancy is a monotonic function that will continually provide system improvement even as the code rate approaches zero. However, this is not the case for codes operating in a real-time communication system. As the rate of a code varies from minimum to maximum (0 to 1), it is interesting to observe the effects shown in Figure 6.

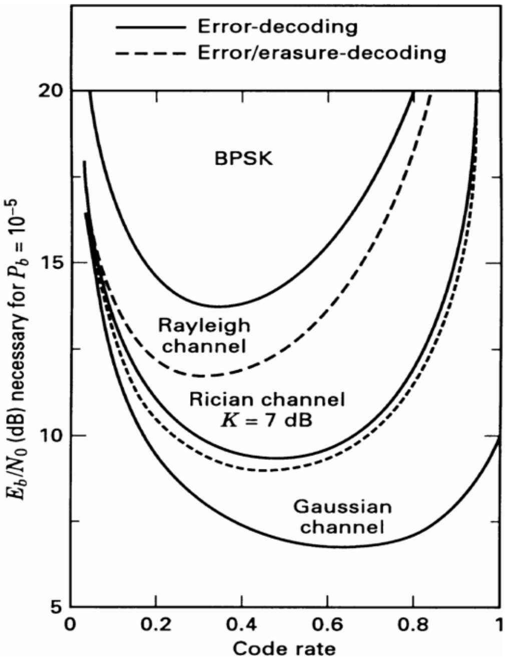


Figure 6
BPSK plus Reed-Solomon (31, *k*) decoder performance as a function of code rate.

Here, the performance curves are plotted for BPSK modulation and an R-S (31, *k*) code for various channel types. Figure 6 reflects a real-time communication system, where the price paid for error-correction coding is bandwidth expansion by a factor equal to the inverse of the code rate. The curves plotted show clear optimum code rates that minimize the required E_b/N_0. The optimum code rate is about 0.6 to 0.7 for a Gaussian channel, 0.5 for a Rician-fading channel (with the ratio of direct to reflected received signal power, K = 7 dB), and 0.3 for a Rayleigh-fading channel. Why is there an E_b/N_0 degradation for very large rates (small redundancy) and very low rates (large redundancy)? It is easy to explain the degradation at high rates compared to the optimum rate. Any code generally provides a coding-gain benefit; thus, as the code rate approaches unity (no coding), the system will suffer worse error performance. The degradation at low code rates is more subtle because in a real-time communication system using both modulation and coding, there are two mechanisms at work. One mechanism works to improve error performance, and the other works to degrade it. The improving mechanism is the coding; the greater the redundancy, the greater will be the error-correcting capability of the code. The degrading mechanism is the energy reduction per channel symbol (compared to the data symbol) that stems from the increased redundancy (and faster signaling in a real-time communication system). The reduced symbol energy causes the demodulator to make more errors. Eventually, the second mechanism wins out, and thus at very low code rates the system experiences error-performance degradation.

Let’s see if we can corroborate the error performance versus code rate in Figure 6 with the curves in Figure 2. The figures are really not directly comparable because the modulation is BPSK in Figure 6 and 32-ary MFSK in Figure 2. However, perhaps we can verify that R-S error performance versus code rate exhibits the same general curvature with MFSK modulation as it does with BPSK. In Figure 2, the error performance over an AWGN channel improves as the symbol error- correcting capability, t, increases from t = 1 to t = 4; the t = 1 and t = 4 cases correspond to R-S (31, 29) and R-S (31, 23) with code rates of 0.94 and 0.74 respectively. However, at t = 8, which corresponds to R-S (31, 15) with code rate = 0.48, the error performance at $P_B = 10^{-5}$ degrades by about 0.5 dB of E_b/N_0 compared to the t = 4 case. From Figure 2, we can conclude that if we were to plot error performance versus code rate, the curve would have the same general “shape” as it does in Figure 6. Note that this manifestation cannot be gleaned from Figure 1, since that figure represents a decoder transfer function, which provides no information about the channel and the demodulation. Therefore, of the two mechanisms at work in the channel, the Figure 1 transfer function only presents the output-versus-input benefits of the decoder, and displays nothing about the loss of energy as a function of lower code rate.

FINITE FIELDS

In order to understand the encoding and decoding principles of nonbinary codes, such as Reed-Solomon (R-S) codes, it is necessary to venture into the area of finite fields known as *Galois Fields* (GF). For any prime number, p, there exists a finite field denoted GF(p) that contains p elements. It is possible to extend GF(p) to a field of p^m elements, called an *extension field* of GF(p), and denoted by GF(p^m), where m is a nonzero positive integer. Note that GF(p^m) contains as a subset the elements of GF(p). Symbols from the extension field GF(2^m) are used in the construction of Reed-Solomon (R-S) codes.

The binary field GF(2) is a subfield of the extension field GF(2^m), in much the same way as the real number field is a subfield of the complex number field.

Besides the numbers 0 and 1, there are additional unique elements in the extension field that will be represented with a new symbol α. Each nonzero element in GF(2^m) can be represented by a power of α. An *infinite* set of elements, *F*, is formed by starting with the elements {0, 1, α}, and generating additional elements by progressively multiplying the last entry by α, which yields the following:

$$F = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^j, \ldots\} = \{0, \alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^j, \ldots\} \qquad (9)$$

To obtain the *finite* set of elements of GF(2^m) from *F*, a condition must be imposed on *F* so that it may contain only 2^m elements and is closed under multiplication. The condition that closes the set of field elements under multiplication is characterized by the irreducible polynomial shown below:

$$\alpha^{(2m-1)} + 1 = 0 \ \text{ or } \ \alpha^{(2m-1)} = 1 = \alpha^0 \qquad (10)$$

Using this polynomial constraint, any field element that has a power equal to or greater than 2^m - 1 can be reduced to an element with a power less than 2^m - 1, as follows:

$$\alpha^{(2^m+n)} = \alpha^{(2^m-1)}\ \alpha^{n+1} = \alpha^{n+1} \qquad (11)$$

Thus, Equation (10) can be used to form the finite sequence *F** from the infinite sequence *F* as follows:

$$\begin{aligned} F^* &= \left\{\ 0, 1, \alpha, \alpha^2, \ldots, \alpha^{2^m-2}, \alpha^{2^m-1}, \alpha^{2^m}, \ldots \right\} \\ &= \left\{\ 0, \alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^{2^m-2}, \alpha^0, \alpha^1, \alpha^2, \ldots \right\} \end{aligned} \qquad (12)$$

Therefore, it can be seen from Equation (12) that the elements of the finite field, GF(2^m), are as follows:

$$\mathrm{GF}(2^m) = \left\{\ 0,\ \alpha^0,\ \alpha^1,\ \alpha^2,\ .\ .\ .\ .\ ,\ \alpha^{2^m-2}\ \right\} \qquad (13)$$

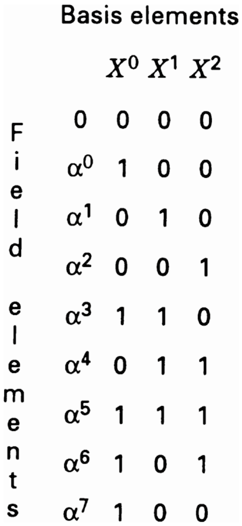
Addition in the Extension Field GF(2^m)

Each of the 2^m elements of the finite field, GF(2^m), can be represented as a distinct polynomial of *degree* m - 1 or less. The degree of a polynomial is the value of its highest-order exponent. We denote each of the nonzero elements of GF(2^m) as a polynomial, $a_i(X)$, where at least one of the m coefficients of $a_i(X)$ is nonzero. For $i = 0,1,2,\ldots,2^m - 2$,

$$\alpha^i = a_i(X) = a_{i,0} + a_{i,1}X + a_{i,2}X^2 + \ldots + a_{i,m-1}X^{m-1} \qquad (14)$$

Consider the case of $m = 3$, where the finite field is denoted GF(2^3). Figure 7 shows the mapping of the seven elements $\{\alpha^i\}$ and the zero element, in terms of the basis elements $\{X^0, X^1, X^2\}$ described by Equation (14). Since Equation (10) indicates that $\alpha^0 = \alpha^7$, there are seven nonzero elements or a total of eight elements in this field. Each row in the Figure 7 mapping comprises a sequence of binary values representing the coefficients $a_{i,0}$, $a_{i,1}$, and $a_{i,2}$ in Equation (14). One of the benefits of using extension field elements $\{\alpha^i\}$ in place of binary elements is the compact notation that facilitates the mathematical representation of nonbinary encoding and decoding processes. Addition of two elements of the finite field is then defined as the modulo-2 sum of each of the polynomial coefficients of like powers,

$$\alpha^i + \alpha^j = (a_{i,0} + a_{j,0}) + (a_{i,1} + a_{j,1})X + \ldots + (a_{i,m-1} + a_{j,m-1})X^{m-1} \qquad (15)$$



	Basis elements		
Field elements	X^0	X^1	X^2
0	0	0	0
α^0	1	0	0
α^1	0	1	0
α^2	0	0	1
α^3	1	1	0
α^4	0	1	1
α^5	1	1	1
α^6	1	0	1
α^7	1	0	0

Figure 7

Mapping field elements in terms of basis elements for GF(8) with $f(x) = 1 + x + x^3$.

A Primitive Polynomial Is Used to Define the Finite Field

A class of polynomials called *primitive polynomials* is of interest because such functions define the finite fields GF(2^m) that in turn are needed to define R-S codes. The following condition is necessary and sufficient to guarantee that a polynomial is primitive. An irreducible polynomial

$f(X)$ of degree m is said to be primitive if the smallest positive integer n for which $f(X)$ divides $X^n + 1$ is $n = 2^m - 1$. Note that the statement A divides B means that A divided into B yields a nonzero quotient and a zero remainder. Polynomials will usually be shown low order to high order. Sometimes, it is convenient to follow the reverse format (for example, when performing polynomial division).

Example 1: Recognizing a Primitive Polynomial

Based on the definition of a primitive polynomial given above, determine whether the following irreducible polynomials are primitive.

a. $1 + X + X^4$

b. $1 + X + X^2 + X^3 + X^4$

Solution

a. We can verify whether this degree $m = 4$ polynomial is primitive by determining whether it divides $X^n + 1 = X^{(2m-1)} + 1 = X^{15} + 1$, but does not divide $X^n + 1$, for values of n in the range of $1 \leq n < 15$. It is easy to verify that $1 + X + X^4$ divides $X^{15} + 1$, and after repeated computations it can be verified that $1 + X + X^4$ will not divide $X^n + 1$ for any n in the range of $1 \leq n < 15$. Therefore, $1 + X + X^4$ is a primitive polynomial.

b. It is simple to verify that the polynomial $1 + X + X^2 + X^3 + X^4$ divides $X^{15} + 1$. Testing to see whether it will divide $X^n + 1$ for some n that is less than 15 yields the fact that it also divides $X^5 + 1$. Thus, although $1 + X + X^2 + X^3 + X^4$ is irreducible, it is not primitive.

Table 1

Some Primitive Polynomials

m	
3	$1 + X + X^3$
4	$1 + X + X^4$
5	$1 + X^2 + X^5$
6	$1 + X + X^6$
7	$1 + X^3 + X^7$
8	$1 + X^2 + X^3 + X^4 + X^8$
9	$1 + X^4 + X^9$
10	$1 + X^3 + X^{10}$
11	$1 + X^2 + X^{11}$
12	$1 + X + X^4 + X^6 + X^{12}$
13	$1 + X + X^3 + X^4 + X^{13}$
14	$1 + X + X^6 + X^{10} + X^{14}$
15	$1 + X + X^{15}$
16	$1 + X + X^{3 +} X^{12} + X^{16}$
17	$1 + X^3 + X^{17}$
18	$1 + X^7 + X^{18}$
19	$1 + X + X^{2 +} X^5 + X^{19}$
20	$1 + X^3 + X^{20}$
21	$1 + X^2 + X^{21}$
22	$1 + X + X^{22}$
23	$1 + X^5 + X^{23}$
24	$1 + X + X^{2 +} X^7 + X^{24}$

Two arithmetic operations, addition and multiplication, can be defined for the GF(2^3) finite field. Addition is shown in Table 2, and multiplication is shown in Table 3 for the nonzero elements only.

The multiplication rules in Table 3 follow the usual procedure, in which the product of the field elements is obtained by adding their exponents modulo-(2^m - 1), or for this case, modulo-7.

Table 2
Addition Table

	α^0	α^1	α^2	α^3	α^4	α^5	α^6
α^0	0	α^3	α^6	α^1	α^5	α^4	α^2
α^1	α^3	0	α^4	α^0	α^2	α^6	α^5
α^2	α^6	α^4	0	α^5	α^1	α^3	α^0
α^3	α^1	α^0	α^5	0	α^6	α^2	α^4
α^4	α^5	α^2	α^1	α^6	0	α^0	α^3
α^5	α^4	α^6	α^3	α^2	α^0	0	α^1
α^6	α^2	α^5	α^0	α^4	α^3	α^1	0

Table 3
Multiplication Table

	α^0	α^1	α^2	α^3	α^4	α^5	α^6
α^0	α^0	α^1	α^2	α^3	α^4	α^5	α^6
α^1	α^1	α^2	α^3	α^4	α^5	α^6	α^0
α^2	α^2	α^3	α^4	α^5	α^6	α^0	α^1
α^3	α^3	α^4	α^5	α^6	α^0	α^1	α^2
α^4	α^4	α^5	α^6	α^0	α^1	α^2	α^3
α^5	α^5	α^6	α^0	α^1	α^2	α^3	α^4
α^6	α^6	α^0	α^1	α^2	α^3	α^4	α^5

A Simple Test to Determine Whether a Polynomial Is Primitive

There is another way of defining a primitive polynomial that makes its verification relatively easy. For an irreducible polynomial to be a primitive polynomial, at least one of its roots must be a primitive element. A *primitive element* is one that when raised to higher-order exponents will yield all the nonzero elements in the field. Since the field is a finite field, the number of such elements is finite.

Example 2: A Primitive Polynomial Must Have at Least One Primitive Element

Find the m = 3 roots of $f(X) = 1 + X + X^3$, and verify that the polynomial is primitive by checking that at least one of the roots is a primitive element. What are the roots? Which ones are primitive?

Solution

The roots will be found by enumeration. Clearly, $\alpha^0 = 1$ is not a root because $f(\alpha^0) = 1$. Now, use Table 2 to check whether α^1 is a root. Since

$$f(\alpha) = 1 + \alpha + \alpha^3 = 1 + \alpha^0 = 0$$

α is therefore a root.

Now check whether α^2 is a root:

$$f(\alpha^2) = 1 + \alpha^2 + \alpha^6 = 1 + \alpha^0 = 0$$

Hence, α^2 is a root.

Now check whether α^3 is a root.

$$f(\alpha^3) = 1 + \alpha^3 + \alpha^9 = 1 + \alpha^3 + \alpha^2 = 1 + \alpha^5 = \alpha^4 \neq 0$$

Hence, α^3 is *not* a root. Is α^4 a root?

$$f(\alpha^4) = \alpha^{12} + \alpha^4 + 1 = \alpha^5 + \alpha^4 + 1 = 1 + \alpha^0 = 0$$

Yes, it is a root. Hence, the roots of $f(X) = 1 + X + X^3$ are α, α^2, and α^4. It is not difficult to verify that starting with any of these roots and generating higher-order exponents yields all of the seven nonzero elements in the field. Hence, each of the roots is a primitive element. Since our verification requires that at least one root be a primitive element, the polynomial is primitive.

Reed-Solomon Encoding

Equation (2), repeated below as Equation (16), expresses the most conventional form of Reed-Solomon (R-S) codes in terms of the parameters *n*, *k*, *t*, and any positive integer *m* > 2.

$$(n, k) = (2^m - 1, 2^m - 1 - 2t) \tag{16}$$

where $n - k = 2t$ is the number of parity symbols, and *t* is the symbol-error correcting capability of the code. The generating polynomial for an R-S code takes the following form:

$$\mathbf{g}(X) = g_0 + g_1 X + g_2 X^2 + \ldots + g_{2t-1} X^{2t-1} + X^{2t} \tag{17}$$

The degree of the generator polynomial is equal to the number of parity symbols. R-S codes are a subset of the Bose, Chaudhuri, and Hocquenghem (BCH) codes; hence, it should be no surprise that this relationship between the degree of the generator polynomial and the number of parity symbols holds, just as for BCH codes. Since the generator polynomial is of degree 2*t*, there must be precisely 2*t* successive powers of α that are roots of the polynomial. We designate the roots of $\mathbf{g}(X)$ as α, α^2, …, α^{2t}. It is not necessary to start with the root α; starting with any power of α is possible. Consider as an example the (7, 3) double-symbol-error correcting R-S code. We describe the generator polynomial in terms of its $2t = n - k = 4$ roots, as follows:

$$\begin{aligned}
\mathbf{g}(X) &= (X - \alpha)(X - \alpha^2)(X - \alpha^3)(X - \alpha^4) \\
&= \left(X^2 - (\alpha + \alpha^2)X + \alpha^3\right)\left(X^2 - (\alpha^3 + \alpha^4)X + \alpha^7\right) \\
&= (X^2 - \alpha^4 X + \alpha^3)(X^2 - \alpha^6 X + \alpha^0) \\
&= X^4 - (\alpha^4 + \alpha^6)X^3 + (\alpha^3 + \alpha^{10} + \alpha^0)X^2 - (\alpha^4 + \alpha^9)X + \alpha^3 \\
&= X^4 - \alpha^3 X^3 + \alpha^0 X^2 - \alpha^1 X + \alpha^3
\end{aligned}$$

Following the low order to high order format, and changing negative signs to positive, since in the binary field +1 = –1, $\mathbf{g}(X)$ can be expressed as follows:

$$\mathbf{g}(X) = \alpha^3 + \alpha^1 X + \alpha^0 X^2 + \alpha^3 X^3 + X^4 \tag{18}$$

Reed-Solomon Decoding

Assume that during transmission the codeword becomes corrupted so that two symbols are received in error. (This number of errors corresponds to the maximum error-correcting capability of the code.) For this seven-symbol codeword example, the error pattern, **e**(*X*), can be described in polynomial form as follows:

$$\mathbf{e}(X) = \sum_{n=0}^{6} e_n X^n \tag{19}$$

For this example, let the double-symbol error be such that

$$\begin{aligned} \mathbf{e}(X) &= 0 + 0X + 0X^2 + \alpha^2 X^3 + \alpha^5 X^4 + 0X^5 + 0X^6 \\ &= (000) + (000)X + (000)X^2 + (001)X^3 + (111)X^4 + (000)X^5 + (000)X^6 \end{aligned} \tag{20}$$

In other words, one parity symbol has been corrupted with a 1-bit error (seen as α^2), and one data symbol has been corrupted with a 3-bit error (seen as α^5). The received corrupted-codeword polynomial, **r**(*X*), is then represented by the sum of the transmitted-codeword polynomial and the error-pattern polynomial as follows:

$$r(X)=U(X)+e(X) \tag{21}$$

Following Equation (21), we add **U**(*X*) from to **e**(*X*) to yield **r**(*X*), as follows:

$$\begin{aligned} \mathbf{r}(X) &= (100) + (001)X + (011)X^2 + (100)X^3 + (101)X^4 + (110)X^5 + (111)X^6 \\ &= \alpha^0 + \alpha^2 X + \alpha^4 X^2 + \alpha^0 X^3 + \alpha^6 X^4 + \alpha^3 X^5 + \alpha^5 X^6 \end{aligned} \tag{22}$$

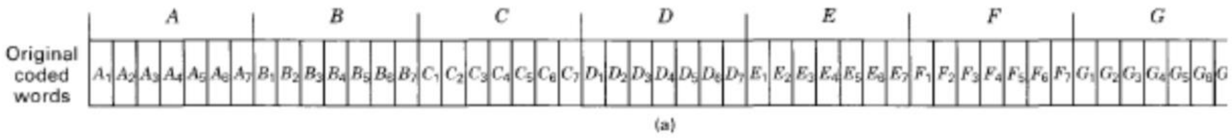
In this example, there are four unknowns—two error locations and two error values. Notice an important difference between the nonbinary decoding of $\mathbf{r}(X)$ that we are faced with in Equation (22) and binary decoding; in binary decoding, the decoder only needs to find the error locations . Knowledge that there is an error at a particular location dictates that the bit must be “flipped” from 1 to 0 or vice versa. But here, the nonbinary symbols require that we not only learn the error locations, but also determine the correct symbol values at those locations. Since there are four unknowns in this example, four equations are required for their solution.

INTERLEAVING AND CONCATENATED CODES

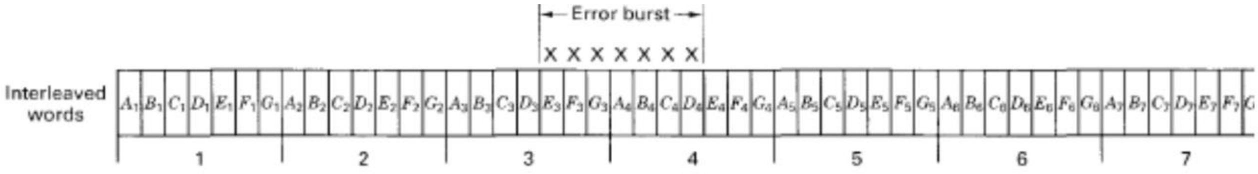
A channel that has memory is one that exhibits mutually dependent signal transmission impairments. A channel that exhibits multipath fading, where signals arrive at the receiver over two or more paths of different lengths, is an example of a channel with memory. The effect is that the signals can arrive out of phase with each other, and the cumulative received signal is distorted. Wireless mobile communication channels, as well as ionospheric and tropospheric propagation channels, suffer from such phenomena. Also, some channels suffer from switching noise and other burst noise (e.g., telephone channels or channels disturbed by pulse jamming). All of these time-correlated impairments result in statistical dependence among successive symbol transmissions. That is, the disturbances tend to cause errors that occur in bursts, instead of as isolated events.

Under the assumption that the channel has memory, the errors no longer can be characterized as single randomly distributed bit errors whose occurrence is independent from bit to bit. Most block or convolutional codes are designed to combat random independent errors. The result of a channel having memory on such coded signals is to cause degradation in error performance. Coding techniques for channels with memory have been proposed, but the greatest problem with such coding is the difficulty in obtaining accurate models of the often time-varying statistics of such channels. One technique, which only requires a knowledge of the duration or span of the channel memory, not its exact statistical characterization, is the use of time diversity or interleaving.

The interleaver shuffles the code symbols over a span of several block lengths (for block codes) or several constraint lengths (for convolutional codes). The span required is determined by the burst duration. A simple example is shown below.


A
B
C
D
E
F
G
Original coded words
A_1 A_2 A_3 A_4 A_5 A_6 A_7
B_1 B_2 B_3 B_4 B_5 B_6 B_7
C_1 C_2 C_3 C_4 C_5 C_6 C_7
D_1 D_2 D_3 D_4 D_5 D_6 D_7
E_1 E_2 E_3 E_4 E_5 E_6 E_7
F_1 F_2 F_3 F_4 F_5 F_6 F_7
G_1 G_2 G_3 G_4 G_5 G_6 G

(a)


Error burst
X X X X X X X
Interleaved words
A_1 B_1 C_1 D_1 E_1 F_1 G_1 A_2 B_2 C_2 D_2 E_2 F_2 G_2 A_3 B_3 C_3 D_3 E_3 F_3 G_3 A_4 B_4 C_4 D_4 E_4 F_4 G_4 A_5 B_5 C_5 D_5 E_5 F_5 G_5 A_6 B_6 C_6 D_6 E_6 F_6 G_6 A_7 B_7 C_7 D_7 E_7 F_7 G
1
2
3
4
5
6
7

(b)

BLOCK INTERLEAVING

A block interleaver accepts the coded symbols in blocks from the encoder, permutes the symbols, and then feeds the rearranged symbols to the modulator. The usual permutation of the block is accomplished by filling the columns of an M-row-by N-column (M x N) array with the encoded sequence. After the array is completely filled, the symbols are then fed to the modulator one row at a time and transmitted over the channel. Figure below illustrates an example of an interleaver with M = 4 rows and N = 6 columns.

The most important characteristics of such a block interleaver are as follows:

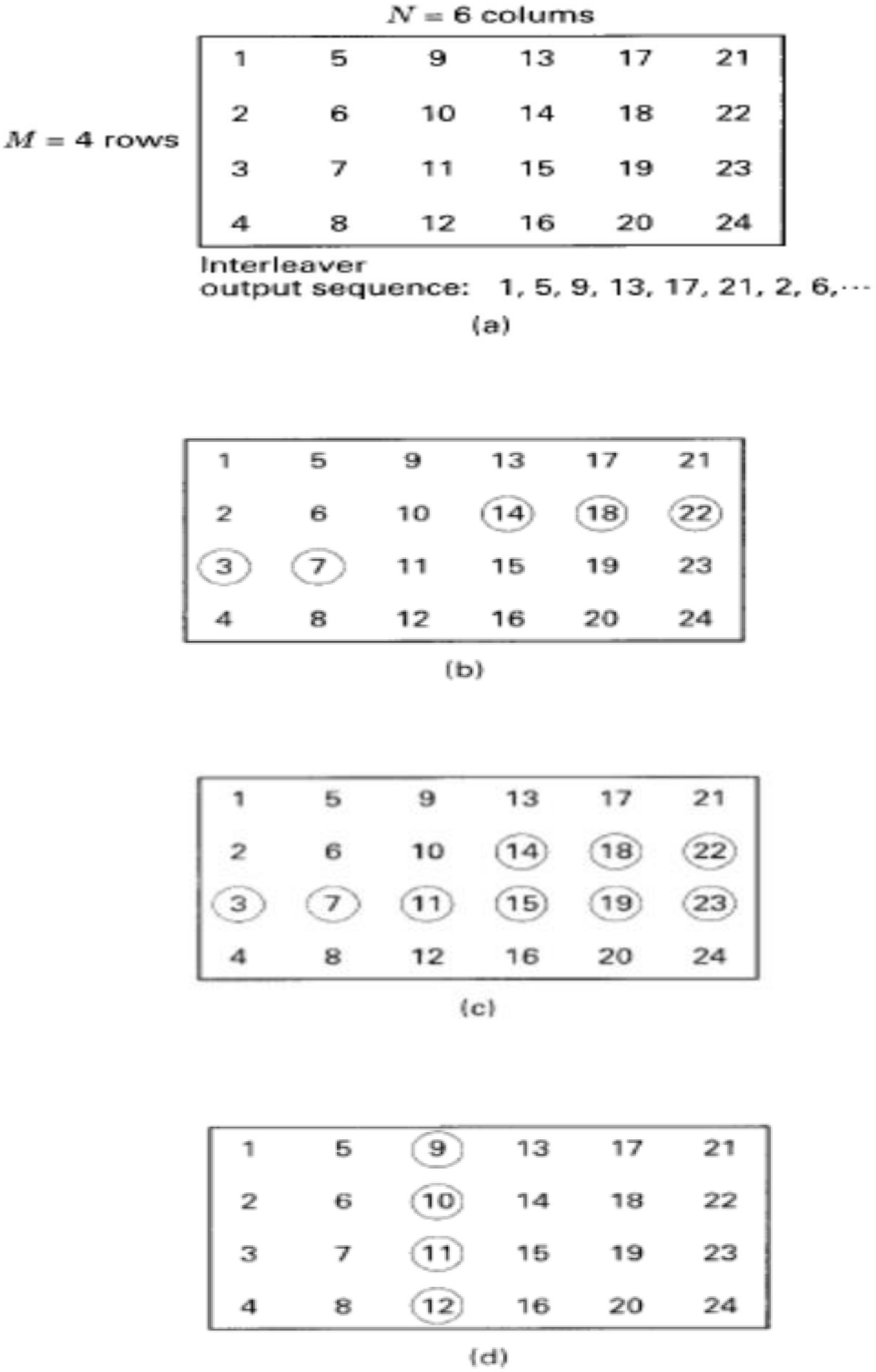
Any burst of less than N contiguous channel symbol errors results in isolated errors at the deinterlever output that are separated from each other by at least M symbols. Any bN burst of errors, where b > 1, results in output bursts from the deinterleaver of no more than I b l symbol errors. Each output burst is separated from the other bursts by no less than M – LbJ symbols. The notation lx l means the smallest integer no less than x, and LxJ means the largest integer no greater than x.

A periodic sequence of single errors spaced N symbols apart results in a single burst of errors of length Mat the deinterleaver output.

The interleaver/deinterleaver end-to-end delay is approximately 2MN symbol times. To be precise, only M (N- 1) + 1 memory cells need to be filled before transmission can begin (as soon as the first symbol of the last column of the M x N array is filled). A corresponding number needs to be filled at the receiver before decoding begins. Thus the minimum end-to-end delay is (2M N- 2M+ 2) symbol times, not including any channel propagation delay.

The memory requirement is MN symbols for each location (interleaver and deinterleaver).

However, since the M x N array needs to be (mostly) filled before it can be read out, a memory of 2MN symbols is generally implemented at each location to allow the emptying of one M x N array while the other is being filled, and vice versa.



1	5	9	13	17	21
2	6	10	14	18	22
3	7	11	15	19	23
4	8	12	16	20	24

Interleaver output sequence: 1, 5, 9, 13, 17, 21, 2, 6,···

(a)

1	5	9	13	17	21
2	6	10	14	18	22
3	7	11	15	19	23
4	8	12	16	20	24

(b)

1	5	9	13	17	21
2	6	10	14	18	22
3	7	11	15	19	23
4	8	12	16	20	24

(c)

1	5	9	13	17	21
2	6	10	14	18	22
3	7	11	15	19	23
4	8	12	16	20	24

(d)

CONVOLUTIONAL INTERLEAVING:

Convolutional interleavers have been proposed by Ramsey and Forney. The structure proposed by Forney appears in fig. below.

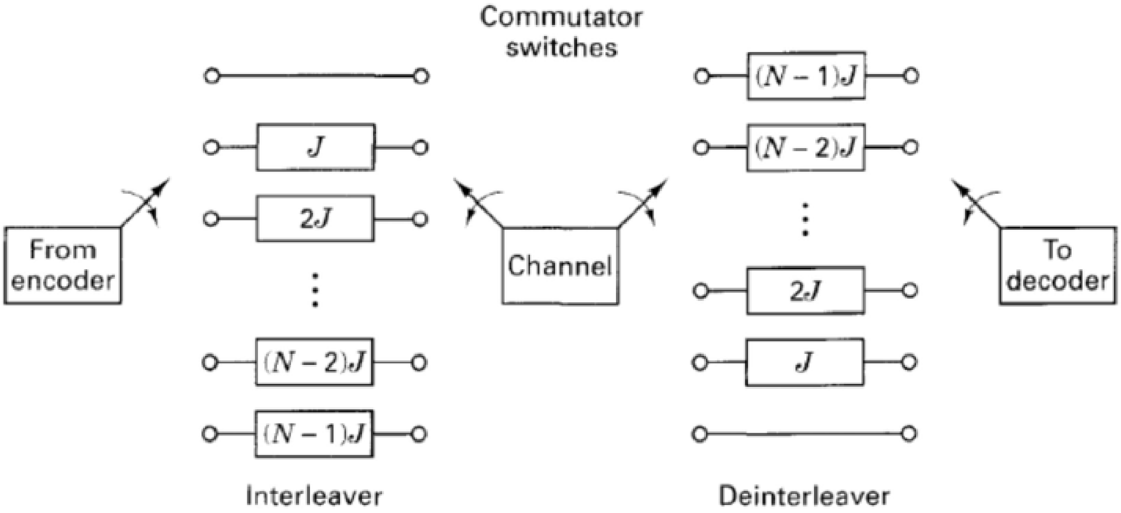


Fig. Shift register implementation of a convolutional interleaver/deinterleaver.

The code symbols are sequentially shifted into the bank of N registers; each successive register provides J symbols more storage than did the preceding one. The zeroth register provides no storage (the symbol is transmitted immediately). With each new code symbol the commutator switches to a new register, and the new code symbol is shifted in while the oldest code symbol in that register is shifted out to the modulator/transmitter. After the (N - 1)th register, the commutator returns to the zeroth register and starts again. The deinterleaver performs the inverse operation, and the input and output commutators for both interleaving and de interleaving must be synchronized.

CONCATENATED CODES:

A concatenated code is one that uses two levels of coding, an inner code and an outer code, to achieve the desired error performance. Figure below illustrates the order of encoding and decoding.

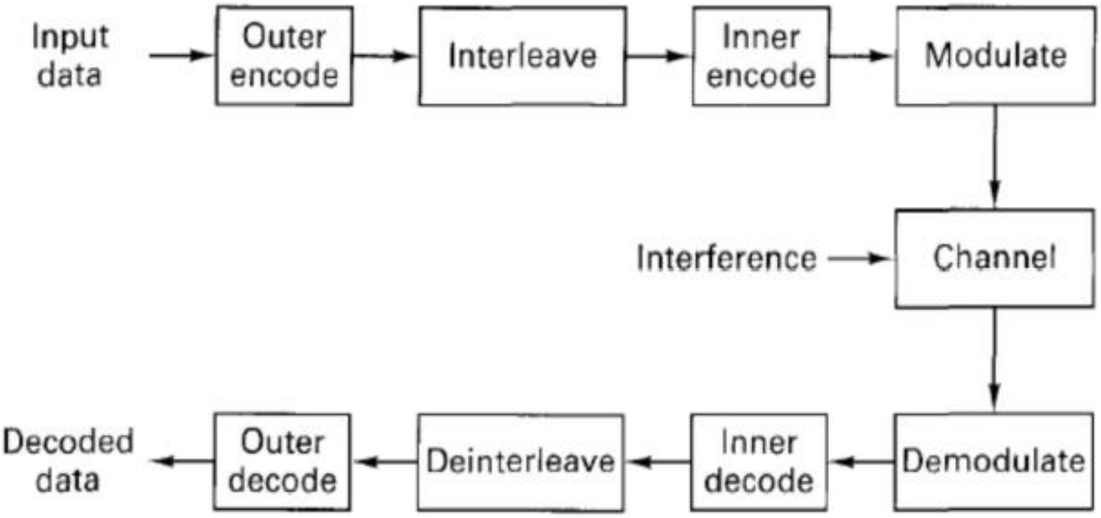


Fig. Block diagram of a concatenated coding system.

The inner code, the one that interfaces with the modulator/demodulator and channel, is usually configured to correct most of the channel errors.

The outer code, usually a higher-rate (lower-redundancy) code then reduces the probability of error to the specified level.

The primary reason for using a concatenated code is to achieve a low error rate with an overall implementation complexity which is less than that which would be required by a single coding operation.

The interleaver required to spread any error bursts that may appear at the output of the inner coding operation.

One of the most popular concatenated coding systems uses a Viterbi-decoded convolutional inner code and a Reed-Solomon (R-S) outer code. with interleaving between the two coding steps.

CODING AND INTERLEAVING APPLIED TO THE COMPACT DISC DIGITAL AUDIO SYSTEM:

Philips & Sony Corp. defined a standard for digital storage & reproduction of audio signals called compact disc(CD) digital audio system. World standard 120 mm diameter CD.

- Stores digitized audio waveform.
- Sampled at 44.1 ksamples per second for 20 KHz BW to 2 levels (16 bits per sample).
- Dynamic range 96 dB, harmonic distortion = 0.005%.
- Stores about 10^{10} bits.

Scratches & other damage to CD causes burst like errors.

Approximately 4000 bits (2.5 mm) burst errors can be corrected. Prob. of bit error, $P_B = 10^{-4}$

Hierarchy of errors control in CIRC system—

(i) Decode first attempts for error correction.

If error correction capability is exceeded, decoder goes for reassure correction. If the erasure correction capability is exceeded the decoder attempts to conceal unreliable data samples by interpolation between reliable neighbouring samples.
If the interpolation capability is exceeded, the decoder simply mutes the system for the duration of unreliable samples.

CIRC ENCODING:

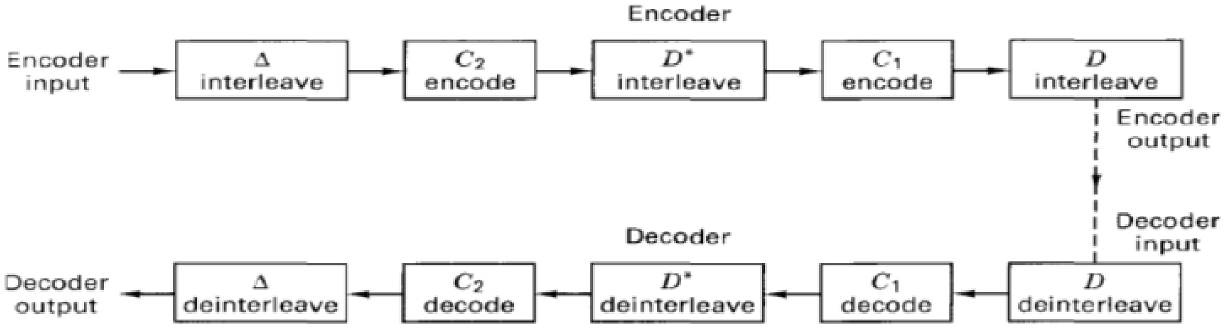


Fig. Block Diagram of CIRC Encoder & Decoder

The steps are as follows:-

L1 interleave. Even-numbered samples are separated from odd-numbered samples by two frame times in order to scramble uncorrectable but detectable byte errors. This facilitates the interpolation process.

C2 encode. Four Reed-Solomon (R-S) parity bytes are added to the 11-interleaved 24-byte frame, resulting in a total of n = 28 bytes. This (28, 24) code is called the outer code.

D* interleave. Here each byte is delayed a different length, thereby spreading errors over several codewords. C2 encoding together with D* interleaving have the function of providing for the correction of burst errors and error patterns that the C1 decoder cannot correct.

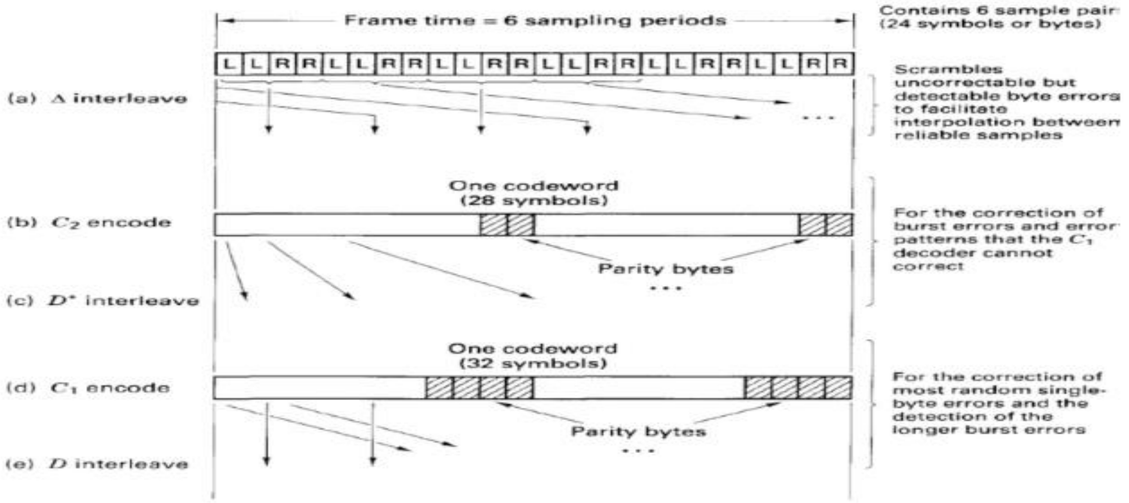

Frame time = 6 sampling periods
Contains 6 sample pair
(24 symbols or bytes)
L L R R L L R R L L R R L L R R L L R R L L R R
(a) Δ interleave
Scrambles uncorrectable but detectable byte errors to facilitate interpolation between reliable samples
One codeword
(28 symbols)
(b) C_2 encode
Parity bytes
For the correction of burst errors and error patterns that the C_1 decoder cannot correct
(c) D^* interleave
One codeword
(32 symbols)
(d) C_1 encode
Parity bytes
For the correction of most random single-byte errors and the detection of the longer burst errors
(e) D interleave

Fig. Compact disc encoder. (a)~ interleave. (b) C2 encode. (c) D* interleave. (d) C1 encode. (e) D interleave.

C1 encode. Four R-S parity bytes are added to the k = 28 bytes of the D*-interleaved frame, resulting in a total of n = 32 bytes. This (32, 28) code is called the inner code.

D interleave. The purpose is to cross-interleave the even bytes of a frame with the odd bytes of the next frame. By this procedure, two consecutive bytes on the disc will always end up in two different codewords. Upon decoding, this interleaving, together with the C1 decoding, results in the correction of most random single errors and the detection of longer burst errors.

CIRC DECODING:

The benefits of CIRC are best seen at the decoder, where the processing steps, shown in Figure are in the reverse order of the encoder steps. The decoder steps are as follows:

D deinterleave. This function is performed by the alternating delay lines marked D. The 32 bytes (B_{i1},…, B_{i32}) of an encoded frame are applied in parallel to the 32 inputs of the D deinterleaver. Each delay is equal to the duration of 1 byte, so that the information of the even bytes of a frame is cross-deinterleaved with that of the odd bytes of the next frame.

C1 decode. The D deinterleaver and the C1 decoder are designed to correct a single byte error in the block of 32 bytes and to detect larger burst errors. If multiple errors occur, the C1 encoder passes them on unchanged, attaching to all 28 remaining bytes an erasure flag, sent via the dashed lines (the four parity bytes used in the C1 decoder are no longer retained).

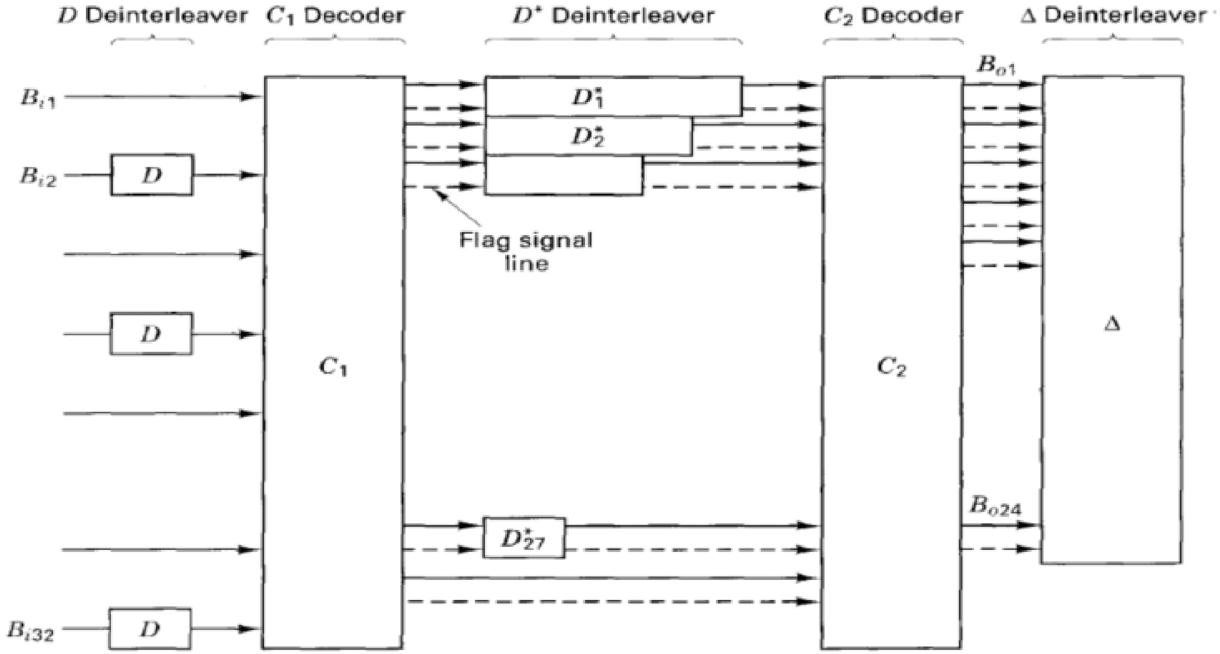


Fig. Compact Disc Decoder.

D* deinterleave. Due to the different lengths of the deinterleaving delay lines D*(1, …, 27) errors that occur in one word at the output of the C1 decoder are spread over a number of words at the input of the C2 decoder. This results in reducing the number of errors per input word of the C2 decoder, enabling the C2 decoder to correct these errors C2 decode. The C2 decoder is intended for the correction of burst errors that the C1 decoder could not correct. If the C2 decoder cannot correct these errors, the 24-byte codeword is passed on unchanged to the ~ deinterleaver and the associated positions are given an erasure flag via the dashed output lines, Bob,…, Bo24·

Δ deinterleave. The final operation deinterleaves uncorrectable but detected byte errors in such a way that interpolation can be used between reliable neighbouring samples.

TURBO CODES

Powerful codes uses concatenation. Turbo codes finds its origin in the will to compensate for the dissymmetry of the concatenated decoder. In this concept of feedback is used.

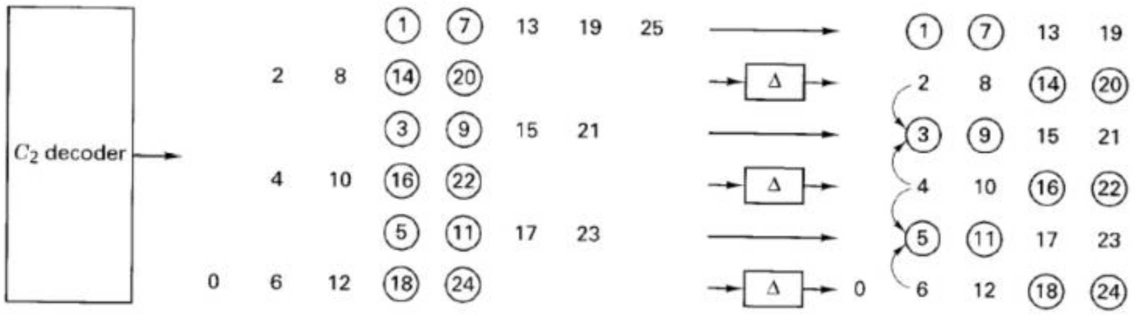


Fig. Effect of interleaving. (Rightmost event is at the earliest time).

A refinement of the concatenated encoding structure plus an iterative algorithm for the decoding the associated code sequence. Introduced in 1993 by Berrou, Glavieus & Thitimashime. Achieved a BER of 10^{-5} with rate ½ over AWGN channel & BPSK modulation at E_b/N_0=0.7 dB. Uses soft decisions information between the two decoders and iterates it several times to produce more reliable decisions.

TURBO CODE CONCEPTS

The mathematical foundations of hypothesis testing rests on Bayes' theorem. For communications engineering, where applications involving an AWGN channel are of great interest, the most useful form of Bayes' theorem expresses the a posteriori probability (APP) of a decision in terms of a continuous-valued random variable x as

$$P(d = i|x) = \frac{p(x|d = i)\, P(d = i)}{p(x)} \qquad i = 1, \cdots, M$$

$$p(x) = \sum_{i=1}^{M} p(x|d = i)\, P(d = i)$$

where P (d = i/x) is the APP, and d = i represents data d belonging to the ith signal class from a set of M classes. Further, p(x ld = i) represents the probability density function (pdf) of a received continuous-valued data-plus-noise signal x, conditioned on the signal class d = i.

Also, p(d = i), called the a priori probability, is the probability of occurrence of the ith signal class. Typically, x is an "observable" random variable or a test statistic that is obtained at the output of a demodulator or some other signal processor. Therefore, p(x) is the pdf of the received signal x, yielding the test statistic over the entire space of signal classes. In the above equation, for a particular observation, p(x) is a scaling factor since it is obtained by averaging over all the classes in the space. Lower case p is used to designate the pdf of a continuous-valued random variable, and upper case P is used to designate probability (a priori and APP).

Module-II: MODULATION & CODING TRADE OFFS

GOALS OF THE COMMUNICATIONS SYSTEM DESIGNER

System trade-offs are fundamental to all digital communication designs. The goals of the designer may include any of the following (1) to maximize transmission bit rate R; (2) to minimize probability of bit error P_B; (3) to minimize required power, or equivalently, to minimize required bit energy to noise power spectral density E_b/N_0; (4) to minimize required system bandwidth W; (5) to maximize system utilization, that is, to provide reliable service for a maximum number of users with minimum delay and with maximum resistance to interference; and (6) to minimize system complexity, computational load, and system cost. A system designer may seek to achieve all these goals simultaneously. However, goals 1 and 2 are clearly in conflict with goals 3 and 4; they call for simultaneously maximizing R, while minimizing P_B, E_b/N_0, and W. There are several constraints and theoretical limitations that necessitate the trading off of any one system requirement with each of the others:

The Nyquist theoretical minimum bandwidth requirement

The Shannon-Hartley capacity theorem (and the Shannon limit)

Government regulations (e.g., frequency allocations)

Technological limitations (e.g., state-of-the-art components)

Other system requirements (e.g., satellite orbits)

Some of the realizable modulation and coding trade-offs can best be viewed as a change in operating point on one of two performance planes. These planes will be referred to as the error probability plane and the bandwidth efficiency plane, and they are described in the following sections.

ERROR PROBABILITY PLANE

Figure illustrates the family of P_B versus E_b/N_O curves for the coherent detection of orthogonal signaling (Fig. a) and multiple phase signaling (Fig b). The modulator uses one of its M = 2k waveforms to represent each k-bit sequence, where M is the size of the symbol set. Figure a illustrates the potential bit error improvement with orthogonal signaling as k (or M) is increased. For orthogonal signal sets, such as orthogonal frequency shift keying (FSK) modulation, increasing the size of the symbol set can provide an improvement in PB, or a reduc tion in the E_b/N_o) required, at the cost of increased bandwidth. Figure b illustrates potential bit error degradation with nonorthogonal signaling as k (or M) increases. For nonorthogonal signal sets, such as multiple phase shift keying (MPSK) modulation, increasing the size of the symbol set can reduce the bandwidth requirement, but at the cost of a degraded PB' or an increased Eh/No requirement. We shall refer to these families of curves (Figure a or b) as error probability performance curves, and to the plane on which they are plotted as an error probability plane. Such a plane describes the locus of operating points available for a particular type of modulation and coding. For a given system information rate, each curve in the plane can be associated with a different fixed minimum required bandwidth; therefore, the set of curves can be termed equibandwidth curves.

As the curves move in the direction of the ordinate, the required transmission bandwidth increases; as the curves move in the opposite direction, the required bandwidth decreases. Once a modulation and coding scheme and an available E_b/N_o are determined, system operation is characterized by a particular point in the error probability plane. Possible trade-offs can be viewed as changes in the operating point on one of the curves or as changes in the operating point from one curve to another curve of the family. These trade-offs are seen in Figure a and b as changes in the system operating point in the direction shown by the arrows.

Movement of the operating point along line 1, between points a and b, can be viewed as trading off between P_B and E_b/N_o performance (with W fixed). Similarly, movement along line 2, between points c and d, is seen as trading P_B versus W (with E_b/N_o fixed). Finally, movement along line 3, between points e and f, illustrates trading W versus E_b/N_o (with B fixed). Movement along line 1 is effected by increasing or decreasing the available Eh/No. This can be achieved, for example, by increasing transmitter power, which means that the trade-off might be accomplished simply by "turning a knob," even after the system is configured. However, the other trade-offs (movement along line 2 or line 3) involve some changes in the system modulation or coding scheme, and therefore need to be accomplished during the system design phase. The advent of software radios will even allow changes to a system's modulation and coding by programmable means.

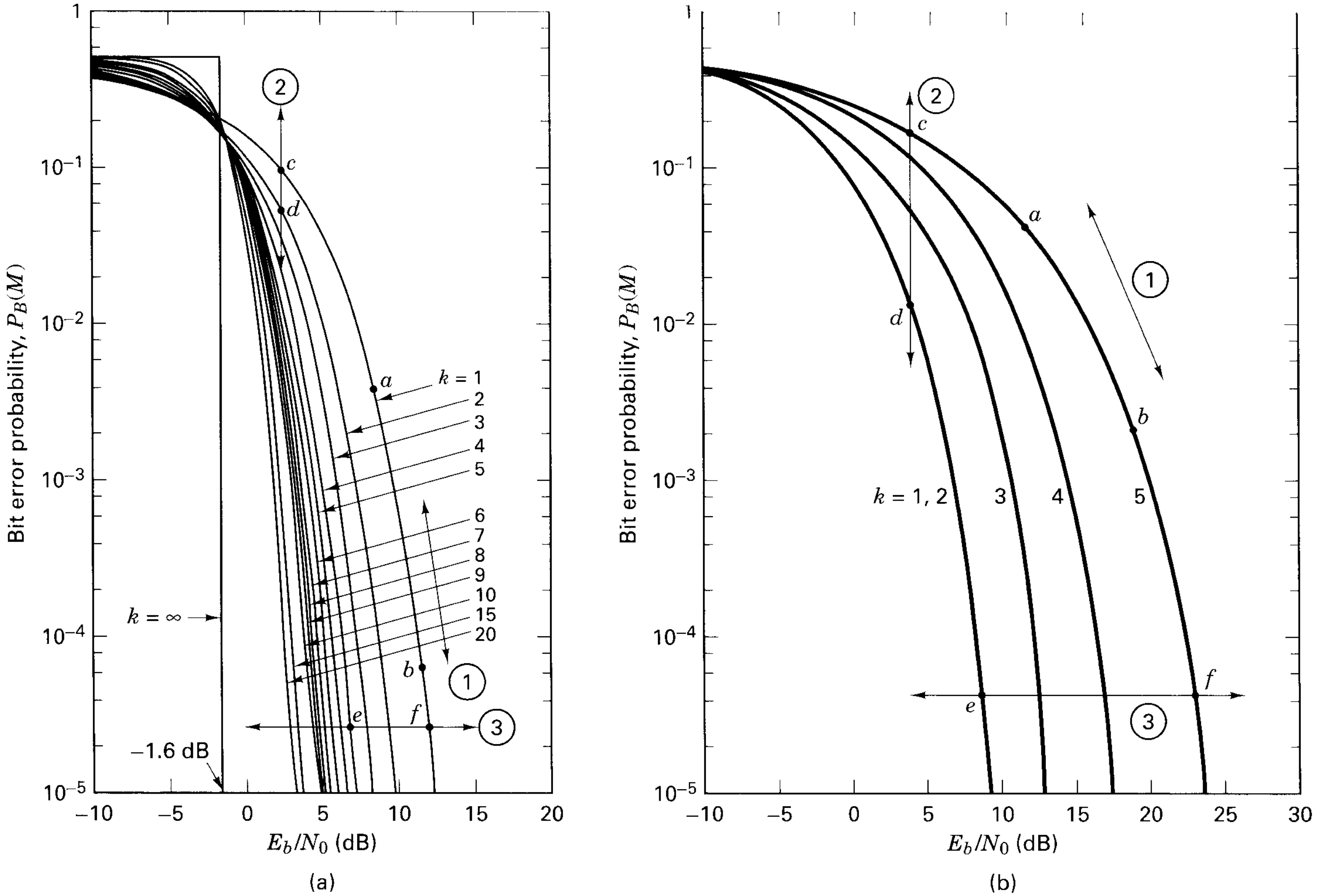


Figure ! Bit error probability versus E_b/N_0 for coherently detected M-ary signaling. (a) Orthogonal signaling. (b) Multiple phase signaling.

NYQUIST MINIMUM BANDWIDTH

Every realizable system having some nonideal filtering will suffer from intersymbol interference (lSl)—the tail of one pulse spilling over into adjacent symbol intervals so as to interfere with correct detection. Nyquist showed that the theoretical minimum bandwidth (Nyquist bandwidth) needed for the baseband transmission of R symbols per second without 1ST is $R_s/2$ hertz. This is a basic theoretical con straint, limiting the designer's goal to expend as little bandwidth as possible. In practice, the Nyquist minimum bandwidth is expanded by about 10% to 40%, because of the constraints of real filters. Thus, typical baseband digi tal communication throughput is reduced from the ideal 2 symbols/s/Hz to the range of about 1.8 to 1.4 symbols/s/Hz. From its set of M symbols, the modulation or coding system assigns to each symbol a k-bit meaning, where $M = 2^k$ Thus, the number of bits per symbol can be expressed as $k = \log_2 M$, and the data rate or bit rate R must be k times faster than the symbol rate R, as expressed by the basic relationship

$$R = kR_s \quad \text{or} \quad R_s = \frac{R}{k} = \frac{R}{\log_2 M}$$

For signaling at a fixed symbol rate, Equation shows that, as k is increased, the data rate R is increased. In the case of MPSK, increasing k, thereby results in an in creased bandwidth efficiency R/W measured in bits/s/Hz. For example, movement along line 3, from point e to point fin Figure b, represents trading E_b/N_o for a re duced bandwidth requirement. In other words, with the same system bandwidth, one can transmit MPSK signals at an increased date rate and hence at an increased R/W.

SHANNON-HARTLEY CAPACITY THEOREM

Shannon showed that the system capacity C of a channel perturbed by additive white Gaussian noise (AWGN) is a function of the average received signal power S, the average noise power N, and the bandwidth W. The capacity relationship (Shannon—Hartley theorem) can he stated as

$$C = W \log_2 \left(1 + \frac{S}{N}\right)$$

When W is in hertz and the logarithm is taken to the base 2. as shown. the capacity is given in bits/s. It is theoretically possible to transmit information over such a channel at any rate R. where $R \leq C$, with an arbitrarily small error probability by using a sufficiently complicated coding scheme. For an information rate R> C. it is not possible to find a code that can achieve an arbitrarily small error probability.

Shannon’s work showed that the values of S, N, and W set a limit on transmission. rate, not on error probability. Shannon used the above equation to graphically ex hibit a hound for the achievable performance of practical systems. This plot, shown in Figure, gives the normalized channel capacity C/W in bits/s/Hz as a function of the channel signal-to-noise ratio (SNR). A related plot, shown in Figure in dicates the normalized channel bandwidth W/C in Hz/bits/s as a function of SNR in the channel. Figure is sometimes used to illustrate the power-bandwidth trade off inherent in the ideal channel. However, it is not a pure trade-off because the detected noise power is proportional to bandwidth:

$N=N_0W$

Substituting and rearranging the new equation becomes:

$$\frac{C}{W} = \log_2\left(1 + \frac{S}{N_0W}\right)$$

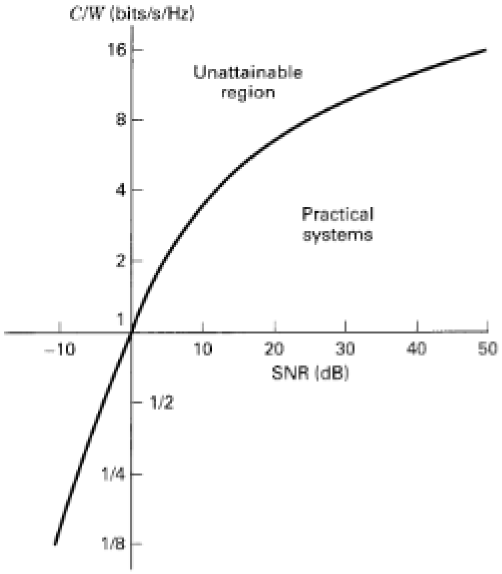


Fig. Normalized channel capacity vs channel SNR.

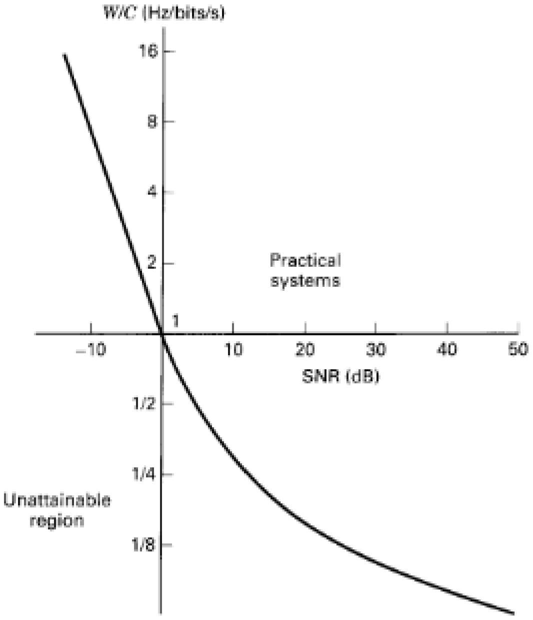


Fig. Normalized channel bandwidth vs channel SNR.

For the case where channel capacity is equal to transmission bit rate, R=C, we can write

$$\frac{S}{N_0 C} = \frac{E_b}{N_0}$$

Hence the original equation can be modified as

$$\frac{C}{W} = \log_2 \left[1 + \frac{E_b}{N_0} \left(\frac{C}{W} \right) \right]$$

$$2^{C/W} = 1 + \frac{E_b}{N_0} \left(\frac{C}{W} \right)$$

$$\frac{E_b}{N_0} = \frac{W}{C} (2^{C/W} - 1)$$

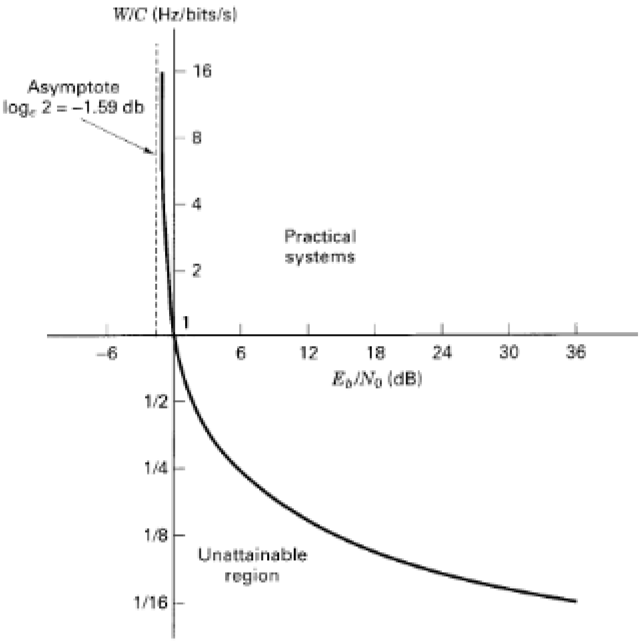
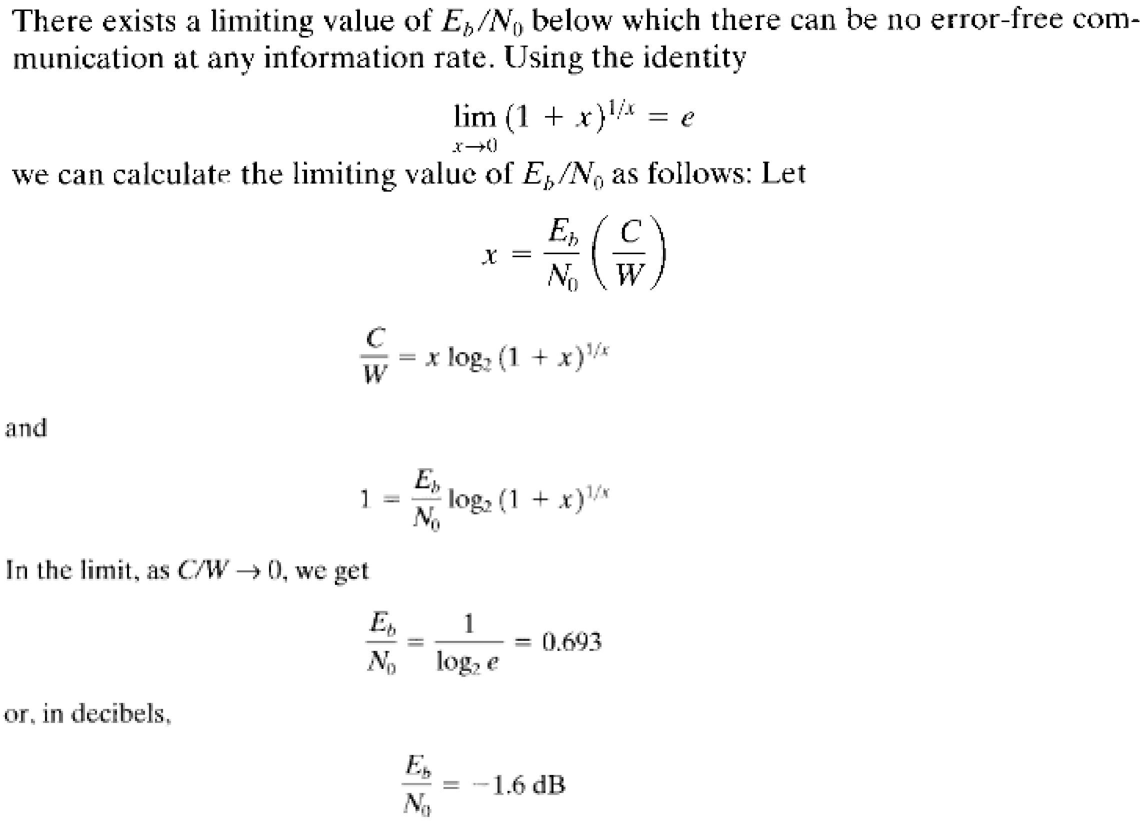


Fig. Normalized channel bandwidth vs channel E_b/N_0.

SHANNON LIMIT



There exists a limiting value of E_b/N_0 below which there can be no error-free communication at any information rate. Using the identity

$$\lim_{x\to 0}(1+x)^{1/x}=e$$

we can calculate the limiting value of E_b/N_0 as follows: Let

$$x=\frac{E_b}{N_0}\left(\frac{C}{W}\right)$$

$$\frac{C}{W}=x\log_2(1+x)^{1/x}$$

and

$$1=\frac{E_b}{N_0}\log_2(1+x)^{1/x}$$

In the limit, as $C/W\to 0$, we get

$$\frac{E_b}{N_0}=\frac{1}{\log_2 e}=0.693$$

or, in decibels,

$$\frac{E_b}{N_0}=-1.6\text{ dB}$$

This value of E_b/N_0 is called the Shannon limit. On Figure the Shannon limit is the P_B, versus E_b/N_0 curve corresponding to k → ∞. The curve is discontinuous going from a value of P_B =1/2 to P_B=0 at E_b/N_0=—l.6 dB. It is not possible in practice to reach the Shannon limit, because as k increases without bound, the bandwidth requirement and the implementation complexity increases without bound. Shannon’s work provided a theoretical proof for the existence of codes that could improve the P_B performance or reduce the E_b/N_0 required, from the levels of the uncoded binary modulation schemes to levels approaching the limiting curve. For a bit error probability of 10^{-5} binary phase-shift-keying (13l’SK) modulation requires an E_b/N_0 of 9.6 dB (the optimum uncoded binary modulation). Therefore for this case, Shannon’s work promised the existence of a theoretical performance improvement of 11.2 dB over the performance of optimum uncoded binary modulation, through the use of coding techniques. Today, most of that promised improvement (as much as 10 dB) is realizable with turbo codes. Optimum system design can best he described as a ‘‘arch for rational compromises or trade-offs among the various constraints and conflicting goals. The modulation and coding trade-off, that is, the selection of modulation and coding techniques to make the best use of transmitter power and channel bandwidth, is important, since there are strong incentives to reduce the cost of generating power and to conserve the radio spectrum.

BANDWIDTH-EFFICIENCY PLANE

Using Equation above, we can plot normalized channel bandwidth W/C in Hz/bits/s versus E_B/N_0 as shown in Figure above. Here, with the abscissa taken as E_B/N_0, We see the true power-bandwidth trade-off at work. It can be shown that well-designed systems tend to operate near the “knee” of this power-bandwidth trade off curve for the ideal (R = C) channel. Actual systems are frequently within 10 dB or less of the performance of the ideal. The existence of the knee means that systems seeking to reduce the channel bandwidth they occupy or to reduce the signal power they require must make an increasingly unfavorable exchange in the other parameter. For example, from Figure, an ideal system operating at an E_B/N_0 of 1.8 dB and using a normalized bandwidth of 0.5 Hz/bits/s would have to increase E_B/N_0 to 20 dB to reduce the bandwidth occupancy to 0.1 Hz/bits/s. Trade-offs in the other direction are similarly inequitable.

Using Equation above we can also plot C/W versus E_b/N_o. This relationship is shown plotted on the R/W versus E_b/N_o plane in Figure. We shall denote this plane as the bandwidth-efficiency plane. The ordinate R/W is a measure of how much data can he communicated in a specified bandwidth within a given time: it therefore reflects how efficiently the bandwidth resource is utilized. The abscissa is E_b/N_o, in units of decibels. For the case in which R = C in Figure, the curve represents a boundary that separates a region characterizing practical communication systems from a region where such communication systems are not theoretically possible. Like Figure, the bandwidth-efficiency plane in Figure sets the limiting performance that can be achieved by practical systems. Since the abscissa in Figure is E_b/N_o rather than SNR. Figure is more useful for comparing digital communication modulation and coding trade-offs than is Figure. Note that Fig. illustrates bandwidth efficiency versus E_b/N_o for single-carrier systems. For multiple-carrier systems, bandwìdth efficiency is also a function of carrier spacing (which depends on the modulation type). The trade-off becomes how closely the carriers can be spaced without suffering an unacceptable amount of adjacent channel interference (ACI).

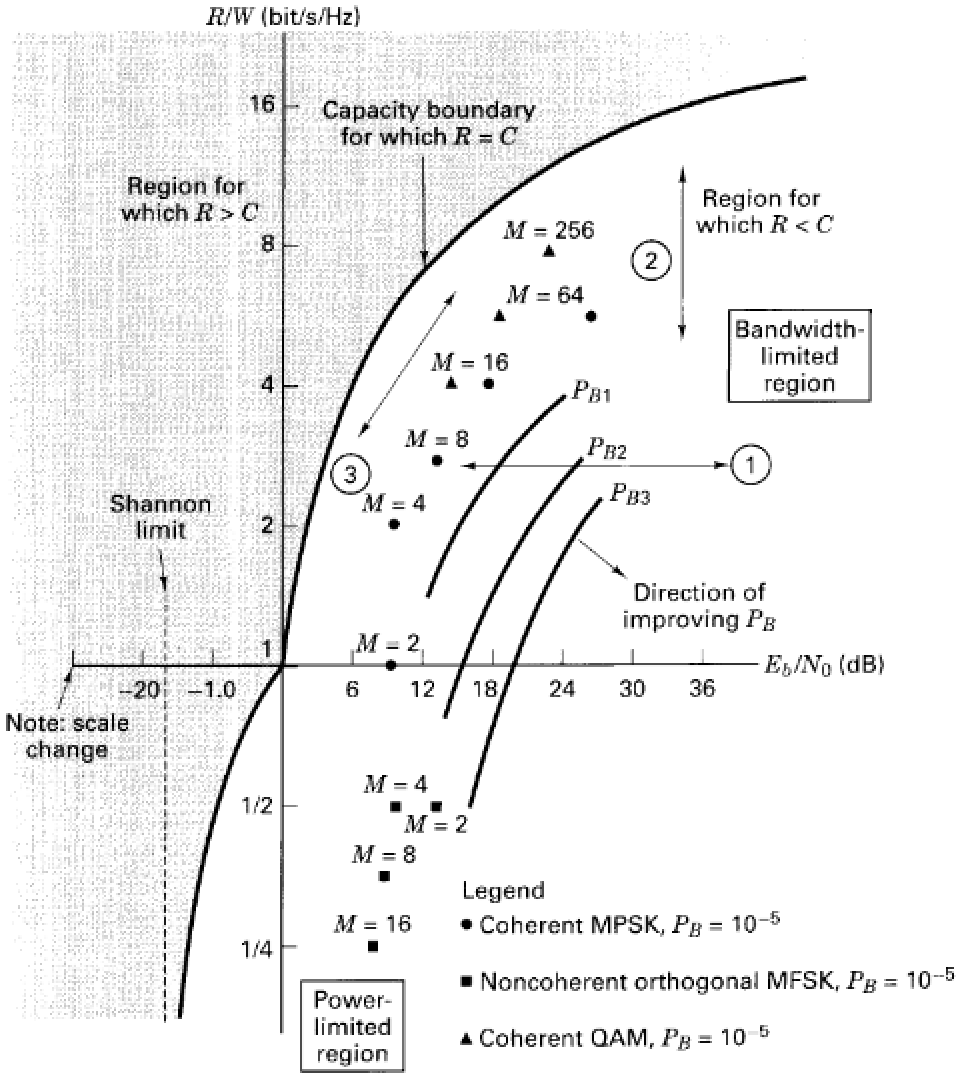


Fig. Bandwidth-efficiency Plane.

MODULATION AND CODING TRADE-OFFS

Figure below is useful in pointing out analogies between the two performance planes. The error-probability plane of earlier Figure and the bandwidth-efficiency plane of Figure. Figure a and b represent the same planes respectively. They have been redrawn as symmetrical by choosing appropriate scales. In each case the arrows and their labels describe the general effect of moving an operating point in the direction of the arrow by means of appropriate modulation and coding techniques. The notations G, C, and F stand for the trade-off considerations “Gained or achieved,” “Cost or expended” and “Fixed or unchanged” respectively. The parameters being traded are P_B, W, R/W, and P (power or S/N). Just as the movement of an operating point toward the Shannon limit in Figure can

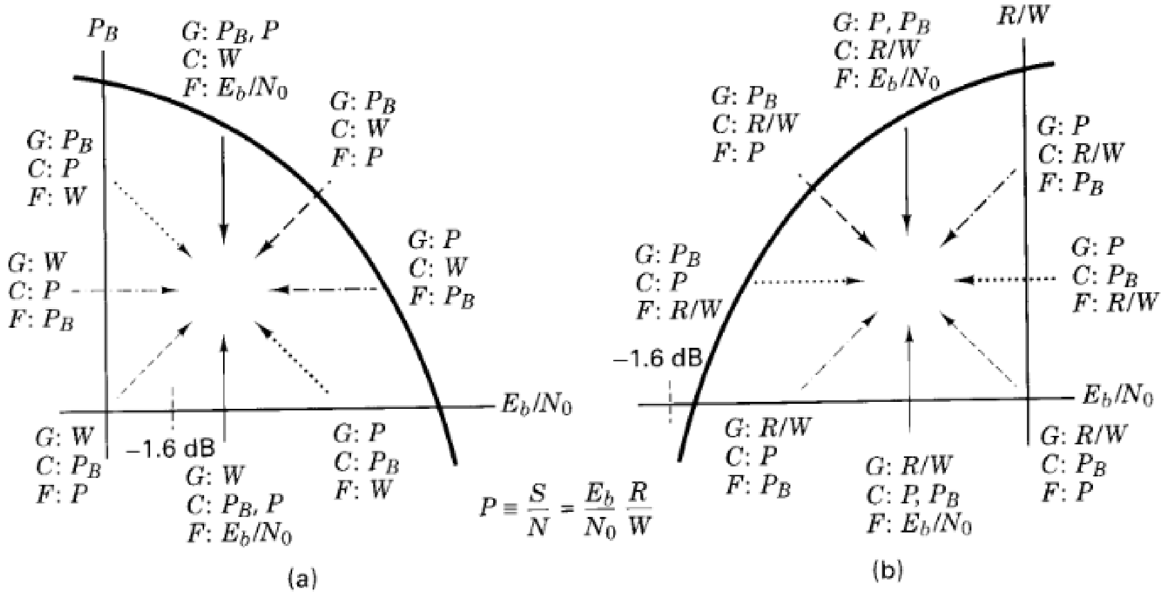


Fig. Modulation/Coding trade-off. a) Error probability plane, b) Bandwidth efficiency plane.

achieve improved P_B or reduced required transmitter power at the cost of bandwidth, so too movement toward the capacity boundary in Figure can improve bandwidth efficiency at the cost of increased required power or degraded P_B . Most often these trade-offs are examined with a fixed P_B, in mind. Therefore, the most interesting arrows are those having bit error probability (marked F: P_B). There are tour such arrows on Figure, two on the error probability plane and two on the bandwidth-effieiencs plane. Arrows marked with the same pattern indicate correspondence between the two planes. System operation can he characterized by either of these two planes.
The planes represent two ways of looking at some of the key system parameters: each plane highlights slightly different aspects of the overall design problem. The error probability plane tends to be most useful with power–limited systems whereas when we move from curve to curve, the bandwidth requirements are only inferred. while the bit error probability is clearly displayed. The bandwidth-efficiency plane is generally more useful for examining bandwidth-limited systems, here, as we move from curve to curve, the bit-error probability is only inferred, but the bandwidth requirements are explicit. The two system trade-oft planes, error probability and bandwidth efficiency.

DEFINING, DESIGNING, AND EVALUATING DIGITAL COMMUNICATION SYSTEMS

The criteria for choosing modulation and coding schemes, based on whether a system is bandwidth limited or power limited are reviewed for several system examples.

The design of any digital communication system begins with a description of the channel (received power, available bandwidth, noise statistics and other impairments, such as fading), and a definition of the system requirements (data rate and error performance). Given the channel description, we need to determine design choices that best match the channel and meet the performance requirements. An orderly set of transformations and computations has evolved to aid in characterizing a system's performance. Once this approach is understood, it can serve as the format for evaluating most communication systems. In this section, we deal with real-time communication systems, where the term coded (or uncoded) refers to the presence (or absence) of error-correction coding schemes involving the use of redundant bits and expanded bandwidth.

The details on examples used in this context can be referred from the textbook.

Two primary communications resources are the received power and the available transmission bandwidth. In many communication systems, one of these resources may be more precious than the other, and hence most systems can be classified as either bandwidth limited or power limited. In bandwidth-limited systems, spectrally efficient modulation techniques can be used to save bandwidth at the expense of power whereas in power-limited systems, power-efficient modulation techniques can be used to save power at the expense of bandwidth. In both bandwidth-and power-limited systems. error- correction coding (often called channel coding) can be used to save power or to improve error performance at the expense of bandwidth. Trellis-coded modulation (TCM) schemes have been used to improve the error performance of bandwidth-limited channels without any increase in bandwidth.

BANDWIDTH-EFFICIENT MODULATION

The primary objective of spectrally efficient modulation techniques is to maximize bandwidth efficiency. The increasing demand for digital transmission channels has led to the investigation of spectrally efficient modulation techniques to maximize bandwidth efficiency and thus help ameliorate the spectral congestion problem.

Some systems have additional modulation requirements besides spectral efficiency. For example, satellite systems with highly nonlinear transponders require a constant envelope modulation. This is because the nonlinear transponder produces extraneous sidebands when passing a signal with amplitude fluctuations (due to a mechanism called AM-to-PM conversion). These sidebands deprive the information signals of sonic of their portion of transponder power, and also can interfere with nearby channels (adjacent channel interference) or with other communication systems (co-channel interference). Offset QPSK (OOPSK) and Minimum shift keying (MSK) are two examples of constant envelope modulation schemes that are attractive for systems using nonlinear transponders.

Details on QPSK and OQPSK can be referred from the textbook.

MODULATION AND CODING FOR BANDLIMITED CHANNELS

Recently, however, there has been considerable interest in techniques that can provide coding gain for bandlimited channels. The motivation is to enable the reliable transmission of higher data rates over voice-grade channels. The potential gain is about 3 bits/symbol (for a given signal-to-noise ratio) or, alternatively, a given error performance could be achieved with a power savings of 9 dB.

The greatest interest is in the following three separate coding research areas:

1. Optimum signal constellation boundaries (choosing a closely packed signal subset from any regular array or lattice of candidate points)

2. Higher-density lattice structures (adding improvement to the signal subset choice by starting with the densest possible lattice for the space)

3. Trellis-coded modulation (combined modulation and coding techniques for obtaining coding gain for bandlimited channels)

The first two areas are not ‘true” error control coding schemes. By true error control coding” we refer to those techniques that employ some structured redundancy to improve the error performance. Only the third technique, trellis-coded modulation, invokes redundancy. An example on commercial telephone modems is given in the text book.

TRELLIS-CODED MODULATION

The error-correction codes when used in real-time communication systems, provide improvements in error performance at the cost of bandwidth expansion. For both block codes and convolutional codes, transforming each input data k-tuple into a larger output codeword n-tuple requires additional transmission bandwidth. Therefore, in the past coding generally was not popular for bandlimited channels such as telephone channels, where signal bandwidth expansion is not practical. Since about 1984, however, there has been active interest in combined modulation and coding schemes, called trellis-coded modulation (TCM), that achieve error-performance improvements without expansion of signal bandwidth. TCM schemes use redundant nonbinary modulation in combination with a finite—state machine (the encoder). What is a finite-state machine, and what is meant by its state? Finite-state machine is the general name given to a device that has a memory of past signals; the adjective finite refers to the fact that there are only a finite number of unique states that the machine can encounter. What is meant by the state of a finite—state machine? In the most general sense, the state consists of the smallest amount of information that together with a current input to the machine, can predict the output of the machine. The state provides some knowledge of the past signaling events and the restricted set of possible outputs in the future. A future state is restricted by the past state.

For each symbol interval, a TCM finite-state encoder selects one of a set of waveforms, thereby generating a sequence of coded waveforms to be transmitted. The noisy received signals are detected and decoded by a soft-decision maximum-likelihood detector/decoder. In conventional systems involving modulation and coding, it is common to separately describe and implement the detector and the decoder. With TCM systems, however, these functions must be treated jointly. Coding gain can he achieved without sacrificing data rate or without increasing either bandwidth or power. At first, it may seem that this statement violates sonic basic principle of power-bandwidth, error-probability trade-off. However, there is still a trade-off involved, since TCM achieves coding gain at the expense of decoder complexity.

TCM combines a multilevel/phase modulation signaling set with a trellis coding scheme. The term “trellis-coding scheme” refers to any code system that has memory (a finite stale machine), such as a convolutional code. Multilevel/phase signals have constellations involving multiple amplitudes, multiple phases, or combinations of multiple amp1itudes and multiple phases. In other words, a TCM signal is best represented by any signal set (greater than binary) whose vector representations can he depicted on a plane. A trellis-coding scheme is one that can be characterized with a state-transition (trellis) diagram, similar to the trellis diagrams describing convolutional codes. Coding gains can he realized with block codes or trellis codes, but only trellis codes will he considered because the availability of the Viterbi decoding algorithm makes trellis decoding simple and efficient. Ungerboeck showed that in the presence of AWGN, TCM schemes can yield net coding gains of about 3-dB relative to uncoded systems with relative ease, while gains of about 6-dB can he achieved with greater complexity. More on TCM can be referred from the text book.

MODULE III: Introduction to Security

- INTRODUCTION:

(i) What is to be protected?

(ii) What are the likely pitfalls?

(iii) What can happen if we don't set up the right security policies, framework and technology implementations?

- NEED FOR SECURITY/ SECURITY GOALS:

(i) Digitization: Information storage is nowadays electronic, hence, it becomes crucial.

(ii) Confidentiality: information hidden from unauthorized access.

(iii) Integrity: information protected from unauthorized change.

(iv) Availability: information available to an authorized entity when needed.

- SECURITY APPROACHES:

Trusted system: A computer system that can be trusted to a specified extent to enforce a specified security policy.

(a) Earlier they were the primary interest of the military.

(b) Banking

(c) Finance

- SECURITY MODELS:

(i) No security: Simplest case to implement no security at all.

(ii) Security through Obscurity: System is aloof or hidden from the world.

(iii) Host security: Security for each host is enforced individually.

(iv) Network Security: Control of network access to various hosts and their services rather than individual host.

- ASPECTS OF GOOD SECURITY POLICY OR SECURITY MANAGEMENT PRACTICES:

(i) Affordability: cost effectiveness.

(ii) Functionality: mechanism of providing security.

(iii) Cultural Issues: policy holding people's expectations, working style and beliefs.

(iv) Legality: policy holding legal requirements.

- POINTS TO BE ENSURED AFTER SECURITY POLICY IS IMPLEMENTED:

(i) Explanation of the policy to all concerned.

(ii) Outline everybody’s responsibilities.

(iii) Use simple language in all communications.

(iv) Accountability should be established.

(v) Provide for exceptions and periodic reviews.

- PRINCIPLES OF SECURITY:

(i) Confidentiality: Only the sender and the intended recipient(s) should be able to access the contents of the message.

(ii) Authentication: It helps in establishing proof of identities. It ensures that the origin of the message is correctly identified.

(iii) Integrity: Content of the message is to be protected.

(iv) Non repudiation: Denial of sent message is to be restricted.

(v) Access Control: Who should be able to access what?

(vi) Availability: Resources should be available to authorized parties at all times.

(vii) Ethical and Legal issues: individual’s right to privacy versus the greater good of a larger entity (e.g. company, society etc.).

- ETHICAL ISSUES:

(a) Privacy: right of individual to control personal information.

(b) Accuracy: responsibility for authenticity, fidelity and accuracy of information.

(c) Property: Find out the owner of the information and control access.

(d) Accessibility: deals with the issue of what information does an organization have the right to collect and the measures to safeguard against any unforeseen eventualities.

- LEGAL ISSUES:

(a) International: International Cybercrime treaty

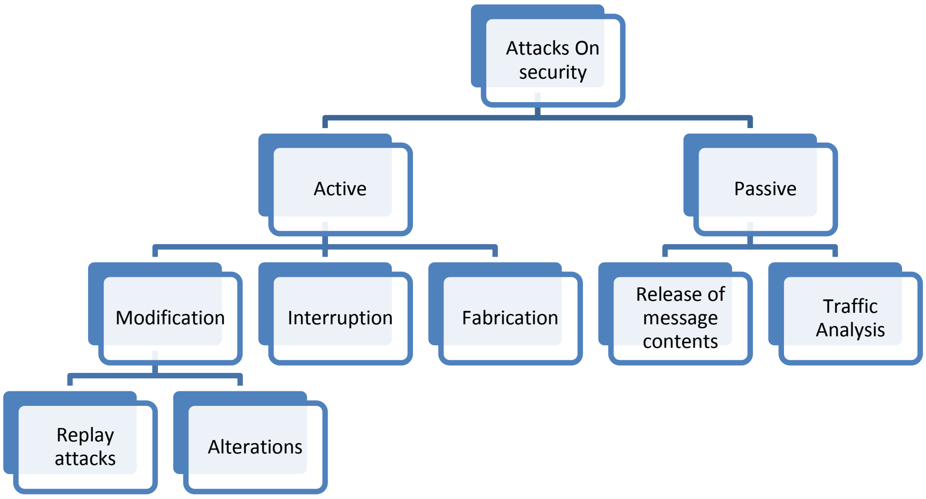
(b) Federal: FERPA, HIPAA, DMCA

(c) State: UCITA, SB 1386

(d) Organization: computer use policy

TYPES OF ATTACKS:

PRINCIPLES	ATTACKS ON SECURITY
Confidentiality	Interception
Integrity	Modification
Authenticity	Fabrication
Non repudiation	----
Availability	Interruption
Access Control	----



ANOTHER ASPECT OF ATTACKS:

(i) Application Level Attack

(ii) Network Level Attack

MECHANISM FOR ABOVE ATTACKS:

(i) Virus

(ii) Worm

(iii) Trojan Horse

(iv) Applets & Active X controls

(v) Cookies

(vi) JavaScript

- CRYPTOGRAPHIC TECHNIQUES

CRYPTOGRAPHY:

It is the art of achieving security by encoding messages to make them non-readable.

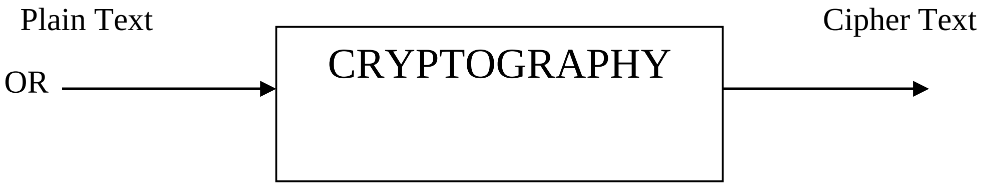
CRYPTANALYSIS:

It 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.

CRYPTOLOGY:

It is a combination of cryptography and cryptanalysis.

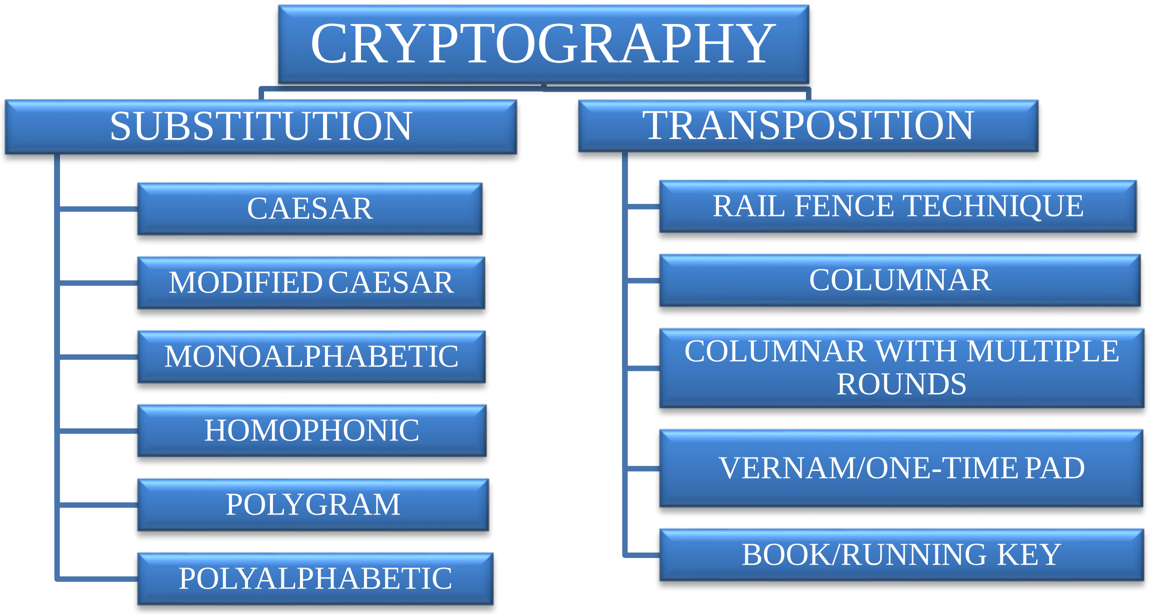
PLAIN TEXT AND CIPHER TEXT:



Clear Text

Plain Text: Message readable by anyone that has access.

Cipher Text: Message readable by only authorized person.



SUBSTITUTION: Changing one character with another without change in position.

TRANSPOSITION: Order or position of characters changes.

PRODUCT CIPHER: When both substitution and transposition are applied together it's called product cipher.

- SUBSTITUTION TECHNIQUES:

(i) Caesar Cipher: Replace each alphabet with an alphabet that is 3 places down from it.

(ii) Modified Caesar Cipher: Replace each alphabet with any alphabet and follow the same rule for other alphabets.

(iii) Monoalphabetic cipher: Replace each alphabet with a random alphabet.

(iv) Homophonic cipher: Replace each alphabet with another random alphabet chosen from a set of alphabets.

(v) Polygram cipher: Replace block of alphabets with another block.

(vi) Polyalphabetic cipher: Replace alphabets with keys.

- TRANSPOSITION TECHNIQUES:

(i) Rail fence:

1. Write down the plain text message as a sequence of diagonals.

2. Read the plain text written in step-1 as a sequence of rows.

(ii) Columnar:

1. Write the plain text message row by row in a rectangle of a predefined size.

2. Read the message column by column in random order.

(iii) Columnar with multiple rounds :

1. Write the plain text message row by row in a rectangle of a predefined size.

2. Read the message column by column in random order.

3. Message obtained for CT of round -1.

4. Repeat steps 1-3 multiple times.

(iv) One time pad or Vernam cipher:

1. Treat each PT alphabet as a number in an increasing sequence (A=0, B=1...Z=25).

2. Do the same for each character of the input CT.

3. Add each number corresponding the PT alphabet to the corresponding input cipher text alphabet number.

4. If the sum thus produced is greater than 26, subtract 26 from it.

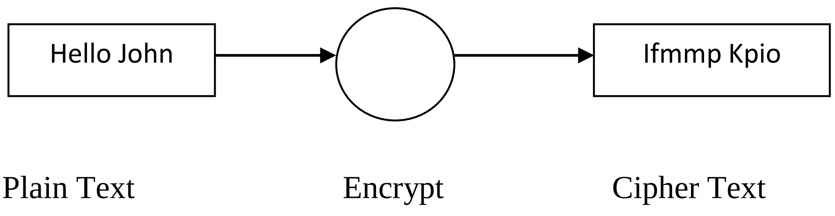
5. Translate each no. of the sum back to the corresponding alphabet. This gives the output CT.

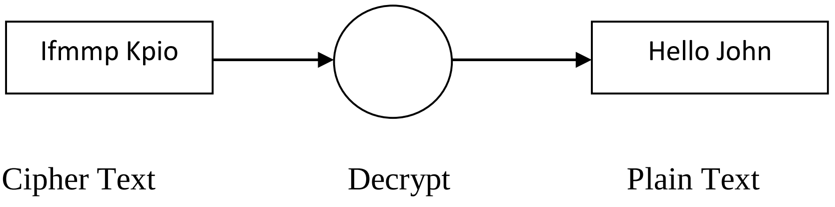
(v) Book cipher or running key cipher:

1. Some portion of text from book is used, that serves the purpose of one-time pad.

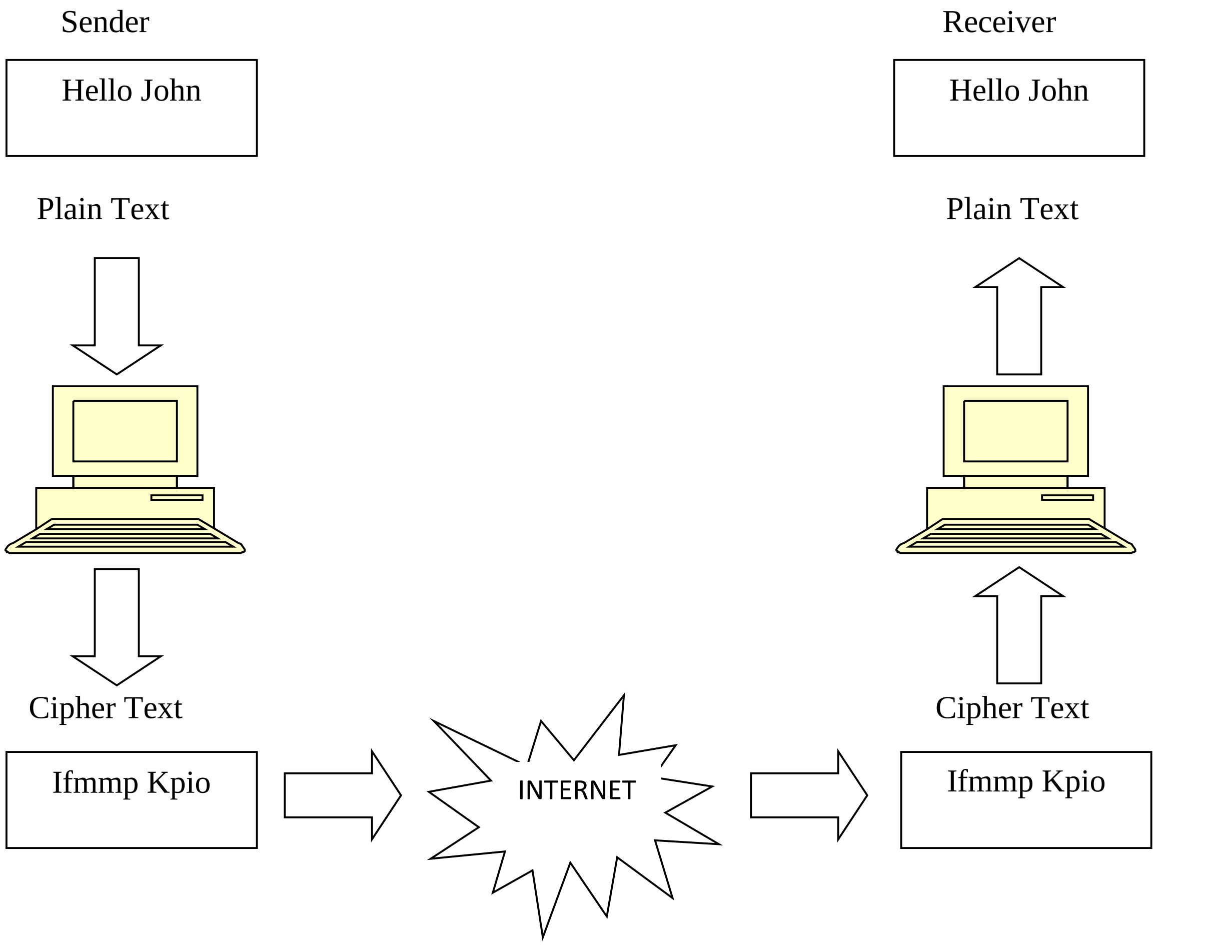
2. Add the one-time pad to the PT to generate CT as in Vernam cipher.

ENCRYPTION & DECRYPTION:

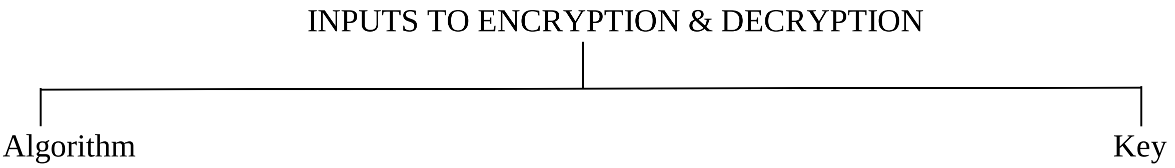




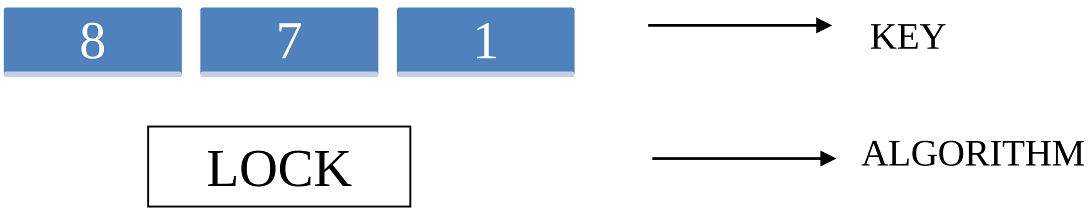
Encryption transforms a plain-text message into cipher text, whereas decryption transforms a cipher text message back into plain text.



Every encryption and decryption process has two aspects: the *algorithm* and the *key* used for encryption and decryption i.e.



Example:



In general, the algorithm used for encryption and decryption processes is usually known to everybody. However, it is the key used for encryption and decryption that makes the process of cryptography secure.

SYMMETRIC & ASYMMETRIC KEY CRYPTOGRAPHY

INPUTS TO ENCRYPTION & DECRYPTION

Symmetric Key Cryptography | Asymmetric Key Cryptography

Symmetric key cryptography involves the use of same key for encryption and decryption whereas Asymmetric key cryptography involves the use of one key for encryption and another different key for decryption.

- PROBLEM OF KEY DISTRIBUTION:

(i) For n persons, the number of lock & key pairs is [n*(n-1)]/2 – very large.

(ii) Record of lock-and-key pair to be maintained by a trustworthy party; duplicate key to be issued in case of missing key which is a time consuming process.

- DIFFIE-HELLMAN KEY-EXCHANGE/AGREEMENT ALGORITHM:

Steps:

1. P & Q agree on two large prime numbers n & g. These need not be kept secret i.e. P & Q can use an insecure channel to agree on them.

2. P chooses another large random number x and calculates A such that:

$A = g^X \bmod n.$

3. P sends the number A to Q.

4. Q independently chooses another large random integer y and calculates B such that:

$B = g^y \bmod n.$

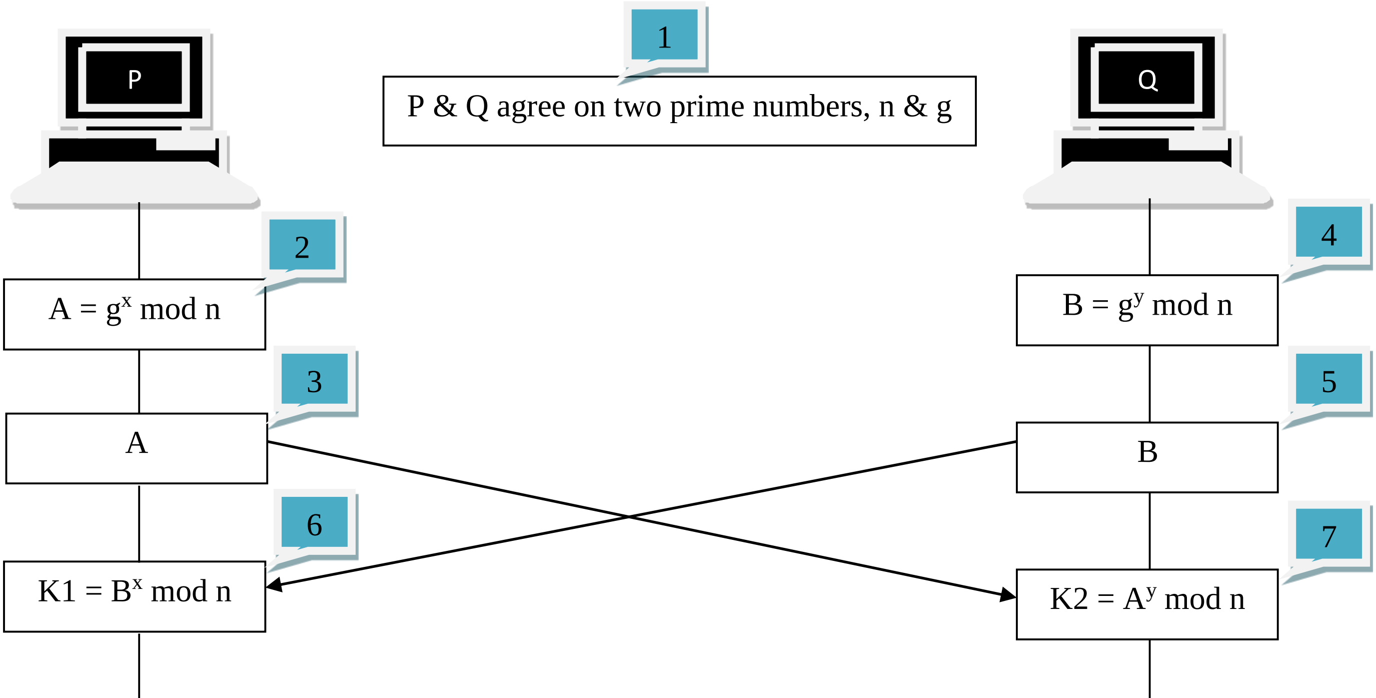
5. Q sends the number B to P.

6. P now computes the secret key K1 as follows:

$K1 = B^x \bmod n.$

7. Q now computes the secret key K2 as follows:

$K2 = A^y \bmod n.$



As it turns out K1=K2=K thus becomes the shared symmetric key between P & Q

- PROBLEMS WITH DIFFIE-HELLMAN ALGORITHM:

(i) The algorithm suffers from man-in-the-middle attack/ bucket-brigade attack.

(ii) The name bucket-brigade comes from the way firefighters of yesteryears formed a line between the fire & water source, and passed full buckets towards the fire & the empty buckets back.

(iii) P wants to communicate with Q securely& sends n, g to Q.

(iv) Attacker T listens to the conversation between P & Q; picks up n, g and forwards n, g as they were to Q.

(v) P, T and Q all select random numbers x & y.

(vi) Based on these values all calculate A & B.

(vii) T intercepts A sent by P and sends his own A to Q instead.

(viii) T intercepts B sent by Q and sends his own B to P instead.

(ix) P calculates K1, Q calculates K2 and T calculates both K1 and K2.

(x) Thus T communicates with P securely using shared symmetric key & on the other hand he communicates with Q securely using a different shared symmetric key. Only then he receives messages from P, views/manipulates them & forwards them to Q and vice-versa. P & Q will think that they are communicating with each other but T would be the man-in-the-middle.

- PREVENTION OF MAN-IN-THE-MIDDLE ATTACK:

--- This attack can be prevented if P & Q authenticate each other before beginning to exchange information which proves to P that Q is indeed Q & not someone else (e.g. T) posing as Q. Similarly Q can also get convinced that P is genuine as well.

- ASYMMETRIC KEY OPERATION:

(i) A & B do not jointly approach T for lock & key pair. Instead, B alone approaches T, obtains a lock and a key(K1) that can seal the lock, and sends the lock and key K1 to A. B tells A that A can use that lock and key to seal the box before sending the sealed box to B.

(ii) B possesses a different but related key (K2) which is obtained by B from T along with the lock and key K1, only which can open the lock.

(iii) It is guaranteed that no other key can open the lock. Since one key (K1) is used for locking & another for unlocking we call this scheme as Asymmetric Key operation.

(iv) T is highly trustworthy & efficient agency by the government.

(v) As K1is meant for locking & is available to the general public, it is called as public key. The other key K2 is strictly held by A as secret/private, thus it is called private key.

(vi) Thus, for n number of users only n public keys & n private keys are required or in total 2n keys are required.

- STEGANOGRAPHY:

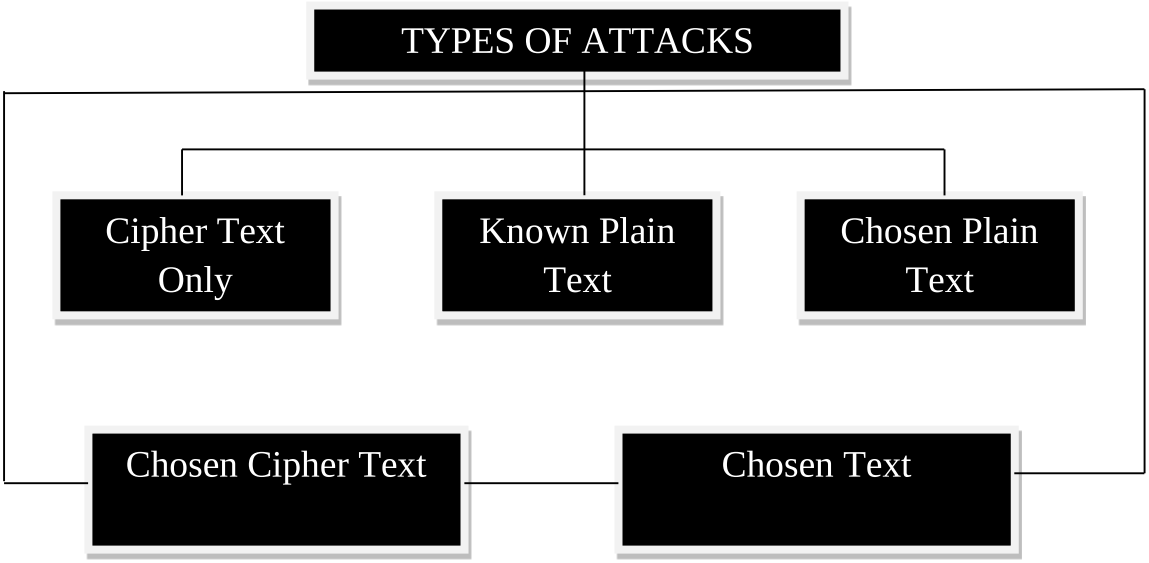
Steganography is a technique that facilitates hiding of a message that is to be kept secret inside other messages. This results in the concealment of the secret message itself.

KEY RANGE & KEY SIZE:

Key Size: (i) It is the measure of strength of a cryptographic key.

(ii) It is measured in terms of bits & represented in binary number system.

POSSIBLE TYPE OF ATTACKS



1. Cipher-Text Only Attack: Here, the attacker doesn't have any clue about the plain text. She has some or all of the cipher text. The attacker analyses the text at leisure to try & find out the original plain text. Based on the frequency of letters, the attacker makes an attempt to guess the plain text. The more text available to the attacker, the more are the chances of a successful attack.

2. Known Plain-Text Attack: In this case, the attacker knows some pairs of plain text & corresponding cipher text for those pairs. Using this information, the attacker tries to find other pairs & therefore know more & more of the plain text.

3. Chosen Plain-Text Attack: Here, the attacker selects a plain-text block and tries to look for encryption of the same in the cipher text. The attacker is able to choose the messages to

encrypt. Based on this, the attacker intentionally picks patterns of cipher text that results in obtaining more information about the key.

4. Chosen Cipher-Text Attack: Here, the attacker knows the cipher text to be decrypted, the encryption algorithm that was used to produce this cipher text, and the corresponding plain text block. The attacker's job is to discover the key used for encryption.

5. Chosen-Text Attack: It is a combination of chosen plain-text attack and chosen cipher-text attack.

- CASE STUDY: DENIAL OF SERVICE (DOS) ATTACKS

(i) Purpose of DOS: To flood/overhaul a network so as to deny the authentic users services of the network.

(ii) Mechanism: A typical mechanism is with the help of SYN requests. On the internet, a client & server communicate using TCP/IP protocol. This involves the creation of a TCP connection between the client & the server, before they can exchange any data.

(a) The client sends a SYN request to the server. A SYN request indicates to the server that the client is requesting for a TCP connection with it.

(b) The server responds back to the client with an acknowledgement, which is technically called as SYN ACK.

(c) The client is then expected to acknowledge the server's SYN ACK.

--- Attacker performs step (a), server performs step (b) but the attacker doesn't perform step(c).The client sends many such SYN requests to the same server & doesn't perform step(c) in any of the requests. Thus a lot of incomplete SYN requests could bring the server to a halt.

--- In step (a) the attacker forges the source address i.e. the attacker puts the source address as the address of a non-existing client. Therefore, when the server executes step (b), the SYN ACK never reaches any client at all, fooling the server.

--- The attacker launches a Distributed DOS attack. Here the attacker sends many SYN requests to the server from many physically different client computers. Thus, even if the server detects DOS attack, it cannot do much by blocking SYN requests coming from a particular IP address- there are many such requests from a variety of forged clients.

(iii) Prevention:

(a) Investigate the incoming packets and look for a particular pattern. If such a pattern emerges, then try blocking incoming packets from the concerned IP addresses.

(b) Configure the services offered by a particular application so that it never accepts more than a particular number of requests in a specified time interval.

(c) Blocking a particular IP address, port number or a combination of such factors can also prevent DOS.

(d) As a precaution, have a backup of the firewall and the servers ready. If the main machine is compromised, it should be quickly brought down & the backup can take its place until a proper clean-up is performed.

Computer-based symmetric key cryptographic key algorithm include DES (and its variations), IDEA, RC5 and Blowfish.

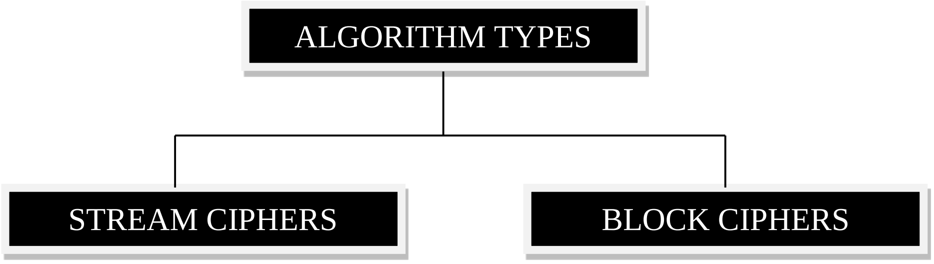
- ALGORITHM TYPES & MODES

Algorithm Type: The size of plain text to be encrypted in each step of the algorithm.

Algorithm Mode: It defines the details of the cryptographic algorithm, once the type is defined.

ALGORITHM TYPES

Cipher text from plain text can be done in two basic ways as:



(i) Stream Ciphers: The plain text is encrypted one bit at a time and decrypted one bit at a time. It relies only on confusion.

(ii) Block Ciphers: The plain text is encrypted one block of text at a time and decrypted a block at a time. It uses both confusion and diffusion.

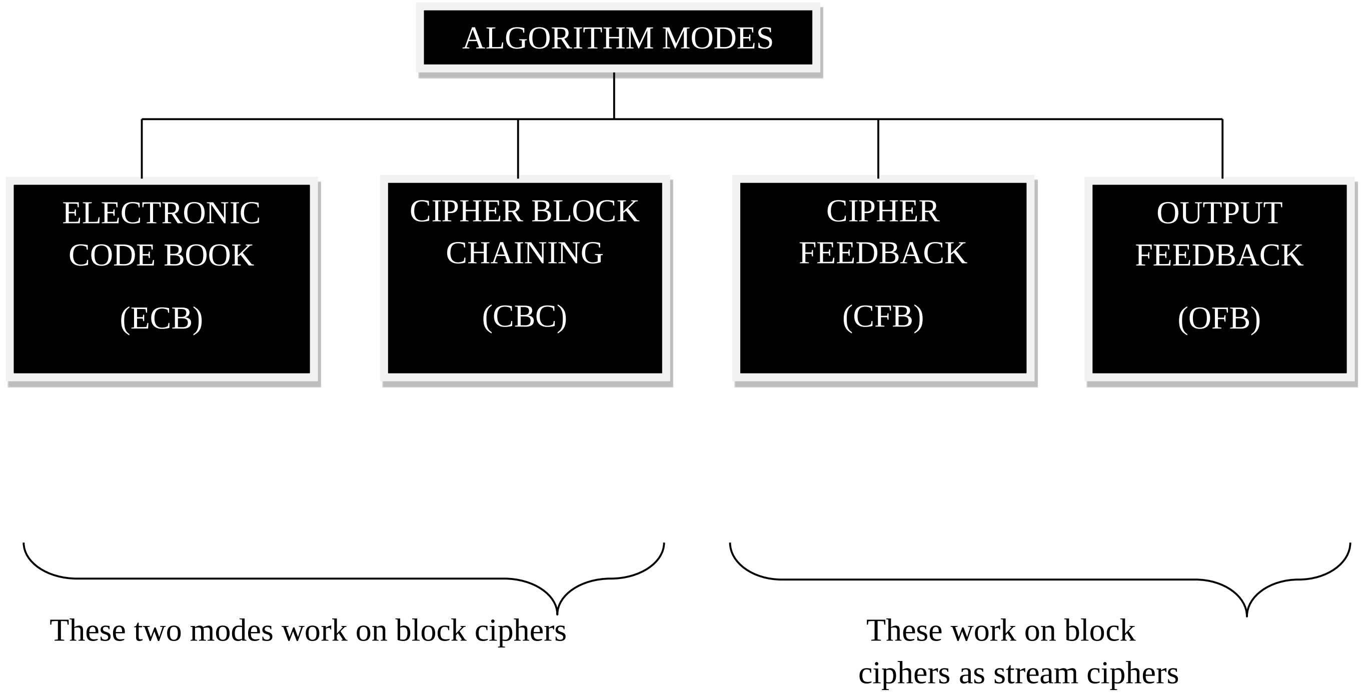
(iii) Grouping: It means how many times the plain text is scrambled in various ways to generate the cipher text.

(iv) Confusion: It is a technique of ensuring that a cipher text gives no clue about the original plain text. It is achieved by substitution techniques.

(v) Diffusion: It increases the redundancy of the plain text by spreading it across rows and columns. It is achieved by transposition/permutation techniques.

- ALGORITHM MODES

An algorithm mode is combination of series of basic algorithm steps on block cipher, and some kind of feedback from the previous step.


ALGORITHM MODES
ELECTRONIC CODE BOOK (ECB)
CIPHER BLOCK CHAINING (CBC)
CIPHER FEEDBACK (CFB)
OUTPUT FEEDBACK (OFB)
These two modes work on block ciphers
These work on block ciphers as stream ciphers

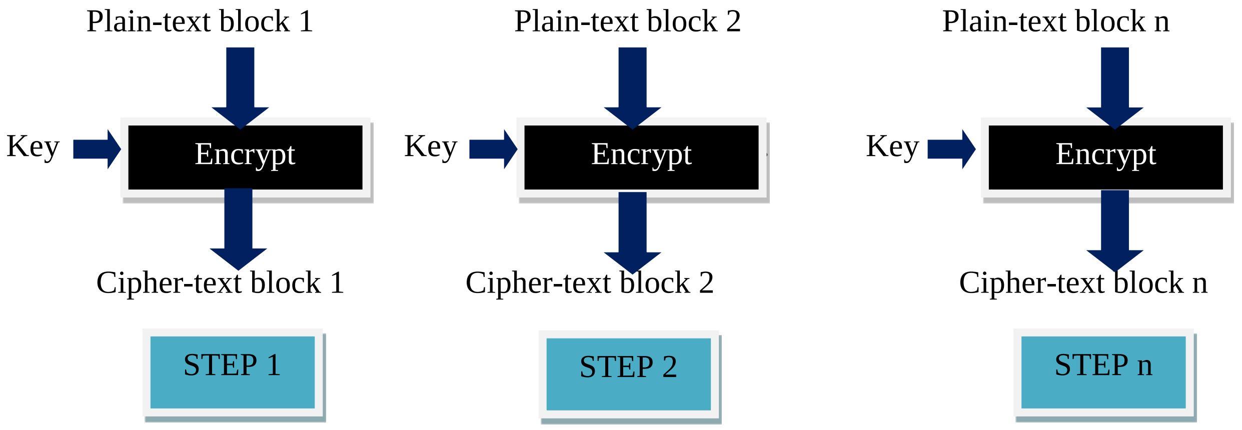
(i) Electronic Code Book (ECB) Mode:

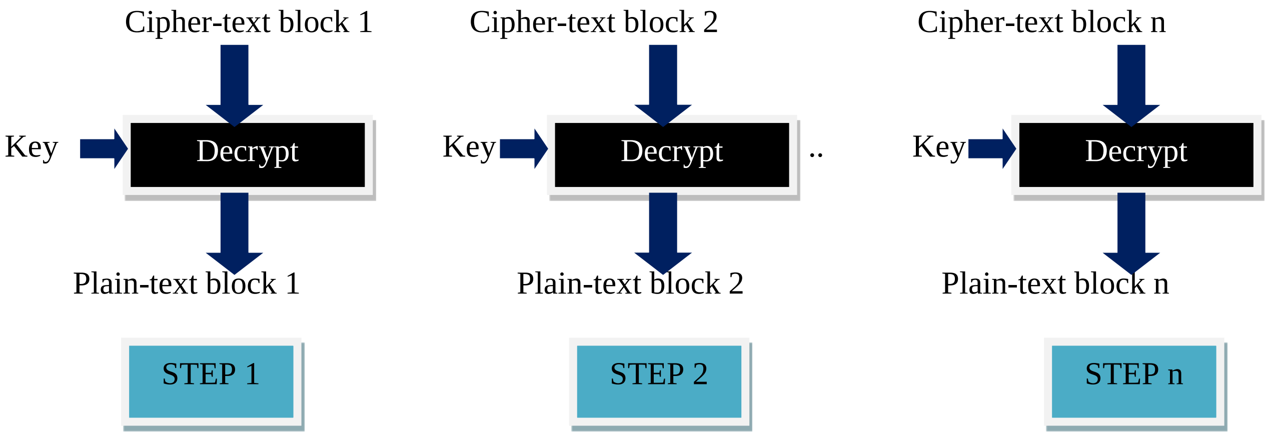
(a) It is the simplest mode of operation.

(b) Plain text message is divided into blocks of 64 bits each.

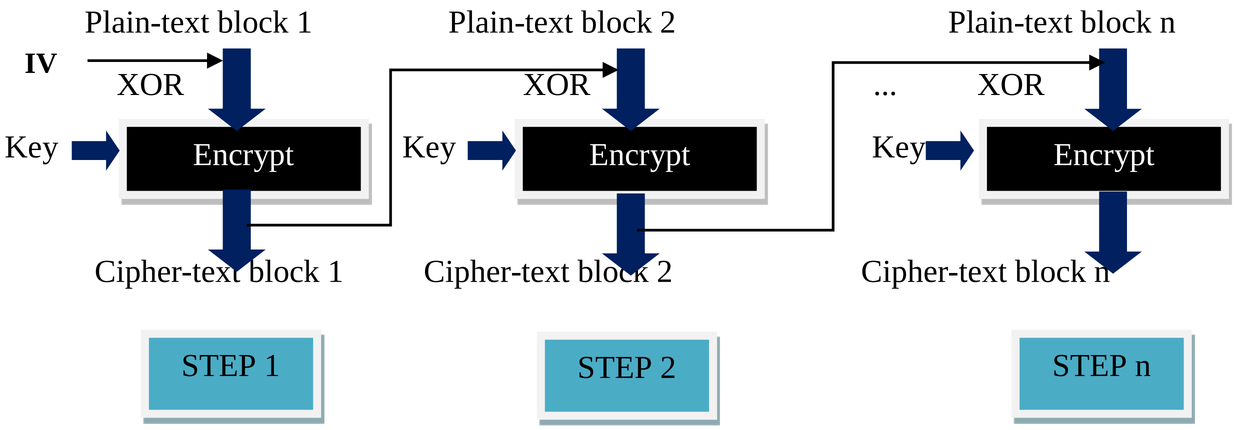
(c) Each such block is then encrypted independently of the other blocks.

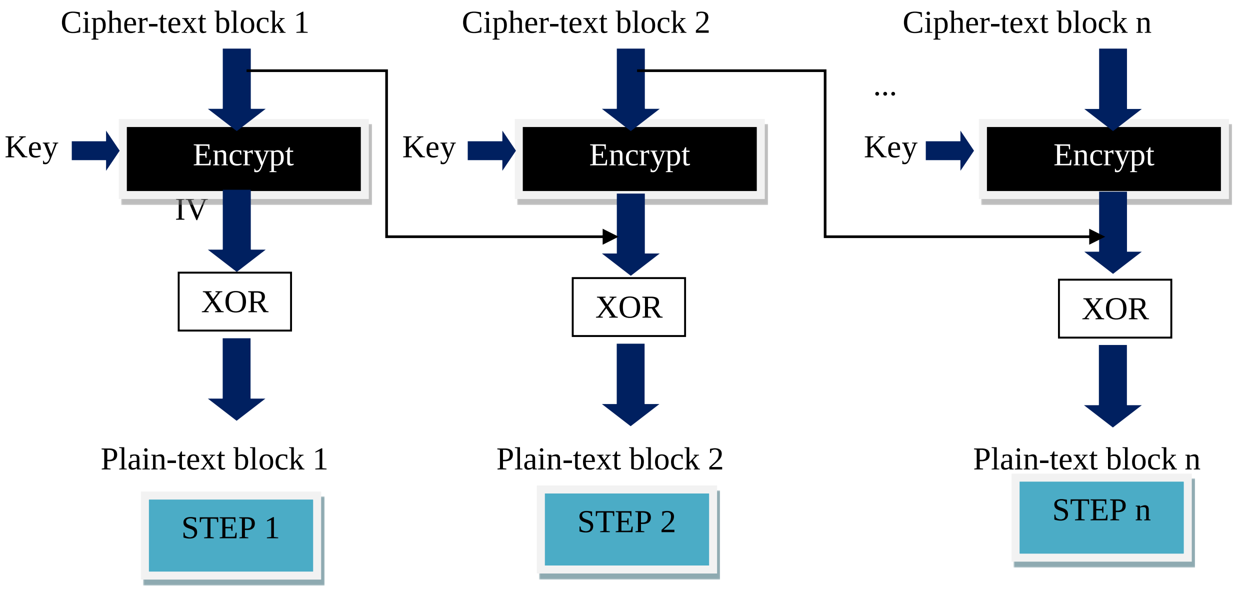
(d) For all blocks in a message, the same key is used for encryption.



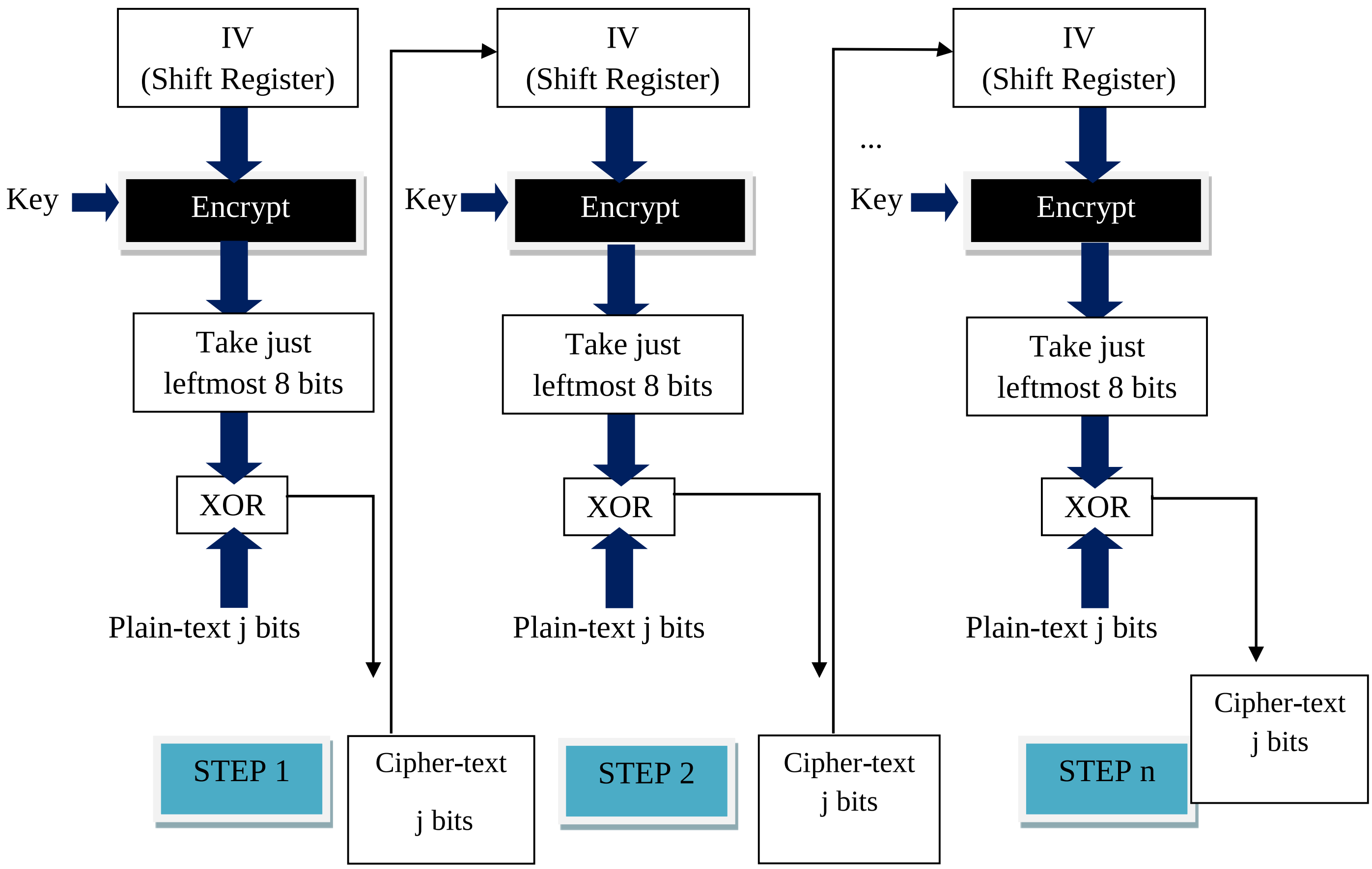


(ii) Cipher Block Chaining (CBC) Mode:

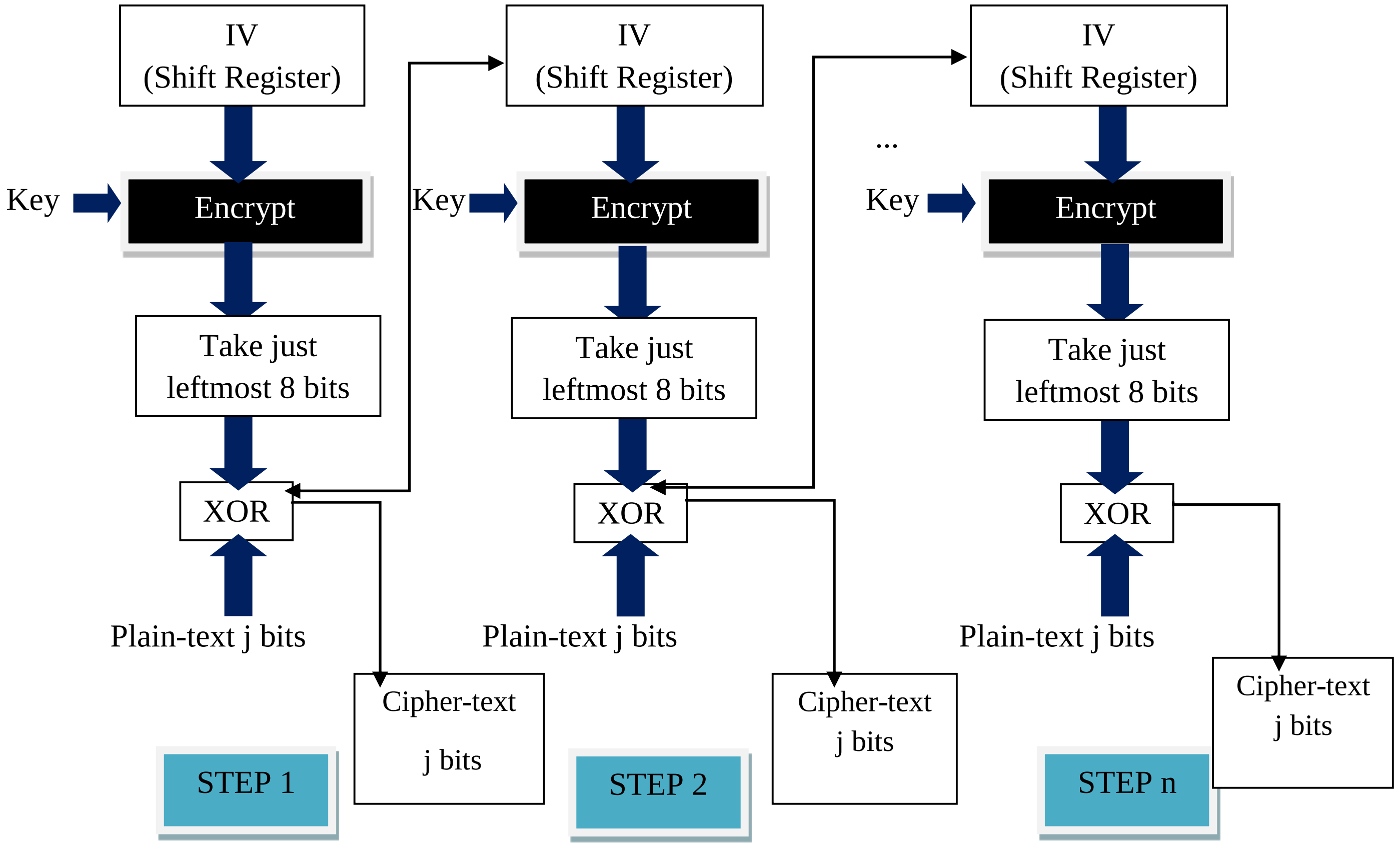




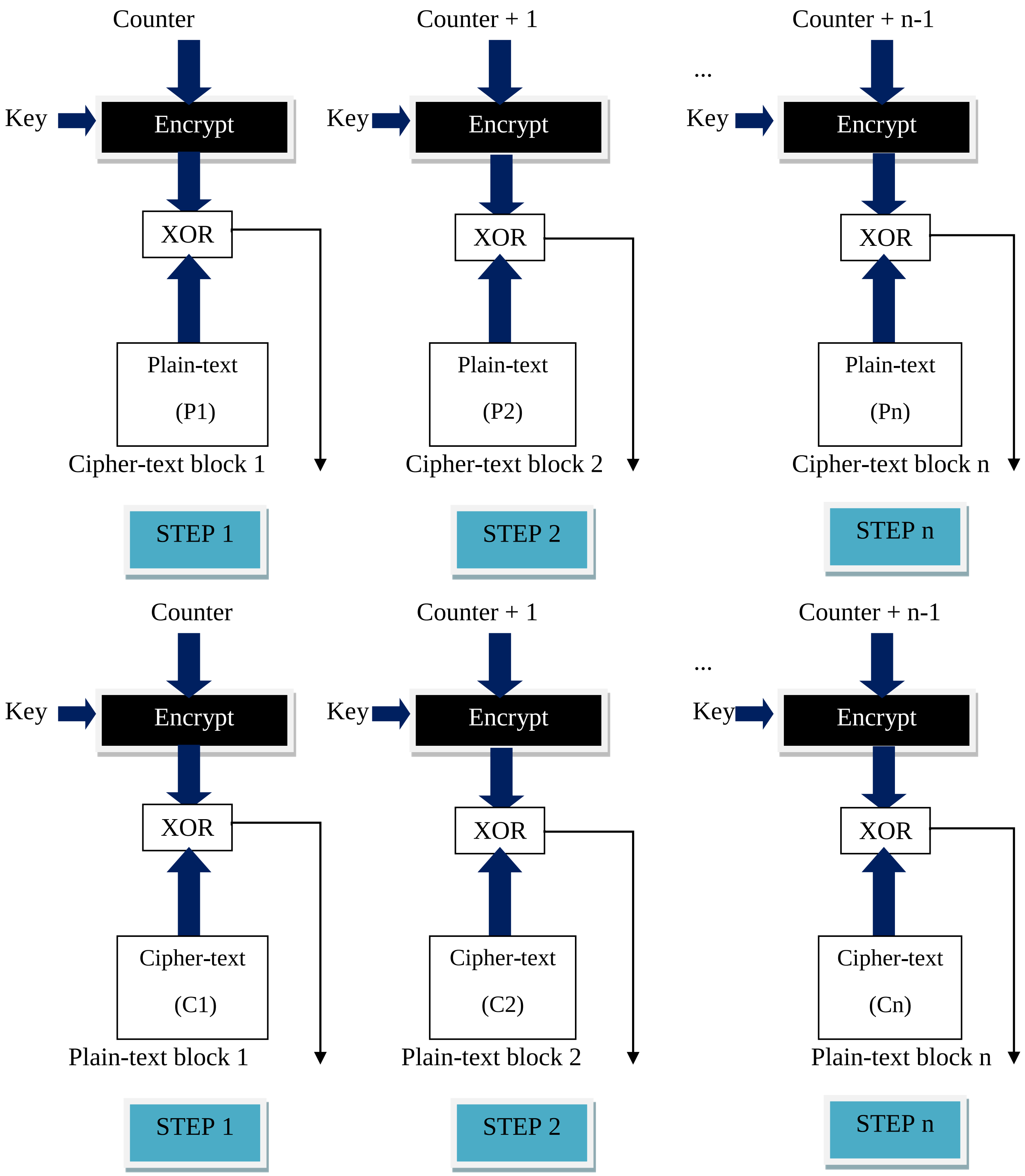
(iii) Cipher Feedback (CFB) Mode:



(iv) Output Feedback(OFB) Mode:



(v) Counter (CTR) Mode:



ALGORITHM MODE	DETAILS	USAGE
ELECTRONIC CODE BOOK (ECB)	The same key independently encrypts blocks of text, 64 bits at a time.	Transmitting a single value in a secure fashion (e.g. password or key used for encryption).
CIPHER BLOCK CHAINING (CBC)	64 bits of cipher text from the previous step and 64 bits of plain text of the next step are XORed together	Encrypting blocks of text Authentication.
CIPHER FEEDBACK (CFB)	K bits of randomized cipher text from the previous step and K bits of plain text of the next step are XORed together.	Transmitting encrypted stream of data Authentication.
OUTPUT FEEDBACK (OFB)	Similar to CFB, except that the input to the encryption step is the preceding DES output.	Transmitting encrypted stream of data.
COUNTER (CTR)	A counter and plain-text block are encrypted together, after which the counter is incremented.	Block-oriented transmissions Applications needing high speed.

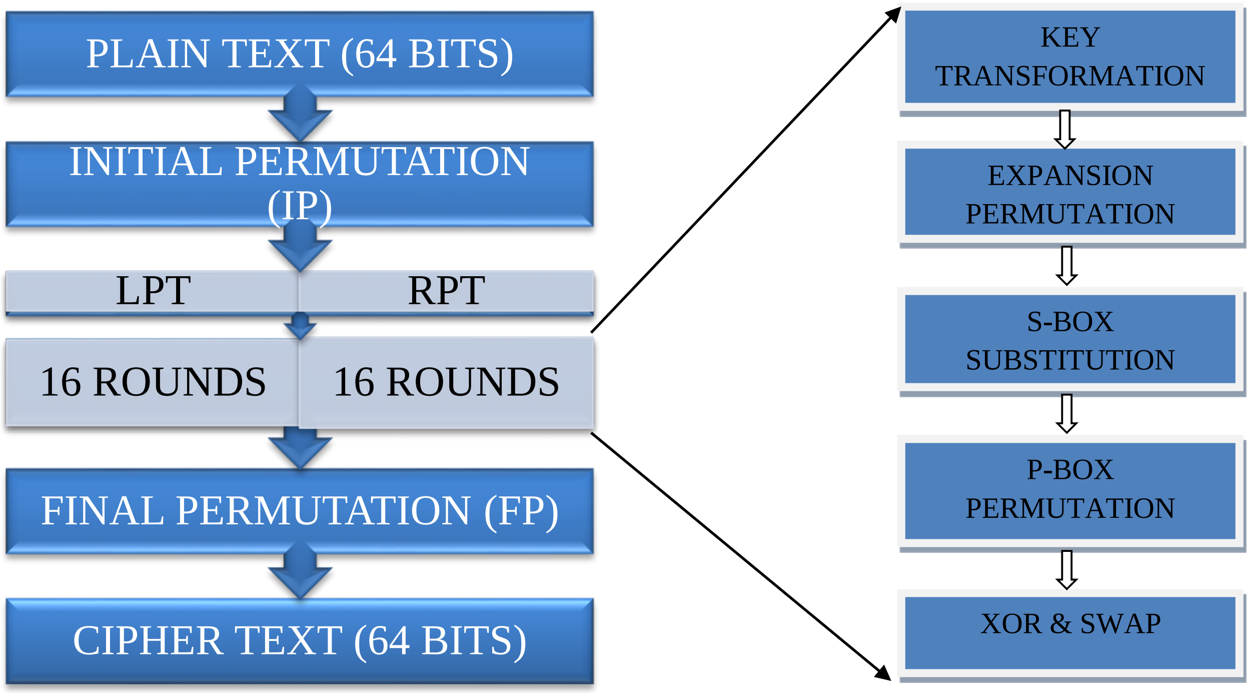
FEATURE	ECB	CBC	CFB	OFB/Counter
Security-related problems	Plain text patterns are not hidden. Input to the block cipher is the same as the plain text, and is not randomized. Plain text is easy to manipulate, blocks of text can be removed, repeated, or exchanged.	Plain-text blocks can be removed from the beginning and end of the message, and bits of the 1st block can be altered.	Plain-text blocks can be removed from the beginning and end of the message, and bits of the 1st block can be altered.	Plain text is easy to manipulate. Altering cipher text alters plain text directly.
Security-related advantages	The same key can be used for encrypting multiple messages.	XOR of plain text with previous cipher-text block hides the plain text. The same key can be used for encrypting multiple messages.	Plain-text patterns are hidden. The same key can be used for encrypting multiple messages, by using different IV. Input to the block cipher is randomized.	Plain-text patterns are hidden. The same key can be used for encrypting multiple messages, by using different IV. Input to the block cipher is randomized.

Problems related to effectiveness	Size of cipher text is more than the plain text size by one padding block. Pre-processing is not possible.	Size of cipher text is more than the plain text size by one block. Pre-processing is not possible. Parallelism cannot be introduced in encryption.	Size of cipher text is same as that of the plain text size. Parallelism cannot be introduced in encryption.	Size of cipher text is same as that of the plain text size. Parallelism cannot be introduced (OFB only).

- OVERVIEW OF SYMMETRIC KEY CRYPTOGRAPHY

Symmetric key cryptography is referred to as Secret key cryptography or private key cryptography. Here only one key is used & the same key is used for both encryption & decryption of messages.

DATA ENCRYPTION STANDARD (DES)



DES WORKING MECHANISM:

STEP-1: BASIC PRINCIPLES

(i) DES is a block cipher.

(ii) It encrypts data in blocks of 64 bits each i.e. 64 bits of plain text goes as input to DES which produces 64 bits of cipher text.

(iii) The same algorithm & key are used for encryption & decryption with minor differences.

(iv) The key length is 56 bits.

(v) The initial key consists of 64 bits; however, before the DES process even starts, every 8th bit of the key is discarded to produce a 56-bit key i.e. bit positions 8,16,24,32,48,56 and 64 are discarded (before discarding, these bits can be used for parity checking to ensure that the key doesn't contain any errors).

STEP-2: INITIAL PERMUTATION (IP):

(i) It happens only once & it happens before the 1st round.

(ii) IP replaces the 1st bit of the original plain-text block with the 58th bit of the original plain-text block, the 2nd bit with the 50th bit & so on as shown below:

Bit position in the plain-text block	To be overwritten with the contents of this bit position
1	58
2	50
3	42
...	...
64	7

(iii) After IP, the resultant 64 –bit permuted text block is divided into two half blocks. Each half block consists of 32 bits. The left block is called LPT & the right block is called RPT.

(iv) 16 rounds are performed on these two blocks.

STEP-3: ROUNDS:

I. KEY TRANSFORMATION:

(i) From the 56-bit key a 48-bit sub key is generated using a process called key transformation.

(ii) For this, the 56-bit key is divided into two halves, each of 28-bits.

(iii) These halves are circularly shifted left by one or two positions, depending on the round.

(iv) For rounds 1,2,9 or 16 the shift is done by only one position whereas for other rounds the circular shift is done by two positions as shown below :

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
No. of key bits shifted	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

After an appropriate shift, 48 of the 56 bits are selected. For selecting 48 of the 56 bits, the following table is used.

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

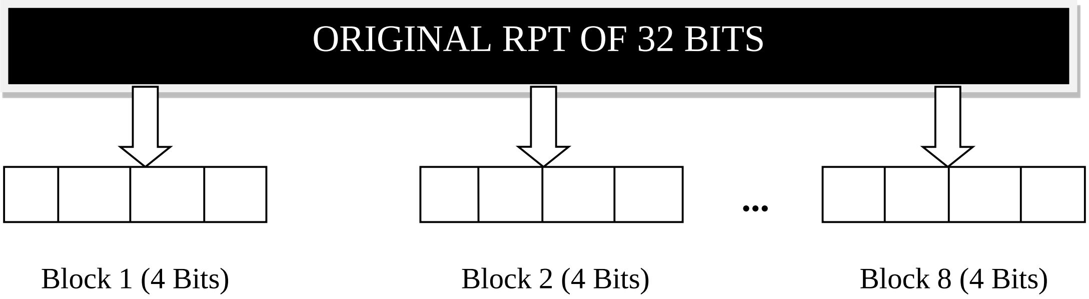
For instance, after the shift, bit no. 14 moves into the 1st position, bit no. 17 moves into 2nd position & so on. Bit no. 18 is discarded like 7 others to reduce the 56-bit key to the 48-bit key. Since the key-transformation process involves permutation as well as selection of a 48-bit subset of the original 56-bit key, it is called *compression permutation*.

Note: Because of the compression permutation technique, a different subset of key bits is used in each round. That makes DES more difficult to crack.

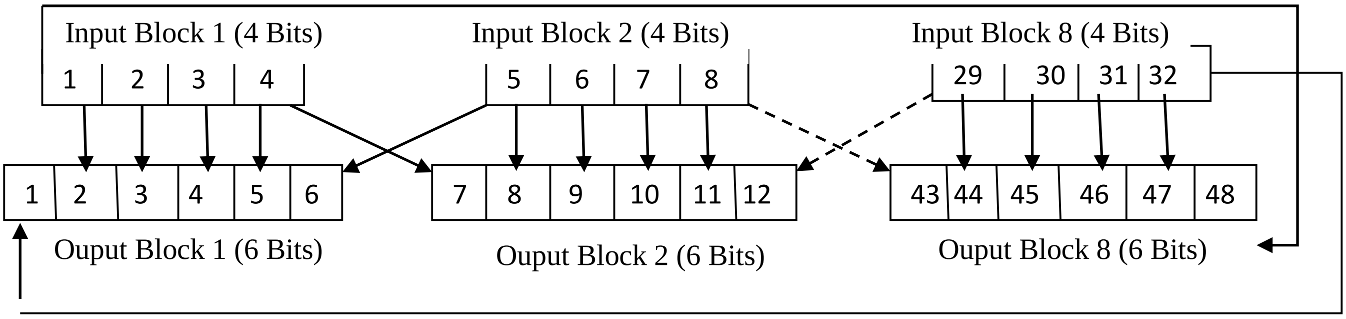
II. EXPANSION PERMUTATION:

(i) During expansion permutation, the RPT is expanded from 32 bits to 48 bits. Besides increasing the bit size from 32 to 48, the bits are permuted as well, hence the name expansion permutation which is done as follows:

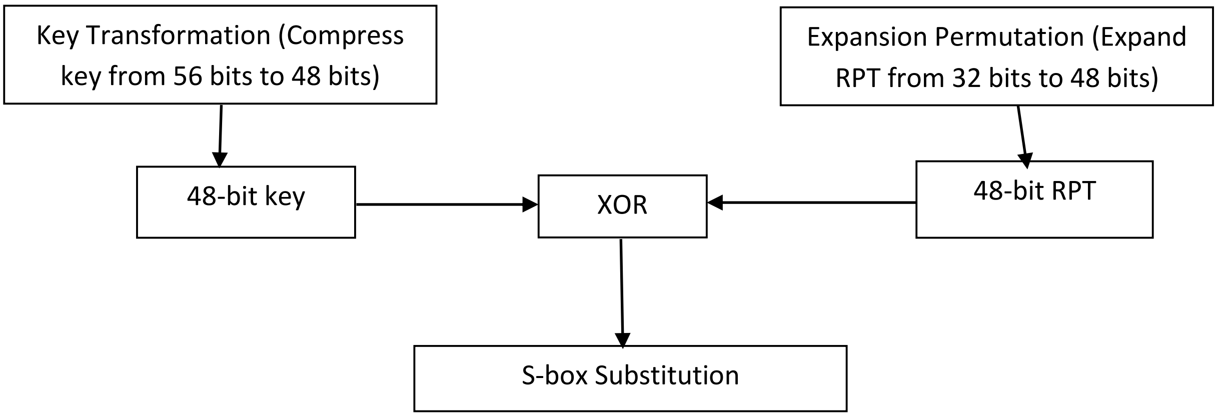
(a) The 32-bit RPT is divided into 8 blocks, with each block consisting of 4 bits.



(b) Next each 4-bit block of the above step is then expanded to a corresponding 6-bit block, i.e. per 4-bit block, 2 more bits are added. These 2 bits are the repeated 1st & 4th bits of the 4-bit block. The 2nd & 3rd bits are written down as they were in the input as shown in the figure below.



(c) Then the expansion permutation process expands the 32-bit RPT to 48 bits. Now, the 48-bit key is XORed with the 48-bit RPT and the resulting output is given to the next step.


Key Transformation (Compress key from 56 bits to 48 bits)
Expansion Permutation (Expand RPT from 32 bits to 48 bits)
48-bit key
XOR
48-bit RPT
S-box Substitution

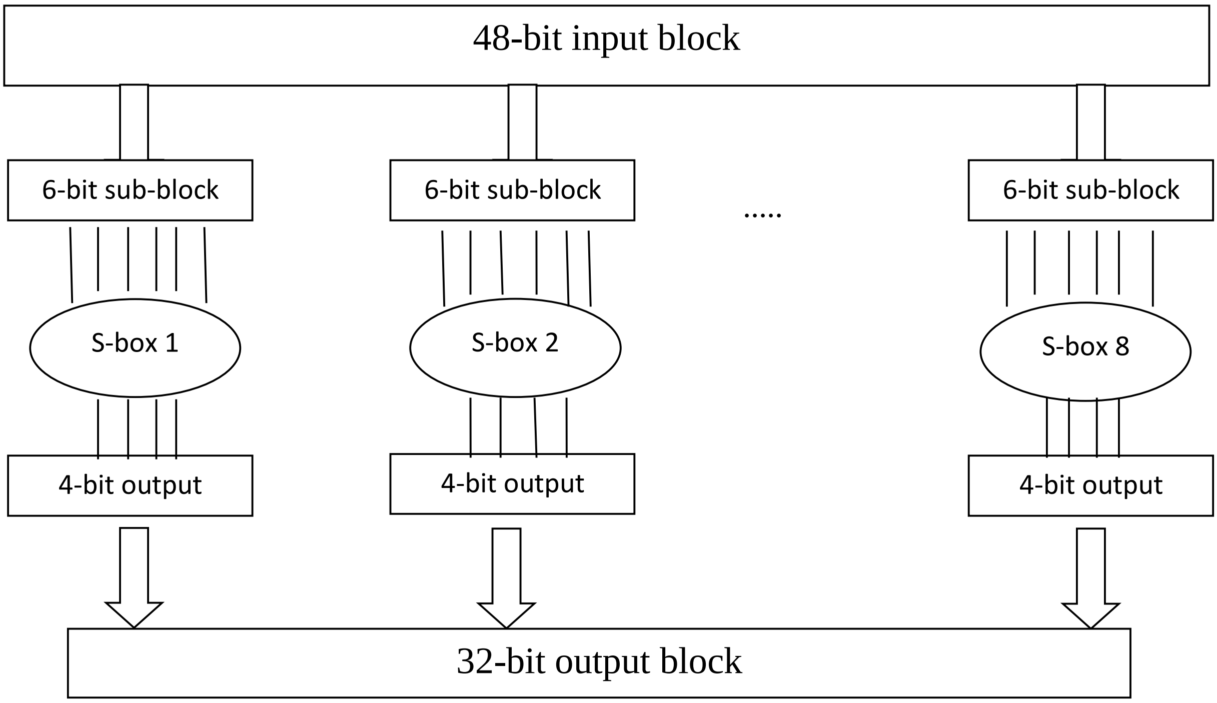
III. S-BOX SUBSTITUTION:

(i) It is the process that accepts 48-bit input from the XOR operation involving the compressed key and expanded RPT, and produces a 32-bit output using the substitution technique. The substitution is performed by 8 substitution boxes (also called S-boxes).

(ii) Each 8 S-boxes has a 6-bit input and a 4-bit output.

(iii) The 48-bit input block is divided into 8 sub-blocks (each containing 6 bits), and each sub-block is given to an S-box.

(iv) The S-box transforms the 6–bit input into a 4-bit output as shown in the figure below.



IV. P-BOX PERMUTATION:

(i) The output of S-box consists of 32 bits.

(ii) These 32-bits are permuted using a P-box. This straightforward permutation mechanism involves simple permutation (i.e. replacement of each bit with another bit, as specified in the P-box table, without any expansion or compression). This is called P-box permutation.

V. XOR AND SWAP:

(i) The left half portion i.e. LPT is XORed with the output produced by P-box permutation. The result of this XOR operation becomes the new RPT. The old RPT becomes the new left half, in a process of swapping.

STEP 4: FINAL PERMUTATION:

(i) At the end of 16 rounds the final permutation is performed (only once).

(ii) It is simple transposition & the output is the 64-bit encrypted block.

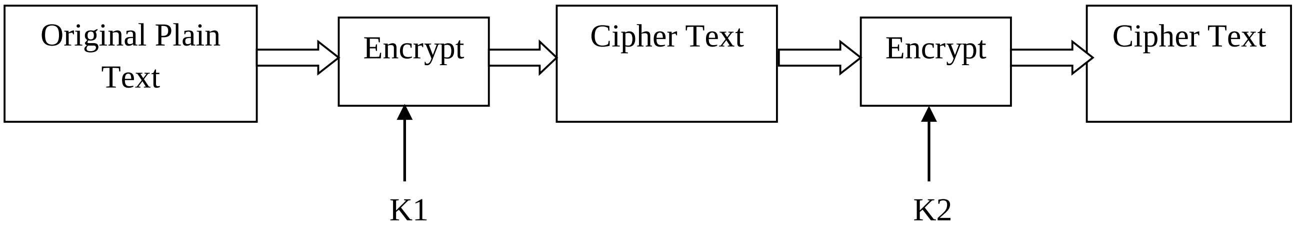
DES Decryption:

Its decryption is similar to encryption where the key positions are reversed for all the rounds.

- VARIATIONS OF DES:

I. DOUBLE DES:

ENCRYPTION:

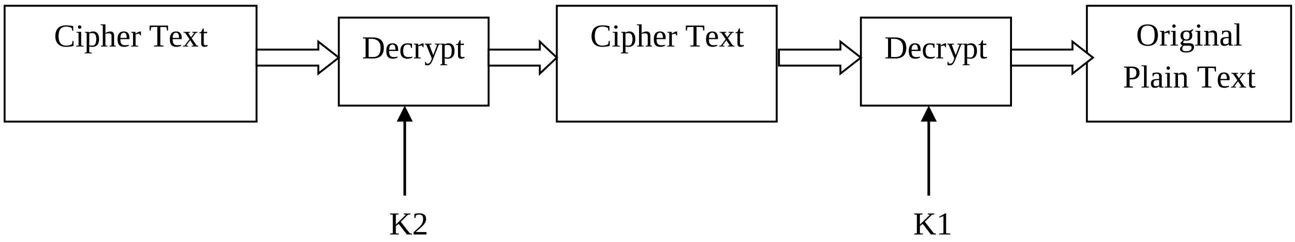


(i) Double DES does what DES normally does only once.

(ii) Double DES has two keys say K1 & K2.

(iii) It 1st performs DES on the original plain text using K1 to get the encrypted text. It then again performs DES on the encrypted text but with the other key K2. The final output is the encryption of encrypted text (i.e. the original plain text encrypted twice with 2 different keys).

DECRYPTION:



(i) The decryption process follows the reverse order; the doubly encrypted cipher-text block is 1st decrypted using the key K2 to produce the singly encrypted cipher text. This cipher-text block is then decrypted using the key K1 to obtain the original plain-text block.

DRAWBACK:

--- It suffers from meet-in-the-middle attack.

Meet-in-the-middle attack:

(i) In the 1st step, the cryptanalyst calculates the value of T or E_{K1}(P) i.e. 1st encryption operation on plain-text block P.

(ii) In the 2nd step, the cryptanalyst finds the value of T from the right-hand side D_{K2}(C).

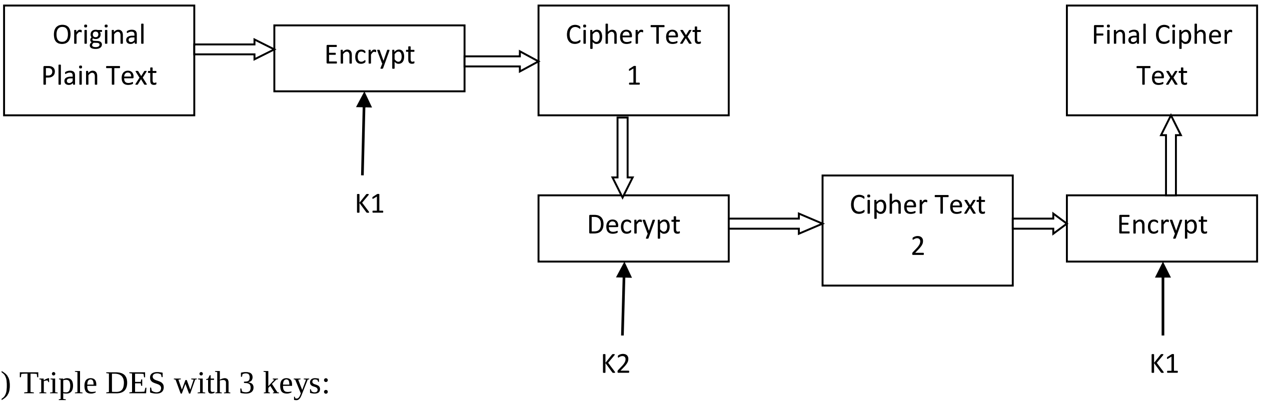
(iii) The cryptanalyst creates a table of E_{K1}(P) for all possible values of K1 and then performs D_{K2}(C) for all possible values of K2; if he gets the same T for both encrypt with K1 and decrypt with K2 operations, he knows P,C,K1and K2.

(iv) This attack is possible but requires a lot of memory.

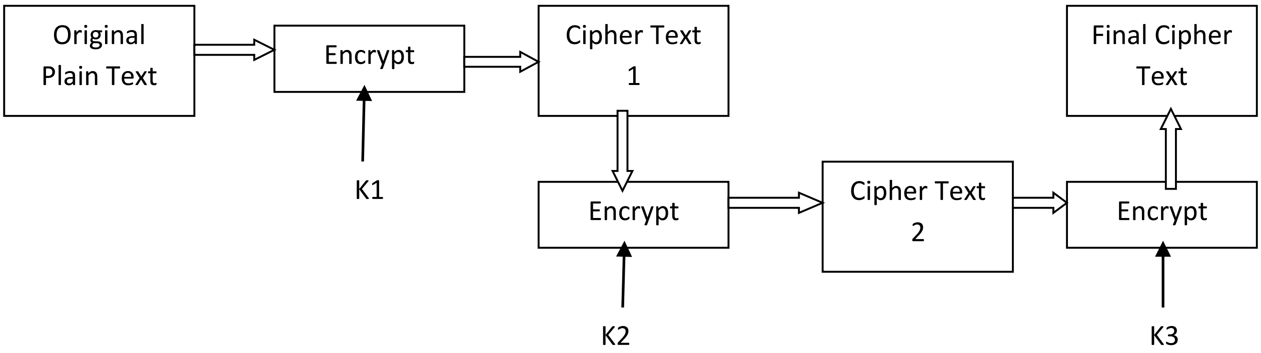
II. TRIPLE DES:

It has two variations:

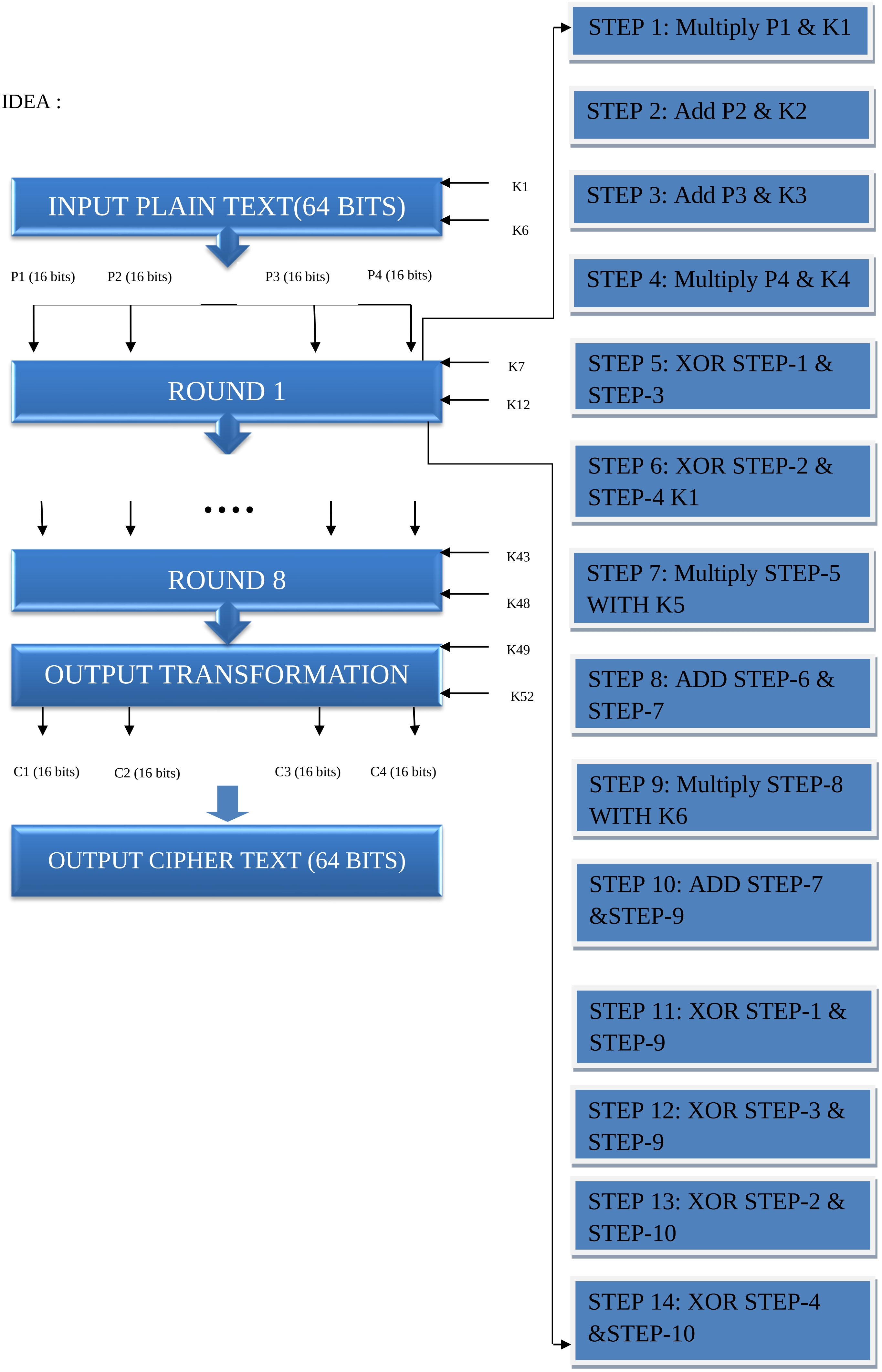
(i) Triple DES with 2 keys



(ii) Triple DES with 3 keys:



IDEA :



STEP 1: Multiply P1 & K1

STEP 2: Add P2 & K2

STEP 3: Add P3 & K3

STEP 4: Multiply P4 & K4

STEP 5: XOR STEP-1 & STEP-3

STEP 6: XOR STEP-2 & STEP-4 K1

STEP 7: Multiply STEP-5 WITH K5

STEP 8: ADD STEP-6 & STEP-7

STEP 9: Multiply STEP-8 WITH K6

STEP 10: ADD STEP-7 &STEP-9

STEP 11: XOR STEP-1 & STEP-9

STEP 12: XOR STEP-3 & STEP-9

STEP 13: XOR STEP-2 & STEP-10

STEP 14: XOR STEP-4 &STEP-10

IDEA WORKING MECHANISM:

STEP-1: BASIC PRINCIPLES:

(i) IDEA is a block cipher.

(ii) Like DES it also works on 64-bit plain text blocks.

(iii) The key is however longer & consists of 128 bits.

(iv) IDEA is reversible like DES i.e. same algorithm can be used for encryption & decryption.

(v) IDEA uses both diffusion & confusion for encryption.

(vi) The 64-bit input plain text block is divided into 4 portions of plain text each of size 16-bits i.e. P1 to P4.

(vii) P1 to P4 are the inputs to the 1st round of the algorithm. There are 8 such rounds.

(viii) In each round 6 sub-keys are generated from the original key, each sub-key of 16-bits.

STEP-2: ROUNDS:

(i) There are 8 rounds in IDEA.

(ii) Each round involves a series of operations on 4 data blocks using 6 keys.

(iii) Operations involve modulo 2^{16} Multiplication, modulo $2^{16}+1$ Addition & XOR.

STEP-3: SUBKEY GENERATION FOR A ROUND:

(i) Each of the 8 rounds makes use of 6 subkeys (8x6=48 subkeys are required for the rounds) and the final output transformation uses 4 subkeys (making a total of 48+4=52 subkeys overall).

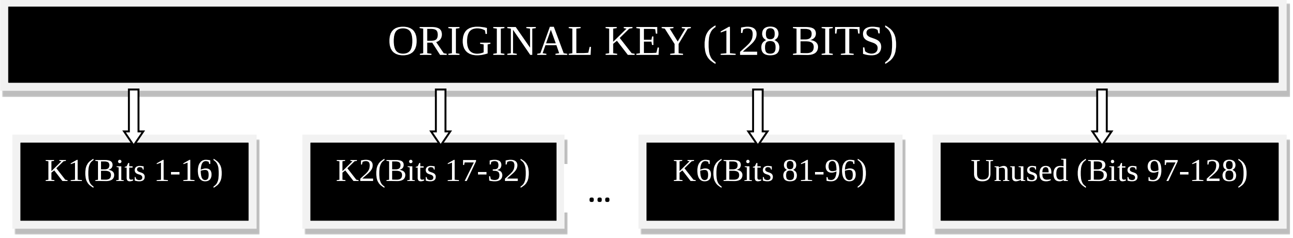
SUBKEY Generation:

(a) 1st Round:

→ Initial key consists of 128 bits from which subkeys K1 to K6 are generated for the 1st round.

→Since K1 to K6 consists of 16 bits each, out of the original 128 bits, the 1st 96 bits are used for the 1st round.

→At the end of 1st round, bits 97 to 128 of the original key are unused.



(b) 2nd Round:

→ For 2nd round, we can utilize 32 unused key bits at positions 97 to 128 which gives 2 subkeys each of 16 bits.

→The remaining 64 bits for the 2nd round are found by key shifting. The original key is shifted left circularly by 25 bits, i.e. the 26th bit of the original key moves to the 1st position & becomes the 1st bit after the shift, & the 25th bit of the original key moves to the last position & becomes the 128th bit after the shift.

STEP-4: OUTPUT TRANSFORMATION:

(i) It is a on-time operation.

(ii) It takes place at the end of the 8th round.

(iii) The input to the output transformation is the output of the 8th round i.e. a 64 bit value divided into 4 sub-blocks each of 16 bits and 4 subkeys are supplied.

STEP-5: SUBKEY GENERATION FOR OUTPUT TRANSFORMATION:

(i) At the end of 8th & the final round, the key is exhausted & shifted. Therefore, in this round, the 64 bits makeup subkeys K1 to K4, which are used as the 4 subkeys for this round.

DECRYPTION: The decryption process is similar to that of encryption process.

STRENGTH OF IDEA:

IDEA uses 128-bit key which is double than the key size of DES. Thus to break into IDEA, 2^{128} encryption operations would be required.

RC4 :

(i) RC4 was designed by Ron Rivest of RSA Security in 1987. The official name for this algorithm was “Rivest Cipher 4”. However because of its ease of reference, the acronym RC4 has stuck.

(ii) RC4 is a stream cipher. This means that the encryption happens byte-by-byte. However, this can be changed to bit-by-bit encryption.

RC4 WORKING MECHANISM:

(i) RC4 generates a pseudorandom stream of bits called keystream. This is combined with the plain text using XOR for encryption.

(ii) There is a variable length key consisting of 1 to 256 bytes. This key is used to initialize a 256-byte state vector with elements identified as S [0], S [1]...S [255].

(iii) To perform an encryption or decryption operation, one of these 256 bytes of S is selected and processed, output as k.

(iv) After this, the entries in S are permuted once again.

(v) There are overall 2 processes involved:

(a) Initialization of S

(b) Stream Generation

(a) Initialization Of S:

a.1 Choose a key (K) of length between 1 to 256 bytes.

a.2 Set the values in the state vector S equal to the values from 0 to 255 in an ascending order. In other words, we should have S[0]=0,S[1]=1,...,S[255]=255.

a.3 Create another temporary array T. If length of the key K (keylen) is 256 bytes, copy K into T as is. Otherwise, after copying K to T, whatever are the remaining positions in T are filled with the values of K again. At the end, T should be completely filled.

(b) Stream generation:

b.1 The initial key array K is discarded.

b.2 For again looping for i=0 to 255, we swap S[i] with another byte in S as per the mechanism decided by the implementation of S. Once we exhaust the 255 positions, we need to start at S[0]..

b.3 For encryption, k is XORed with the next byte of the plain text. For decryption, k is XORed with the next byte of the cipher text.

RC5:

(i) RC5 is a symmetric-key block encryption algorithm developed by Ron Rivest.

(ii) It is quite fast as it uses only the primitive computer operations (such as addition, XOR, shift etc.).

(iii) It allows for a variable number of rounds, and a variable bit-size key to add to the flexibility.

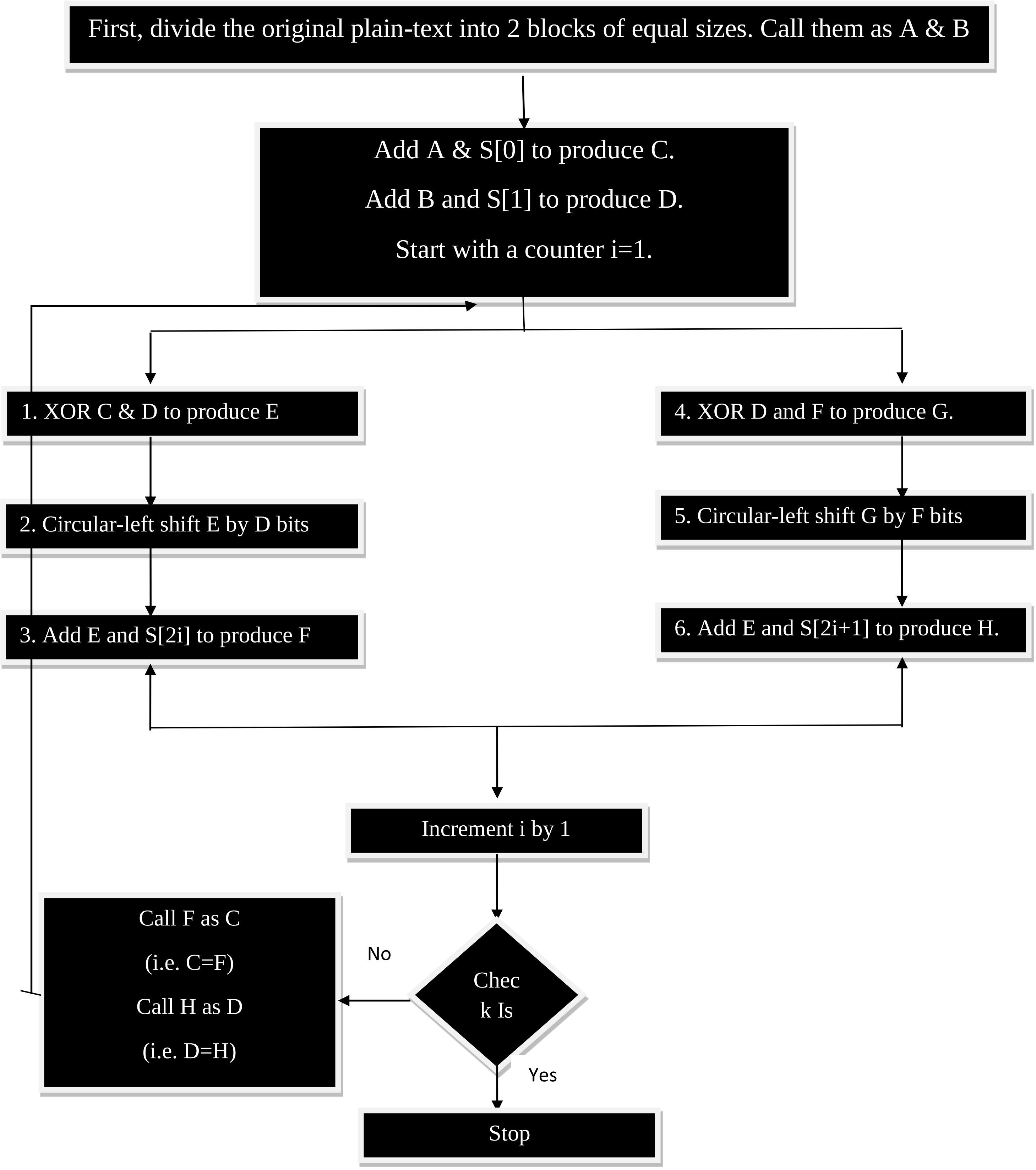
(iv) It requires less memory for execution, and is therefore, suitable not only for desktop computers but also for smart cards and other devices that have a small memory capacity.

RC5 WORKING MECHANISM:

(i) Here, the word size (i.e. input plain-text block size), the number of rounds and number of 8-bit bytes of the key, all can be of variable length.

Parameter	Allowed Values
Word size in bits(RC5 encrypts 2-word blocks at a time)	16,32,64
Number of rounds	0-255
Number of 8-bit bytes(octets) in the key	0-255

(ii) The output resulting from RC5 is the cipher text, which has the same size as the input plain text. Since RC5 allows for variable values in the 3 parameters, a particular instance of RC5 algorithm is denoted as RC5-w/r/b where w=word size in bits, r=number of rounds, b= number of 8-bit bytes in the key.


First, divide the original plain-text into 2 blocks of equal sizes. Call them as A & B
Add A & S[0] to produce C.
Add B and S[1] to produce D.
Start with a counter i=1.
1. XOR C & D to produce E
2. Circular-left shift E by D bits
3. Add E and S[2i] to produce F
4. XOR D and F to produce G.
5. Circular-left shift G by F bits
6. Add E and S[2i+1] to produce H.
Increment i by 1
Chec
k Is
No
Yes
Call F as C
(i.e. C=F)
Call H as D
(i.e. D=H)
Stop

BLOWFISH:

(i) Blowfish was developed by Bruce Schneier.

(ii) Blowfish was designed with the following objectives:

(a) Fast: Encryption rate on 32-bit microprocessor is 26 clock cycles per byte.

(b) Compact: Blowfish can execute in less than 5KB memory.

(c) Simple: Blowfish uses only primitive operations, such as addition, XOR and table look-up, making its design and implementation simple.

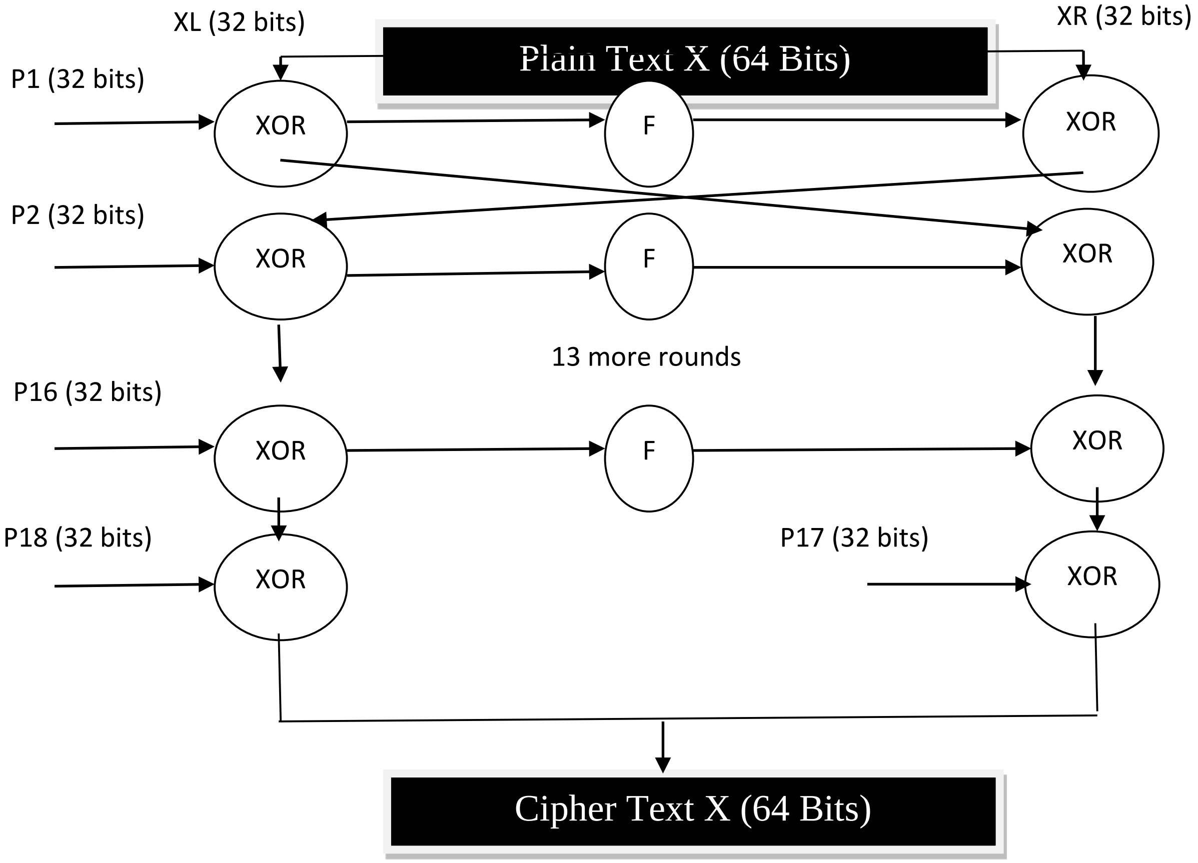
(d) Secure: Blowfish has a variable key length up to maximum of 448 bits long, making it both flexible and secure.

BLOFISH WORKING MECHANISM:

Blowfish encrypts 64-bit blocks with a variable length key. It contains 2 parts as follows:

(a) Subkey Generation: This process converts the key upto 448 bits long to subkeys totalling 4168 bits.

(b) Data Encryption: This process involves the iteration of a simple function 16 times. Each round contains a key-dependent permutation and key-and data-dependent substitution.



- AES:

(i) In 1990s the US government wanted to standardize a cryptographic algorithm, which was to be used universally by them. It was called Advanced Encryption Standard (AES).

Main features of AES:

(i) Symmetric & parallel structure: This gives implementers of the algorithm a lot of flexibility and stands up cryptanalysis attacks.

(ii) Adapted to modern processors: It works well with modern processors like Pentium, RISC, Parallel.

(iii) Suited to Smart Cards: The algorithm works well with smart cards.

AES WORKING MECHANISM:

i. One time initialization processes:

(a) Expand the 16-byte key to get the actual key block to be used.

(b) Do one time initialization of the 16-byte plain text block (called state).

(c) XOR the state with the key block.

ii. For each round, do the following:

(a) Apply S-box to each of the plain-text bytes.

(b) Rotate row k of the plain-text block (i.e. state) by k bytes.

(c) Perform a mix columns operation.

(d) XOR the state with the key block.

- ASYMMETRIC KEY CRYPTOGRAPHIC ALGORITHMS

INTRODUCTION:

Asymmetric key cryptography is a class of cryptographic algorithms which requires two separate keys, one of which is secret (or private) and one of which is public, for encryption and decryption. The term "asymmetric" comes from the use of different keys to perform these opposite operations, each the inverse of the other, as contrasted with conventional symmetric cryptography which relies on the same key to perform both.

The conceptual difference between symmetric and asymmetric cryptography systems are based on how these systems keep a secret.

Asymmetric cryptography is based on personal secrecy i.e. the secret is unshared. Each person creates and keeps his own secret. For n number of people an asymmetric cryptography system requires n personal keys against $\frac{n(n-1)}{2}$ shared keys for a symmetric cryptography system. There are other aspects of security that require asymmetric key cryptography, such as authentication and digital signature.

In symmetric key cryptography, symbols are permuted or substituted, whereas in asymmetric key cryptography numbers are manipulated by applying mathematical functions to them.

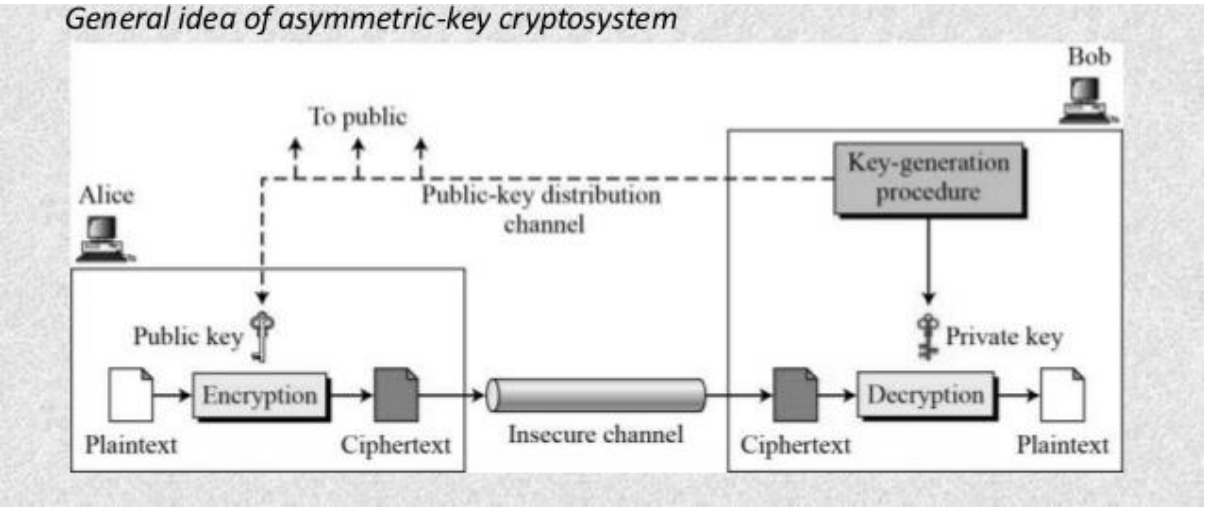
HISTORY OF ASYMMETRICKEY CYPTOGRAPHY

- Whitfield Diffie and Martin Hellman come up with the concept of asymmetric key cryptography in the mid-1970s.
- Rivest, Shamir and Adleman develop first major asymmetric key cryptosystem based on Diffie-Hellman framework in 1977 and published in 1978.
- In 1973 the British Communication Electronics Security Group (CSEG) came up with another asymmetric key cryptosystem, but it was not made publicly available until 1998.

KEYS

Asymmetric key cryptography uses two separate keys; one private and one public. Together they are called a key pair. Although different, the two parts of this key pair are mathematically linked. The public key is used to encrypt plain text, whereas the private key is used to decrypt cipher text.

OVERVIEW OF ASYMMETRIC KEY CRYPTOGRAPHY



The figure above shows the general idea of asymmetric key cryptography system. The key generation procedure creates a private key and a public key. The public key is distributed over a key distribution channel. Though secrecy of such a channel is not necessary, it must provide authentication and integrity. The plain text is encrypted using the public key of the recipient and sent over an unsecure channel. The recipient is able to decrypt the cipher text using his own private key.

It can be noted that the recipient can receive and decrypt cipher text from multiple senders using the same pair of keys; however the sender needs n public keys to send an encrypted text to n recipients.

- THE RSA ALGORITHM

The most common asymmetric key cryptosystem is the RSA cryptography algorithm named after its inventors Rivest, Shamir and Adleman.

The RSA algorithm is based on the mathematical fact that it is easy to find and multiply large prime numbers together but it is extremely difficult to factor their product. The prime numbers used in RSA algorithm are very large (made up of 100 or more digits).

Steps of RSA Algorithm

1. Choose two large prime numbers P and Q.
2. Calculate N = P x Q
3. Select the public key E such that it is not a factor of (P-1) and (Q-1).
4. Select the private key such that (D x E) mod (P-1) x (Q-1) = 1
5. The cipher text is generated as CT = PT^E mod N
6. The plain text is generated as PT = CT^D mod N

Security of RSA

The main possible attacks on RSA are,

- Plain Text Attacks
    - Short message attack
    - Cycling attack
    - Unconcealed message attack
- Chosen Cipher Text Attack
- Factorization Attack
- Attacks on Encryption key
- Attacks on Decryption key
    - Revealed decryption exponent attack
    - Low decryption exponent attack

Symmetric Key Vs Asymmetric Key Cryptography

Symmetric Key Cryptography	Asymmetric Key Cryptography
• Based on sharing secrecy • Same key is used both for encryption and decryption • Faster execution • Key agreement is a problem • Number of keys required among multiple users is large • Size of cipher text is same as the plain text • Used for encryption and decryption only.	• Based on personal secrecy • Different keys are used for encryption and decryption • Slower execution • No key agreement problem • Number of keys required is same as the participants • Size of cipher text is larger than the plain text • Used for encryption, decryption as well as digital signature

ASYMMETRIC & SYMMETRIC KEY CRYPTOGRAPHY TOGETHER

In practice, symmetric key cryptography and asymmetric key cryptography are combined together to have a very efficient security solution.

The sender encrypts the plain text with a symmetric key cryptography algorithm using a key called one time symmetric key. This one time symmetric key is again encrypted using the public key of the recipient. This process is called key wrapping of the symmetric key.

Now the sender puts the cipher text and the encrypted key in a digital envelope and sends it to the recipient. The recipient can obtain the key used for encrypting the plain text by decrypting the received key by his own private key.

MODULE IV: DIGITAL SIGNATURES

The concept of digital signature is based upon message authentication and integrity. A scheme, in which the sender encrypts the message with his private key, forms the basis of digital signature.

The sender encrypts the plain text using his private key and sends it to the recipient. The recipient can decrypt the cipher text using the sender's public key, which verifies that the received message was indeed sent from the rightful person and prevents non repudiation. Any middle person who can intercept the message can decrypt the message as the public key of the sender is known to everyone; however the middle person can't modify the contents of the message as it would require the private key of the sender to encrypt it again.

- MESSAGE DIGEST

While all the encryption algorithms ensure that a message can't be accessed by an unauthorised person, they don't ensure the integrity of the message itself i.e. if the message contents have been tampered during transit. To ensure integrity, message digest of a message is calculated and sent along with the original message.

A message digest is the summary or fingerprint of the message. A message digest must have the following properties.

- Given a message, it should be easy to find its message digest but given a message digest, it should be very difficult to find the original message.
- A message should always produce the same message digest.
- Message digest for two different messages must always be different. If two different messages have the same message digest, then a collision is said to have occurred.

A receiver can calculate the message digest of the received message and compare it to the received message digest to know if the message has been altered. Message digest is also called hash of a message. Some of the popular message digest algorithms are MD5, SHA, and HMAC.

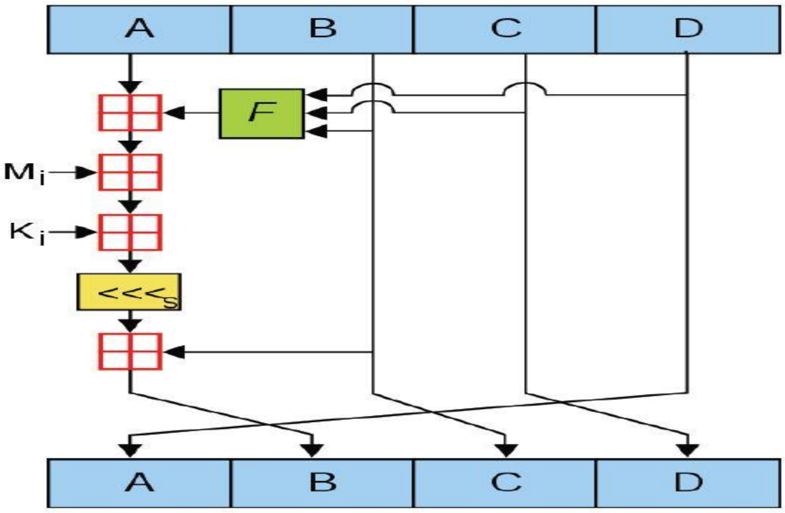
- MD5

MD5 algorithm was developed by Ronald Rivest in 1991. MD5 is a modified version of MD4.

MD5 algorithm produces a message digest of length 128 bits. It consists of 4 rounds with 16 operations in each round.

The original message is padded such that its length is 64 bits less than any multiple of 512. The padding is done by adding a single 1 followed by required number of 0s. A 64 bits long string representing the length of the original message is then appended to the padded message making the total size of the message a multiple of 512.

The message digest calculation process can be described as follows.



Here A, B, C and D are four chaining variables which are initialised before the start of the algorithm. Each variable contains a 32 bits hexadecimal number which are updated in each round. The chaining variables are initialised as follows.

- A: $(01\ 23\ 45\ 67)_H$
- B: $(89\ AB\ CD\ EF)_H$
- C: $(FE\ DC\ BA\ 98)_H$
- D: $(76\ 54\ 32\ 10)_H$

F is a non-linear function defined by the chaining variables. F is different for each round. The non-linear function F for different rounds is defined as follows.

- $F(B,C,D) = (B \wedge C) \vee (\backsim B \wedge D)$
- $F(B,C,D) = (B \wedge D) \vee (C \wedge \backsim D)$
- $F(B,C,D) = (B \oplus C \oplus D)$
- $F(B,C,D) = \ C \oplus (B \vee \backsim D)$

Where $\wedge, \vee, \oplus, \backsim$ denote logical AND, OR, XOR and NOT operations respectively.

M_i is the 32 bits block of the message input and K_i is 32 bits constant different for each round. The diagram above depicts only one round. There are four such rounds. At the end of all operations each chaining variable contains 32 bits hexadecimal number which are added together to form 128 bits message digest.

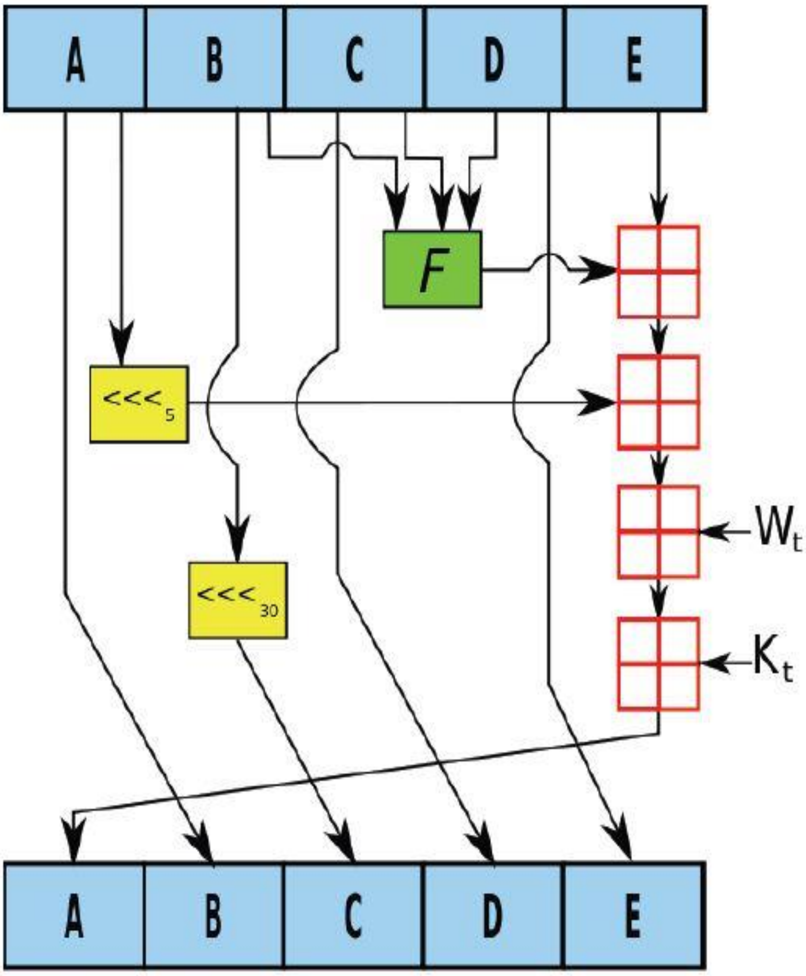
- SECURE HASH ALGORITHM (SHA)

SHA is a message digest algorithm designed by the United States National Security Agency as a federal information processing standard in 1993. It uses similar principles as that of MD4 and MD5.

SHA-1 produces a message digest of length 160 bits. It operates on 5 chaining variables and has 80 rounds of operations. The padding of the message is same as MD5. The fifth chaining variable E is initialised as,

- E: $(\ 0F\ 1E\ 2D\ 3C\ )_H$

The process can be described as follows.



F is a non-linear function operating on three chaining variables as input and changes with the rounds. The chaining variables are constantly updated and shifted in each round. W_t is the expanded message input block and K_t is a constant. Similar to the MD5 algorithm, the chaining variables contain 32 bits hexadecimal number at the end of all rounds which are appended together to produce the 160 bits message digest or hash.

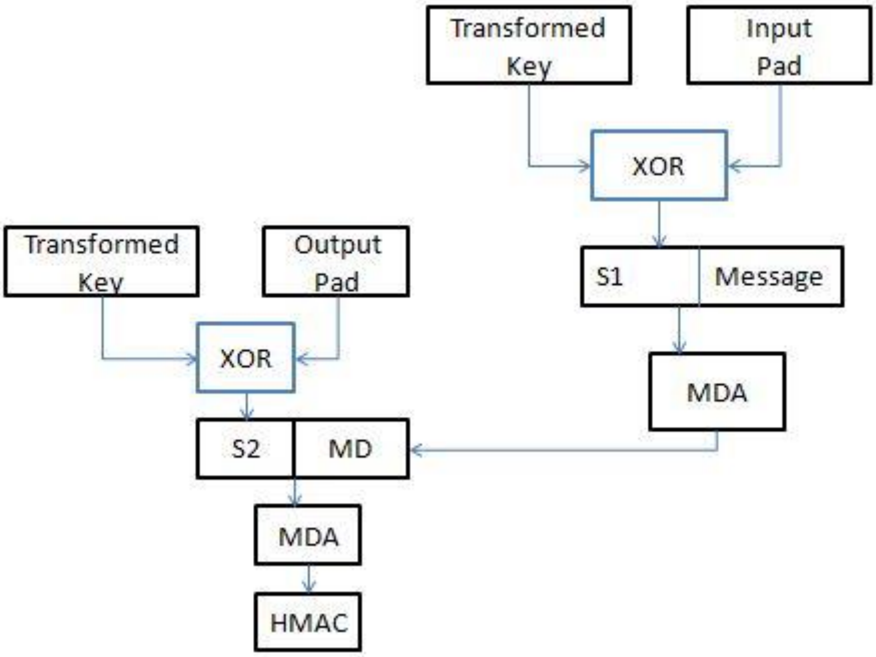
- MESSAGE AUTHENTICATION CODE (MAC)

The concept of MAC is quite similar to that of a message digest. The difference between a message digest and MAC is that, message digest doesn't involve any cryptographic process whereas MAC requires the sender and the receiver to share a symmetric key which is used to generate the MAC and hence involves a cryptographic process.

It may be used to simultaneously verify both the data integrity and the authentication of a message.

HASH BASED MAC (HMAC)

HMAC is a mandatory security implementation for internet protocol (IP) security and secure socket layer (SSL) protocol. The fundamental idea behind HMAC is to reuse existing message digest algorithms such as MD5 and SHA-1. Additionally it uses the shared symmetric key to encrypt the message digest to produce the output MAC.



The diagram above outlines the process of HMAC. The message is divided into blocks of b bits. The length of the shared key K is made equal to the number of bits per message block (i.e. b). If b is larger than K then the key is padded and if b is smaller than K then key is truncated. This is called the transformed key.

The input pad and output pad are two binary strings of size equal to that of the transformed key. MDA denotes message digest algorithm and it can be either MD5 or SHA-1.

- KNAPSACK ALGORITHM

→ Ralph Merkle and Martin Hellman developed the 1st algorithm for public-key encryption i.e. the Knapsack algorithm.

→ This is a simple problem; given a pile of objects each with different weights, is it possible to put some of them in a bag (i.e. knapsack) in such a way that the knapsack has a certain weight?

→ If M_1, M_2,.....,M_n are the given values and S is the sum, find out b_i such that:

$S=b_1M_1+b_2M_2+.....+b_nM_n$

Each b_i can be 0 or 1.

→ A block of plain text equal in length to the number of items in the pile would select the items in the knapsack. The cipher text is the resulting sum.

Plain Text	0 1 1 0 1 1	1 1 1 0 0 0	0 1 0 1 1 0
Knapsack	1 7 8 12 14 20	1 7 8 12 14 20	1 7 8 12 14 20
Cipher Text	7+8+14+20=49	1+7+8=16	7+12+14=33

ELGAMAL DIGITAL SIGNATURE:

→The ElGamal digital-signature scheme uses the same keys, but a different algorithm.

→ The algorithm creates two digital signatures. In the verification step, these two signatures are tallied.

- PUBLIC KEY INFRASTRUCTURE

INTRODUCTION:

Public Key Infrastructure (PKI) technology is the central focus in Internet security.

Digital Certificates are termed *passports* on the Web.

-- Certification Authorities (CA)

-- Registration Authorities (RA)

-- Relation between one CA with another

-- Root CA

-- Self-Signed Certificates

-- Cross Certification

-- Validating digital certificates; protocols: CRL, OCSP & SCVP

-- Maintaining & achieving user keys.

-- Roaming certificates

-- PKIX & PKCS standards for digital certificates

-- XML Security

DIGITAL CERTIFICATES:

I. To tackle the problem of key exchange or key agreement digital certificates were introduced.
II. A digital certificate is a small computer file.
III. A digital certificate establishes the relation between a user and his/her public key.
IV. A digital certificate contains the user name & the user's public key. This will prove that a particular public key belongs to a particular user.
V. Similarity between Passport & corresponding Digital Certificate entry :

Passport Entry	Corresponding Digital Certificate Entry
Full Name	Subject Name
Passport Number	Serial Number
Valid From	Same
Valid to	Same
Issued By	Issuer Name
Photograph and Signature	Public Key

CERTIFICATION AUTHORITY (CA):

-- A Certification Authority (CA) is a trusted agency that can issue digital certificates.

-- The authority of acting as a CA has to be with someone who everybody trusts. Consequently, the governments in various countries decide who can & who cannot be a CA.

-- Usually, a CA is a reputed organization such as a:

(a) Post Office

(b) Financial Institution

(c) Software Company

-- Most popular CA in the world are:

(a) VeriSign

(b) Entrust Safescrypt Limited

(c) Subsidiary of Satyam Infoway Limited – 1st Indian CA in Feb. 2002.

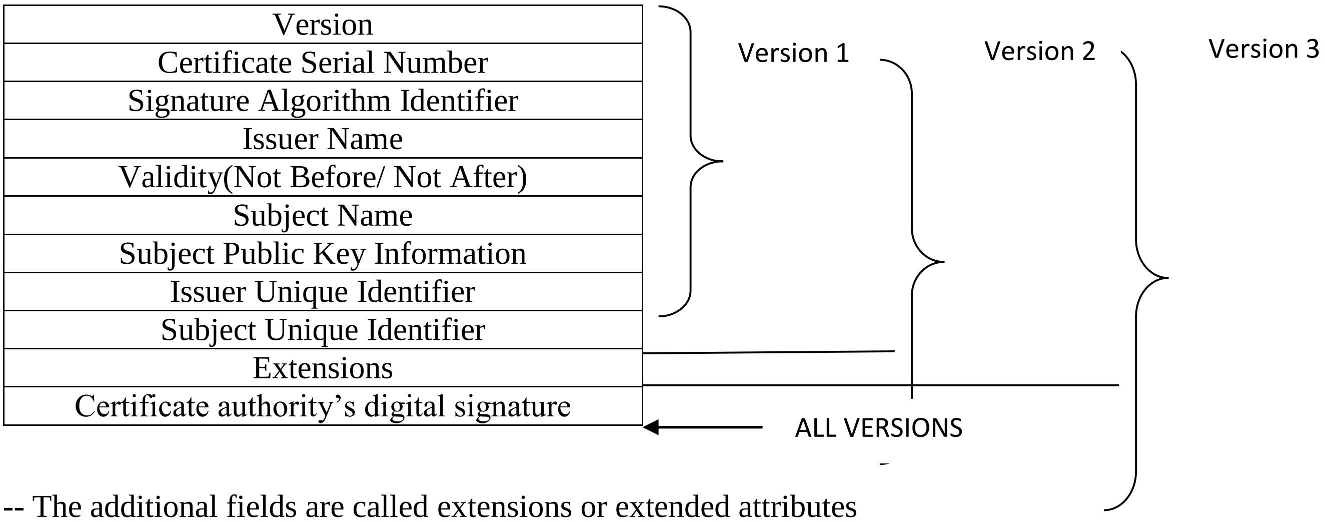
-- Thus, CA has the authority to issue digital certificates to individuals and organizations, who want to use those certificates in asymmetric-key cryptographic applications.

TECHNICAL DETAILS OF DIGITAL CERTIFICATE:

-- Standard X.509 defines the structure of digital certificate which was introduced by ITU (International Telecommunication Union) in 1988; at that time it was part of X.500.

-- X.509 was revised twice, hence current version is X.509V3

-- The IETF (Internet Engineering Task Force) published the RFC2459 for the X.509 standard in 1999. The structure of X.509V3 digital certificate is as follows:



Version
Certificate Serial Number
Signature Algorithm Identifier
Issuer Name
Validity(Not Before/ Not After)
Subject Name
Subject Public Key Information
Issuer Unique Identifier
Subject Unique Identifier
Extensions
Certificate authority's digital signature

-- The additional fields are called extensions or extended attributes

Version 1 of X.509 digital certificate:

FIELD	DESCRIPTION
Version	Identifies a particular version of X.509 protocol, which is used for this digital certificate. Currently, this field can contain 1, 2 or 3.
Certificate Serial Number	Contains unique integer number generated by CA.
Signature Algorithm Identifier	Identifies the algorithm used by the CA to sign this certificate.
Issuer Name	Identifies the Distinguished Name (DN) of the CA that created & signed this certificate.
Validity (Not Before/ Not After)	Contains two date-time values (Not Before & Not After), which specify the time frame within which the certificate should be considered valid. These values generally specify the date & time up to seconds or milliseconds.
Subject Name	Identifies the DN of the end entity (i.e. the user or organization) to whom this certificate refers. This field must contain an entry unless an alternative name is defined in Version 3 extensions.
Subject Public Key Information	Contains the subject's public key & algorithms related to that key. This field can never be blank.

Version 2 of the X.509 digital certificate:

FIELD	DESCRIPTION
Issuer Unique Identifier	Helps identify a CA uniquely if two or more CAs have used the same Issuer Name over time.
Subject Unique Identifier	Helps identify a subject uniquely if two or more subjects have used the same Subject Name over time.

Version 2 of the X.509 digital certificate:

FIELD	DESCRIPTION
Authority Key Identifier	A CA may have multiple private-public key pairs. This field defines which of these key pairs is used to sign (and hence, which corresponding key should be used to verify) this certificate
Subject Key Identifier	A subject may have multiple private-public key pairs. This field defines which of those key pairs is used to sign (and the corresponding key used to verify).
Key Usage	Defines the scope of operations of the public key of this particular certificate.
Extended Key Usage	Can be used in addition to or in the place of Key Usage Field. Specifies which protocols this certificate can interoperate with.

Private key Usage Period	Allows defining different usage period limits for the private & public keys corresponding to this certificate.
Certificate policies	Defines the policies & optional qualifier information that the CA associates with a given certificate.
Policy Mappings	Used only the subject of a given certificate is also a CA.
Subject Alternative Name	Optionally defines one or more alternative names for the subject of a given certificate.
Issuer Alternative Name	Optionally defines one or more alternative names for the issuer of a given certificate.
Subject Directory Attributes	Can be used to provide additional information about the subject.
Basic Constraints	Indicates whether the subject in this certificate may act as a CA.
Name Constraints	Specifies the name space.
Policy Constraints	Used only for CA certificates.

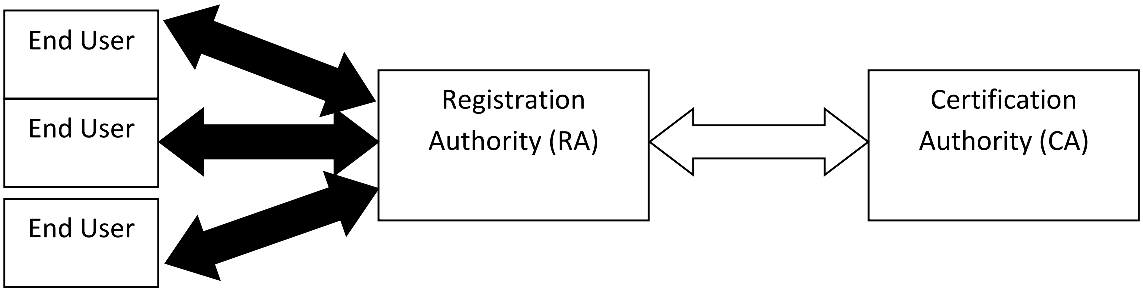
DIGITAL CERTIFICATE CREATION:

I. Parties Involved :

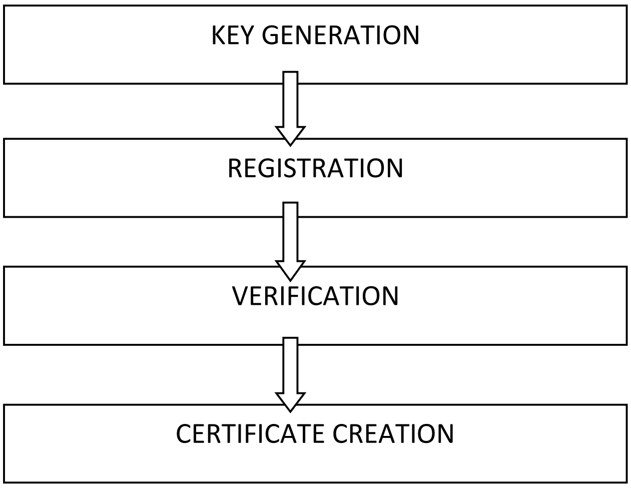
(a) Issuer CA (Certification Authority)

(b) Subject (End User)

(c) Third Party RA (Registration Authority)



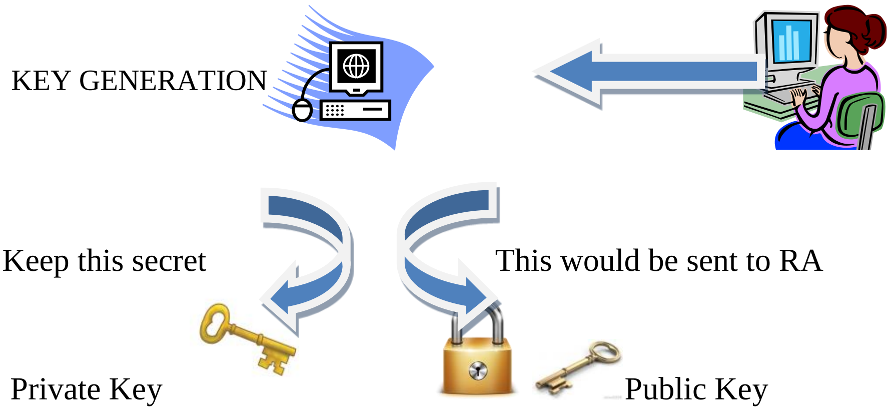
II. Certificate Creation Steps :



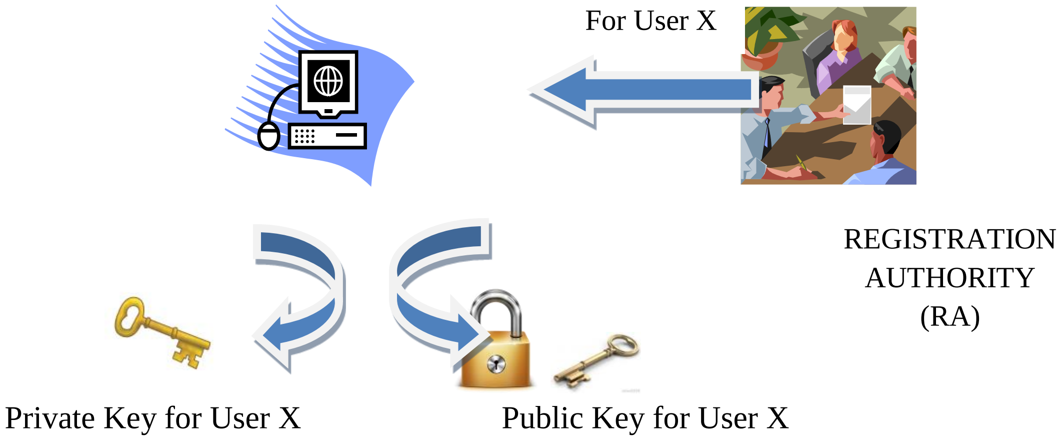
(a) Key generation: The initiation begins with the subject (i.e. user/ organization) who wants to obtain a certificate. There are 2 different approaches:

(i) Subject-end: The subject can generate a private key & public key pair using some software. This software is usually part of a Web browser or server. The subject keeps the private key, thus generated, safe. The subject then sends the public key along with other information & evidences about herself to the RA.

(ii) RA-end: RA generates a key pair on the subject's (user's) behalf.



Subject generating its own key pair



RA generating a key pair on behalf of the subject

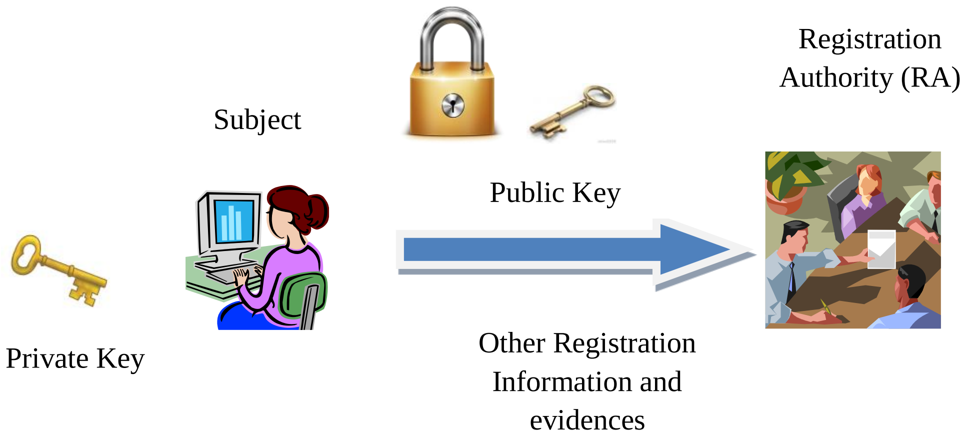
(b) Registration: This step is only required when user generates the key pair in the 1st step. If RA generates the key pair on user's behalf, this step will also be a part of the 1st step itself.

-- Assuming that the user has generated the key pair, the user now sends the public key & the associated registration information (e.g. subject name, as it is desired to appear on the digital certificate) & all the evidence about herself to the RA.

-- For this, the S/W provides a wizard in which the user enters the data & when all data is correct, submits it.

-- The data then travels across the network/Internet to the RA.

-- The format for the requests has been standardized & is called Certificate Signing Request (CSR). This is one of the Public Key Cryptography Standards (PKCS).


Subject
Registration Authority (RA)
Public Key
Private Key
Other Registration Information and evidences

III. VERIFICATION: After the registration process is complete, the RA has to verify the user's credentials. This verification is in two respects, as follows :
(a) Firstly, the user needs to verify the user's credentials such as the evidences provided are correct, & that they are acceptable. If the user were actually an organization, then the RA would perhaps like the business records, historical documents and credibility proofs. If it is an individual user then simpler checks are in call, such as verifying the postal address, e-mail id, phone no, passport or driving-license details can be sufficient.
(b) The 2nd check is to ensure that the user who is requesting for the certificate does indeed process the private key corresponding to the public key that is sent as a part of the certificate request to the RA. This is called checking the Proof of Possession (POP) of the private key.
Approaches by RA to perform POP:
-- The RA can demand that the user must digitally sign his/her Certificate Signing Request (CSR) using his/her private key. If the RA can verify the signature correctly using the public key of the user, the RA can believe that the user indeed possesses the private key.
-- Alternatively, at this stage, the RA can create a random number challenge, encrypt it with the user's public key & send the encrypted challenge to the user. If the user can successfully decrypt the challenge using his/her private key, the RA can assume that the user possesses the right private key.
-- Thirdly, the RA can actually generate a dummy certificate for the user, encrypt it with user's public key & send it to the user. The user can decrypt it only if he/she can decrypt the encrypted certificate, and obtain the plain-text certificate.

IV. CERTIFICATE CREATION: Assuming all the steps so far have been successful, the RA passes on all the details of the user to the CA.
-- The CA does its own verification & creates a digital certificate for the user.
-- The CA sends the certificate to the user & retains a copy with itself.
-- CA's copy is maintained in a certificate directory.
-- Clients can access information from the central repository using a directory access protocol like Lightweight Directory Access Protocol (LDAP). LDAP allows users & applications to access X.500 directories, depending on their privileges.

TRUST ON DIGITAL CERTIFICATES:

(A) INTRODUCTION:

(i) It contains information (public key) about an user.

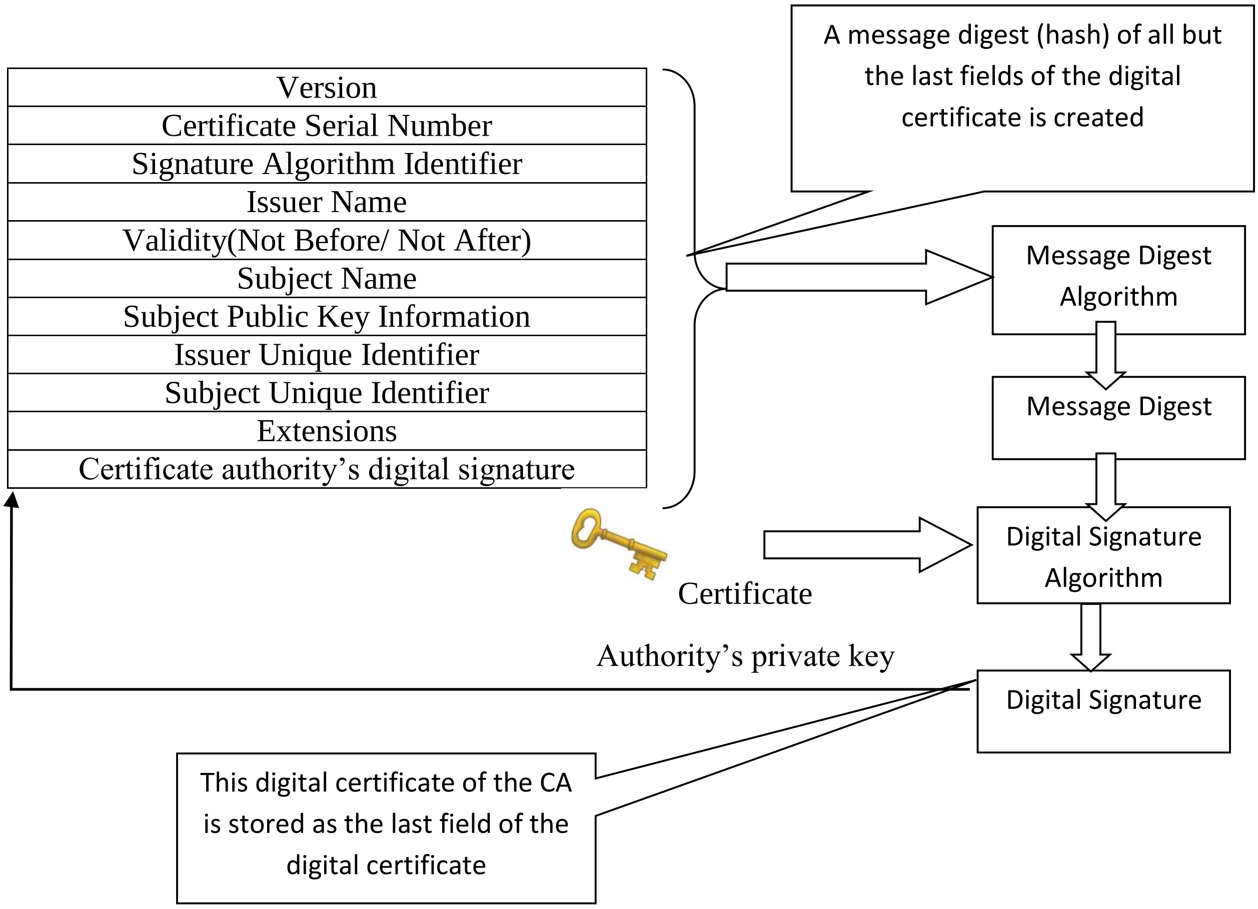
(ii) It is in a predetermined format & is stamped & signed by the authority.

(B) HOW CA SIGNS A CERTIFICATE?

-- 1st we would verify the CA's signature, for which we would use the CA's public key & check if it can de-sign the certificate correctly. If the designing works correctly, we can consider the certificate to be valid one.

(i) As shown in the figure below, before issuing a digital certificate to a user, the CA 1st calculates a message digest over all the fields of the certificate (using a standard message digest algorithm such as MD5 or SHA-1) & then encrypts the message digest with its private key (using an algorithm such as RSA0 to form the CA's digital signature.

(ii) The CA then inserts its digital signature thus calculated, as the last field in the digital certificate of the user.


Version
Certificate Serial Number
Signature Algorithm Identifier
Issuer Name
Validity(Not Before/ Not After)
Subject Name
Subject Public Key Information
Issuer Unique Identifier
Subject Unique Identifier
Extensions
Certificate authority's digital signature
A message digest (hash) of all but the last fields of the digital certificate is created
Message Digest Algorithm
Message Digest
Digital Signature Algorithm
Certificate
Authority's private key
Digital Signature
This digital certificate of the CA is stored as the last field of the digital certificate

(C) HOW A DIGITAL CERTIFICATE CAN BE VERIFIED:

The verification consists of the following steps:

(i) User passes all fields except the last one of the received digital certificate to a message-digest algorithm. This algorithm should be the same as the one used by the CA while signing the certificate. The CA mentions the algorithm used for signing along with the signature in the certificate, so the user here knows which algorithm is to be used.

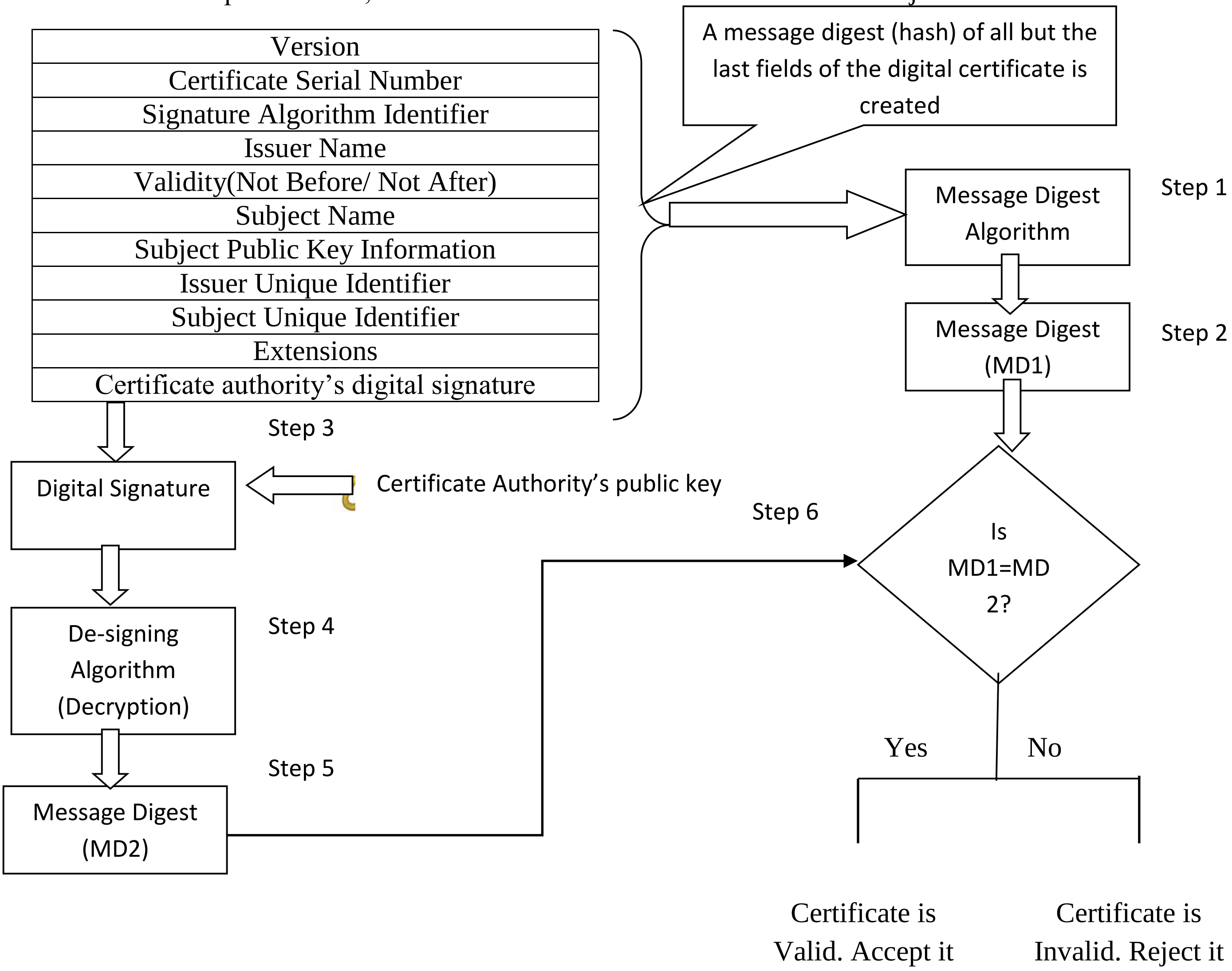
(ii) The MD algorithm calculates a MD (hash) of all fields of the certificate except for the last one (say MD1).

(iii) The user now extracts the digital signature of the CA from the certificate.

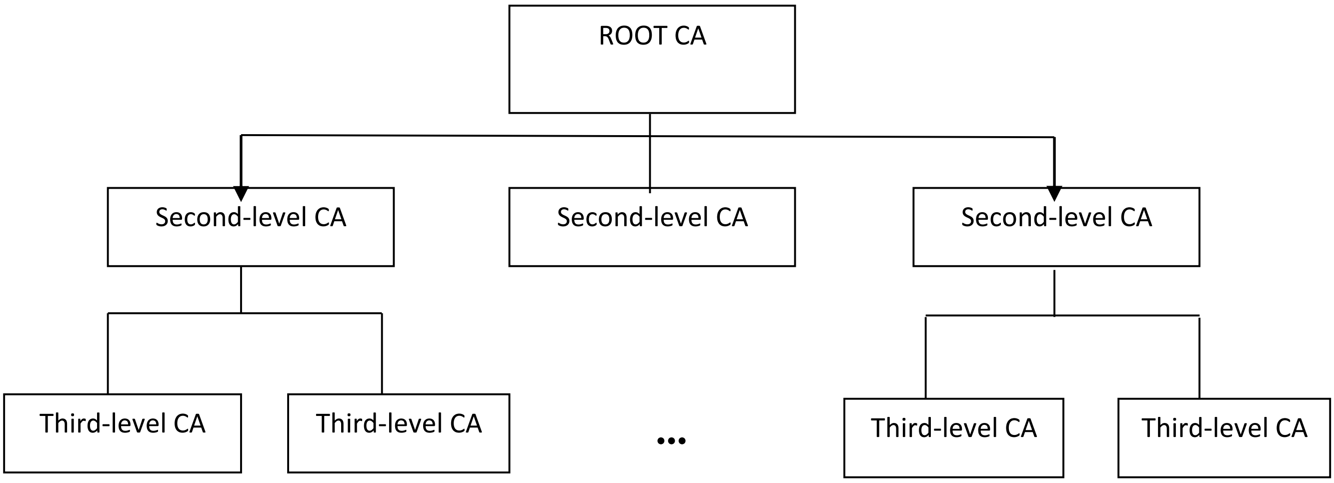
(iv) The user de-signs the CA's signature.

(v) This produces another MD say MD2.

(vi) Now the user compares MD1 with that of MD2. If the two match, the user is convinced that the digital certificate was indeed signed by the CA with its private key. If the comparison fails, the user doesn't trust the certificate & hence rejects it.

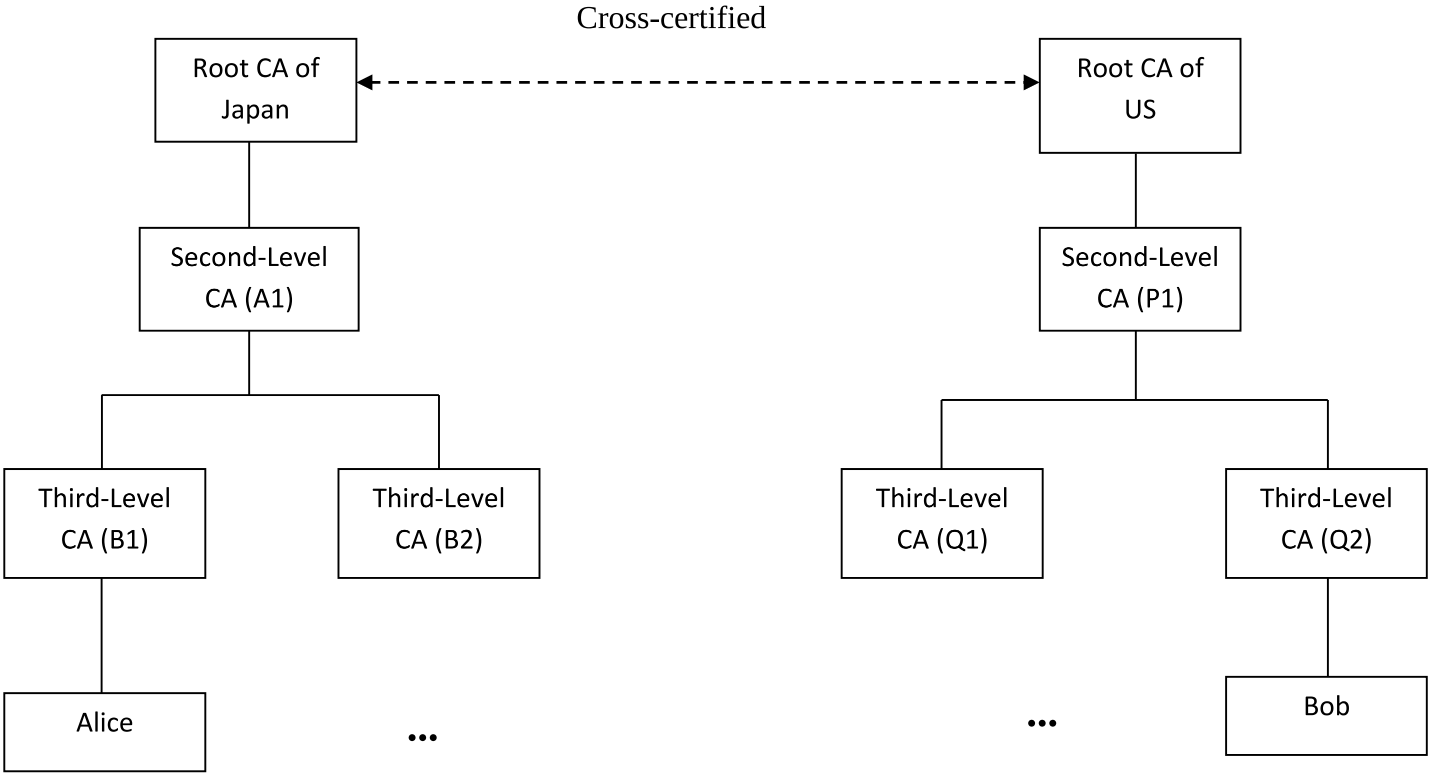

Version
Certificate Serial Number
Signature Algorithm Identifier
Issuer Name
Validity(Not Before/ Not After)
Subject Name
Subject Public Key Information
Issuer Unique Identifier
Subject Unique Identifier
Extensions
Certificate authority's digital signature
A message digest (hash) of all but the last fields of the digital certificate is created
Message Digest Algorithm
Step 1
Message Digest (MD1)
Step 2
Step 3
Digital Signature
Certificate Authority's public key
Step 6
Is MD1=MD 2?
De-signing Algorithm (Decryption)
Step 4
Step 5
Message Digest (MD2)
Yes
No
Certificate is Valid. Accept it
Certificate is Invalid. Reject it

CERTIFICATE HIERARCHIES & SELF-SIGNED DIGITAL CERTIFICATES:



Self-signed Certificate: The certificate of the root CA is a self-signed certificate i.e. the root CA signs its own certificate.

Cross-Certification:



- CERTIFICATE REVOCATION:

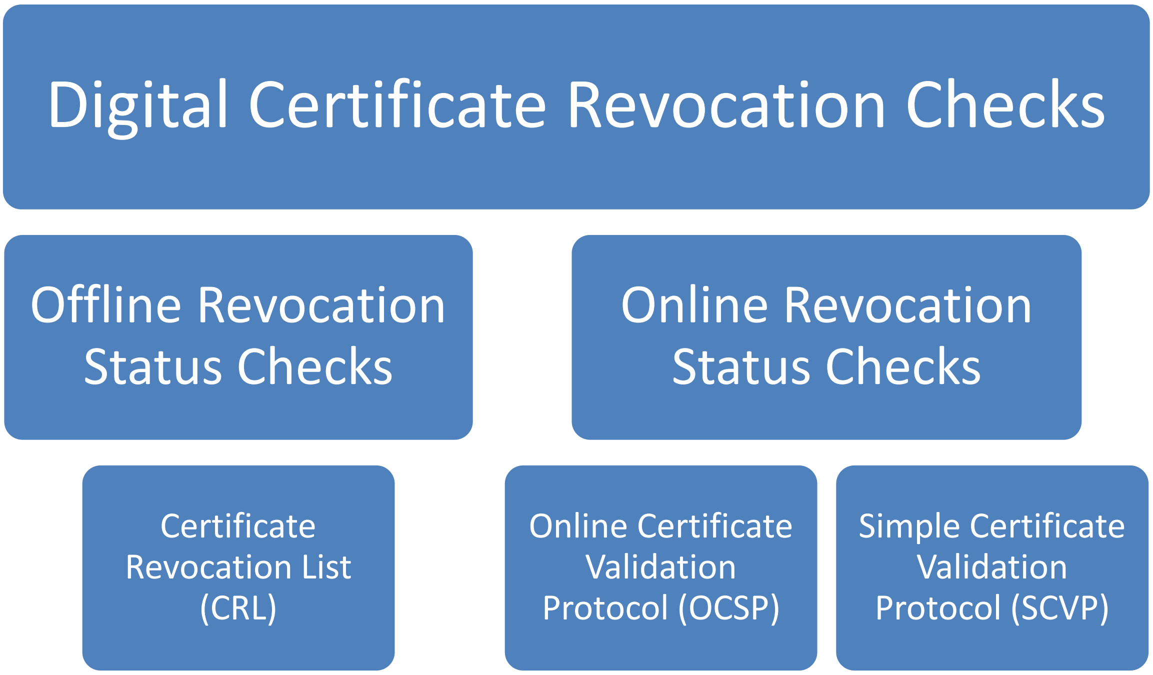
Need for Revocation:

(i) The holder of the digital certificate reports that the private key corresponding to the public key specified in the digital certificate is compromised.

(ii) The CA realizes that it had made some mistake while issuing a certificate.

(iii) The certificate holder leaves a job, and the certificate was issued specifically while the person was employed in that job.

CERTIFICATE REVOCATION STATUS MECHANISMS:


Digital Certificate Revocation Checks
Offline Revocation Status Checks
Online Revocation Status Checks
Certificate Revocation List (CRL)
Online Certificate Validation Protocol (OCSP)
Simple Certificate Validation Protocol (SCVP)

(i) Offline Certificate Revocation Status Checks:

(a) CRL: The Certificate Revocation List (CRL) is the primary means of checking the status of a digital certificate offline.

-- CRL is a list of certificates published regularly by each CA identifying all the certificates that have been revoked through the life of the CA.

-- This list doesn’t involve certificates whose validity period is over.

-- Each CA issues its own CRL. The respective CA signs each CRL, so the CRL can be easily verified.

-- A CRL is simply a sequential file that grows over time to include all the certificates that have not been expired, but have been revoked.

(a) Certificate Expiry Check: Compare the current date with the validity period of the certificate to ensure that the certificate has not expired.\

(b) Signature Check: Check that user’s certificate can be verified in terms of the signature of his CA.

(c) Certificate Revocation Check: Consult the latest CRL issued by user’s CA to ensure that user’s certificate is not listed there as a revoked certificate.

Base CRL: A one-time up-to-date CRL sent to the user from the CA who want to use the CRL services.

Delta CRL: The changes made to the base CRL at the time of next update.

Delta CRL Indicator: The indicator that informs the user that this CRL file is not a complete, comprehensive CRL file, but instead it is a delta CRL.

Delta Information: Indicator contained in base CRL which informs a user that delta CRLs are also available corresponding to this base CRL.

Disadvantages:

(a) Latency – certificate revocation check processing time

(b) Large size

(c) Likelihood of being stale

(ii) Online Certificate Revocation Status Checks:

(a) OCSP (Online Certificate Status Protocol):

-- It can be used to check if a given digital certificate is valid at a particular moment.

-- It allows the certificate validators to check for status of certificates in real time, thus providing for a quicker, simpler & more efficient mechanism for digital certificate validations.

-- No downloading required.

→ The CA provides a server called OCSP responder. This server contains the latest certificate revocation information. The requestor (client) has to send a query (called OCSP request) about a particular certificate to check if it is revoked or not

→ The OCSP responder consults the server's X.500 directory (in which the CA continuously feeds the certificate revocation information) to see if the particular certificate is valid or not.

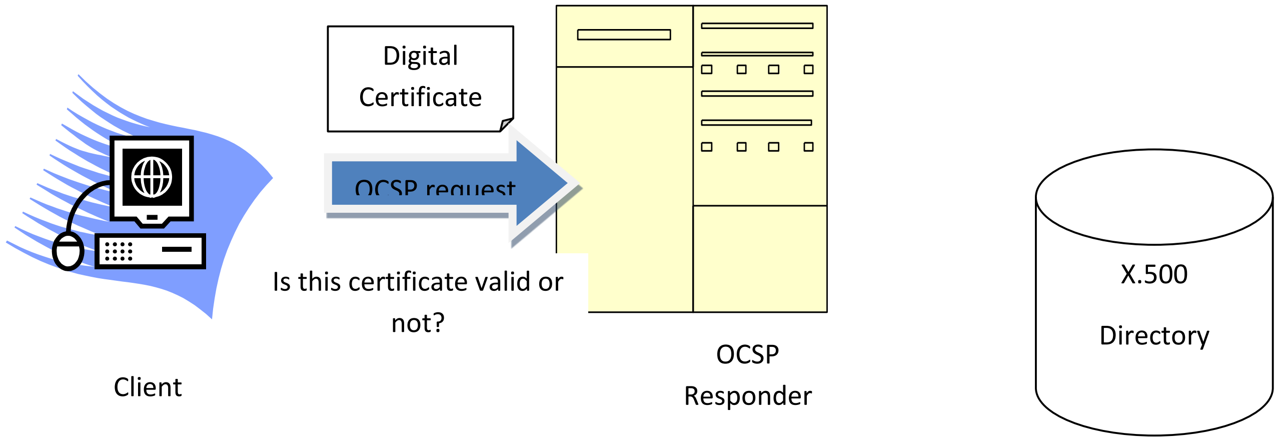
→ Based on the result of the status check from the X.500 directory lookup, the OCSP responder sends back a digitally signed OCSP response for each of the certificates in the original request to the client. This response can take one of the 3 forms:

(a) Good

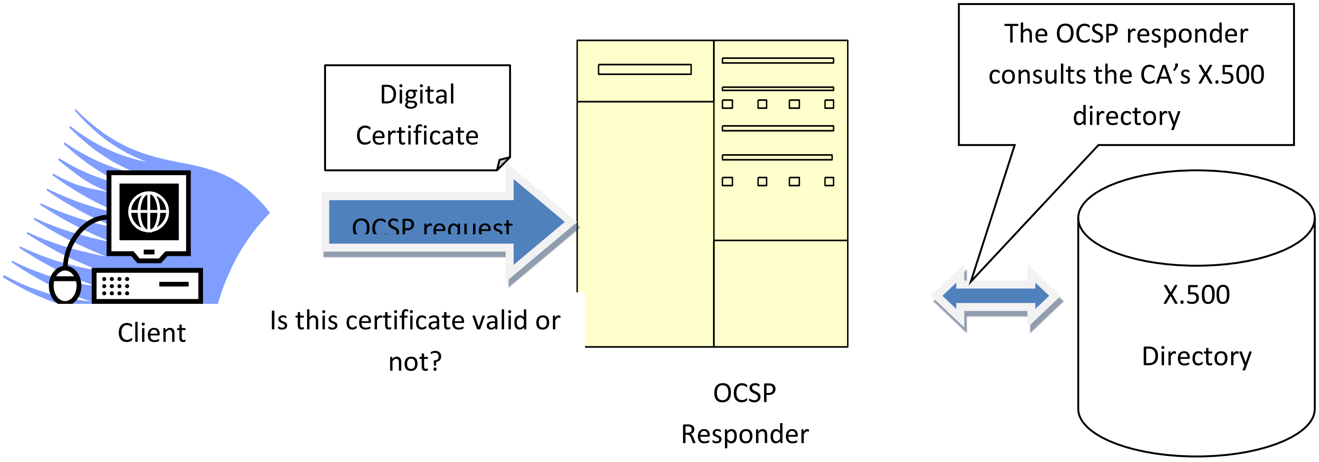
(b) Revoked

(c) Unknown

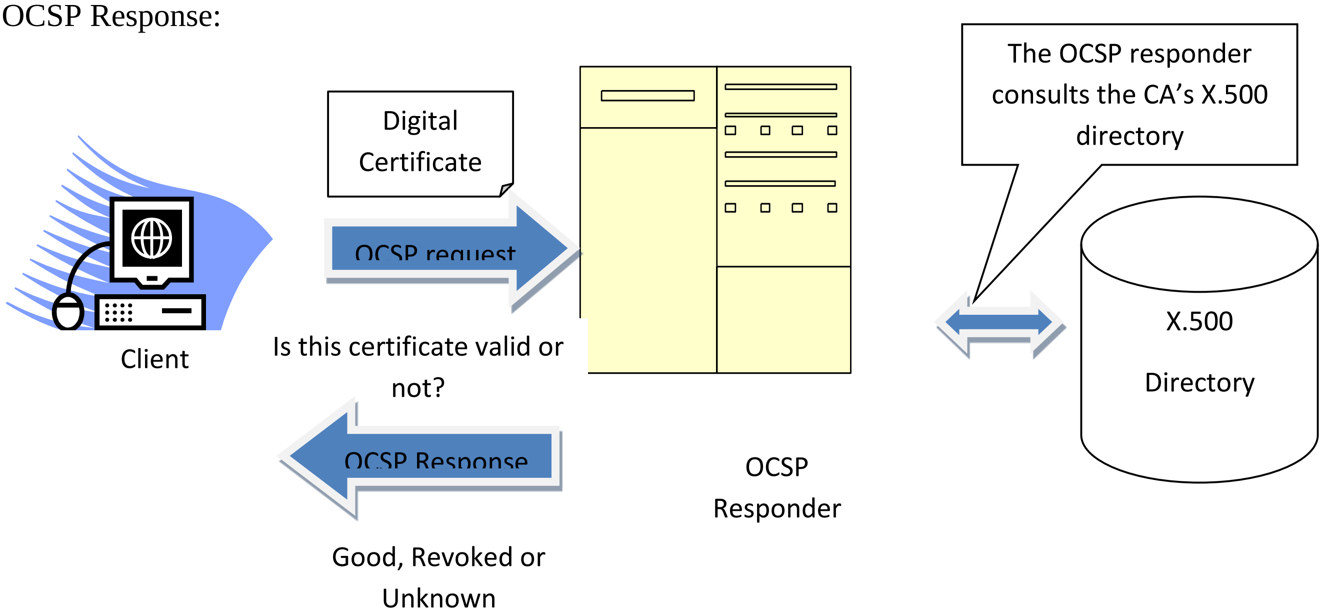
OCSP Request:



OCSP Certificate revocation status check:


Digital Certificate
OCSP request
Client
Is this certificate valid or not?
OCSP Responder
The OCSP responder consults the CA's X.500 directory
X.500 Directory

OCSP Response:



(b) SCVP (Simple Certificate Validation Protocol):

-- It is in the draft stage.

-- It is an online certificate status reporting protocol, designed to deal with the drawbacks of OCSP.

Difference between OCSP & SCVP:

Point	OCSP	SCVP
Client Request	The client sends just the certificate serial number to the server	The client sends the entire certificate to the server. Consequently, the server can perform many more checks
Chain of trust	Only the given certificate is checked.	The client can provide a collection of the intermediate certificates which the server can check

Checks	The only check is whether the certificate is revoked or not.	The client can request for additional checks, type of revocation information to be considered etc.
Returned information	Only the status of the certificate id returned by the server.	The client can specify what additional information it is interested in.
Additional Features	None	The client can request for a certificate to be checked for a background event.

CERTIFICATE TYPES:

(i) Email Certificates: It includes the user's email id. This is used to verify that the signer of an email message has an email id that is the same as it appears in that user's certificate.

(ii) Server-side SSL Certificates: These certificates are useful for merchants who want to allow buyers to purchase goods or services from their online Web site.

(iii) Client-side SSL Certificates: These certificates allow a merchant to verify a client.

(iv) Code-signing Certificates: Many people do not like to download client-side code such as Java applets or ActiveX controls, because of the inherent risks associated with them. In order to alleviate these concerns, the code can be signed by the signer.

- PRIVATE KEY MANAGEMENT:

(i) Protecting Private Keys:

Mechanism	Description
Password Protection	This is most simplest & common mechanism. The private key is stored on the hard disk of the user's computer as a disk file. This file can be accessed only with the help of a password or PIN.
PCMCIA cards	The Personal Computer Memory Card International Association (PCMCIA) are actually chip cards. The private key is stored on such a card.
Tokens	A token stores the private key in an encrypted format. To decrypt it, user has to provide OTP valid for only that particular access.
Biometrics	The private key is associated with a unique characteristic of an individual like fingerprint, retina scan or voice comparison etc.
Smart Cards	In a smart card, the private key of the user is stored in a tamperproof card.

-- The export of private key from one location to another needs a cryptographic standard by the name PKCS#12. This allows a user to export his/her digital certificate & private key in the form of a computer file. The certificate and the private key must be protected as they are moved to another location. For this PKCS#12 standard ensures that they are encrypted using a symmetric key, which is derived from the user's private-key protection password.

(ii) Multiple Key Pairs:

-- The PKI approach also recommends that in serious business applications, users should possess multiple digital certificates, which also means multiple key pairs.

-- The need for this is that one certificate could be strictly used for signing & another for encryption. This ensures that the loss of one of the private keys doesn't affect the complete operations of the user.

Guidelines:

(a) The private key that is used for digital signing (non-repudiation) must not be backed up or archived after it expires. It must be destroyed. This ensures that it is not used by someone else for signing on behalf of the person at a future date.

(b) In contrast, the private key used for encryption/decryption must be backed up after its expiry, so that the encrypted information can be recovered even at a later date.

(iii) Key Update:

Expiry of a certificate can be dealt with in one of the following 2 ways:

(a) The CA reissues a new certificate based on the original key pair.

(b) A fresh key pair is generated, & the CA issues a new certificate based on that new key pair.

The key update process itself can be handled in 2 ways:

(a) The end user has to detect that the certificate is about to expire, a request the CA to issue another one.

(b) The expiry date of the certificate is automatically checked every time it is used, and as soon as it is about to expire, its renewal request is sent to the CA.

(iv) Key Archival:

The CA must plan for & maintain the history of the certificates & the keys of its users to provide future assistance.

THE PKIX MODEL:

-- The X.509 standard defines digital-certificate structure, format & fields. It also specifies the procedure for distributing the public keys. In order to extend such standards & make them universal, the IETF formed the PKIX (Public Key Infrastructure X.509) working group.

PKIX SERVICES:

(a) Registration: End entity (subject) makes itself known to CA usually via RA.

(b) Initialization: It deals with basic problems, such as methodology of verifying that the end-entity is talking to the right CA.

(c) Certification: CA creates a digital certificate for the end-entity & returns it to the end-entity, maintains a copy for its own records, & also copies it in public directories, if required.

(d) Key-Pair Recovery: Keys used for encryption may be required to be recovered at a later date for decrypting some old documents. Key archival & recovery services can be provided by a CA or by an independent key-recovery system.

(e) Key Generation: PKIX specifies that the end-entity should be able to generate private-and public-key pairs, or the CA/RA should be able to do this for the end-entity & then distribute these keys securely to the end-entity.

(f) Key Update: This allows a smooth transition from one expiring key pair to a fresh one, by the automatic renewal request & response.

(g) Cross-Certification: Helps in establishing trust models, so that end-entities that are certified y different CAs can cross verify each other.

(h) Revocation: PKIX provides support for the checking of the certificate status in two modes: online (using OCSP) or offline (using CRL).

PKIX ARCHITECTURAL MODEL:

PKIX has developed comprehensive documents that describe 5 areas of its architectural model:

(a)X.509 V3 Certificate & V2 Certificate Revocation List Profiles: PKIX has grouped all the options that are deemed fit for Internet users, as profile of Internet users.

(b) Operational protocols: These define the underlying protocols that provide the transport mechanism for delivering certificates, CRLs & other management & status information to a PKI user.

(c) Management protocols: These protocols enable exchange of information between the various PKI entities.

(d) Policy Outlines: PKIX defines the outlines for Certificate Policies (CP) & Certificate Practice Statements (CPS) in RFC2527. These define policies for the creation of a document such as a CP, which determines what considerations are important when choosing a type of certificate for a particular application domain.

(e) Timestamp & Data Certification Services: Timestamping service is provided by a trusted 3rd party called Timestamp Authority. The purpose is to sign a message to guarantee that it existed prior to a specific date & time.

PUBLIC KEY CRYPTOGRAPY STANDARDS (PKCS) :

-- PKCS model was initially developed by RSA Laboratories.

-- The main purpose is to standardize Public Key Infrastructure (PKI). The standardization in many respects, such as formatting, algorithms & APIs. This would help organizations develop and implement inter-operable PKI solutions, rather than everyone choosing their own standard.

Standard	Purpose	Details
PKCS#1	RSA Encryption Standard	The RSA encryption standard. This standard defines mechanisms for encrypting and signing data using the RSA public key system.
PKCS#2	RSA Encryption Standard for Message Digests	This standard outlined the message-digest calculation. However this is now merged with PKCS#1 & doesn't have an independent existence.
PKCS#3	Diffie-Hellman Key Agreement Standard	The Diffie-Hellman key-agreement standard. This defines the Diffie-Hellman key agreement protocol.
PKCS#4	NA	Merged with PKCS#1
PKCS#5	Password based Encryption (PBE)	The password-based encryption standard (PBE). This describes a method to generate a Secret Key based on a password.
PKCS#6	Extended Certificate Syntax Standard	This is currently being phased out in favor of X509 v3.
PKCS#7	Cryptographic Message Syntax Standard	The cryptographic message syntax standard. This defines a generic syntax for messages which have cryptography applied to it.
PKCS#8	Private Key Information Standard	The private-key information syntax standard. This defines a method to store Private Key Information.
PKCS#9	Selected Attribute Types	This defines selected attribute types for use in Other PKCS standards.
PKCS#10	Certificate Request Syntax Standard	The certification request syntax standard. This describes a syntax for certification requests.
PKCS#11	Cryptographic Token Interface Standard	The cryptographic token interface standard. This defines a technology

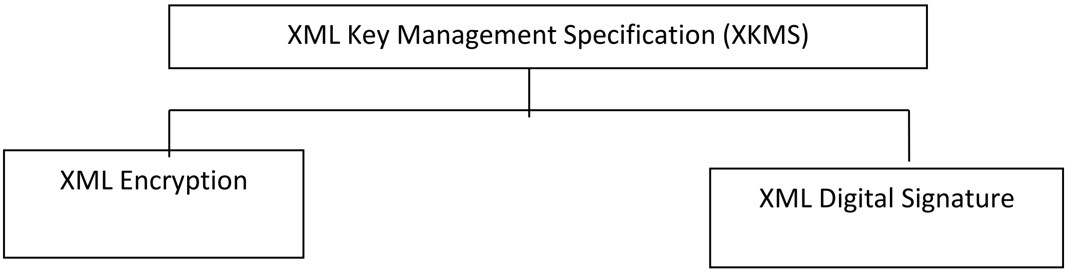
		independent programming interface for cryptographic devices such as smartcards.
PKCS#12	Personal Information Exchange Syntax Standard	The personal information exchange syntax standard. This describes a portable format for storage and transportation of user private keys, certificates etc.
PKCS#13	Elliptic Curve Cryptography Standard	The elliptic curve cryptography standard. This describes mechanisms to encrypt and sign data using Elliptic curve cryptography.
PKCS#14	Pseudo-Random Number Generation Standard	This covers pseudo random number generation (PRNG). This is currently under active development.
PKCS#15	Cryptographic Token Information syntax standard	The cryptographic token information format standard. This describes a standard for the format of cryptographic credentials stored on cryptographic tokens.

XML, PKI & SECURITY:

-- The technology of PKI is quite promising, but it lacks operability among vendor solutions.

-- The EXtensile Markup Language (XML) is at the centerstage of the modern world of technology. XML forms the backbone of the upcoming technologies, such as Web services.

XML & Security:



XML ENCRYPTION:

-- The XML encryption can encrypt an entire document, or its selected portions.

-- One or all the following portion of XML document can be encrypted:

(a) Entire XML document.

(b) An element & all its sub-elements

(c) The content portion of XML document\

(d) A reference to a resource outside of an XML document.

STEPS INVOLVED IN XML ENCRYPTION:

(i) Select the XML to be encrypted.

(ii) Convert the data to be encrypted in a canonical form (optional).

(iii) Encrypt the result using public key encryption.

(iv) Send the encrypted XML document to the intended recipient.

XML DIGITAL SIGNATURE:

Elements in XML digital signature process:

Element	Description
SignedInfo	Contains the signature itself.
Canonicalization Method	Specifies the algorithm used to canonicalize the SignedInfo element, before it is digested as a part of signature creation.
Signature Method	Specifies the algorithm used to transform the canonicalized SignedInfo element into the SignatureValue element.
Reference	Includes the mechanism used for calculating the message digest & the resulting digest value over the original data.
KeyInfo	Indicates the key that can be used to validate the digital signature.
Transforms	Specifies the operations performed before calculating the digest, such as compression, encoding etc.
Digest Method	Specifies the algorithm used to calculate the message digest.
Digest Value	Contains the message digest of the original message.

STEPS:

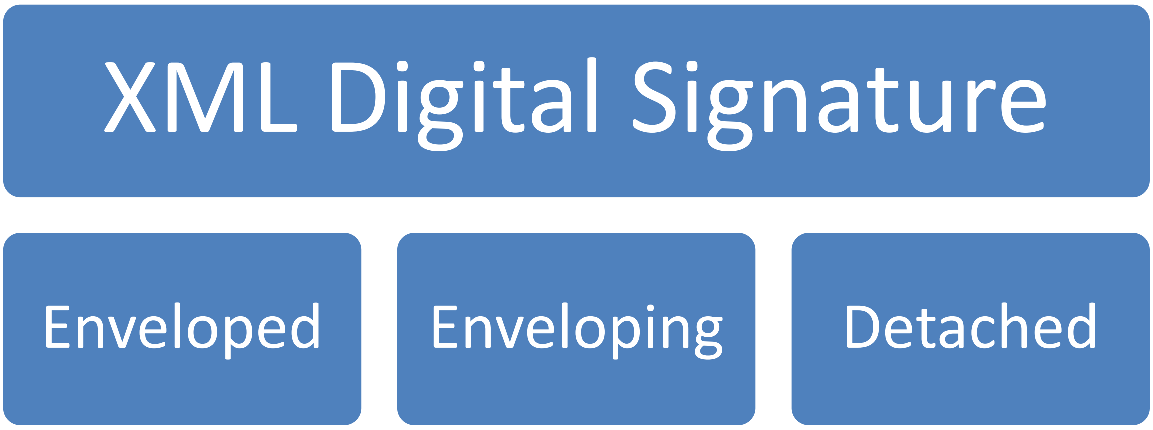
(i) Create a SignedInfo element with SignatureModel, Canonicalization Method, & References.

(ii) Canonicalize the XML document.

(iii) Calculate the Signaturevalue, depending on the algorithms specified in the SignedInfo element.

(iv) Create the digital signature which also includes the SignedInfo, KeyInfo, and SignatureValue elements.

CLASSIFICATION OF DIGITAL SIGNATURES:


XML Digital Signature

Enveloped

Enveloping

Detached

(a) Enveloped: The signature is inside the original document which is being digitally signed.

(b) Enveloping: The original document is inside the signature.

(c) Detached: It has no enveloping concept at all, it is separate from the original document.

- INTERNET-SECURITY PROTOCOLS

BASIC CONCEPTS:

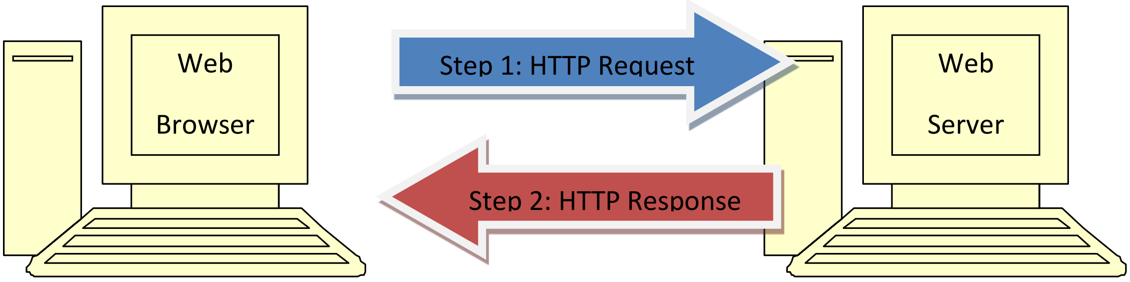
(i) STATIC WEB PAGES:

-- In static web pages, browser sends an HTTP request, the server sends an HTTP response and the communication between them ends.

-- A Web page written in HTML is created by an application developer & is stored on a Web server. Whenever any user requests for that page, the Web server sends back the page without performing any additional processing. All it does is to locate the page on its hard disk, add HTTP headers, and send back an HTTP response.

-- Thus the contents of the Web page doesn't change depending on the request- they are always same. Hence the name static has been assigned.

STATIC WEB PAGE:



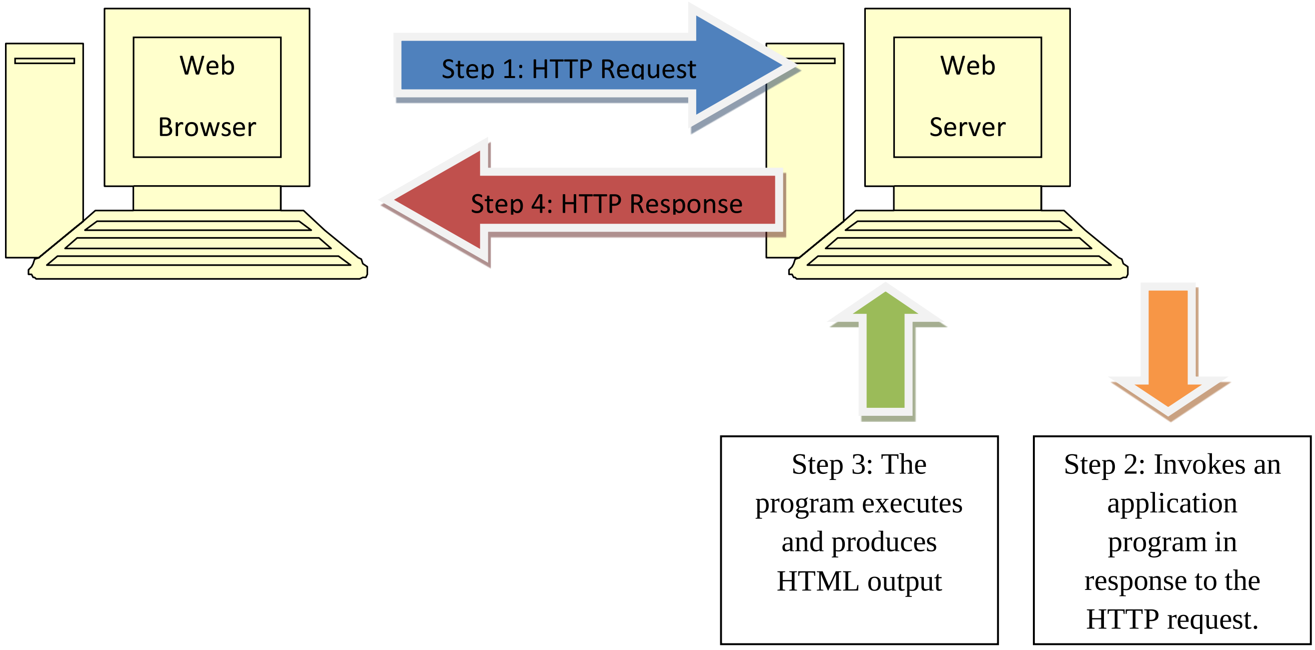
(ii) DYNAMIC WEB PAGES:

-- The contents of the dynamic web page changes depending on the number of parameters.

-- It involves server-side programming.

-- When the user requests for a dynamic Web page, the server actually invokes a program that resides on its hard disk. The program in turn might access databases, perform transaction processing etc. However in any case, the program outputs HTML, which is used to construct an HTTP response by the Web server. The Web server sends the HTTP response thus formed, back to the Web browser.

DYNAMIC WEB PAGE:



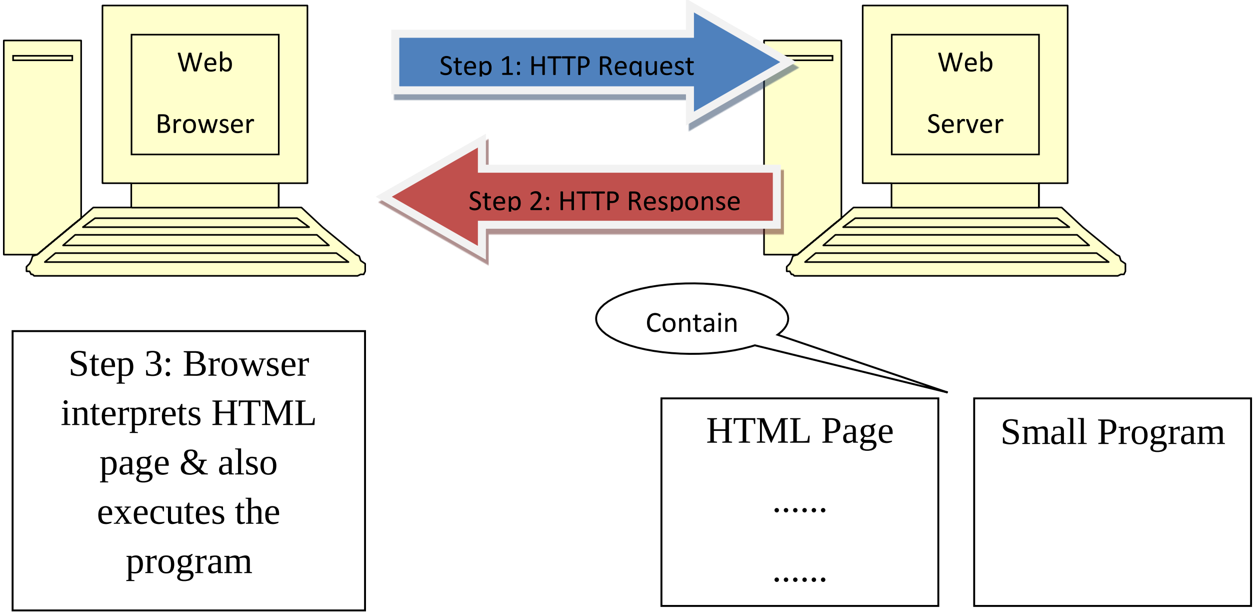
Step 3: The program executes and produces HTML output

Step 2: Invokes an application program in response to the HTTP request.

(iii) ACTIVE WEB PAGES:

-- When a client sends an HTTP request for an active Web page, the Web server sends back an HTTP response that contains an HTML page as usual. In addition, the HTML page also contains a small program that executes on the client computer inside the Web browser.

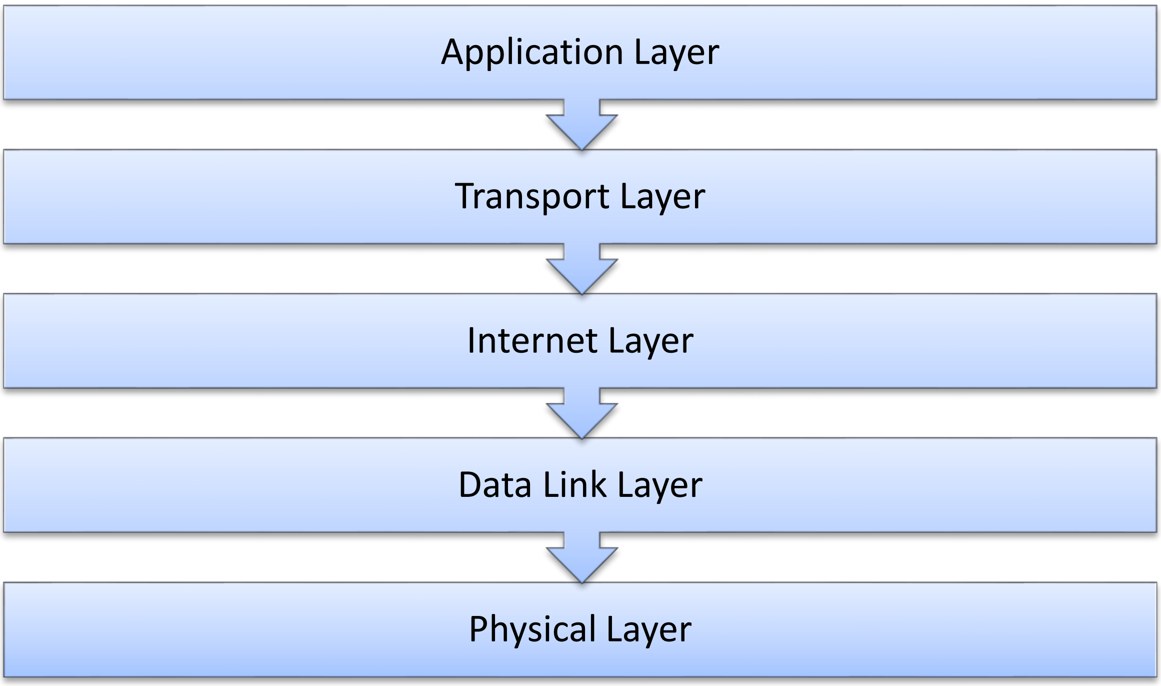
ACTIVE WEB PAGE:



-- Client Pull: When the client keeps requesting information automatically from the server (i.e. the client pulls information) after a specified interval, this technology is called client pull.

(iv) TCP/IP: This software is a translator that is a combination of many protocols that facilitate communication between computers over the Internet. It specifies how a browser should identify a server, how it should send an HTTP request to server, how should a server respond, what to do in case of an error etc.

TCP/IP Layers



SECURE SOCKET LAYER (SSL):

-- It is an Internet protocol for the secure exchange of information between a Web browser & a Web server.

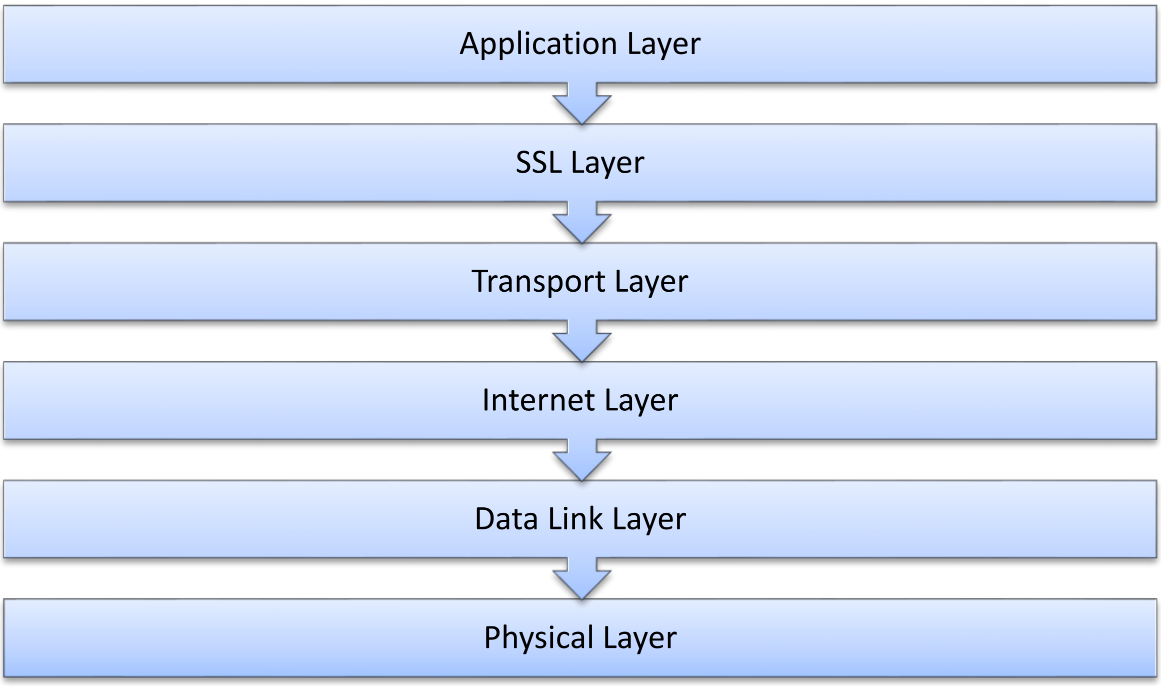
-- It provides 2 basic security services:

(a) Authentication

(b) Confidentiality

-- It provides a secure pipe between Web browser & Web server.

Position of SSL in TCP/IP:



WORKING OF SSL:

SSL has 3 sub-protocols:

(1) Handshake Protocol

(2) Record Protocol

(3) Alert Protocol

(1) The Handshake Protocol:

-- It consists of a series of messages between client & server with following format:

Type	Length	Content
1 Byte	3 Bytes	1 or more bytes

The handshake protocol is made up of 4 phases:

(a) Establish security capabilities

(b) Server authentication & key exchange

(c) Client authentication & key exchange

(d) Finish

Phase 1: Establish Security Capabilities:

This is used to initiate a logical connection & establish the security capabilities associated with that connection.

This consists of 2 messages:

(a) client hello

(b) server hello

Client hello parameters:

(a) Version: This field identifies the highest version of SSL that the client can support. This can be 2,3 or 3.1

(b) Random: This field is useful for the later, actual communication between the client & the server. It consists of 2 sub-fields:

(i) A 32-bit date-time field that identifies the current system date & time on the client computer.

(ii) A 28-byte random number generated by the random-number generator software built inside the client computer.

(c) Session Id: This is a variable-length session identifier. If this field contains a non-zero value, it means that there is already a connection between the client & the server, and the client wishes to update the parameters of that connection. A zero value indicates that the client wants to create a new connection to the server.

(d) Cipher suite: This list contains a list of the cryptographic algorithms supported by the client in the decreasing order of preference.

(e) Compression Method: This field contains a list of compression algorithms supported by the client.

Server hello parameters:

(a) Version: This field identifies the lower versions suggested by the client & the highest supported by the server.

(b) Random: This field has the same structure as the Random field of the client. However, the Random value generated by the server is completely independent of the client's Random value.

(c) Session Id: If the session id value sent by the client was non-zero, the server uses the same value. Otherwise, the server creates a new session id & puts it in this field.

(d) Cipher Suite: Contains a single cipher suite, which the server selects from a list sent earlier by the client.

(e) Compression Method: Contains a compression algorithm, which the server selects from a list sent earlier by the client.

Phase 2: Server Authentication & Key exchange:

The server initiates this second of the SSL handshake, and is the sole sender of all the messages in this phase. The client is the sole recipient of all these messages. This phase contains 4 steps:

(a) Certificate: The server sends its digital certificate & the entire chain leading up to root CA to the client. This helps the client to authenticate the server using the server's public key from the server's certificate.

(b) Server Key Exchange: It is optional. It is used only if the server doesn't send its digital certificate to the client in the Step 1 above.

(c) Certificate Request: The server can request for the client's digital certificate.

(d) Server hello done: It indicates to the client that its portion of the hello message is complete. This indicates to the client that the client can now verify the certificates sent by the server & ensure that all the parameters sent by the server are acceptable.

Phase 3: Client Authentication & Key Exchange:

The client initiates this 3rd phase of the SSL handshake, & is the sole sender of all the messages in this phase. The server is the sole recipient of all these messages. This phase contains 3 steps:

(a) Certificate: It is optional. This is performed only if the server had requested for the client's digital certificate.

(b) Client Key Exchange: It allows the client to send information to the server, but in the opposite direction. This information is related to the symmetric key that both the parties will use in this session.

(c) Certificate Verify: It is necessary if the server had demanded client authentication.

Phase 4: Finish: The client initiates this 4th phase of the SSL handshake, which the server ends. It contains 4 steps:

(a) Change Cipher specs

(b) Finished

(c) Change Cipher specs

(d) Finished

The 1st two messages are from the client. The server responds back with two identical messages.

(2) The Record protocol:

This comes into picture in SSL after a successful handshake is completed between the client & server. This protocol provides 2 services:

(a) Confidentiality: This is achieved by using the secret key that is defined by the handshake protocol.

(b) Integrity: The handshake protocol also defines a shared secret key (MAC) that is used for assuring the message integrity.

Steps in Record Protocol:

(a) Application data: The SSL record protocol takes an application message as input.

(b) Fragmentation: It fragments the message into smaller blocks.

(c) Compression: The fragmented blocks are compressed.

(d) Addition of MAC: Using the shared secret key established previously in the handshake protocol, the Message Authentication Code (MAC) for each block is calculated.

(e) Encryption: Using the symmetric key established previously in the handshake protocol, the output of the previous step is now encrypted.

(f) Append Header: Finally a header is added to the encrypted block.

(3) The Alert Protocol:

When either the client or the server detects an error, the detecting party sends an alert message to the other party.

If the error is fatal, both the parties immediately close the SSL connection.

Both the parties also destroy the session identifiers, secrets & keys associated with this connection before it is terminated.

Closing & Resuming SSL Connections: Before ending their connection, the client & server must inform each other that their side of the connection is ending. Each party sends a Close notify to the other party. This ensures a graceful closure of the connection.

- SECURE HYPERTEXT TRANSFER PROTOCOL (SHTTP):

-- It is a set of security mechanisms defined for protecting the Internet traffic.

-- This includes the data-entry forms & Internet-based transactions.

-- It supports both authentication & encryption of HTTP traffic between the client & the server.

-- The key difference between SSL & SHTTP is that SHTTP works at the level of individual messages.

-- It can encrypt & sign individual messages.

-- SSL can’t perform digital signatures.

TIME STAMPING PROTOCOL (TSP):

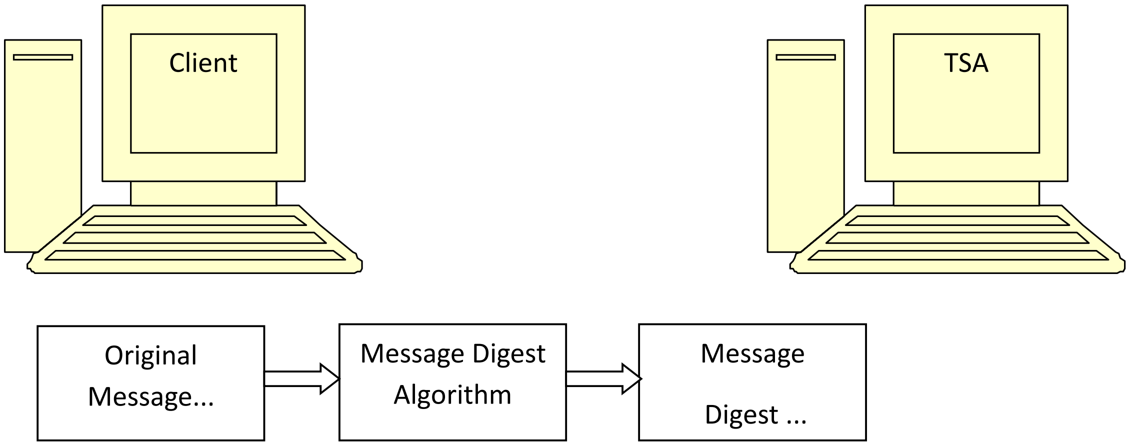
TSP provides proof that a certain piece of data existed at a particular time. This PKI service is provided by an authority called Time Stamping Authority (TSA).

TSP is a simple request-response protocol similar to HTTP.

Working of TSP:

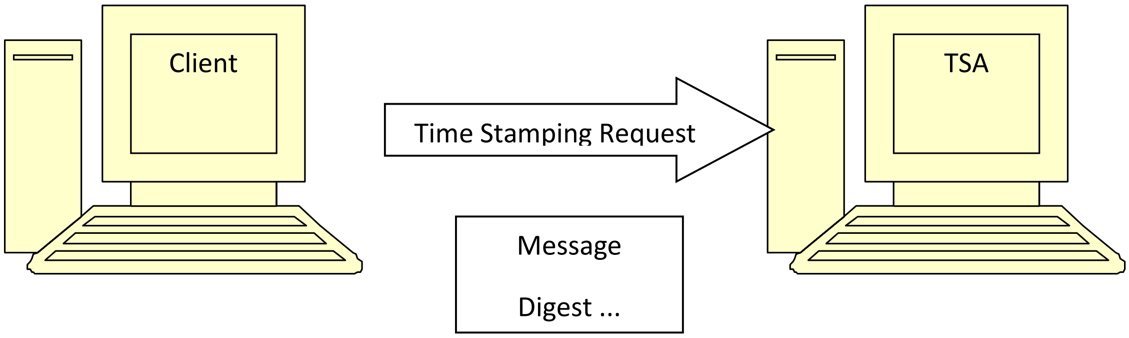
Step 1: Message Digest Calculation:

Firstly, the client requiring a timestamp calculates a MD of the original massage, which needs a timestamp from the TSA. The client should use a standard message digest algorithm such as MD5 or SHA-1 for this purpose.



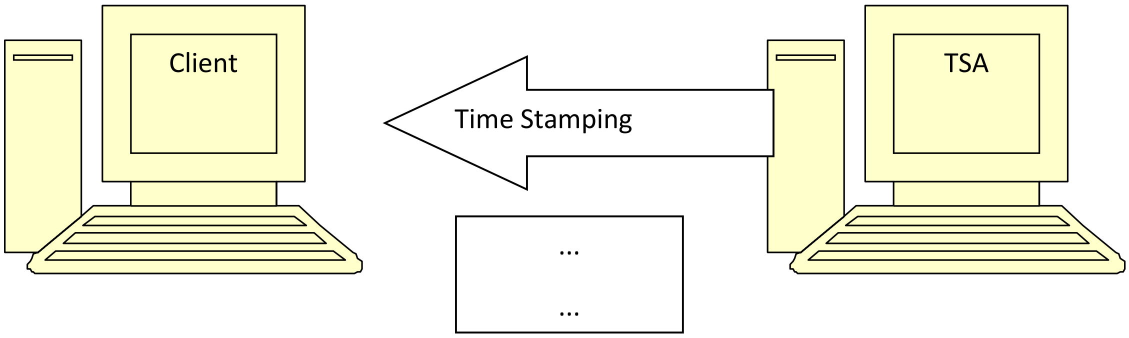
Step 2: Time Stamping Request:

Now, the client sends the message digest calculated in step 1 to the Time Stamp Authority (TSA) for getting it time stamped. This is called Time Stamping Request.



Step 3: Time Stamping Response:

In response to the client's request, the TSA might decide to grant or reject the time stamp. If it decides to accept the request & process it, it signs the client's request together with the timestamp by the TSA private key. Regardless, it returns a Time Stamping Response back to the client.



SECURE ELECTRONIC TRANSACTION (SET):

-- SET is an open encryption & security specification that is designed for protecting credit-card transactions on the Internet.

SET Services:

(i) It provides a secure communication channel among all the parties involved in an e-commerce transaction.

(ii) It provides authentication by the use of digital certificates.

(iii) It ensures confidentiality, because the information is only available to the parties involved in a transaction, and that too only when & where necessary.

SET Participants:

(a) Cardholder: Using the Internet, consumers & corporate purchasers interact with merchants for buying goods & services. A cardholder is an authorized holder of a payment card such as Master Card or Visa that has been issued by an issuer.

(b) Merchant: A merchant is a person or organization that wants to sell goods or services to card holders.

(c) Issuer: It is a financial institution that provides a payment card to a cardholder.

(d) Acquirer: This is a financial institution that has a relationship with merchants for processing payment-card authorizations & payments.

(e) Payment Gateway: This is a task that can be taken up by the acquirer or it can be taken up by an organization as a dedicated function.

(f) Certification Authority: This is a authority that is trusted to provide public key certificates to cardholders, merchants & payment gateways.

The SET PROCESS:

(a) The Customer Opens an Account: The customer opens a credit-card account that supports electronic payment mechanisms & the SET protocol.

(b) The Customer Receives a Certificate: After the customer's identity is verified, the customer receives a digital certificate from a CA.

(c) The Merchant Receives a Certificate: A merchant that wants to accept a certain brand of credit cards must possess a digital certificate.

(d) The Customer Places an Order: This is a typical shopping cart process wherein a customer browses the list of items available.

(e) The merchant is verified: The merchant also sends its digital certificate to the customer.

(f) The Order & Payment details are sent: The customer sends both order & payment details to the merchant along with the customer's digital certificate.

(g) The Merchant Requests Payment Authorization: The merchant forwards the payment details by the customer to the payment gateway via the acquirer & requests the payment gateway to authorize the payment.

(h) The Payment Gateway Authorizes the Payment: Using the credit-card information received from the merchant, the payment gateway verifies the details of the customer's credit card with the help of the issuer, and either authorizes or rejects the payment.

(i) The Merchant Confirms the Order: Assuming that the payment gateway authorizes the payment, the merchant sends a confirmation of the order to the customer.

(j) The Merchant Provides Goods or Services: The merchant now ships the goods or provides the services as per the customer's orders.

(k) The Merchant Requests Payment: The payment gateway receives a request from the merchant for making the payment.

SSL Vs SET:

Issue	SSL	SET
Main Aim	Exchange of data in an encrypted form	E-commerce related payment mechanism
Certification	Two parties exchange certificates	All the involved parties must be certified by a trusted 3rd party.
Authentication	Mechanisms not strong	Strong mechanisms for authenticating all the parties involved.
Risk Of Merchant Fraud	Possible, since customer gives financial data to merchant.	Unlikely, since customer gives financial data to payment gateway.
Risk Of Customer Fraud	Possible, no mechanism exists if a customer refuses to pay later.	Customer has to digitally sign payment instructions.
Action in case of customer fraud	Merchant is liable	Payment gateway is liable
Practical Usage	High	Low at the moment, expected to grow

3-D SECURE PROTOCOL:

In spite of its advantages, SET has one limitation: it doesn't prevent a user from providing someone else's credit-card number.

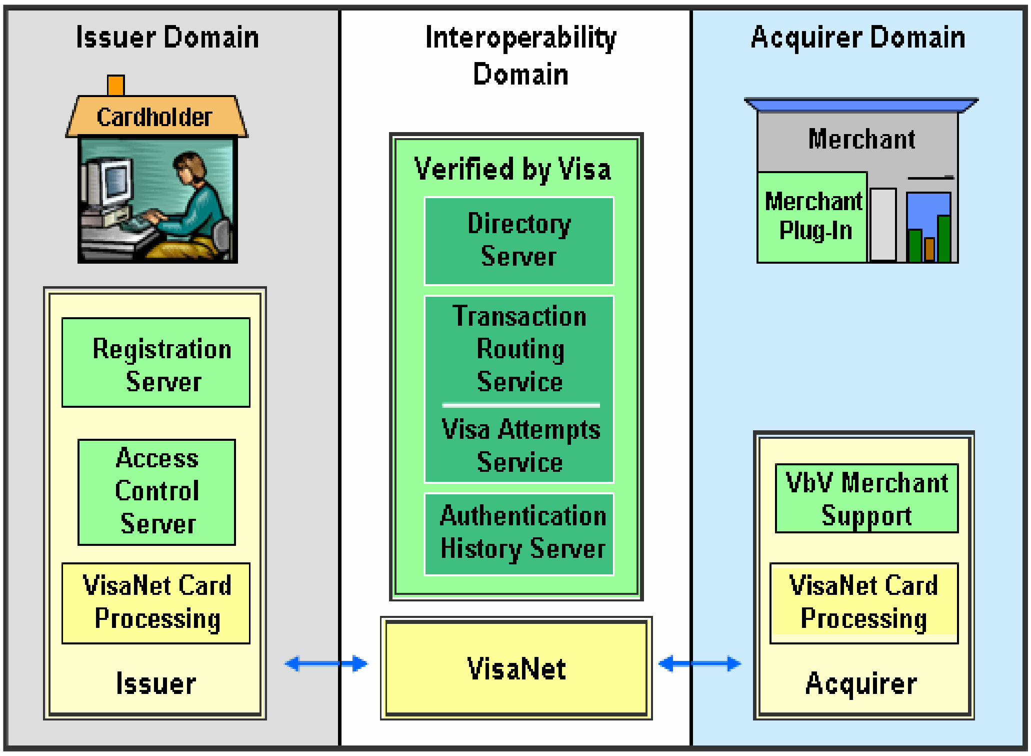
In 3-D secure protocol, the cardholder has to enrol on the issuer bank's Enrolment Server.

Protocol:

Step-1: The user shops using the shopping cart on the merchant site & decides to pay the amount.

Step-2: The user is redirected to the issuer bank's site. The user is asked for a static password. The bank verifies this password & then sends the appropriate success/failure message to the merchant, based on which the merchant takes an appropriate decision & shows the corresponding screen to the user.

The 3 Domains of 3-D Secure:



TRANSACTION FLOWS:

The Verified by Visa program includes two transaction flows for the cardholder:
• Cardholder enrolment or activation in the Verified by Visa program as described as Cardholder Enrolment/Activation.
• Cardholder authentication during an online purchase at a participating merchant described as Online Purchases.

(A) Cardholder Enrolment/Activation:
-- Issuers may provide one or more of several enrolment options to their cardholders. Cardholders may enter a password that is used for authentication when shopping at the website of a merchant that participates in the Verified by Visa program or be authenticated by entering the requested information to verify their identity. Cardholders can also select a Personal

Assurance Message that assures them that the password prompt window is actually from their card issuer.

(i) Enrolment: Activation During Shopping:
The most popular form of cardholder enrolment in Verified by Visa is known as "Activation During Shopping." This option facilitates a more targeted adoption rate and has helped enrol a critical mass of cardholders into Verified by Visa. With Activation during Shopping, cardholders who have not set up their Verified by Visa password receive a request to do so when shopping at a participating merchant. The cardholder is presented with a series of security questions to be answered to verify the cardholder's identity. Once the identity is confirmed, the cardholder is asked to select a password.

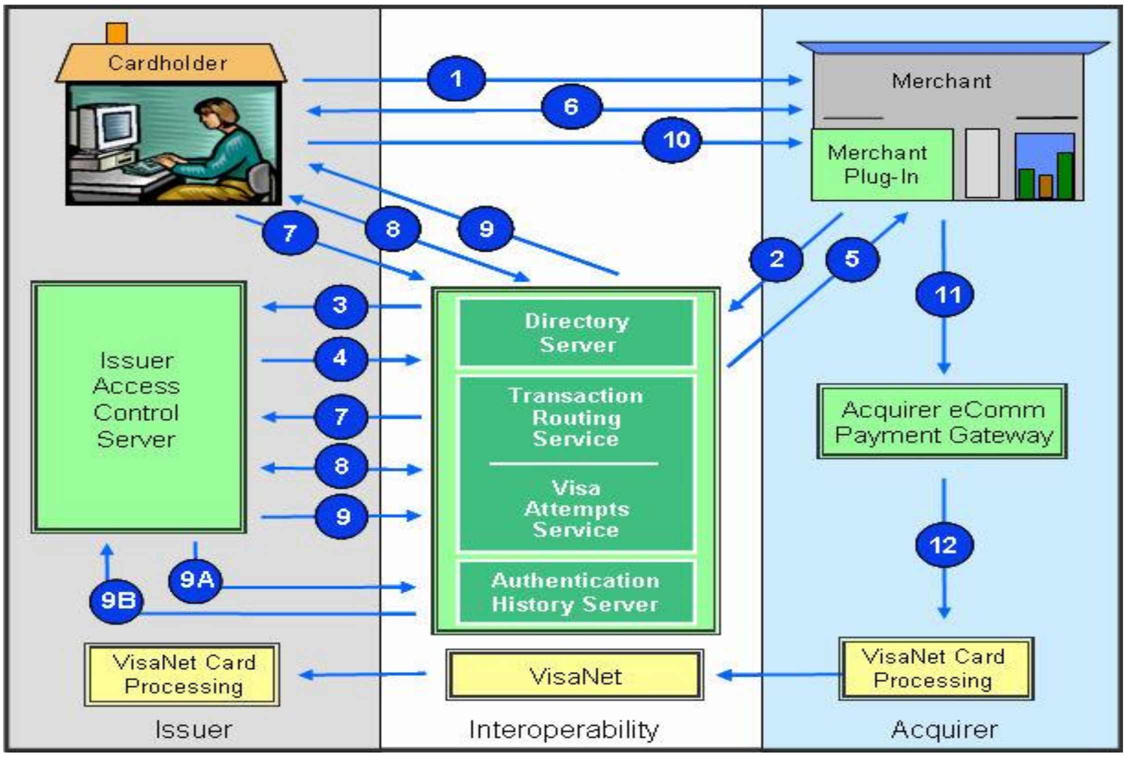
(ii) Enrolment: Cardholder Registration:
The second common form of enrolment is to give cardholders the option to visit a Verified by Visa website provided by the financial institution that issued their Visa card. After cardholders enter their card number, they are presented with a series of security questions to be answered to verify the cardholder's identity. Once the identity is confirmed, the cardholder is asked to select a password and a Personal Assurance Message.

(iii) Enrolment: Registration Complete:
After enrolment is complete, each time the cardholder makes an online purchase at a participating merchant's website, a Verified by Visa authentication page will appear to verify the identity of the cardholder, as described in Online Purchases.

(B) Online Purchases:
After enrolling as described above, the cardholder is ready to use Verified by Visa at any participating merchant. Figure below illustrates the purchase transaction flow, which is described in the remainder of this section. Descriptions of each transaction step are in the sections that follow.

Step 1: Cardholder Finalizes Purchase:
The cardholder browses at a participating merchant's website, adds items to the shopping cart, provides information required for checkout (by key entering data or by using an electronic wallet, a merchant one click service, or some other form-fill method), then clicks "Buy". The merchant now has all necessary data, including Primary Account Number (PAN) of the card presented for the purchase.
Steps 2-7 that follow are invisible to the cardholder.
Step 2: Merchant Server Plug-in Initiates 3-D Secure Processing:
When the cardholder clicks buy, the Merchant Server Plug-in (MPI) is activated. The MPI sends the PAN and other information to the Visa Directory Server to determine whether the card is in a participating range.

Purchase Transaction Flow:



Step 3: Visa Directory Server Processes Request

If merchant authentication is successful, the Visa Directory Server forwards the merchant query to the appropriate Access Control Server (ACS) to determine whether authentication (or proof of authentication attempt) is available for the card PAN. If merchant authentication fails, the Directory Server returns an Error and the Verified by Visa transaction is terminated. If no appropriate ACS is available or the cardholder is not participating in Verified by Visa, the Visa Directory Server routes the request to the Visa Attempts Service which will process the authentication on behalf of the issuer.

Step 4. ACS Responds to Visa Directory Server

The issuer ACS, or Visa Attempts Service if an issuer ACS is not available, determines whether authentication is available for the card's PAN, prepares a response, and sends it to the Visa Directory Server.

Step 5: Visa Directory Server Returns Response

The Visa Directory Server returns the ACS response (or its own) to the MPI.
If authentication is available, the response includes the URL of the Visa Transaction Routing Service and the issuer ACS to which the merchant will send the Payer Authentication Request.

Step 6: MPI Sends Payer Authentication Request

If authentication (or proof of an attempted authentication) is not available, then the MPI advises the merchant commerce server that authentication is not available, and processing continues with Step 12. The MPI sends the Payer Authentication Request to the ACS via the Visa Transaction Routing Service via the cardholder's device (PC browser or other device), using the URL received in Step 5. The Payer Authentication Request contains information about the purchase transaction.

Step 7: ACS Receives Payer Authentication Request

The Visa Transaction Routing Service receives the Payer Authentication Request and forwards it to the appropriate issuer ACS.

Step 8: ACS Authenticates Cardholder

The ACS formats an authentication request for the cardholder. The authentication request is returned via the Visa Transaction Routing Service to the cardholder's browser. The cardholder may be authenticated using processes applicable to the PAN (password, Activation During Shopping, etc.).

Step 9: ACS Returns Authentication Results

The ACS returns the signed Payer Authentication Response to the Visa Transaction Routing Service which forwards the response to the MPI via the cardholder's device.

- Step 9A: Whether or not authentication was successful, the ACS sends a copy of the Payer Authentication Response, including related data, to the Authentication History Server.
- Step 9B: The Authentication History Server provides an acknowledgment response that the Payer Authentication Response transaction data was received.

The Authentication History Server serves as the database of record for dispute resolution.

Step 10: MPI Receives Payer Authentication Response

The cardholder's device forwards the signed Payer Authentication Response to the MPI.

Step 11: MPI Processes Response

The MPI validates the signature on the Payer Authentication Response along with other data in the response. The MPI, then, passes the results of the authentication attempt to the merchant commerce server.

Step 12: Authorization Processing

Based on the data received from the MPI, the merchant commerce server determines whether to proceed with authorization. If the merchant commerce server advises the MPI that authentication failed, the merchant should request another form of payment from the shopper.

If authorization is appropriate:

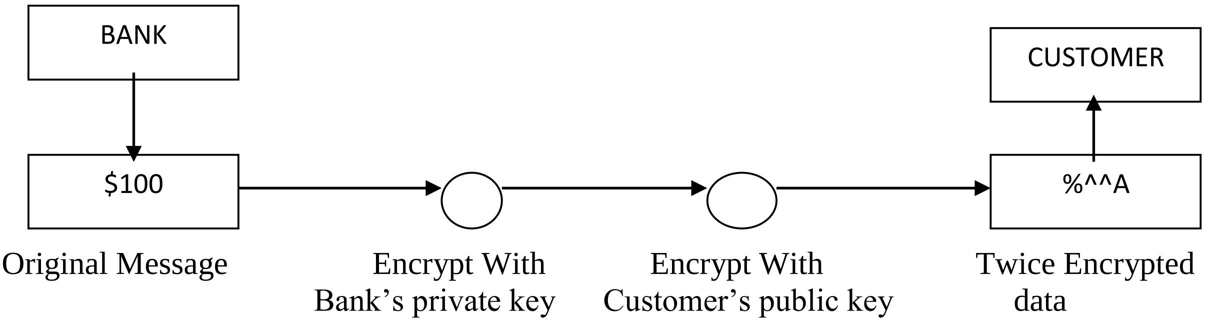
- The merchant commerce server sends an authorization request to the merchant's acquirer or merchant payment processor. The authorization request includes the Electronic Commerce Indicator (ECI) appropriate to the authentication status and the CAVV, when required.
- The acquirer sends the authorization request, including Verified by Visa authentication information, to the issuer via VisaNet.
- The issuer receives and processes the authorization request. When the CAVV is passed in BASE I, either the issuer or Visa on the issuer's behalf, will perform CAVV verification. The issuer returns an authorization response. The issuer may choose to approve or to decline the authorization request for reasons unrelated to the Verified by Visa authentication (e.g., insufficient funds, closed account, etc.).
- If the issuer authorizes the transaction, the merchant displays an order confirmation as usual, providing the cardholder with details about the order, delivery, and the merchant's customer service.

ELECTRONIC MONEY:

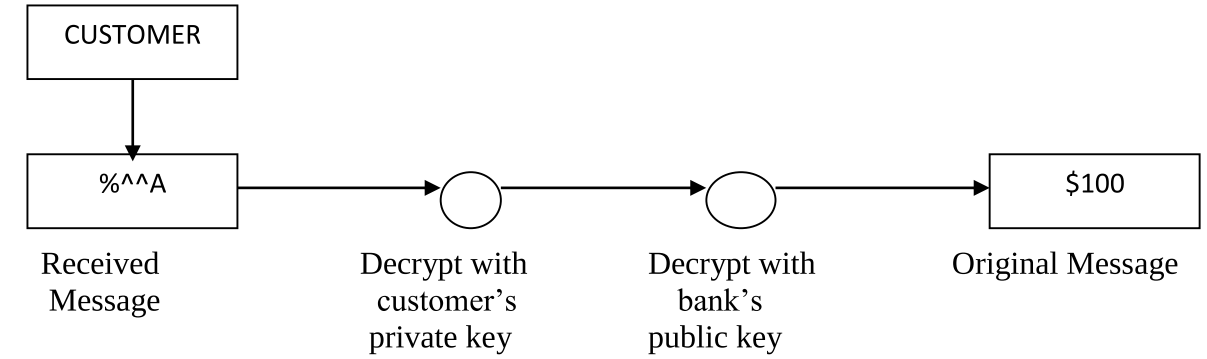
Electronic Money also called electronic cash or digital cash is one more way of making payments on the Internet. It is the money represented by computer files.

Security Mechanisms in Electronic Money:

Step 1: The bank sends the electronic money to the customer as shown in the figure below:


BANK
CUSTOMER
$100
%^^A
Original Message
Encrypt With Bank's private key
Encrypt With Customer's public key
Twice Encrypted data

Step 2: The customer receives the money & decrypts it as shown in the figure:

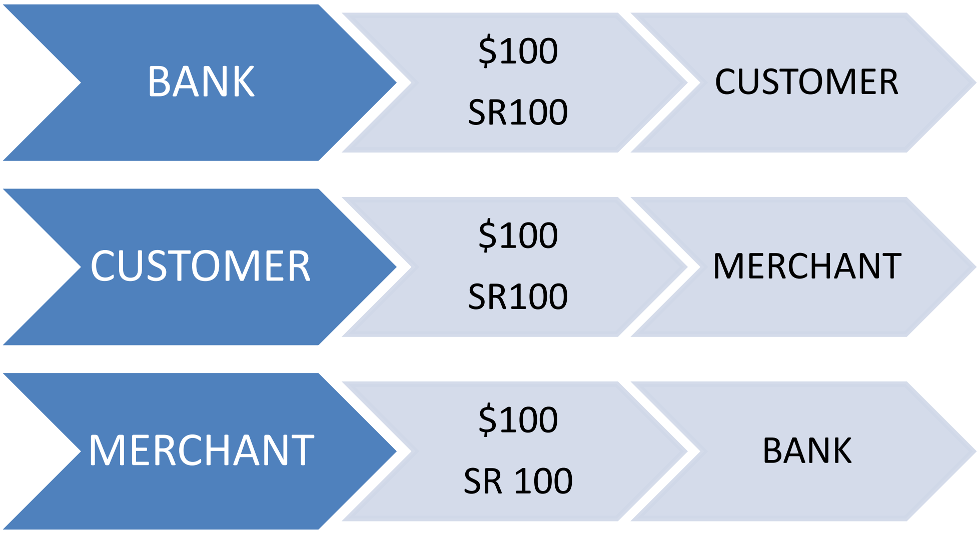

CUSTOMER
%^^A
$100
Received Message
Decrypt with customer's private key
Decrypt with bank's public key
Original Message

TYPES OF ELECTRONIC MONEY:

Classification based on Tracking of the money:

(a) Identified Electronic Money: It works more or less like a credit card. The progress of the identified electronic money from the very 1st time it is issued by the bank to one of its customers, up to its final return to the bank can be easily tracked by the bank.

STEPS INVOLVED IN IDENTIFIED ELECTRONIC MONEY:



(i) The Bank generates the serial number & sends it along with the electronic money to the customer.

(ii) The customer spends the money so the merchant has it now.

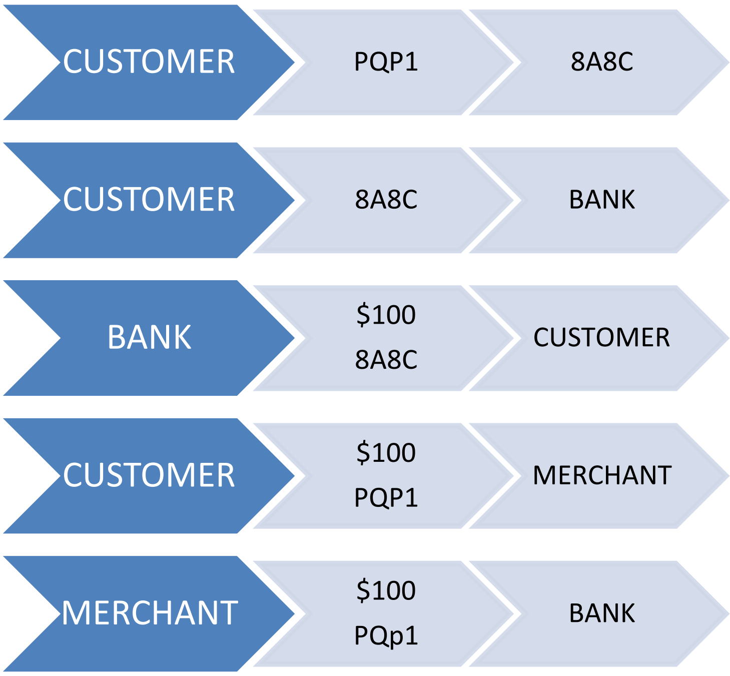
(iii) The merchant now wants to encash the electronic money from the bank. The money still has the same serial number.

(b) Anonymous Electronic Money: It is also called blinded money, works as real hard cash. There is no trace of how the money was spent. Products like DigiCash provide this kind of electronic money to Internet users to spend by tying up with banks.

In identified electronic money, the bank creates the serial number whereas in anonymous electronic money the customer generates the serial number.

STEPS INVOLVED IN ANONYMOUS ELECTRONIC MONEY:

Original Number Blinded Number



(i) The customer generates a random number, & from it, creates another number called as blinded number.

(ii) The customer sends the blinded number to the bank.

(iii) The bank sends the electronic money along with the blinded number to the customer.

(iv) During the actual transaction, the customer doesn't use the blinded number, instead he uses the original number.

(v) The merchant & the bank now have the original number-they can't trace the money as they don't know the relationship between the original number & the blinded number.

Classification based on the involvement of the bank in the transaction:

(a) Online Electronic Money: In this type, the bank must actively participate in the transaction between the customer & the merchant.

(b) Offline Electronic Money: In this type, the bank doesn't participate in the transaction between the customer & the merchant.

DOUBLE SPENDING PROBLEM:

If we combine the two ways of classifying electronic money, there are 4 possibilities:

(i) Identified Online Electronic Money

(ii) Identified Offline Electronic Money

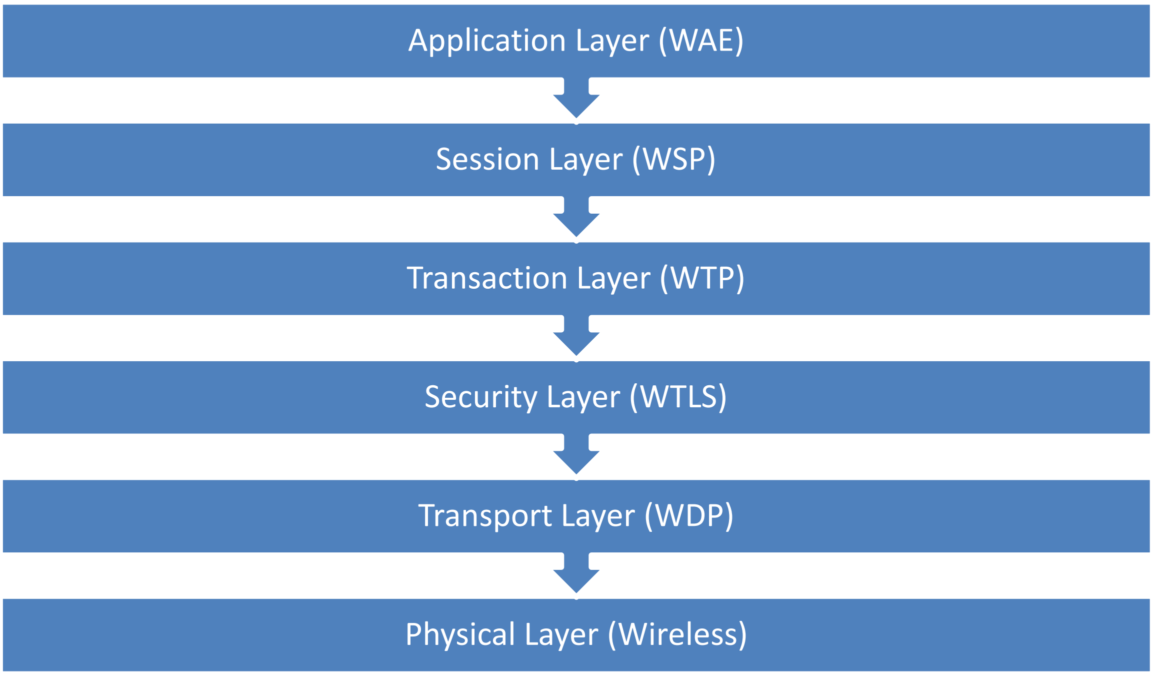
(iii) Anonymous Online Electronic Money

(iv) Anonymous Offline Electronic Money

Of the four, the last one can create double spending problem. A customer could arrange for anonymous electronic money by using the blinded money concept. Later on, he could spend it offline more than once in quick succession with two different merchants. Since the bank is not involved in any of the two transactions, the fact that the same money is being spent twice cannot be prevented.

WIRELESS APPLICATION PROTOCOL (WAP) SECURITY:

WAP Stack:



Security layer: The security layer in WAP stack is also called Wireless Transport Layer Security (WTLS) protocol. It provides:

(a) Privacy: Ensures that the messages passing between the client & the server are not accessible to anybody else.

(b) Server Authentication: It gives the client a confidence that the server is indeed what it is depicting as, and not someone who is posing as the server, with or without malicious intentions.

(c) Client Authentication: It gives the server a confidence that the client is indeed what it is depicting as.

(d) Data Integrity: It ensures that no one can tamper with the messages going between the client & the server, by modifying the contents in any manner.

GSM SECURITY:

There are 3 aspects in GSM Security:

(i) Subscriber Identity Authentication

(ii) Signalling Data Confidentiality

(iii) User data Confidentiality

Achieving Security in GSM:

(a) Authentication: The process begins with a challenge-response mechanism. The network sends a 128-bit random number to the subscriber when authentication begins. After this, 32-bit signed response using the authentication algorithm (A3) & the subscriber authentication key (Ki) is prepared by the handset, and sent back to the network. The network retrieves its value of Ki from its database, performs the same operation using the A3 algorithm on the original 128-bit random number, and compares this result with the one received from the handset. If the two match, the user is considered as successfully authenticated.

(b) Signalling & Data Confidentiality: The SIM contains the ciphering key generation algorithm(A8). This is used to produce the 64-bit ciphering key (Kc).

(c) Voice & Data Security: The A5 algorithm is used to encrypt the voice & data traffic between the user’s handset & the GSM network.

USER AUTHENTICATION MECHANISMS

INTRODUCTION:

One of the key aspects of cryptography & network/Internet security is authentication. Authentication ensures that the claimant is really what he/she claims to be.

AUTHENTICATION BASICS: It is the way of determining an identity to the required level of assurance.

TECHNIQUES FOR AUTHENTICATION:

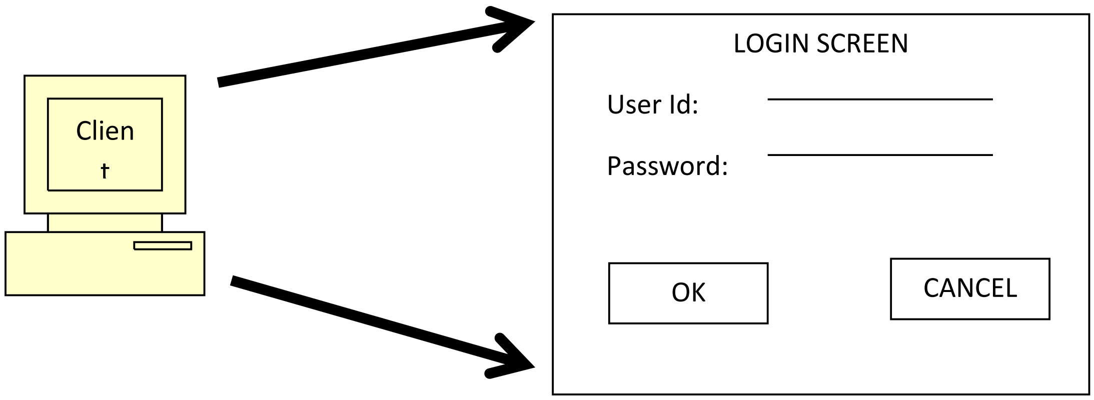
PASSWORDS: These are the most common form of authentication.

Def: A password is a string of alphabets, numbers and special characters, which is supposed to be known only to the entity (usually a person) that is being authenticated.

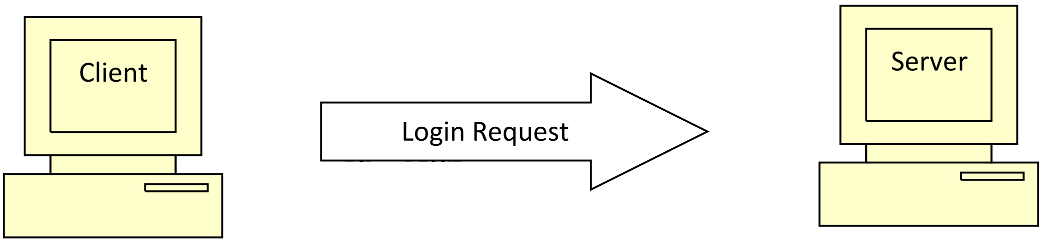
(a) Clear Text Passwords: Every user in the system is assigned a user id and an initial password. The user changes the password periodically for security reasons. The password is stored in clear text in the user database against the user id on the server.

Mechanism:

Step-1: Prompt for user id and password

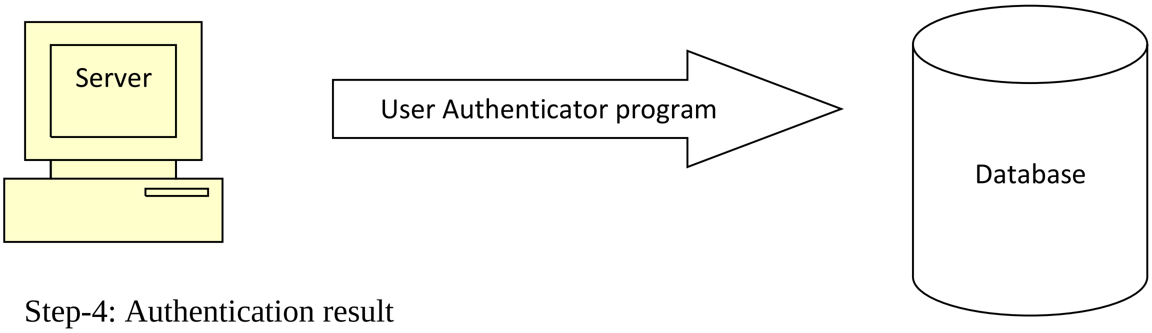


Step-2: User enters user id and password

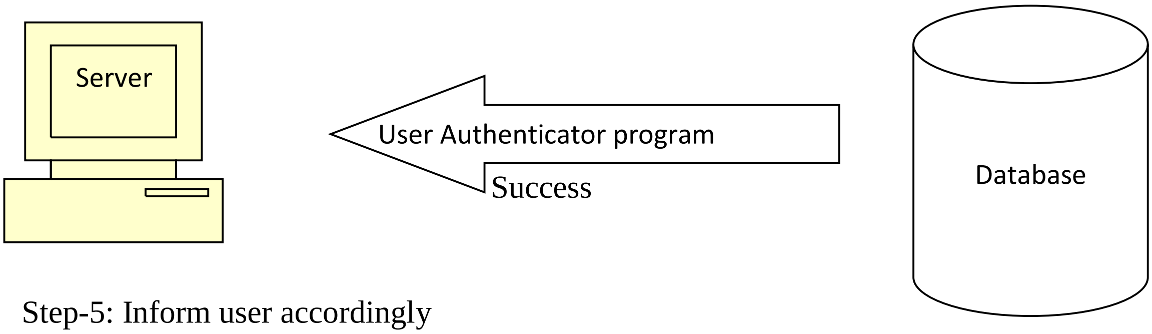


Id=user1
Password = april

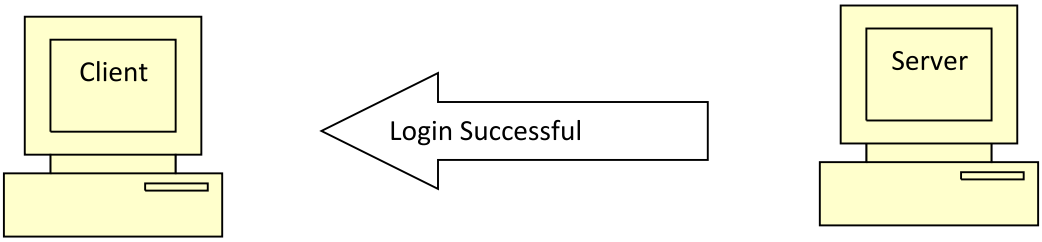
Step-3: User id and password validation



Step-4: Authentication result



Step-5: Inform user accordingly



Problems with this approach:

Problem-1: Database contains passwords in clear text.

If an attacker succeeds in obtaining an access to the database, the whole list of user ids and passwords is available to the attacker.

Problem-2: Password travels in clear text from the user's computer to the server

If the attacker breaks into the communication link between the user's computer and the server, the attacker can easily obtain the clear text password.

(ii) Something derived from passwords:

Instead of storing the password as it is, or in encrypted format, it can be stored as an output of an algorithm implemented on it. When the user wants to get authenticated, he enters the password and his computer performs the same algorithm locally, and sends the derived password to the server, where it is verified.

(a) Message Digests of passwords:

Step-1: Storing message digests as derived passwords in the user database: The MDs are stored as derived passwords in the user DB.

Step-2: User Authentication: when a user wants to b authenticated, he enters the user-id and password as usual. The user's computer computes the message digest of the password & sends the user id & message digest of the password to the server for authentication.

Step-3: Server-side validation: The user-id and the message digest of the password travel to the server over the communication link. The server passes these values to the user authenticator program which validates the user-id and the message digest of the password against the database and returns an appropriate response back to the server. The server uses the result of this operation to return an appropriate message back to the user.

(b) Adding randomness: This method ensures that although the message digest of the password is always the same, the exchange of information between the user's computer & the server is never the same.

Step-1: Storing message digests as derived passwords in the user database: The message digests are stored as derived passwords in the user DB.

Step-2: User sends a login request: the user sends the login request only with his user id.

Step-3: Server creates a random challenge: When the server receives a user's login request containing the user-id alone, it first checks to see if the user id is a valid one. If it is not, it sends an appropriate error message back to the user. If the user is valid, the server now creates a random challenge & sends it back to the user.

Step-4: User signs the random challenge with the message digest of the password: The application displays the password entry screen to the user. In response, the user enters the password. The application executes the appropriate message digest algorithm on the user's computer to create a message digest of the password entered by the user. This message digest of the password is now used to encrypt the random challenge received from the server.

Step-5: Server verifies the encrypted random challenge received from the user: The server receives the random challenge. In order to verify that the random challenge was indeed encrypted by the message digest by the user's password, the server must perform an identical operation, which can be achieved in 2 ways:

(i) The server can decrypt the encrypted random challenge received from the user with the message digest of the user's password.

(ii) The server can simply encrypt its own version of random challenge with the message digest of the user's password. If this encryption produces an encrypted random challenge, which matches the random challenge received from the user, the server can be assured that the random challenge was indeed encrypted by the message digest of the user's password.

Step-6: Server returns an appropriate message back to the user: Finally the sever sends an appropriate message back to the user depending on the previous operations yielded success or failure.

(c) Password encryption:

This method defines the encryption of passwords & then sending it to the server for authentication.

(i) Encrypt the password before it is stored in the user's computer.

(ii) Encrypt the password before it is sent to the server.

AUTHENTICATION TOKENS:

An authentication token is an extremely useful alternative to a password. An authentication token is a small device that generates a new random value every time it is used.

Step-1: Creation of token:

Whenever an authentication token is created, the corresponding random seed is generated for the token by the authentication server. This seed is stored or pre-programmed inside the token.

Step-2: Use of token:

An authentication token automatically generates pseudorandom numbers called as one time passwords (OTP). The user id and password travels to the server as part of the login request. The server obtains the seed corresponding to the user id from the user database using a seed retrieval program. It then calls another Password validation program to which the server gives the seed & the OTP.

Step-3: Server returns an appropriate message back to the user:

Finally the server sends an appropriate message back to the user, depending on whether the previous operations yielded success or failure.

CERTIFICATE BASED AUTHENTICATION:

It is based on the digital certificate of a user specified by FIPS-196 standard. It is stronger than password-based authentication as the user is expected to have certificate & not know password.

WORKING

Step-1: Creation, storage and distribution of digital certificates

Here, the digital certificates are created by the CA for each user and the certificates are sent to the respective users.

Step-2: Login Request

During a login request, the user sends only her user id to the server.

Step-3: Server creates a random challenge

When the server receives the user's login request containing the user id only, it 1st checks to see if the user id is a valid one. If it is not, it sends appropriate error message else it creates a random challenge & sends it back to the user. The random challenge can travel as PT from the server to the user's computer.

Step-4: User signs the random challenge

The user has to now sign the random challenge with her private key. User enters the secret password to open up the private key file. After the user enters the correct password, the user's private key file is opened by the application. It retrieves the private key from that file & uses it to encrypt the random challenge received from the server to create the user's digital signature. The server now needs to verify the user's signature. For this purpose, the server consults the user DB to obtain the user's public key. It then uses this public key to decrypt the signed random challenge received from the user. After this, it compares this decrypted random challenge with its original random challenge.

Step-5: Server returns an appropriate message back to the user

Finally, the server sends an appropriate message back to the user, depending on whether the previous operations yielded success or failure.

BIOMETRIC AUTHENTICATION:

A biometric device works on the basis of some human characteristics such as fingerprint, voice or pattern of lines in the iris of the eye. The user DB contains a sample of user's biometric characteristics. During authentication, the user is required to provide another sample of the user's biometric characteristics. This is matched with the one in the DB & if the two samples are the same, the user is considered to be a valid one.

KERBEROS:

Many real-life systems use an authentication protocol called as Kerberos.

Working:

There are 4 parties involved in the Kerberos protocol:

(a) Alice: The client workstation

(b) Authentication Server (AS): Verifies the user during login.

(c) Ticket Granting Server (TGS): Issues tickets to certify proof of identity.

(d) Bob: The server offering services such as network printing, file sharing or an application program.

Step-1: Login

Alice, the user sits down at an arbitrary workstation & enters her name. The workstation sends her name in plain text to the AS. In response, the AS performs several actions. It first creates a package of the user name & a randomly generated session key (KS). It encrypts this package with the symmetric key the AS shares with the TGS. The output of this step is called as Ticket Granting Ticket (TGT). The AS then combines the TGT with KS & encrypts the two together using a symmetric key derived from the password of Alice (KA). After this message is received, Alice's workstation asks her for the password. When Alice enters it, the workstation generates the symmetric key (KA) derived from the password & uses that key to extract the session key (KS) & the TGT. The workstation destroys the password of Alice from its memory immediately, to prevent an attacker from stealing it.

Step-2: Obtaining a Service Granting Ticket (SGT)

Now after a successful login, Alice wants to make use of Bob- the email server, for some e-mail communication. For this, Alice would inform her workstation that she needs to contact Bob. Therefore, Alice needs a ticket to communicate with Bob. At this juncture, Alice's workstation creates a message intended for the Ticket Granting Server which contains the following items:

(a) The TGT as in Step-1

(b) The id of the server (Bob) whose services Alice is interested in

(c) The current timestamp, encrypted with the same session key (KS).

Once the TGS is satisfied of the credentials of Alice, the TGS creates a session key KAB, for Alice to have secure communication with Bob. TGS sends it twice to Alice: once combined with Bob's id (Bob) and encrypted with the session key (KS), and a second time, combined with Alice's id and encrypted with Bob's secret key (KB).

Step-3: User contacts Bob for accessing the server

Alice can now send KAB to Bob in order to enter into a session with him. Since this exchange is also desired to be secure, Alice can simply forward KAB encrypted with Bob's secret key to Bob.

NETWORK SECURITY

TCP/IP:

Internet is based on the Transmission Control Protocol/ Internet Protocol (TCP/IP) protocol suite. TCP/IP contains 5 main layers:

(a) Application
(b) Transport
(c) Network or Internet
(d) Data Link
(e) Physical

TCP Segment:

(a) Source port number: This 2-byte number signifies the port number of the source computer, corresponding to the application that is sending this TCP segment.

(b) Destination port number: This 2-byte number signifies the port number of the destination computer, corresponding to the application that is expected to receive the TCP segment.

(c) Sequence number: This 4-byte field determines the number assigned to the 1st byte of the data portion contained in the TCP segment.

(d) Acknowledgement Number: If the destination host receives a segment with sequence number X correctly, it sends X+1 as the acknowledgement number back to the source. Thus, this 4-byte number defines the sequence number that the source is expecting from the destination as a receipt of the correct delivery.

(e) Header Length: This 4-bit field specifies the number of 4-byte words in the TCP header.

(f) Reserved: This 6-byte field is received for future use & is currently unused.

(g) Flag: This 6-bit field defines 6 different control flags, each one of them occupying one bit.

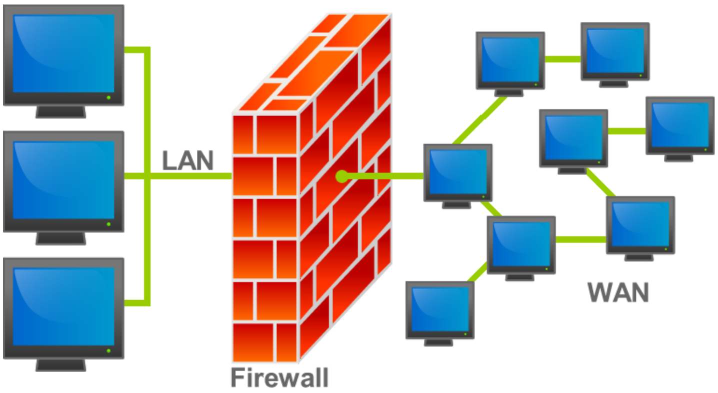
(h) Window size: This field determines the size of the sliding window that the other party must maintain.

(i) Checksum: This 16-bit field contains the checksum for facilitating the error detection and correction.

(j) Urgent pointer: This field is used in situations where data in a TCP segment is more important or urgent than the other data in the same TCP connection.

FIREWALLS

In computing, a firewall is a network security system that controls the incoming and outgoing network traffic based on an applied rule set. A firewall typically establishes a barrier between a trusted, secure internal network and another network (e.g., the Internet) that is assumed not to be secure and trusted.



Types of Firewall

There are different types of firewalls depending on where the communication is taking place, where the communication is intercepted and the state that is being traced.

(a) Network layer or packet filters

(i) Network layer firewalls, also called packet filters, operate at a relatively low level of the TCP/IP protocol stack, not allowing packets to pass through the firewall unless they match the established rule set. The firewall administrator may define the rules; or default rules may apply. The term "packet filter" originated in the context of BSD operating systems.

(ii) Network layer firewalls generally fall into two sub-categories, stateful and stateless. Stateful firewalls maintain context about active sessions, and use that "state information" to speed packet processing. Any existing network connection can be described by several properties, including source and destination IP address, UDP or TCP ports, and the current stage of the connection's lifetime (including session initiation, handshaking, data transfer, or completion connection). If a packet does not match an existing connection, it will be evaluated according to the ruleset for new connections. If a packet matches an existing connection based on comparison with the firewall's state table, it will be allowed to pass without further processing.

(iii) Stateless firewalls require less memory, and can be faster for simple filters that require less time to filter than to look up a session. They may also be necessary for filtering stateless network protocols that have no concept of a session. However, they cannot make more complex decisions based on what stage communications between hosts have reached.

(iv) Newer firewalls can filter traffic based on many packet attributes like source IP address, source port, destination IP address or port, destination service like WWW or FTP. They can filter based on protocols, TTL values, net block of originator, of the source, and many other attributes.

(v) Commonly used packet filters on various versions of Unix are *IPFilter* (various), *ipfw* (FreeBSD/Mac OS X), *NPF* (NetBSD), *PF* (OpenBSD, and some other BSDs), *iptables*/*ipchains* (Linux).

(b) Application-layer

(i) Application-layer firewalls work on the application level of the TCP/IP stack (i.e., all browser traffic, or all telnet or ftp traffic), and may intercept all packets traveling to or from an application. They block other packets (usually dropping them without acknowledgment to the sender).

(ii) On inspecting all packets for improper content, firewalls can restrict or prevent outright the spread of networked computer worms and trojans. The additional inspection criteria can add extra latency to the forwarding of packets to their destination.

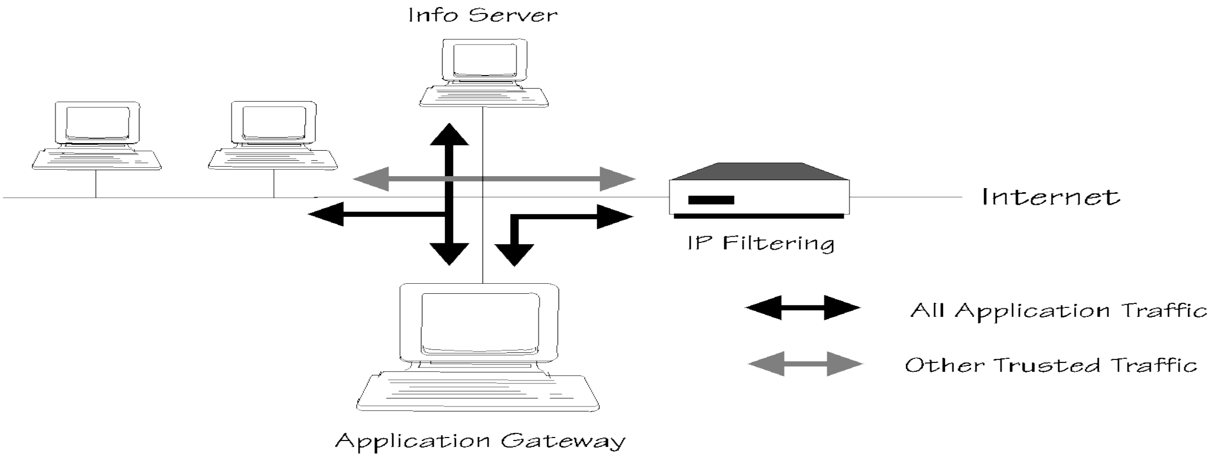
(iii) Application firewalls function by determining whether a process should accept any given connection. Application firewalls accomplish their function by hooking into socket calls to filter the connections between the application layer and the lower layers of the OSI model. Application firewalls that hook into socket calls are also referred to as socket filters. Application firewalls work much like a packet filter but application filters apply filtering rules (allow/block) on a per process basis instead of filtering connections on a per port basis. Generally, prompts are used to define rules for processes that have not yet received a connection. It is rare to find application firewalls not combined or used in conjunction with a packet filter.

(iv) Also, application firewalls further filter connections by examining the process ID of data packets against a ruleset for the local process involved in the data transmission. The extent of the filtering that occurs is defined by the provided ruleset. Given the variety of software that exists, application firewalls only have more complex rulesets for the standard services, such as sharing services. These per process rulesets have limited efficacy in filtering every possible association that may occur with other processes. Also, these per process rulesets cannot defend against modification of the process via exploitation, such as memory corruption exploits. Because of these limitations, application firewalls are beginning to be supplanted by a new generation of application firewalls that rely on mandatory access control (MAC), also referred to as sandboxing, to protect vulnerable services.

Firewall Configurations

(a) Screened Host Firewall, Single-homed bastion

The screened host firewall is a more flexible firewall than the dual-homed gateway firewall, however the flexibility is achieved with some cost to security. The screened host firewall is often appropriate for sites that need more flexibility than that provided by the dual-homed gateway firewall.

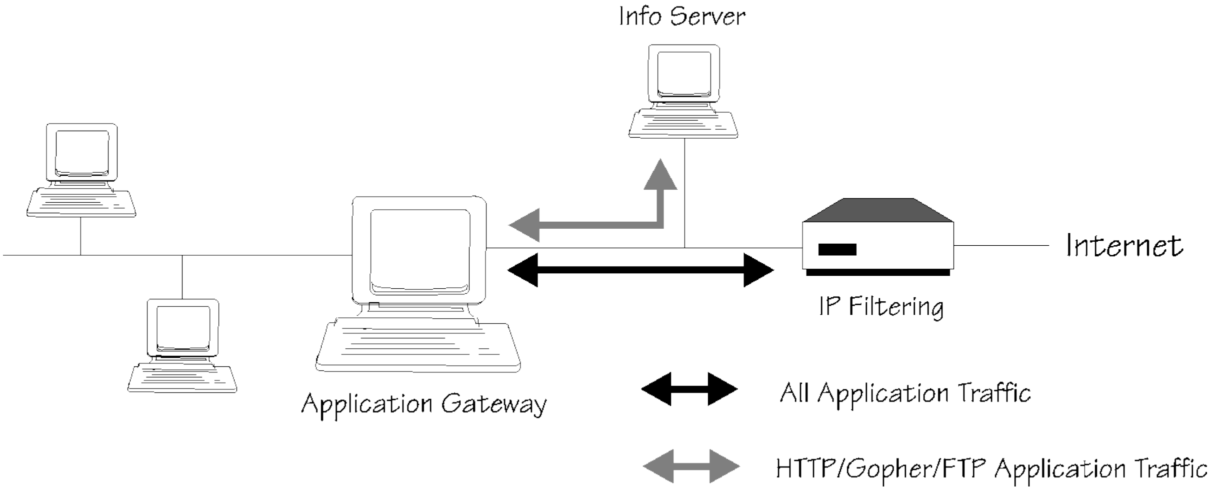


The screened host firewall combines a packet-filtering router with an application gateway located on the protected subnet side of the router. The application gateway needs only one network interface. The application gateway's proxy services would pass TELNET, FTP, and other services for which proxies exist, to site systems. The router filters or *screens* inherently dangerous protocols from reaching the application gateway and site systems. It rejects (or accepts) application traffic according to the following rules:

- application traffic from Internet sites to the application gateway gets routed,
- all other traffic from Internet sites gets rejected, and
- the router rejects any application traffic originating from the inside unless it came from the application gateway.

(b) Screened Host Firewall, Dual-homed bastion

The dual-homed gateway is a better alternative to packet filtering router firewalls. It consists of a host system with two network interfaces, and with the host's IP forwarding capability disabled (i.e., the default condition is that the host can no longer route packets between the two connected networks). In addition, a packet filtering router can be placed at the Internet connection to provide additional protection. This would create an inner, screened subnet that could be used for locating specialized systems such as information servers and modem pools. Unlike the packet filtering firewall, the dual-homed gateway is a complete block to IP traffic between the Internet and protected site. Services and access is provided by proxy servers on the gateway.



This type of firewall implements the second design policy, i.e., deny all services unless they are specifically permitted, since no services pass except those for which proxies exist. The ability of the host to accept source-routed packets would be disabled, so that no other packets could be passed by the host to the protected subnet. It can be used to achieve a high degree of privacy since routes to the protected subnet need to be known only to the firewall and not to Internet systems (because Internet systems cannot route packets directly to the protected systems). The names and IP addresses of site systems would be hidden from Internet systems, because the firewall would not pass DNS information.

A simple setup for a dual-homed gateway would be to provide proxy services for TELNET and FTP, and centralized e-mail service in which the firewall would accept all site mail and then forward it to site systems. Because it uses a host system, the firewall can house software to require users to use authentication tokens or other advanced authentication measures. The firewall can also log access and log attempts or probes to the system that might indicate intruder activity.

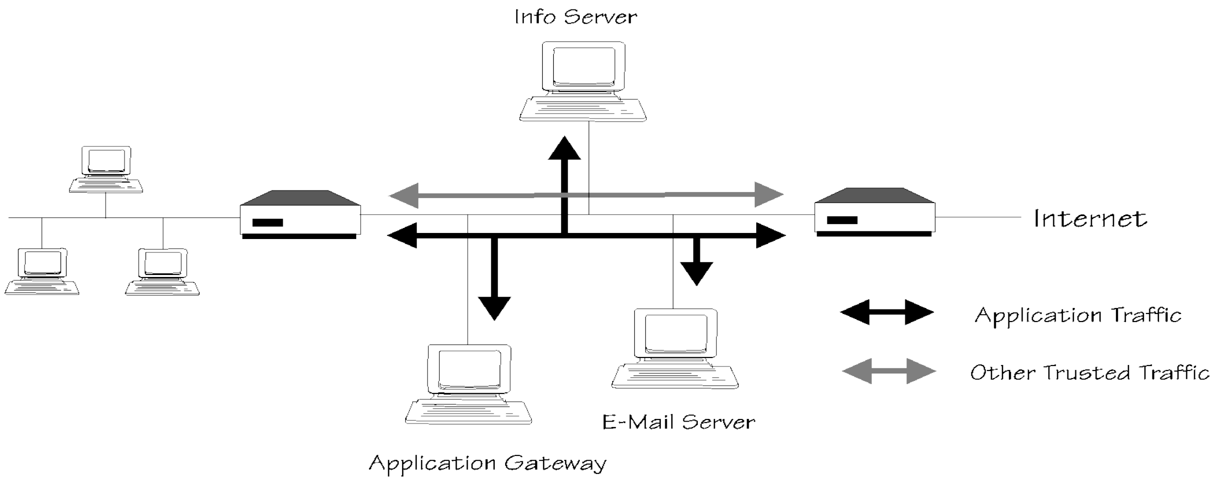
The dual-homed gateway firewall, as well as the screened subnet firewall mentioned later in this chapter, provides the ability to segregate traffic concerned with an information server from other traffic to and from the site. An information server could be located on the subnet between the gateway and the router, as shown in figure. Assuming that the gateway provides the appropriate proxy services for the information server (e.g., ftp, gopher, or http), the router can prevent direct Internet access to the firewall and force access to go through the firewall. If direct access is permitted to the server (which is the less secure alternative), then the server's name and IP address can be advertised by DNS. Locating the information server there also adds to the security of the site, as any intruder penetration of the information server would still be prevented from reaching site systems by the dual-homed gateway.

The inflexibility of the dual-homed gateway could be a disadvantage to some sites. Since all services are blocked except those for which proxies exist, access to other services cannot be opened up; systems that require the access would need to be placed on the Internet side of the gateway. However, a router could be used as shown in figure to create a subnet between the gateway and the router, and the systems that require extra services could be located there (this is discussed more in section with screened subnet firewalls).

Another important consideration is that the security of the host system used for the firewall must be very secure, as the use of any vulnerable services or techniques on the host could lead to break-ins. If the firewall is compromised, an intruder could potentially subvert the firewall and perform some activity such as to re-enable IP routing.

(c) Screened Subnet Firewall

The screened subnet firewall is a variation of the dual-homed gateway and screened host firewalls. It can be used to locate each component of the firewall on a separate system, thereby achieving greater throughput and flexibility, although at some cost to simplicity. But, each component system of the firewall needs to implement only a specific task, making the systems less complex to configure.



Here two routers are used to create an inner, *screened* subnet. This subnet (sometimes referred to in other literature as the ``DMZ") houses the application gateway, however it could also house information servers, modem pools, and other systems that require carefully-controlled access. The router shown as the connection point to the Internet would route traffic according to the following rules:

- application traffic from the application gateway to Internet systems gets routed,
- e-mail traffic from the e-mail server to Internet sites gets routed,
- application traffic from Internet sites to the application gateway gets routed,
- e-mail traffic from Internet sites to the e-mail server gets routed,
- ftp, gopher, etc., traffic from Internet sites to the information server gets routed, and
- all other traffic gets rejected.

The outer router restricts Internet access to specific systems on the screened subnet, and blocks all other traffic to the Internet originating from systems that should not be originating connections (such as the modem pool, the information server, and site systems). The router would be used as well to block packets such as NFS, NIS, or any other vulnerable protocols that do not need to pass to or from hosts on the screened subnet.

The inner router passes traffic to and from systems on the screened subnet according to the following rules:

- application traffic from the application gateway to site systems gets routed,
- e-mail traffic from the e-mail server to site systems gets routed,
- application traffic to the application gateway from site systems get routed,
- e-mail traffic from site systems to the e-mail server gets routed,
- ftp, gopher, etc., traffic from site systems to the information server gets routed,
- all other traffic gets rejected.
- Thus, no site system is directly reachable from the Internet and vice versa, as with the dual-homed gateway firewall. A big difference, though, is that the routers are used to direct traffic to specific systems, thereby eliminating the need for the application gateway to be dual-homed. Greater throughput can be achieved, then, if a router is used as the gateway to the protected subnet. Consequently, the screened subnet firewall may be more appropriate for sites with large amounts of traffic or sites that need very high-speed traffic.
- The two routers provide redundancy in that an attacker would have to subvert *both* routers to reach site systems directly. The application gateway, e-mail server, and information server could be set up such that they would be the only systems ``known" from the Internet; no other system name need be known or used in a DNS database that would be accessible to outside systems. The application gateway can house advanced authentication software to authenticate all inbound connections. It is, obviously, more involved to configure, however the use of separate systems for application gateways and packet filters keeps the configuration more simple and manageable.
- The screened subnet firewall, like the screened host firewall, can be made more flexible by permitting certain ``trusted" services to pass between the Internet and the site systems. However, this flexibility may open the door to exceptions to the policy, thus weakening the effect of the firewall. In many ways, the dual-homed gateway firewall is more desirable because the policy cannot be weakened (because the dual-homed gateway cannot pass services for which there is no proxy). However, where throughput and flexibility are important, the screened subnet firewall may be more preferable.

- As an alternative to passing services directly between the Internet and site systems, one could locate the systems that need these services directly on the screened subnet. For example, a site that does not permit X Windows or NFS traffic between Internet and site systems, but needs to anyway, could locate the systems that need the access on the screened subnet. The systems could still maintain access to site systems by connecting to the application gateway and reconfiguring the inner router as necessary. This is not a perfect solution, but an option for sites that require a high degree of security.
- There are two disadvantages to the screened subnet firewall. First, the firewall can be made to pass ``trusted'' services around the application gateway(s), thereby subverting the policy. This is true also with the screened host firewall, however the screened subnet firewall provides a location to house systems that need direct access to those services. With the screened host firewall, the ``trusted'' services that get passed around the application gateway end up being in contact with site systems. The second disadvantage is that more emphasis is placed on the routers for providing security. As noted, packet filtering routers are sometimes quite complex to configure and mistakes could open the entire site to security holes.

Firewall Limitations

(a) Insider's Intrusion: A firewall is designed to thwart outside attacks, hence it is ineffective to inside intrusions.
(b) Direct Internet Traffic: If the firewall is one of the entry-exit points, a user can bypass the firewall.
(c) Virus Attacks: A firewall can't be expected to scan every incoming file or packet for viruses.

IP SECURITY

Internet Protocol Security (IPsec) is a protocol suite for securing Internet Protocol (IP) communications by authenticating and encrypting each IP packet of a communication session. IPsec includes protocols for establishing mutual authentication between agents at the beginning of the session and negotiation of cryptographic keys to be used during the session. IPsec can be used in protecting data flows between a pair of hosts (*host-to-host*), between a pair of security gateways (*network-to-network*), or between a security gateway and a host (*network-to-host*).

Internet Protocol security (IPsec) uses cryptographic security services to protect communications over Internet Protocol (IP) networks. IPsec supports network-level peer authentication, data origin authentication, data integrity, data confidentiality (encryption), and replay protection.

IPsec is an end-to-end security scheme operating in the Internet Layer of the Internet Protocol Suite, while some other Internet security systems in widespread use, such as Transport Layer Security (TLS) and Secure Shell (SSH), operate in the upper layers at Application layer. Hence, only IPsec protects any application traffic over an IP network. Applications can be automatically secured by IPsec at the IP layer.

Security Architecture

The IPsec suite is an open standard. IPsec uses the following protocols to perform various functions.

- Authentication Headers (AH) provide connectionless integrity and data origin authentication for IP datagrams and provides protection against replay attacks.

- Encapsulating Security Payloads (ESP) provide confidentiality, data-origin authentication, connectionless integrity, an anti-replay service (a form of partial sequence integrity), and limited traffic-flow confidentiality.
- Security Associations (SA) provide the bundle of algorithms and data that provide the parameters necessary for AH and/or ESP operations. The Internet Security Association and Key Management Protocol (ISAKMP) provides a framework for authentication and key exchange, with actual authenticated keying material provided either by manual configuration with pre-shared keys, Internet Key Exchange (IKE and IKEv2), Kerberized Internet Negotiation of Keys (KINK), or IPSECKEY DNS records.

Authentication Header

Authentication Header (AH) is a member of the IPsec protocol suite. AH guarantees connectionless integrity and data origin authentication of IP packets. Further, it can optionally protect against replay attacks by using the sliding window technique and discarding old packets.

- In IPv4, the AH protects the IP payload and all header fields of an IP datagram except for mutable fields (i.e. those that might be altered in transit), and also IP options such as the IP Security Option (RFC-1108). Mutable (and therefore unauthenticated) IPv4 header fields are DSCP/ToS, ECN, Flags, Fragment Offset, TTL and Header Checksum.
- In IPv6, the AH protects most of the IPv6 base header, AH itself, non-mutable extension headers after the AH, and the IP payload. Protection for the IPv6 header excludes the mutable fields: DSCP, ECN, Flow Label, and Hop Limit.

Encapsulating Security Payload

Encapsulating Security Payload (ESP) is a member of the IPsec protocol suite. In IPsec it provides origin authenticity, integrity and confidentiality protection of packets. ESP also supports encryption-only and authentication-only configurations, but using encryption without authentication is strongly discouraged because it is insecure. Unlike Authentication Header (AH), ESP in transport mode does not provide integrity and authentication for the entire IP packet. However, in Tunnel Mode, where the entire original IP packet is encapsulated with a new packet header added, ESP protection is afforded to the whole inner IP packet (including the inner header) while the outer header (including any outer IPv4 options or IPv6 extension headers) remains unprotected. ESP operates directly on top of IP, using IP protocol number 50.

Security association

The IP security architecture uses the concept of a security association as the basis for building security functions into IP. A security association is simply the bundle of algorithms and parameters (such as keys) that is being used to encrypt and authenticate a particular flow in one direction. Therefore, in normal bi-directional traffic, the flows are secured by a pair of security associations.

Security associations are established using the Internet Security Association and Key Management Protocol (ISAKMP). ISAKMP is implemented by manual configuration with pre-shared secrets, Internet Key Exchange (IKE and IKEv2), Kerberized Internet Negotiation of

Keys (KINK), and the use of IPSECKEY DNS records. RFC 5386 defines Better-Than-Nothing Security (BTNS) as an unauthenticated mode of IPsec using an extended IKE protocol.

In order to decide what protection is to be provided for an outgoing packet, IPsec uses the Security Parameter Index (SPI), an index to the security association database (SADB), along with the destination address in a packet header, which together uniquely identify a security association for that packet. A similar procedure is performed for an incoming packet, where IPsec gathers decryption and verification keys from the security association database.

For multicast, a security association is provided for the group, and is duplicated across all authorized receivers of the group. There may be more than one security association for a group, using different SPIs, thereby allowing multiple levels and sets of security within a group. Indeed, each sender can have multiple security associations, allowing authentication, since a receiver can only know that someone knowing the keys sent the data. Note that the relevant standard does not describe how the association is chosen and duplicated across the group; it is assumed that a responsible party will have made the choice.

Modes of Operation:

IPsec can be implemented in a host-to-host transport mode, as well as in a network tunnelling mode.

(a) Transport Mode: In transport mode, only the payload of the IP packet is usually encrypted and/or authenticated. The routing is intact, since the IP header is neither modified nor encrypted; however, when the authentication header is used, the IP addresses cannot be translated, as this always will invalidate the hash value. The transport and application layers are always secured by hash, so they cannot be modified in any way (for example by translating the port numbers). A means to encapsulate IPsec messages for NAT traversal has been defined by RFC documents describing the NAT-T mechanism.

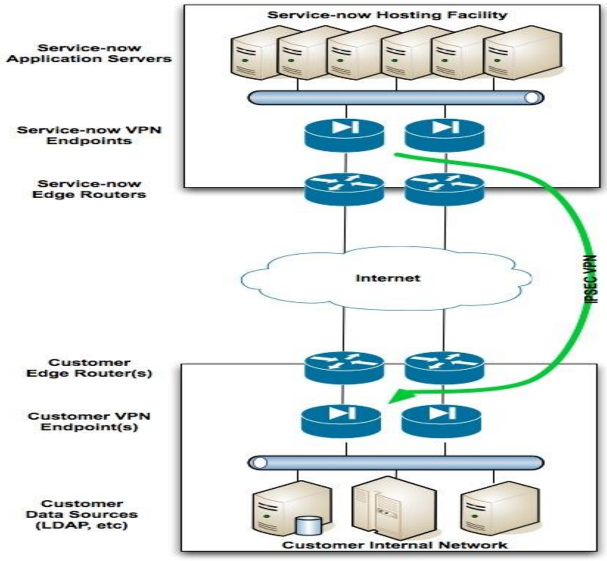
(b) Tunnel Mode : In tunnel mode, the entire IP packet is encrypted and/or authenticated. It is then encapsulated into a new IP packet with a new IP header. Tunnel mode is used to create virtual private networks for network-to-network communications (e.g. between routers to link sites), host-to-network communications (e.g. remote user access) and host-to-host communications (e.g. private chat). Tunnel mode supports NAT traversal.

VIRTUAL PRIVATE NETWORKS:

A virtual private network (VPN) extends a private network across a public network, such as the Internet. It enables a computer or network-enabled device to send and receive data across shared or public networks as if it were directly connected to the private network, while benefiting from the functionality, security and management policies of the private network.[1] A VPN is created by establishing a virtual point-to-point connection through the use of dedicated connections, virtual tunnelling protocols, or traffic encryption. Major implementations of VPNs include OpenVPN and IPsec.

A VPN connection across the Internet is similar to a wide area network (WAN) link between websites. From a user perspective, the extended network resources are accessed in the same way as resources available within the private network.[2] One major limitation of traditional VPNs is that they are point-to-point, and do not tend to support or connect broadcast domains. Therefore communication, software, and networking, which are based on layer 2 and broadcast packets, such as NetBIOS used in Windows networking, may not be fully supported or work exactly as they would on a real LAN. Variants on VPN, such as Virtual Private LAN Service (VPLS), and layer 2 tunnelling protocols, are designed to overcome this limitation.

VPNs allow employees to securely access their company's intranet while traveling outside the office. Similarly, VPNs securely connect geographically separated offices of an organization, creating one cohesive network. VPN technology is also used by individual Internet users to secure their wireless transactions, to circumvent geo restrictions and censorship, and to connect to proxy servers for the purpose of protecting personal identity and location.


Service-now Hosting Facility
Service-now Application Servers
Service-now VPN Endpoints
Service-now Edge Routers
Internet
IPSEC VPN
Customer Edge Router(s)
Customer VPN Endpoint(s)
Customer Data Sources (LDAP, etc)
Customer Internal Network

Types of VPN

Early data networks allowed VPN-style remote connectivity through dial-up modems or through leased line connections utilizing Frame Relay and Asynchronous Transfer Mode (ATM) virtual circuits, provisioned through a network owned and operated by telecommunication carriers. These networks are not considered true VPNs because they passively secure the data being transmitted by the creation of logical data streams. They have been replaced by VPNs based on IP and IP/Multiprotocol Label Switching (MPLS) Networks, due to significant cost-reductions and increased bandwidth provided by new technologies such as Digital Subscriber Line (DSL) and fibre-optic networks.

VPNs can be either remote-access (connecting a computer to a network) or site-to-site (connecting two networks). In a corporate setting, remote-access VPNs allow employees to access their company's intranet from home or while traveling outside the office, and site-to-site VPNs allow employees in geographically disparate offices to share one cohesive virtual network. A VPN can also be used to interconnect two similar networks over a dissimilar middle network; for example, two IPv6 networks over an IPv4 network.[

VPN systems may be classified by:

- The protocols used to tunnel the traffic
- The tunnel's termination point location, e.g., on the customer edge or network-provider edge
- Whether they offer site-to-site or network-to-network connectivity
- The levels of security provided

- The OSI layer they present to the connecting network, such as Layer 2 circuits or Layer 3 network connectivity

Security Mechanisms

VPNs cannot make online connections completely anonymous, but they can usually increase privacy and security. To prevent disclosure of private information, VPNs typically allow only authenticated remote access and make use of encryption techniques.

VPNs provide security by the use of tunnelling protocols and often through procedures such as encryption. The VPN security model provides:

- Confidentiality such that even if the network traffic is sniffed at the packet level (see network sniffer and Deep packet inspection), an attacker would only see encrypted data
- Sender authentication to prevent unauthorized users from accessing the VPN
- Message integrity to detect any instances of tampering with transmitted messages

Secure VPN protocols:

- Internet Protocol Security (IPsec) as initially developed by the Internet Engineering Task Force (IETF) for IPv6, which was required in all standards-compliant implementations of IPv6 before RFC 6434 made it only a recommendation. This standards-based security protocol is also widely used with IPv4 and the Layer 2 Tunnelling Protocol. Its design meets most security goals: authentication, integrity, and confidentiality. IPsec uses encryption, encapsulating an IP packet inside an IPsec packet. De-encapsulation happens at the end of the tunnel, where the original IP packet is decrypted and forwarded to its intended destination.
- Transport Layer Security (SSL/TLS) can tunnel an entire network's traffic (as it does in the OpenVPN project and SoftEther VPN project) or secure an individual connection. A number of vendors provide remote-access VPN capabilities through SSL. An SSL VPN can connect from locations where IPsec runs into trouble with Network Address Translation and firewall rules.
- Datagram Transport Layer Security (DTLS) - used in Cisco AnyConnect VPN and in OpenConnect VPN to solve the issues SSL/TLS has with tunnelling over UDP.
- Microsoft Point-to-Point Encryption (MPPE) works with the Point-to-Point Tunnelling Protocol and in several compatible implementations on other platforms.
- Microsoft Secure Socket Tunnelling Protocol (SSTP) tunnels Point-to-Point Protocol (PPP) or Layer 2 Tunnelling Protocol traffic through an SSL 3.0 channel. (SSTP was introduced in Windows Server 2008 and in Windows Vista Service Pack 1.)
- Multi Path Virtual Private Network (MPVPN). Ragula Systems Development Company owns the registered trademark "MPVPN".
- Secure Shell (SSH) VPN - OpenSSH offers VPN tunnelling (distinct from port forwarding) to secure remote connections to a network or to inter-network links. OpenSSH server provides a limited number of concurrent tunnels. The VPN feature itself does not support personal authentication.