

Introduction to Cryptography

Prepared by Dr. Mohammad Alkhatib

Associate Professor

Major: Information Security



Lecture #3: DES Algorithm



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY



Cryptography Algorithms

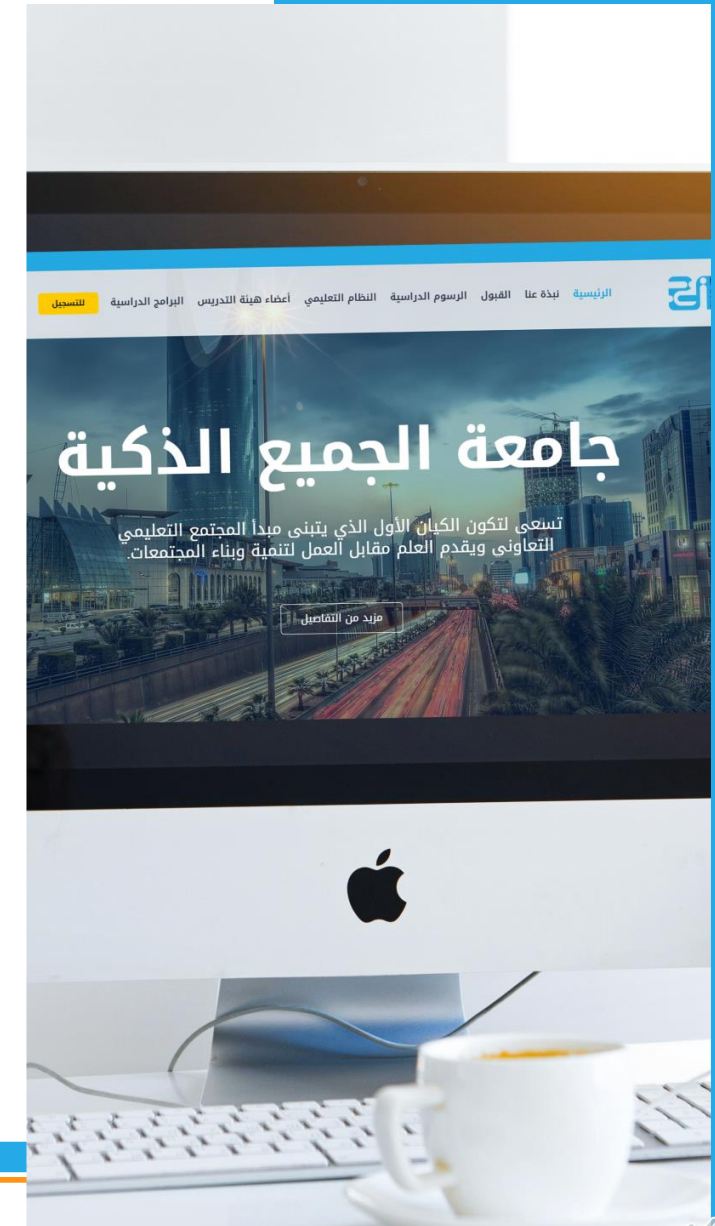
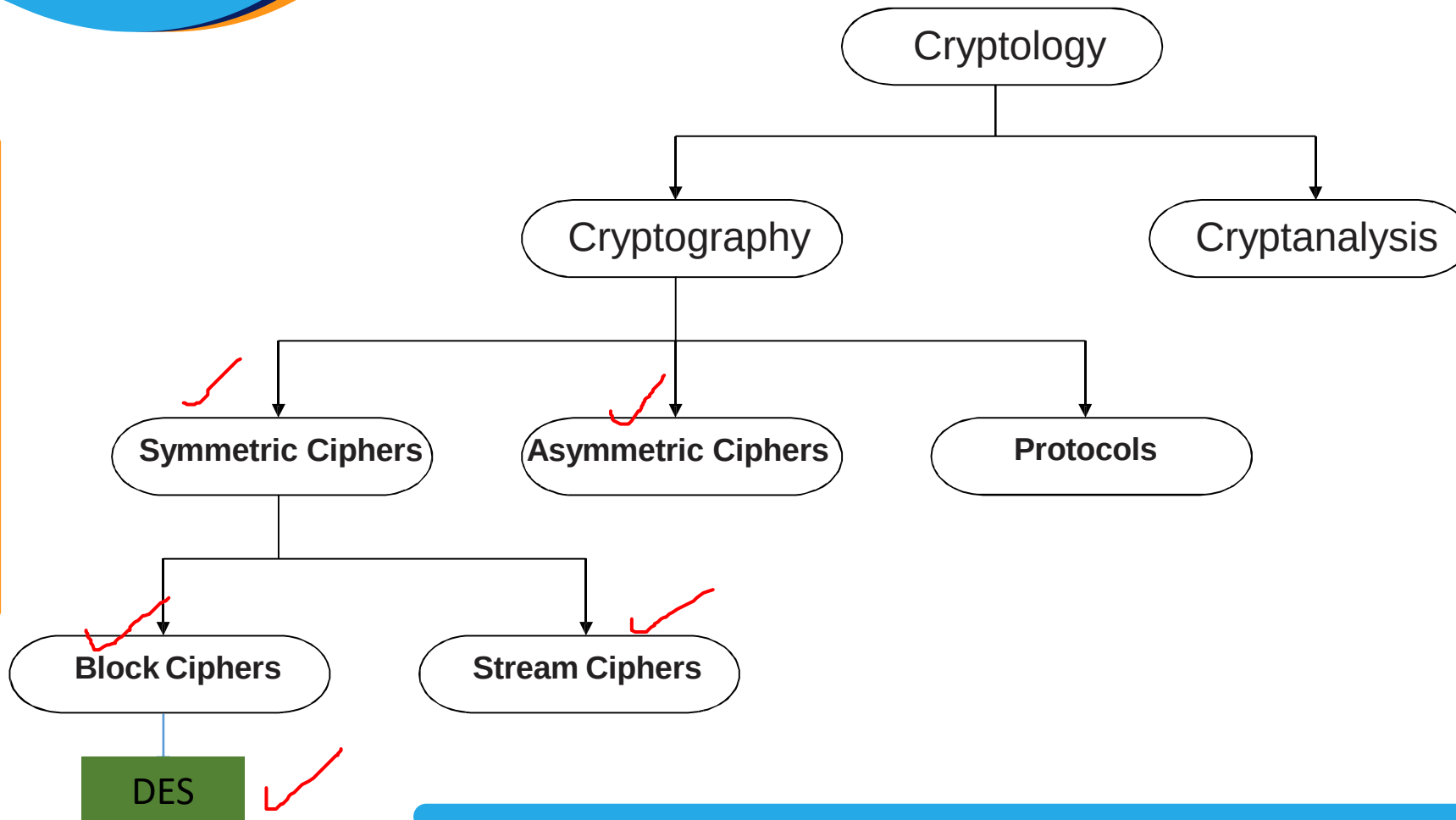
DES Algorithm

- Introduction to DES
- High-level description
- Confusion & Diffusion for secure block cipher
- Feistel Network
- DES Encryption Process
- Internal Structure of DES Algorithm

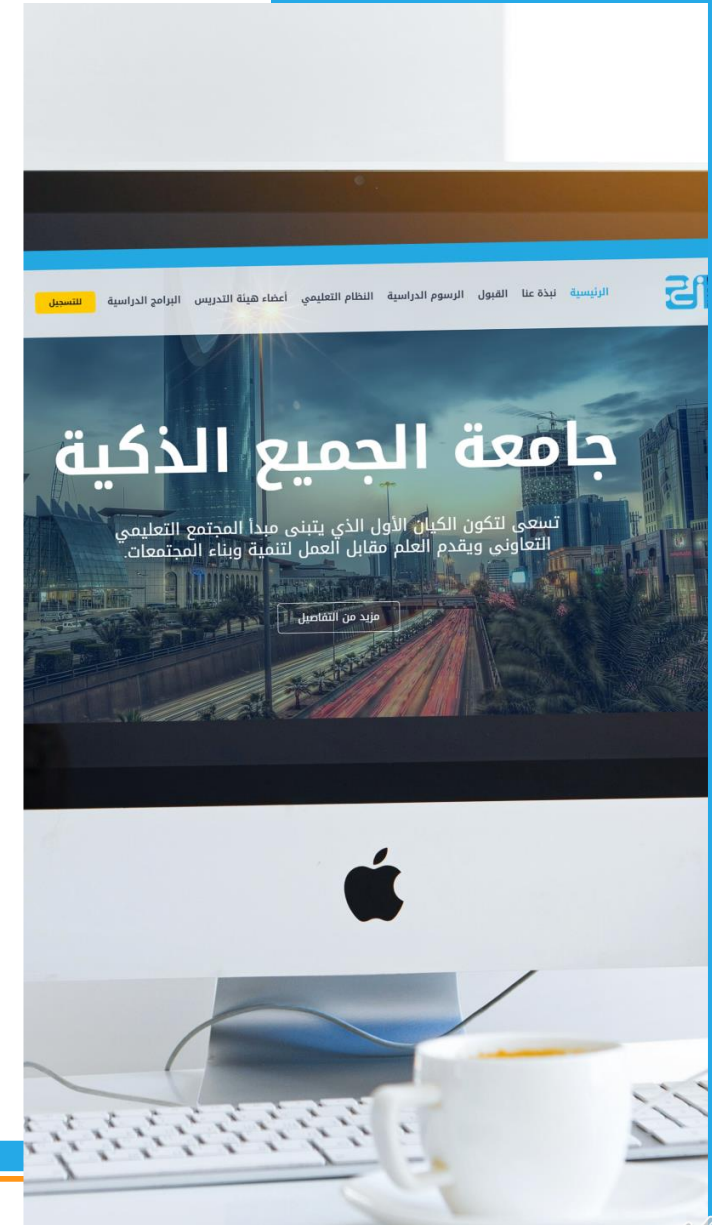


Module 02

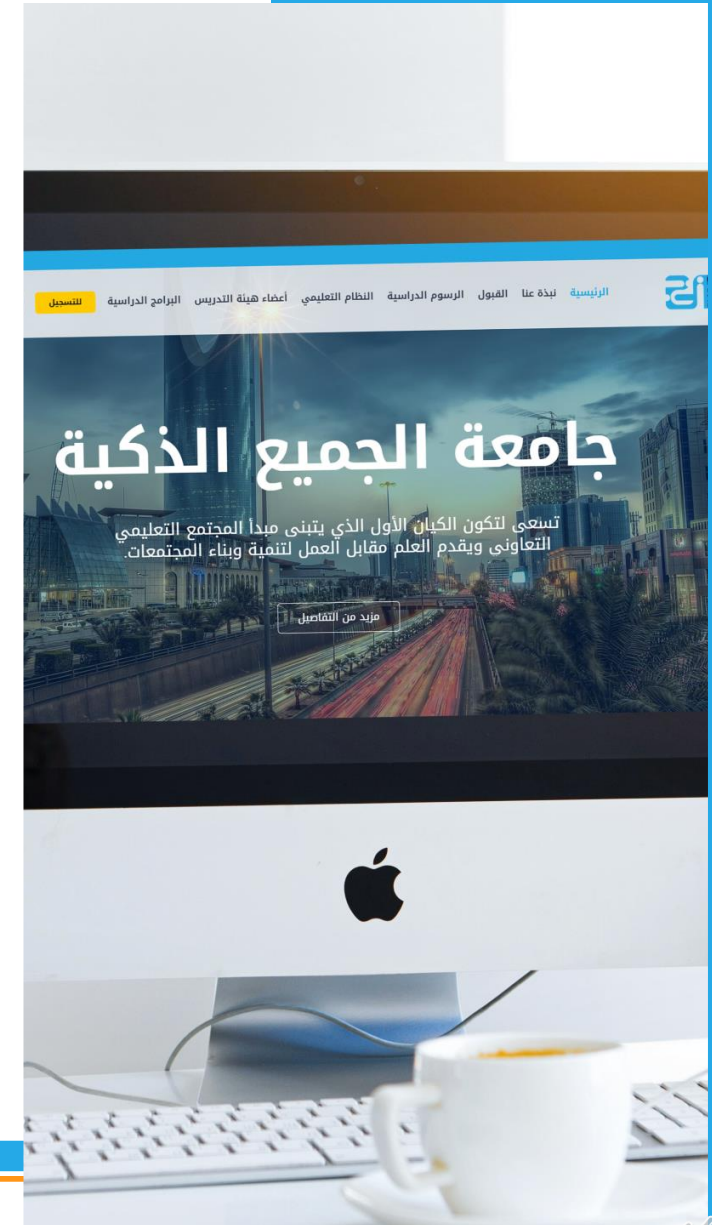
Classification of the field of Cryptology



- ❑ In 1972, the US National Bureau of Standards (NBS) called for proposals for a standardized cipher in the USA.
- ❑ The most promising proposal was submitted in 1974 by a team of cryptographers working at IBM organization.
- ❑ The algorithm IBM submitted was based on the cipher *Lucifer*.
- ❑ *Lucifer* is a *Feistel* cipher which encrypts blocks of 64 bits using a key size of 128 bits.

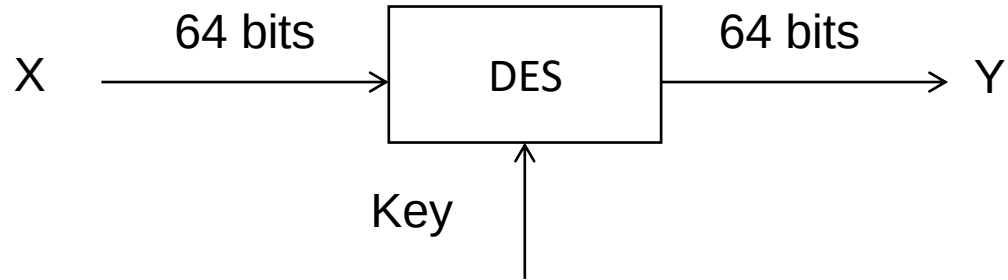


- ❑ The *National Security Agency (NSA)* convinced IBM to reduce the Lucifer key length of 128 bit to 56 bit, which made the cipher much more vulnerable to brute-force attacks.
- ❑ In 1977 the NBS finally released all specifications of the modified IBM cipher as the Data Encryption Standard.
- ❑ Finally, in 1999, DES was replaced by the Advanced Encryption Standard (AES).

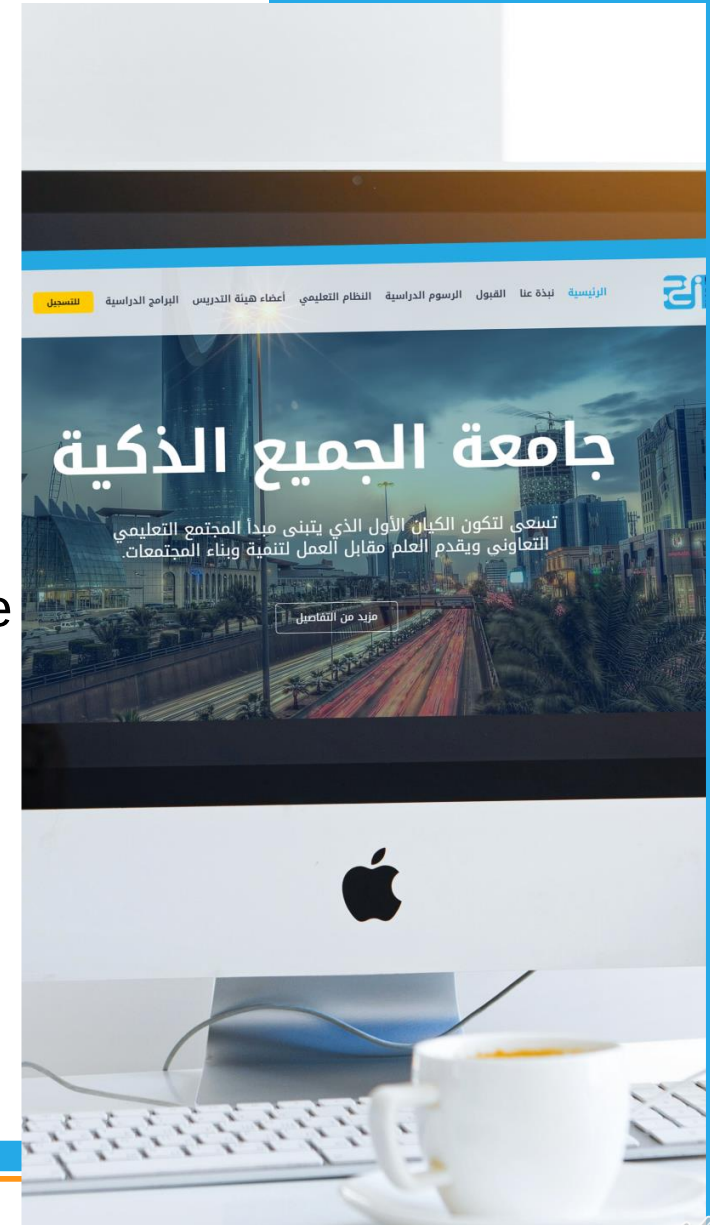


Module 02

High-level Description



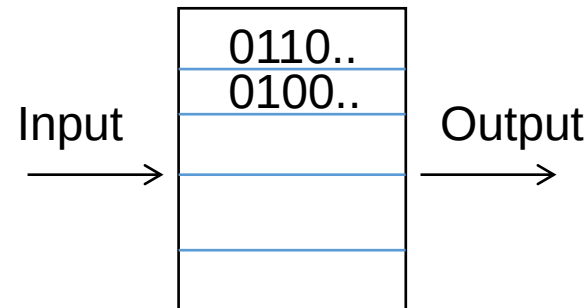
- ❑ DES is symmetric cryptosystem that uses a single key, which should be shared between both parties in the communication channel.
- ❑ DES encrypts a block of 64 bits at a time
- ❑ Key size = 56 bits



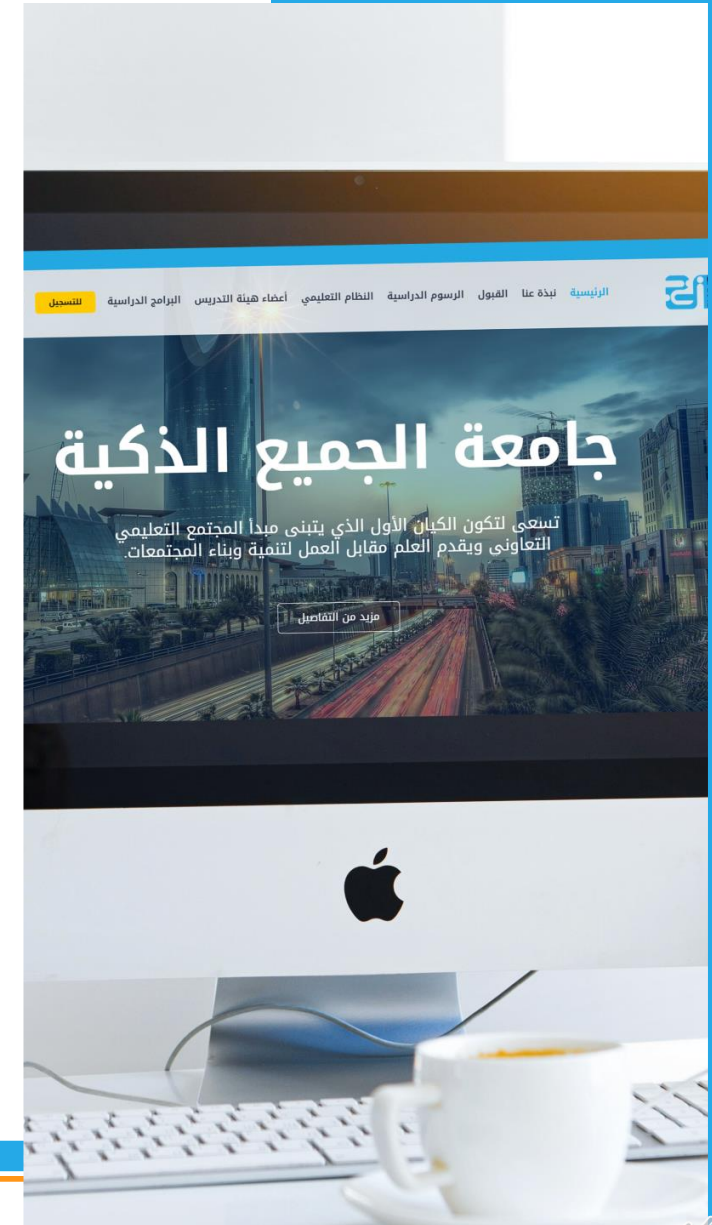
Module 02

Building a Block Cipher

- Generally, building a cipher is considered an art more than a science
 - According to Shannon (the developer of information theory), there are two primitive automatic operations that should be considered to build secure cipher:
1. **Confusion** is an encryption operation where the relationship between key and ciphertext is obscured. Example: the substitution table



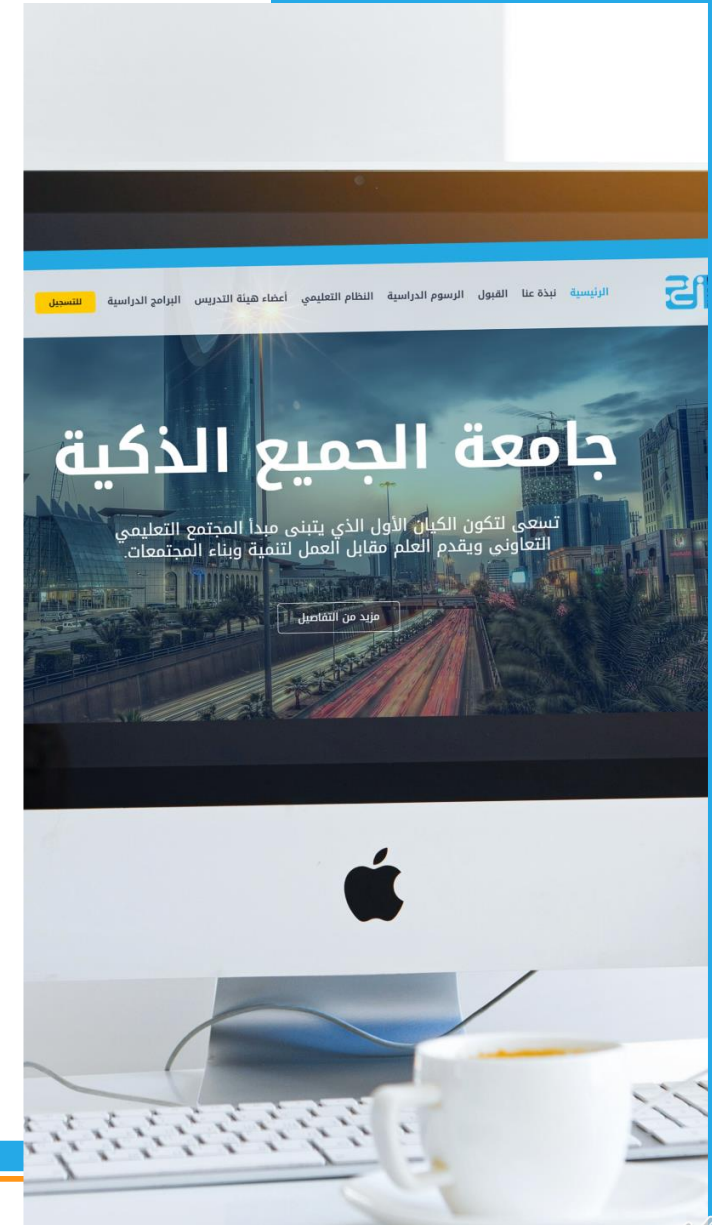
Confusion Element



2. Diffusion is an encryption operation where the influence of one plaintext symbol is spread over many ciphertext symbols with the goal of hiding statistical properties of the plaintext.

A simple diffusion element is the bit permutation, which is used frequently within DES.

DES combines confusion and diffusion many times to build a strong block cipher

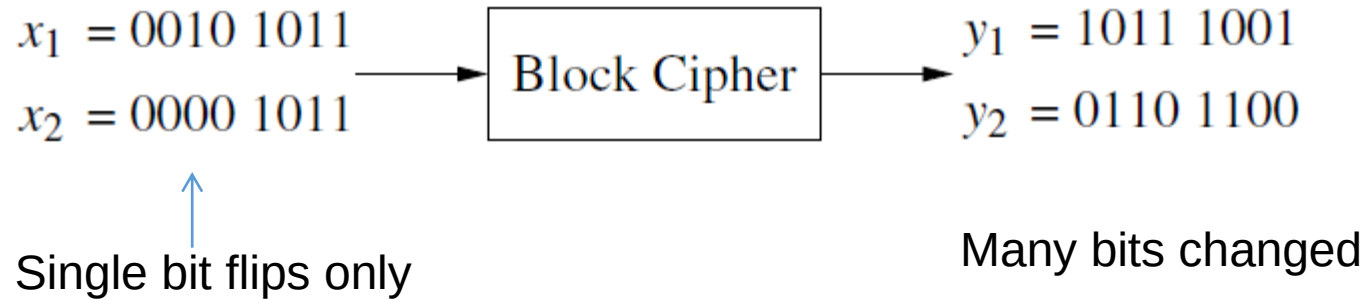


Module 02

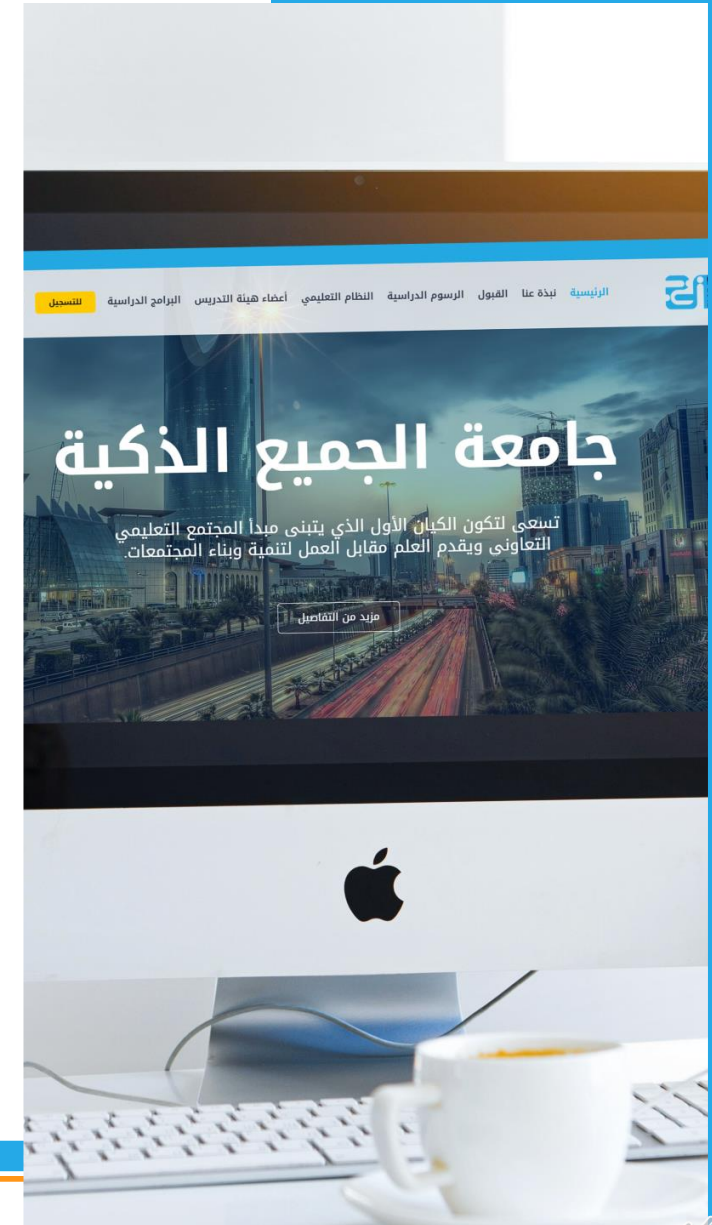
Building a Block Cipher

Diffusion: The influence of one plaintext symbol is spread over many ciphertext symbols.....

Example:



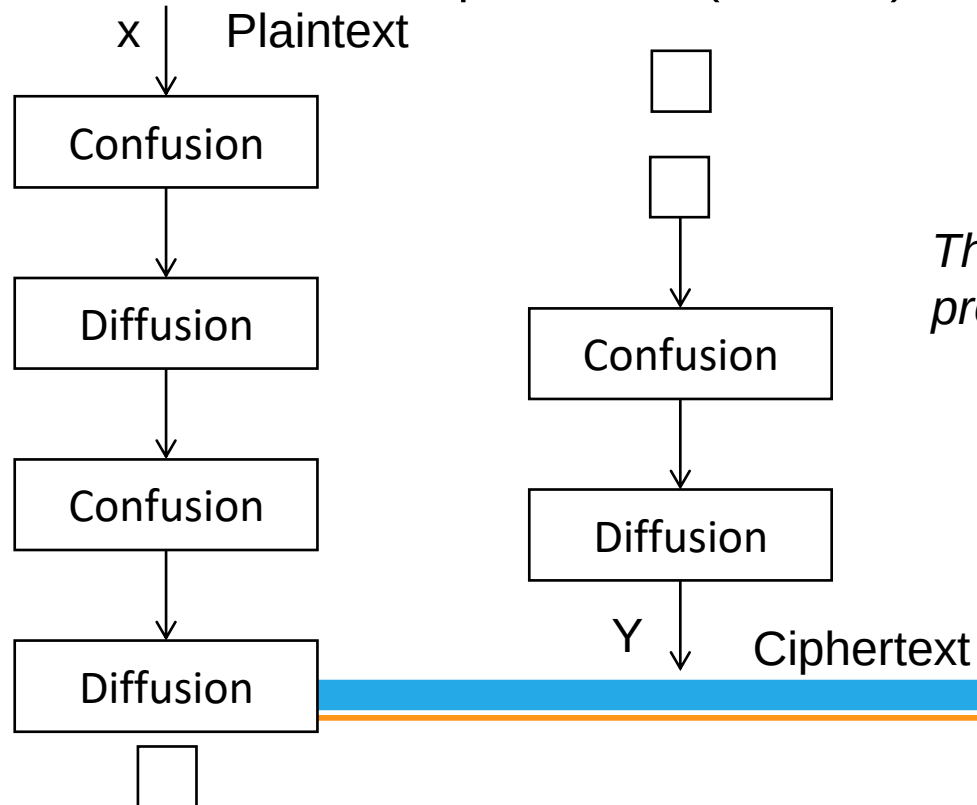
Note that although both plaintexts are approximately identical except for one bit.
The resulting cipher text is completely different



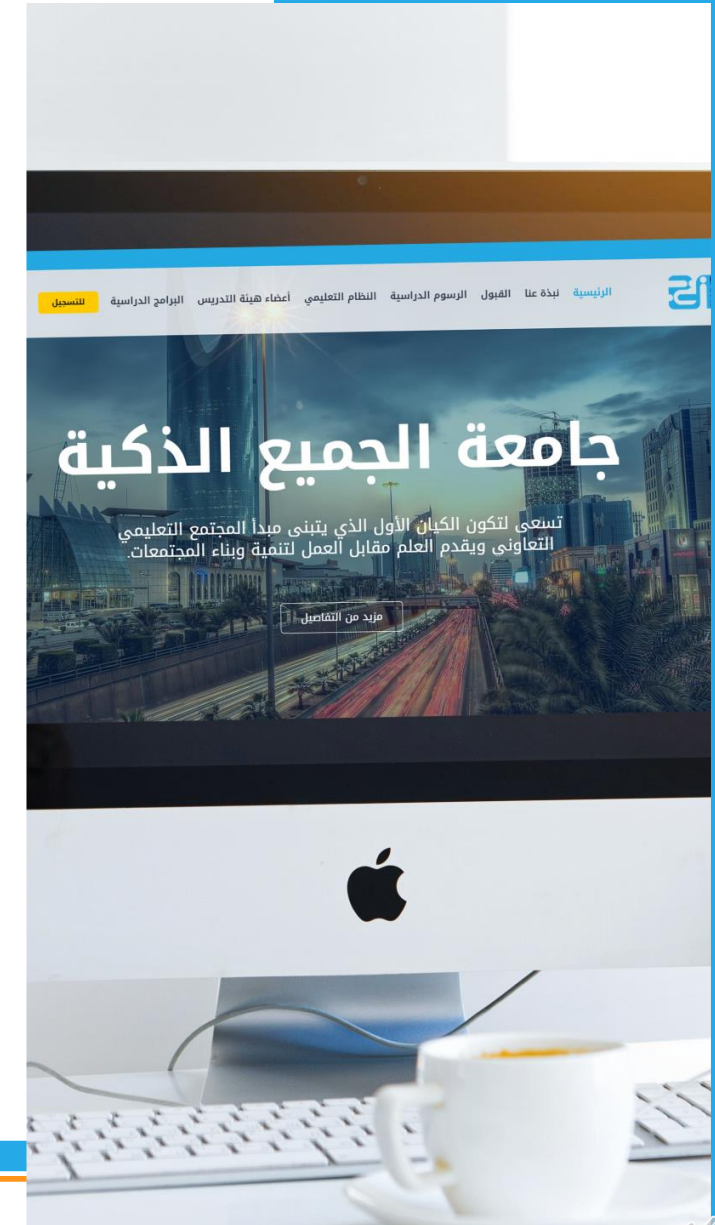
Module 02

Building a Block Cipher

1. First, we start with a plaintext
2. Perform a confusion operation (this could be done using a substitution table)
3. Do a diffusion (bit permutation) operation
4. Do not stop!! Perform the above processes (2, and 3) several time

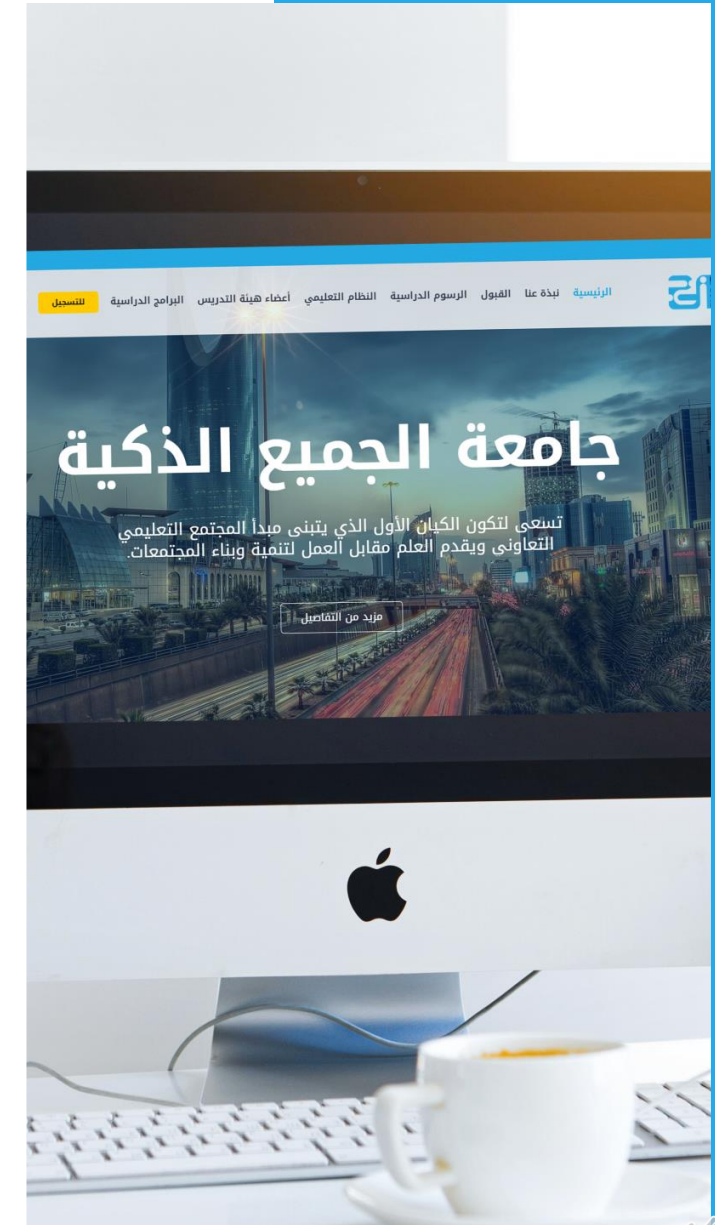
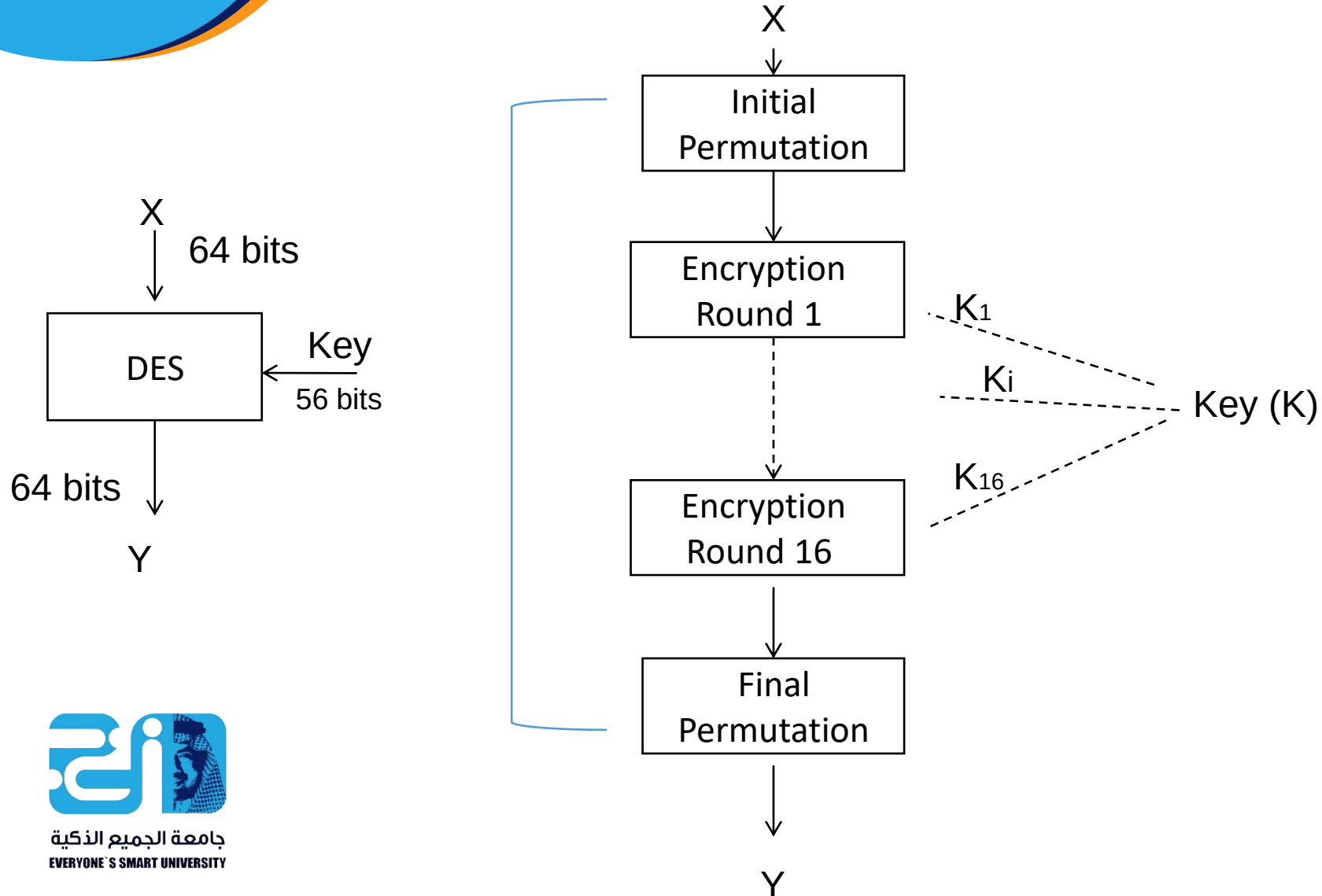


*This is called a
product cipher*

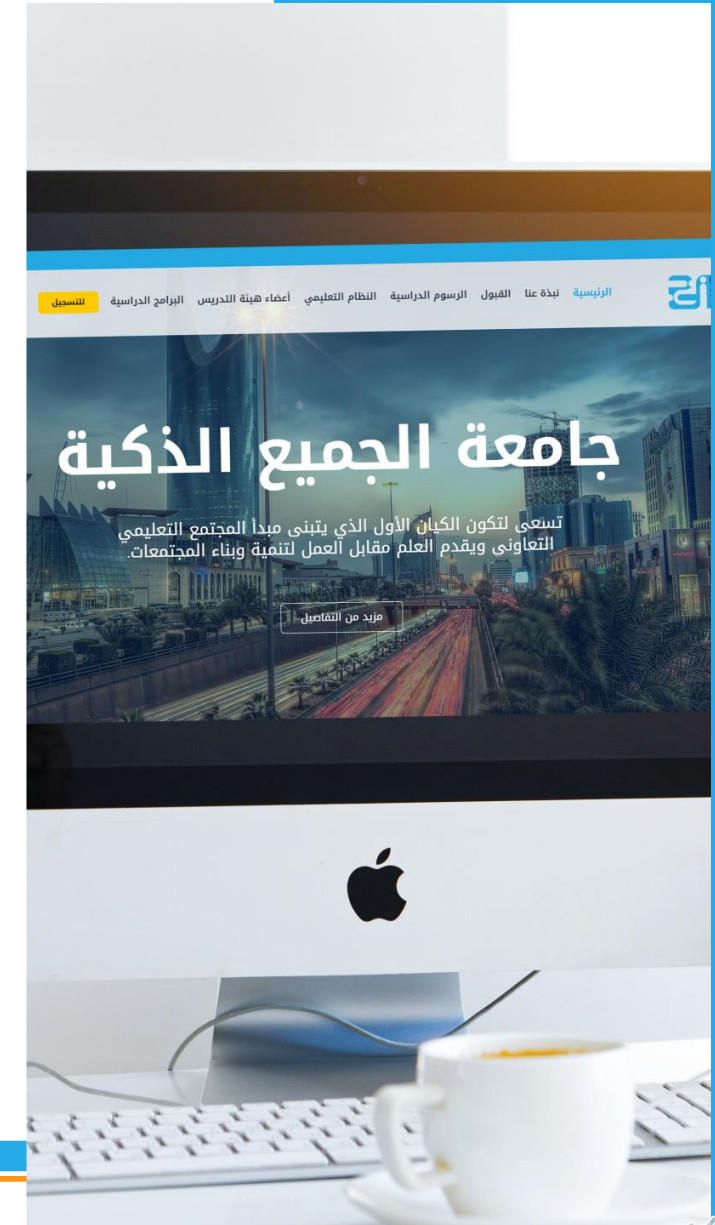


Module 02

Overview of DES Algorithm



- ❑ DES encrypts blocks of length of 64 bits with a key of size of 56
- DES is an iterative algorithm, which means that for each block of plaintext (64 bits), encryption is handled in 16 rounds which all perform the identical operation.
- ❑ In every (encryption/decryption) round a different sub-key is used
- ❑ All sub-keys k_i are derived from the main key k

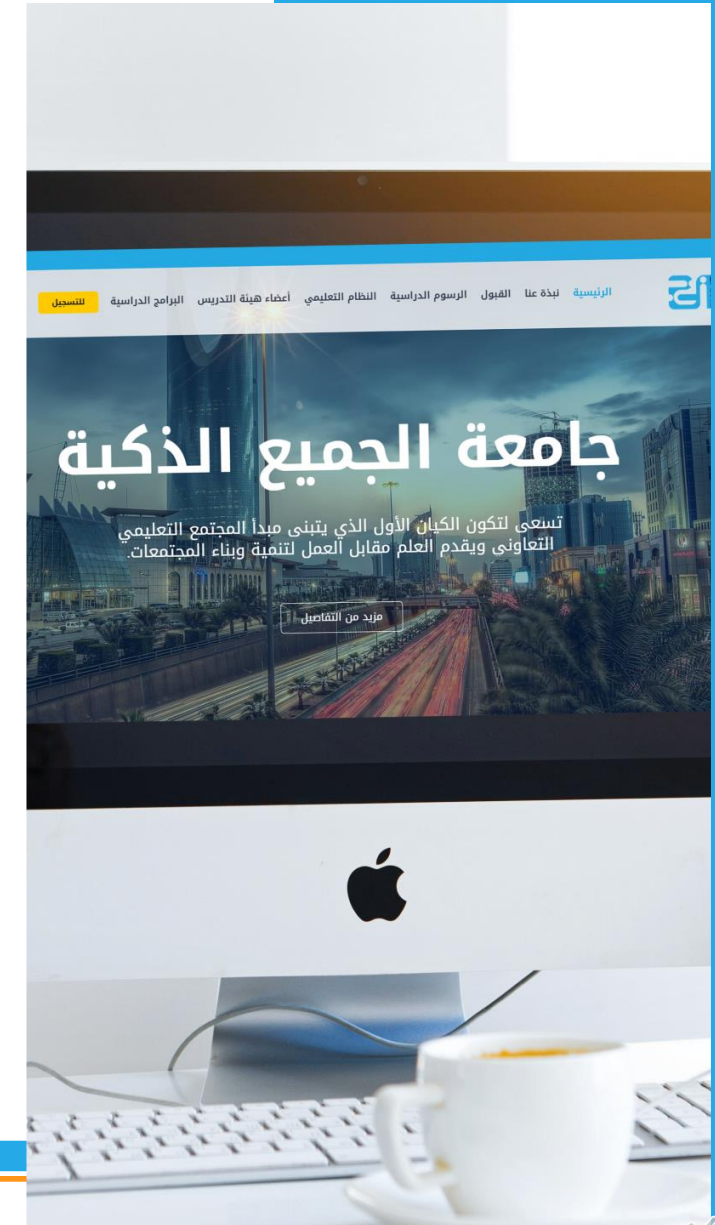


- ❑ DES and many modern ciphers use a Feistel network

(Cryptographic Structure)

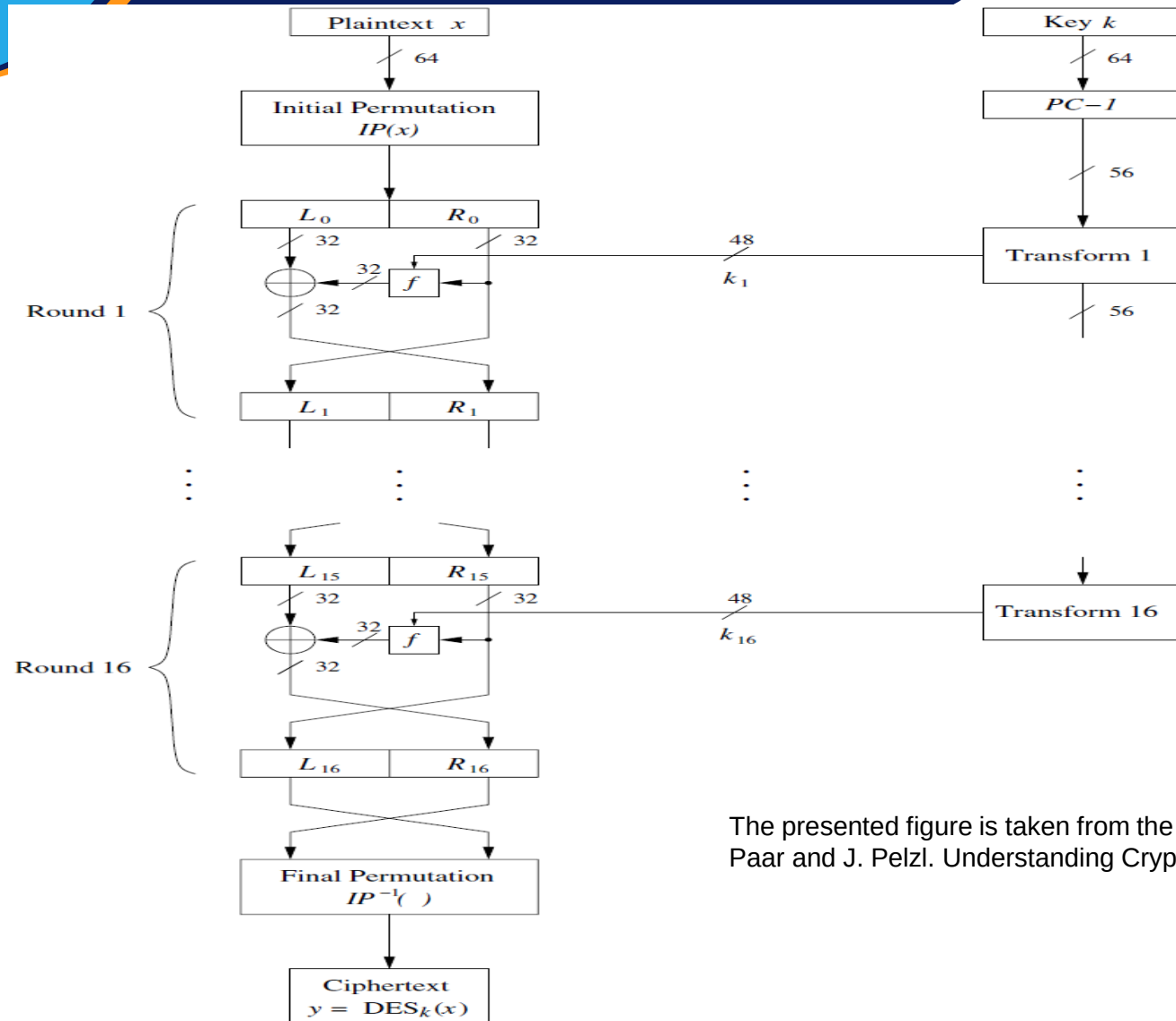
- ❑ In addition to its potential cryptographic strength, *the Feistel network uses* almost the same operation for encryption and decryption processes.

- ❑ The only difference between encryption and decryption processes is that decryption requires a reversed key schedule, compared to the sub-keys used in the encryption process



Module 02

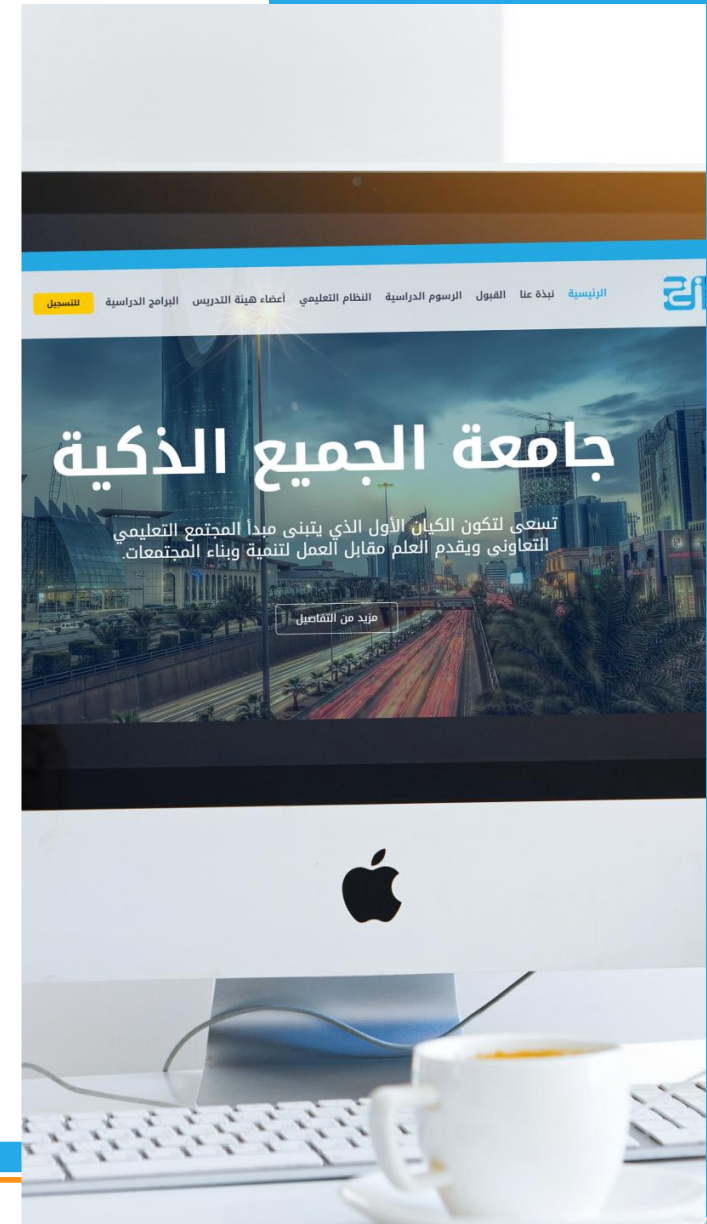
Overview of DES Algorithm



The presented figure is taken from the book: C. Paar and J. Pelzl. Understanding Cryptography

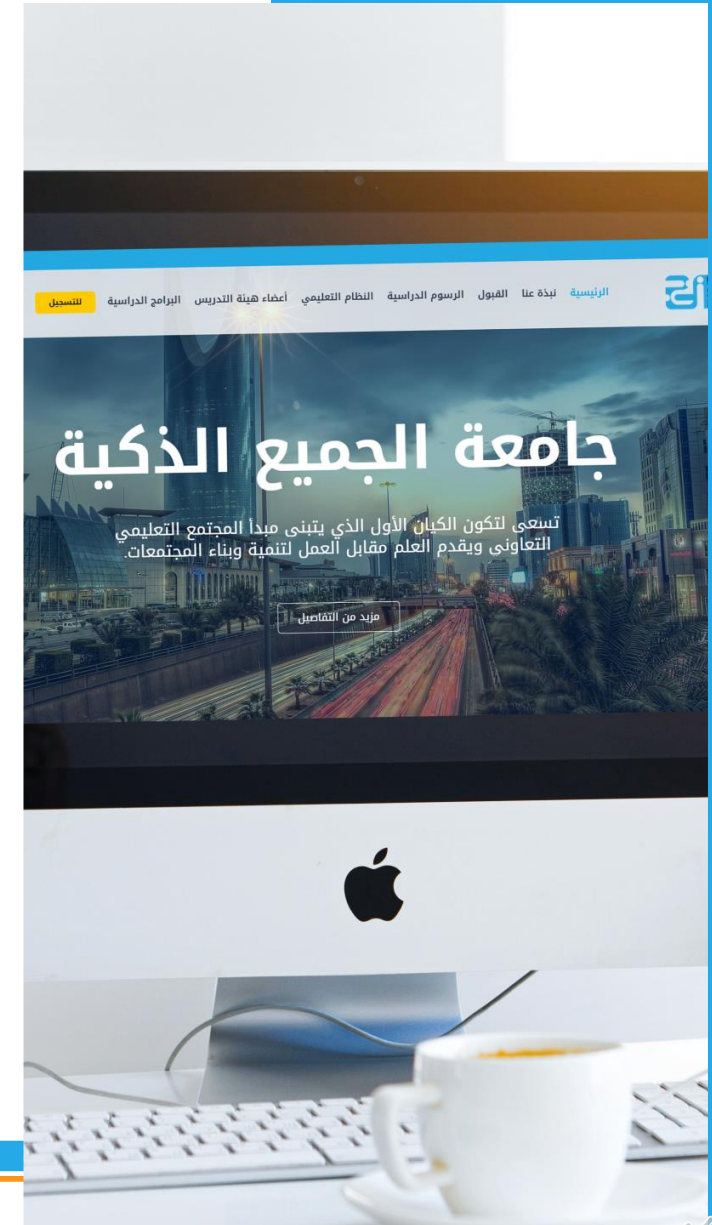


جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY



Notes:

- The Feistel structure consists of 16 rounds
- All sub-keys are derived from the main key k
- Each block of plaintext (input) and ciphertext (output) consists of 64 bits
- XOR is an essential operation in DES algorithm
- The actual encryption key is 56 bits length
- The final permutation IP^{-1} is the inverse of the initial permutation IP .



Module 02

Fiestel Structure of DES Algorithm

The plaintext is encrypted through several stages as following:

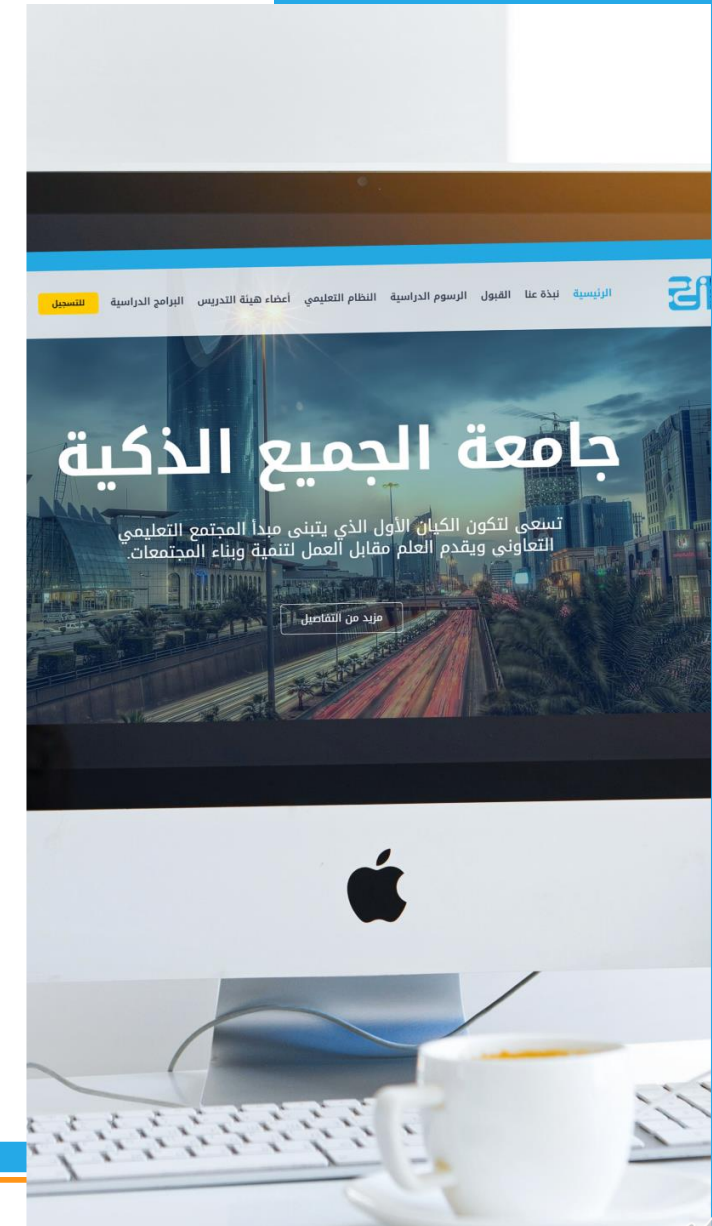
1. Initial bitwise permutation (*IP*) of a 64-bit plaintext x ,
2. The plaintext is then split into two halves L_0 and R_0 , which are the inputs to the Fiestel structure, each half includes 32 bits.
3. The right half R_i and the sub key k_i are fed into the function F
4. The output of the F function is XORed with the left 32-bit half L_i
5. The right and left half are swapped

The steps 3 – 5 repeat in the next round and can be expressed as:

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

$$R_i = L_{i-1} \oplus f(R_{i-1}, k_i), \text{ where } i = 1, \dots, 16$$

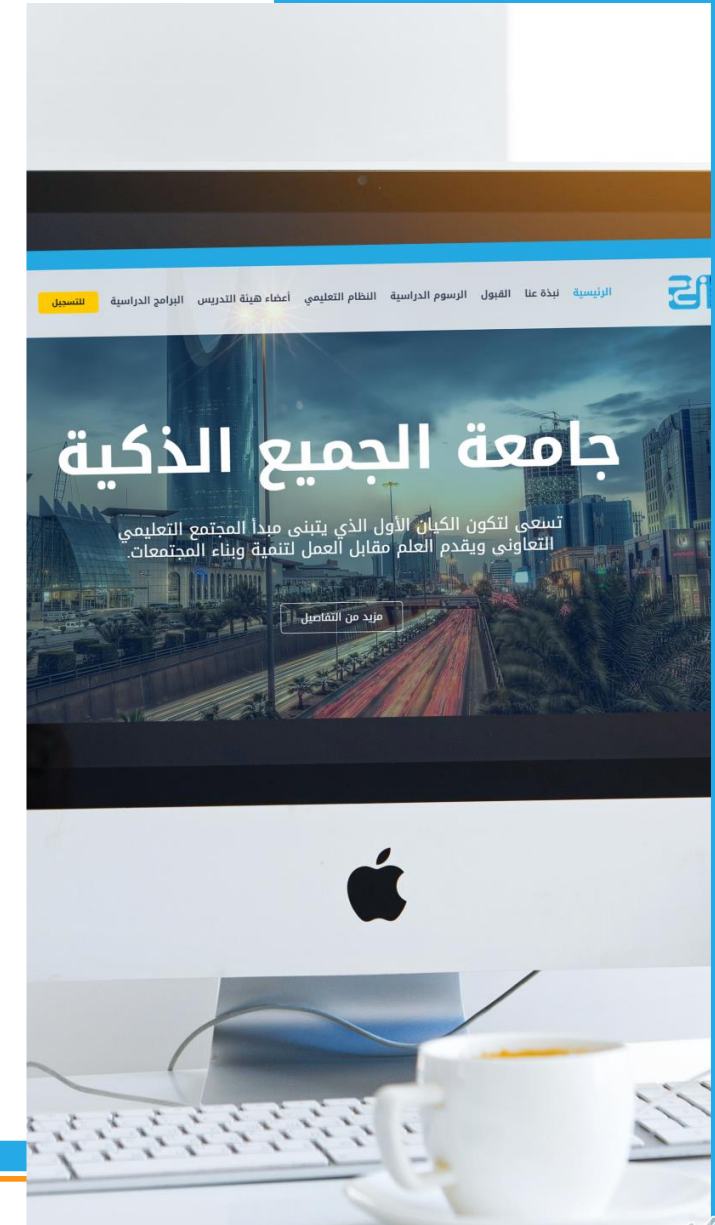
6. After round 16, the 32-bit halves L_{16} and R_{16} are swapped again
7. The final permutation IP^{-1} is the last operation of DES



Module 02

Fiestel Structure of DES Algorithm

- ❑ The Feistel structure really only encrypts /decrypts half of the input bits per each round.
- ❑ The left half is encrypted. The right half is copied to the next round unchanged. The right half is not encrypted with the f function.
- ❑ In order to better understand the role of the f -function, you can imagine it as a pseudorandom generator with the two input parameters R_{i-1} and k_i . The output of the pseudorandom generator is then used to encrypt the left half L_{i-1} with an XOR operation.
- ❑ Now, if the output of the f function is not predictable for an attacker, this results in a strong encryption method.

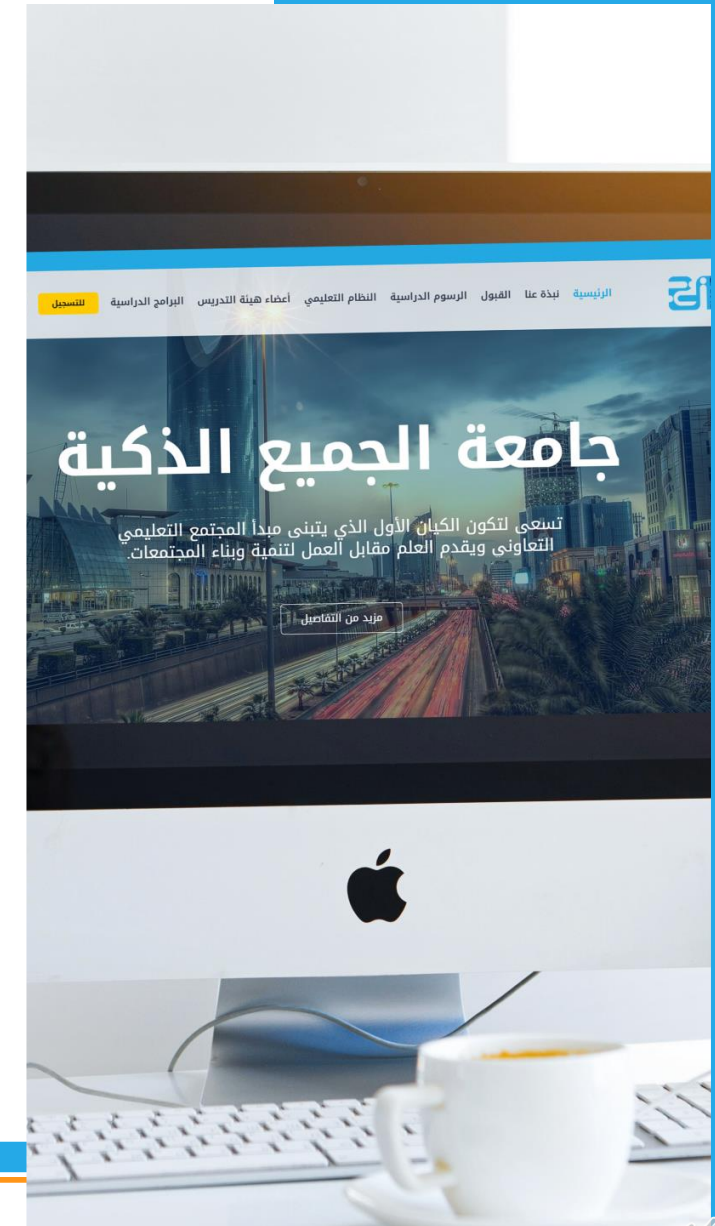


Module 02

Internal Structure of DES Algorithm

There are four building blocks in DES cipher

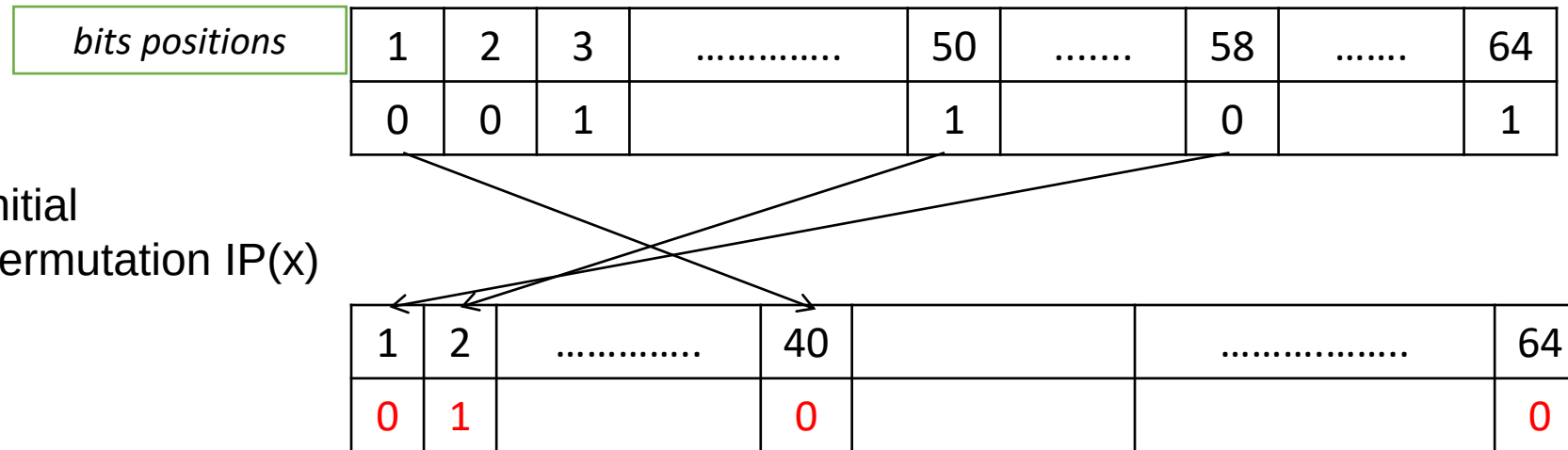
1. Initial and Final Permutations (IP and IP^{-1})
2. Actual DES Encryption Rounds
3. The F-Function
4. Key Schedule



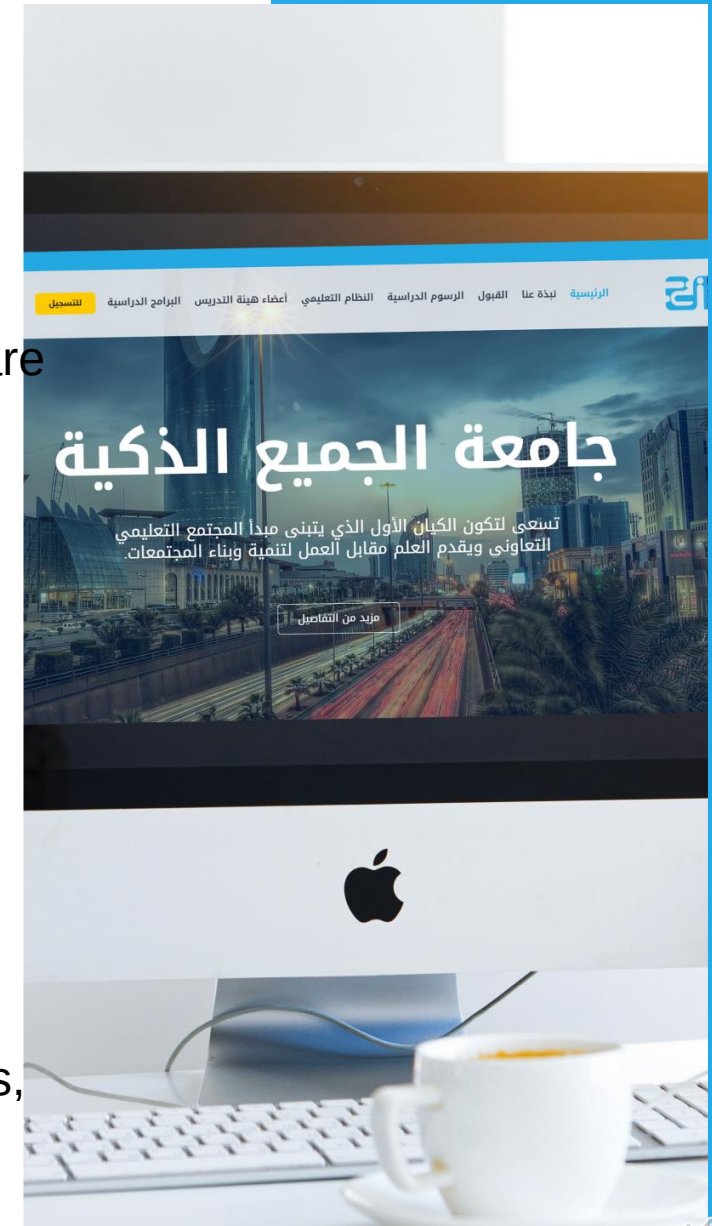
Module 02

Internal Structure (Initial & Final Permutation)

- ❑ The *initial permutation* IP and the *final permutation* IP^{-1} are bitwise permutations
- ❑ They can be easily implemented in hardware but are not particularly fast in software
- IP and IP^{-1} are random permutations.



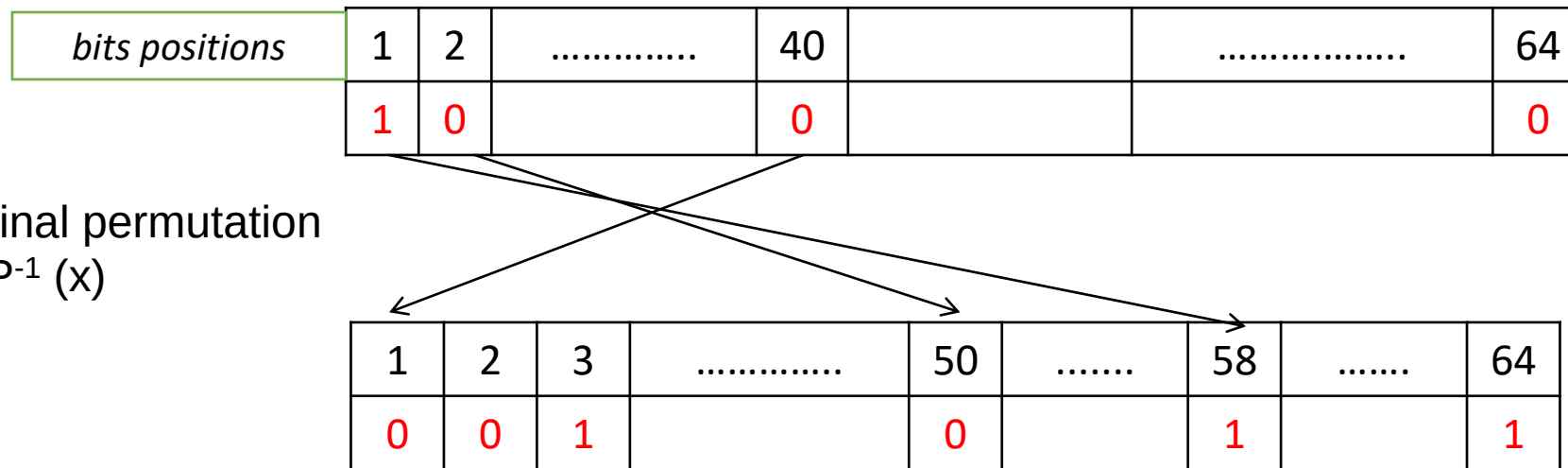
- ❑ Note that bit No. 1 is copied to the bit position No. 40 (*example*)
- It is a kind of random permutation. In hardware, you introduce a wire that copies bits, where in software implementation loop statements are required to implement the permutations



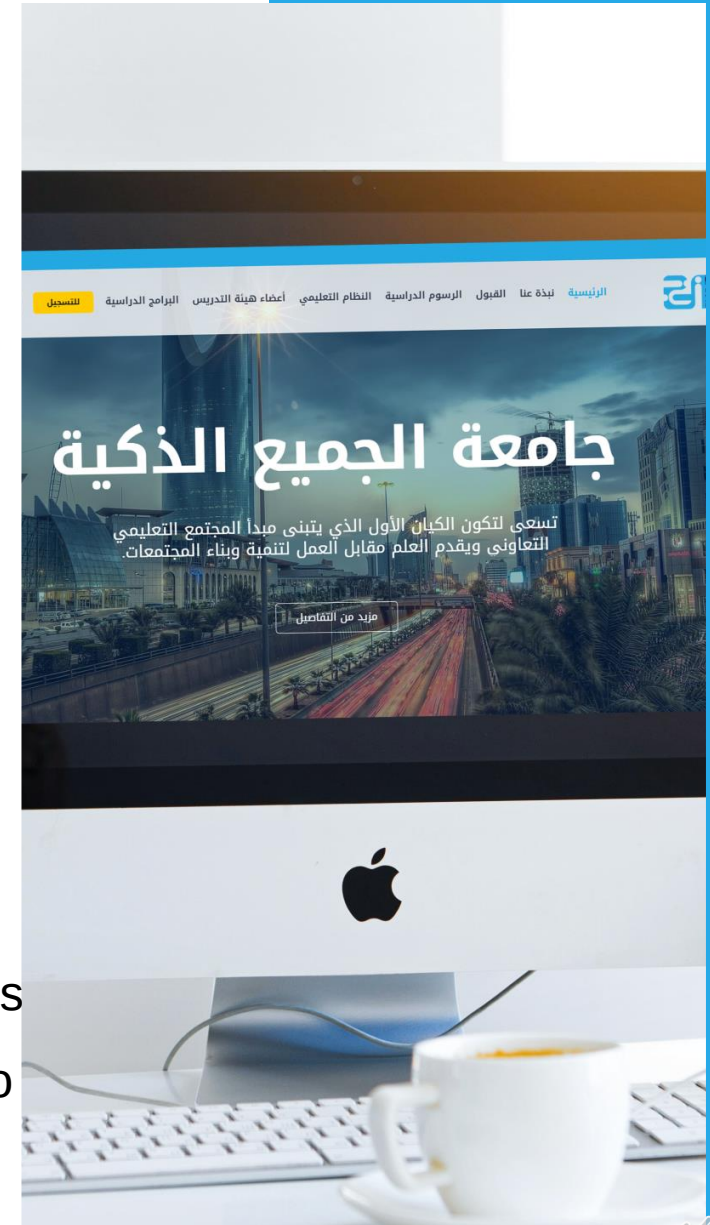
Module 02

Internal Structure (Initial & Final Permutation)

❑ The final permutation IP^{-1} performs the inverse operation of IP



❑ Note that the first bit (which was in position No. 50 before the initial permutation) is mapped (copied) to the bit position No. 50 again, where the second bit is mapped to the position No. 58



Module 02

Internal Structure (Initial & Final 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

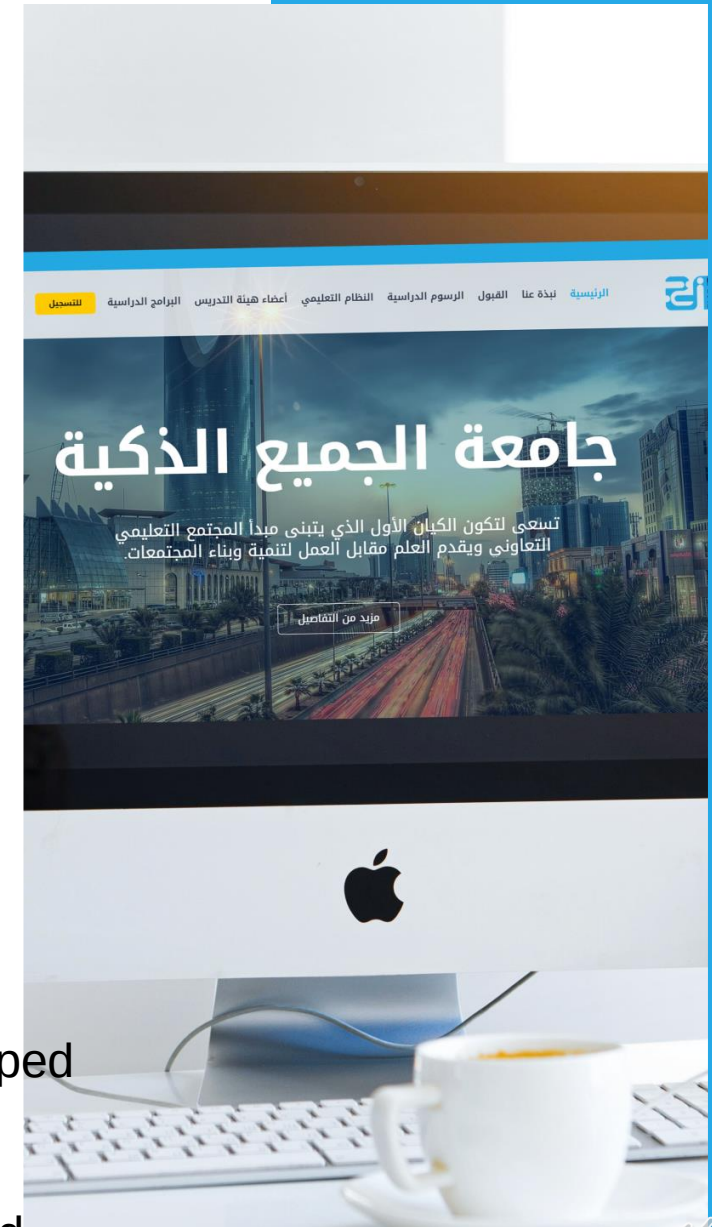
Table1: IP Transformation

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

Table٢: IP^{-1} Transformation

■Note that input bit No 58 is mapped to output position 1, and input bit No 50 is mapped to the second output position.

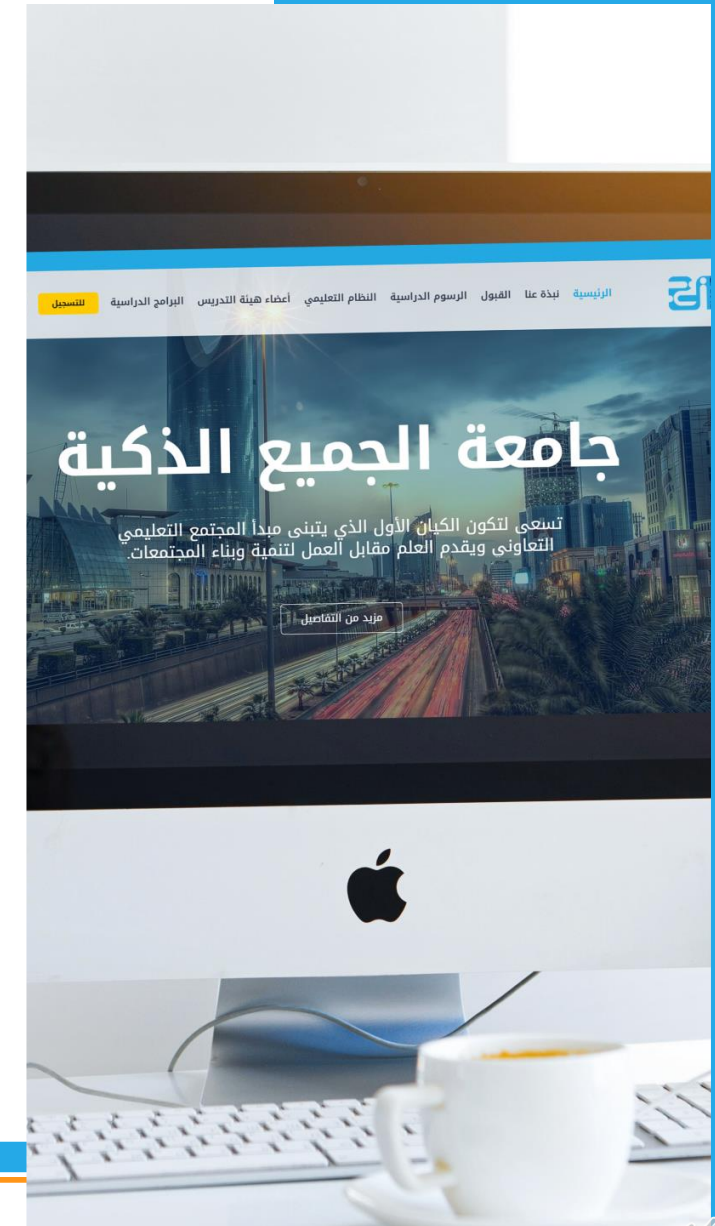
■Note that IP^{-1} performs the inverse operation of IP (for example: bit No. 1 is mapped again to the bit position No. 58)



Module 02

Internal Structure (Initial & Final Permutation)

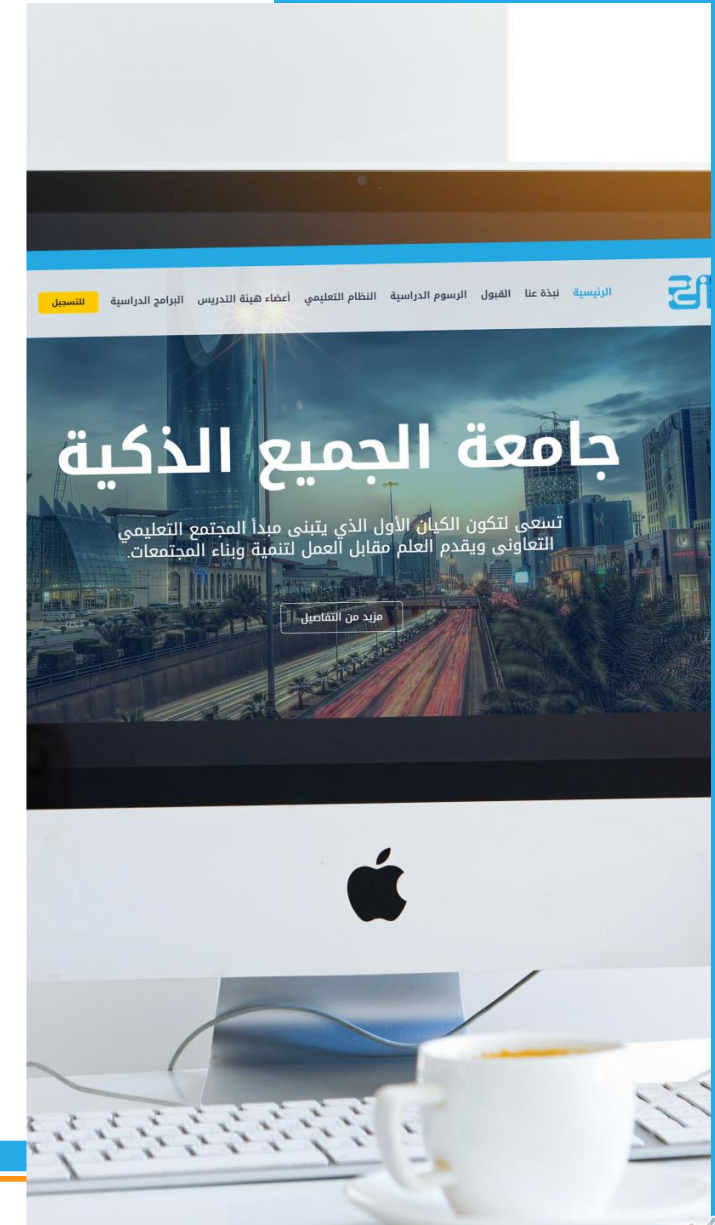
- ❑ **Question1:** Can we rely on permutations to provide security for DES algorithm since it is one of the encryption techniques?
- ❑ **Answer: NO !!** Both permutations do not increase the security of DES
- ❑ The permutations details are publicly known, and the tables in previous slide are published
- ❑ The attacker can easily break the cryptographic algorithm by simply reversing the permutation operations



Module 02

Internal Structure (Initial & Final Permutation)

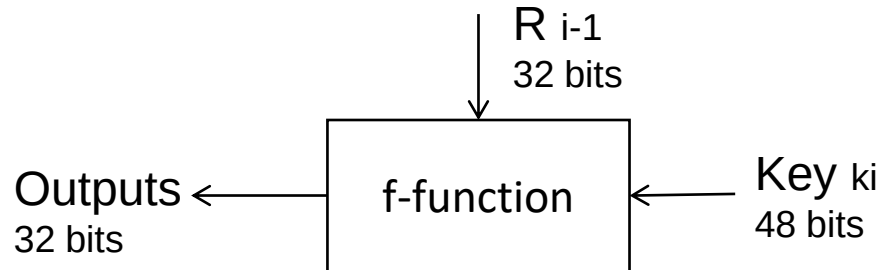
- ❑ Therefore, the initial and final permutations have nothing to do with cryptography and security
- ❑ It has been mentioned in the literature, that the main purpose of IP and IP⁻¹ is to address electrical engineering issues that were encountered when developing DES
- ❑ Permutation is fast and simple in hardware, but relatively slow in software implementations



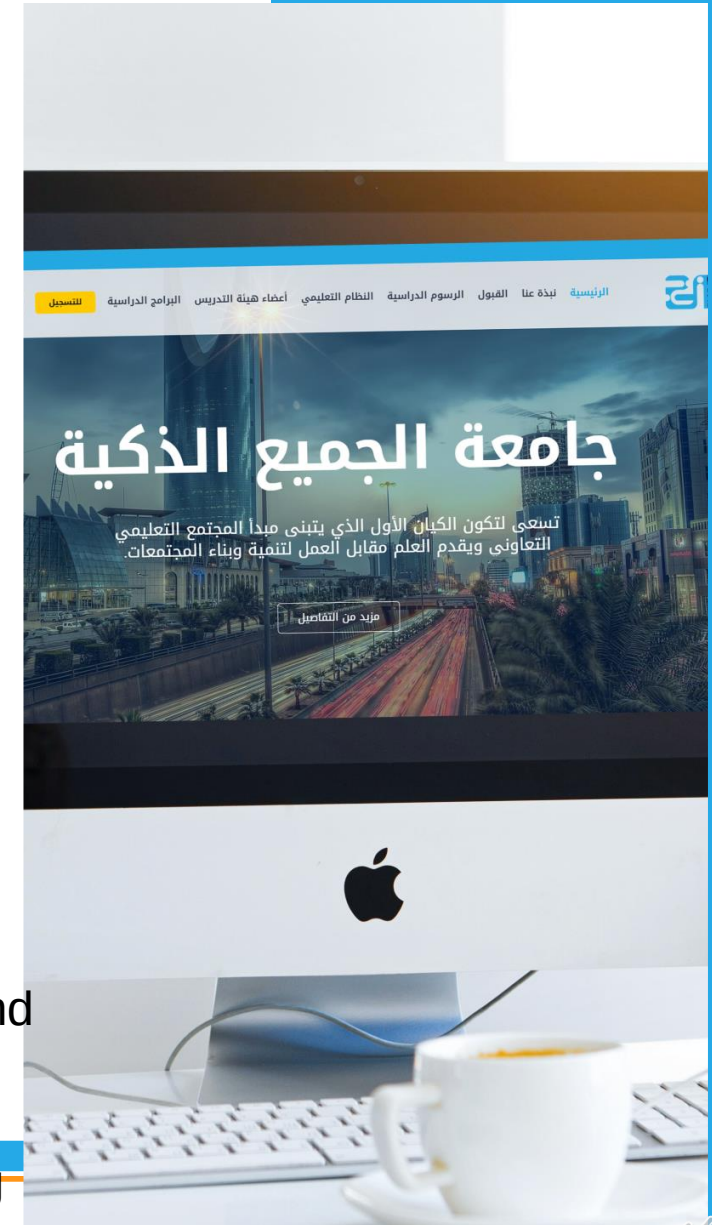
Module 02

Internal Structure of DES (The f –Function)

- ❑ The f-function plays a crucial role for the security of DES algorithm
- ❑ The actual encryption of DES algorithm happens in the f-function
- ❑ The f-function provides the two basic properties of ciphers, i.e., confusion and diffusion



- ❑ The f-function should be designed carefully to ensure that it generates efficient random numbers
- ❑ Note that f-function takes the right half R_{i-1} of the output of the previous round and the current round key k_i as input.
- ❑ The output of the f -function is used as an XOR-mask (*key stream*) for encrypting the left half input bits L_{i-1} .



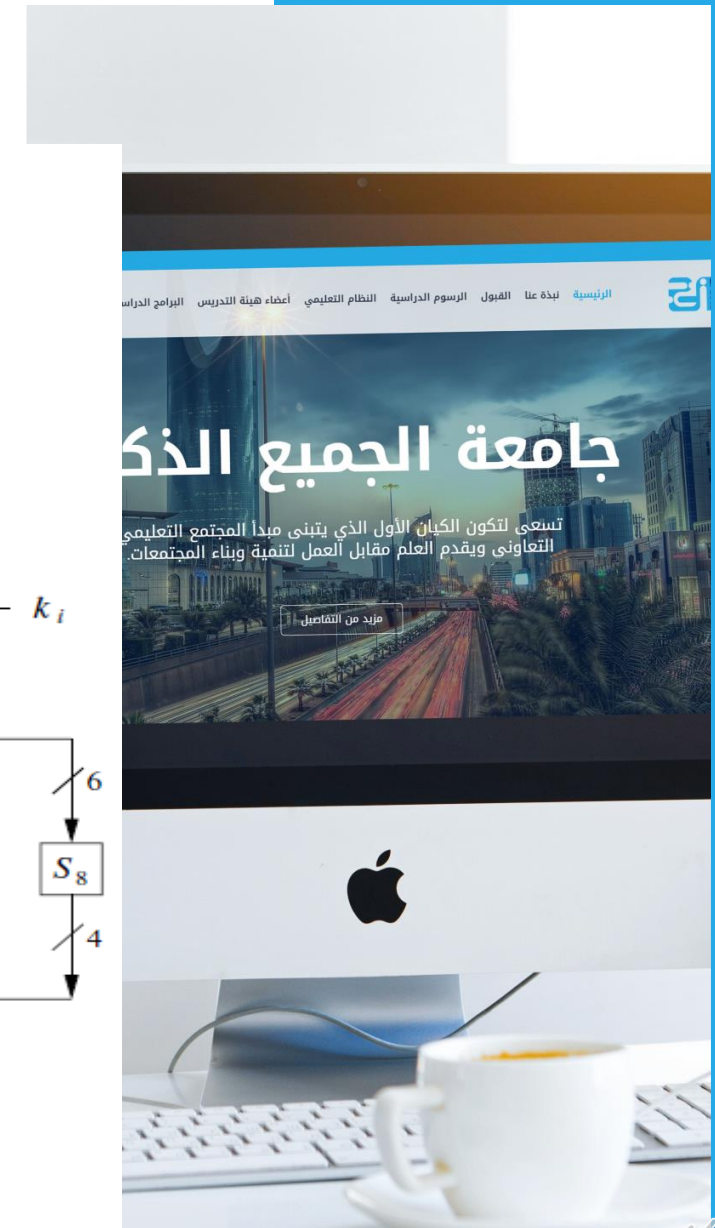
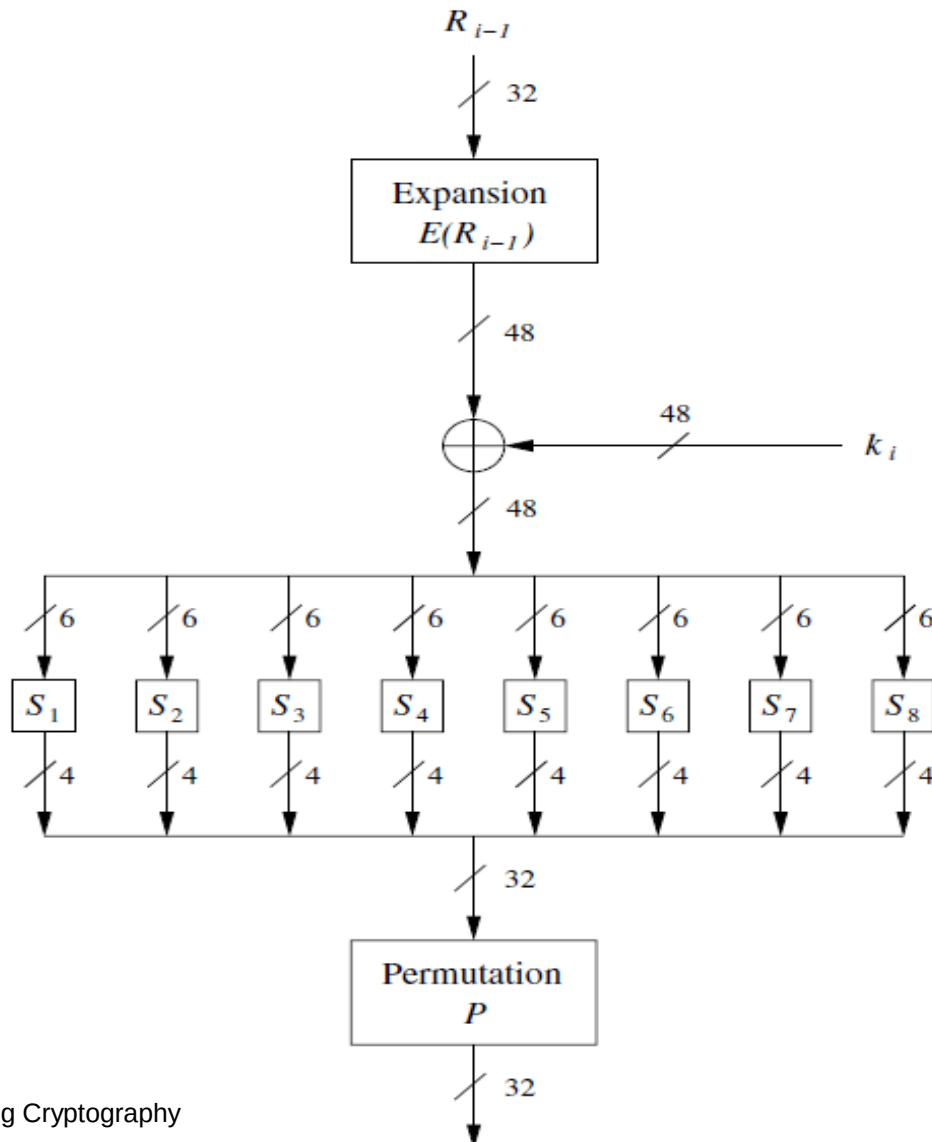
Module 02

Internal Structure of DES (The f –Function)

❑ Note that there are four main components

the f-function:

1. Expansion Box (E-box)
2. XOR operation
3. S-boxes
4. The permutation within the f-function



Module 02

Internal Structure of DES (The f –Function)

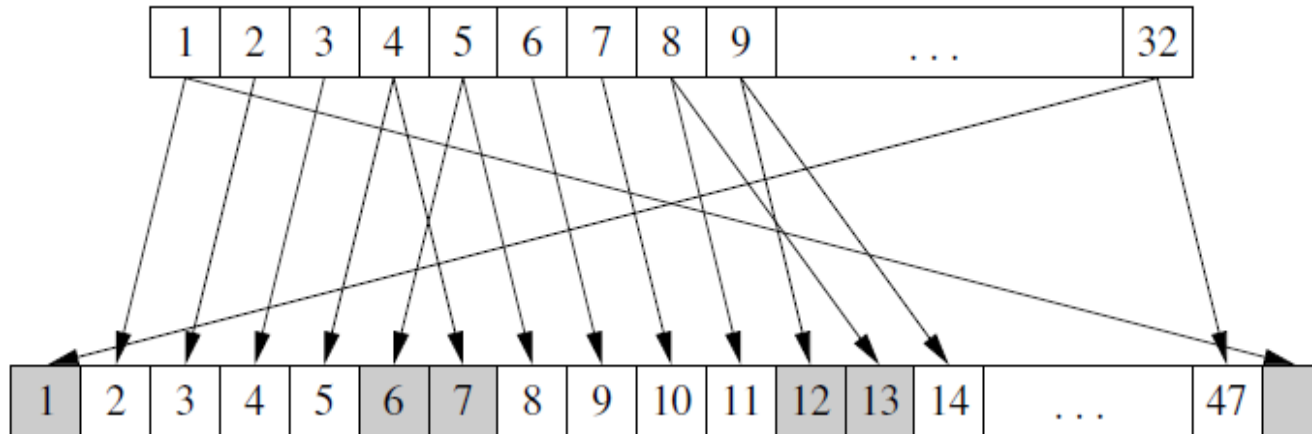
❑ The major operations of the f-function can be described as follows:

1. The 32-bit input is expanded to 48 bits, which can be done by partitioning the input into eight 4-bit blocks and by expanding each block to 6 bits

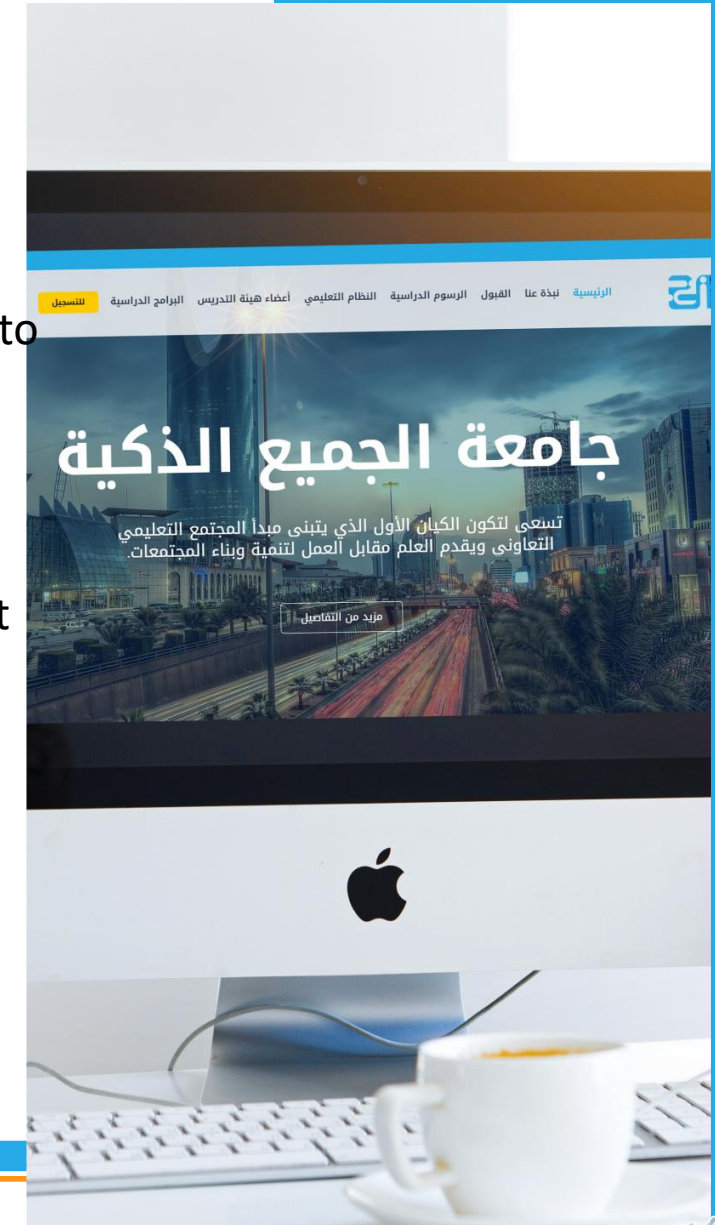
This happens in the E-box, which can be considered a special type of permutation.

It can be seen from the following figure, which presents an example for the bit swaps, that first block consists of the bits (1.2.3.4). the second one of (5.6.7.8). etc

bits positions



bits positions



Module 02

Internal Structure of DES (The f –Function)

❑ The following table shows that exactly 16 of the 32 input bits appear twice in the output.

❑ An input bit never appears twice in the same 6-bit output block.

Therefore, the expansion box increases the diffusion behavior of DES; certain input bits influence two different output locations.

❑ Then, the 48-bit result of the expansion process is XORed with the round key k_i (48-bits). The result will be eight blocks each one of them consists of 6 bits

❑ Next, the eight 6-bit blocks are fed into eight different substitution boxes (also called S-boxes)

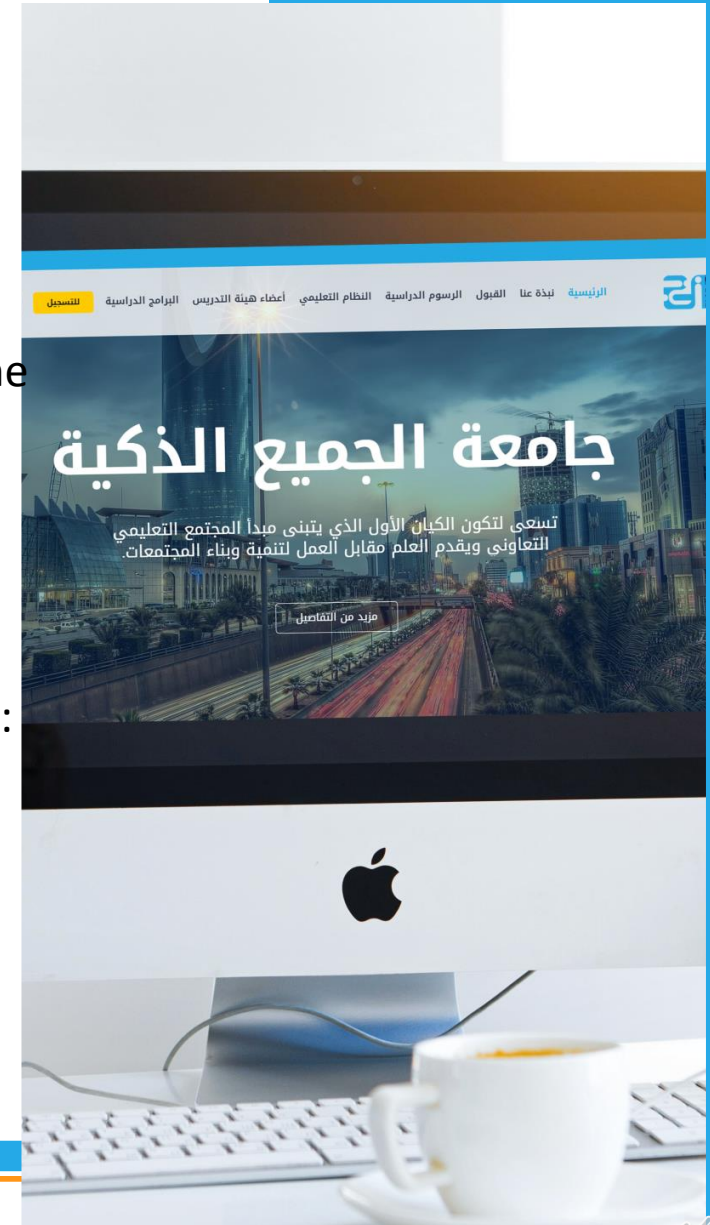
E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1



Module 02

Internal Structure of DES (The f –Function)

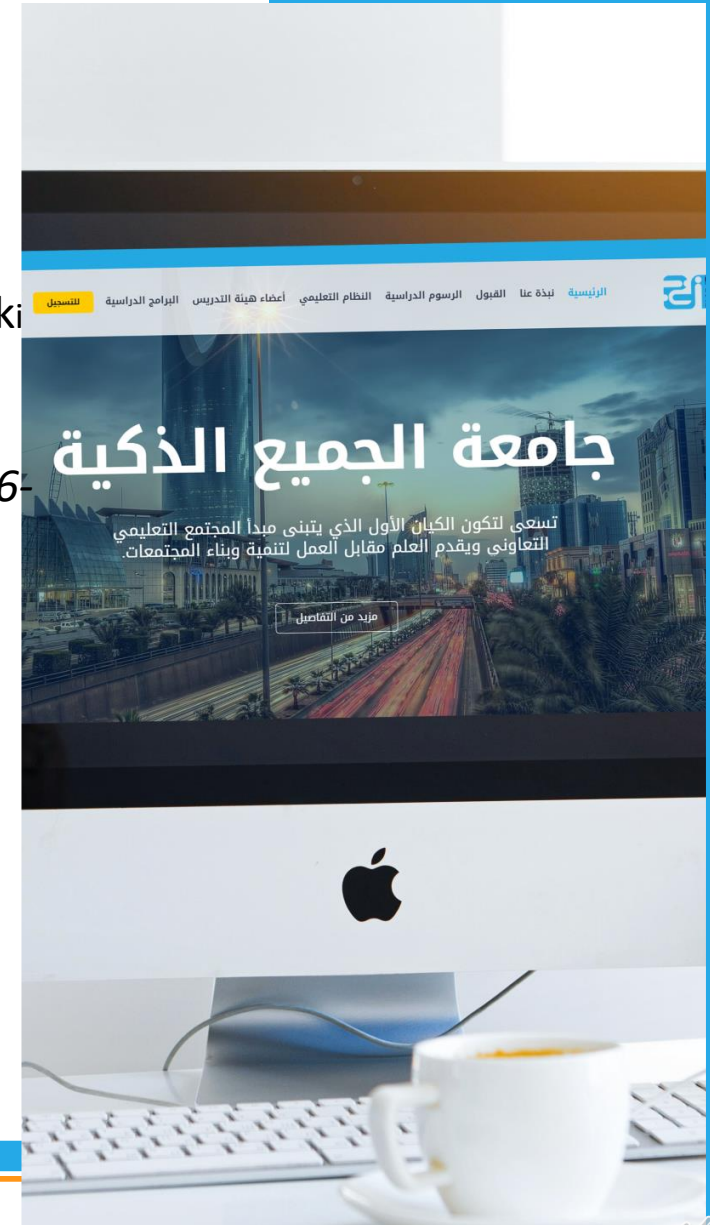
- ❑ It is worth mentioning here that E-box provides the **diffusion** property
- ❑ It can be noticed that some bits are repeated twice in the plaintext block (R_{i-1}), while the other bits occur only once
- ❑ In particular, 16 bits happen once, and 16 bits happen twice, so the total number of bits after the expansion $\rightarrow 16 + (16+16) = 48$ bits.
- ❑ By performing the expansion function, it can be ensured that if one bit flips (for example: bit in position NO. 1, this will change bits in different outputs locations).



Module 02

Internal Structure of DES (The f –Function)

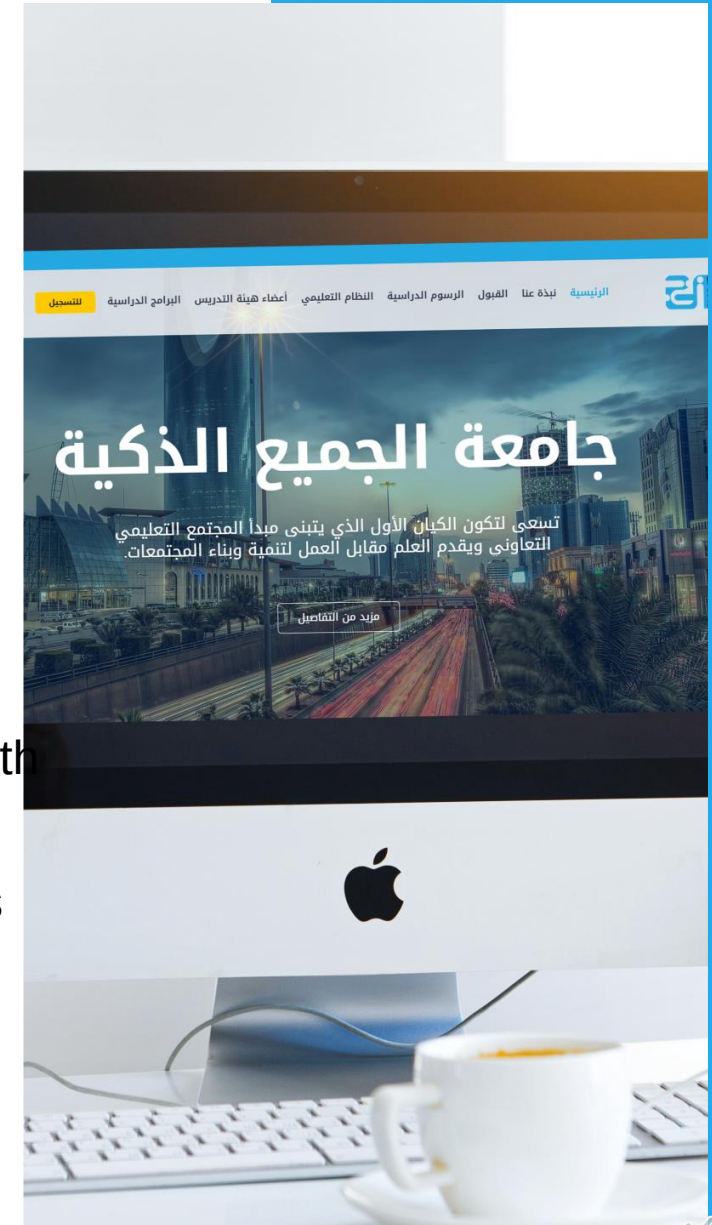
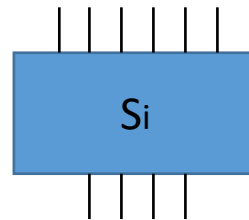
- ❑ At the next stage of the f-function, an XOR operation is performed between the sub key K_i and the expanded bits of R_{i-1} (each of the two blocks contain 48 bits now)
- ❑ The result of the XOR operation is the input to the s-boxes (*48 bits distributed into eight 6-bits blocks*)



Module 02

Internal Structure of DES (S-box)

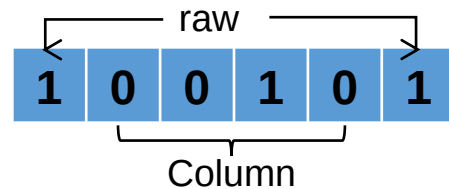
- ❑ S-boxes represent the heart of the DES algorithm
- ❑ S-boxes provide an important property of cryptographic algorithm, which is the **confusion** property.
- ❑ There are eight S-boxes (*also called substitution boxes*) in DES cipher
- ❑ There is a block of 6-bits inputs to each S_i box, and a block of 4-bits outputs.
- ❑ Each S-box represents a lookup table that maps a 6-bit input to a 4-bit output.
- ❑ Each S-box contains $2^6 = 64$ entries, which are typically represented by a table with 16 columns and 4 rows
- ❑ Each entry (data item) in a S-box is a 4-bit value. So, the input is a block of 6-bits and the output is a block of 4-bits



Module 02

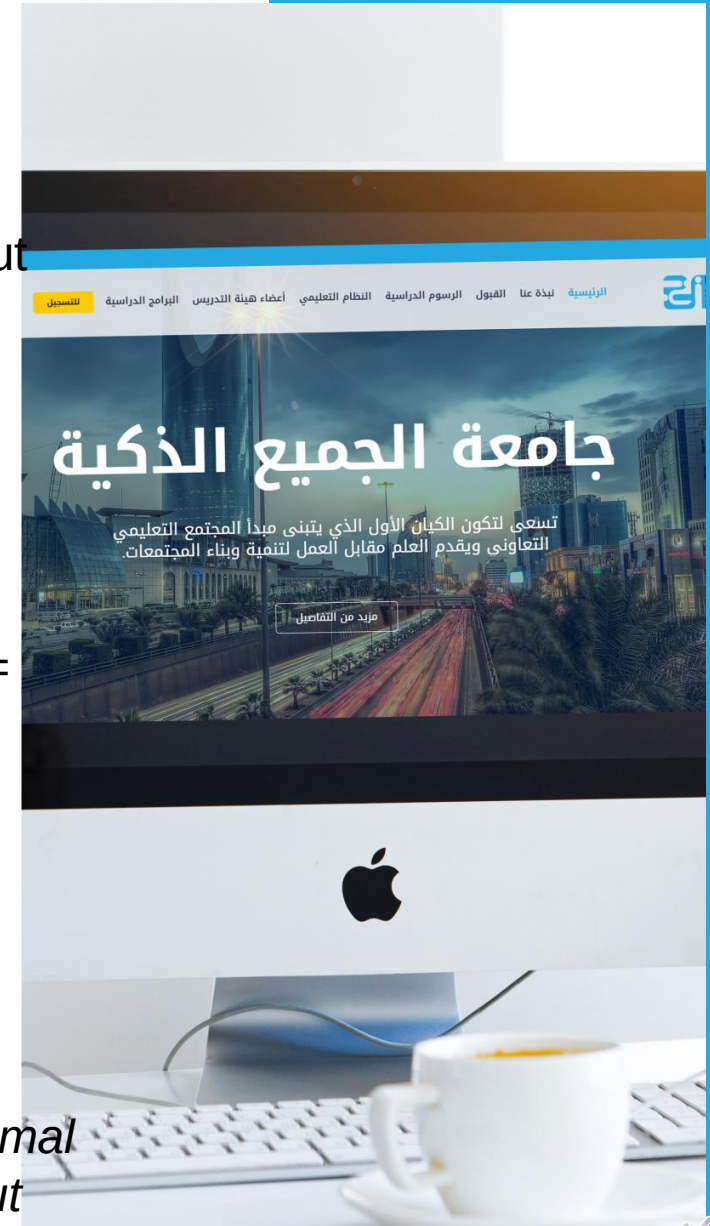
Internal Structure of DES (S-box)

- ❑ The most significant bit (MSB) and the least significant bit (LSB) of each 6-bit input select the row of the table, while the four inner bits select the column.
- ❑ For example, consider the first S-box S_1 below, The S-box input $b = (100101)_2$ indicates the row $11_2 = 3$ (i.e., fourth row, numbering starts with 00_2),
The column $0010_2 = 2$ (i.e., the third column).
- ❑ Therefore, if the input b is fed into S-box S_1 , the output of $S_1(37 = 100101_2) = 8 = 1000_2$.



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

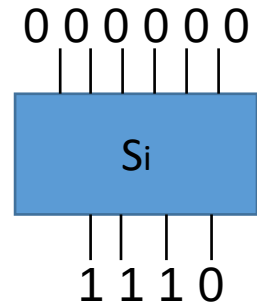
Notice that all numbers are presented in decimal system to shortcut the text



Module 02

Internal Structure of DES (S-box)

❑ According to the previous illustration of S-boxes, you may have all inputs 0s and then you get the 4-bits output 1110 , Such that:



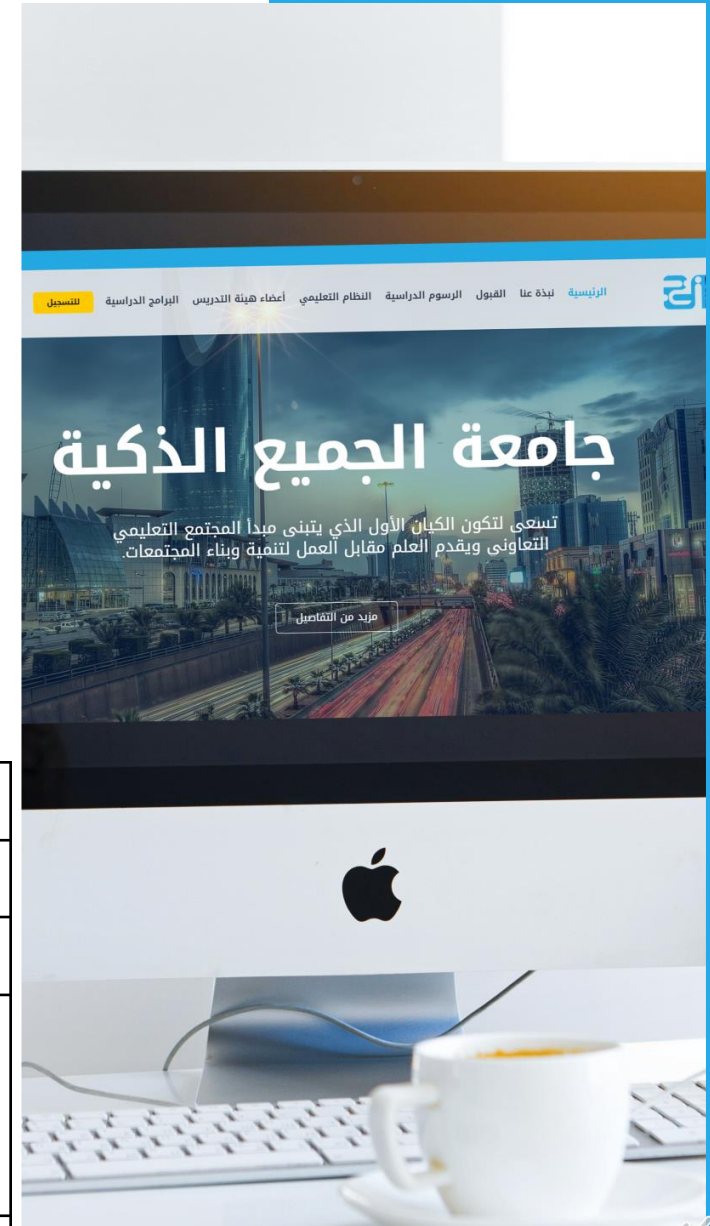
❑ So, if the input is 1 1 1 1 1 1 , the output will be 1 1 0 1 = 13

❑ Question: is there any rule behind this behaviour

❑ Answer: No!! The rule is the table itself

❑ The following slides list all S-boxes in DES algorithm

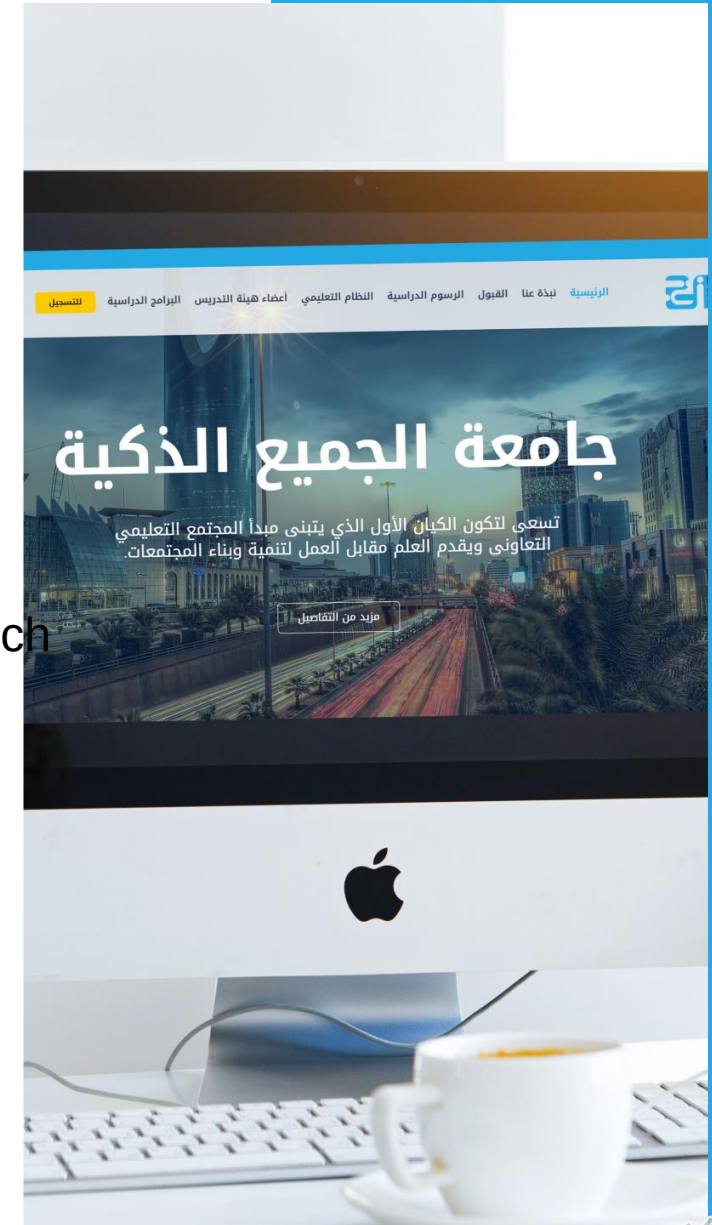
0	1100
1	0110
2	1100
.	.
.	.
.	.
63	1111



Module 02

Internal Structure of DES (S-box)

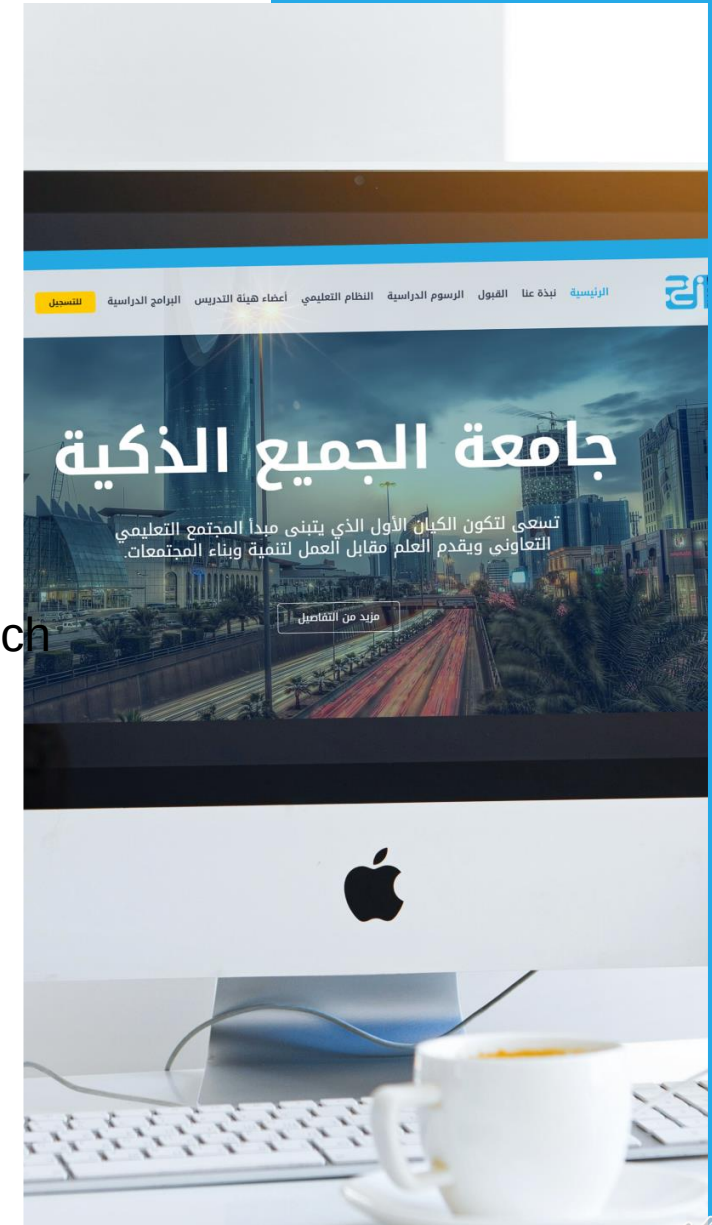
- ❑ The S-boxes were carefully designed to also thwart advanced mathematical attacks, namely, the *differential cryptanalysis*.
- ❑ The differential cryptanalysis attack was first discovered (publically) in the research community in 1990.
- ❑ At that time IBM claimed that they have knowledge about this attack since developing DES algorithm



Module 02

Internal Structure of DES (S-box)

- ❑ The S-boxes were carefully designed to also thwart advanced mathematical attacks, namely, the *differential cryptanalysis*.
- ❑ The differential cryptanalysis attack was first discovered (publically) in the research community in 1990.
- ❑ At that time IBM claimed that they have knowledge about this attack since developing DES algorithm



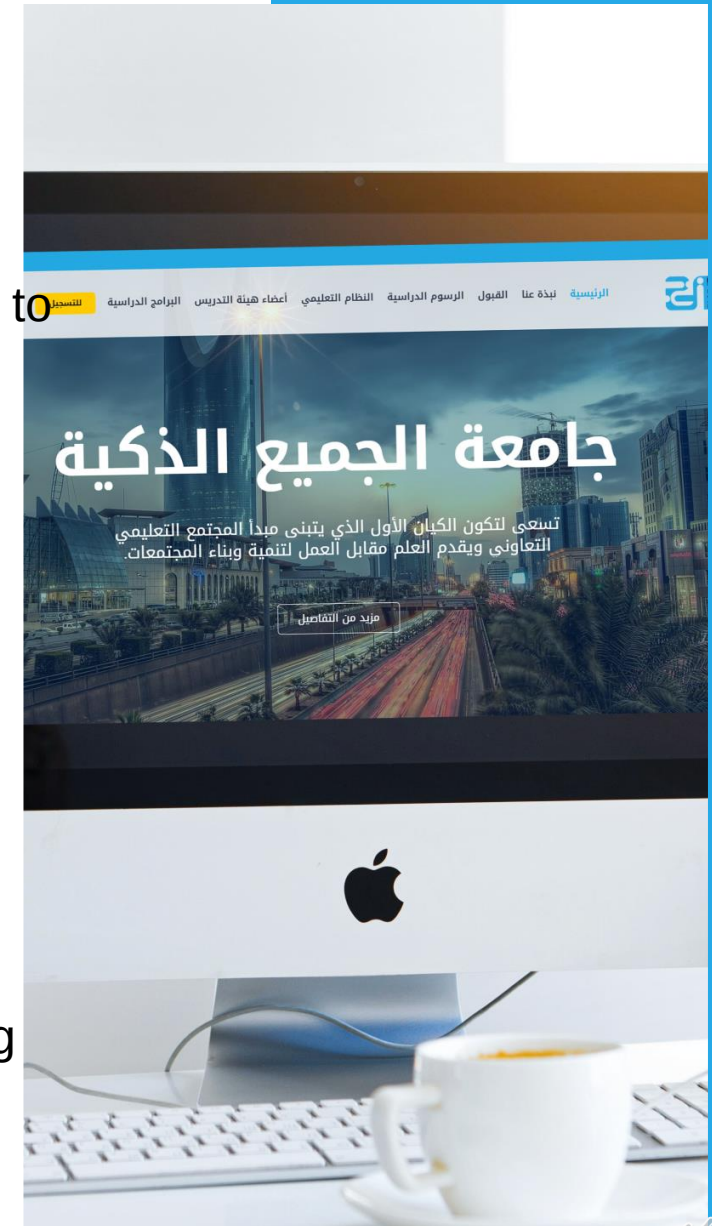
Module 02

Internal Structure of DES (S-box)

❑ At the final stage of the f-function the 32-bit output is permuted bitwise according to the P permutation, which is given in the following 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								

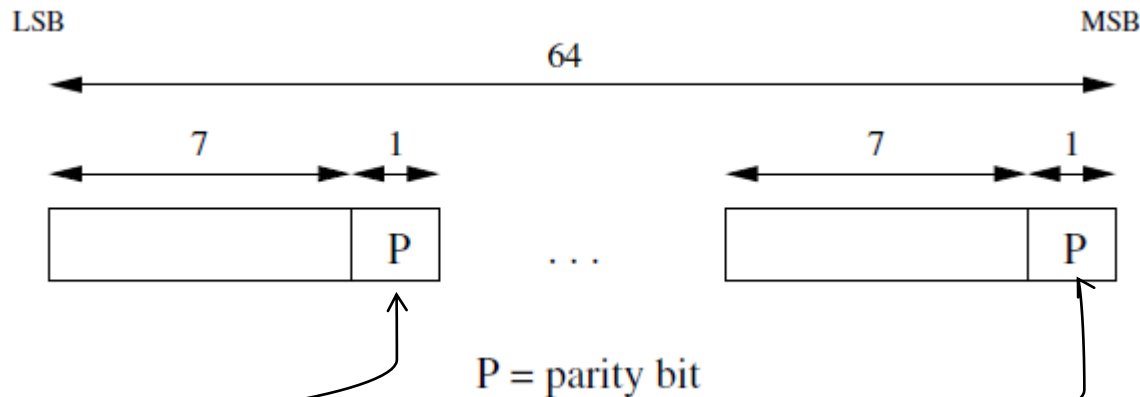
❑ Unlike the initial permutation IP and its inverse IP^{-1} , the permutation P in the f-function introduces diffusion because the four output bits of each S-box are permuted in such a way that they affect several different S-boxes in the following round.



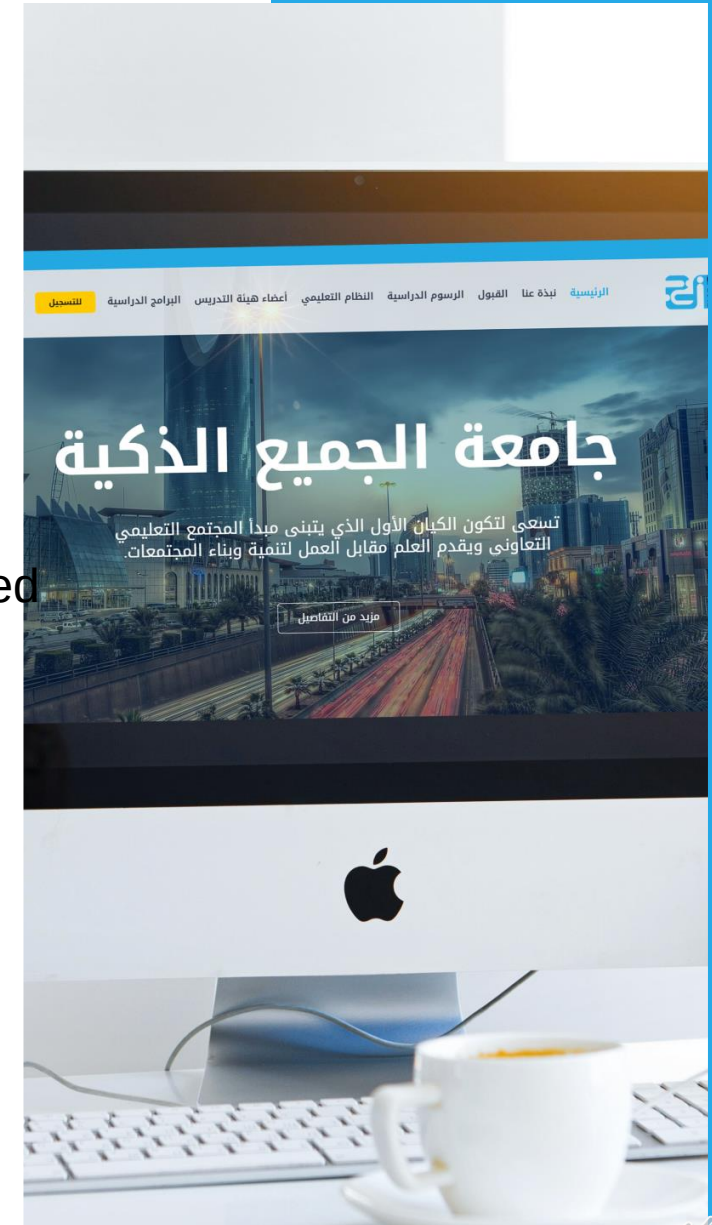
Module 02

Key Schedule

- ❑ The major objective of the key schedule component is to derive 16 round keys k_i , each consisting of 48 bits, from the original 56-bit key
- ❑ The round key is also called sub-key
- ❑ The actual key length of DES is 56 bits not 64
- ❑ The 64-bit key is first reduced to 56 bits by ignoring every eighth bit, which is called the odd parity bit



Notice the
location of
the parity bit



Module 02

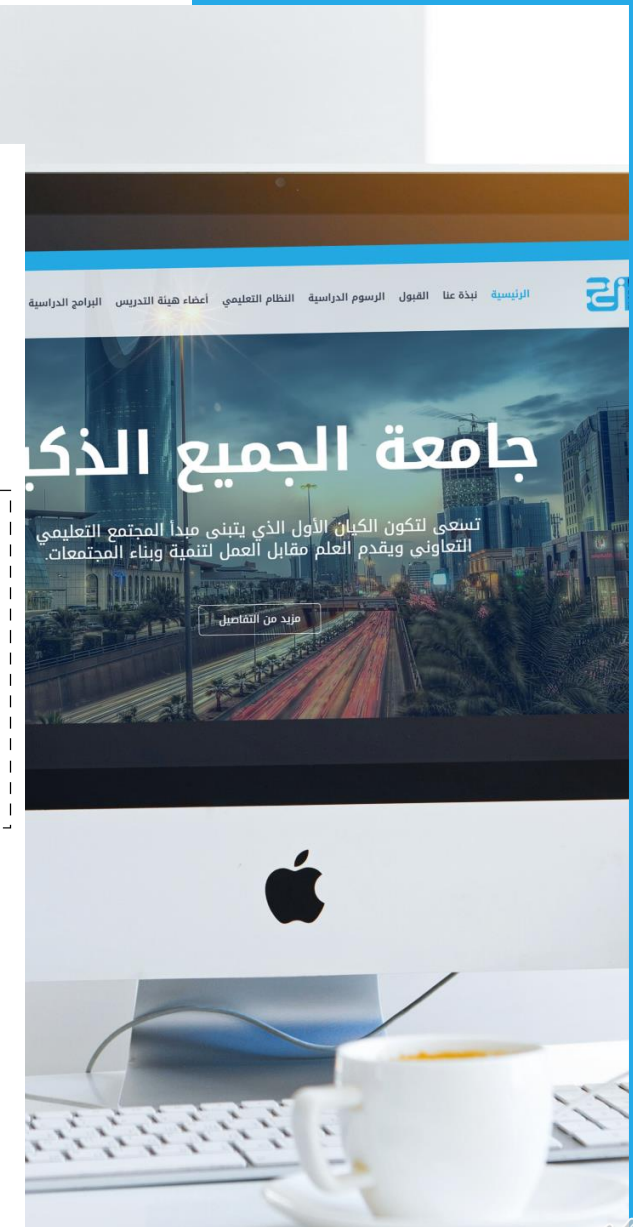
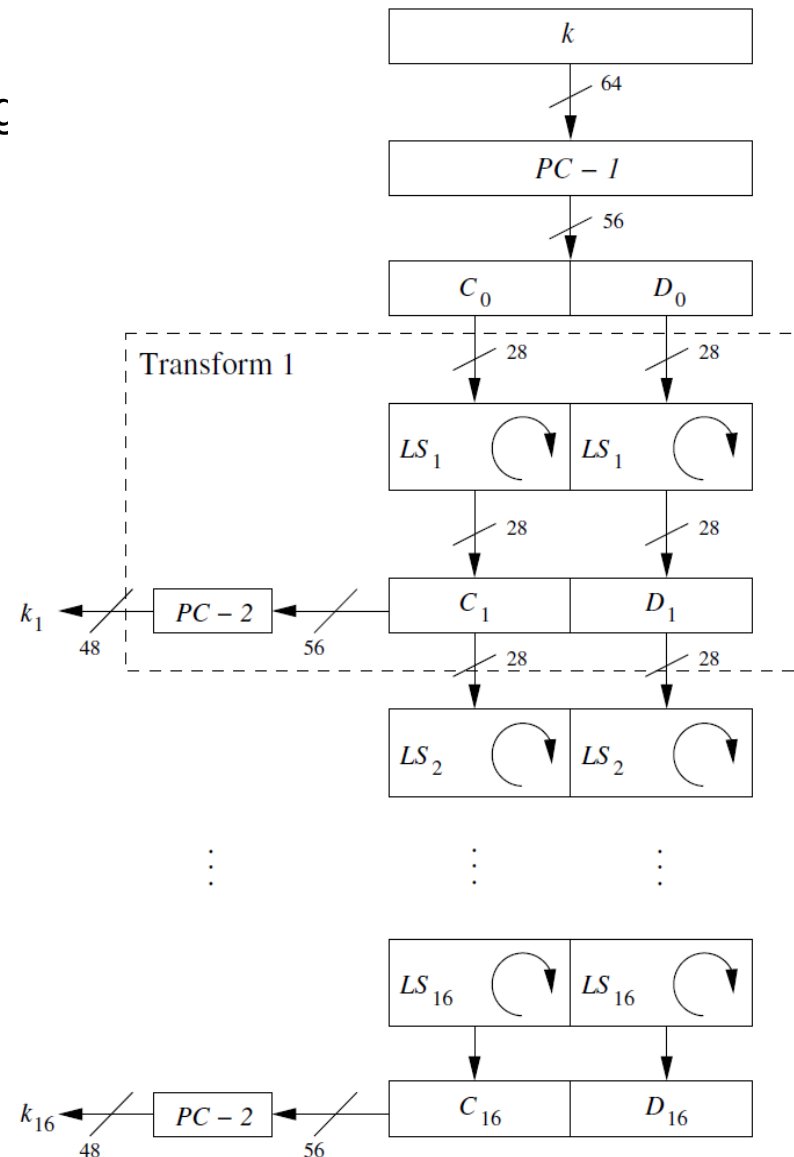
Key Schedule

❑ The following figure shows the different building blocks of key schedule in DES algorithm

❑ PC-1

❑ Rotation

❑ PC-2



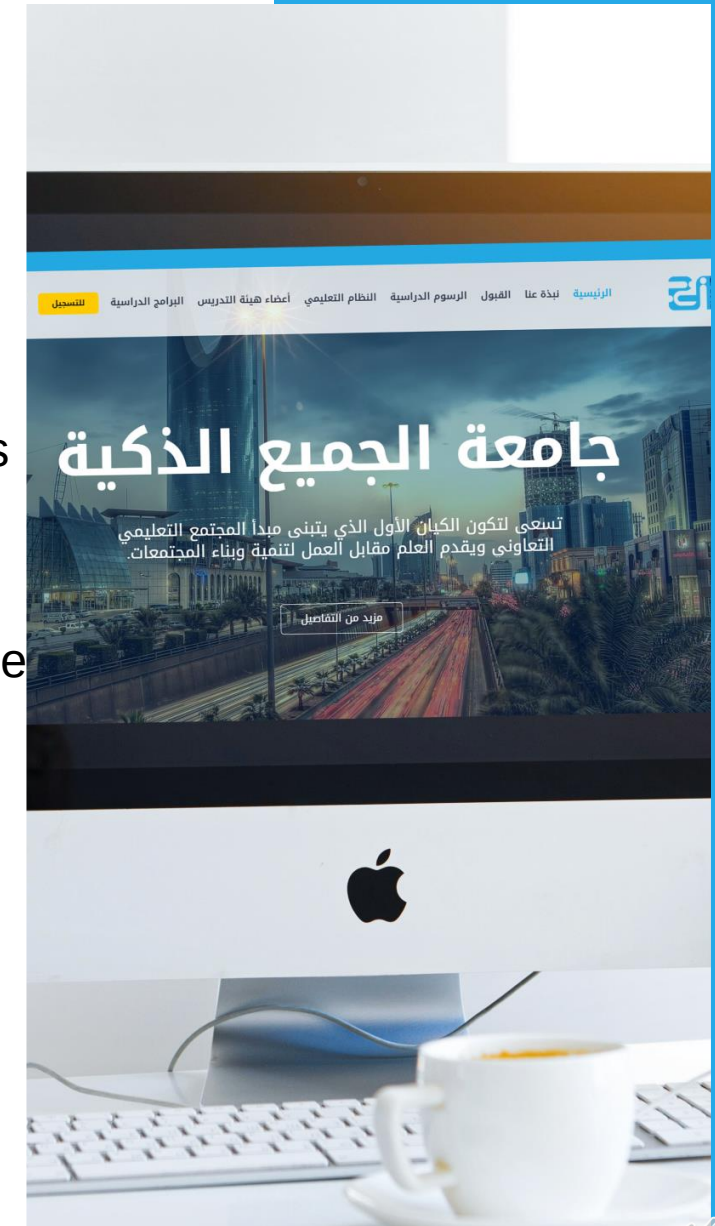
Module 02

Key Schedule

- ❑ How to compute the 16 sub keys for all encryption rounds?
- ❑ Key schedule consists of simple operations
- ❑ PC-1 : Stands for permuted choice one. In this initial permutation, the key length is reduced to 56 bits by stripping the parity (eight) bits.
- ❑ The exact bit permutations (connections) that are realized by PC-1 are given in the following table

$PC - 1$							
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

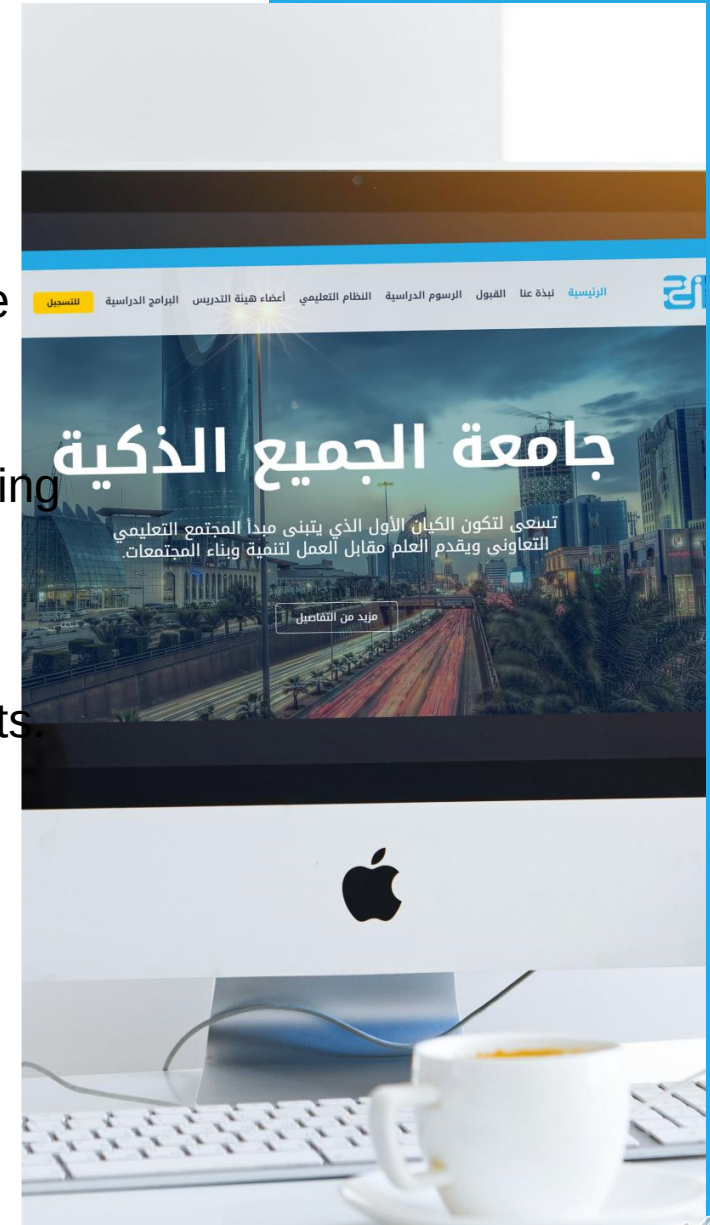
The presented tables is taken from the book: C. Paar and J. Pelzl. Understanding Cryptography



Module 02

Key Schedule

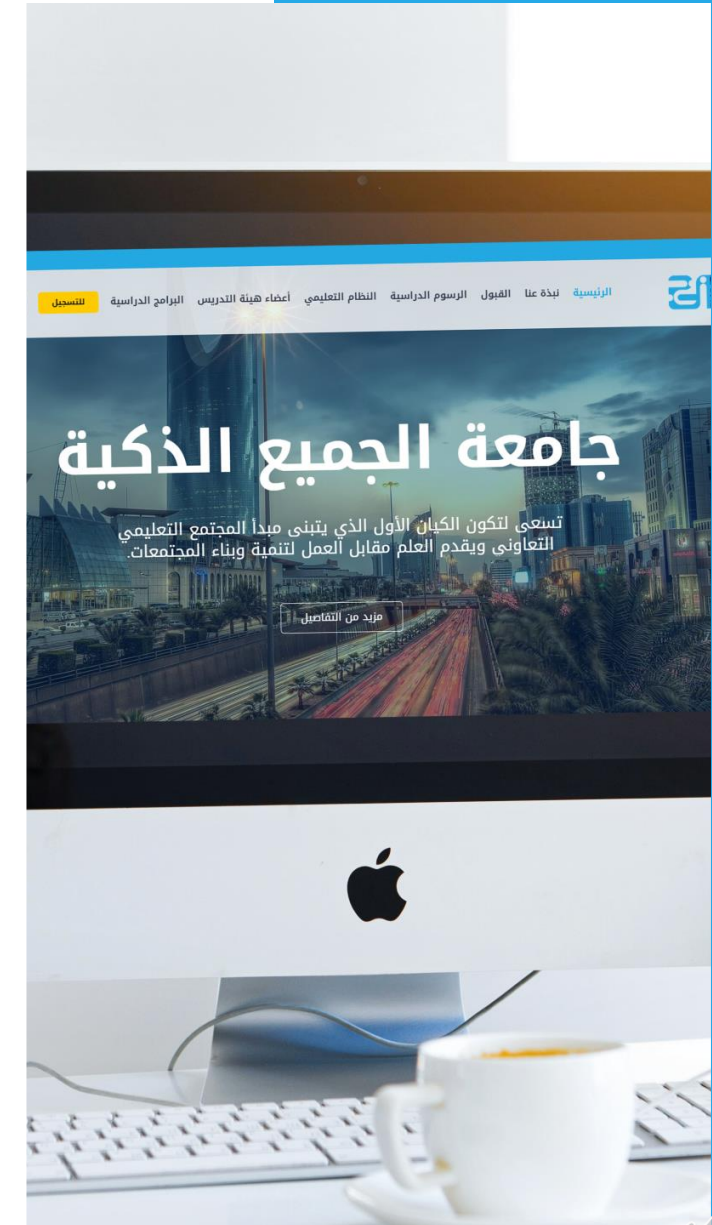
- ❑ After PC-1 step, the resulting 56-bit key is split into two halves C_0 and D_0 , and the actual key schedule starts.
- ❑ The two 28-bit halves are shifted (rotated) left by **one** or **two** bit positions depending on the round No. i according to the following rules:
 - i. In rounds $i = 1, 2, 9, 16$, the two halves are rotated left by one bit.
 - ii. In the other rounds where $i \neq 1, 2, 9, 16$, the two halves are rotated left by two bits.



Module 02

Key Schedule

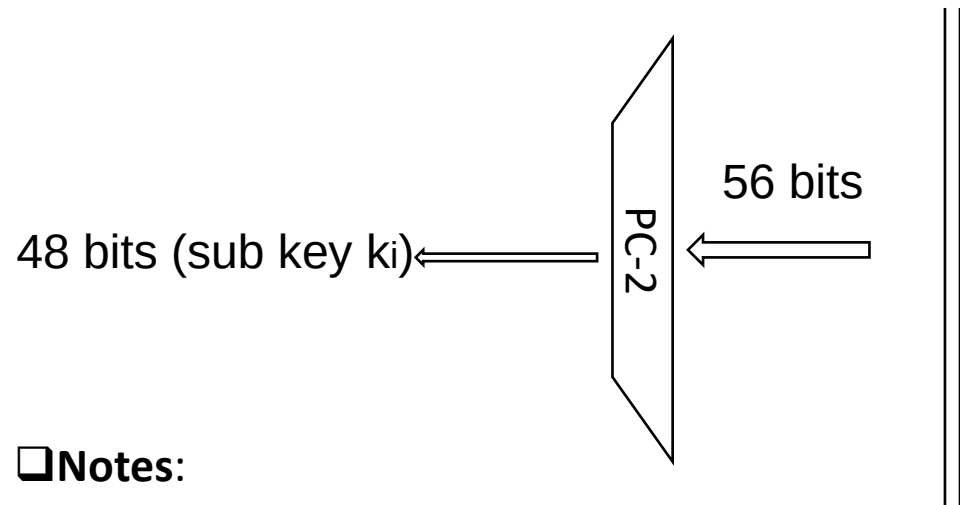
- ❑ The total number of rotation positions (bit position shifted) is $4 \cdot 1 + 12 \cdot 2 = 28$ bits.
- ❑ Therefore, in round 16 the key $(C_{16}, D_{16}) =$ the original key (C_0, D_0) , in particular, $C_{16} = C_0$, and $D_{16} = D_0$
- ❑ This is very useful for the decryption key schedule where the sub-keys have to be generated in reversed order
- ❑ Then, the 56 bits, represented by (C_i, D_i) are fed to the second permutation (PC-2)



Module 02

Key Schedule

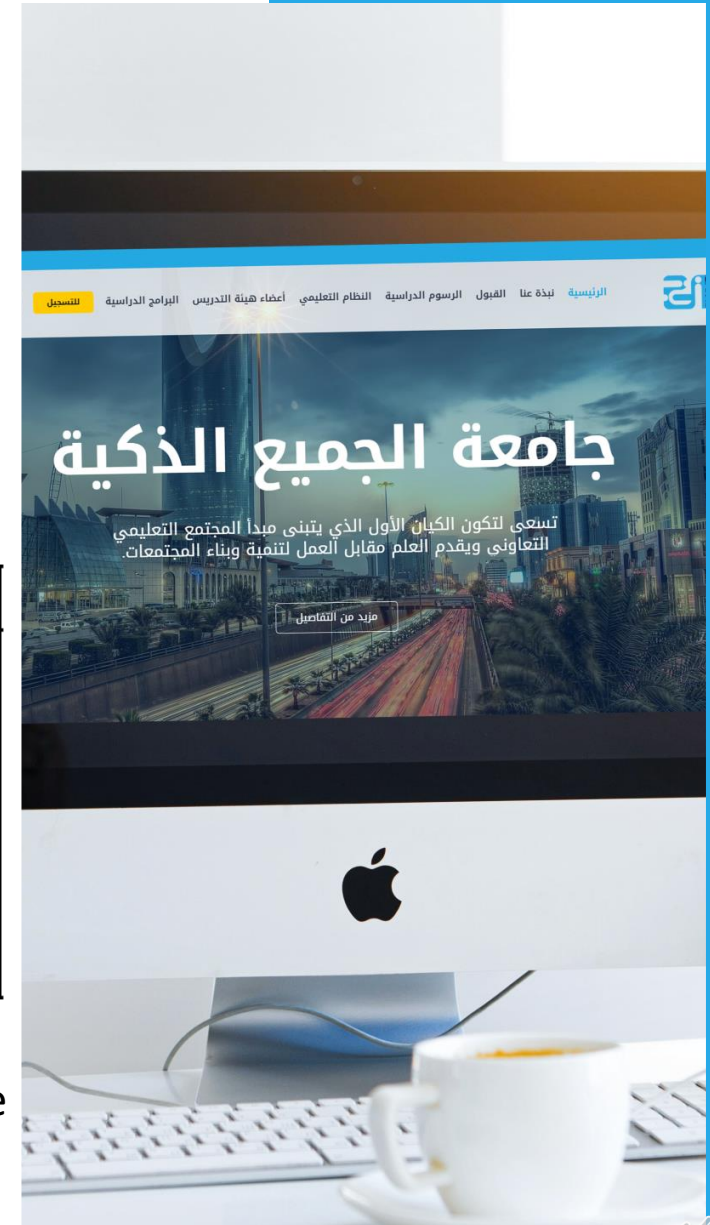
- ❑ PC-2 stands for permuted choice 2. At this stage, the round key k_i is derived.
- ❑ The two halves are permuted bitwise again with $PC-2$
- ❑ PC-2 permutes the 56 input bits coming from C_i and D_i and ignores 8 of them.
- ❑ The exact bit-connections of PC-2 are given in the following table.



$PC - 2$							
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

❑ Notes:

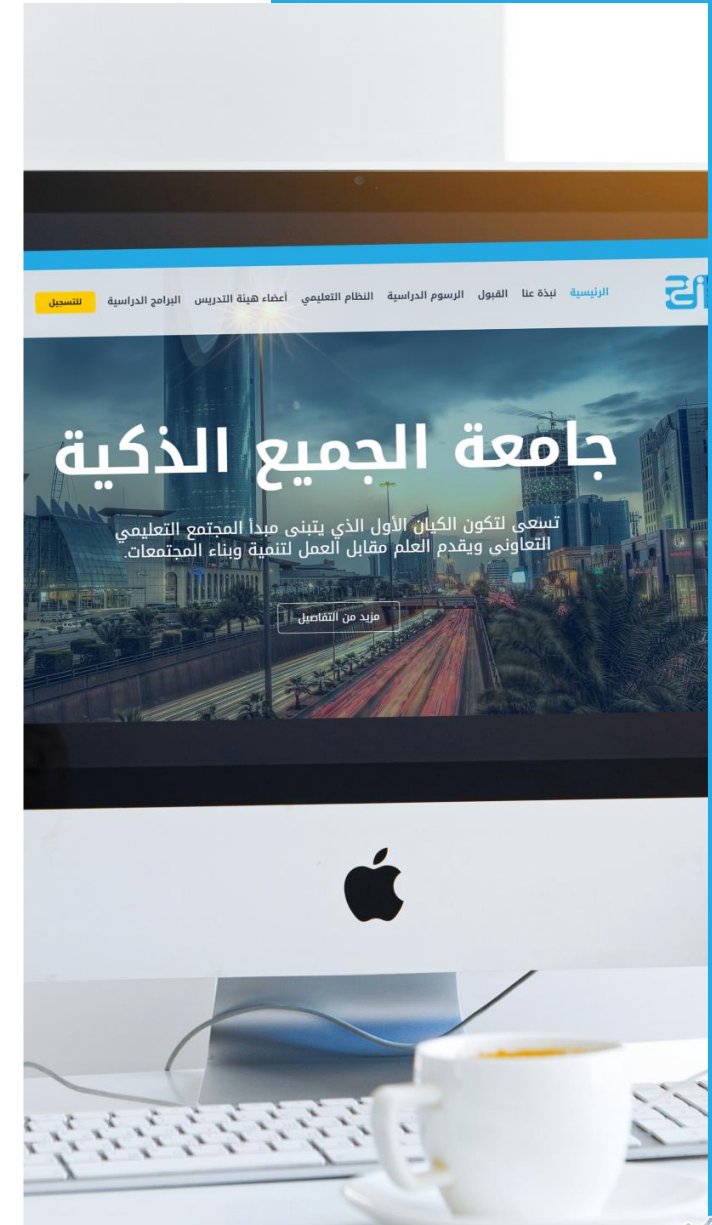
- There is no mathematical relation or any kind of operations between the 56 bits and the 48 bits resulted from PC-2, It is a 48 bits selection only
- Therefore, each sub key $K_1, K_2, \dots, K_{16}$ is merely permutation of the original 56 bits key



Module 02

DES Decryption

- ❑ The DES algorithm is based on a Feistel network.
- ❑ Therefore, decryption in DES is essentially the same function as encryption.
- ❑ In comparison with the encryption function, only the key schedule is reversed, which means that in decryption round No.1, subkey 16 is needed; in round No. 2, subkey 15; ... etc.
- ❑ Thus, during the decryption process, the key schedule algorithm has to generate the round keys as the sequence $k_{16}, k_{15}, \dots, k_1$.
- ❑ **Question:** How can we generate the sub key k_{16} from the original key K

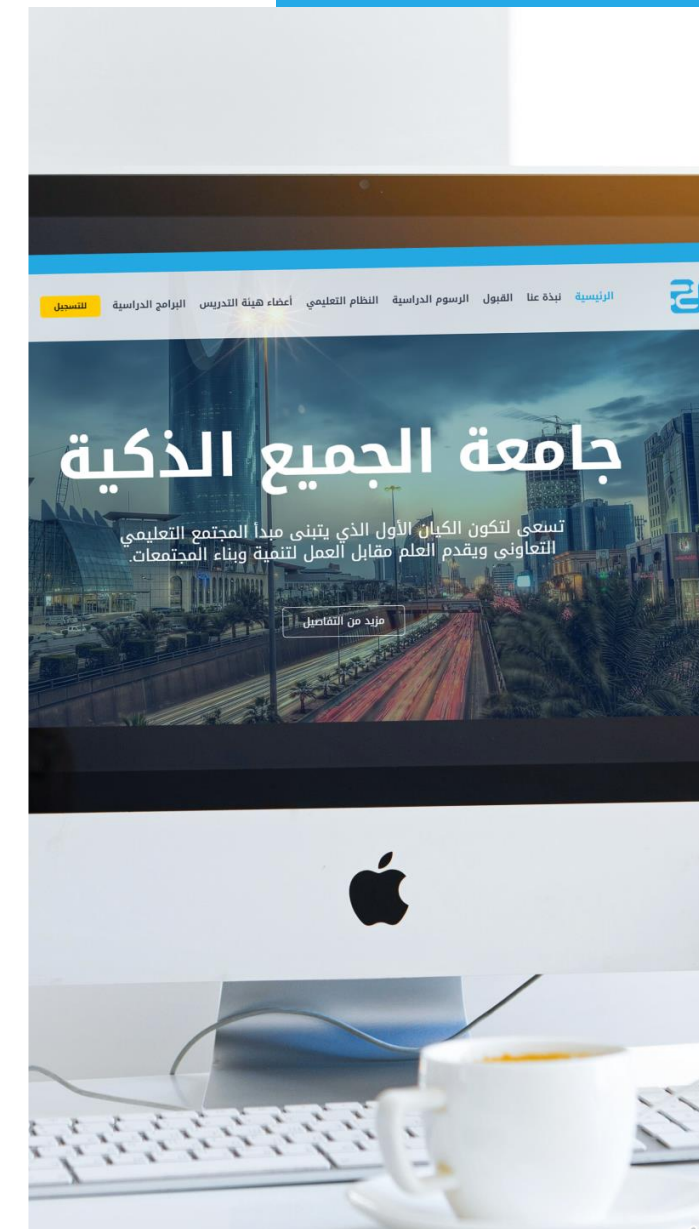
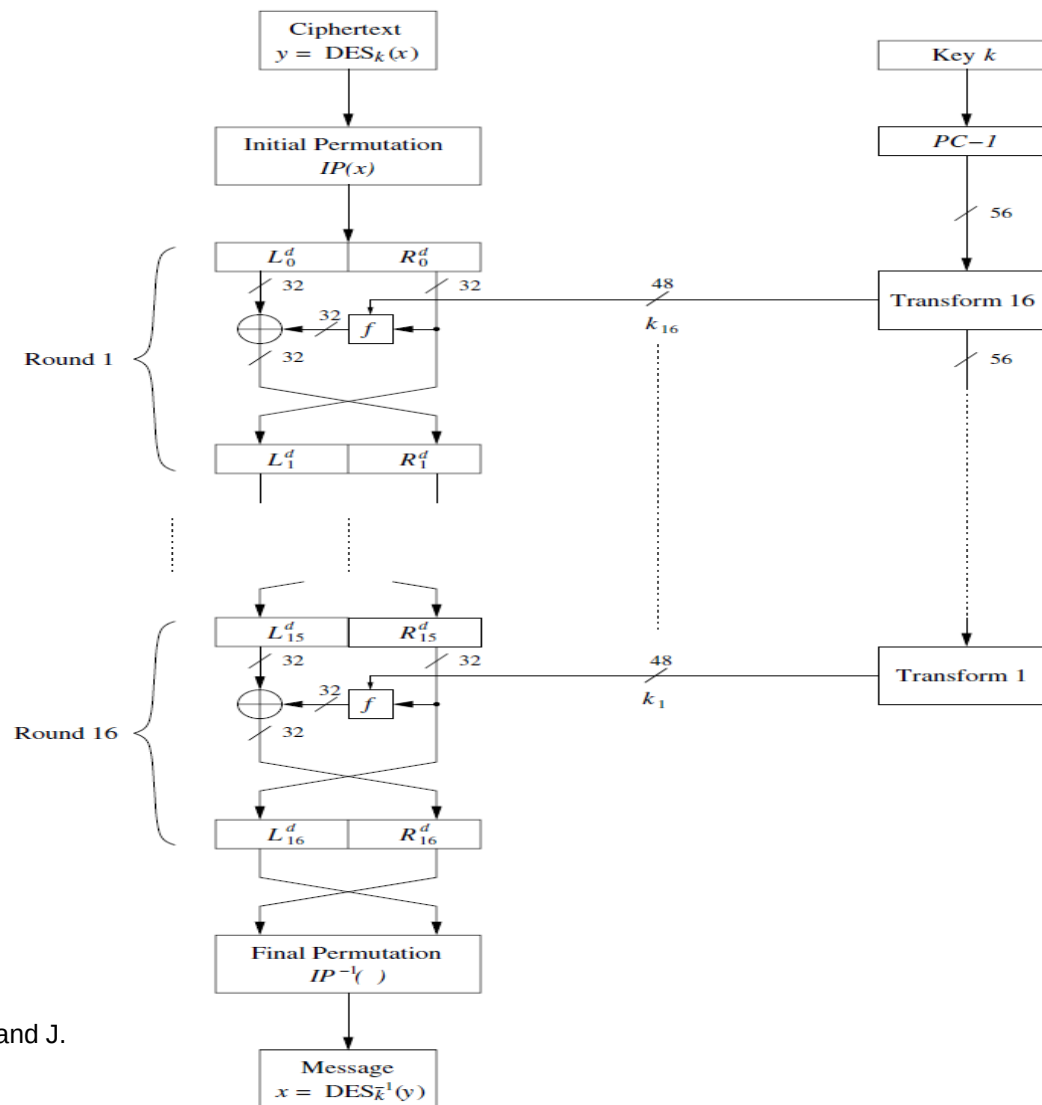


Module 02

Reversed Key Schedule

DES decryption algorithm

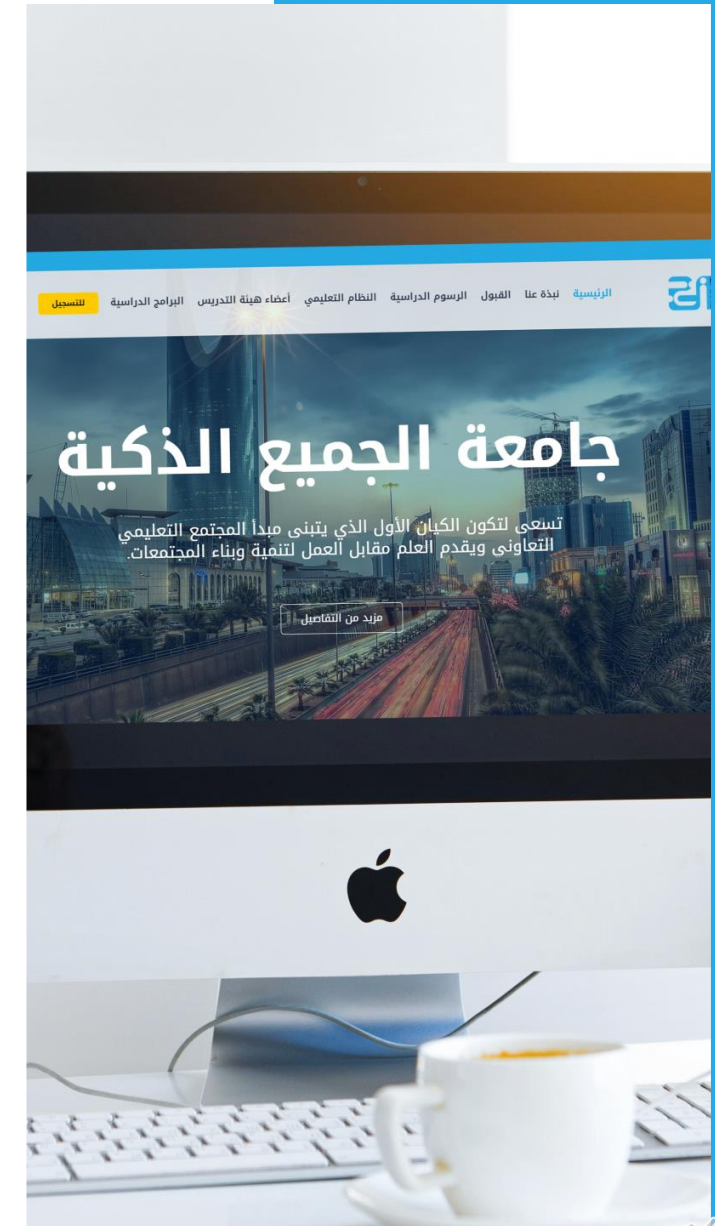
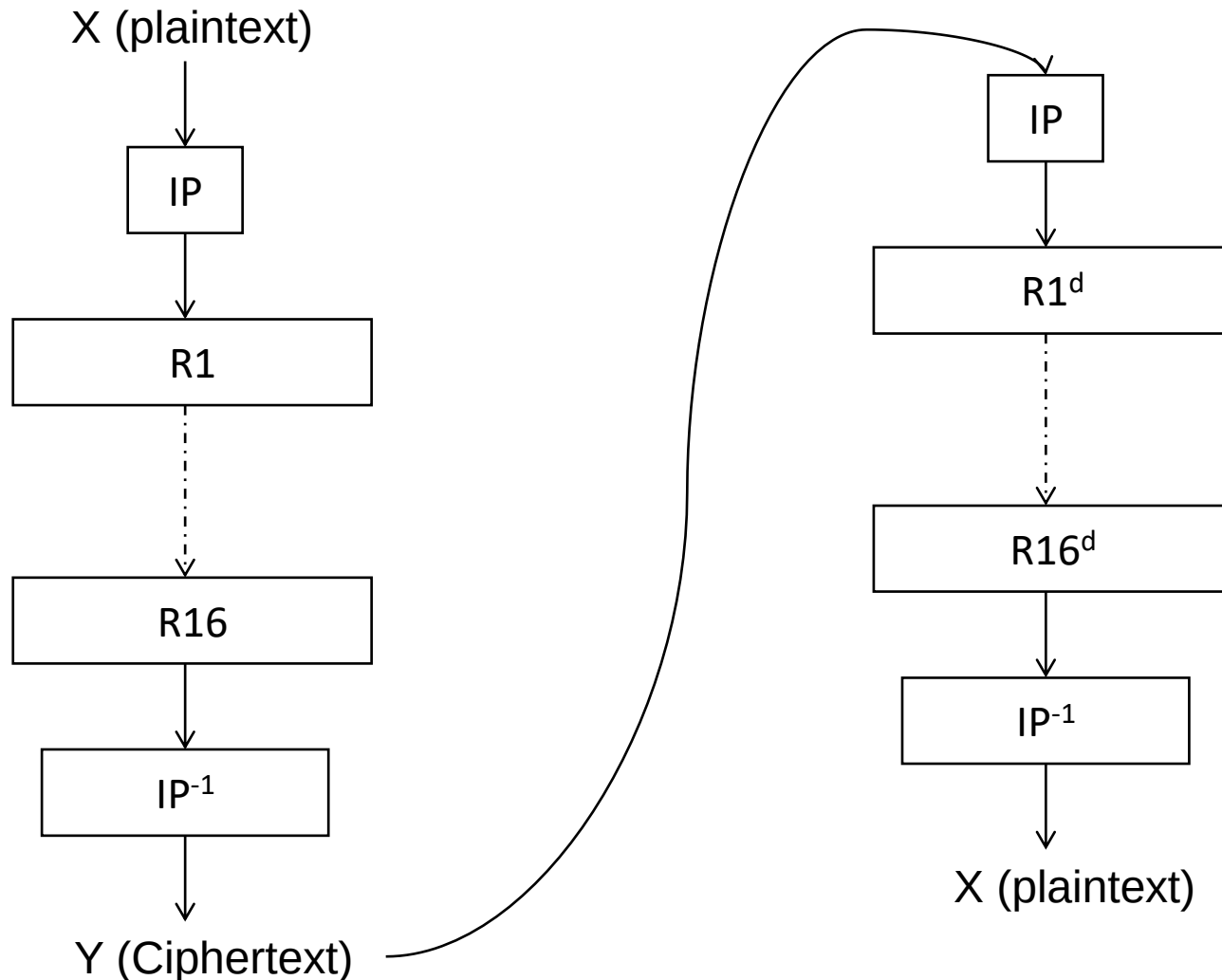
Note that the only difference is within the key schedule process



Module 02

Reversed Key Schedule

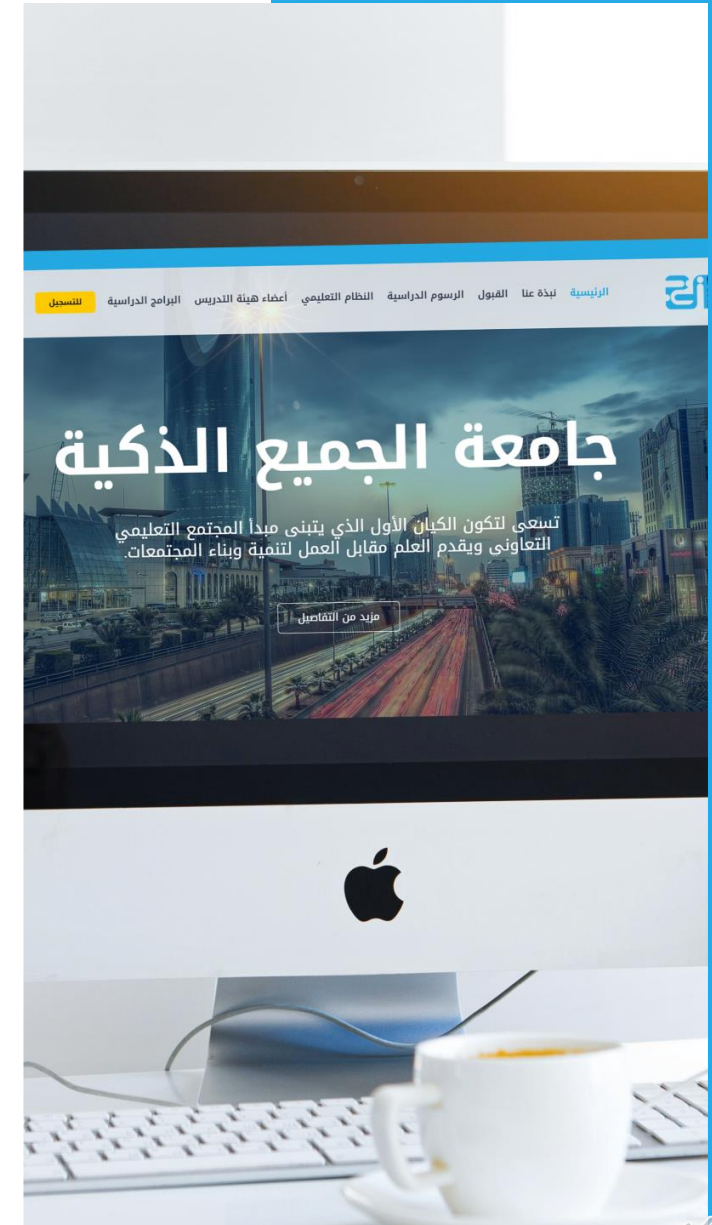
DES decryption algorithm



Module 02

Reversed Key Schedule

- ❑ The IP in decryption process reverses the IP^{-1} in encryption process
- ❑ Decryption round No.1 reverses the encryption round No.16, decryption round No.2 reverses the encryption round No.15,, decryption round No.16 reverses the encryption round No.1.
- ❑ Finally, the IP^{-1} decryption reverses the IP encryption.
- ❑ So, after the decryption process, the original plaintext is obtained

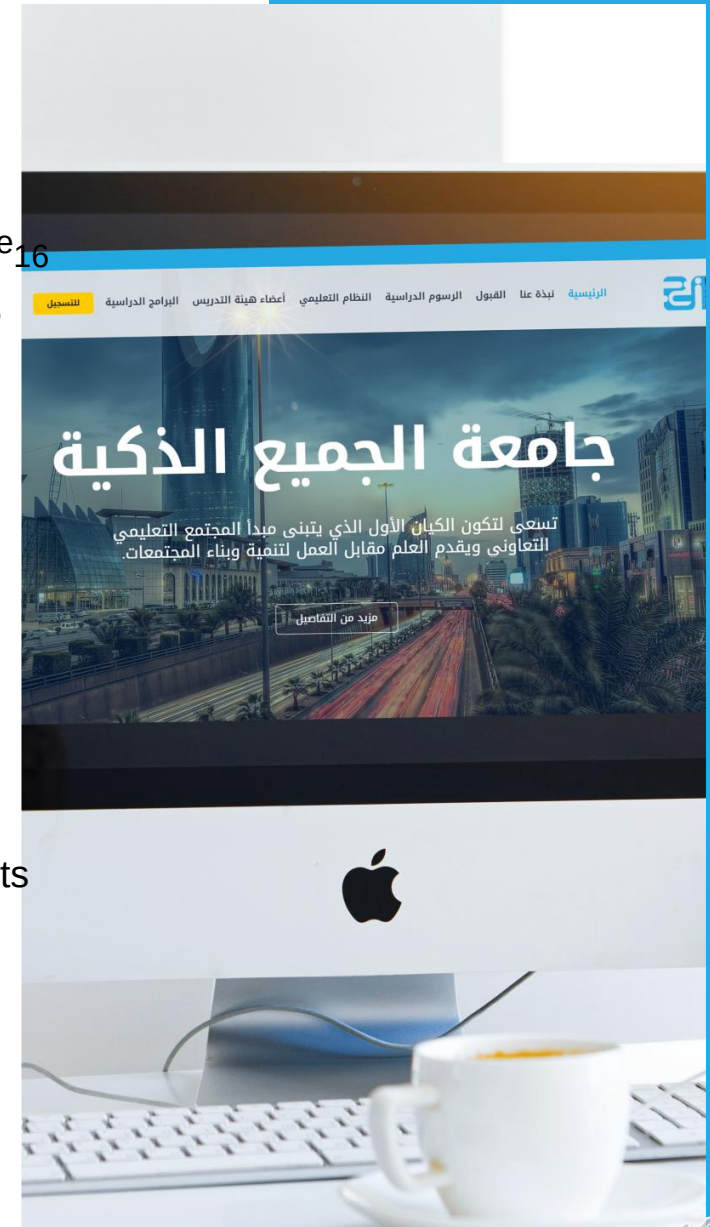
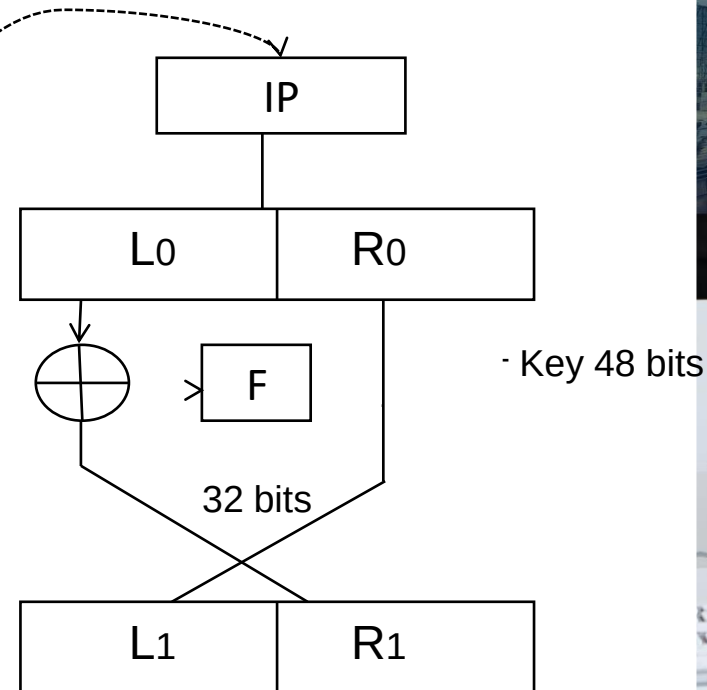
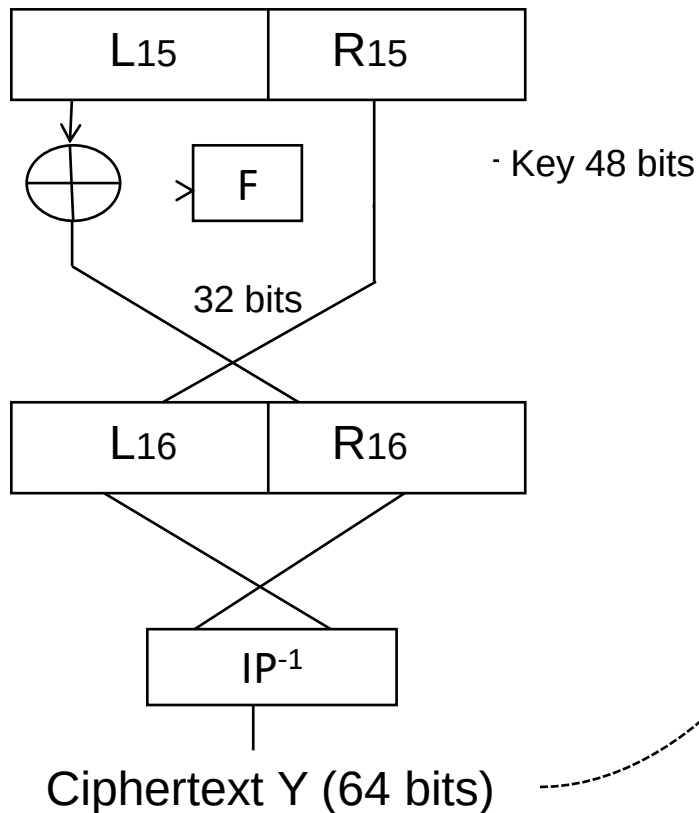


Module 02

Decryption in DES Algorithm

❑ **Objective:** To show that the decryption round R^d_1 reverses the encryption round R^e_{16}

❑ **Note** that the only difference between encryption and decryption processes in DES (and also in Feistel networks) is that decryption uses a reversed key schedule, compared to the encryption process.



Module 02

Attacks against DES Algorithm

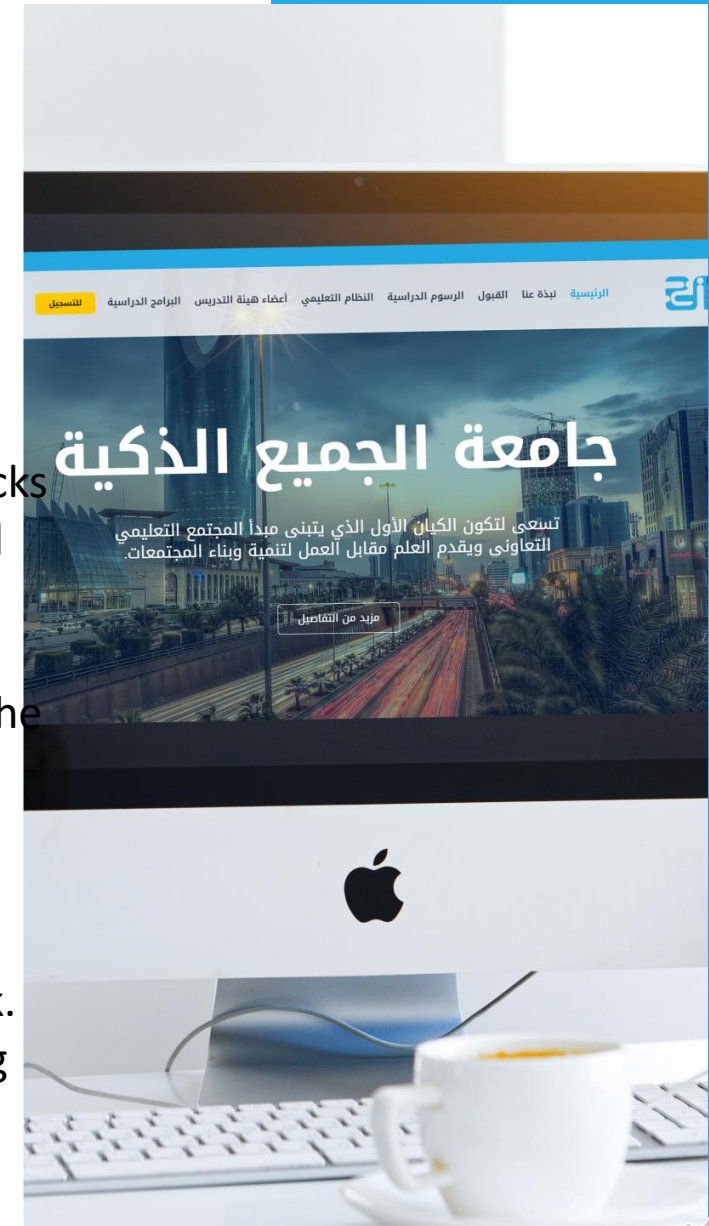
According to previous researches, there are two families of attacks on DES cryptographic algorithm:

A) Analytical Attacks

Researchers in the field of cryptography tried to develop mathematical analytical attacks against DES algorithm since 1974, but they did not succeed until Shamir discovered the **differential attack** in 1988.

However, this attack requires that the attacker should know 2^{47} bits of plaintext and the corresponding ciphertext values, which is difficult to achieve. Moreover, this attack is very hard to practice.

Another considerable attack was developed by Japanese researchers is called **linear cryptanalysis** attack. It shows minor improvement compared to the differential attack. The linear cryptanalysis attack requires 2^{47} bits of the plaintext and the corresponding ciphertext values (*both attacks can be classified as known plaintext attacks*)



Module 02

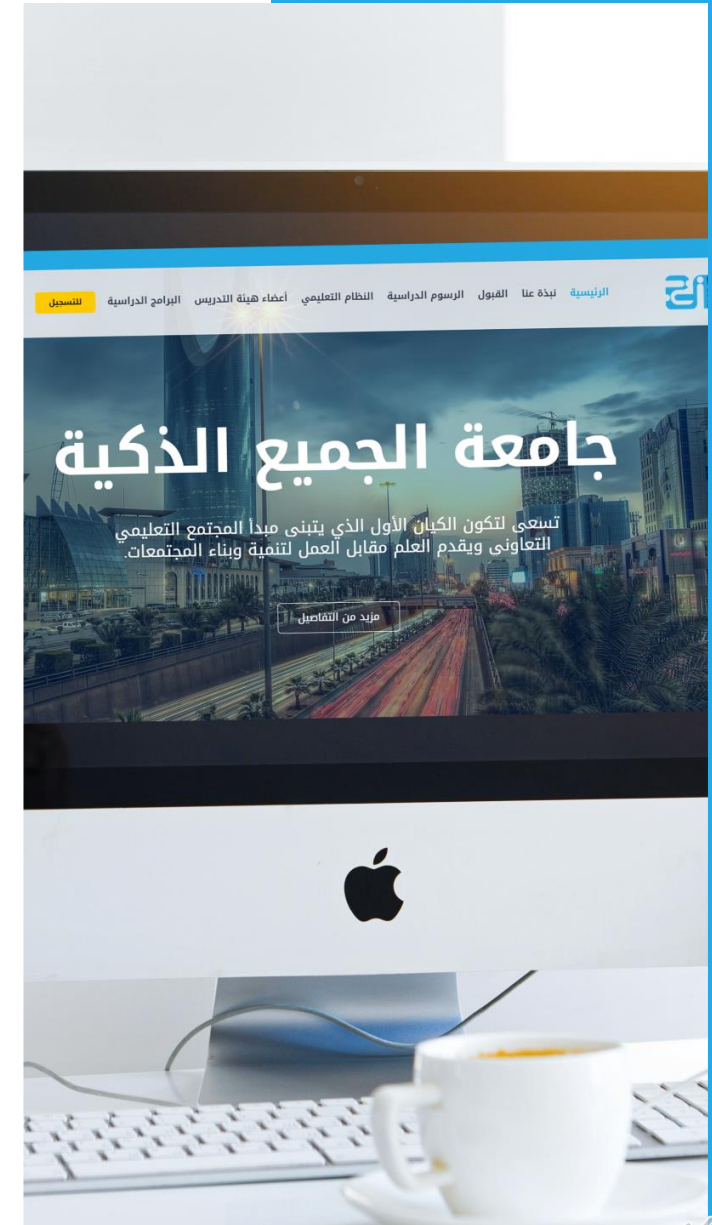
Attacks against DES Algorithm

B) Brute-force-attack

❑ It requires that the attacker knows the first few bits (block of bits, which equals 8 bytes in the case of DES algorithm) of the plaintext as well as the ciphertext

Then the attacker tries all possible key values. Once the attacker reveals the known plaintext from the given ciphertext, it can be assumed that he discovered the key

In 1977, Whitfield Diffie and Martin Hellman estimated that it was possible to build an exhaustive keysearch machine for approximately \$20,000,000. (a special purpose machine)



Module 02

Attacks against DES Algorithm

- ❑ It can be noticed from reviewing the cryptanalysis efforts against DES algorithm is that the major weakness point in DES is the key length.
- ❑ The key space of 2^{56} is considered too small taking into account the latest advances in the field of information technologies.
- ❑ Therefore, researchers recommended that DES is not secure anymore and should not be used as standard encryption algorithm.
- ❑ In particular, DES was replaced by the Advanced Encryption Standard (AES) in 2000

