LECTURE NOTES

CRYPTOGRAPHY AND NETWORK SECURITY

Name Of The Programme	**B.Tech-CSE**
Name Of The Subject	**Cryptography and Network Security**
Regulations	**R-19**
Year and Semester	**4th Year, 1st Semester**
Faculty Name	**Mr.Siva Kumar Ronanki**

COLLEGE OF ENGINEERING
(AUTONOMOUS)

GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING (Autonomous)
Approved by AICTE, New Delhi and Affiliated to Andhra University
Re-accredited by NAAC with "A" Grade with a CGPA of 3.47/4.00
Madhurawada, Visakhapatnam - 530 048.

UNIT-I

COMPUTER AND NETWORK SECURITY CONCEPTS: Computer Security concepts, the OSI Security architecture, Security attacks, Security services, Security mechanisms, a model for Network Security.

CLASSICAL ENCRYPTION TECHNIQUES: Symmetric cipher model, Substitution techniques, Transposition techniques, Steganography.

BLOCK CIPHERS AND THE DATA ENCRYPTION STANDARD: Traditional block cipher structure, the Data Encryption Standard, the strength of DES, Block cipher design principles.

BLOCK CIPHER OPERATION: Multiple encryption and triple DES, Electronic codebook, Cipher block chaining mode.

RANDOM BIT GENERATION AND STREAM CIPHERS: Principles of pseudorandom number generation, pseudorandom number generators, Stream ciphers, RC4

WHAT IS NETWORK SECURITY ?

Network Security consists of the *provisions and policies* adapted by network Administrator to *prevent and monitor* unauthorized access, misuse, modification, or denial of a computer network and network-accessibleresources.

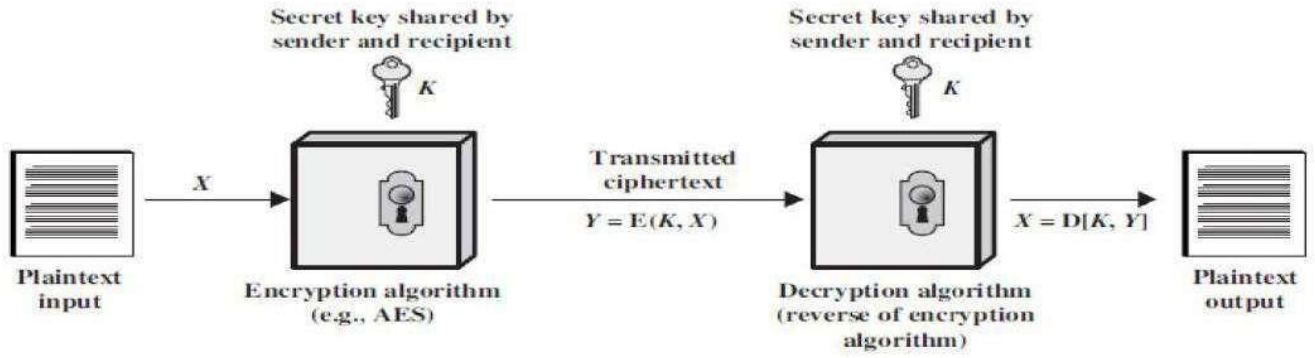
WHAT IS CRYPTOGRAPHY?

Cryptography is the *study of secure communications techniques* that allow only the sender and intended recipient of a message to view its contents.

The term is derived from the Greek word *kryptos*, which means hidden.

MODEL FOR NETWORK SECURITY - TERMINOLOGY

- Plaintext - the original message
- Cipher text - the coded message
- Cipher - algorithm for transforming plaintext to cipher text
- Key - info used in cipher known only to sender/receiver
- Encipher (Encrypt) - converting plaintext to cipher text
- Decipher (Decrypt) - recovering cipher text from plaintext
- Cryptography - study of encryption principles/methods
- Cryptanalysis (code breaking) - the study of principles/ methods of deciphering cipher text *without* knowing key
- Cryptology - the field of both cryptography and cryptanalysis



SECURITY GOALS

- Data Confidentiality
  - Keep data and communication secret
  - Privacy of personal financial/health records, etc.
  - Military and commercial relevance
- Data Integrity
  - Protect reliability of data against tampering
  - Can we be sure of the source and content of information?
- System Availability
  - Data/resources should be accessible when needed
  - Protection against denial of service attacks

Cryptographic Attacks

Accessing of data by unauthorized entity is called as attack

Passive Attacks
Active Attacks

Passive Attacks:

In a passive attack, the attacker's goal is just to obtain information. This means that the attack does not modify data or harm the system.

Active Attacks:

An active attack may change the data or harm the system. Attacks that threaten the integrity and availability are active attacks.

- Passive attacks
  - Interception
    - Release of message contents
    - Traffic analysis
- Active attacks
  - Interruption, modification, fabrication
    - Masquerade
    - Replay
    - Modification
    - Denial of service

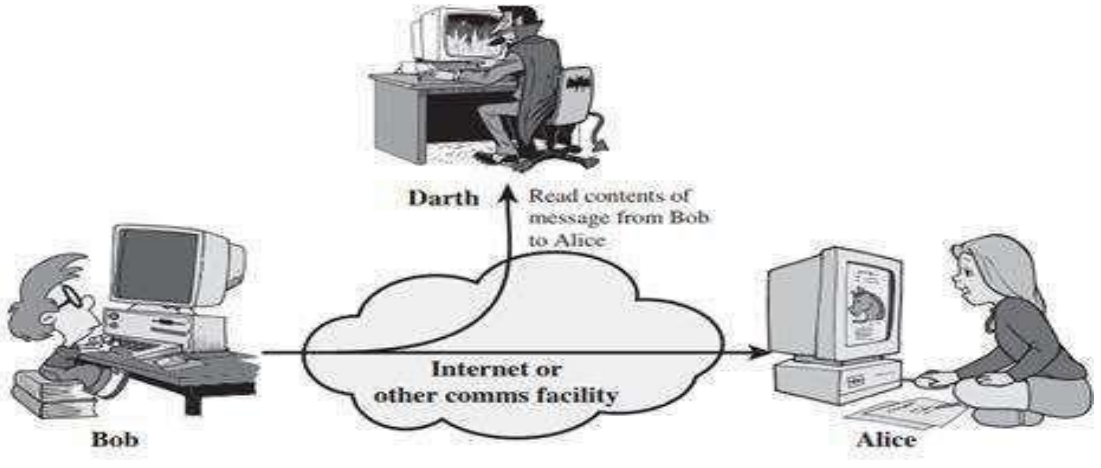
Passive Attacks

(a) Release of message content –

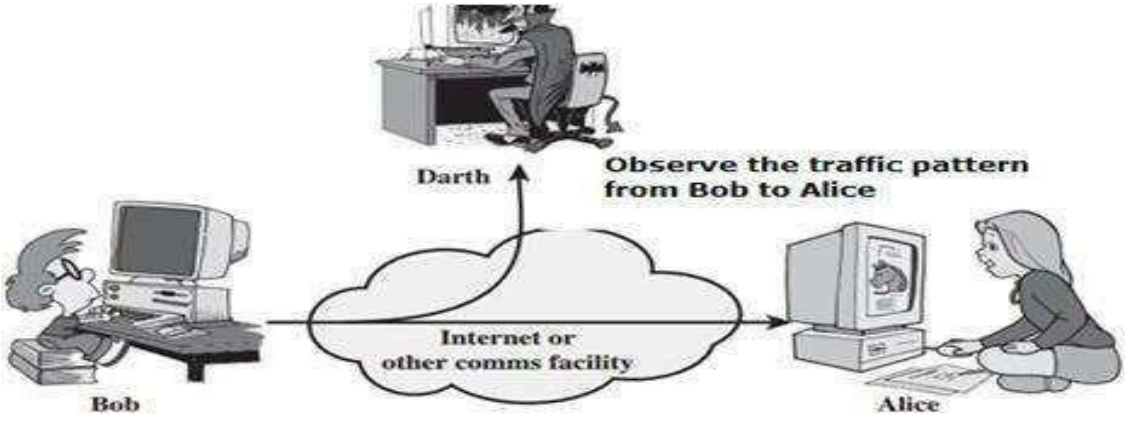
Capture and read the content transmissions.

(b) Traffic Analysis–

- can't read the information, but observe the pattern
- determine the location and identity of communicating parties
- observe frequency and length of communication



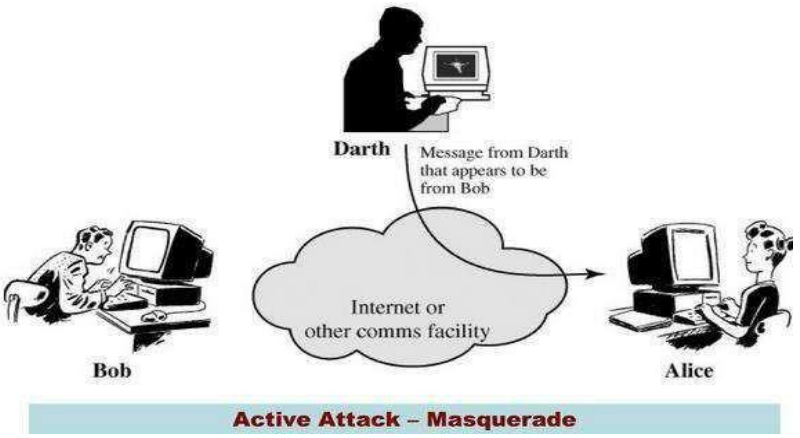
(a) Release of Message content



(b) Traffic Analysis

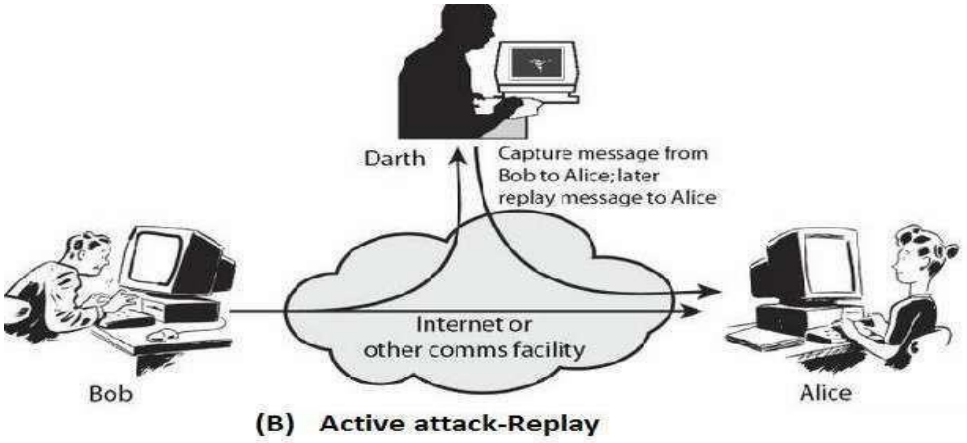
Active Attacks

(a) Masquerading: Masquerading or snooping happens when the attacker impersonates somebody else.



(a) Replay–

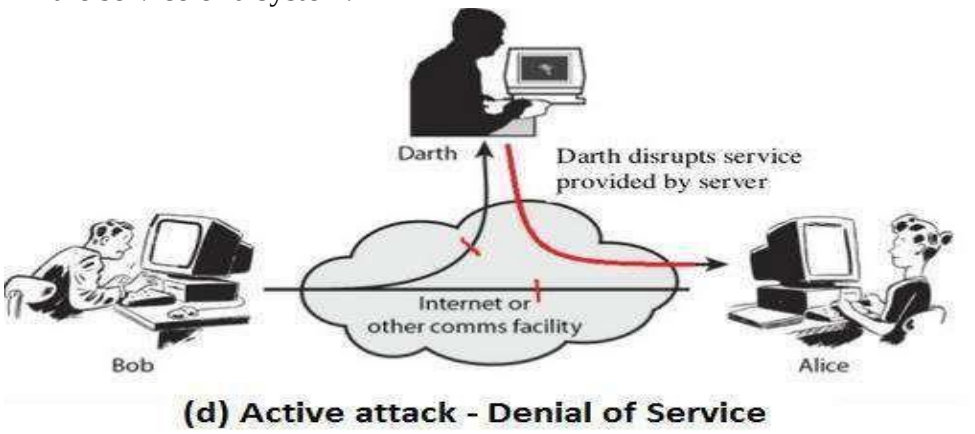
The attacker obtains a copy of a message sent by a user and later tries to replay it.



(a) Modification: After intercepting or accessing information, the attacker modifies the information then

send to receiver.

(a) Denial of service: Denial of service (Dos) is a very common attack.it may slow down or totally interrupt the service of a system.



Cryptographic Attacks Categories

Cryptographic attacks can be broadly categorized into two distinct types:

- Cryptanalytic
- Non-Cryptanalytic

Cryptanalytic Attacks:

- These attacks are combinations of statistical and algebraic techniques aimed at discover the secret key of a cipher.
- The attacker thus guesses the key and looks for the distinguishing property. If the property is detected, the guess is correct otherwise the next guess is tried.

Non-Cryptanalytic Attacks:

- The other types of attacks are non-cryptanalytic attacks, which do not explain the mathematical weakness of the cryptographic algorithm.

SERVICES AND MECHANISM

Security Services

ITU-T (X .800) is provided by protocol layer of transmission that defines security services ensures security of the data transfer

- Data Confidentiality: It is designed to protect data from disclosure attack.. That is, it is designed to prevent snooping and traffic analysis attack.
- Data Integrity: It is designed to protect data from modification, insertion, deletion and replaying by an adversary
- Authentication: It provides the authentication of the party at the other end of the line.
- Non-repudiation: It protects against repudiation by either the sender or the receiver of the data.
- Access Control: It provides protection against unauthorized access to data

Security Mechanism:

ITU-T recommends Security mechanisms to provide the security service

- Encipherment: The use of mathematical algorithms to transform data into a form that is not readily understandable
- Data Integrity:A variety of mechanisms used to assure the integrity of a data unit or stream of data units.
- Digital Signature:A digital signature is a means by which the sender can electronically sign the data and the receiver can electronically verify the signature.

- Authentication Exchange: A mechanism intended to ensure the identity of an entity by means of information exchange.
- Routing Control: Enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected.
- Traffic Padding: Inserting bogus data to prevent traffic analysis.
- Notarization: The use of a trusted third party to assure certain properties of a data exchange.
- Access Control: A variety of mechanisms that enforce access rights to resources.

Relation Security Services and Mechanisms

➢ Security Mechanism: A mechanism that is designed to detect, prevent, or recover from a security attack.

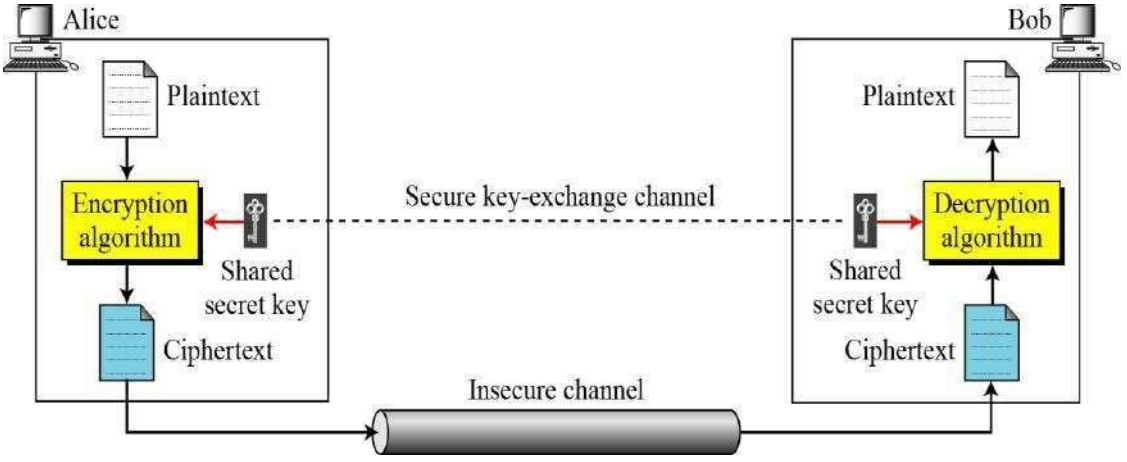
➢Security Service: A service that enhances the security of data processing systems and information transfers. A security service makes use of one or

more security mechanisms.

Services	Mechanisms
Confidentiality	Encryption, routing control
Integrity	Digital Signature, Encryption
Authentication	Encryption, Digital Signature
Non-repudiation	Digital Signature, Notarization
Access Control	Interactive Proofs, access control mechanisms and policies.

Symmetric Key Cipher

The sender and receiver of message use a single common key to encrypt and decrypt messages.



If P is the plaintext, C is the ciphertext, and K is the key,

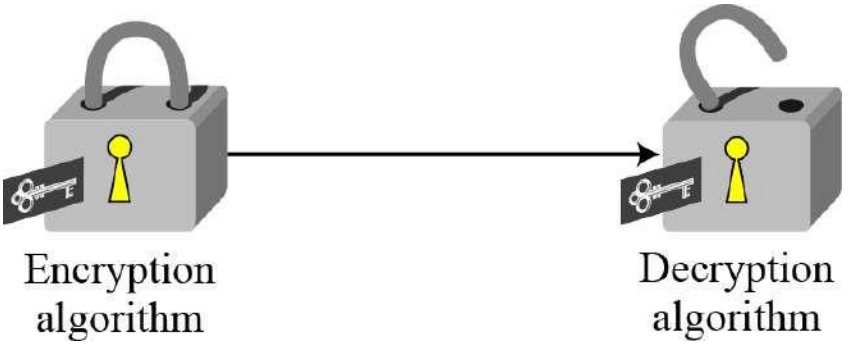
$$\text{Encryption: } C = E_k(P) \qquad \text{Decryption: } P = D_k(C)$$

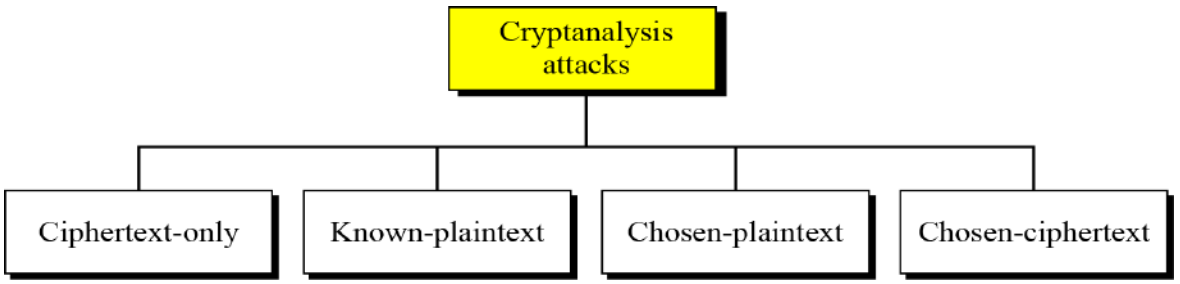
$$\text{In which, } D_k(E_k(x)) = E_k(D_k(x)) = x$$

We assume that Bob creates P_1; we prove that $P_1 = P$:

$$\textbf{Alice: } C = E_k(P) \qquad \textbf{Bob: } P_1 = D_k(C) = D_k(E_k(P)) = P$$

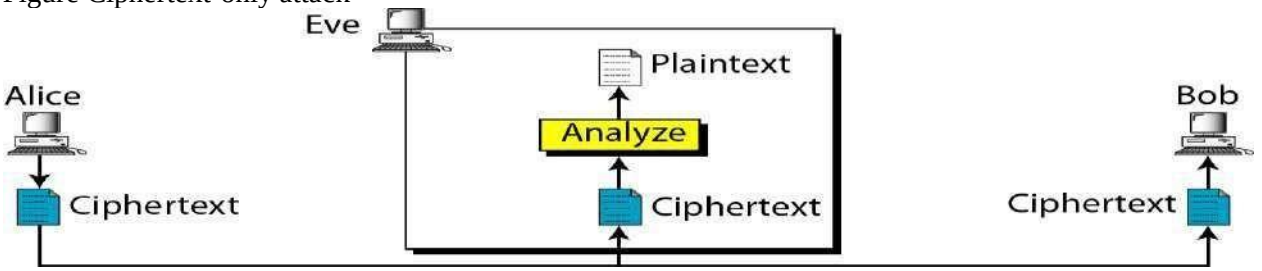
Figure Locking and unlocking with the same key





Ciphertext-Only Attack

Figure Ciphertext-only attack



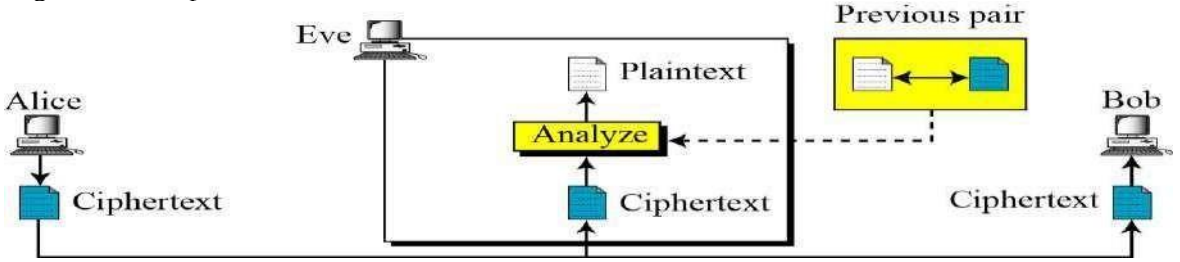
In Ciphertext-Only Attack , the attacker knows only some cipher text. He try to find corresponding key and plain text using various methods.

Brute-Force attack: Attacker tries all possible keys. We assume that he knows key domain

Statistical attack: The cryptanalyst can benefit from some inherent charactersistics of the plain text languageto perform statistical attack. Example: Letter E is most frequently used character in English.

Known-Plaintext Attack

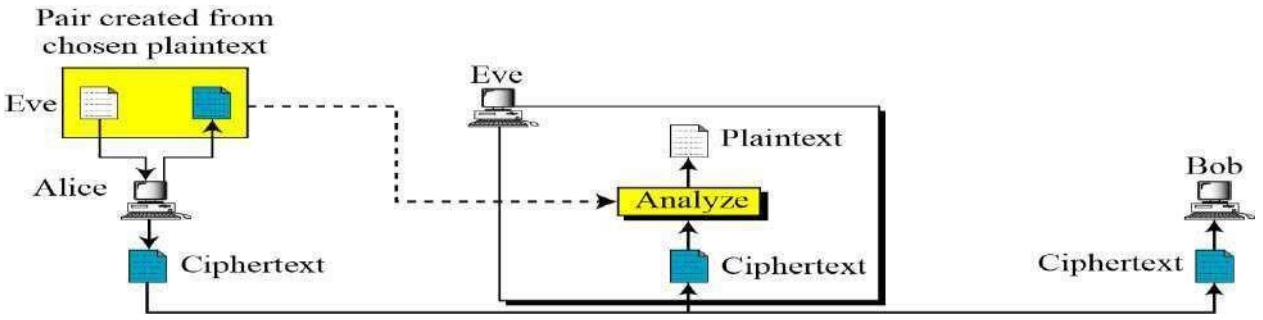
Figure Known-plaintext attack



In this attack, he know some cipher text and plain text pairs that were sent previously by Alice to Bob. Attacker haskept both cipher text and plain text to use them to break the next secrete message.

Chosen-Plaintext Attack

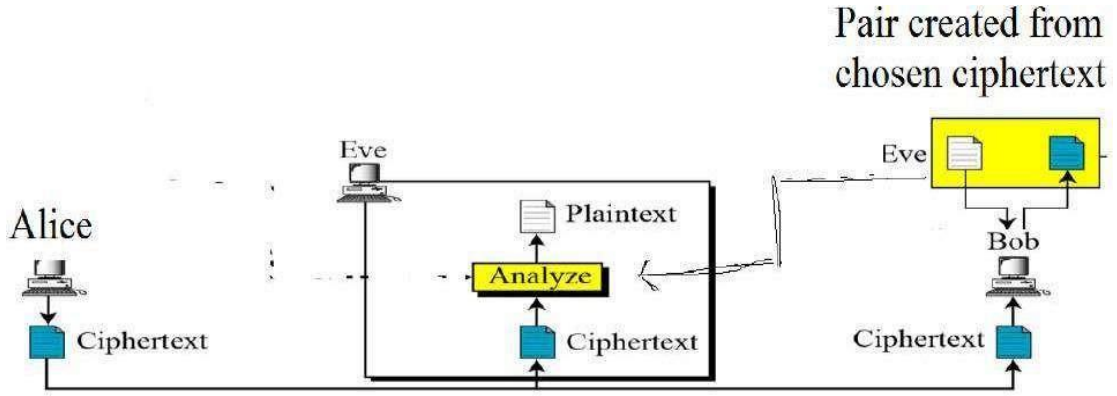
Figure Chosen-plaintext attack



This is similar to known-plaintext attack, but plaintext/cipher text pairs have been choosen by the attacker . This can happen when attacker has access to Alice computer. She can choose some plaintext and interpret ciphertext.

Chosen-Ciphertext Attack

Figure Chosen-Ciphertext attack



This is similar to Chosen Plaintext attack except eve chooses some ciphertext and decrypt it to from a cipher/plain text pairs. This can happen when Eve has access to Bob computer.

Categories of Traditional Ciphers

1. **SUBSTITUTION CIPHERS**
   A substitution cipher replaces one character with another
2. **TRANSPOSITION CIPHERS**
   A Transposition cipher reorders symbols

1. SUBSTITUTION CIPHERS

A substitution cipher replaces one symbol with another. Substitution ciphers can be categorized as either monoalphabetic ciphers or polyalphabetic ciphers.

Note:

A substitution cipher replaces one symbol with another.

Monoalphabetic Ciphers:

In monoalphabetic substitution, the relationship between a symbol in the plaintext to a symbol in the ciphertext is always one-to-one.

Example 1

The following shows a plaintext and its corresponding ciphertext. The cipher is probably monoalphabetic because both l's (else) are encrypted as O's.

Plaintext: hello **Ciphertext:** KHOOR

Example 2

The following shows a plaintext and its corresponding ciphertext. The cipher is not monoalphabeticbecause each l (el) is encrypted by a different character. The first l (el) is encrypted with N;the second as Z

Plaintext: hello **Ciphertext:** ABNZF

Additive Cipher

The simplest monoalphabetic cipher is the additive cipher. This cipher is sometimes called a shift cipher and sometimes a Caesar cipher, but the term additive cipher better reveals its mathematical nature.

Figure Plaintext and ciphertext in Z_{26}

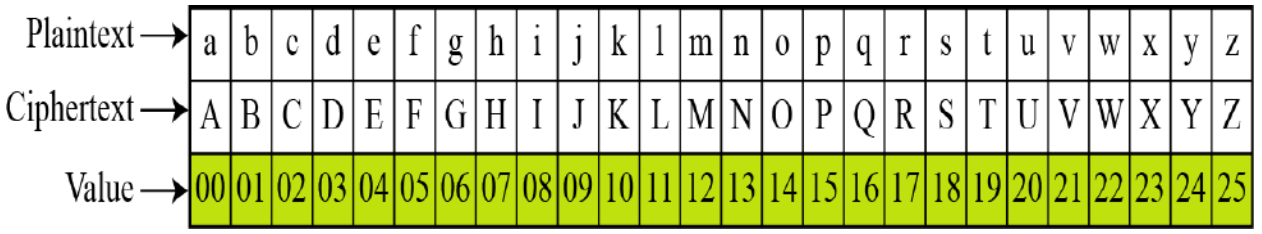
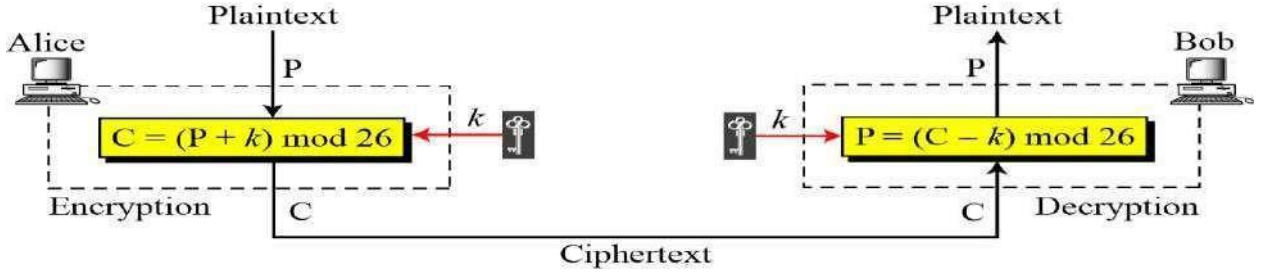


Figure Additive cipher



Note:

When the cipher is additive, the plaintext, ciphertext, and key are integers in Z_{26}.

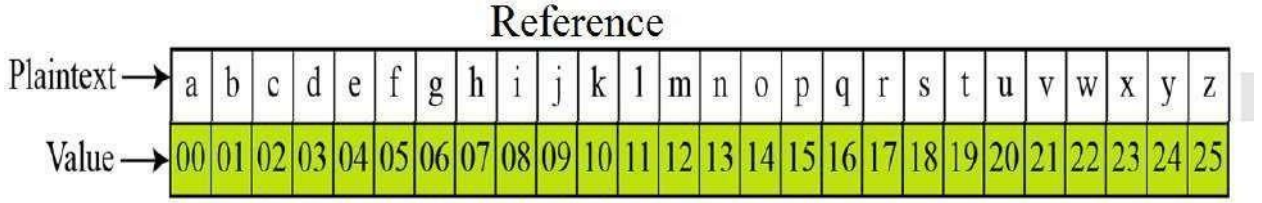
Example:

Use the additive cipher with key = 15 to encrypt the message “hello”.

Solution

We apply the encryption algorithm to the plaintext, character by character:

Plaintext: h → 07	Encryption: (07 + 15) mod 26	Ciphertext: 22 → W
Plaintext: e → 04	Encryption: (04 + 15) mod 26	Ciphertext: 19 → T
Plaintext: l → 11	Encryption: (11 + 15) mod 26	Ciphertext: 00 → A
Plaintext: l → 11	Encryption: (11 + 15) mod 26	Ciphertext: 00 → A
Plaintext: o → 14	Encryption: (14 + 15) mod 26	Ciphertext: 03 → D



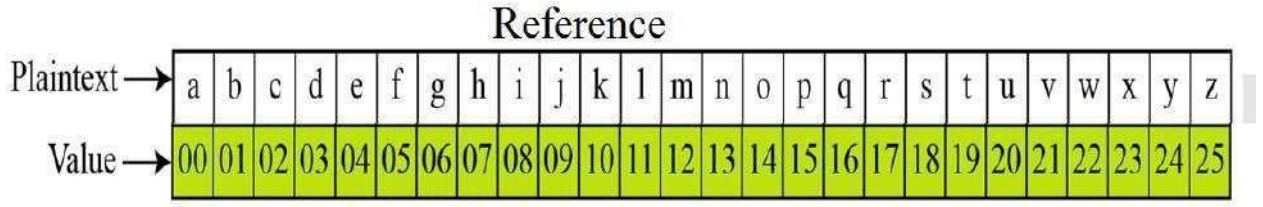
Example:

Use the additive cipher with key = 15 to decrypt the message “WTAAD”.

Solution

We apply the decryption algorithm to the plaintext character by character:

Ciphertext: W → 22	Decryption: (22 – 15) mod 26	Plaintext: 07 → h
Ciphertext: T → 19	Decryption: (19 – 15) mod 26	Plaintext: 04 → e
Ciphertext: A → 00	Decryption: (00 – 15) mod 26	Plaintext: 11 → l
Ciphertext: A → 00	Decryption: (00 – 15) mod 26	Plaintext: 11 → l
Ciphertext: D → 03	Decryption: (03 – 15) mod 26	Plaintext: 14 → o



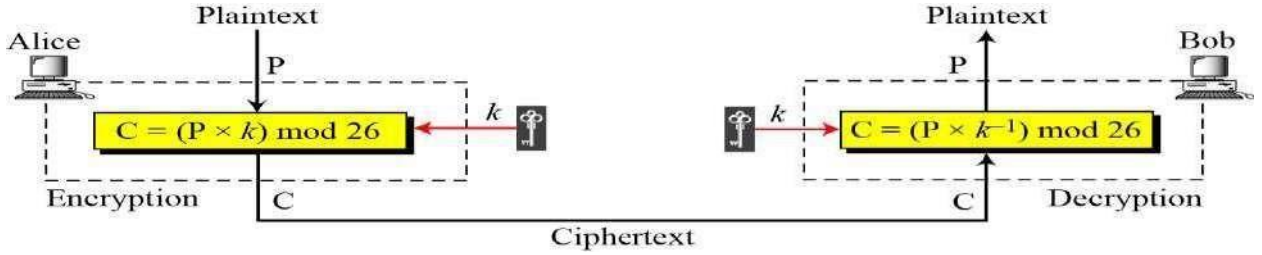
Shift Cipher and Caesar Cipher

Historically, additive ciphers are called shift ciphers. Julius Caesar used an additive cipher to communicate with his officers. For this reason, additive ciphers are sometimes referred to as the Caesar cipher. Caesar used a key of 3 for his communications.

Note:

Additive ciphers are sometimes referred to as shift ciphers or Caesar cipher

Multiplicative Ciphers



Note:

In a multiplicative cipher, the plaintext and ciphertext are integers in Z_{26}; the key is an integer in Z_{26}*.

Example1:

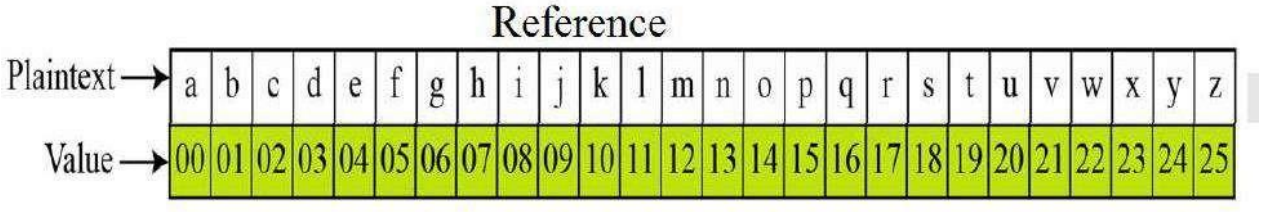
What is the key domain for any multiplicative cipher?

Solution: The key needs to be in Z_{26}*. This set has only 12 members: 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25.

Example2:

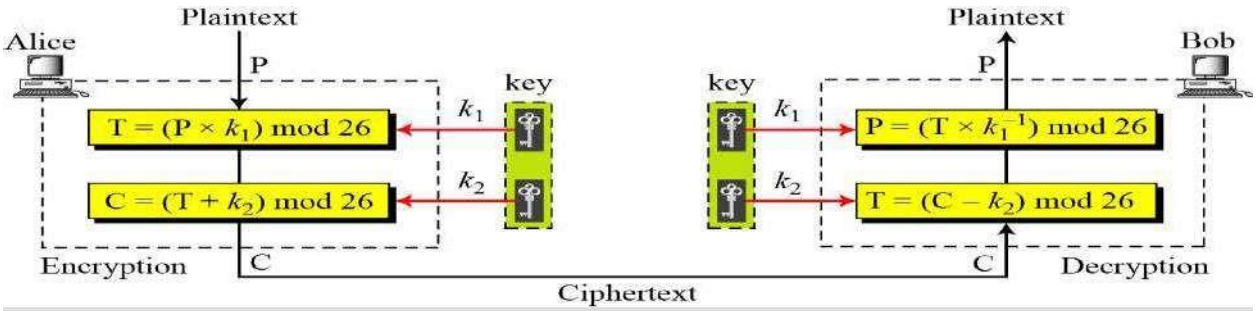
We use a multiplicative cipher to encrypt the message "hello" with a key of 7. The ciphertext is "XCZZU".

Plaintext: h → 07	Encryption: (07 × 07) mod 26	ciphertext: 23 → X
Plaintext: e → 04	Encryption: (04 × 07) mod 26	ciphertext: 02 → C
Plaintext: l → 11	Encryption: (11 × 07) mod 26	ciphertext: 25 → Z
Plaintext: l → 11	Encryption: (11 × 07) mod 26	ciphertext: 25 → Z
Plaintext: o → 14	Encryption: (14 × 07) mod 26	ciphertext: 20 → U



Affine Ciphers:

Combine additve and multiplicative Ciphers



$$C = (P \times k_1 + k_2) \bmod 26 \qquad P = ((C - k_2) \times k_1^{-1}) \bmod 26$$

where k_1^{-1} is the multiplicative inverse of k_1 and $-k_2$ is the additive inverse of k_2

Example1:

The affine cipher uses a pair of keys in which the first key is from Z_{26}* and the second is from Z_{26}.

The size of the key domain is 26 × 12 = 312.

Example2:

Use an affine cipher to encrypt the message "hello" with the key pair (7, 2).

P: h → 07	Encryption: (07 × 7 + 2) mod 26	C: 25 → Z
P: e → 04	Encryption: (04 × 7 + 2) mod 26	C: 04 → E
P: l → 11	Encryption: (11 × 7 + 2) mod 26	C: 01 → B
P: l → 11	Encryption: (11 × 7 + 2) mod 26	C: 01 → B
P: o → 14	Encryption: (14 × 7 + 2) mod 26	C: 22 → W

Monoalphabetic Substitution Cipher

Because additive, multiplicative, and affine ciphers have small key domains, they are very vulnerable tobrute-force attack.

A better solution is to create a mapping between each plaintext character and the corresponding ciphertextcharacter. Alice and Bob can agree on a table showing the mapping for each character.

Figure An example key for monoalphabetic substitution cipher

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

We can use the key in Figure to encrypt the message

this message is easy to encrypt but hard to find the key

The ciphertext is

ICFVQRVVNEFVRNVSIYRGAHSLIOJICNHTIYBFGTICRXRS

Reference

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

Polyalphabetic Ciphers

In polyalphabetic substitution, each occurrence of a character may have a different substitute. The relationship between a character in the plaintext to a character in the ciphertext is one-to-many. Example 'a' can be enciphered as 'D' in the beginning of the text, but as 'N' at the middle.

Polyalphabetic has advantage of hiding the letter frequency.

Example: Autokey Cipher

$P = P_1P_2P_3 \ldots$ $C = C_1C_2C_3 \ldots$ $k = (k_1, P_1, P_2, \ldots)$

Encryption: $C_i = (P_i + k_i) \bmod 26$ Decryption: $P_i = (C_i - k_i) \bmod 26$

Example:

Assume that Alice and Bob agreed to use an autokey cipher with initial key value k1 = 12. Now

Plaintext:	a	t	t	a	c	k	i	s	t	o	d	a	y
P's Values:	00	19	19	00	02	10	08	18	19	14	03	00	24
Key stream:	*12*	*00*	*19*	*19*	*00*	*02*	*10*	*08*	*18*	*19*	*14*	*03*	*00*
C's Values:	12	19	12	19	02	12	18	00	11	7	17	03	24
Ciphertext:	**M**	**T**	**M**	**T**	**C**	**M**	**S**	**A**	**L**	**H**	**R**	**D**	**Y**

Alice wants to send Bob the message "Attack is today". Enciphering is done character

TRANSPOSITION CIPHERS

A transposition cipher does not substitute one symbol for another, instead it changes the location of the symbols. A symbol in the first position may appaer in the tenth position of the cipher. A symbol in the eighth position may appear in the first osition of the cipher.

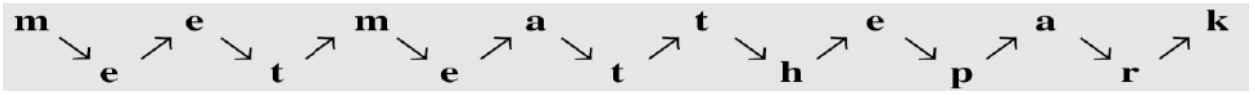
Note: A transposition cipher reorders symbols

Keyless Transposition Ciphers

Simple transposition ciphers, which were used in the past, are keyless.

Example 1:

A good example of a keyless cipher using the first method is the rail fence cipher. The ciphertext iscreated reading the pattern row by row. For example, to send the message " ***Meet me at the park***" to Bob, Alice writes



She then creates the ciphertext "MEMATEAKETETHPR**".**

Example 2:

Alice and Bob can agree on the number of columns and use the second method. Alice writes the same plaintext, row by row, in a table of four columns.

m	**e**	**e**	**t**
m	**e**	**a**	**t**
t	**h**	**e**	**p**
a	**r**	**k**	

She then creates the ciphertext "MMTAEEHREAEKTTP" by transmitting the characters column by column. Bob receives the cipher text and follows the reverse process to get plain text .

Example:

m	**e**	**e**	**t**
m	**e**	**a**	**t**
t	**h**	**e**	**p**
a	**r**	**k**	

The cipher in previous example is actually a transposition cipher. **The following shows the permutation of each character in the plaintext into the ciphertext based on the positions.**

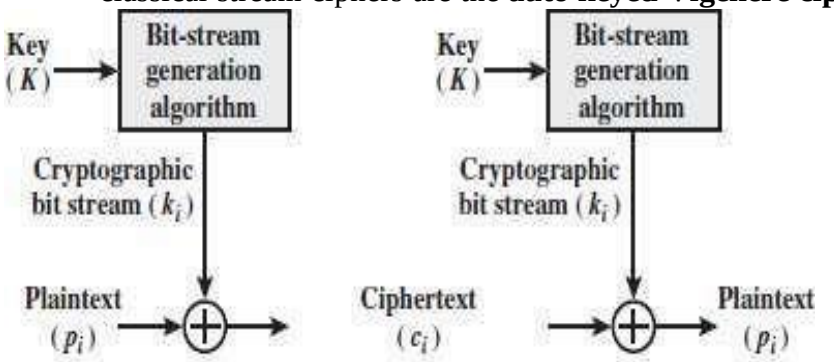
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
01	05	09	13	02	06	10	13	03	07	11	15	04	08	12

The second character in the plaintext has moved to the fifth position in the ciphertext; the third character hasmoved to the ninth position; and so on. Although the characters are permuted, there is a **pattern** in the permutation: (01, 05, 09, 13), (02, 06, 10, 13), (03, 07, 11, 15), and (4, 8, 12). In each section, the difference between the two adjacent numbers is 4.

Stream Ciphers and Block Ciphers

Stream Ciphers

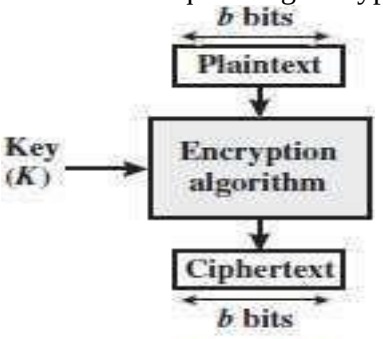
- A stream cipher is one that encrypts a digital data stream one bit or one byte at a time. Examples of classical stream ciphers are the **auto keyed Vigenère cipher and the Vernam cipher.**



(a) Stream cipher using algorithmic bit-stream generator

Block Ciphers

A **block cipher** is one in which a block of plaintext is treated as a whole and used to produce a cipher textblock of equal length. Typically, a block size of 64 or 128 bits is used.

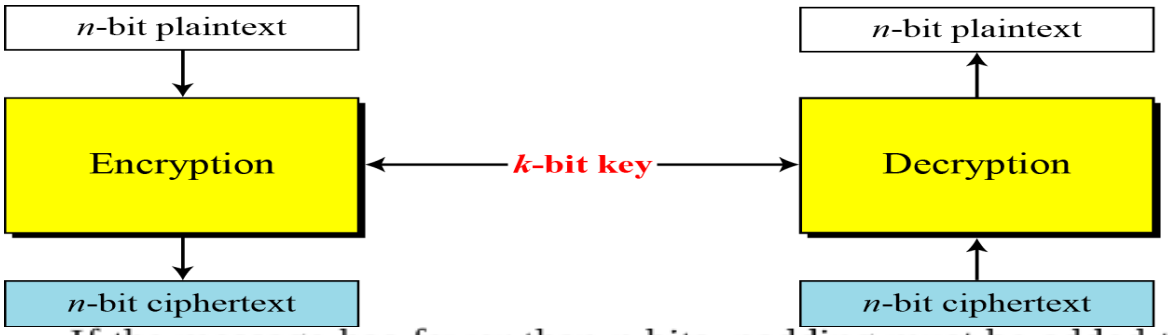


(b) Block cipher

Modern Block Ciphers

A symmetric-key modern block cipher encrypts an n-bit block of plaintext or decrypts an n-bit block of cipher text. The encryption or decryption algorithm uses a k-bit key. The Decryption algorithm must be theinverse of the encryption algorithm and must use the same secrete key.

Figure A modern block cipher



If the message has fewer than *n* bits, padding must be added to make it an *n*-bit block; if the message has more than *n* bits, it should be divided into *n*-bit blocks and the appropriate padding must be added to the last block if necessary. The common values for *n* are 64, 128, 256, or 512 bits.

Example: How many padding bits must be added to a message of 100 characters if 8-bit ASCII is used for encoding and the block cipher accepts blocks of 64 bits?

Solution

Encoding 100 characters using 8-bit ASCII results in an 800-bit (100x8) message. The plaintext must be divisible by 64. If | M | and |Pad| are the length of the message and the length of the padding,

$$|M| + |Pad| = 0 \bmod 64 \quad \rightarrow \quad |Pad| = -800 \bmod 64 \quad \rightarrow \quad 32 \bmod 64$$

A modern block cipher can be designed to act as a substitution cipher or a transposition cipher.

To be resistant to exhaustive-search attack, a modern block cipher needs to be designed as a substitution cipher.

Example

Suppose that we have a block cipher where $n = 64$. If there are 10 1's in the ciphertext, how many trial-and-error tests does Eve need to do to recover the plaintext from the intercepted ciphertext in each of the following cases?

a. The cipher is designed as a substitution cipher.
b. The cipher is designed as a transposition cipher.

Solution

a) In the first case, Eve has no idea how many 1's are in the plaintext. Eve needs to try all possible 2^{64} 64-bit blocks to find one that makes sense.
b) In the second case, Eve knows that there are exactly 10 1's in the plaintext. Eve can launch anexhaustive-search attack using only those 64-bit blocks that have exactly 101's.

Components of a Modern Block Cipher

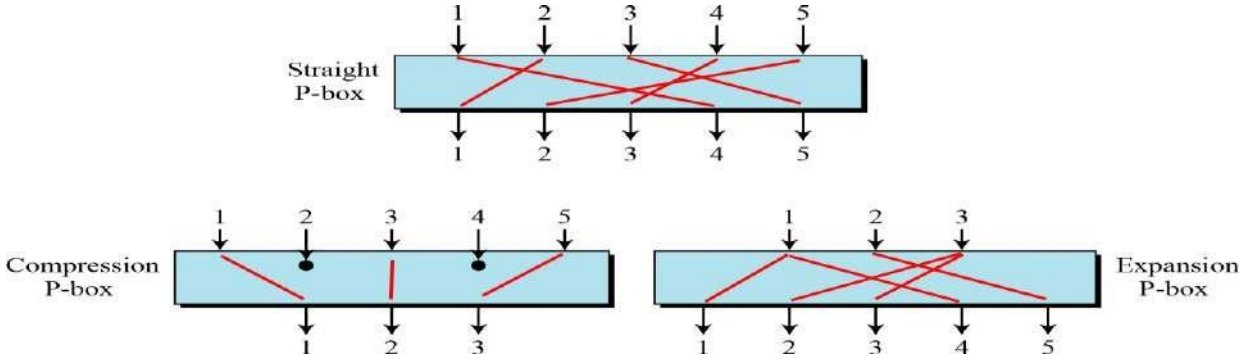
Modern block ciphers normally are keyed substitution ciphers in which the key allows only partial mappingsfrom the possible inputs to the possible outputs. It usses P-Boxes,S-Boxes.

P-Boxes

P-Boxes(also called ad D-Box means Diffusion box)

A P-box (permutation box) parallels the traditional transposition cipher for characters. It transposes bits.

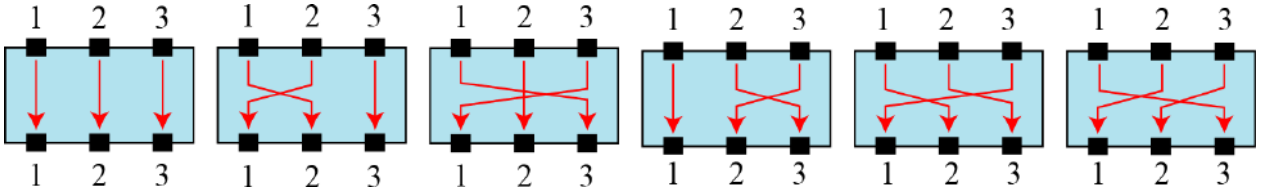
Three types of P-boxes



Example

Figure shows all 6 possible mappings of a 3 × 3 P-box.

The possible mappings of a 3 × 3 P-box



Straight P-Boxes

Table Example of a permutation table for a straight P-box(64x64)At output of P-Box:

Input 58 goes to 1st position, input 50 goes to 2nd position, input 42 to 3rd position,….

58	50	42	34	26	18	10	02	60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06	64	56	48	40	32	24	16	08
57	49	41	33	25	17	09	01	59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05	63	55	47	39	31	23	15	07

Example

Design an 8 × 8 permutation table for a straight P-box that moves the two middle bits (bits 4 and 5) in the input word to the two ends (bits 1 and 8) in the output words. Relative positions of other bits should not be changed.

Solution

We need a straight P-box with the table [4 1 2 3 6 7 8 5]. The relative positions of input bits 1, 2, 3, 6, 7, and 8 have not been changed, but the first output takes the fourth input and the eighth output takes the fifthinput.

Compression P-Boxes

Example of a 32 × 24 permutation table

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

Some of the input bits are blocked at output: example: 7,8,9,15,16,23,24,25

Expansion P-Boxes

Example of a 12 × 16 permutation table

01	09	10	11	12	01	02	03	03	04	05	06	07	08	09	12

1,3,9,12 are mapped to two outputs

P-Boxes: Invertibility

A straight P-Box is invertible, that means we use straight P-Box in encryption cipher and its inverse in decryption cipher.

Note

A straight P-box is invertible, but compression and expansion P-boxes are not.

Example

Figure shows how to invert a permutation table represented as a one-dimensional table.

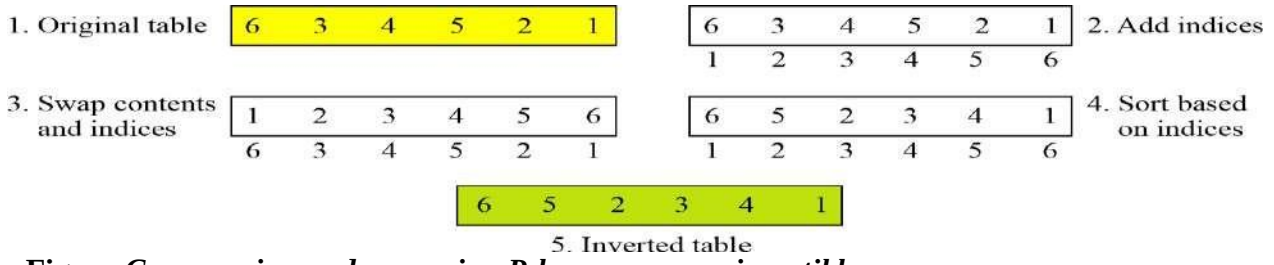
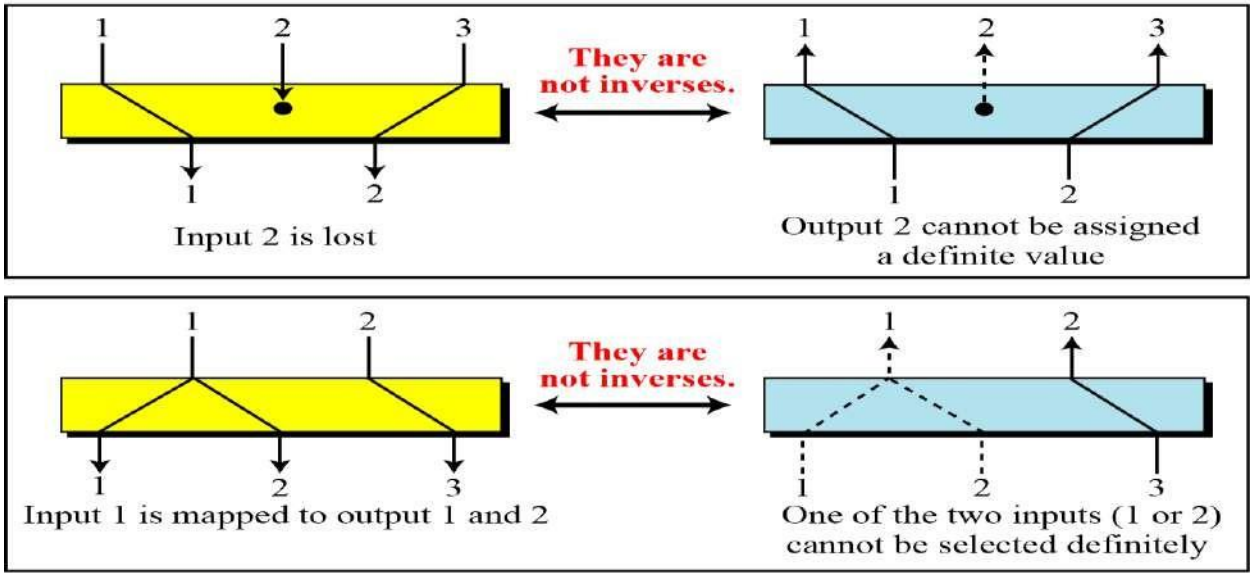


Figure ***Compression and expansion P-boxes are non-invertible***

Compression P-box



Expansion P-box

S-Box

An S-box (substitution box) can be thought of as a small substitution cipher

Note

An S-box is an $m \times n$ substitution unit, where m and n are not necessarily the same.

Linear S-Box: if the inputs are x1,x2,x3… and outputs are y1,y2,y3… and relationship between them is

$Y1=f1(x_1,x_2,x_3..)$,

$Y2=f2\ (x_1,x_2,x_3..)$

…..

Then above relation can be expressedas $Y1=a_{11}x_1+a_{12}x_2+\ldots$

$Y2=a_{21}x_1+a_{22}x_2+\ldots$

Example: In a nonlinear s-box, such boxes can have 'and' terms like x_1x_2, x_3x_5…

In an S-box with three inputs and two outputs, we have

$$y_1 = x_1 \oplus x_2 \oplus x_3 \qquad y_2 = x_1$$

The S-box is linear because $a_{1,1} = a_{1,2} = a_{1,3} = a_{2,1} = 1$ and $a_{2,2} = a_{2,3} = 0$. The relationship can be represented by matrices, as shown below:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Example

In an S-box with three inputs and two outputs, we have

$$y_1 = (x_1)^3 + x_2 \qquad y_2 = (x_1)^2 + x_1x_2 + x_3$$

where multiplication and addition is in GF(2). The S-box is nonlinear because there is no linear relationship between the inputs and the outputs.

Example

The following table defines the input/output relationship for an S-box of size 3 × 2. The leftmost bit of the input defines the row; the two rightmost bits of the input define the column. The two output bits are values on the cross section of the selected row and column.

Leftmost bit

	00	01	10	11
0	00	10	01	11
1	10	00	11	01

Rightmost bits

Output bits

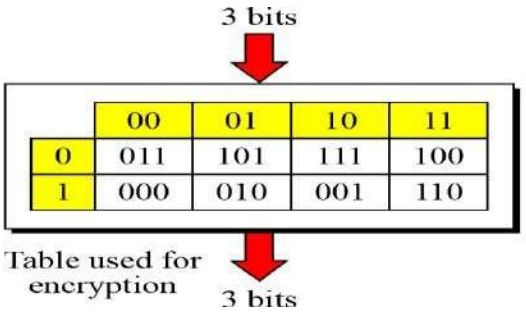
Based on the table, an input of 010 yields the output 01. An input of 101 yields the output of 00.

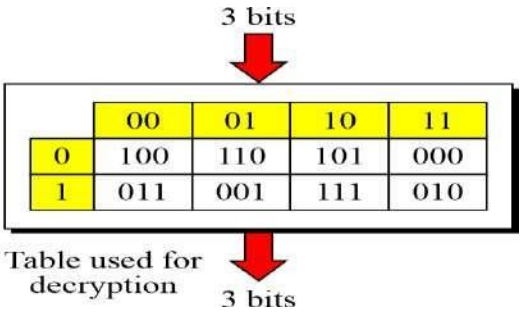
S-Boxes: Invertibility

An S-box may or may not be invertible. In an invertible S-box, the number of input bits should be the same as the number of output bits.

Example

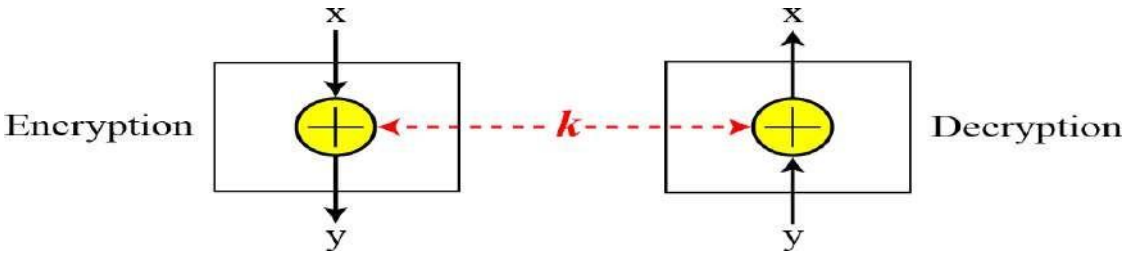
Figure shows an example of an invertible S-box. For example, if the input to the left box is 001, the output is 101. The input 101 in the right table creates the output 001, which shows that the two tables are inverses of each other.





Exclusive-OR

An important component in most block ciphers is the exclusive-or operation.
Invertibility of the exclusive-or operation



Product Ciphers

Shannon introduced the concept of a product cipher. A product cipher is a complex cipher combining substitution, permutation, and other components .
Combination of S-box and P-box transformation—a product cipher.
Two classes of product ciphers:

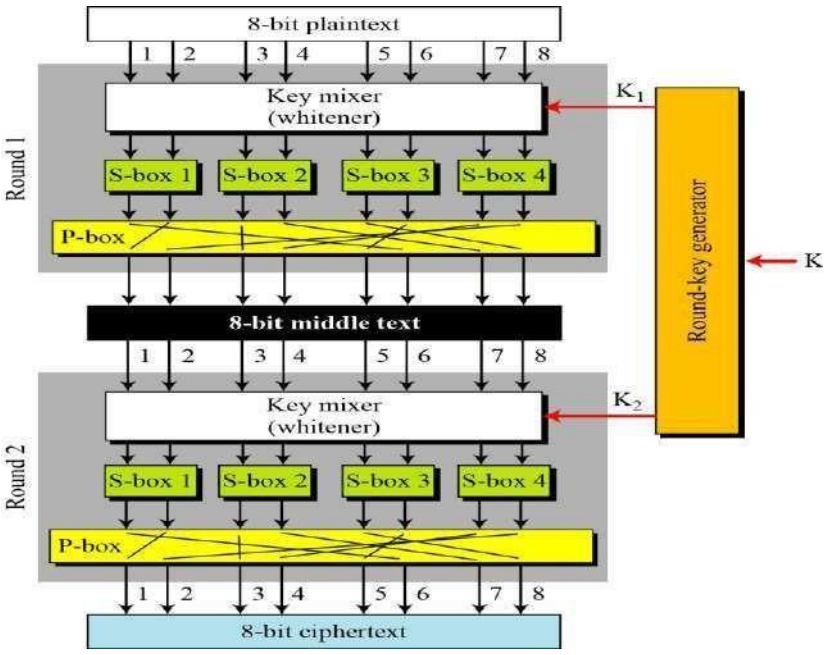
a) Feistel ciphers, Example DES(data encryption standard)
b) Non-feistel Ciphers, Example AES(Advanced Encryptin system)

Diffusion
The idea of diffusion is to hide the relationship between the ciphertext and the plaintext.

Confusion
The idea of confusion is to hide the relationship between the ciphertext and the key.

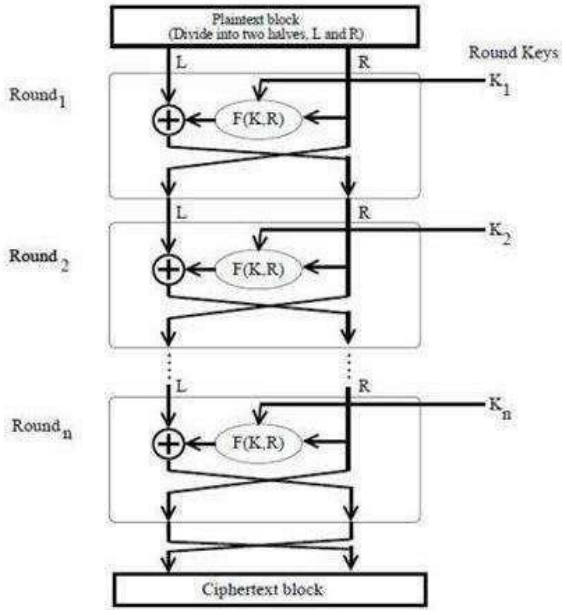
Rounds
Diffusion and confusion can be achieved using iterated product ciphers where each iteration is a combination of S-boxes, P-boxes, and other components.

Figure *A product cipher made of tworounds*



Feistel Cipher Structure:

- Feistel Cipher is not a specific scheme of block cipher. It is a design model from which manydifferent block ciphers are derived.
- DES is just one example of a Feistel Cipher.
- A cryptographic system based on Feistel cipher structure uses the same algorithm for bothencryption and decryption.
- The input block to each round is divided into two halves that can be denoted as L and R for the lefthalf and the right half.
- In each round, the right half of the block, R, goes through unchanged. But the left half, L, goes through an operation that depends on R and the encryption key. First, we apply an encrypting function 'f' that takes two input − the key K and R. The function produces the output f(R,K). Then,we XOR the output of the mathematical function with L.



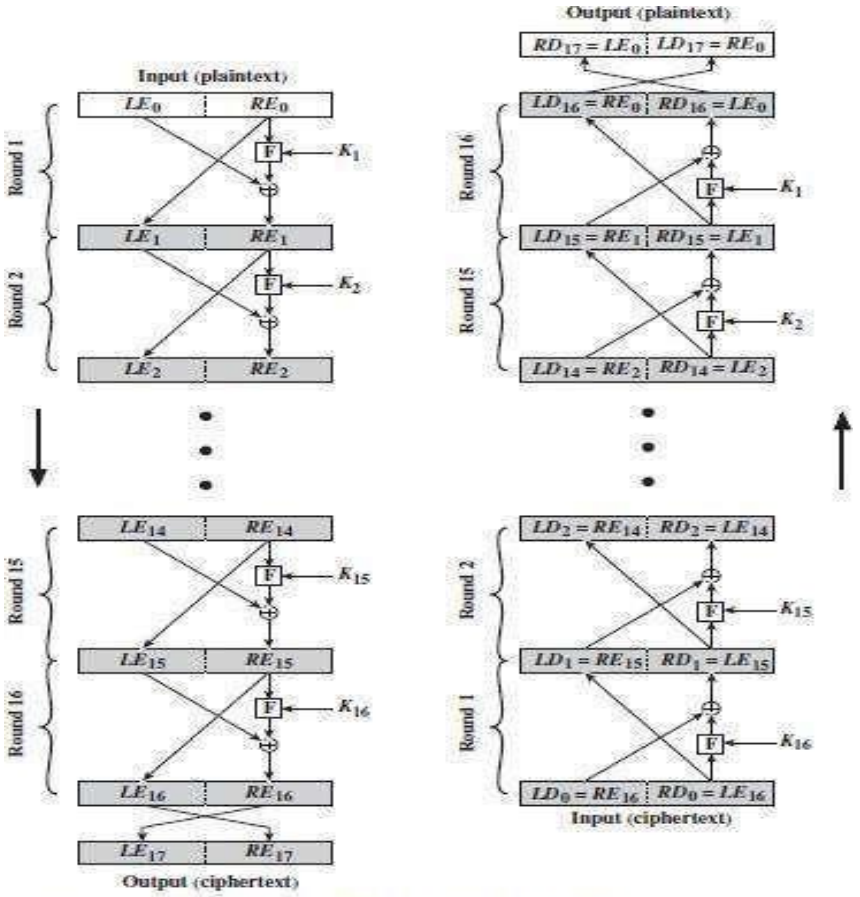


Figure 3.3 Feistel Encryption and Decryption (16 rounds)

$$RE_{i-1} = LE_i$$

$$LE_{i-1} = RE_i \oplus F(RE_{i-1}, K_i) = RE_i \oplus F(LE_i, K_i)$$

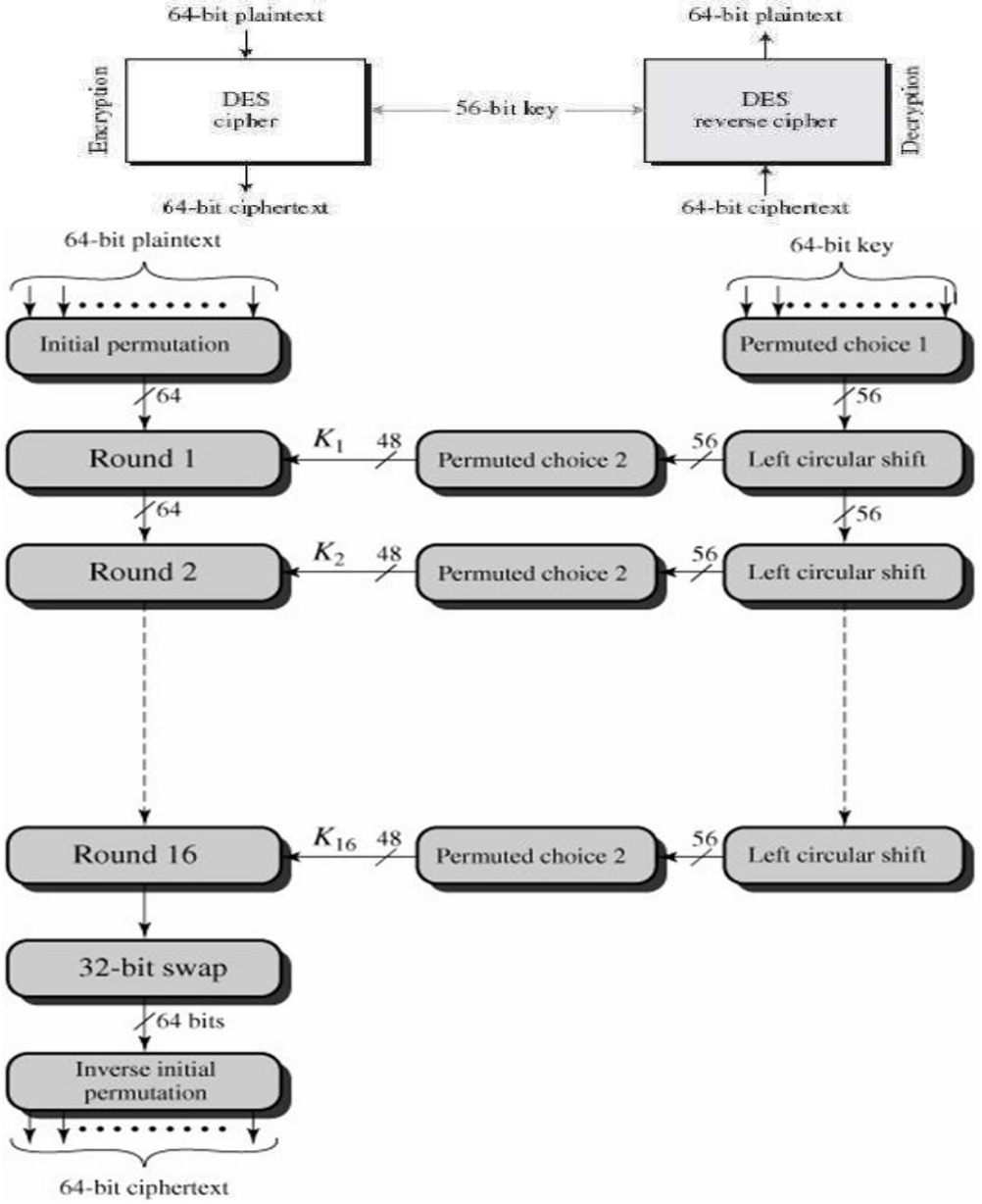
Block Cipher Design Principles

Block size: Larger block sizes mean greater security (all other things being equal) but reduced encryption/decryption speed for a given algorithm. The greater security is achieved by greater diffusion.
Key size: Larger key size means greater security but may decrease encryption/ decryption speed. The greater security is achieved by greater resistance to brute-force attacks and greater confusion
Number of rounds: The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security. A typical size is 16 rounds.
Subkey generation algorithm: Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis.
Round function F: Again, greater complexity generally means greater resistance to cryptanalysis.
Diffusion And Confusion:- The terms *diffusion* and *confusion* were introduced by Claude Shannon to capture the two basic building blocks (Plain Text & Cipher Text) for any cryptographic system.

Data Encryption Standard (DES)

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).
DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64 -bit.

Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key arenot used by the encryption algorithm (function as check bits only).



DES Symmetric key Block Cipher algorithm. DES follows Feistel cipher structure.
Plain Text Block Size : 64
Bits Cipher Text Size : 64 Bits
Master Key Size : 64 / 56 Bits No. Of Rounds 16
Round Key / Subkey Size: 48 Bits.

Initial Permutation & Inverse Initial Permutation

The initial permutation and its inverse are defined by tables, as shown in Tables.
The tables are to be interpreted as follows.
The input to a table consists of 64 bits numbered from 1 to 64.
The 64 entries in the permutation table contain a permutation of the numbers from 1 to 64.

Each entry in the permutation table indicates the position of a numbered input bit in the output, which alsoconsists of 64 bits.

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other.

Note:

Initial Permutation & Inverse Initial Permutations have no cryptography significance in

DES.Input Table

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

(a) Initial Permutation (IP)

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

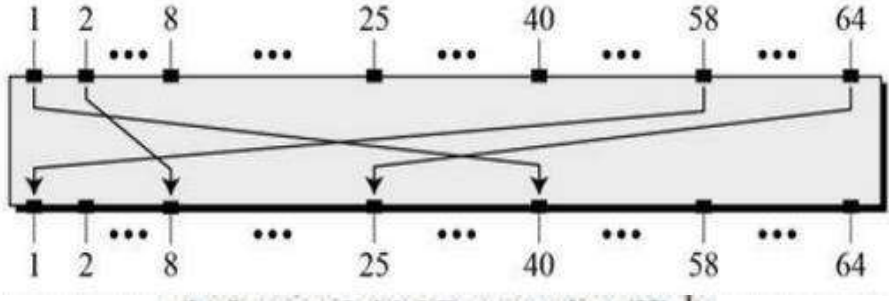
In output

At 1st place

58At 2nd

place 50

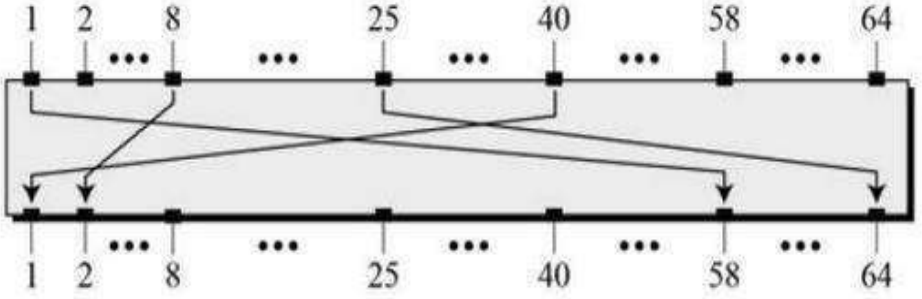
At 3rd place 42 ..



(b) Inverse Initial Permutation (IP^{-1})

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

In output
At 1st place 40
At 2nd place 8
At 3rd place 48



..

Rounds

The left and right halves of each 64- bit intermediate value are treated as separate 32-bit quantities, labeledL (left) and R (right).
As in any classic Feistel cipher, the overall processing at each round can be summarized in the following formulas:

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

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

The round key $\mathbf{K_i}$ is 48 bits. The R input is 32 bits. This **R** input is first expanded to 48 bits by using a tablethat defines a permutation plus an expansion that involves duplication of 16 of the **R** bits.

The resulting 48 bits are XORed with $\mathbf{K_i}$. This 48-bit result passes through a substitution function that produces a 32-bit output, which is permuted as defined by Table.

The role of the **S-boxes** in the function F is illustrated in Figure 3.7.The substitution consists of a set of eightS-boxes, each of which accepts 6 bits as input and produces 4 bits as output. These transformations are defined in Table 3.3, which is interpreted as follows: The first and last bits of the input to box **S_i** form a 2-bitbinary number to select one of four substitutions defined by the four rows in the table for **S_i**. The middle four bits select one of the sixteen columns. The decimal value in the cell selected by the row and column is then converted to its 4-bit representation to produce the output. For example, in S1, for input 011001, the row is 01 (row 1) and the column is 1100 (column 12). The value in row 1, column 12 is 9, so the output is 1001.

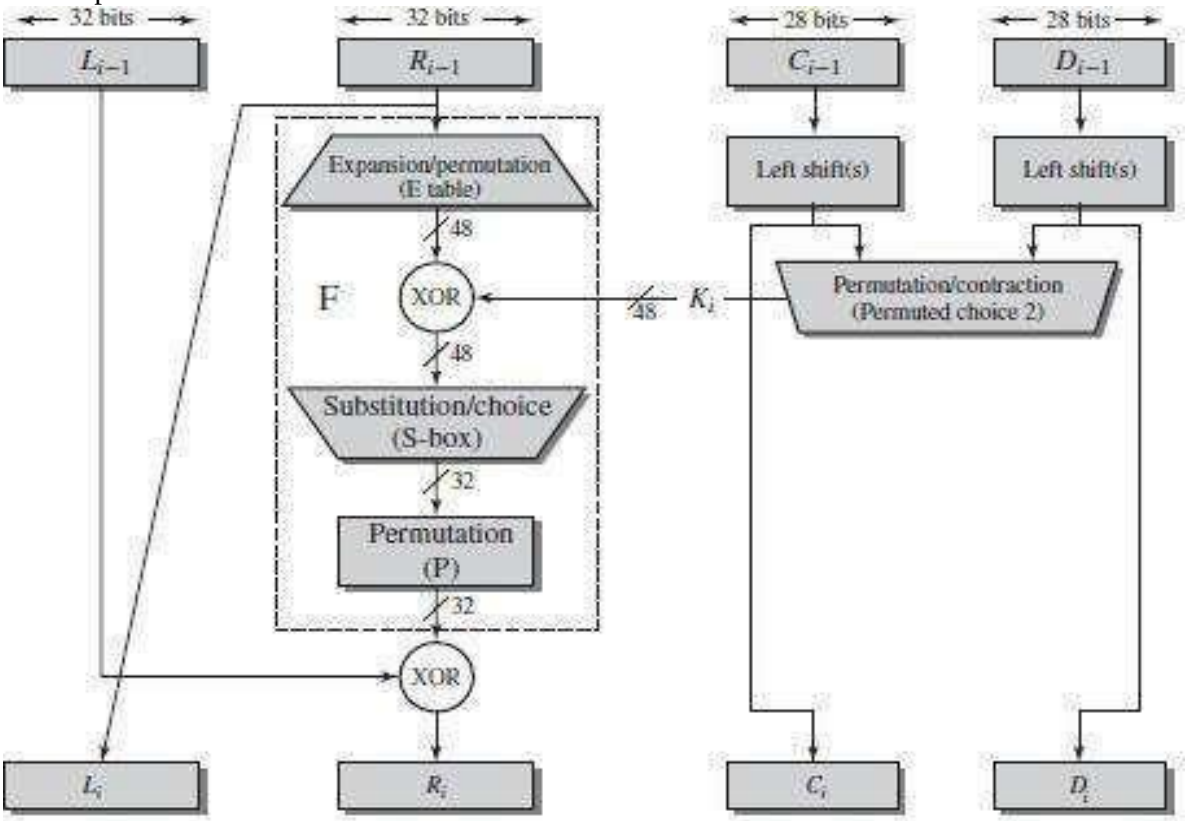


Figure 3.6 Single Round of DES Algorithm

(c) Expansion Permutation (E)

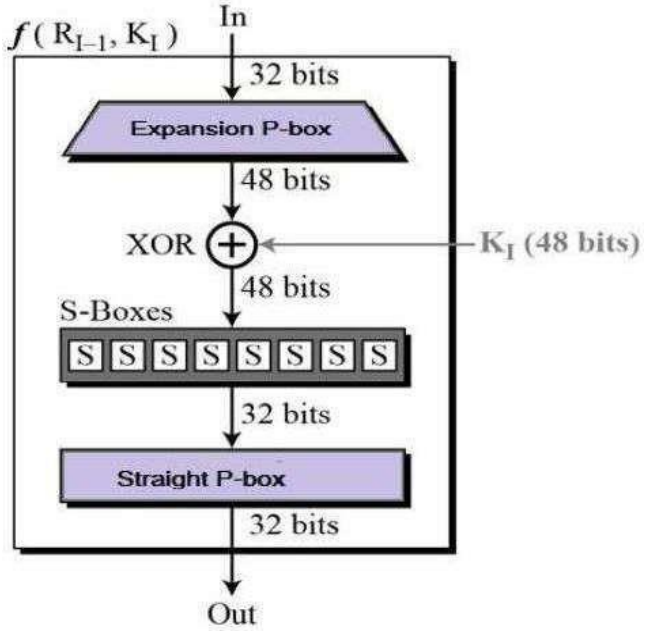
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

(d) Permutation Function (P)

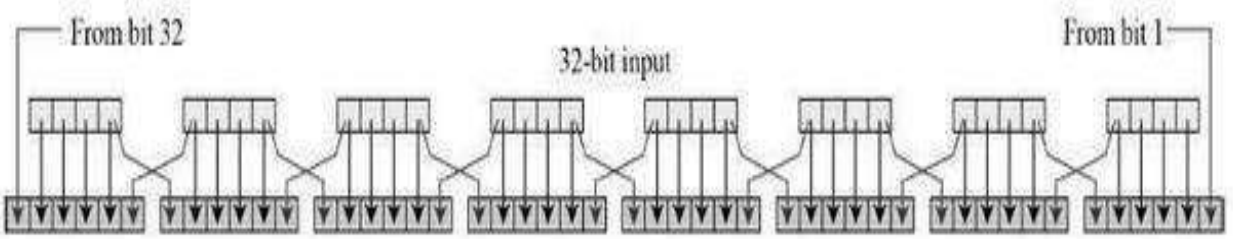
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Round Function

The heart of this cipher is the DES function, *f*. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



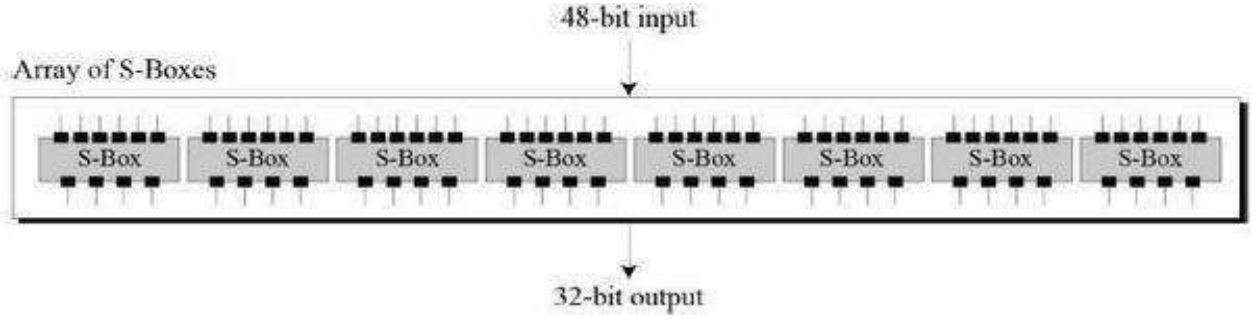
Expansion Permutation Box − Since right input is 32-bit and round key is a 48-bit, we first need to expand right input to 48 bits. Permutation logic is graphically depicted in the following illustration −



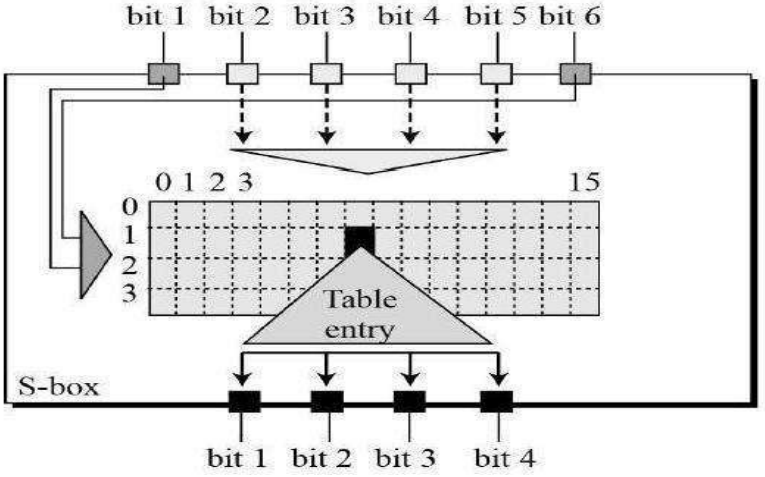
The graphically depicted permutation logic is generally described as table in DES specification illustrated as shown −

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

XOR (Whitener). − After the expansion permutation, DES does XOR operation on the expanded right section and the round key. The round key is used only in this operation.
Substitution Boxes. − The S-boxes carry out the real mixing (confusion). DES uses 8 S-boxes, each with a6-bit input and a 4-bit output. Refer the following illustration −



The S-box rule is illustrated below −



There are a total of eight S-box tables.
The output of all eight s-boxes is then combined in to 32 bit section.

S_5

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

7
8
0
13

S_6

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

10
5
15
9

S_3

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

S_4

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

S_7

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

S_8

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

The 32-bit output from the eight S-boxes is then permuted, so that on the next round, the output from each S-box immediately affects as many others as possible.

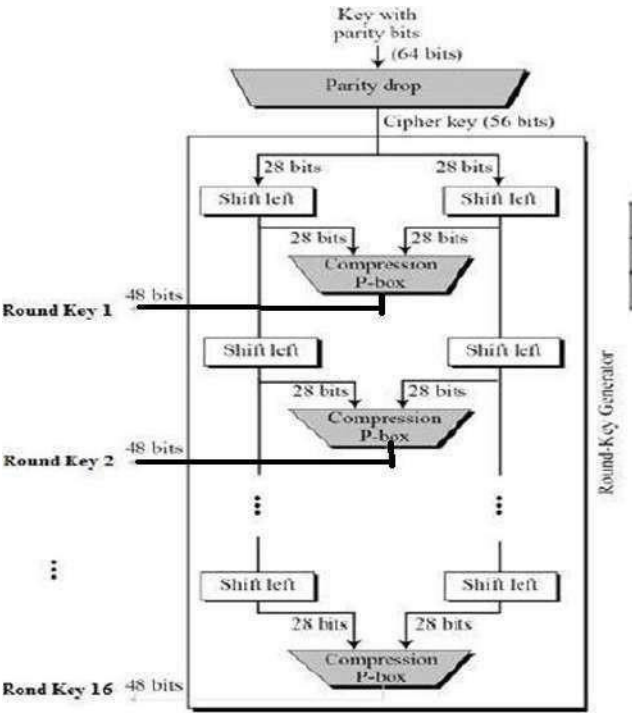
Straight Permutation

− The 32 bit output of S-boxes is then subjected to the straight permutation with rule shown in thefollowing illustration:

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

DES Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. The process of key generation is depicted in the following illustration −



Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits

Table 3.4 DES Key Schedule Calculation

(a) Input Key

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

(b) Permuted Choice One (PC-1)

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

(c) Permuted Choice Two (PC-2)

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

(d) Schedule of Left Shifts

Round Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits Rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

DES Decryption

As with any Feistel cipher, decryption uses the same algorithm as encryption, except that the application ofthe subkeys is reversed.

DES Analysis

Two desired properties of a block cipher are the Avalanche effect and the completeness.

Avalanche effect :

A small change in plaintext results in the very great change in the ciphertext.

EXAMPLE 6.7 To check the avalanche effect in DES, let us encrypt two plaintext blocks (with the same key) that differ only in one bit and observe the differences in the number of bits in each round.

```
Plaintext: 0000000000000000          Key: 22234512987ABB23
Ciphertext: 4789FD476E82A5F1

Plaintext: 0000000000000001          Key: 22234512987ABB23
Ciphertext: 0A4ED5C15A63FEA3
```

Completeness effect:

Completeness effect means that each bit of ciphertext needs to depends on many bits on the plaintext. The diffusion and confusion produced by P-Boxes and S-Boxes in DES, show a very strong completeness effect.

DES Weaknesses Analysis

Weakness in Cipher Design:

It is not clear why the designers of DES used the initial and final permutations; these have no security benefits.

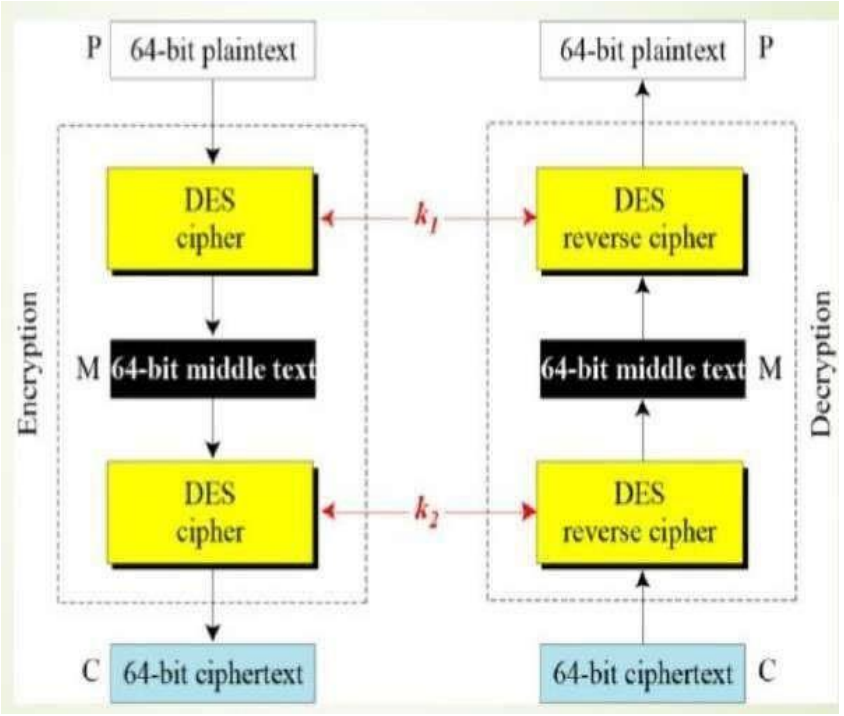
In the expansion permutation, the first and fourth bits of every 4-bit series are repeated.

Weakness in Cipher Key:

DES Key size is 56 bits. To do Brute force attack on a given ciphertext block, the adversary needs to check 2^{56} keys.

With available technology it is possible to check 1 million keys per second

Double – DES

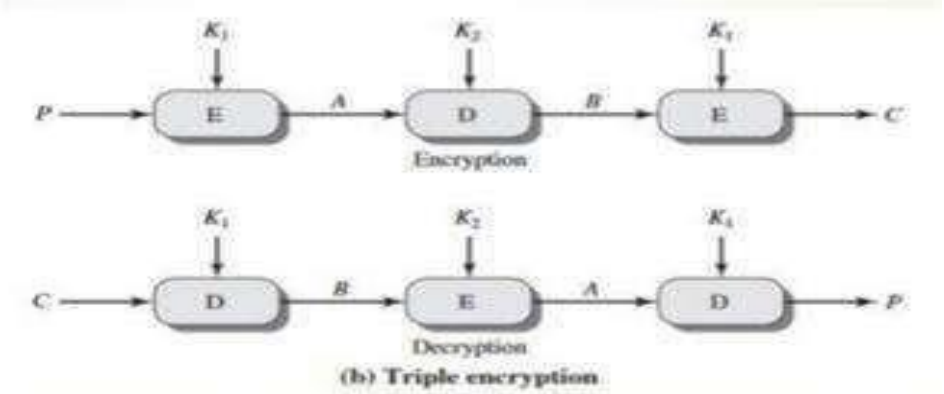


Triple – DES

Triple DES was developed in 1999 by IBM – by a team led by Walter Tuchman. DES prevents a meet-in-the-middle attack. 3-DES has a 168-bit key and enciphers blocks of 64 bits.

3-DES with 2 Keys:

- Use three stages of DES for encryption and decryption.
- The 1st, 3rd stage use K_1 key and 2nd stage use K_2 key.
- To make triple DES compatible with single DES, the middle stage uses decryption in the encryption side and encryption in the decryption side.
- It's much stronger than double DES.



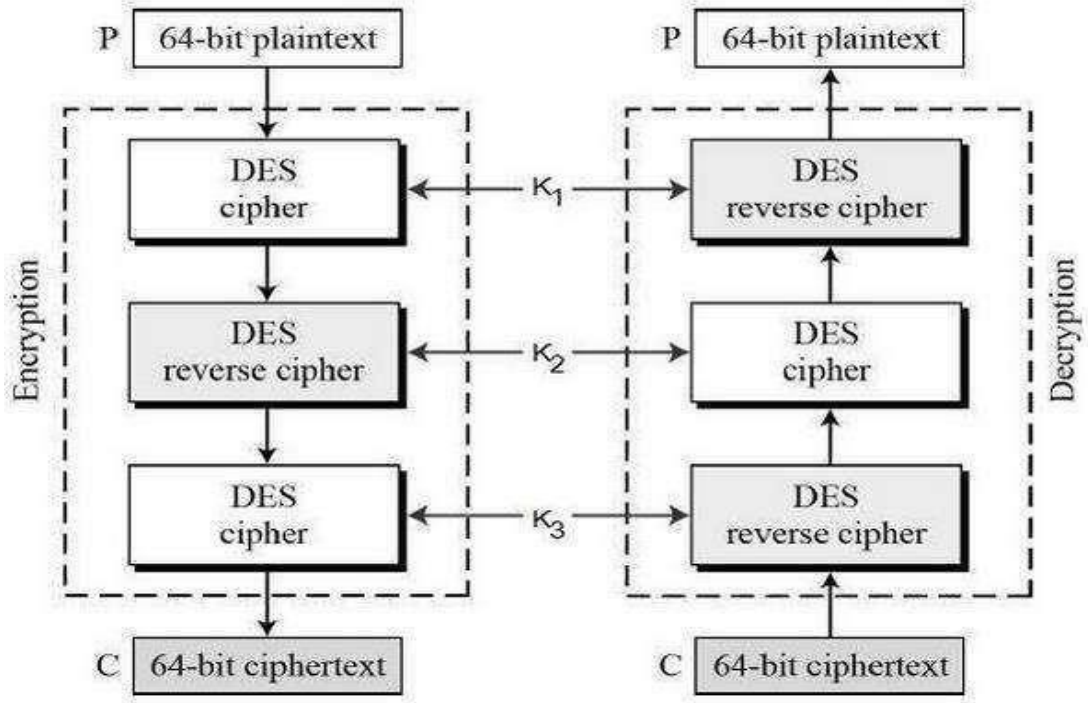
(b) Triple encryption

- The function follows an encrypt-decrypt-encrypt (EDE) sequence.

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

- By the use of triple DES with 2-key encryption, it raises the cost of meet-in-the-middle attack to 2^{112}.
- It has the drawback of requiring a key length of $56 \times 3 = 168$bits which may be somewhat unwieldy.

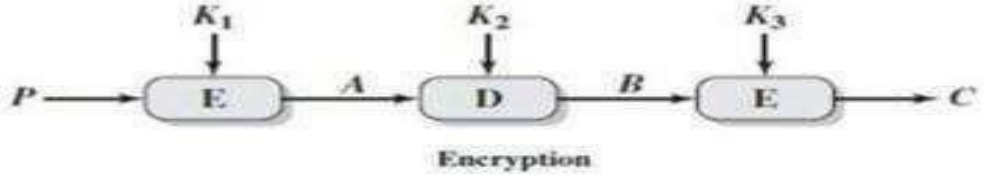
3- DES with 3 Keys:

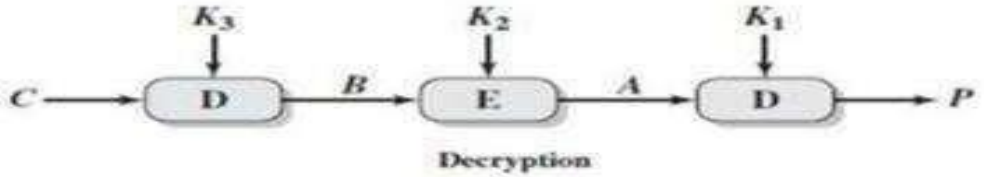


The encryption-decryption process is as follows −
Encrypt the plaintext blocks using single DES with key K_1.
Now decrypt the output of step 1 using single DES with key
K_2.
Finally, encrypt the output of step 2 using single DES with key K_3.
The output of step 3 is the ciphertext.
Decryption of a ciphertext is a reverse process. User first decrypt using K_3, then encrypt with K_2, and finally decrypt with K_1.

- Although the attacks just described appear impractical, anyone using two-key 3DES may feel some concern.
- Thus, many researches now feel that 3-key 3DES is the preferred alternative.
- Use three stages of DES for encryption and decryption with three different keys.
- 3-key 3DES has an effective key length of 168 bits and is defined as,

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





Advanced Encryption Standard(AES Algorithm)

- The Advanced Encryption Standard (AES) was published by the National Institute of Standards and Technology (NIST) in 2001.
- AES is a symmetric block cipher that is intended to replace DES.

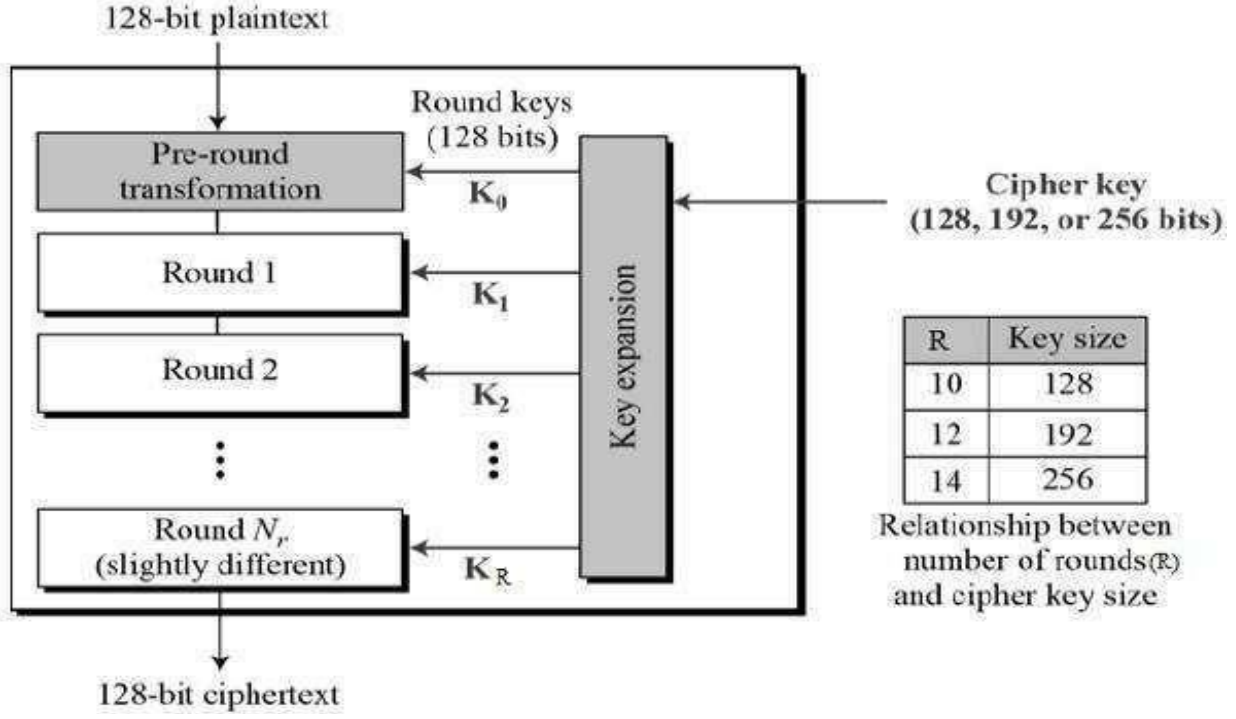
The features of AES are as follows −

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

AES is an iterative rather than Feistel cipher. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).
Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bitsof a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix.

Unlike DES, the number of rounds in AES is variable and depends on the length of the key.
AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The schematic of AES structure is given in the following illustration



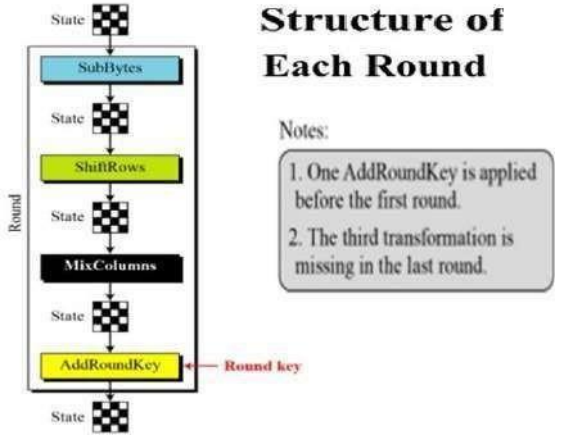
R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

ROUNDS

- Unlike DES, the number of rounds in AES is variable and depends on the length of the key.
- AES uses 10 rounds for 128-bit keys,
- 12 rounds for 192-bit keys and
- 14 rounds for 256-bit keys.
- Each of these rounds uses a different 128-bit round key, which is calculated from the original

AESkey.



Each round comprise of four sub-processes. The first round process is depicted below −

AES Transformations:

There are four transformation functions used in AES Cipher at each round.

1. Substitute Bytes Transformation
2. ShiftRows Transformation
3. MixColumns Transformation
4. AddRoundKey Transformation

1. Byte Substitution (SubBytes)

The 16 input bytes are substituted by values as specified in a table(S-box) given in design.

Each input byte of **State** is mapped into a new byte in the following way:

- The leftmost 4 bits of the byte are used as a row value(in hexadecimal form) and the rightmost 4bits are used as a column value(in hexadecimal form) in S-boxtable.

For example, the hexadecimal value {95} references row 9, column 5 of the S-box, which contains thevalue {2A}. Accordingly, the value {95} is mapped into the value {2A}.

Table 5.2 AES S-Boxes

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

(a) S-box

Here is an example of the SubBytes transformation:

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

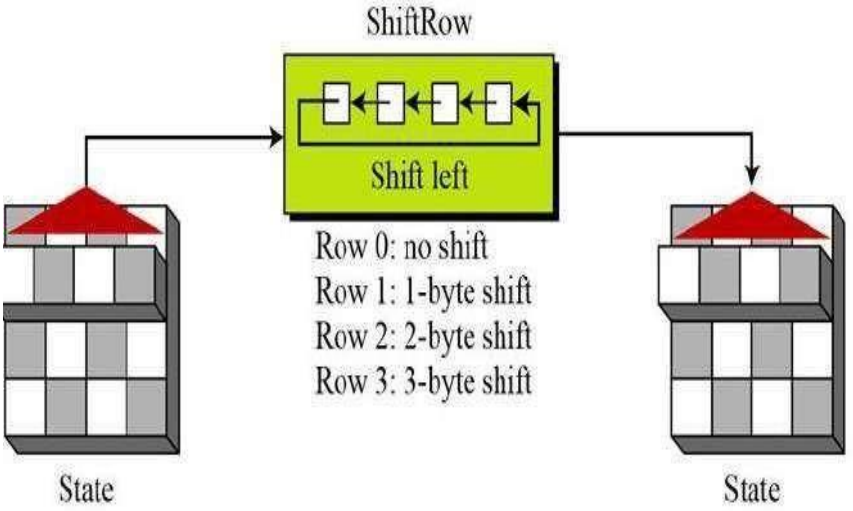
→

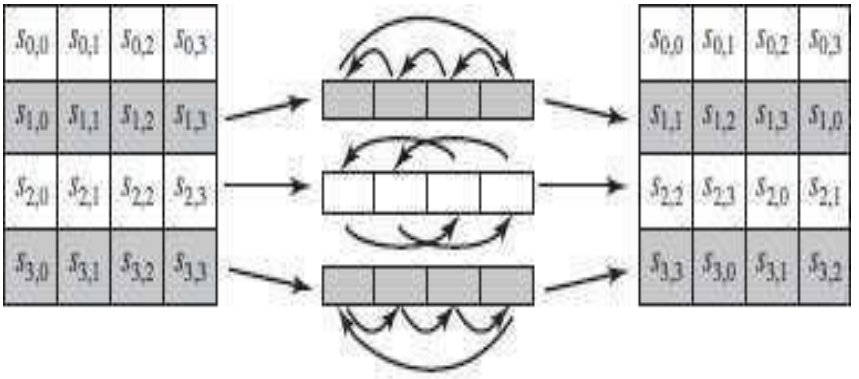
87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

The S-box is constructed in the following fashion (Figure 5.6a).

2. ShiftRows Transformation:

- In this transformation **bytes** are permuted(shifted).
- In the Encryption, the tranformation is called **Shiftrows** and the shifting is to the left.
- The number of shifts depends on the row number(0,1,2,or 3) of the state matrix as shown below:





(a) Shift row transformation

The following is an example of ShiftRows

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

→

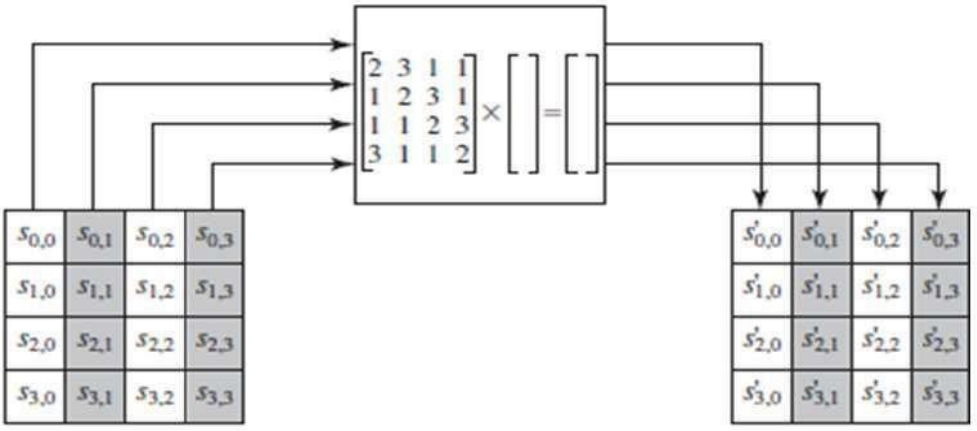
87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

The **inverse shift row transformation**, called InvShiftRows, performs the circular shifts in the opposite direction for each of the last three rows, with a 1-byte circular right shift for the second row, and soon.

3.MixColumns Transformation:

Mixing is the transformaton that changes bits inside byte.
This operation takes 4 bytes(a column) and by multiplying it with a constant matrix then mixes them that produces new bytes.
MixColumn: operates on each column individually. Each byte of a column is mapped into a new value.

It takes a column from state and multiply it with a constant square matrix.
The byte values are represented as polynomials with coefficients in GF(2) and mulitplications are done in $GF(2^8)$



Constant matrices for multiplications:

Figure *Constant matrices used by MixColumns and InvMixColumns*

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \quad \text{Inverse} \quad \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

C $\qquad$ C^{-1}

The following is an example of MixColumns:

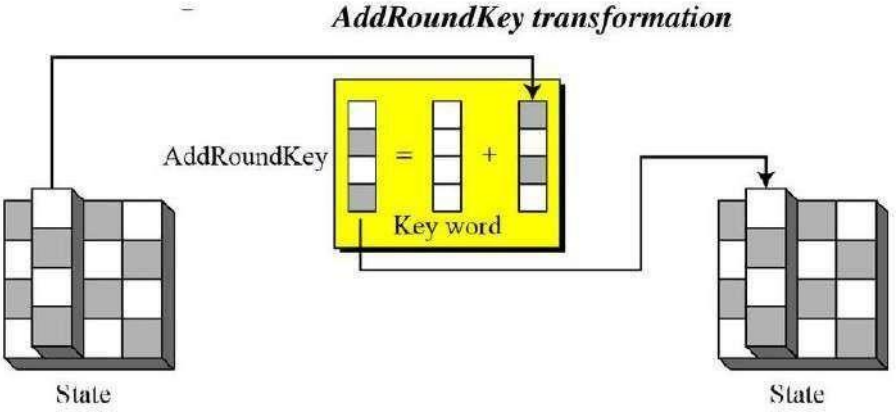
87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

4.AddRoundKey Transformation:

- To make the ciphertext more secrete, we add cipher key to the data in a state.
- AddRoundKey is same as to MixColumns but performs addition operation instead of multiplication.



The following is an example of AddRoundKey

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

⊕

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

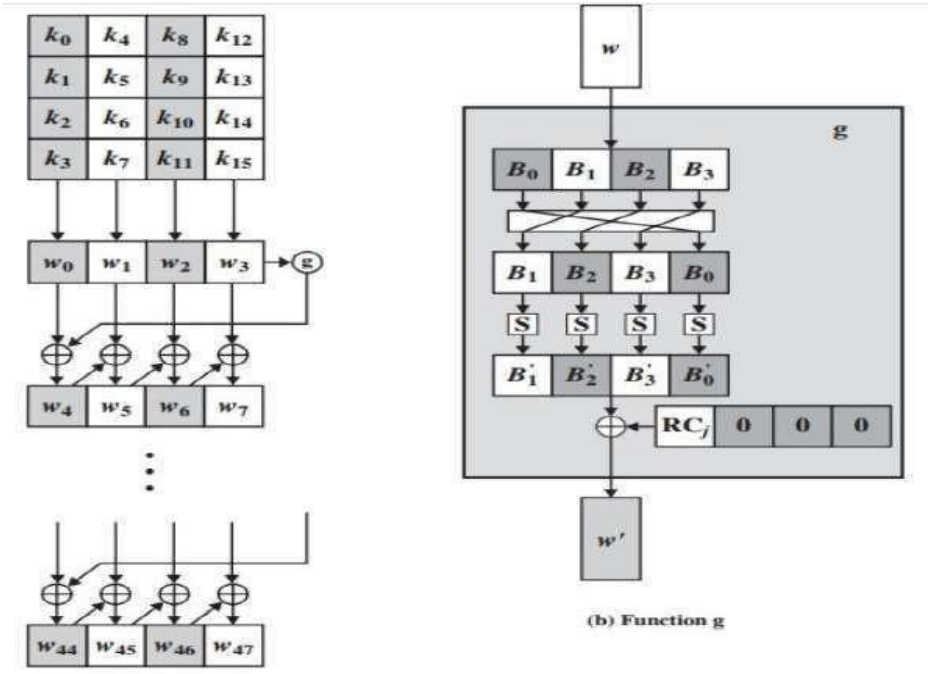
=

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D6

The first matrix is **State**, and the second matrix is the round key.

AES Key Expansion:

- The AES key expansion algorithm takes as input a four-word (16-byte) key and produces a lineararray of 44 words (176 bytes). This is sufficient to provide a four-word round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher.
- The key is copied into the first four words of the expanded key. The remain- der of the expanded key is filled in four words at a time. Each added word **w**[i] depends on the immediately preceding word, **w**[i - 1], and the word four positions back, w[i - 4]. In three out of four cases, a simple XOR is used.
- For a word whose position in the **w** array is a multiple of 4, a more complex function is used. Figure illustrates the generation of the expanded key, using the symbol g to represent that complex function. The function g consists of the following subfunctions.



(b) Function g

(a) Overall algorithm

Figure AES Key Expansion

1. RotWord performs a one-byte circular left shift on a word. This means that an input word $[B_0, B_1, B_2, B_3]$ is transformed into $[B_1, B_2, B_3, B_0]$.
2. SubWord performs a byte substitution on each byte of its input word, using the S-box (Table 5.2a).
3. The result of steps 1 and 2 is XORed with a round constant, Rcon[j].

The round constant is a word in which the three rightmost bytes are always 0. Thus, the effect of an XOR of a word with Rcon is to only perform an XOR on the left-most byte of the word. The round constant is different for each round and is defined as Rcon[j] = (RC[j], 0, 0, 0), with RC[1] = 1, RC[j] = 2 • RC[j−1] and with multiplication defined over the field $GF(2^8)$. The values of RC[j] in hexadecimal are

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36

For example, suppose that the round key for round 8 is

EA D2 73 21 B5 8D BA D2 31 2B F5 60 7F 8D 29 2F

Then the first 4 bytes (first column) of the round key for round 9 are calculated as follows:

i (decimal)	temp	After RotWord	After SubWord	Rcon (9)	After XOR with Rcon	w[i−4]	w[i] = temp ⊕ w[i−4]
36	7F8D292F	8D292F7F	5DA515D2	1B000000	46A515D2	EAD27321	AC7766F3

ANALYSIS OF AES

Security

- *AES was designed after DES. Most of the known attacks on DES were already tested on AES.*
- *Brute-Force Attack*
- *AES is definitely more secure than DES due to the larger-size key.*
- *Statistical Attacks*

- *Numerous tests have failed to do statistical analysis of the ciphertext.*
- *Differential and Linear Attacks*
- *There are no differential and linear attacks on AES as yet.*

Implementation

- *AES can be implemented in software, hardware, and firmware. The implementation can use table lookup process or routines that use a well-defined algebraic structure.*

Simplicity and Cost

- *The algorithms used in AES are so simple that they can be easily implemented using cheap processors and a minimum amount of memory.*