

Types of Systems: -

1. Depending on Encryption/Decryption Key: -

- Symmetric Algorithm: - use one key (which must be secret) for ciphering and deciphering, known as One Key Algorithm.

$C = E_k(P)$ for Encryption

$P = D_k(C)$ for Decryption

A Symmetric Algorithm: - use one key for ciphering and another key for deciphering, known as Public Key Algorithm.

$C = E_{k1}(P)$ for Encryption

$P = D_{k2}(C)$ for Decryption

K1 is Public while K2 is secret.

2. Depending on Cryptographic Techniques: -

The second type of system depends on encryption method.

● **Block Ciphers: -**

• **Stream Ciphers: -**

Block Ciphers:

Is a type of symmetric encryption which operates on blocks of data (means the same key is used for encryption and decryption).

Operate on blocks (groups of bits) with fixed length.

A block cipher breaks messages into fixed sized blocks.

Takes one block (plaintext) at a time and transforms it into another block of the same length using a secret key.

Decryption is performed by applying the reverse transformation to the Ciphertext block using the same key.

Because the same key is used, each repeated sequence in Plaintext becomes the same repeated sequence in Ciphertext, and this could cause security concerns.

• **Popular block ciphers are (Hill Cipher, Playfair Cipher, DES-Data Encryption Standard-, ECB) with using the same key.**

❖ **Hill Cipher:** - In which the size of each block is 2-character, or more, depending on the size of key matrix. In these ciphers, the values of each character in the Ciphertext depends on all the values of all characters in the plaintext, and the key is made of $m \times m$ matrix values, which is considered as a single key.

❖ **Playfair Cipher:** - In which the size of each block equal to 2-characters only that are encrypted together.

❖ **DES (Data Encryption Standard):-** In which each block must be of 64 bits only. This method combines between Substitution and Transposition Cipher, where the type of encryption is called Product Cipher.

❖ **ECB:-** This method has been used by German Submarines (all) in World War II through the use of dictionary contains many words and for every word there is a certain code that this dictionary will be located on both sides sender and receiver.

Advantages of Block Cipher: -

1. The possibility of parallel processing for more than one block at the same time.
2. Encryption is quick because all the time implemented n of encryption.

3. Error that occurs in a given block does not affect the other.

الخطا الذي يحصل في block معين لا يؤثر على غيره.

4. Each block in the Plaintext is encrypted independently.

كل block ضمن النص الصريح يتم تشفيره بصورة مستقلة.

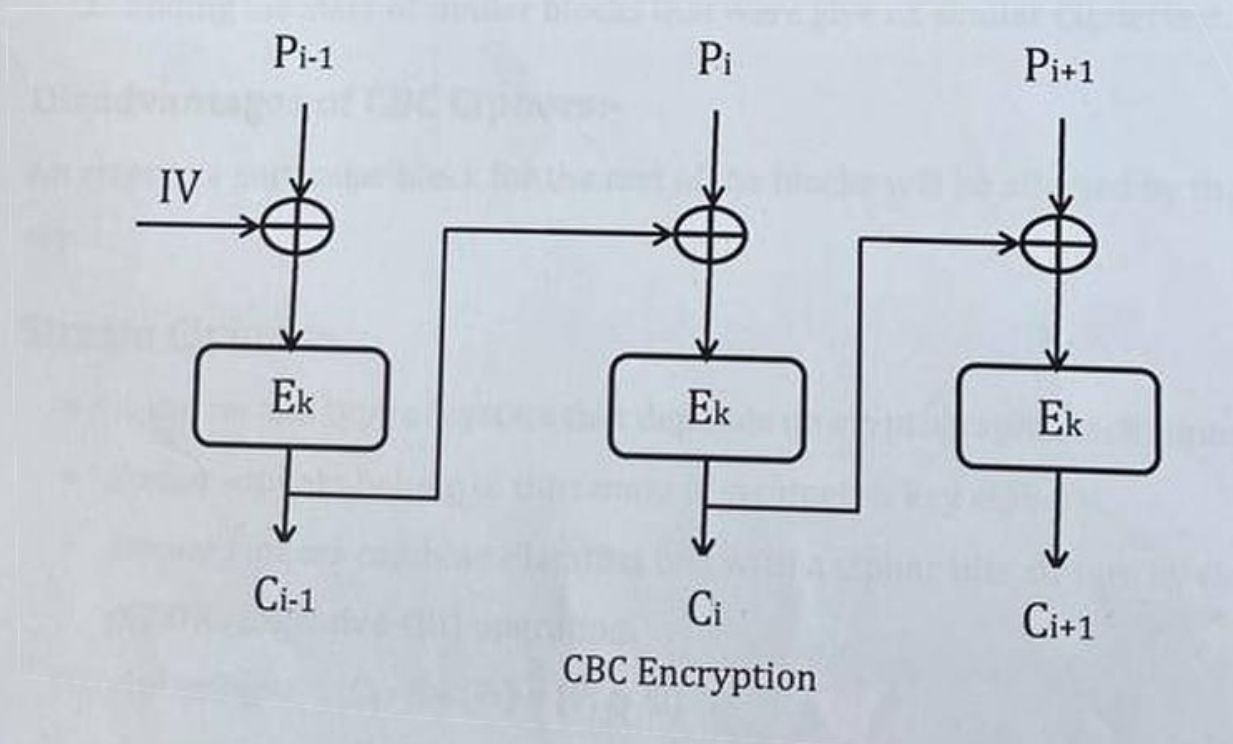
Disadvantages of Block Cipher:-

1. The similar blocks in the plaintext also generate similar blocks in the Ciphertext because all blocks using the same key.
2. Easy addition or deletion can be implemented on blocks.

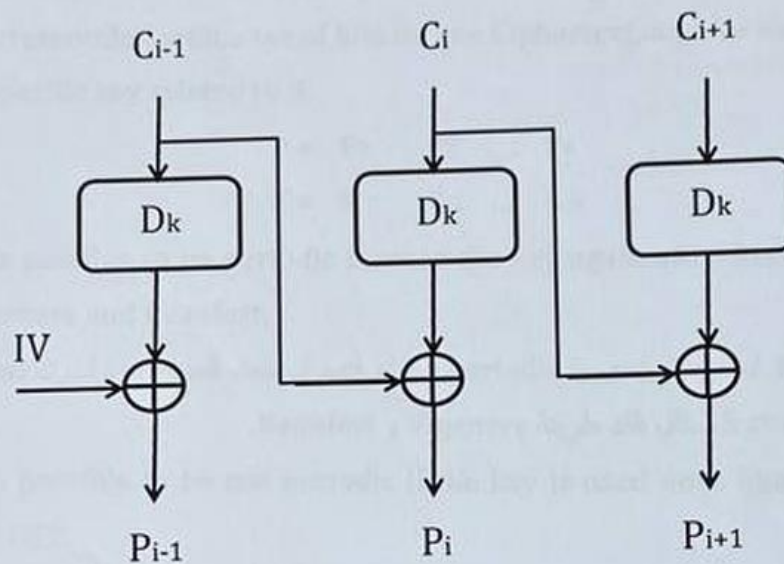
To solve the problem of block cipher is the development of a new way known as **Cipher Block Chaining Mode (CBC)** in which each block depends on the other by connecting them by (Exclusive OR).

For Ciphering :- $C_i = E_k (P_i \oplus C_{i-1})$ For Deciphering :- $P_i = C_{i-1} \oplus D_k (C_i)$

Where :- C_{i-1} is the previous ciphered block.



IV initialization value which is used before first block because no block is found.



CBC Decryption

Advantages of CBC Ciphers:-

1. There is no possibility of addition or deletion.
2. Ending the state of similar blocks that were give us similar Ciphertext.

Disadvantages of CBC Ciphers:-

An error in a particular block for the rest of the blocks will be affected by that error.

Stream Ciphers:-

- Is the second type of system that depends on cryptographic techniques.
- Stream ciphers belong to the family of symmetric key ciphers.
- Stream ciphers combine Plaintext bits with a cipher bits stream by the use of **XOR** (Exclusive-OR) operation.

For ciphering:- $C_i = E_{k_i} (P_i) = (P_i \oplus K_i)$

For deciphering:- $D_k (C_i) = (C_i \oplus K_i)$
 $= ((P_i \oplus K_i) \oplus K_i)$
 $= P_i$

- In this type of ciphering the plaintext is divided into a set of bits that are corresponded with a set of bits for the Ciphertext, and for each bit there is a specific key related to it.

$$P = P_1 \quad P_2 \quad \dots \quad P_n$$

$$K = K_1 \quad K_2 \quad \dots \quad K_n$$

- It is possible to be periodic if reuse the key again after fixed periods, like Vigenere and Beaufort.

Stream Cipher اما ان تكون دورية periodic فيما لو اعيد استخدام المفتاح مرة اخرى بعد فترات ثابتة مثلا بعد 5 characters. مثال ذلك طريقة Vigenere و Beaufort.

- It is possible to be not periodic if the key is used once like Running Key and OTP.

Stream Cipher ممكن ان تكون غير دورية non periodic والتي تستخدم المفتاح مرة واحدة فقط. مثال ذلك Running Key و OTP.

Block Cipher & Stream Cipher Comparison:-

1- Processing or encoding plaintext is done as a fixed length block one by one. A block for example could be 64 or 128 bits in size.	Processing or encoding plaintext is done bit by bit. The block size here is simply one bit.
2- The same key is used to encrypt each of the blocks.	A different key is used to encrypt each of the bits.
3- Usually more complex and slower in operation.	Usually very simple and much faster.
4- More secure in most cases.	Equally secure if properly designed.
5- The key to the cipher text relationship could be very complicated.	Key is often combined with an initialization vector.
6- An error will affect the transformation of all characters in the same block.	An error in the encryption process affects only that character, because each symbol is separately encoded.
7- Slowness of encryption, the person using a block cipher must wait until entire block of plaintext symbols has been received before starting the encryption process.	Speed of transformation, because each symbol is encrypted without regard for any other plain text symbols, each symbol is encrypted as soon as it is read, so the time required to encrypt a symbol depends only on the encryption algorithm itself, not on the time it takes to receive more plaintext.

Feistel Cipher Structure:-

The inputs to the encryption algorithm are a plain text block of length $2w$ bits and a key k . The plain text block is divided into two halves, L_0 and R_0 . The two halves of the data pass through n rounds of processing and then combine to produce the cipher text block. Each round i has as inputs (L_{i-1}) and (R_{i-1}) , derived from the previous round, as well as a sub-key (k_i) derived from the overall k . In general the sub-keys (k_i) are different from k and from each other and are generated from the key by a sub-key generation algorithms.

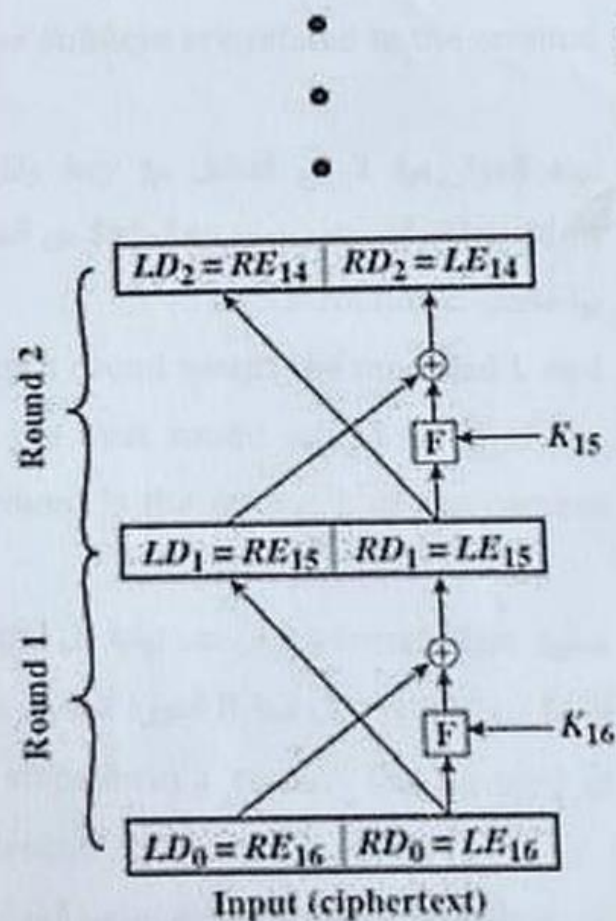
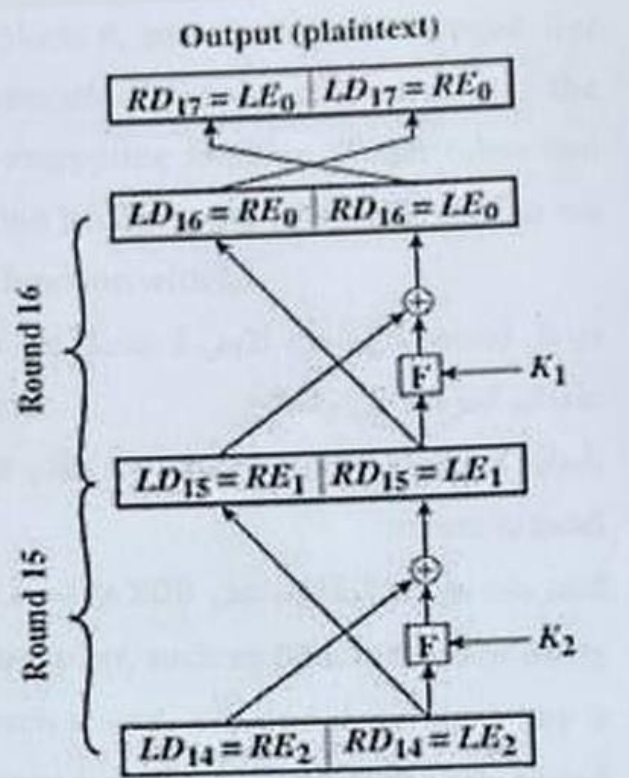
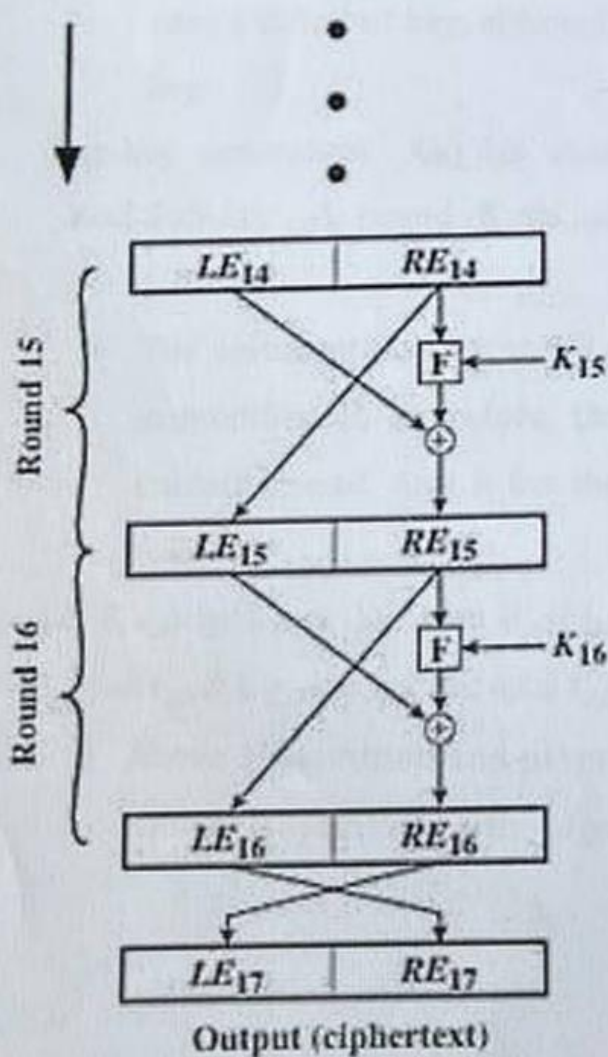
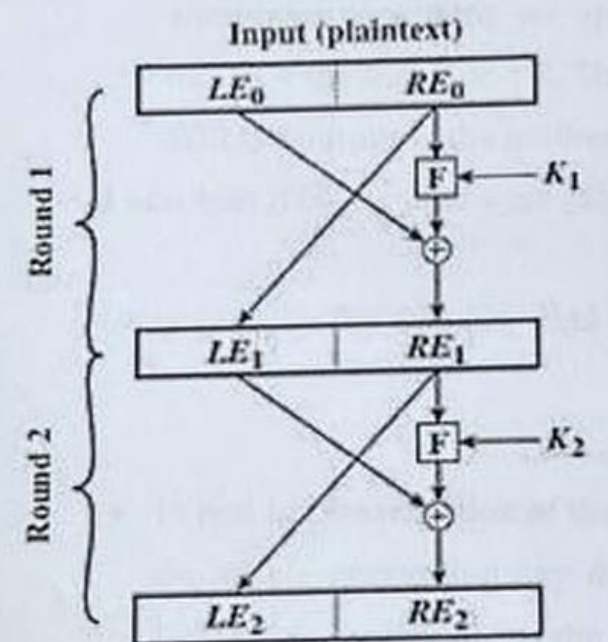
يكون Input لخوارزمية التشفير عبارة عن نص صريح Plaintext بطول $(2W)$ وايضا $key(k)$.
يتم تجزأة Plaintext الى جزأين (L_0, R_0) حيث يمر خلال معالجات دورية تسمى Rounds وبالاخير
تدمج الجزأين مع بعضها للحصول على Ciphertext
كل Round يكون لها Input هي (L_{i-1}, R_{i-1}) من المرحلة السابقة وايضا يكون هنالك Sub-key يتم
اشتقاقه من key الاصلي
بشكل عام تكون قيم Sub-key مختلفة عن key الاصلي وايضا مختلفة عن بعضها حيث يتم توليدها من
المفتاح الاصلي باستخدام خوارزمية Sub-Key Generation Algorithm.

Feistel Cipher Design Elements:-

- **Block Size:-** Larger block sizes mean greater security but reduced encryption/decryption speed.
- **Key Size:-** Larger key size means greater security but may decrease encryption/decryption speed. The most common key length in modern algorithms is 128 bits.
- **Number of Rounds:-** The summary of the *Feistel* cipher is that a single round shows unsuitable security but that multiple rounds offer increasing security. A typical size is 16 rounds.
- **Sub-Keys Generation Algorithm**
- **Round Function (F)**

There are two other considerations in the design of a *Feistel* Cipher:

1. ***Fast software encryption/decryption***
2. ***Ease of Analysis.***



- 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 inputs – the key K and R. The function produces the output $f_{R,K}$. Then we XOR the output of the mathematical function with L.

في كل round فإن الجزء الايمن لا تحصل عليه تغييرات ولكن الجزء الايسر يمر خلال عملية تعتمد فيها على قيم الجزء الايمن والمفتاح
 بالبداية يتم تطبيق (round function f) التي تتعامل مع القيمة Sub-key و الجزء الايمن من النص كمدخلات inputs

المخرجات من هذه الدالة يتم عمل XOR مع الجزء الايسر من النص.

- In real implementation of the **Feistel** Cipher, such as DES, instead of using the whole encryption key during each round, a round-dependent key a **Sub-key** is derived from the encryption key. This means that each round uses a different key, although all these **Subkeys** are related to the original key.

بهذه الخوارزمية لا يتم التعامل مع key ولكن يتم استخدام خوارزمية Sub-key generation algorithm والتي تقوم بتوليد **Sub-key** من المفتاح الاصلي، لذلك كل round يكون **Sub-key** الخاص بها مختلف عن round الاخرى.

- The permutation step at the end of each round swaps the modified L and unmodified R. Therefore, the L for the next round would be R of the current round. And R for the next round is the output L of the current round.

خطوة permutation تكون عند نهاية كل round يتم فيها اجراء عملية تبديل جزء L مع جزء R لذلك فإن L للدورة التالية تمثل قيمة R للدورة الحالية وكذلك R للدورة التالية تمثل قيمة المخرج L للدورة الحالية.

- Above substitution and permutation steps form a 'round'. The number of rounds is specified by the algorithm design.

عدد ال rounds يعتمد على طريقة تصميم الخوارزمية.

- Once the last round is completed then the two sub blocks, 'R' and 'L' are concatenated in this order to form the cipher-text block.

بعد انتهاء اخر round فان sub blocks (L,R) سوف يتم دمجها مع بعضها للحصول على Ciphertext.

The difficult part of designing a **Feistel** Cipher is selection of round function 'f'. In order to be unbreakable scheme, this function needs to have several important properties.

Decryption Process

The process of decryption in **Feistel** cipher is almost similar. Instead of starting with a block of plaintext, the cipher-text block is fed into the start of the **Feistel** structure and then the process thereafter is exactly the same as described in the given illustration. The process is said to be almost similar and not exactly same. In the case of decryption, the only difference is that the sub-keys used in encryption are used in the reverse order.

The final swapping of 'L' and 'R' in last step of the **Feistel** Cipher is essential (primary). If these are not swapped then the resulting cipher-text could not be decrypted using the same algorithm.

فك التشفير في هذه الخوارزمية تكون غالبا مشابهة لعملية التشفير ولكن بدلا من البدا بالنص الصريح يتم البدا ب Ciphertext block، الفرق الوحيد هو في استخدام Sub-key حيث يتم استخدامها بشكل معكوس في حالة فك التشفير.

ملاحظة:- خطوة التبدل تعتبر شيء اساسي ومهم وفي حالة عدم تنفيذها فان الناتج المشفر (Resultant Ciphertext) لا يمكن ان يعمل له Decipher باستخدام نفس الخوارزمية.

Number of Rounds

The number of rounds used in a **Feistel** Cipher depends on desired security from the system. More number of rounds provides more secure system. But at the same time, more rounds mean the inefficient slow encryption and decryption processes.

Encryption	Decryption
Split plaintext block into left and right halves:- Plaintext = (L_0, R_0)	Ciphertext = (L_n, R_n)
For each round $i=1,2,\dots,n$ compute:- $L_i = R_{i-1}$ $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$ where f is round function and K_i is a subkey	For each round $i=n,n-1,\dots,1$ compute:- $R_{i-1} = L_i$ $L_{i-1} = R_i \oplus f(R_{i-1}, K_i)$ where f is round function and K_i is a subkey
Ciphertext = (L_n, R_n)	Plaintext = (L_0, R_0)

Data Encryption Standard (DES):-

The most widely used encryption scheme is defined in the data encryption standard (DES) adopted in 1977 by US Federal Standards, which encrypts 64-bit data using 56-bit key.

The algorithm involves carrying out combinations, substitutions and permutations between the text to be encrypted and the key, while making sure the operations can be performed in both directions (for decryption). The combination of substitutions and permutations is called a **product cipher**.

Overview:-

1. Symmetric Algorithm
2. Block Cipher
3. Uses a combination of Substitution and Transpositions (Permutations).
4. Goes through 16 Cycles (iterations).
5. The overall processing at each cycle can be summarized in the following formulas:-

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

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

6. Plaintext is organized into 64-bit blocks.
7. Uses a 56-bit Key.
8. The complex function involves both permutation and substitution operations. The substitution operation represented as table called "S-Boxes", simply maps each combination of 48 inputs bits into a particular 32-bit pattern.

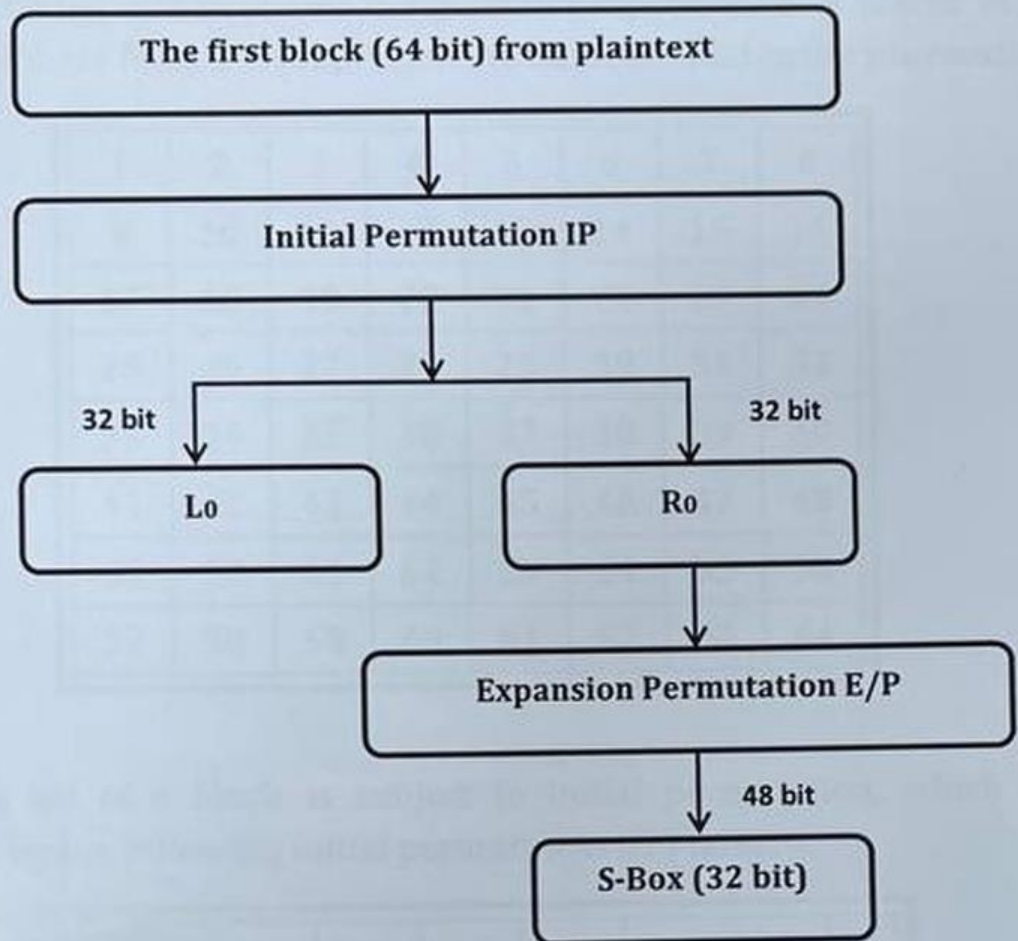
DES Steps:-

1. Initial permutation (IP) on input text (64-bit).
2. Split into right and left halves (32-bit).
3. Take right half and permute it (Expansion Permutation E/P).
4. Work on key (shift) 56-bit, then permute key (48-bits).
5. XOR resulting key with right half... result in 32-bit (S-BOX).
6. Permute result
7. XOR result with left half
8. End of cycle.

Key Transformation:-

1. Starts with 64-bit.
2. Drop every eighth bit = 56 bits (actually 64 bits, 8 bits parity check).
3. Split into two 28-bits halves.
4. Shift each key to the left (a number of bits).
5. Paste both halves.
6. 48-bit key is then permuted.

Plaintext:-



Initial Permutation IP:- is the first step of the data computation.

First the following matrix shows the plaintext represented as blocks of 8-bits, and these numbers from 1-64 represent the number of bit in the plaintext.

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

Firstly, each bit of a block is subject to initial permutation, which can be represented by the following initial permutation (IP) table:-

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

This permutation table shows, when reading the table from left to right then from top to bottom, that the 58th bit of the 64-bit block is in first position, the 50th in second position and so on.

Division into 32-bit blocks :- Once the initial permutation is completed, the 64-bit block is divided into two 32-bit blocks, respectively denoted **L** and **R** (for left and right). The initial status of these two blocks is denoted **L₀** and **R₀**:

L₀	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

R₀	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

It is interesting to note that **L₀** contains all bits having an **even position** in the initial message, whereas **R₀** contains bits with an **odd position**.

Rounds :-

The **L_n** and **R_n** blocks are subject to a set of repeated transformations called **rounds**, shown in this diagram, and the details of which are given below:

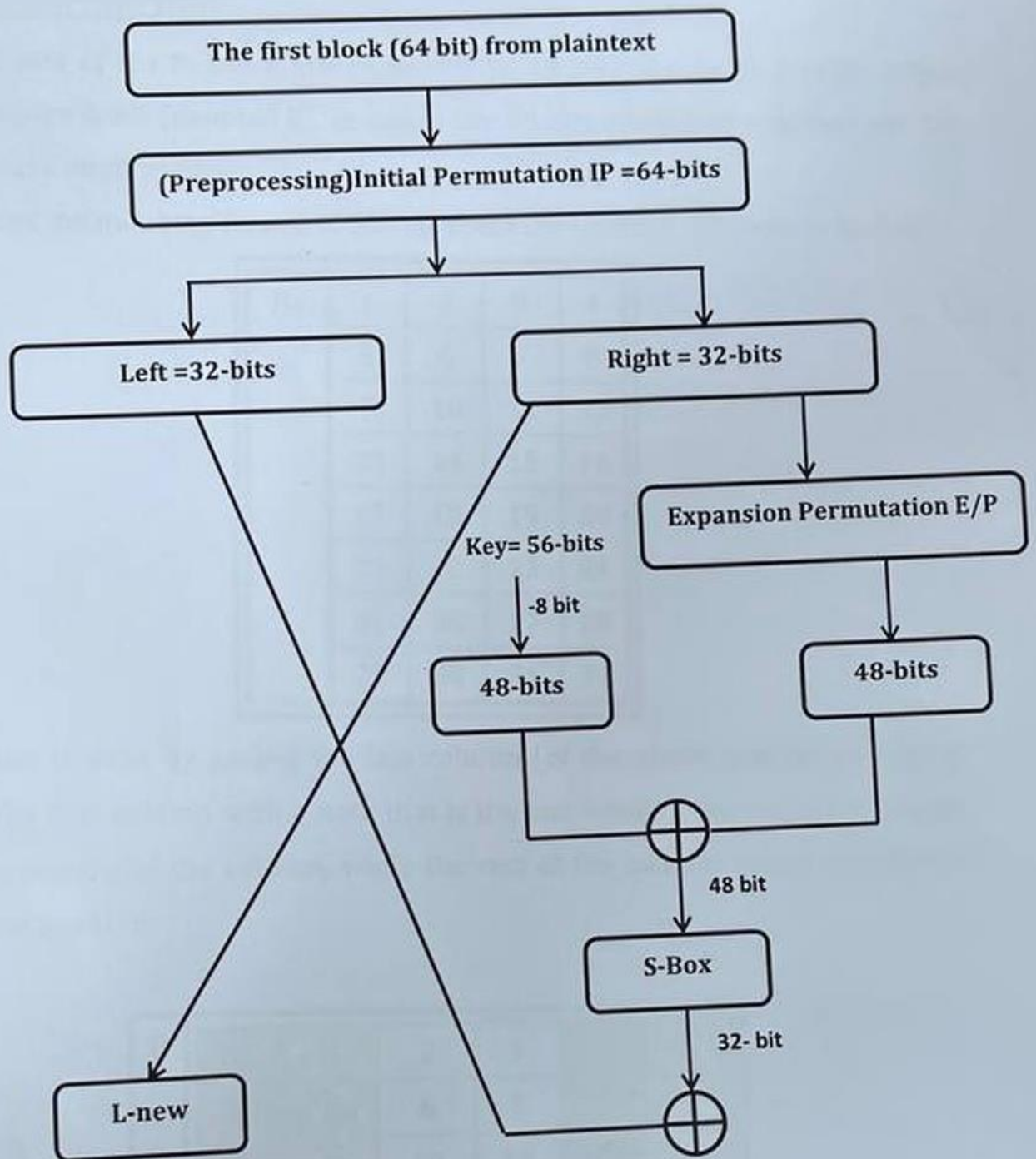
DES Round Structure:-

1. Uses two 32-bit L and R halves.
2. As for any **Feistel** Cipher can describe as:-

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

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

3. F takes 32-bit R half and 48-bit Subkey:-
 - Expand R to 48-bits using Expansion Permutation E/P.
 - Adds to Subkey using XOR.
 - Passes through 8 S-boxes to get 32-bit result.
 - Finally permutes using 32-bit permutation.



Expansion function

The 32 bits of the R_0 block are expanded to 48 bits thanks to a table called an *expansion table* (denoted E), in which the 48 bits are mixed together and 16 of them are duplicated:-

(Note that the numbers from 1 to 32 represent the number of blocks in R_0 half).

R_0	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

Expansion is done by adding the last column (of the above matrix) and put it before the first column with a note that is the last value in the column is placed at the beginning of the column, while the rest of the column values are shifted down one position.

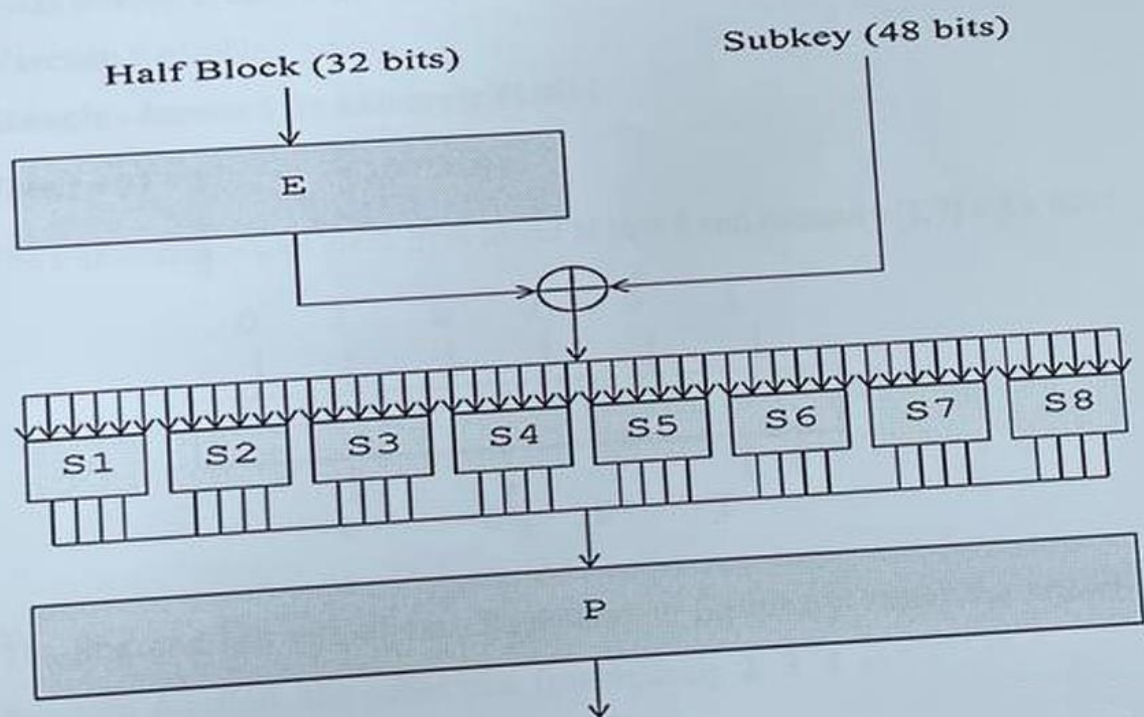
E	32	1	2	3	4	5
	4	5	6	7	8	9
	8	9	10	11	12	13
	12	13	14	15	16	17
	16	17	18	19	20	21
	20	21	22	23	24	25
	24	25	26	27	28	29
	28	29	30	31	32	1

As such, the last bit of R_0 (that is, the 7th bit of the original block) becomes the first, the first becomes the second, etc.

In addition, the bits 1, 4, 5, 8, 9, 12, 13, 16, 17, 20, 21, 24, 25, 28 and 29 of R_0 (respectively 57, 33, 25, 1, 59, 35, 27, 3, 61, 37, 29, 5, 63, 39, 31 and 7 of the original block) are duplicated and scattered in the table.

Exclusive OR with the key:-

The resulting 48-bit table is called R'_0 or $E[R_0]$. The DES algorithm then *exclusive ORs* the first key K_1 with $E[R_0]$. The result of this *exclusive OR* is a 48-bit table we will call R_0 out of convenience (it is not the starting R_0 !).



Substitution function:-

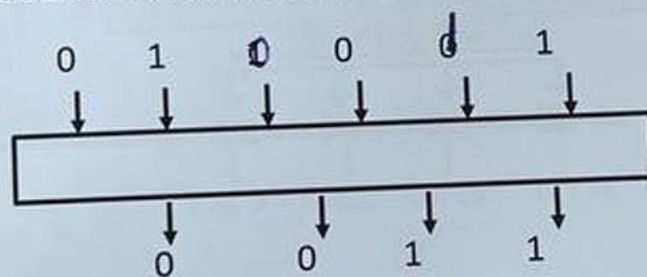
R_0 is then divided into 8 6-bit blocks, denoted R_{0i} . Each of these blocks is processed by **selection functions** (sometimes called **Substitution Boxes** or **Compression Functions S-Boxes**), generally denoted S_i . The s-boxes are substitution based on table of 4 rows and 16 columns. Suppose the block B_j is the six bits $(b_1, b_2, b_3, b_4, b_5, b_6)$. Bits b_1 and b_6 take together, from a two-bit binary number $b_1 b_6$, having a decimal value from 0 to 3, call this value r .

Bits b_2, b_3, b_4 and b_5 taken together from a four-bit binary number $b_2 b_3 b_4 b_5$ having a decimal value from 0 to 15 call the value c . The substitution from the S-Boxes transform each 6-bit block B_j into the 4-bit result show in row r , column c of section S_j of table.

Example:- Assume S_7 in a binary is 010011

Then $r = 01 = 1$ $c = 1001 = 9$

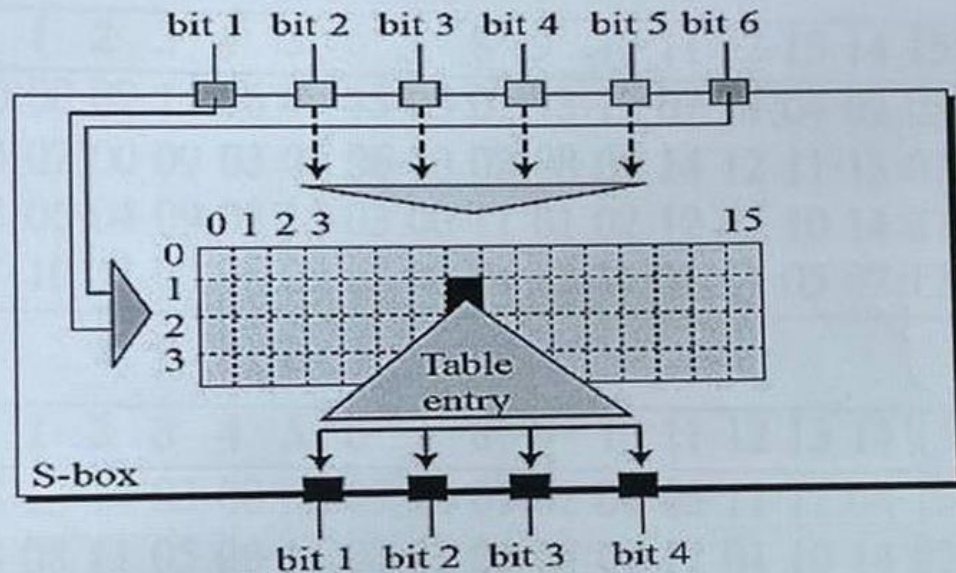
The transformation of block B_7 is found in row 1 and column $= (1, 9) = 3 = 0011$



The first and last bits of each R_{0i} determine (in binary value) the line of the selection function; the other bits (respectively 2, 3, 4 and 5) determine the columns. As the selection of the line is based on two bits, there are 4 possibilities (0, 1, 2, 3). As the selection of the column is based on 4 bits, there are 16 possibilities (0 to 15). Thanks to this information, the selection function "selects" a ciphered value of 4 bits.

Here is the first substitution function, represented by a 4-by-16 table:-

S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13



Let R_{01} equal 101110 . The first and last bits give 10 , that is, 2 in binary value. The bits 2, 3, 4 and 5 give 0111 , or 7 in binary value. The result of the selection function is therefore the value located on line no. 2, in column no. 7. It is the value 11 , or 111 binary.

Each of the 8 6-bit blocks is passed through the corresponding selection function, which gives an output of 8 values with 4 bits each.

Here are the other selection functions:-

S_2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	01	08	14	06	11	03	04	09	07	02	13	12	00	05	10
1	03	13	04	07	15	02	08	14	12	00	01	10	06	09	11	05
2	00	14	07	11	10	04	13	01	05	08	12	06	09	03	02	15
3	13	08	10	01	03	15	04	02	11	06	07	12	00	05	14	09

S_3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	00	09	14	06	03	15	05	01	13	12	07	11	04	02	08
1	13	07	00	09	03	04	06	10	02	08	05	14	12	11	15	01
2	13	06	04	09	08	15	03	00	11	01	02	12	05	10	14	07
3	01	10	13	00	06	09	08	07	04	15	14	03	11	05	02	12

S_4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	07	13	14	03	00	06	09	10	01	02	08	05	11	12	04	15
1	13	08	11	05	06	15	00	03	04	07	02	12	01	10	14	09
2	10	06	09	00	12	11	07	13	15	01	03	14	05	02	08	04
3	03	15	00	06	10	01	13	08	09	04	05	11	12	07	02	14

S_5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	02	12	04	01	07	10	11	06	08	05	03	15	13	00	14	09
1	14	11	02	12	04	07	13	01	05	00	15	10	03	09	08	06
2	04	02	01	11	10	13	07	08	15	09	12	05	06	03	00	14
3	11	08	12	07	01	14	02	13	06	15	00	09	10	04	05	03

S_6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	01	10	15	09	02	06	08	00	13	03	04	14	07	05	11
1	10	15	04	02	07	12	09	05	06	01	13	14	00	11	03	08
2	09	14	15	05	02	08	12	03	07	00	04	10	01	13	11	06
3	04	03	02	12	09	05	15	10	11	14	01	07	06	00	08	13

S_7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	04	11	02	14	15	00	08	13	03	12	09	07	05	10	06	01
1	13	00	11	07	04	09	01	10	14	03	05	12	02	15	08	06
2	01	04	11	13	12	03	07	14	10	15	06	08	00	05	09	02
3	06	11	13	08	01	04	10	07	09	05	00	15	14	02	03	12

S_8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	02	08	04	06	15	11	01	10	09	03	14	05	00	12	07
1	01	15	13	08	10	03	07	04	12	05	06	11	00	14	09	02
2	07	11	04	01	09	12	14	02	00	06	10	13	15	03	05	08
3	02	01	14	07	04	10	08	13	15	12	09	00	03	05	06	11

Each 6-bit block is therefore substituted in a 4-bit block. These bits are combined to form a 32-bit block.

Permutation:-

The obtained 32-bit block is then subject to a permutation **P** here is the table:-

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

Exclusive OR:-

All of these results output from **P** are subject to an *Exclusive OR* with the starting **L₀** (as shown on the first diagram) to give **R₁**, whereas the initial **R₀** gives **L₁**.

Iteration:- All of the previous steps (*rounds*) are repeated 16 times.

Inverse initial permutation:-

At the end of the iterations, the two blocks **L₁₆** and **R₁₆** are re-joined, then subject to 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

The output result is a 64-bit Ciphertext

Example:- If the plaintext = mohammed

To find the initial permutation:-

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
0	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

Using this to convert to binary form :- 128, 64, 32, 16, 8, 4, 2, 1

m = 12 = 00001100
o = 14 = 00001110
h = 7 = 00000111
a = 0 = 00000000
m = 12 = 00001100
m = 12 = 00001100
e = 4 = 00000100
d = 3 = 00000011

1/0	2/0	3/0	4/0	5/1	6/1	7/0	8/0
9/0	10/0	11/0	12/0	13/1	14/1	15/1	16/0
17/0	18/0	19/0	20/0	21/0	22/1	23/1	24/1
25/0	26/0	27/0	28/0	29/0	30/0	31/0	32/0
33/0	34/0	35/0	36/0	37/1	38/1	39/0	40/0
41/0	42/0	43/0	44/0	45/1	46/1	47/0	48/0
49/0	50/0	51/0	52/0	53/0	54/1	55/0	56/0
57/0	58/0	59/0	60/0	61/0	62/0	63/1	64/1

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

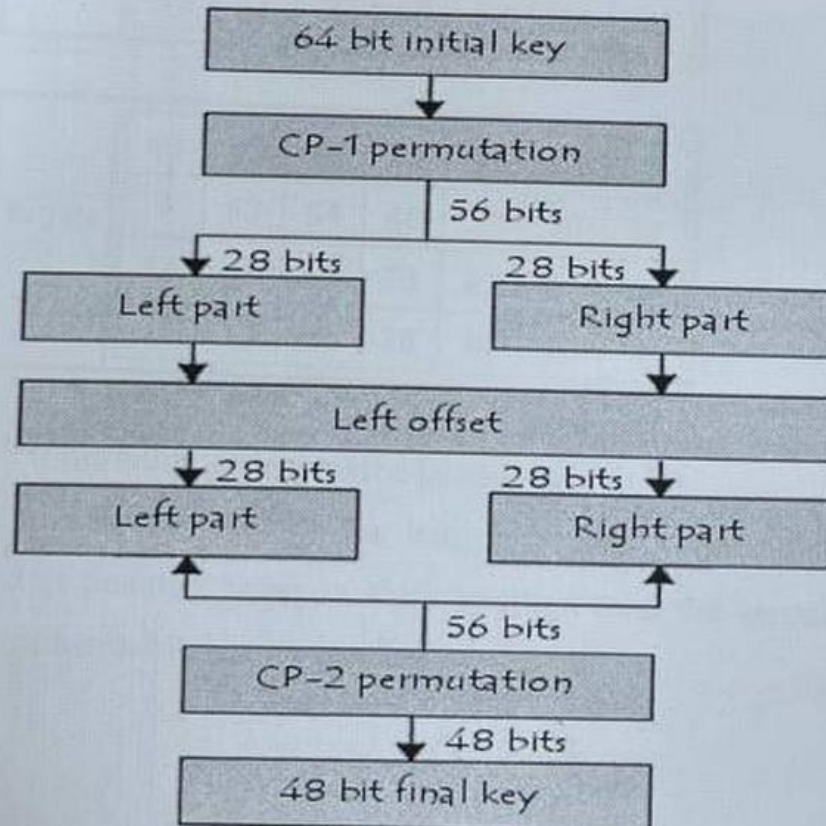
4

IP	0	0	0	0	0	0	0	00000000
	0	0	0	0	0	0	0	00000000
	0	1	1	1	0	1	1	01110111
	1	0	0	0	0	1	0	10000100
	0	0	0	0	0	0	0	00000000
	0	0	0	0	0	0	0	00000000
	0	0	1	1	0	0	1	00110011
	1	0	0	0	0	1	1	10000110

Generation of keys:-

Given that the DES algorithm presented above is public, security is based on the complexity of encryption keys.

The algorithm below shows how to obtain, from a 64-bit key (made of any 64 alphanumeric characters), 8 different 48-bit keys each used in the DES algorithm:



Firstly, the key's parity bits are eliminated so as to obtain a key with a useful length of 56 bits.

The first step is a permutation denoted **PC-1** (permutation choice-1) whose table is presented below:-

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

This table may be written in the form of two tables L_i and R_i (for left and right) each made of 28 bits:-

$58+8=66-65=1$
 $59+8=67-65=2$
 $60+8=68-65=3$

L_i / C_0	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

$62+8=70-63=7$
 $61+8=69-63=6$

R_i / D_0	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

The result of this first permutation is denoted L_0 and R_0 .

These two blocks are then rotated to the left; such that the bits in second position take the first position, those in third position take the second, etc. the bits in first position move to last position.

The two 28-bit blocks are then grouped into one 56-bit block. This passes through a permutation, denoted **PC-2 (permutation choice-2)**, giving a 48-bit block as output, representing the key K_i .

Left Shift (LS):-

Number of rounds	Shifting amount (Offset)
1, 2, 9, 16	1
3, 4, 5, 6, 7, 8, 10	2
15, 14, 13, 12, 11	2

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

Repeating the algorithm makes it possible to give the 16 keys K_1 to K_{16} used in the DES algorithm.