BAHRIA UNIVERSITY
Discovering Knowledge

Block Ciphers &
Data Encryption Standard

Data Encryption & Security (CEN-451)

Spring 2025 (BSE-8A&B)

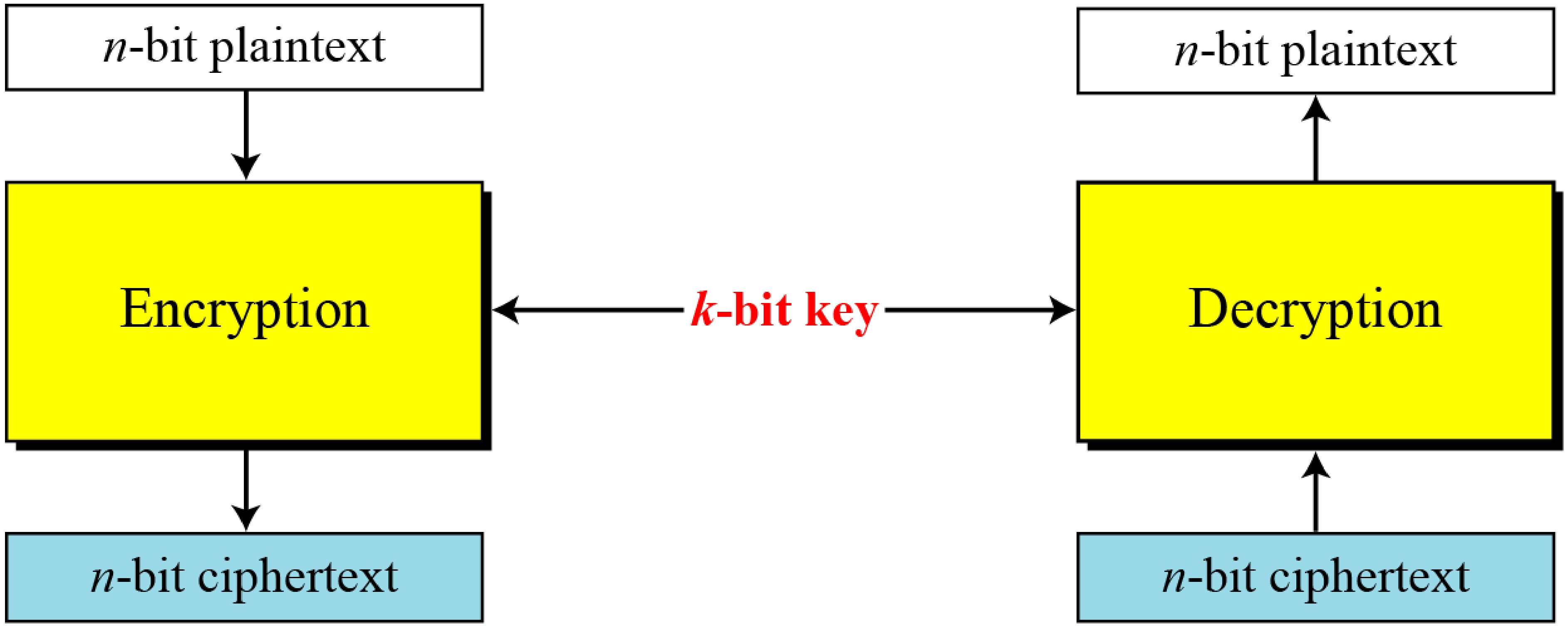
Traditional Ciphers

- Traditional **symmetric-key** ciphers are **character-oriented ciphers**.
- Since information includes numbers, graphics, audio and video, hence we need **bit-oriented ciphers**.
- When converting information (e.g. text) to bits, the number of symbols increase.
- In general, mixing larger number of symbols increases security.

Modern Block Ciphers

- A modern **symmetric-key** block cipher encrypts an **n-bit** block of plaintext or decrypts an **n-bit** block of ciphertext.
- The encryption or decryption algorithm uses a **k-bit** key.
- If the message has fewer than **n** bits, padding must be added to make it an **n-bit** block. *(e.g. padding with 0's only in last block)*
- Common values for **n** are **64, 128, 256** and **512**.

Modern Block Ciphers (Cont.)



Modern Block Ciphers (Cont.)

Block of plaintext is used to produce a ciphertext block of **equal length**

Typically block size of **64, 128, 256 and 512** bits are used

The two users share a **symmetric encryption key**

Majority of **network-based** symmetric cryptographic use block ciphers

Substitution/Transposition in Modern Block Ciphers

- A modern block cipher can be designed to act as a **substitution cipher** or a **transposition cipher**.
- **Case#01:** While using **substitution** cipher, a **64-bit** plaintext block of 12 **0's** and 52 **1's** can be encrypted to a ciphertext block of 34 **0's** and 30 **1's**.
- **Case#02:** While using **transposition** cipher, the same number of **0's** and **1's** would be found in both plaintext and ciphertext.

Product Cipher

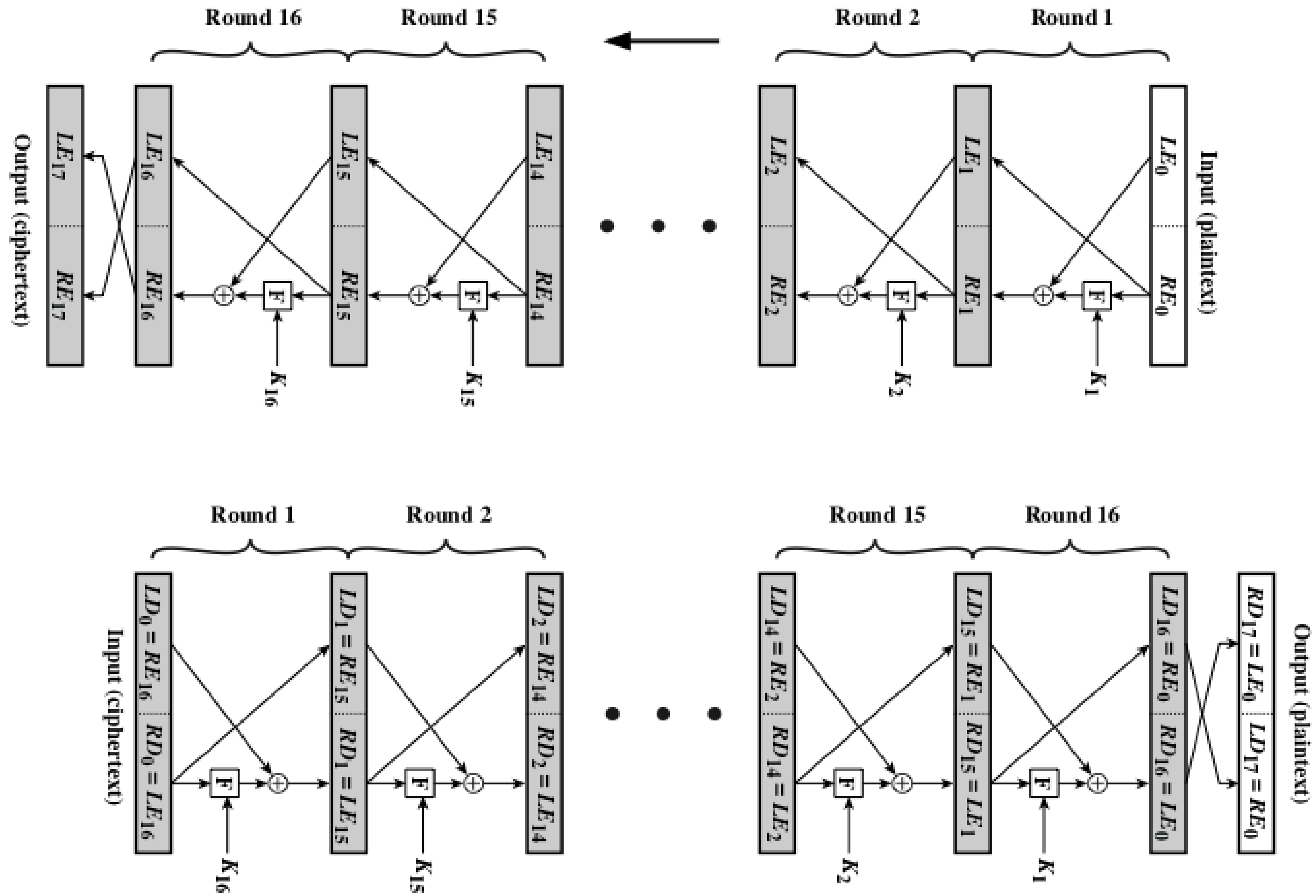
- **Product cipher:** a complex cipher combining **substitution**, **transposition** and other components.

- In product ciphers, two or more simple ciphers are executed in such a way that the product is cryptographically stronger than any of the component ciphers.

- The **product cipher** enables the block ciphers to have two important properties, i.e. **diffusion** and **confusion,** for frustrating the **statistical cryptanalysis**.

Feistel Cipher

- **Feistel** proposed the use of a cipher that alternates **substitutions** and **permutations**.
- The **structure** of Feistel chipher is adopted by many significant symmetric block ciphers that are currently in use.



Round 16
Round 15
Round 2
Round 1
Output (ciphertext)
LE_{17}
RE_{17}
LE_{16}
RE_{16}
LE_{15}
RE_{15}
LE_{14}
RE_{14}
LE_2
RE_2
LE_1
RE_1
LE_0
RE_0
Input (plaintext)
F
K_{16}
K_{15}
K_2
K_1
Round 1
Round 2
Round 15
Round 16
Input (ciphertext)
$LD_0 = RE_{16}$
$RD_0 = LE_{16}$
$LD_1 = RE_{15}$
$RD_1 = LE_{15}$
$LD_2 = RE_{14}$
$RD_2 = LE_{14}$
$LD_{14} = RE_2$
$RD_{14} = LE_2$
$LD_{15} = RE_1$
$RD_{15} = LE_1$
$LD_{16} = RE_0$
$RD_{16} = LE_0$
$RD_{17} = LE_0$
$LD_{17} = RE_0$
Output (plaintext)
K_{16}
K_{15}
K_2
K_1

Feistel Cipher (Cont.)

Main Concepts in Feistel cipher:

- The inputs are a plaintext block of length $2w$ bits and a key K.
- The plaintext block is divided into two halves, LE_0 and RE_0.
- The two halves of the data pass through n rounds of processing and then combine to produce the ciphertext block.
- Each round i has as inputs LE_{i-1} and RE_{i-1} derived from the previous round, as well as a subkey K_i derived from overall K.

Feistel Cipher (Cont.)

Main Concepts in Feistel cipher (Cont.):

- A **substitution** is performed by applying a **round function F** and then taking **exclusive-OR**.
- Following substitution, a **permutation** is performed that consists of the interchange of the two halves of the data.
- All rounds have the same structure.

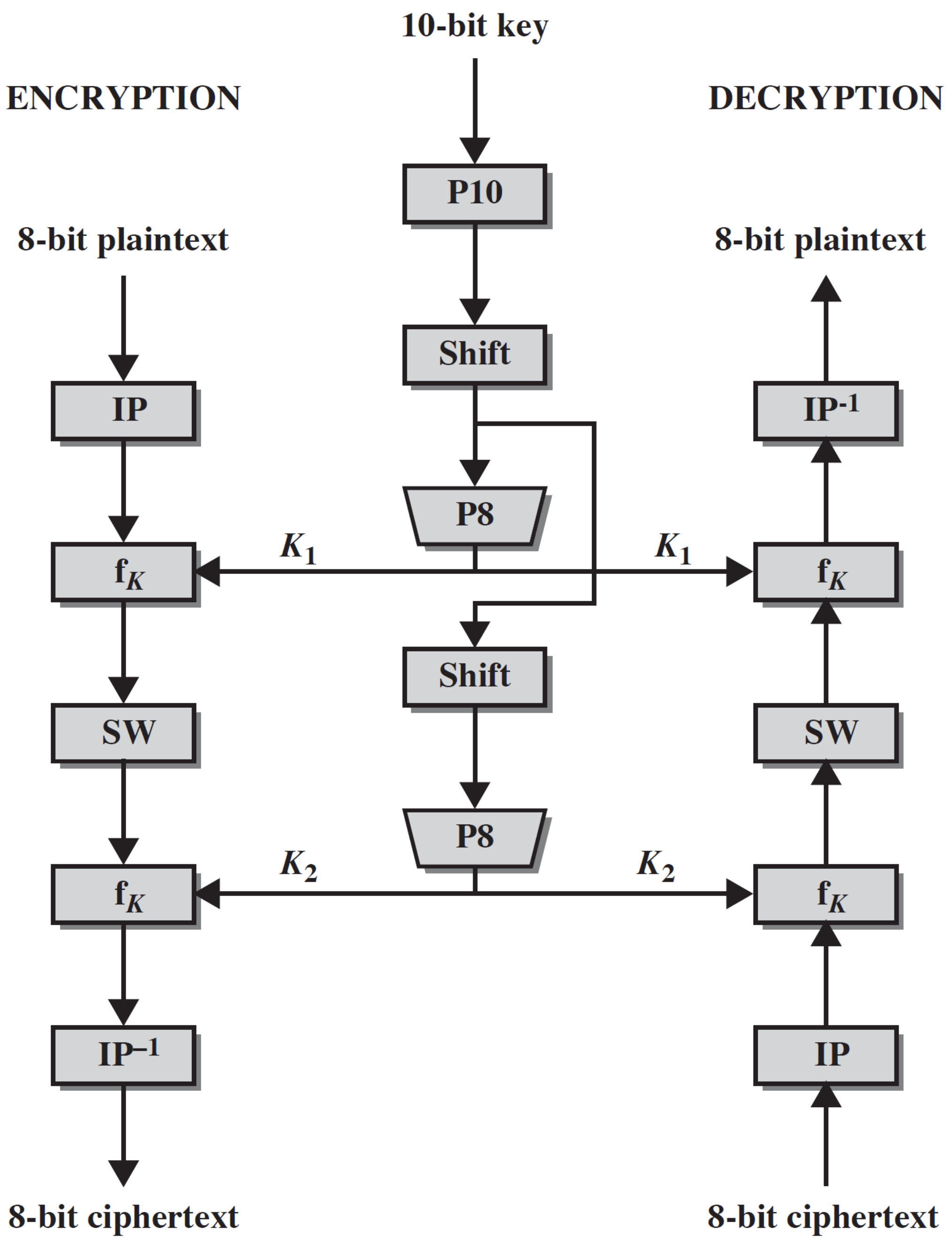
Simplified Data Encryption Standard (S-DES)

S-DES: An Overview

- Until the introduction of the **Advanced Encryption Standard (AES)** in 2001, the **Data Encryption Standard (DES)** was the most widely used encryption scheme.
- **DES** is a modern **symmetric-key block cipher** published by the **National Institute of Standards and Technology (NIST)**.
- A new version of **DES** was issued by the name of **triple DES (3DES)**.

S-DES: An Overview (Cont.)

- A **simplified DES (S-DES)** has similar properties and structure to **DES** with much smaller parameters.
- **Encryption**
  - It takes an **8-bit** block of plain text and a **10-bit** key as input and produces an **8-bit block** of cipher text.
- **Decryption**
  - It takes an **8-bit** block of cipher text and the same **10-bit** key used to produce that Ciphertext as input, and produces the original **8-bit** block of plaintext.



Simplified DES Scheme

S-DES Key Generation

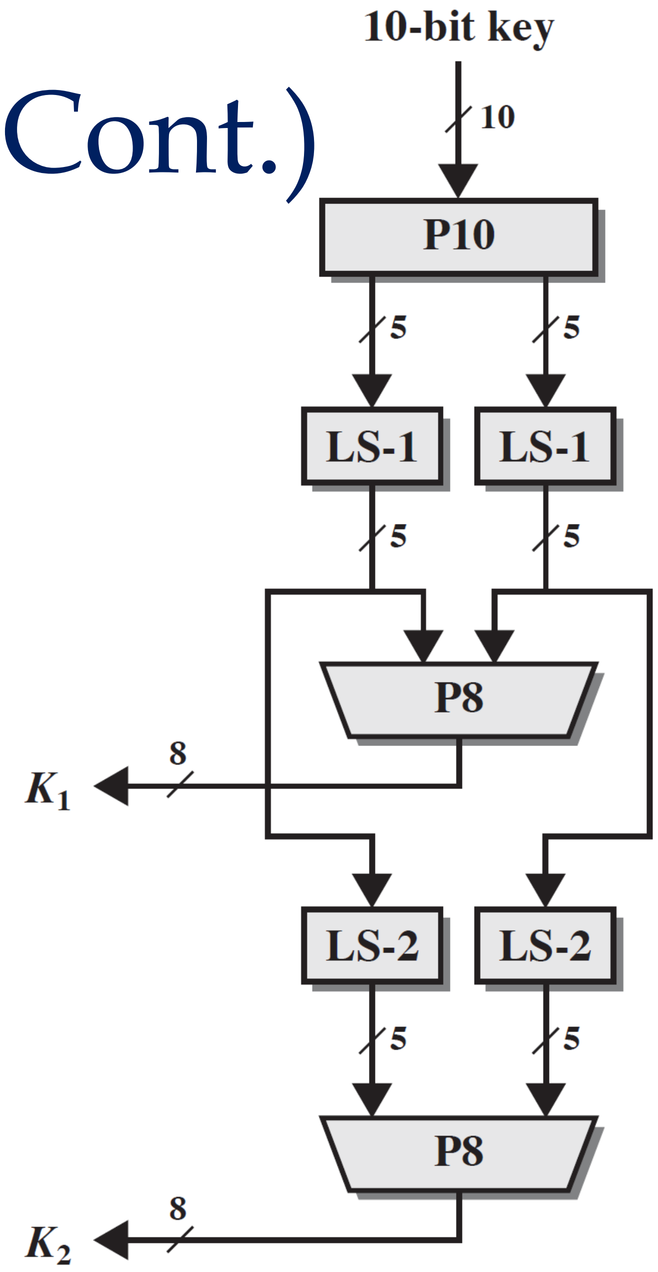
- The keys are produced as:

$$K_1 = P8\ (Shift\ (P10\ (key)))$$

$$K_2 = P8\ (Shift\ (Shift\ (P10\ (key))))$$

S-DES Key Generation (Cont.)

- S-DES depends on the use of a **10-bit** key shared between sender and receiver.
- From this key, two **8-bit** subkeys are produced for use in particular stages of encryption and decryption algorithm.
- The diagram shows the Key Generation for Simplified DES.



S-DES Key Generation (Cont.)

Stages followed to produce the subkeys:

1. Let the **10-bit** key be designated as ***(k*1, *k*2, *k*3, *k*4, *k*5, *k*6, *k*7, *k*8, *k*9, *k*10)**. Then the permutation P10 is defined as:

P10(*k*1, *k*2, *k*3, *k*4, *k*5, *k*6, *k*7, *k*8, *k*9, *k*10) =

(*k*3, *k*5, *k*2, *k*7, *k*4, *k*10, *k*1, *k*9, *k*8, *k*6)

P10									
3	5	2	7	4	10	1	9	8	6

- For example, the key **(1010000010)** is permuted to **(10000 01100)**.

S-DES Key Generation (Cont.)

Stages followed to produce the subkeys (Cont.):

2. Perform a circular left shift **(LS-1)**, or rotation, separately on the first five bits and the second five bits.

- In our example, the result is **(00001 11000)**.

3. Apply **P8**, which picks out and permutes 8 of the 10 bits according to the following rule:

P8							
6	3	7	4	8	5	10	9

- The result is *K*1. In our example, this yields **(10100100)**.

S-DES Key Generation (Cont.)

Stages followed to produce the subkeys (Cont.):

4. Go back to the pair of 5-bit strings produced by the two LS-1 functions and perform a circular left shift of 2 bit positions on each string.

- In our example, the value **(00001 11000)** becomes **(00100 00011)**.

5. Finally, P8 is applied again to produce *K*2.

- In our example, the result is **(01000011)**.

S-DES Encryption Algorithm

The encryption algorithm involves five functions:

1. An initial permutation **(IP)**.
2. A complex function $\mathbf{f_{K1}}$, which involves both substitution and permutation operations and depends on a key input.
3. A simple permutation function **(SW)** that switches the two halves of the data.
4. Again function $\mathbf{f_{K2}}$. ***(The use of multiple stages of substitution and permutation results in a more complex algorithm, which increases the difficulty of cryptanalysis)***
5. Finally, a permutation function that is inverse of the initial permutation **($\mathbf{IP^{-1}}$)**.

S-DES Encryption Algorithm (Cont.)

- We can concisely express the encryption algorithm as a composition of functions:

$$\mathbf{IP^{-1} \circ f_{K_2} \circ SW \circ f_{K_1} \circ IP}$$

$$\mathbf{ciphertext = IP^{-1}\,(f_{K_2}\,(SW\,(f_{K_1}\,(IP\,(plaintext)))))}$$

- Decryption is essentially the reverse of encryption:

$$\mathbf{plaintext = IP^{-1}\,(f_{K_1}\,(SW\,(f_{K_2}\,(IP\,(ciphertext)))))}$$

S-DES Encryption Algorithm (Cont.)

Initial and Final Permutations

- Input to algorithm is an 8-bit block of plaintext.
- Permute the 8-bit block using the **IP function**.

IP							
2	6	3	1	4	8	5	7

- At end of the algorithm, the **IP^{-1} function** is used.

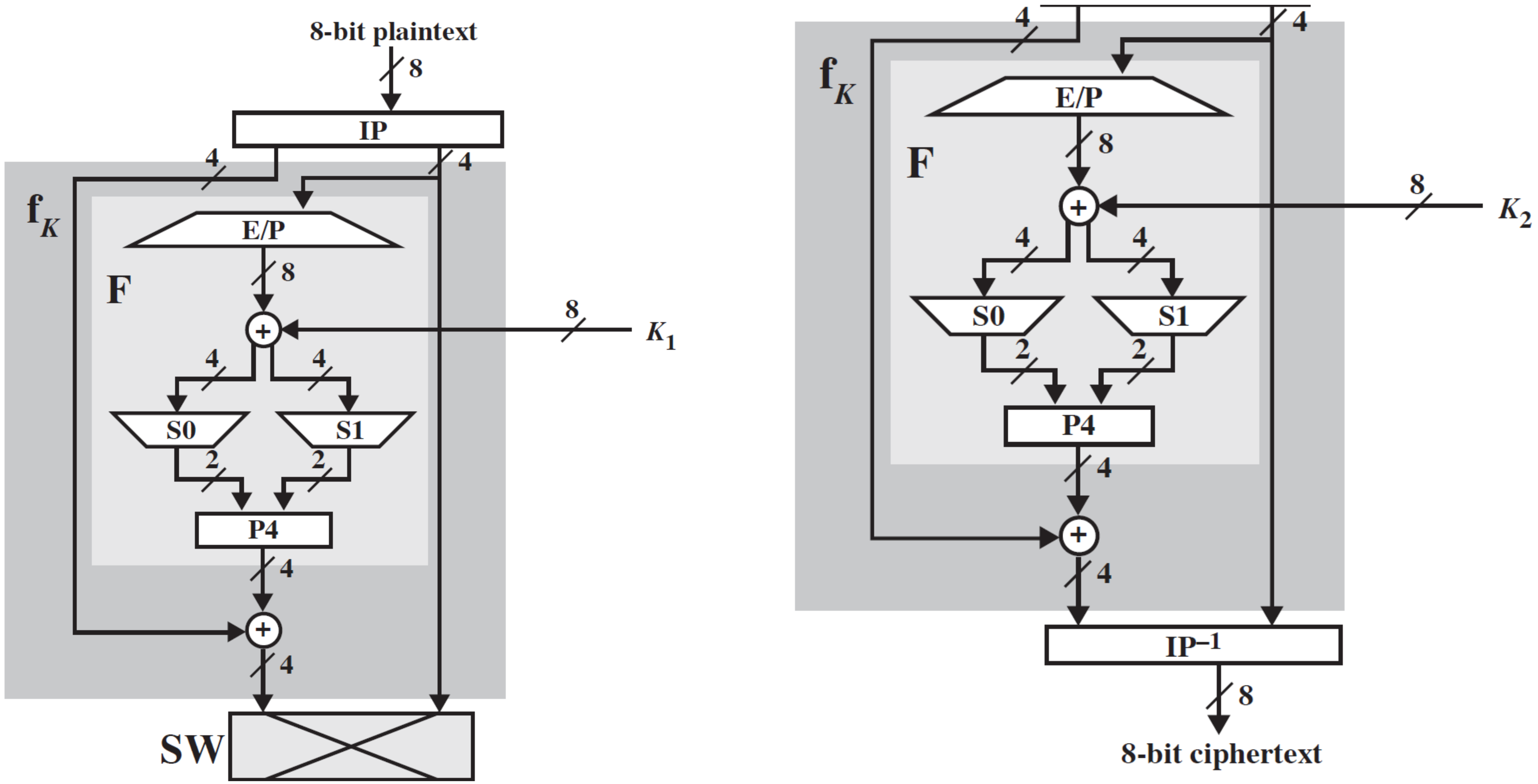
IP^{-1}							
4	1	3	5	7	2	8	6

S-DES Encryption Algorithm (Cont.)

The Function f_K

- The function f_K consists of a combination of **permutation** and **substitution** functions .
- Let L and R be the leftmost 4 bits and rightmost 4 bits of the 8-bit input to f_K.
- Let **F** be a mapping from 4-bit strings to a 4-bit strings.
- We say that:

$$f_K(L, R) = (L \oplus F(R, SK), R),$$ where SK is a subkey


8-bit plaintext
8
IP
4
4
f_K
E/P
8
F
8
K_1
4
4
S0
S1
2
2
P4
4
4
SW
4
4
f_K
E/P
8
F
8
K_2
4
4
S0
S1
2
2
P4
4
4
IP−1
8
8-bit ciphertext

S-DES Encryption Algorithm (Cont.)

- In the mapping **F**, the input is a 4-bit number **(n_1, n_2, n_3, n_4)**.
- First, we apply an expansion/permutation **(E/P)** operation.

E/P							
4	1	2	3	2	3	4	1

- We depict the result in this way:

$$\begin{array}{c|cc|c} n_4 & n_1 & n_2 & n_3 \\ n_2 & n_3 & n_4 & n_1 \end{array}$$

S-DES Encryption Algorithm (Cont.)

- The 8-bit subkey K_1 is added to the **E/P** value using **XOR**:

$$\left.\begin{array}{c} n_4 \oplus k_1 \\ n_2 \oplus k_5 \end{array}\right| \begin{array}{cc} n_1 \oplus k_2 & n_2 \oplus k_3 \\ n_3 \oplus k_6 & n_4 \oplus k_7 \end{array} \left| \begin{array}{c} n_3 \oplus k_4 \\ n_1 \oplus k_8 \end{array}\right.$$

- Let us rename these 8 bits:

$$\left.\begin{array}{c} p_{0,0} \\ p_{1,0} \end{array}\right| \begin{array}{cc} p_{0,1} & p_{0,2} \\ p_{1,1} & p_{1,2} \end{array} \left| \begin{array}{c} p_{0,3} \\ p_{1,3} \end{array}\right.$$

S-DES Encryption Algorithm (Cont.)

- Next we use **S-boxes** to convert a 4-bit input into a 2-bit output.
- First 4 bits ***(first row of preceding matrix)*** are fed into **S-box S0** to produce a 2-bit output.
- Last 4 bits ***(second row of preceding matrix)*** are fed into **S-box S1** to produce the other 2-bit output.
- The **1st** and **4th** input bits are treated as a 2-bit number that specify a **row** of the **S-box**.
- The **2nd** and **3rd** input bits are treated as a 2-bit number that specify a **column** of the **S-box**.

S-DES Encryption Algorithm (Cont.)

- The two **S-boxes** are defined as follows:

$$S0 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{bmatrix} \end{array} \qquad S1 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 1 & 2 & 3 \\ 2 & 0 & 1 & 3 \\ 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 3 \end{bmatrix} \end{array}$$

- The entry in that row and column is the 2-bit output.
- **E.g.**, if $(P_{0,0}\ P_{0,3})$ = **(00)** and $(P_{0,1}\ P_{0,2})$ = **(10)**, then the output is from row **0**, column **2** of **S0**, which is **3**, or **(11)** in binary.
- $(P_{1,0}\ P_{1,3})$ and $(P_{1,1}\ P_{1,2})$ are used to index into a row and column of **S1** to produce an additional 2 bits.

S-DES Encryption Algorithm (Cont.)

- Next, the 4 bits produced by **S0** and **S1** undergo a further permutation as follows:

P4			
2	4	3	1

- By that, we achieve the output of **P4** which is the output of the function **F**.
- Finally, output of **F** is **XOR** with ***L*** (i.e. leftmost 4 bits), to produce the left output of $\mathbf{f}_K$.

S-DES Encryption Algorithm (Cont.)

The Switch Function

- The function **f_K** only alters the leftmost 4 bits of the input.
- The switch function **(SW)** interchanges the left and right 4 bits so that the second instance of **f_K** operates on a different 4 bits.
- In this second instance, **E/P**, **S0**, **S1**, and **P4** functions would remain the same.
- However, the key input would be K_2.

S-DES Encryption Algorithm (Cont.)

- **Example 01:** generate the ciphertext while using the **S-DES** technique provided that an 8-bit plaintext input **01110010** is provided and a 10-bit key **1010000010** is used.
- **Solution:** the resulting 8-bit ciphertext is **01110111**.

Analysis of S-DES

- A brute-force attack on simplified DES is certainly feasible.
- With a 10-bit key, there are only **$2^{10} = 1024$** possibilities.
- Given a ciphertext, an attacker can try each possibility and analyze the result to determine if it is reasonable plaintext.
- What about cryptanalysis? ***Left for the reader to explore.***

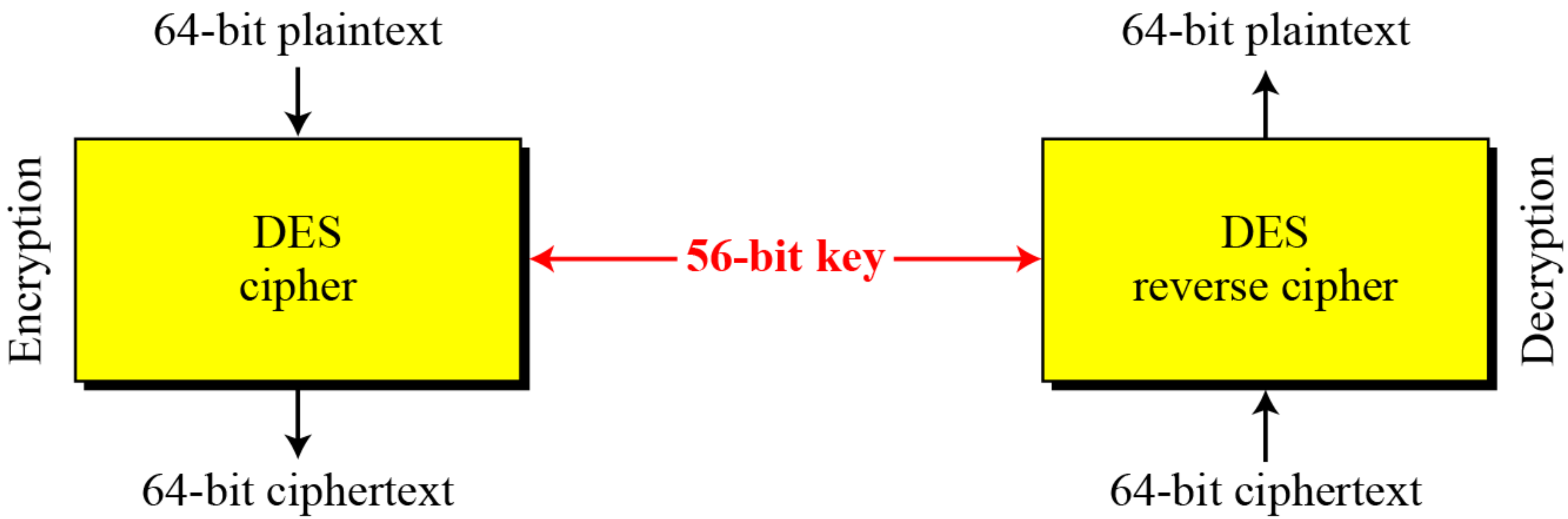
Data Encryption Standard (DES)

DES Encryption

- There are two inputs to encryption function: **plaintext** and **key**.
- Plaintext is a **64 bits** block and the key is **56 bits** in length.

- Processing of plaintext in DES can be divided into four phases:
  1. The **64-bit** plaintext passes through an **IP**.
  2. **Sixteen rounds** of the same function are executed, which involves both **permutation** and **substitution** functions.
  3. Left and right halves of the output are swapped to produce a pre-output.
  4. The pre-output is passed through $\mathbf{IP^{-1}}$.

DES Encryption (Cont.)



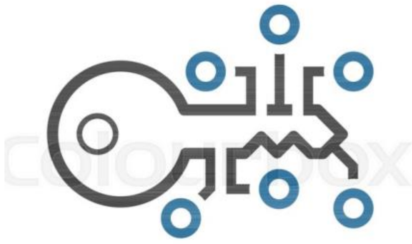
DES Encryption (Cont.)

Q) Why 16 rounds in DES?

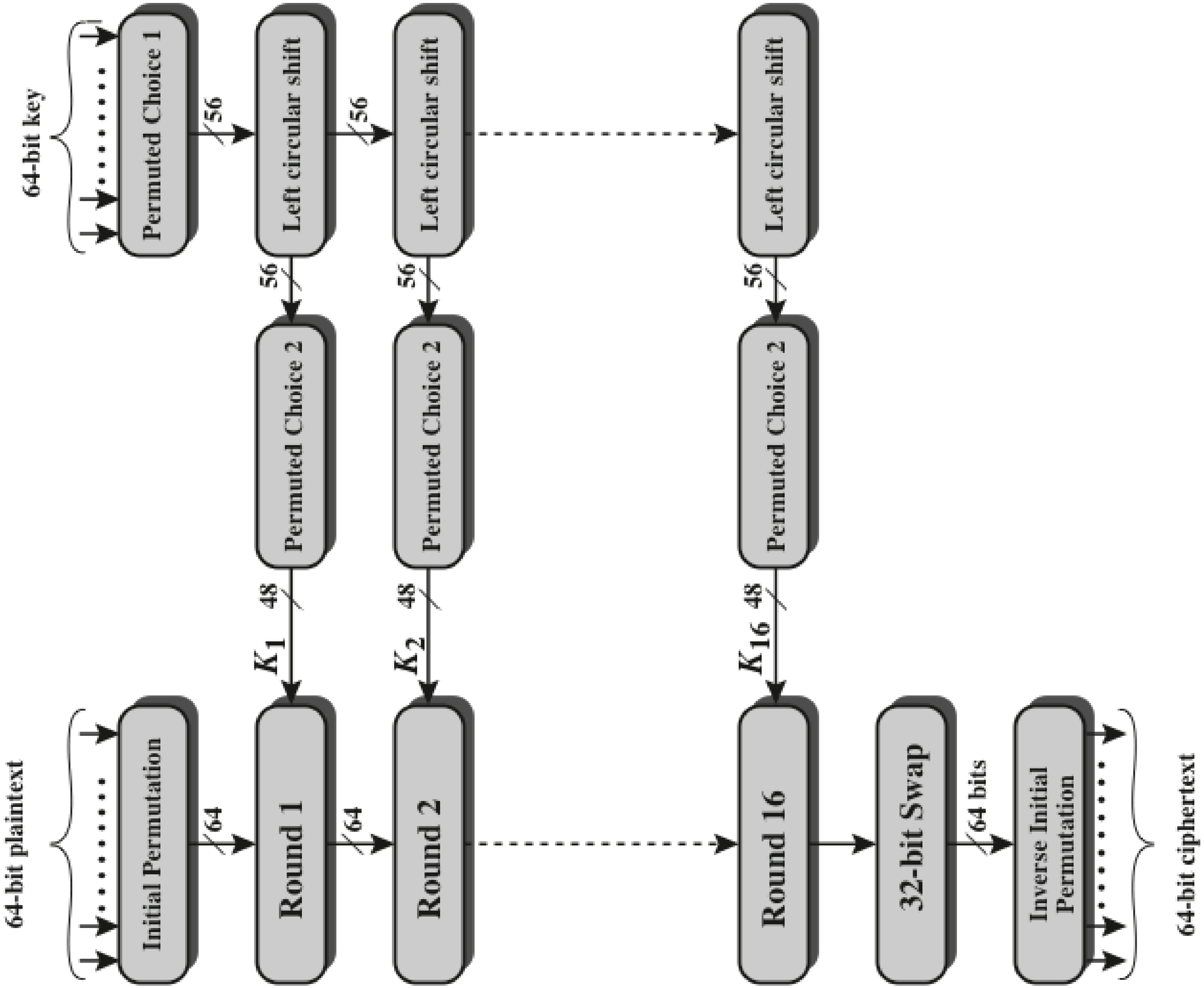
- The goal is to completely scramble the data and key so that every bit of the ciphertext depends on every bit of data and every bit of key.
- After sufficient rounds along with a good algorithm, there should be no correlation between ciphertext and either the original data or key.
- In DES, a minimum of **12 rounds** were needed to sufficiently scramble the key and data together, while the others provided a margin of safety.

DES Encryption (Cont.)

- Initially, the key is passed through a **permutation** function.
- For each of the 16 rounds, a **subkey (*Ki*)** is produced by **left circular shift** and a **permutation**.
- The **permutation** function is same for each round, but a different **subkey** is produced because of the repeated **circular left shift**.



Encryption Key


64-bit key
Permuted Choice 1
56
Left circular shift
56
Left circular shift
Left circular shift
56
56
56
Permuted Choice 2
Permuted Choice 2
Permuted Choice 2
48
48
48
K_1
K_2
K_{16}
64-bit plaintext
Initial Permutation
64
Round 1
64
Round 2
Round 16
32-bit Swap
64 bits
Inverse Initial Permutation
64-bit ciphertext

DES Decryption

- There are two points to remember on DES decryption:
  a. Decryption uses the same algorithm as encryption, except that the application of **subkeys** is reversed.
  b. Additionally, the **initial** and **final** **permutations** are reversed.

DES in Hexadecimal

- We work through an example of DES, with the objective of studying the **hex patterns** that occurs from one step to the next.
- For this example, the plaintext, key and the ciphertext are provided in **hexadecimal**.

Plaintext:	02468aceeca86420
Key:	0f1571c947d9e859
Ciphertext:	da02ce3a89ecac3b

The Avalanche Effect

- **Avalanche effect:** a small change in either **plaintext** or **key** should produce a "significant" change in the **ciphertext**.
- The avalanche effect is a desirable property of any encryption algorithm. In particular, a change in **one bit** of **plaintext** or **key** should produce a change in **many bits** of the **ciphertext**.
- **Q) Why avalanche effect is desirable?**
- **A)** If the change were small, this might provide a way to reduce the size of ***plaintext or key space to be searched***.

- Avalanche Effect in DES: **Change in Plaintext**.
- Table shows the result when **4th** bit of plaintext is changed.
- After just three rounds, 18 bits differ between the two blocks.
- On completion, the two ciphertexts differ in 32 bit positions.

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 Key**.
- Using two keys that differ in only **4th** bit position.
- Original key and altered keys are **0f1571c947d9e859**, **1f1571c947d9e859**.
- Results show that about half of the bits in **ciphertext** differ and the avalanche effect is shown after just a few rounds.

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

Round		δ
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

DES Strength

- There have been concerns about the level of security provided by DES in-terms of **key size** and **nature of algorithm**.

The 56-bit Key Size:

- With a key length of **56 bits**, there are $2^{56} \approx 7.2 \times 10^{16}$ keys.
- With current technologies, it is not even necessary to use special purpose-built hardware to break the code.
- The speed of commercial, off-the-shelf processors threaten the security of DES.

DES Strength (Cont.)

- 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 $= 5.3 \times 10^{21}$ years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	2^{167} ns $= 5.8 \times 10^{33}$ years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	2^{191} ns $= 9.8 \times 10^{40}$ years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	2^{255} ns $= 1.8 \times 10^{60}$ years	1.8×10^{56} years
26 characters (permutation)	Monoalphabetic	$2! = 4 \times 10^{26}$	2×10^{26} ns $= 6.3 \times 10^{9}$ years	6.3×10^{6} years

DES Strength (Cont.)

Nature of DES Algorithm:

- Another concern is that *(Differential, Linear)* cryptanalysis is possible by exploiting characteristics of DES algorithm, where focus of concern is on the **S-boxes**.
- Since the design criteria for **S-boxes** were not made public, there is a suspicion that **S-boxes** were constructed in a way that cryptanalysis is possible by those who know the weaknesses in the **S-boxes**.
- Over the years, a number of **regularities** and **unexpected** behaviors of the **S-boxes** have been discovered.

S-Boxes in DES

S-Boxes

S-box 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-box 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-box 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-box 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-box 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-box 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-box 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-box 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

Initial Permutation

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

Multiple DES

Need for Multiple DES

- Due to the inherent weakness of DES, w.r.t. today's technologies, some organizations use **triple DES (3DES)**.
- In **3DES**, the process of DES is repeated three times for added strength.
- This is performed until organizations can afford to update their equipment to AES capabilities.

Double-DES

- We use 2-DES that encrypts each block with a different key.

$$c = E_{K2}\,(E_{K1}\,(m)\,)$$

- To decrypt

$$m = D_{K1}\,(D_{K2}\,(c)\,)$$

- The 2-DES is expected to provide security equivalent to **56 × 2 = 112** bits.
- However, such a cipher can be attacked by a method called **Meet-in-the-Middle (MIM)** attack.

Triple-DES

- Use three keys, namely K1, K2 and K3. Hence,

$$c = E_{K3}(E_{K2}(E_{K1}(m)))$$

- To decrypt

$$m = D_{K1}(D_{K2}(D_{K3}(c)))$$

- The 3-DES is expected to provide security equivalent to **56 × 3 = 168** bits.
- Triple DES with three keys is used by many applications such as **Pretty Good Privacy (PGP)**.

Thank You!