
CHAPTER 3

BLOCK CIPHERS AND THE DATA ENCRYPTION STANDARD

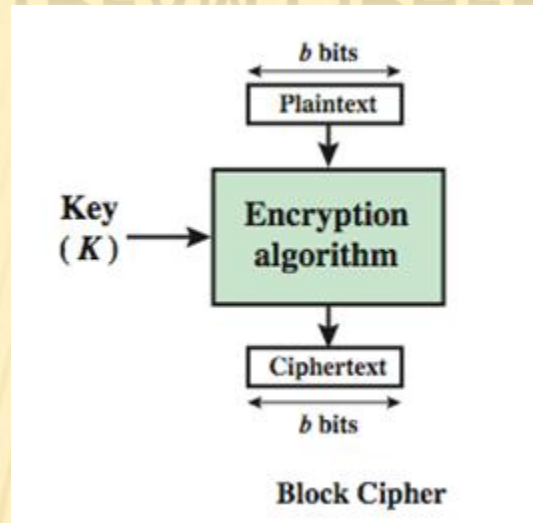
MODERN BLOCK CIPHERS

- The most widely used symmetric cipher : **Data Encryption Standard (DES)**
- DES remains the most important such algorithm although it is replaced by Advanced Encryption Standard (AES).
 - This chapter begins with :
 - Discussion of the general principles of symmetric block ciphers.
 - Next, we cover full DES.
 - A general discussion of block cipher design.

BLOCK VS. STREAM CIPHERS

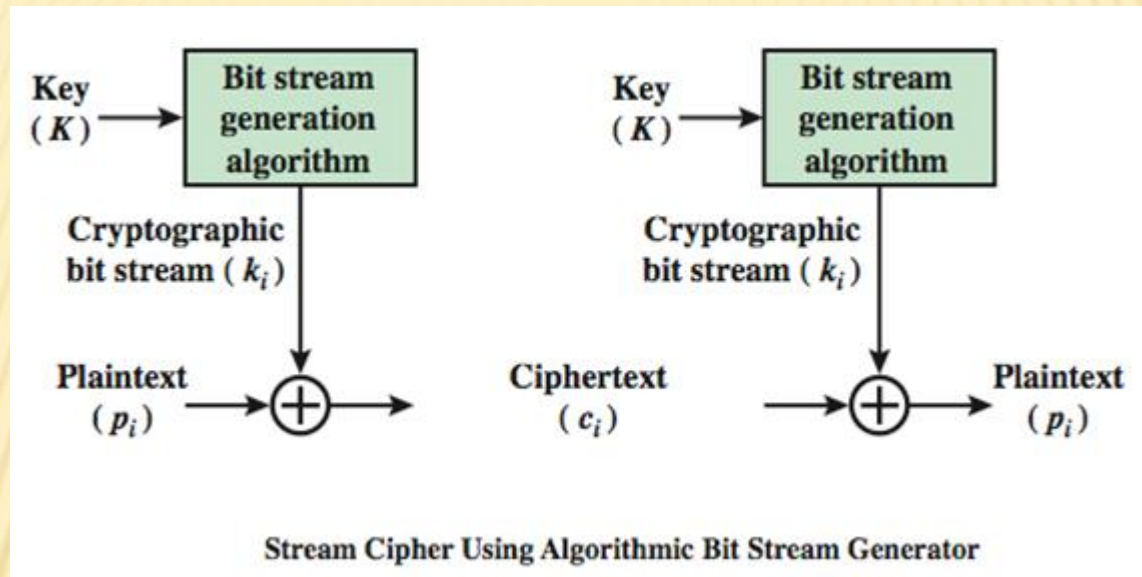
- ❑ Block ciphers process messages in blocks or word (number of bits) at a time, then encrypted or decrypted
- ❑ Stream ciphers process messages in a bit or byte at a time, then encrypted or decrypted
- ❑ many current ciphers are block ciphers
 - ✓ better analyzed
 - ✓ broader range of applications

BLOCK VS. STREAM CIPHERS



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

BLOCK VS. STREAM CIPHERS

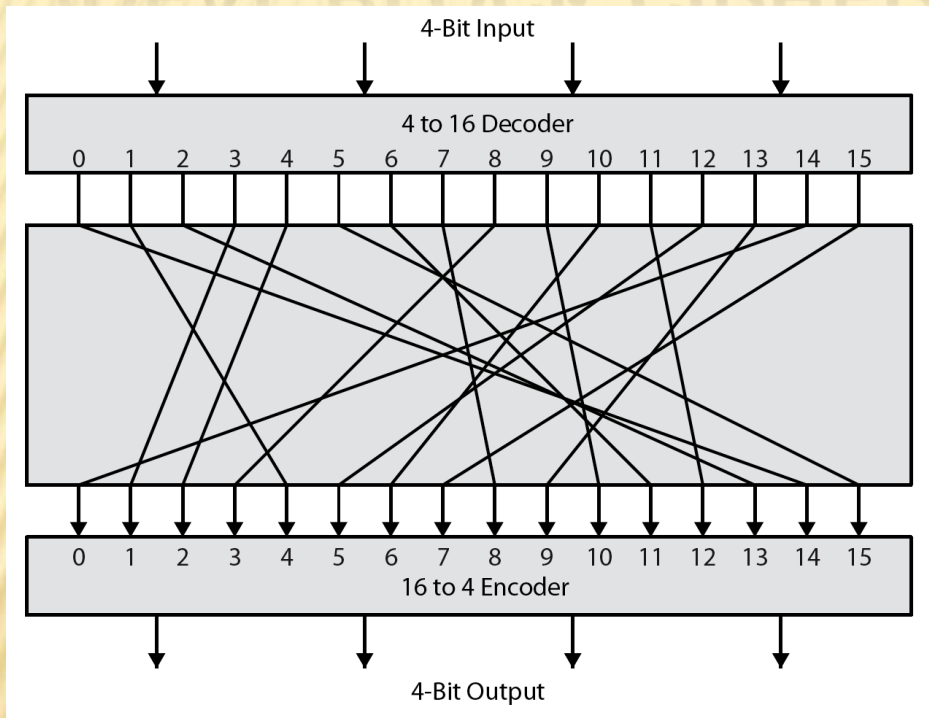


- ❑ A stream cipher, the two users share a symmetric encryption key.
- ❑ Encrypts a digital data stream one bit or one byte at a time. In which the keystream (k) is as long as the plaintext bit stream (p).
- ❑ Ex. , One-time pad version of the Vernam cipher would be used.

BLOCK CIPHER PRINCIPLES

- ❑ Most symmetric block encryption algorithms are based on a structure referred to as a **Feistel block cipher**.
 - The execution of two or more ciphers in sequence, the final result is cryptographically stronger than any of the component ciphers.
- ❑ A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits.
- ❑ Block ciphers look like an extremely large substitution.
- ❑ For n -bit general substitution block cipher, the size of the key is $n \times 2^n$.
- ❑ For a 64-bit block, the key size is $64 \times 2^{64} = 2^{70} = 10^{21}$ bits.

IDEAL BLOCK CIPHER



<i>Plaintext</i>	<i>Ciphertext</i>	<i>Ciphertext</i>	<i>Plaintext</i>
0000	1110	0000	1110
0001	0100	0001	0011
0010	1101	0010	0100
0011	0001	0011	1000
0100	0010	0100	0001
0101	1111	0101	1100
0110	1011	0110	1010
0111	1000	0111	1111
1000	0011	1000	0111
1001	1010	1001	1101
1010	0110	1010	1001
1011	1100	1011	0110
1100	0101	1100	1011
1101	1001	1101	0010
1110	0000	1110	0000
1111	0111	1111	0101

Key Length

Key length 16×4 bits = 64 bits

key length is $2^n \times n$

Actual block size is at least 64 bits

Key length will be $2^{64} \times 64 \approx 10^{21}$ bits

Ability to map any plaintext to any ciphertext is the ideal block cipher

CLAUDE SHANNON AND SUBSTITUTION-PERMUTATION CIPHERS

- Claude Shannon introduced idea of substitution-permutation (S-P) networks from basis of modern block ciphers
- S-P nets are based on two primitive cryptographic operations:
 - substitution (S-box):

Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements.
 - permutation (P-box) :

A sequence of plaintext elements is replaced by a permutation of that sequence. That is, no elements are added or deleted or replaced in the sequence, rather the order in which the elements appear in the sequence is changed.
- Provide confusion & diffusion of message & key

CONFUSION AND DIFFUSION

More practically Shannon suggested combining S&P elements to obtain:

- **Diffusion:** makes relationship between plaintext and ciphertext as complex as possible to thwart attempts to deduce the key.

Message $M = m_1, m_2, m_3, \dots$

$$y_n = \left(\sum_{i=1}^k m_{n+i} \right) \text{ mod } 26$$

Ciphertext letter y_n results from adding k successive plaintext letters (avalanche effect)

- **Confusion:** makes relationship between ciphertext and key as complex as possible to thwart attempts to discover the key

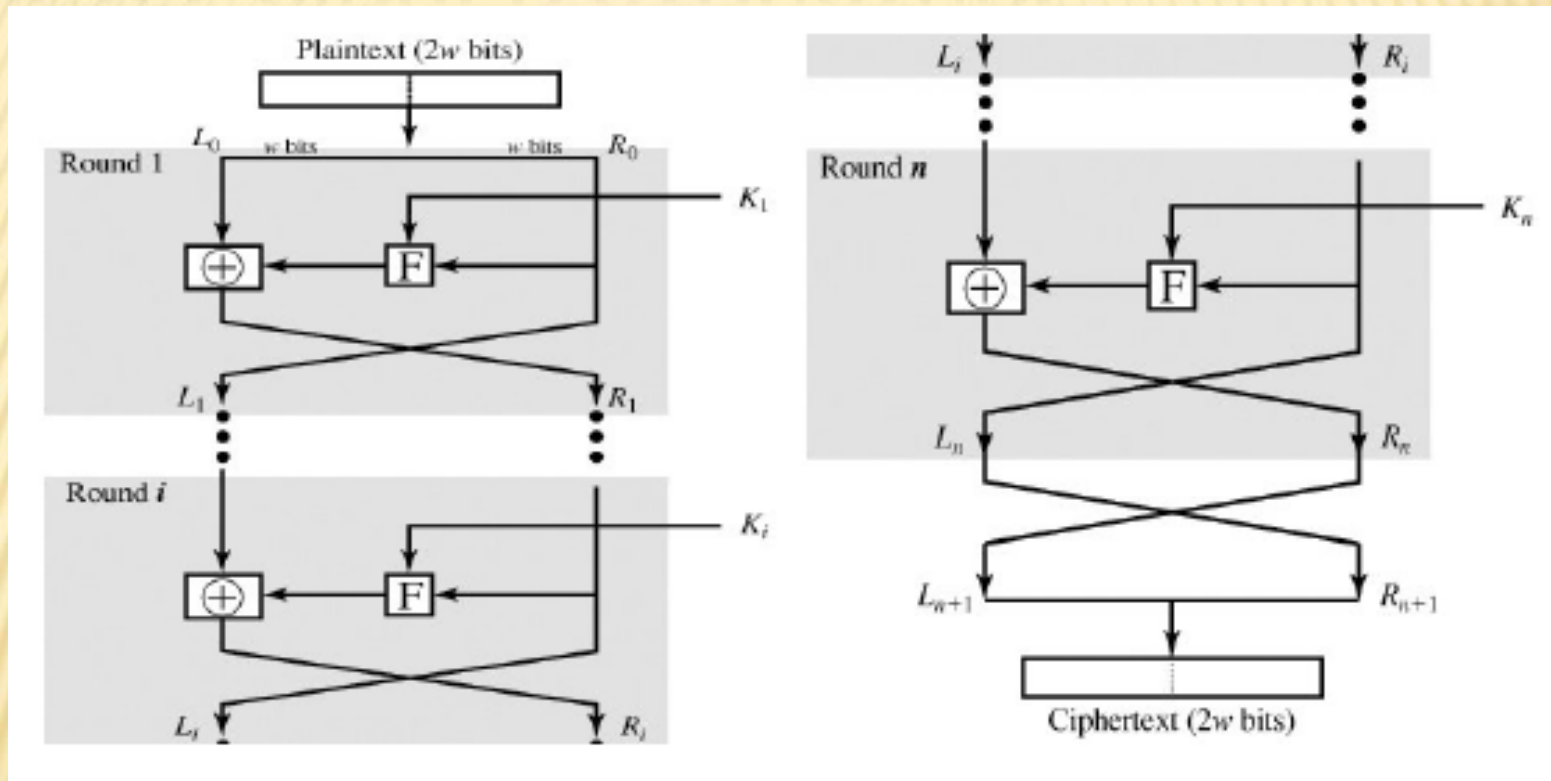
FEISTEL CIPHER STRUCTURE

- ❑ It partitions input block into two halves which are processed through multiple rounds which perform a substitution on left data half, based on round function of right half & subkey, and then have permutation swapping halves.
- ❑ Implements Shannon's S-P net concept

FEISTEL CIPHER STRUCTURE

- ✗ Input
 - + plaintext block of length $2w$
 - + key K
- ✗ Plaintext block divided to L_0, R_0
- ✗ Pass thru n rounds of processing
- ✗ Each round i has
 - + L_{i-1}, R_{i-1} derived from previous round
 - + subkey K_i derived from overall K

FEISTEL CIPHER STRUCTURE



Substitution performed to left half

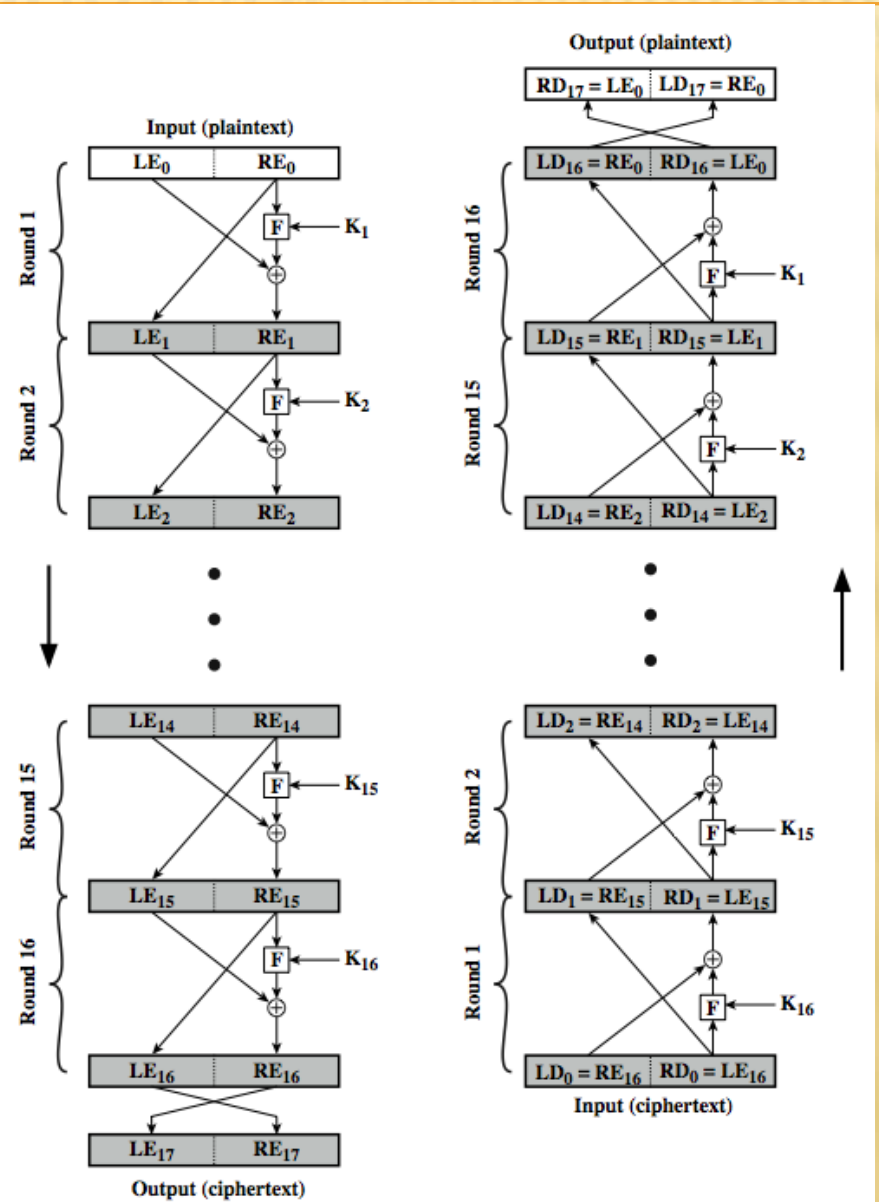
- apply round function F to right half
- take XOR of output with left half
- F is parameterized by round subkey K_i

Permutation of left and right halves

- interchange left and right halves

FEISTEL CIPHER STRUCTURE

- Output of i^{th} encryption round \rightarrow input to $(16-i)^{\text{th}}$ decryption round swapped
- $LE_i || RE_i \equiv RD_{16-i} || LD_{16-i}$



Decryption Proof

- Encryption side

- $LE_{16} = RE_{15}$

- $RE_{16} = LE_{15} \oplus F(RE_{15}, K_{16})$

- Decryption side

- □ $LD_1 = RD_0 = LE_{16} = RE_{15}$

- $LD_0 = RE_{16}$

- $RD_1 = LD_0 \oplus F(RD_0, K_{16})$

- $= RE_{16} \oplus F(RE_{15}, K_{16})$

- $= [LE_{15} \oplus F(RE_{15}, K_{16})] \oplus F(RE_{15}, K_{16})$

- Thus

- □ $RD_1 = [LE_{15} \oplus F(RE_{15}, K_{16})] \oplus F(RE_{15}, K_{16})$

- $= LE_{15}$

The same algorithm with a reversed key order produces the correct result, noting that at every round, the intermediate value of the decryption process is equal to the corresponding value of the encryption process with the two halves of the value swapped.

FEISTEL CIPHER DESIGN ELEMENTS

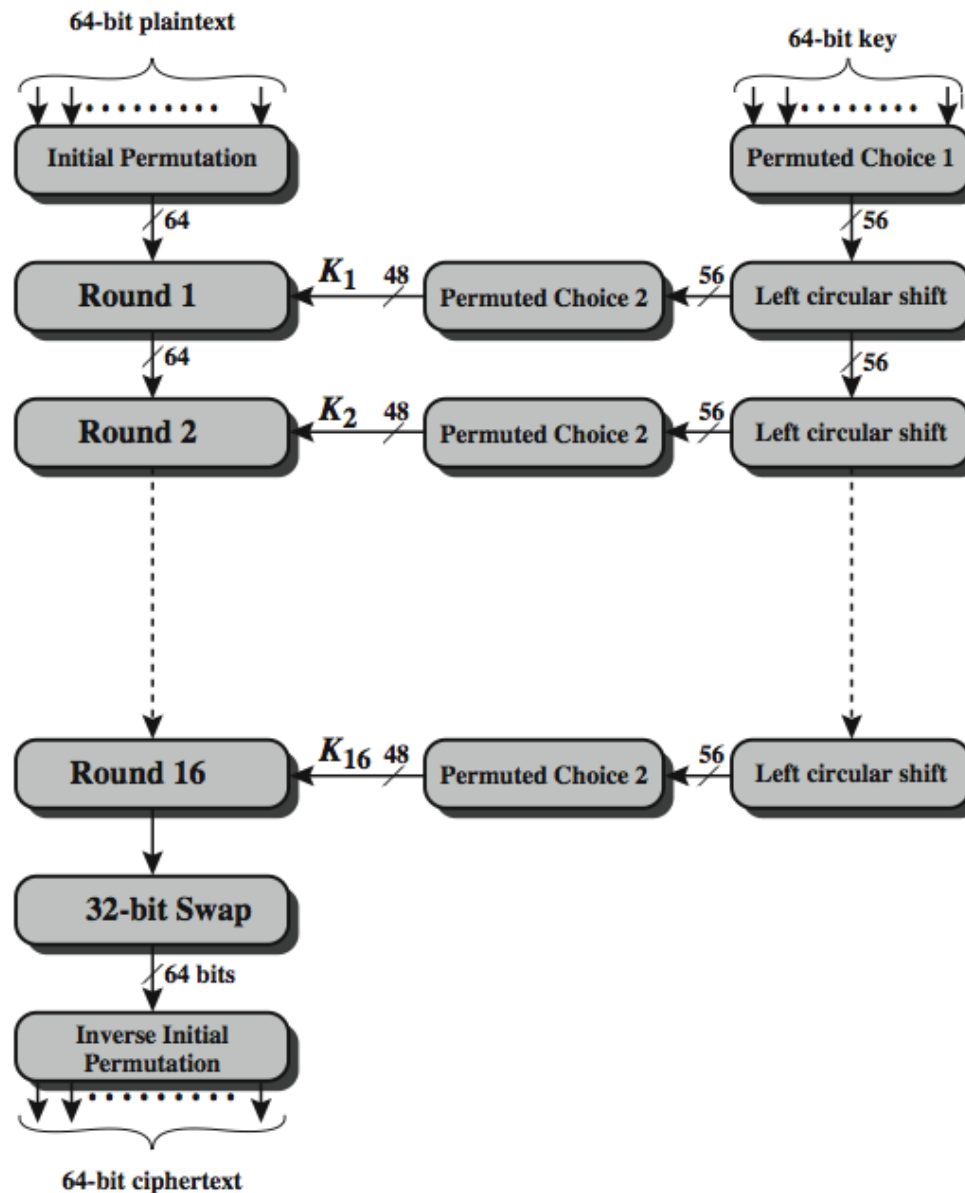
Parameters and Design Features:

- ❑ **Block size:** increasing size improves security, but slows cipher - typical: 64 bit, 128 bit AES
- ❑ **Key size:** increasing size improves security, but slows cipher - typical: 128 bit
- ❑ **Number of rounds** - increasing number improves security, but slows cipher - typical: 16
- ❑ **Sub-key generation algorithm**- greater complexity make cryptanalysis harder, but slows cipher
- ❑ **Round function** - greater complexity make cryptanalysis harder, but slows cipher
- ❑ **Fast software en/decryption** - more recent concern for practical use
- ❑ **Ease of analysis** - for easier validation & testing of strength

DATA ENCRYPTION STANDARD (DES)

- ✘ The most widely used private key block cipher
- ✘ 64-bit plaintext block and 56-bit key
- ✘ Exact structure as Feistel except
 - + initial permutation of plaintext
 - + final permutation of last round's output

DES ENCRYPTION OVERVIEW



DES ENCRYPTION SCHEME

The overall scheme for DES encryption takes as input 64-bits of data and of key.

- The left side shows the basic process for enciphering a 64-bit data block :
 - an initial permutation (IP) which shuffles the 64-bit input block
 - 16 rounds of a complex key dependent round function involving substitutions & permutations
 - a final permutation, being the inverse of IP

- The right side shows the handling of the 56-bit key :
 - an initial permutation of the key (PC1) which selects 56-bits out of the 64-bits input, in two 28-bit halves
 - 16 stages to generate the 48-bit sub keys using a left circular shift and a permutation of the two 28-bit halves.

THE INITIAL PERMUTATION AND ITS INVERSE ARE DEFINED BY FOLLOWING TABLES

Initial Permutation (IP)

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

Inverse Initial Permutation (IP⁻¹)

(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

(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

-
- ✘ The input to a table consists of 64 bits numbered left to right 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 also consists of 64 bits.
 - ✘ Note that the bit numbering for DES reflects IBM mainframe practice, and is the opposite of what we now mostly use - so be careful! Numbers from Bit 1 (leftmost, most significant) to bit 32/48/64 etc (rightmost, least significant).
 - ✘ For example, a 64-bit plaintext value of “675a6967 5e5a6b5a” (written in left & right halves) after permuting with IP becomes “ffb2194d 004df6fb”. Note that example values are specified using hexadecimal.

INITIAL PERMUTATION IP

- IP reorders the input data bits
- even bits to LH half, odd bits to RH half
- quite regular in structure (easy in h/w)
- example:

IP (675a6967 5e5a6b5a) = (ffb2194d 004df6fb)

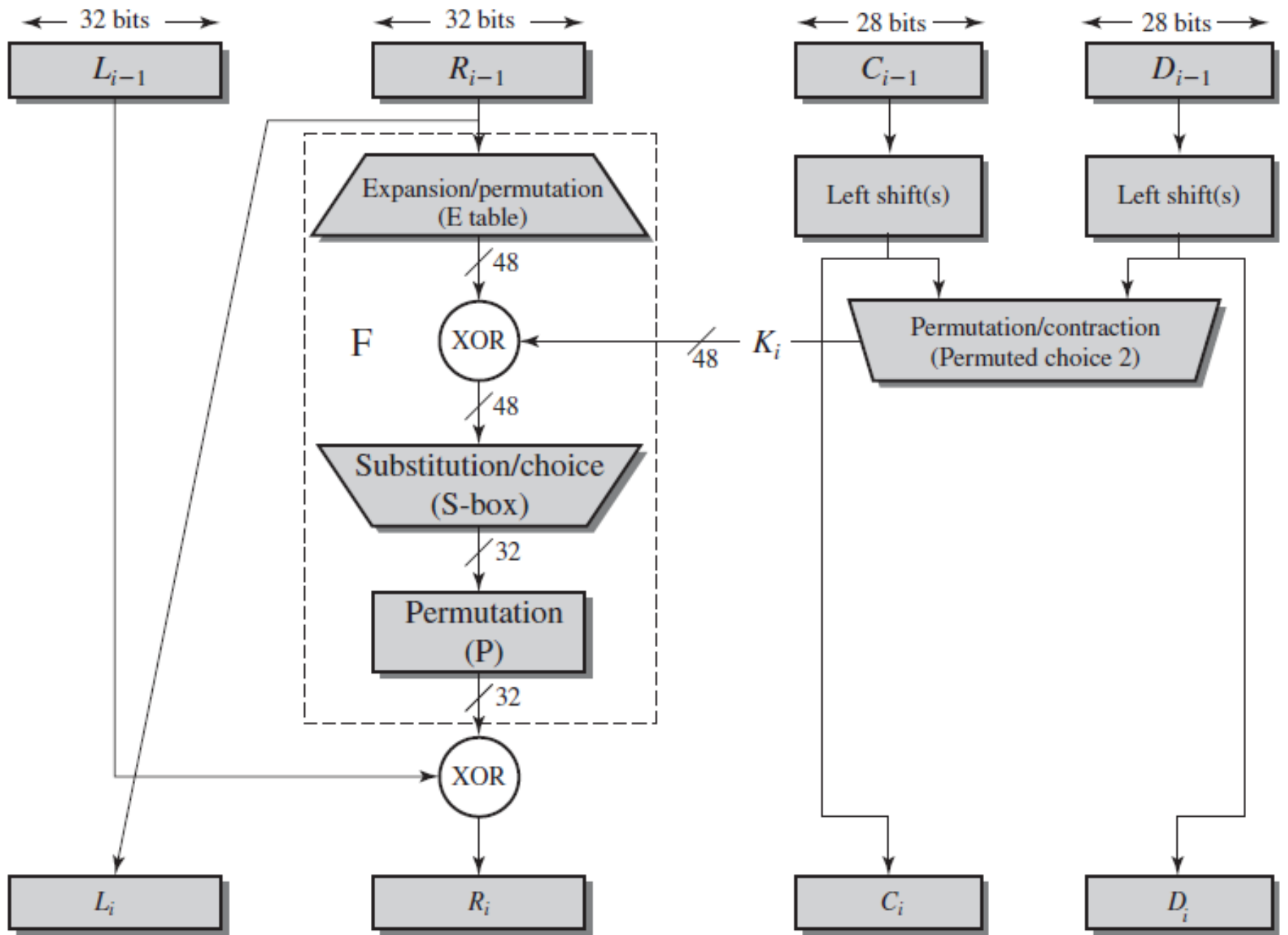
DETAILS OF SINGLE ROUND

- ✘ Uses two 32-bit L & R halves

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

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

- ✘ F takes 32-bit R half and 48-bit subkey:
 - + expands R to 48-bits using perm E
 - + adds to subkey using XOR
 - + passes through 8 S-boxes to get 32-bit result
 - + finally permutes using 32-bit perm P



DES SINGLE ROUND ALGORITHM

The round key K_i is 48 bits.

The R input is 32 bits.

The internal structure of the DES round function F. The R input is first expanded to 48 bits by using expansion table E that defines a permutation plus an expansion that involves duplication of 16 of the R bits.

(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

The resulting 48 bits are XORed with key K_i . This 48-bit result passes through a substitution function comprising 8 S-boxes which each map 6 input bits to 4 output bits, producing a 32-bit output, which is then permuted by permutation P.

(d) Permutation Function (P)

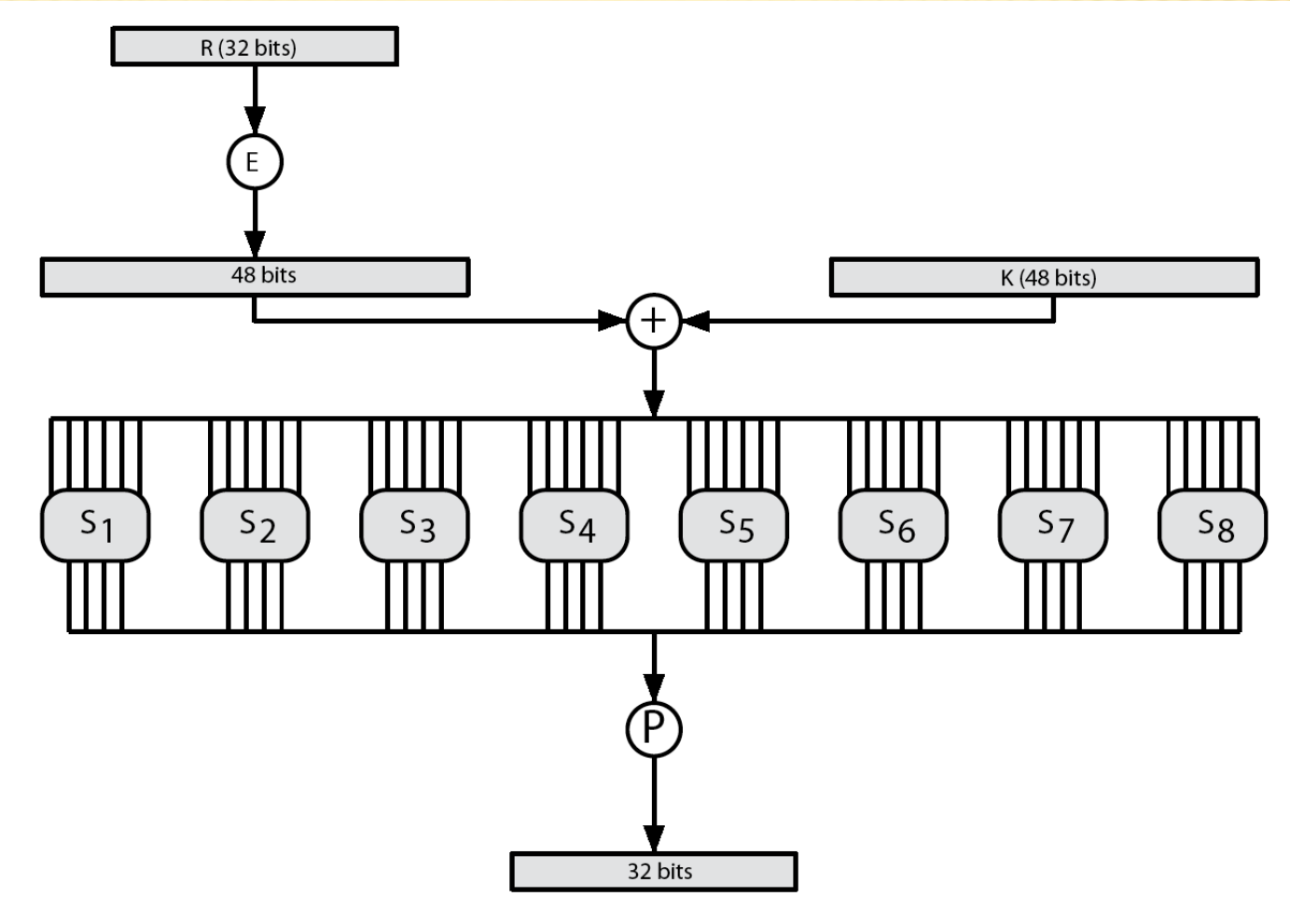
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

s-boxes provide the “confusion” of data and key values, so each S-box output affects as many S-box inputs in the next round as possible, giving “diffusion”.

SUBSTITUTION BOXES S

- Eight S-boxes which map 6 input to 4 bits out
- each S-box is actually 4 little 4 bit boxes
 - outer bits 1 & 6 (**row** bits) select one row of 4
 - inner bits 2-5 (**col** bits) are substituted
 - result is 8 lots of 4 bits, or 32 bits
- row selection depends on both data & key
 - feature known as autoclaving (autokeying)
- example:
$$S(18\ 09\ 12\ 3d\ 11\ 17\ 38\ 39) = 5fd25e03$$

DES ROUND STRUCTURE



S_1

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

 S_2

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

 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

 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

Definition of DES S-Boxes

DISCUSSION:

- ✘ **The substitution consists of a set of (8)S-boxes, each of which accepts 6 bits as input and produces 4 bits as output.**
 - + The first and last bits of the input to box S_i form a 2-bit binary 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 S_1 , 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.

- ✘ **The example lists 8 6-bit values (ie 18 in hex is 011000 in binary, 09 hex is 001001 binary, 12 hex is 010010 binary, 3d hex is 111101 binary etc), each of which is replaced following the process detailed above using the appropriate S-box. ie**
 - + $S_1(011000)$ lookup row 00 col 1100 in S_1 to get 5
 - + $S_2(001001)$ lookup row 01 col 0100 in S_2 to get 15 = f in hex
 - + $S_3(010010)$ lookup row 00 col 1001 in S_3 to get 13 = d in hex
 - + $S_4(111101)$ lookup row 11 col 1110 in S_4 to get 2 etc

DES KEY SCHEDULE

- forms subkeys used in each round
 - Initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
 - 16 stages consisting of:
 - ✗ rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule K**
 - ✗ selecting 24-bits from each half & permuting them by PC2 for use in round function F
- note practical use issues in h/w vs s/w

KEY GENERATION

(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 EXAMPLE

Round	K_i	L_i	R_i
IP		5a005a00	3cf03c0f
1	1e030f03080d2930	3cf03c0f	bad22845
2	0a31293432242318	bad22845	99e9b723
3	23072318201d0c1d	99e9b723	0bae3b9e
4	05261d3824311a20	0bae3b9e	42415649
5	3325340136002c25	42415649	18b3fa41
6	123a2d0d04262a1c	18b3fa41	9616fe23
7	021f120b1c130611	9616fe23	67117cf2
8	1c10372a2832002b	67117cf2	c11bfc09
9	04292a380c341f03	c11bfc09	887fbc6c
10	2703212607280403	887fbc6c	600f7e8b
11	2826390c31261504	600f7e8b	f596506e
12	12071c241a0a0f08	f596506e	738538b8
13	300935393c0d100b	738538b8	c6a62c4e
14	311e09231321182a	c6a62c4e	56b0bd75
15	283d3e0227072528	56b0bd75	75e8fd8f
16	2921080b13143025	75e8fd8f	25896490
IP ⁻¹		da02ce3a	89ecac3b

AVALANCHE EFFECT

- ✘ key desirable property of encryption alg
- ✘ where a change of **one** input or key bit results in changing approx **half** output bits
- ✘ making attempts to “home-in” by guessing keys impossible
- ✘ DES exhibits strong avalanche

STRENGTH OF DES – KEY SIZE

- ✘ 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values
- ✘ brute force search looks hard
- ✘ recent advances have shown is possible
 - + in 1997 on Internet in a few months
 - + in 1998 on dedicated h/w (EFF) in a few days
 - + in 1999 above combined in 22hrs!
- ✘ still must be able to recognize plaintext
- ✘ must now consider alternatives to DES