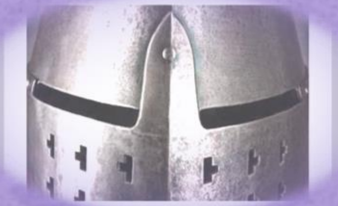


Cryptography and Network Security

Eighth Edition

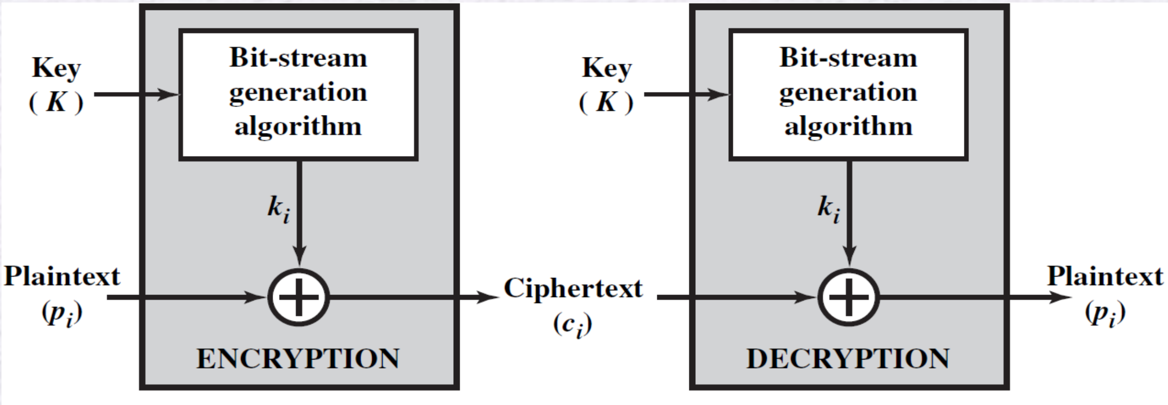
by William Stallings

Chapter 4

Block Ciphers and the Data Encryption Standard

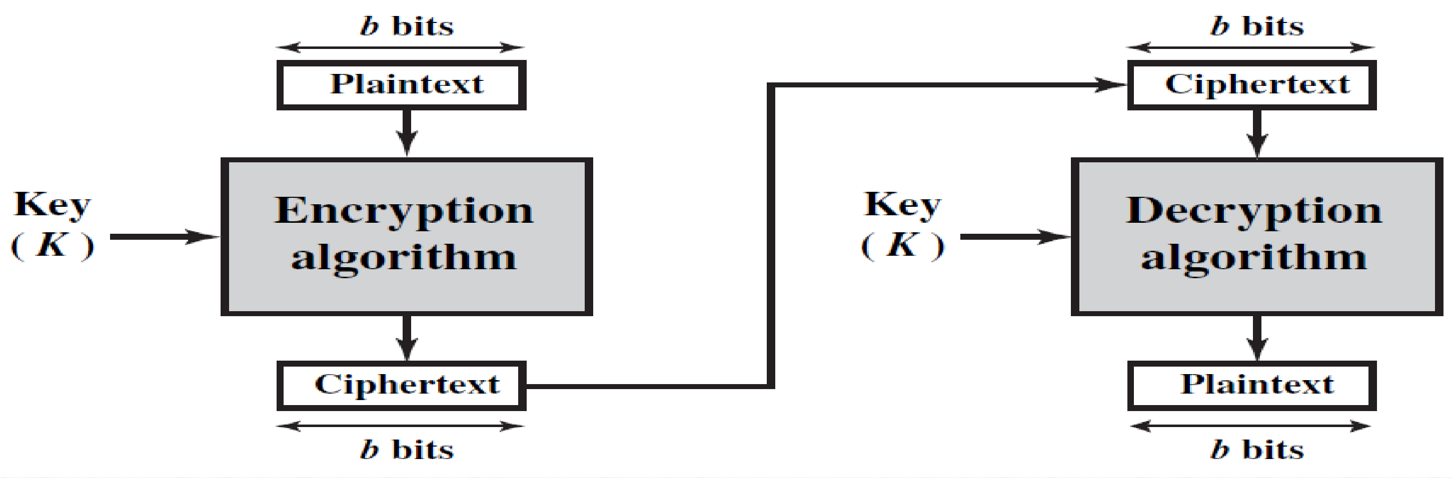
Stream Cipher

- Stream cipher encrypts a digital data stream one bit or one byte at a time.
  - Example, Autokeyed Vigenère cipher, Vernam cipher, and one-time pad version of the Vernam cipher → the ideal case, in which the keystream (k_i) is as long as the plaintext bit stream (p_i).
- If the cryptographic keystream is random, then this cipher is **unbreakable**.
- The keystream must be provided to both users in advance via some independent and secure channel. This introduces insurmountable logistical problems if the intended data traffic is very large.
- Accordingly, for practical reasons, the bit-stream generator must be implemented as an algorithmic procedure, so that the cryptographic bit stream can be produced by both users.
- It must be computationally impractical to predict future portions of the bit stream based on previous portions of the bitstream.



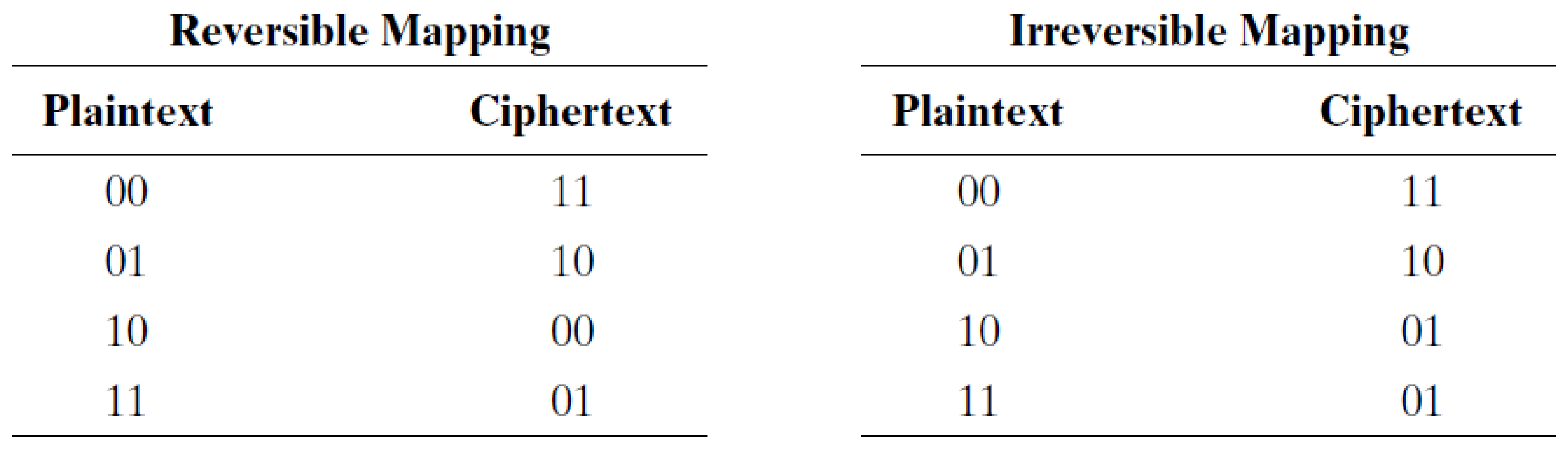
Block Cipher

- In Block Cipher, 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.
- As with a stream cipher, the two users share a symmetric encryption key.



Motivation for the Feistel Cipher Structure

- A **block cipher** operates on a plaintext block of n bits to produce a ciphertext block of n bits → there are 2^n possible different plaintext blocks, and each must produce a unique ciphertext block; **reversible** → for decryption to be possible.



Reversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	00
11	01

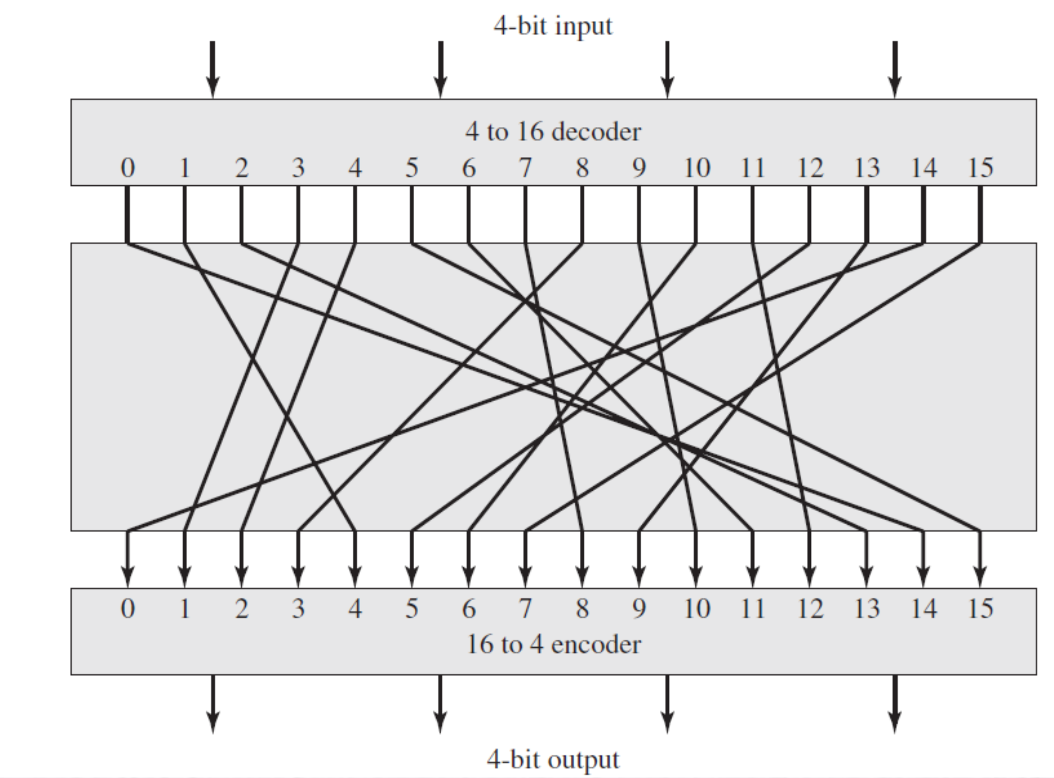
Irreversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	01
11	01

Motivation for the Feistel Cipher Structure

❑ General n-bit-n-bit Block Substitution (shown with n = 4)

➢ This figure illustrates the logic of a general substitution cipher for **n = 4.** A **4**-bit input produces one of **16** possible input states → is mapped by the substitution cipher → a unique one of **16** possible output states; represented by **4** ciphertext bits.

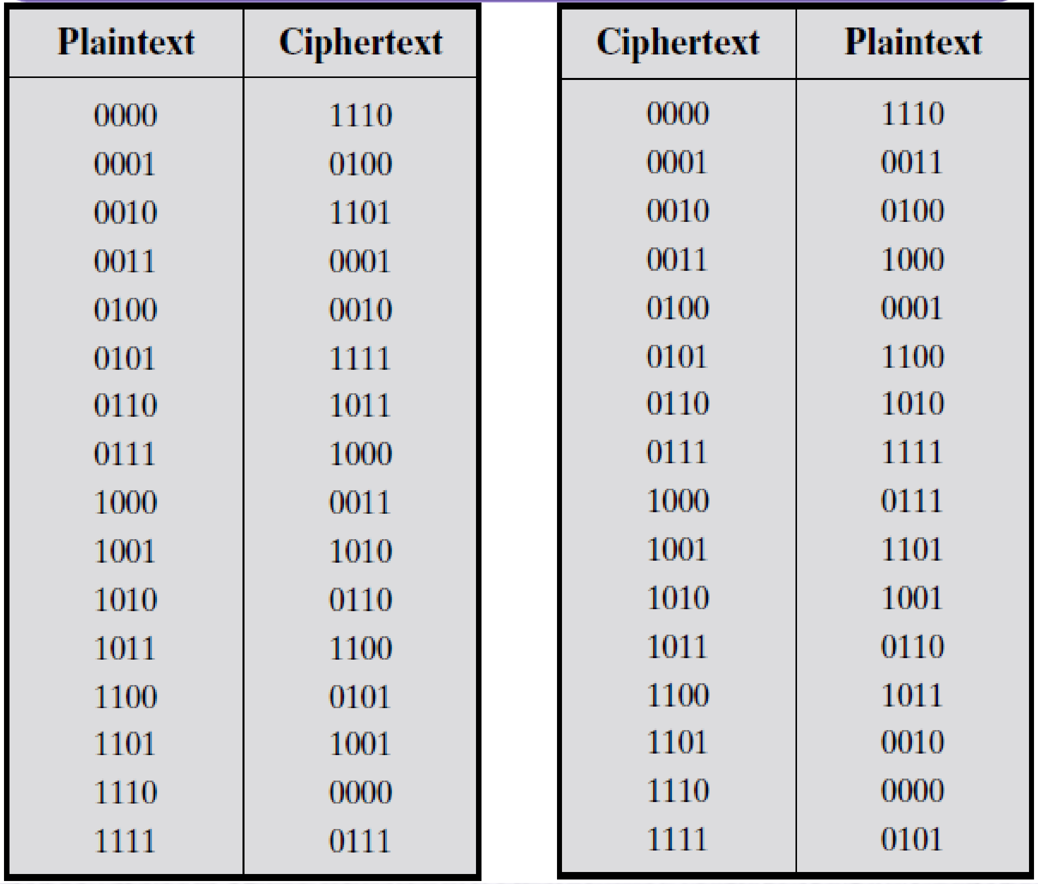


Motivation for the Feistel Cipher Structure

❑ General n-bit-n-bit Block Substitution (shown with n = 4)

- These tables can be used to define any reversible mapping between plaintext and ciphertext.
- For such a transformation, the mapping itself constitutes the key.
- Feistel refers to this as the ideal block cipher; reversible → 2^n! possible transformations, mappings, or keys.
- The key determines the specific mapping from among all possible mappings. Then the required key length is (4 bits) × (2^4 rows) = 64 bits → impractical for large values of n.
- Feistel proposed an approximation to the ideal block cipher by utilizing the concept of a product cipher.

Table: Encryption and decryption tables for substitution cipher of the previous Figure.



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

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

Feistel Cipher

- Feistel proposed the use of a cipher that alternates substitutions and permutations (i.e., product cipher)

Substitutions

- **Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements**

Permutation

- **A sequence of plaintext elements is replaced by a permutation of that sequence.**
- No elements are added or deleted or replaced in the sequence.

- The product cipher alternates ***confusion*** and ***diffusion*** functions. → to **thwart attempts to cryptanalysis**.

Diffusion and Confusion

Diffusion

- **The statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext** **→ to thwart attempts to cryptanalysis**
- This is achieved by having each plaintext digit affect the value of many ciphertext digits; equivalent to having each ciphertext digit be affected by many plaintext digits

Confusion

- **Seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible.** **→ again to thwart attempts to discover the key.**
- Even if the attacker can get some handle on the statistics of the ciphertext, the way in which the key was used to produce that ciphertext is so complex as to make it difficult to deduce the key

- **Feistel Cipher Structure**

(DES uses this structure)

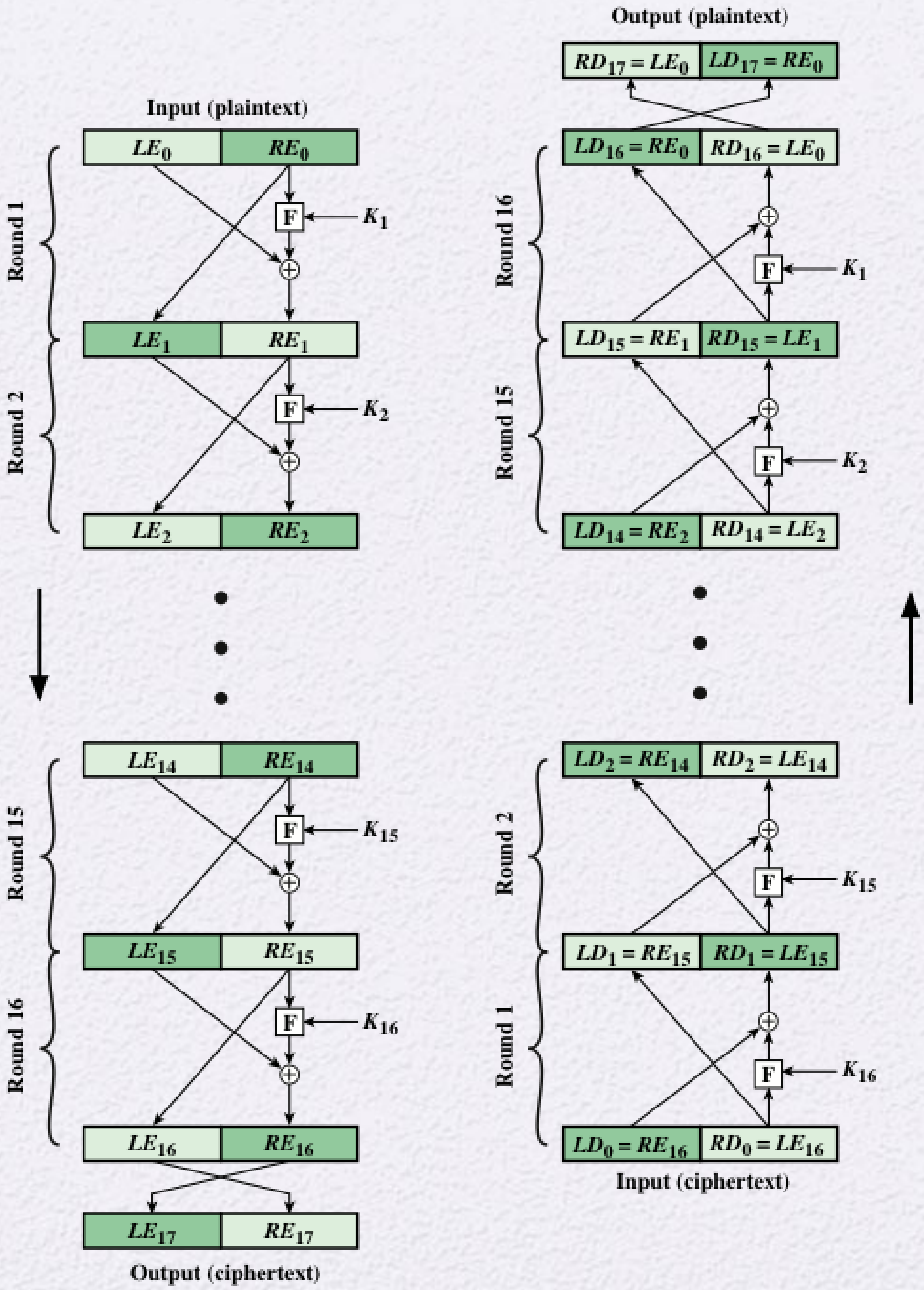
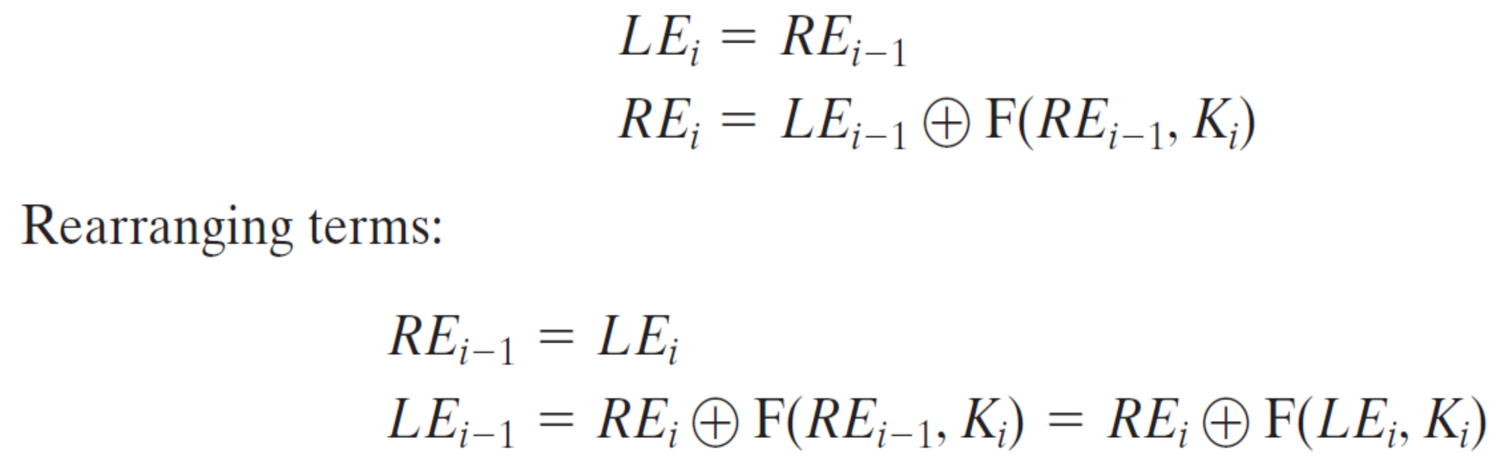


Figure 4.3 Feistel Encryption and Decryption (16 rounds)

Feistel (DES) Decryption Equation



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

Rearranging terms:

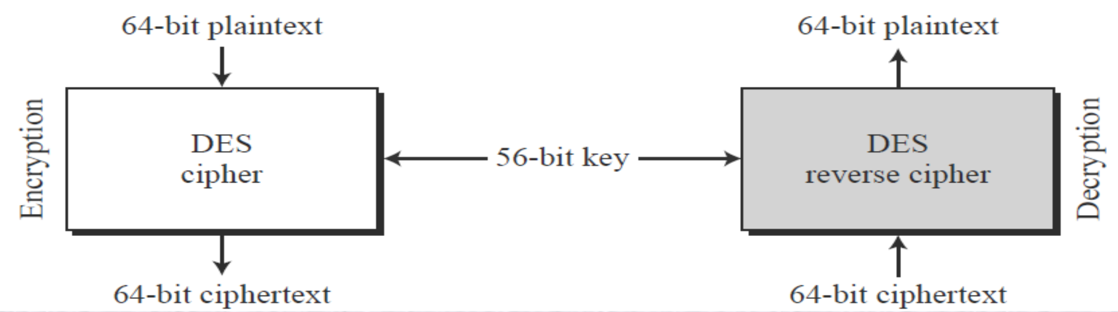
$$RE_{i-1} = LE_i$$
$$LE_{i-1} = RE_i \oplus \mathrm{F}(RE_{i-1}, K_i) = RE_i \oplus \mathrm{F}(LE_i, K_i)$$

- **These equations prove that DES decryption is an inverse process of DES encryption.**

Example

Data Encryption Standard (DES)

- Issued in 1977 by the National Bureau of Standards (NIST now) as Federal Information Processing Standard 46
- Was the most widely used encryption scheme until the introduction of the Advanced Encryption Standard (AES) in 2001
- In DES,
    - Data are encrypted in 64-bit blocks using a 56-bit key
    - The algorithm transforms 64-bit input plaintext in a series of steps into a 64-bit output ciphertext.
    - The same steps, with the same key, are used to reverse the encryption



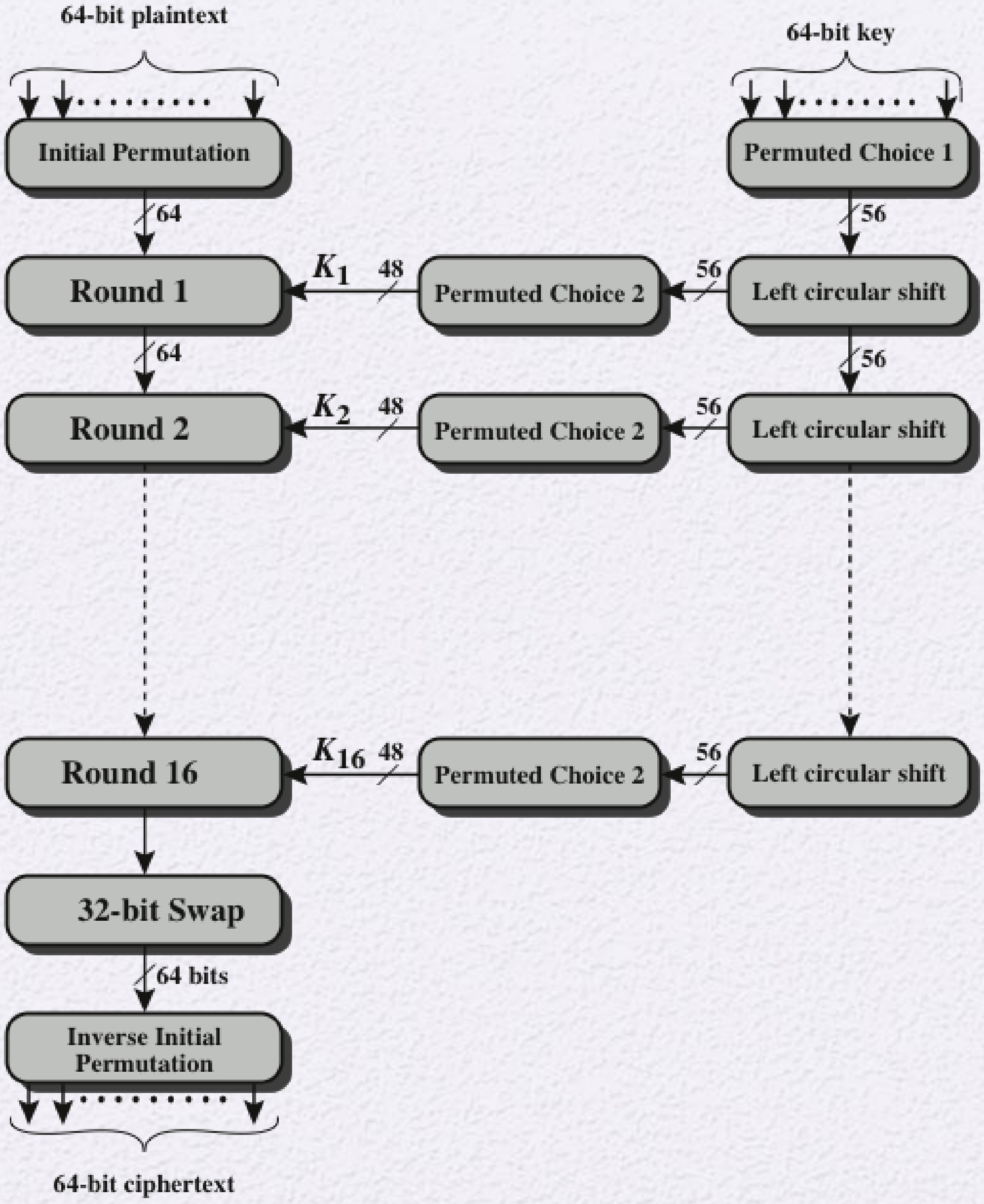
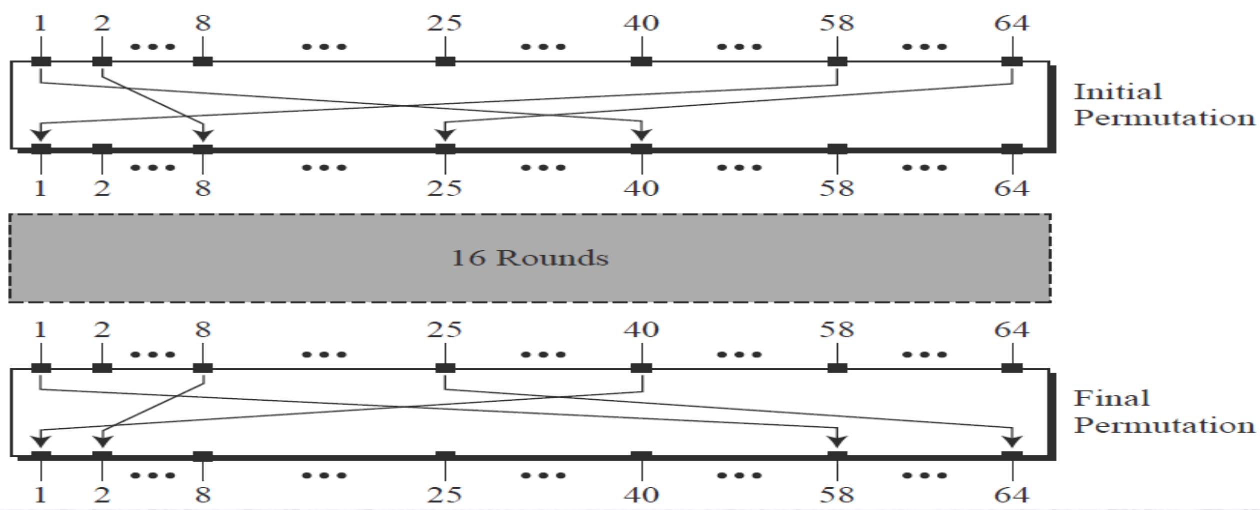


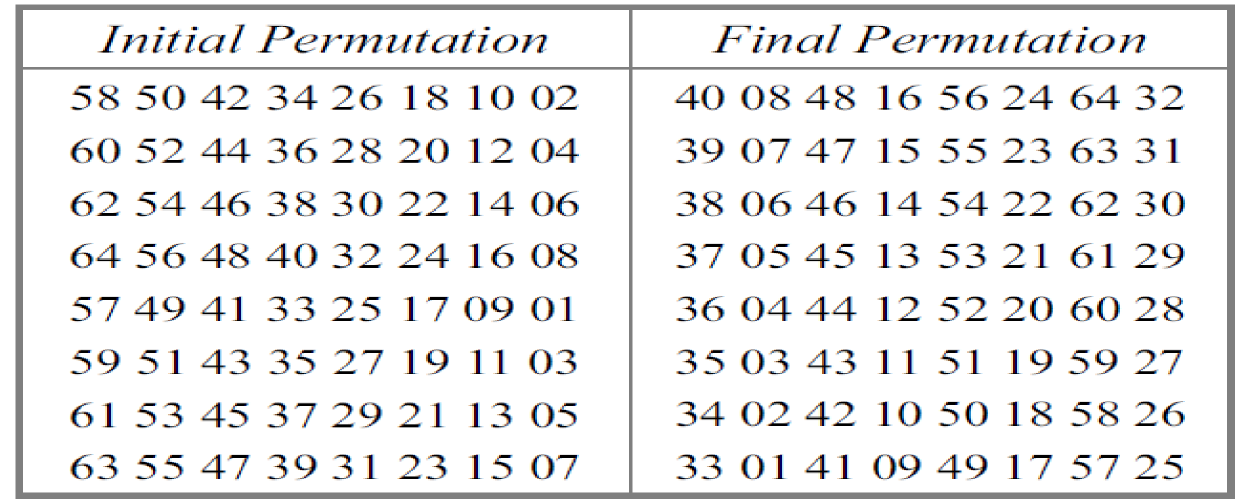
Figure 4.5 General Depiction of DES Encryption Algorithm

Initial and Final Permutations

- Each of these permutations takes a 64-bit input and permutes them according to a predefined rule.
- These permutations are keyless straight permutations that are the inverse of each other.



Initial and Final Permutation Tables



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

Initial and Final Permutations

- **Example,**

Using the initial permutation table, determine the output of the initial permutation box when the input is given in hexadecimal as: `0X0002 0000 0000 0001`

✓ **Solution**

- The input has only two **1**s (bits 15 and bit 64)
- From the previous table, 15 → 63 and 64 → 25
- Then, the output is `0x0000 0080 0000 0002`

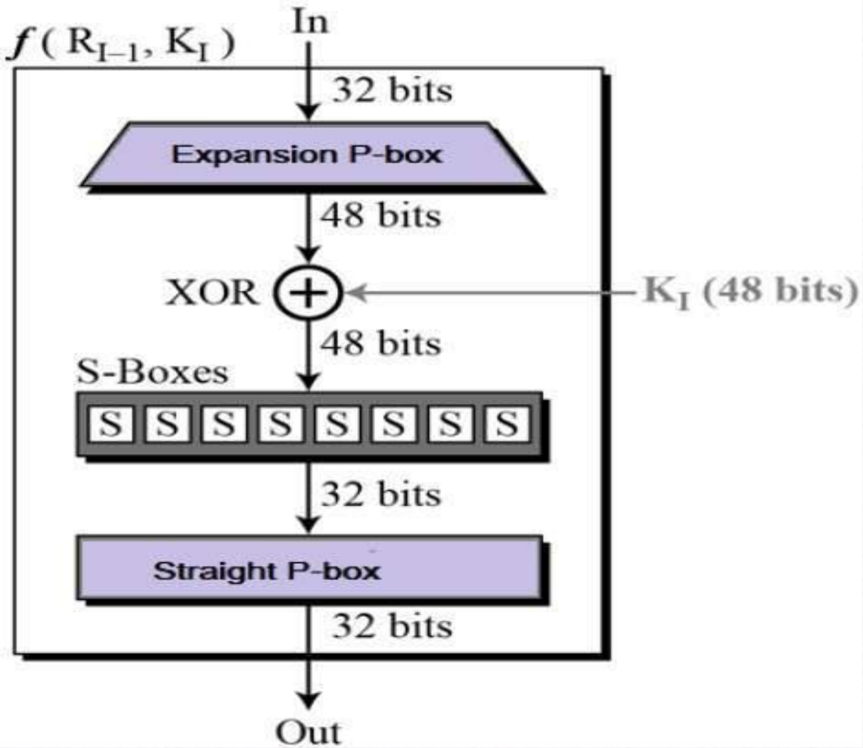
DES Function

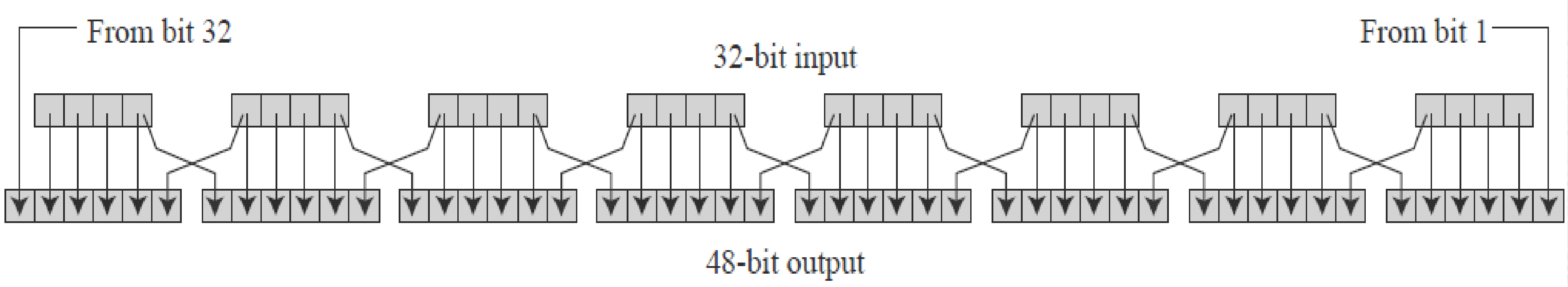
❑ DES Function

It applies a 48-bit key to the rightmost 32 bits (R_{i-1}) to produce a 32-bit output.

- **Expansion P-box**

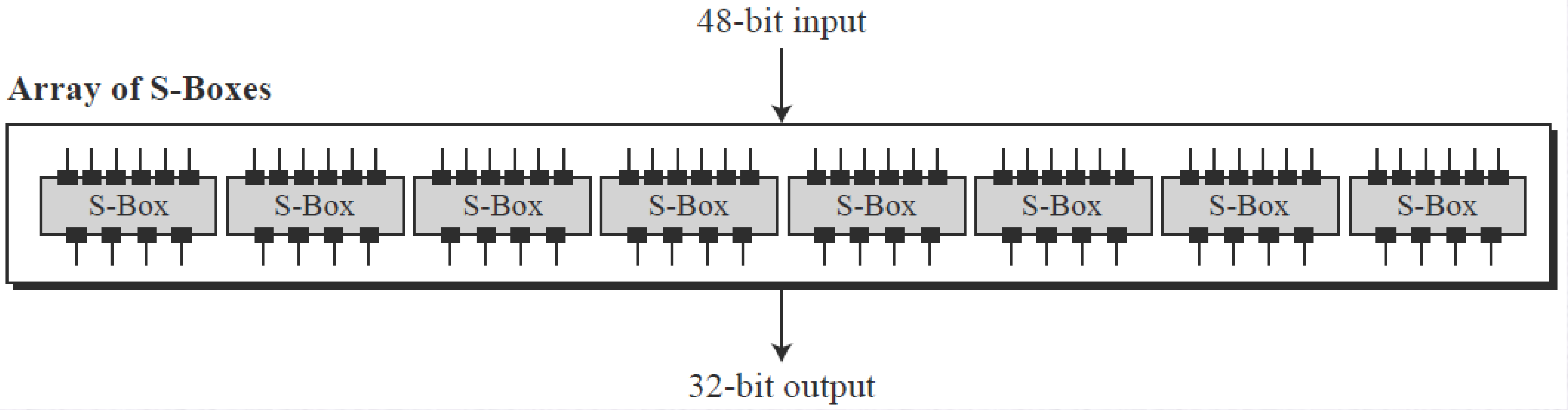
  - Expansion permutation





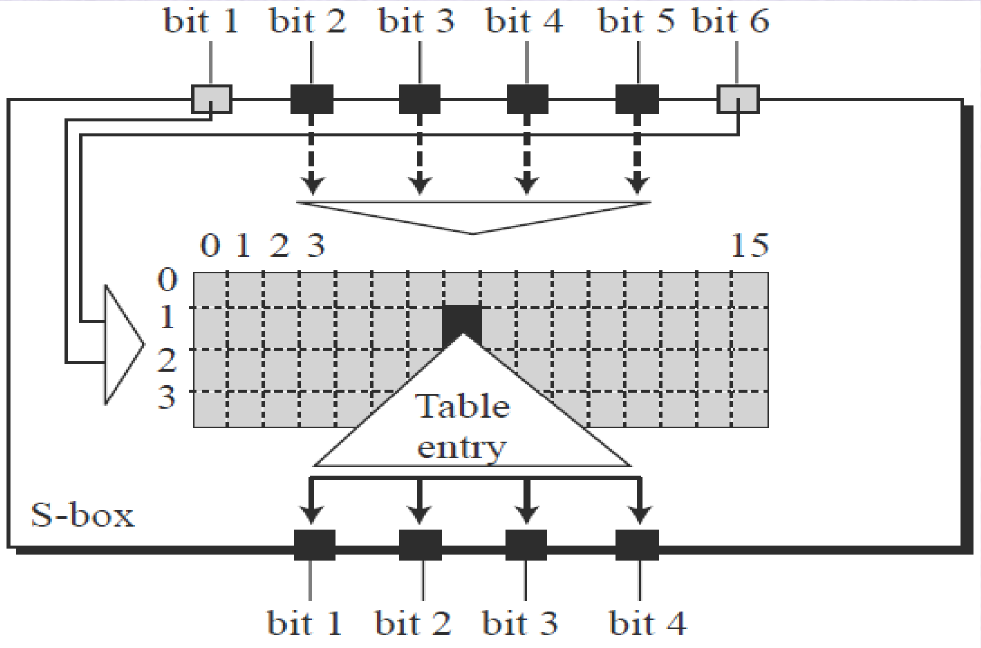
DES Function

- **S-Boxes**
  - Substitution-boxes do the real mixing (**confusion**).
  - DES uses **8** S-boxes, each with a **6-**bit input and a **4-**bit output.



DES Function

- ❑ DES Function
  - ➢ **S-Boxes**
    - The **48**-bit data from XOR is divided into **eight** **6**-bit chunks, and each chunk is fed into a box → The result of each box is **4**-bit; (for **8** boxes→ **8** × **4** = **32** bits).



- The substitution in each box follows a pre-determined rule based on a **4**-row by **16**- column table.
- The combination of bits **1** and **6** of the input defines **one** of **4** rows.
- the combination of bits **2** through **5** defines **one** of the **16** columns.

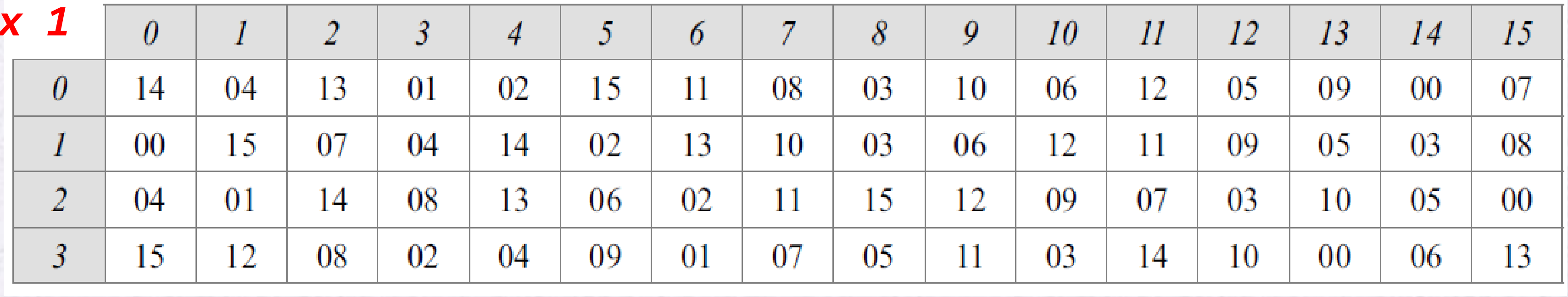
DES Function

❑ DES Function

➢ S-Boxes

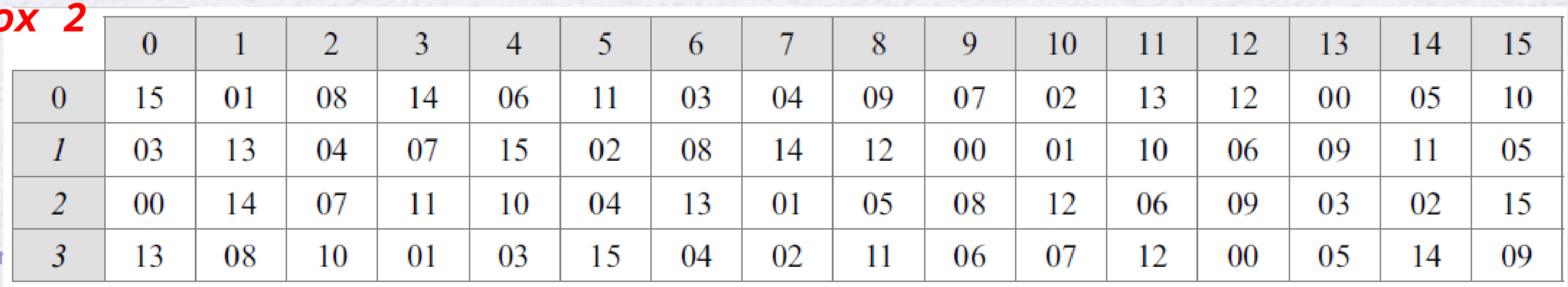
- Because each S-box has its own table, we need eight tables.
- For example,

S-box 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	10	03	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

S-box 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

DES Function

❑ DES Function

➢ **S-Boxes**

- Example,

If the input to S-box 1 is **`100011`**. What is the output?

✓ Solution

`100011`

`11` defines the row ; **`3`**

The remaining bits are **`0001`** defines the column; **`1`**

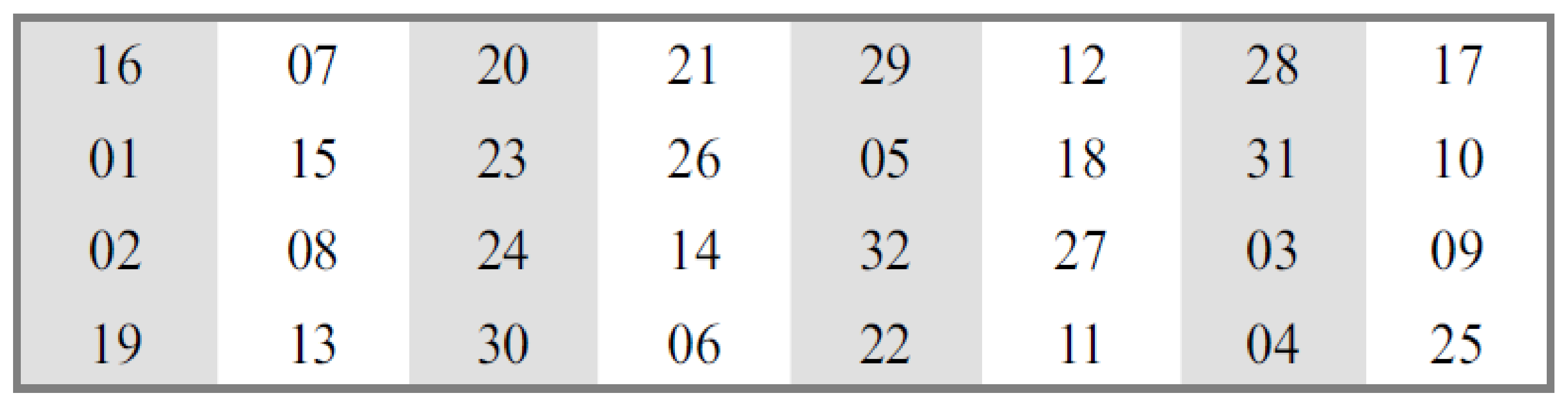
The result is **`12`** in decimal, which in binary is **`1100`**

DES Function

❑ DES Function

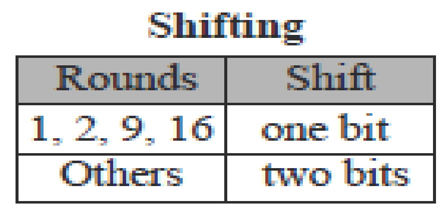
➢ **Straight P-box**

- Straight permutation; **32**-bit input → **32**-bit output.
- Example of **Straight permutation table**



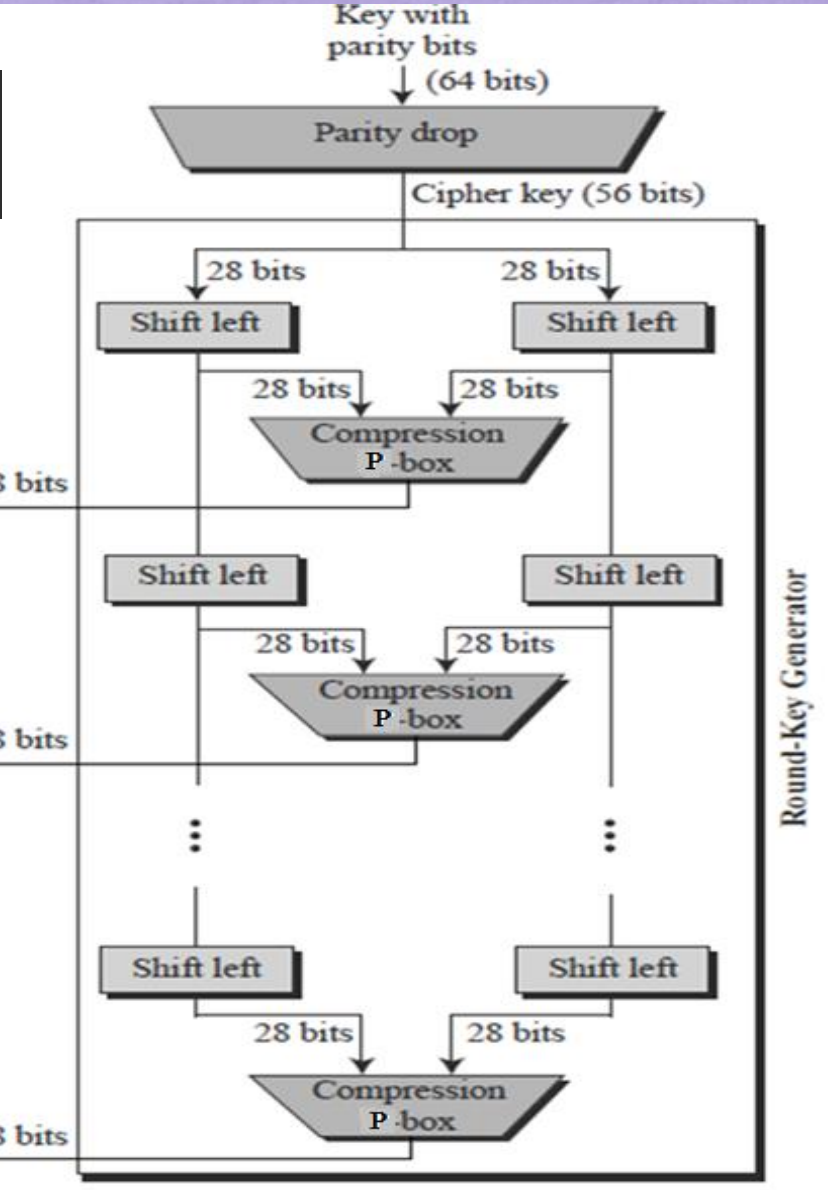
16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

DES Function



Shifting

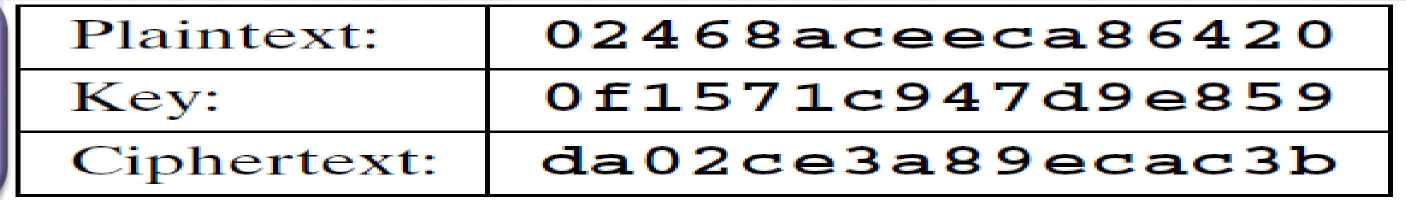
Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits



- ❑ DES Function
  - ➢ **Key Generation**
    - ▪ **Parity Drop**
      - – The preprocess before key expansion; compression transposition step.
      - – It drops the parity bits (bits **8, 16, 24, 32, …, 64**) from the **64**-bit key and **permutes** the rest of the bits according to the flowing Table.

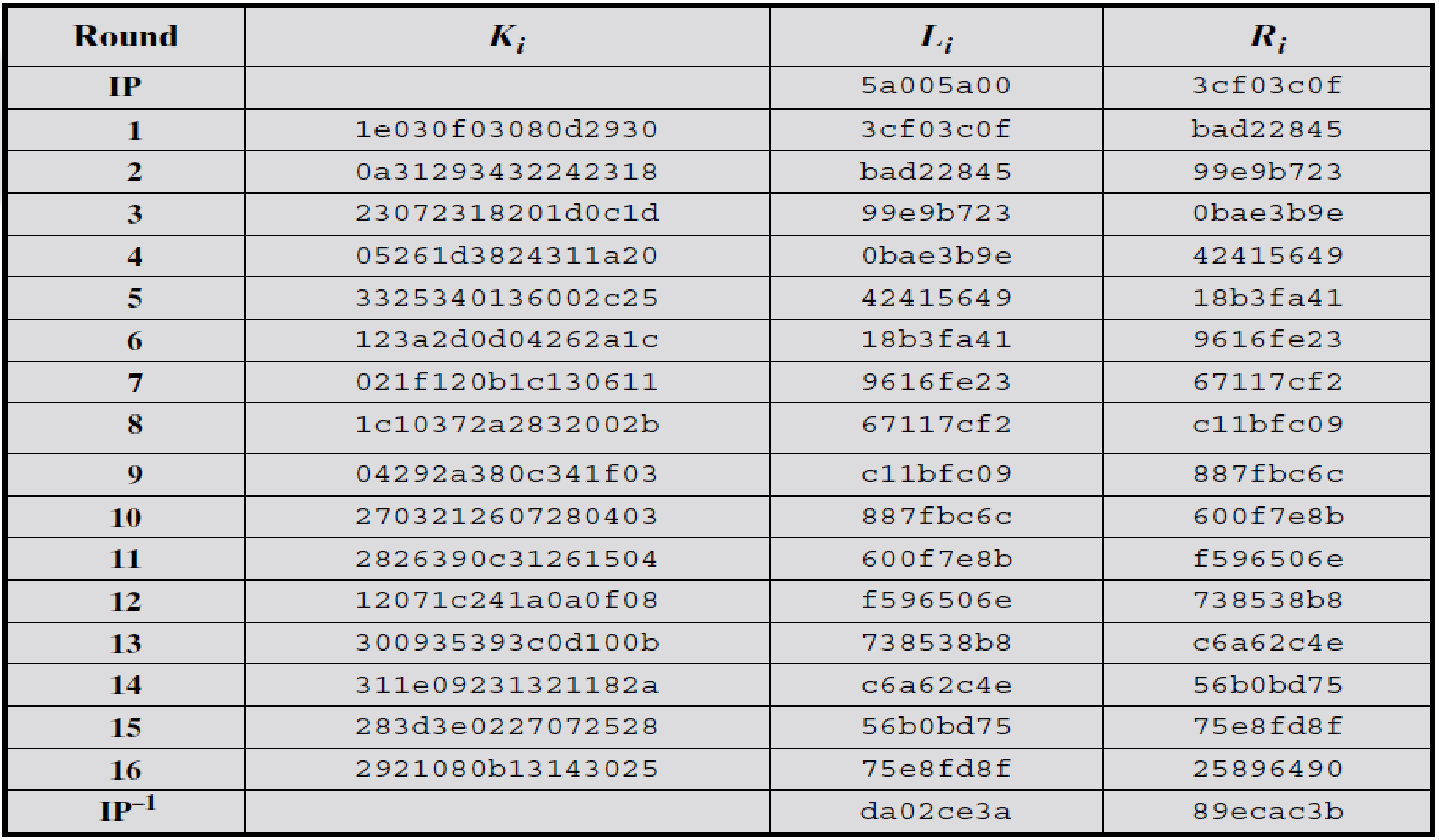
DES Example

The plaintext, key, and resulting ciphertext in hexadecimal



Plaintext:	02468aceeca86420
Key:	0f1571c947d9e859
Ciphertext:	da02ce3a89ecac3b

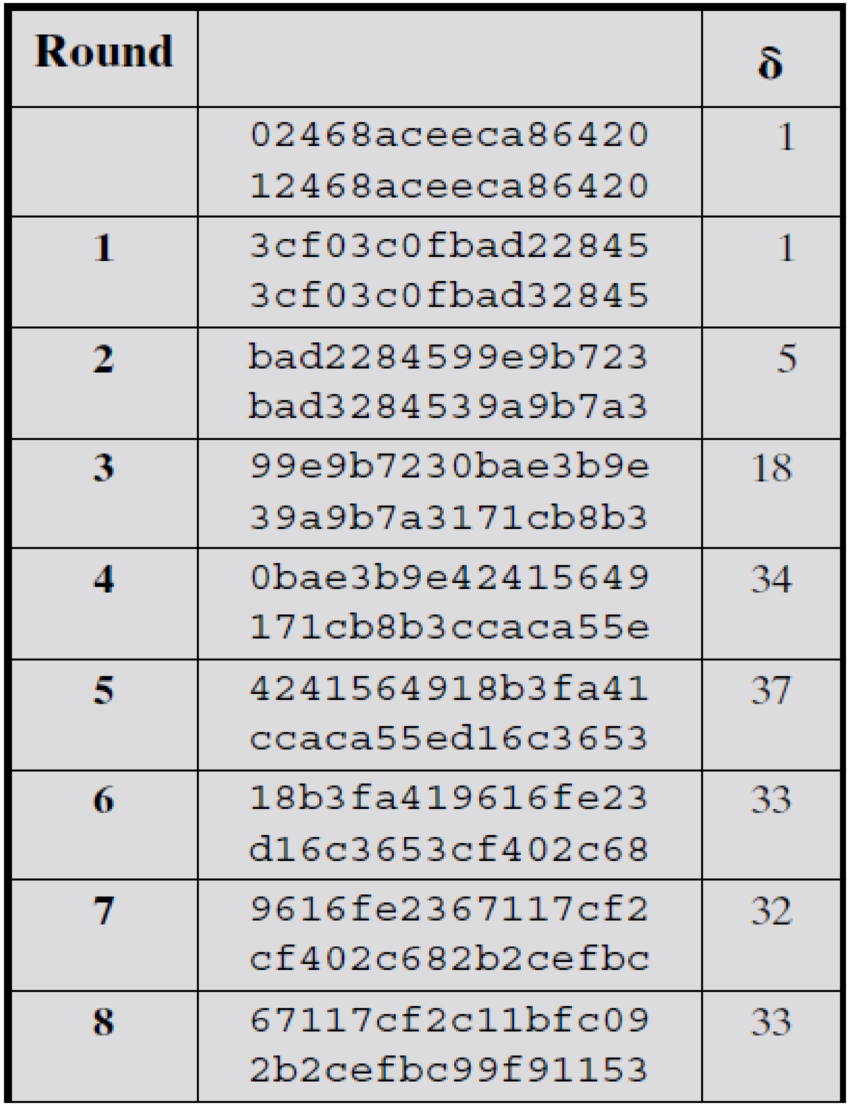
The progression of DES algorithm at each round.



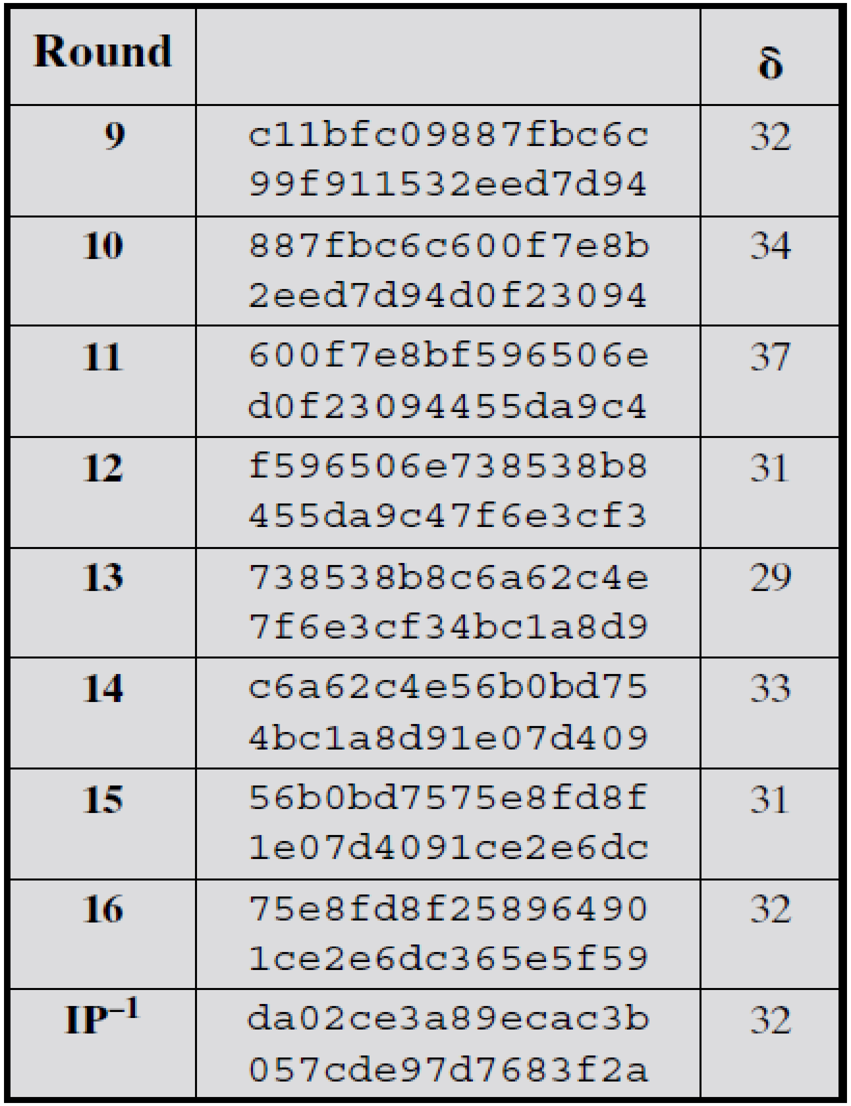
Round	K_i	L_i	R_i
IP		5a005a00	3cf03c0f
1	1e030f03080d2930	3cf03c0f	bad22845
2	0a31293432242318	bad22845	99e9b723
3	23072318201d0c1d	99e9b723	0bae3b9e
4	05261d3824311a20	0bae3b9e	42415649
5	3325340136002c25	42415649	18b3fa41
6	123a2d0d04262a1c	18b3fa41	9616fe23
7	021f120b1c130611	9616fe23	67117cf2
8	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

The Avalanche Effect

- The Avalanche Effect
  - Refers to that a small change in either the plaintext or the key should produce a significant change in the ciphertext.
  - Using the previous example, the following table shows the result when the 4th bit of the plaintext is changed, so that the plaintext is `12468aceeca86420`.
  - The 2nd column of the table shows the intermediate 64-bit values at the end of each round for the two plaintexts.
  - The 3rd column shows the number of bits that differ between the two intermediate values.



Round		δ
	02468aceeca86420 12468aceeca86420	1
1	3cf03c0fbad22845 3cf03c0fbad32845	1
2	bad2284599e9b723 bad3284539a9b7a3	5
3	99e9b7230bae3b9e 39a9b7a3171cb8b3	18
4	0bae3b9e42415649 171cb8b3ccaca55e	34
5	4241564918b3fa41 ccaca55ed16c3653	37
6	18b3fa419616fe23 d16c3653cf402c68	33
7	9616fe2367117cf2 cf402c682b2cefbc	32
8	67117cf2c11bfc09 2b2cefbc99f91153	33

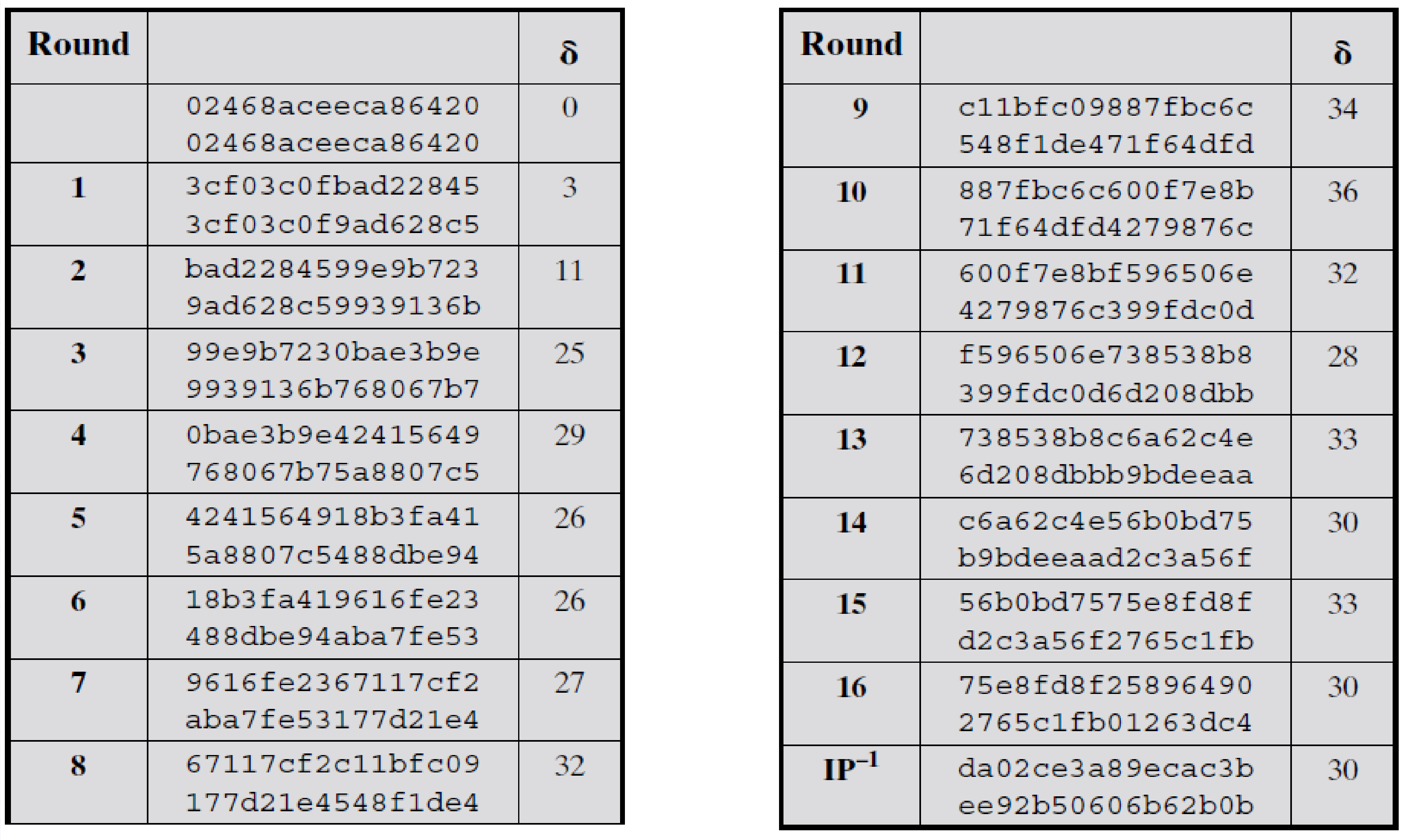


Round		δ
9	c11bfc09887fbc6c 99f911532eed7d94	32
10	887fbc6c600f7e8b 2eed7d94d0f23094	34
11	600f7e8bf596506e d0f23094455da9c4	37
12	f596506e738538b8 455da9c47f6e3cf3	31
13	738538b8c6a62c4e 7f6e3cf34bc1a8d9	29
14	c6a62c4e56b0bd75 4bc1a8d91e07d409	33
15	56b0bd7575e8fd8f 1e07d4091ce2e6dc	31
16	75e8fd8f25896490 1ce2e6dc365e5f59	32
IP^{-1}	da02ce3a89ecac3b 057cde97d7683f2a	32

Avalanche Effect in DES: Change in Plaintext

(Table can be found on page 107 in the textbook)

- The following table shows a similar test using two keys that differ in only the 4th bit position; the original key, `0f1571c947d9e859`, and the altered key, `1f1571c947d9e859`.



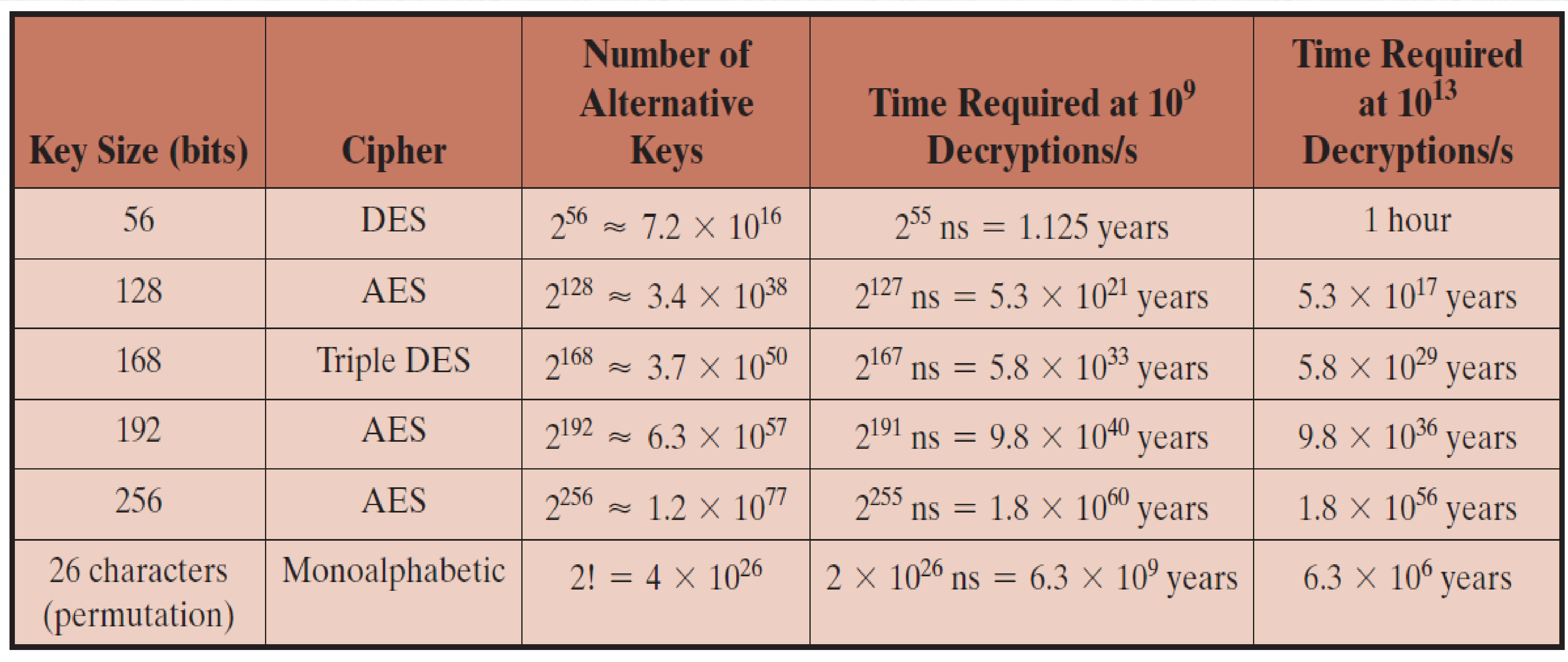
Round		δ
	02468aceeca86420 02468aceeca86420	0
1	3cf03c0fbad22845 3cf03c0f9ad628c5	3
2	bad2284599e9b723 9ad628c59939136b	11
3	99e9b7230bae3b9e 9939136b768067b7	25
4	0bae3b9e42415649 768067b75a8807c5	29
5	4241564918b3fa41 5a8807c5488dbe94	26
6	18b3fa419616fe23 488dbe94aba7fe53	26
7	9616fe2367117cf2 aba7fe53177d21e4	27
8	67117cf2c11bfc09 177d21e4548f1de4	32
9	c11bfc09887fbc6c 548f1de471f64dfd	34
10	887fbc6c600f7e8b 71f64dfd4279876c	36
11	600f7e8bf596506e 4279876c399fdc0d	32
12	f596506e738538b8 399fdc0d6d208dbb	28
13	738538b8c6a62c4e 6d208dbbb9bdeeaa	33
14	c6a62c4e56b0bd75 b9bdeeaad2c3a56f	30
15	56b0bd7575e8fd8f d2c3a56f2765c1fb	33
16	75e8fd8f25896490 2765c1fb01263dc4	30
IP^{-1}	da02ce3a89ecac3b ee92b50606b62b0b	30

Avalanche Effect in DES: Change in Key

(Table can be found on page 107 in the textbook)

Table 4.5

Average Time Required for Exhaustive Key Search



Key Size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 Decryptions/s	Time Required at 10^{13} Decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	2^{167} ns = 5.8×10^{33} years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	2^{191} ns = 9.8×10^{40} years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	2^{255} ns = 1.8×10^{60} years	1.8×10^{56} years
26 characters (permutation)	Monoalphabetic	$2! = 4 \times 10^{26}$	2×10^{26} ns = 6.3×10^{9} years	6.3×10^{6} years

Block Cipher Design Principles: Number of Rounds

The greater the number of rounds, the more difficult it is to perform cryptanalysis

In general, the criterion should be that the number of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack

If DES had 15 or fewer rounds, differential cryptanalysis would require less effort than a brute-force key search

Block Cipher Design Principles: Design of Function F

- The heart of a Feistel block cipher is the function F
- The more nonlinear F, the more difficult any type of cryptanalysis will be

Block Cipher Design Principles: Key Schedule Algorithm

- With any Feistel block cipher, the key is used to generate one subkey for each round

- In general, we would like to select subkeys to maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key