Data Encryption Algorithm

Course Teacher: Dr. Muhammad Usama

Outline

- Data Encryption Algorithm Design Criteria
- Data Encryption Standard
- Multiple DES

Things to know

- Any *message written over a fixed set of symbols* can be represented as a *binary string* (a sequence of 0's and 1's)
- Binary digits 0 and 1 are called *bits*
- To reduce computation overhead, encryption algorithms should only use operations that are easy to implement
- For a binary string X:
  - The length of X, denoted by $|X|$, is the number of bits in X
  - If $|X| = l$, X is an l-bit binary string
  - Denote the concatenation of *X* and *Y* can by XY or $X \,||\, Y$

What is Encryption?

- Encryption
  - Make plain text messages ***unreadable***
  - The ***unreadable*** text can be converted back to its original form
- Common encryption method: use secret keys and algorithms
  - Conventional encryption algorithms (***symmetric-key*** encryption algorithms): ***Same key*** for encryption and decryption
  - Public-key encryption algorithms (***asymmetric-key*** encryption algorithms): ***Different keys*** for encryption and decryption

Example: Substitution

- A one-to-one mapping of characters; e.g.

substitute a with d, b with z, c with t, etc

- Unreadable to untrained eyes, this method maintains the statistical structure of the underlying language (e.g., character frequency)
- In English, the letter “e” appears most frequently of all single letters
- The letter with the highest frequency in the unreadable text is likely the letter “e”
- The method can be applied to other letters and letter sequences to find the original message

XOR Encryption

- The exclusive-OR operation, denoted by ⊕ ***or XOR***, is a simple binary operation used in encryption
- **XOR encryption:** Divide a string into blocks of equal length and encrypt each block with a secrete key of the same size of the block
- **For example**, if we use a block size of 8 (1 byte), on a two-character (2 byte) string M, we use an 8-bit Encryption key (such as: 1100 1010) on M twice:

M:		1111 1111 0000 0000
K:	⊕	1100 1010 1100 1010
C:		0011 0101 1100 1010

We can decrypt C using the same key; i.e., we simply XOR C with K to get M:

C:		0011 0101 1100 1010
K:	⊕	1100 1010 1100 1010
M:		1111 1111 0000 0000

- This is simple and easy to implement
- But it is not secure, for knowing any one pair (M_i,C_i) will reveal K:

$$M_i \oplus C_i = M_i \oplus (M_i \oplus K) = K!$$

Criteria of Data Encryption

- XOR encryption is secure if a ***key is only used once***, but it's unpractical
- How about ***keeping encryption algorithms private?***
- To study the security of encryption algorithms, we assume that ***everything except the encryption keys are publicly disclosed***, and the ***keys are reusable***
- Good encryption algorithms must satisfy the following criteria:

  -***Efficiency***

  -Resistance to ***Statistical Analysis***

  -Resistance to ***Brute-Force Attacks***

  -Resistance to ***Mathematical Analysis Attacks***

Efficiency

- Operations used in the algorithms must be ***easy to implement on hardware and software***
- Execution of the algorithms should consume only ***moderate resources***
- ***Time complexity and space complexity*** must be kept within a ***small*** constant factor of the input size
- Common operations:
  - ***XOR***
  - ***Permutations***: one-to-one mapping
  - ***Substitution***: many-to-one mapping
  - ***Circular shift***: a special form of permutation
  - ***Operations on finite fields***

Resistance to Statistical Analysis

- Analysing the frequencies of characters in C, one can find out the original characters in M they correspond to
- ***Diffusion*** and ***confusion*** are standard methods to flatten statistical structure
  - ***Diffusion***: Each bit in ***C*** should depend on multiple bits (as evenly as possible) in ***M***
    - Diffusion can be obtained by executing a ***fixed sequence of operations*** for a fixed number of rounds on strings generated from the previous round
  - ***Confusion***: Each bit in ***C*** should depend on multiple bits (as evenly as possible) in the secrete key ***K***
    - Confusion can be obtained by ***generating sub-keys from K*** and using ***different sub-keys in different rounds***

Resistance to Brute-Force Attacks

- The strength of an encryption algorithm depends on ***its operations and the key length***
- Suppose the encryption key is ***l-bit long***, with ***2^l possible keys***
- If Eve the ***eavesdropper*** attains a ciphertext message C and knows the algorithm used to encrypt it, she can ***try all keys one at a time*** until she decrypts the message into something makes sense
- Thus, the time complexity of a brute-force attack is in the order of ***2^l***
- Under current technologies, it is believed that ***$l = 128$ would be sufficient***
- The time complexity of a brute-force attack is often used as the benchmark for other cryptanalysis attacks: If an attack with a time complexity substantially less than 2^l is found, the attack is considered useful

Resistance to Other Attacks

- Other common attacks e.g., ***chosen-plaintext attacks*** and ***mathematical attacks***
- **Chosen-plaintext Attacks:**
  - Obtain a specific M encrypted to C
  - Use this pair (M, C) to find out the key used
  - Example: XOR encryption

    If Eve knows (M, C) she can find K easily:

    $C = (M \oplus K)$

    $M \oplus C = M \oplus (M \oplus K)$

    $M \oplus C = K$
- **Mathematical Attacks:**
  - Use mathematical methods to decipher encrypted messages
    - Differential Cryptanalysis, Linear Cryptanalysis, Algebraic Cryptanalysis.
    - Require sophisticated mathematics

Outline

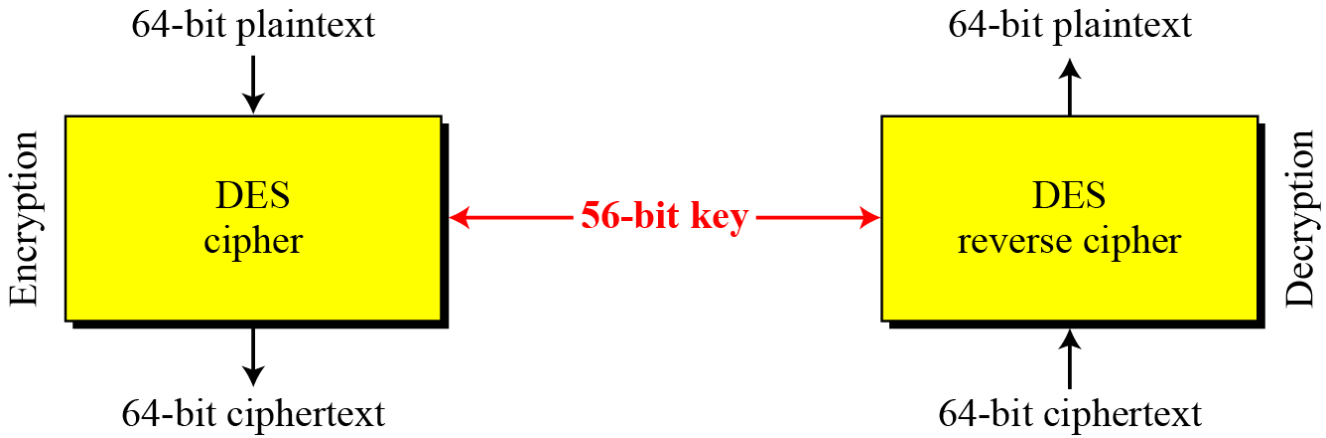
- Data Encryption Algorithm Design Criteria
- Data Encryption Standard
- Multiple DES

Data Encryption Standard (DES)

- Published by the US National Bureau of Standards (NBS) in 1977
- A concrete implementation of the Feistel Cipher Scheme (FCS), invented by Horst Feistel
- Symmetrical encryption and decryption structures
- Use four basic operations: XOR, permutations, substitution, and circular shift
- Widely used from mid-70’s to early-2000’s.
- Phased out by AES and other better encryption algorithms

DES: Overview

- DES is a block cipher.
- Encryption and decryption with DES

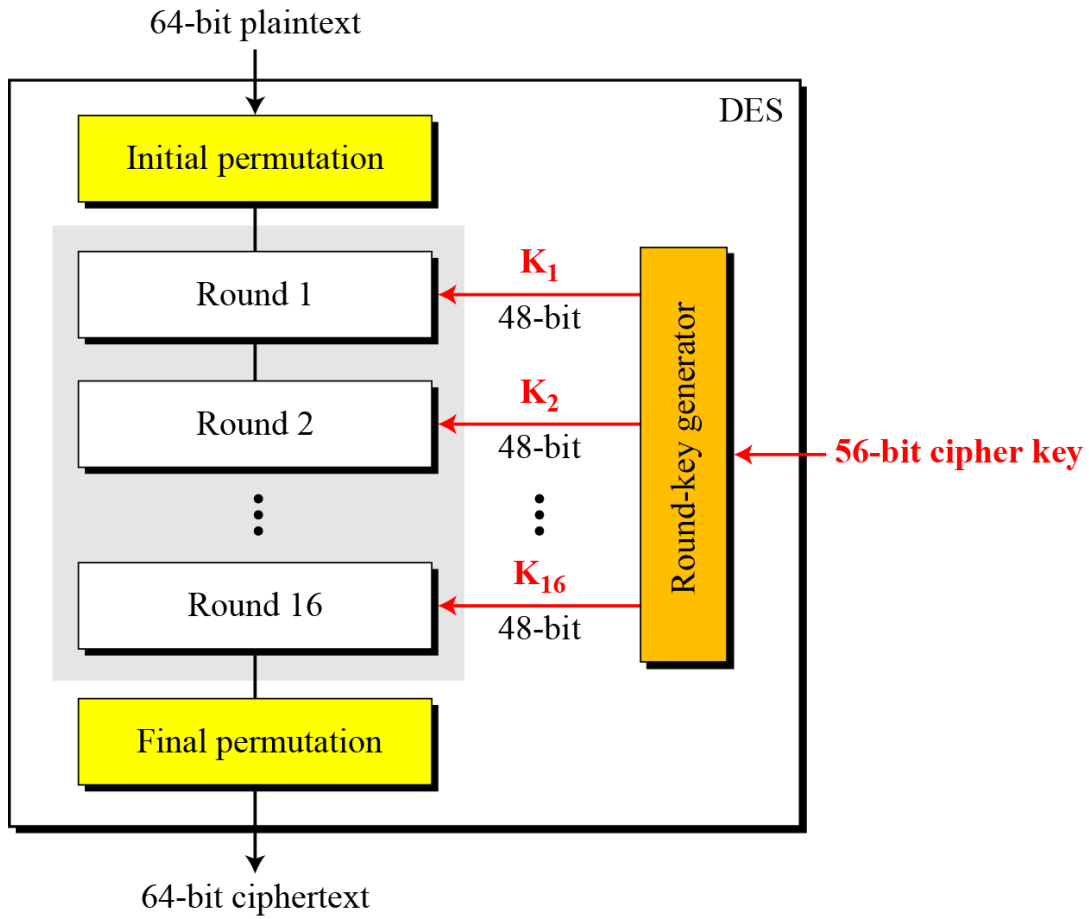


DES Encryption

- 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 IP^{-1}.

DES Encryption - General structure of DES



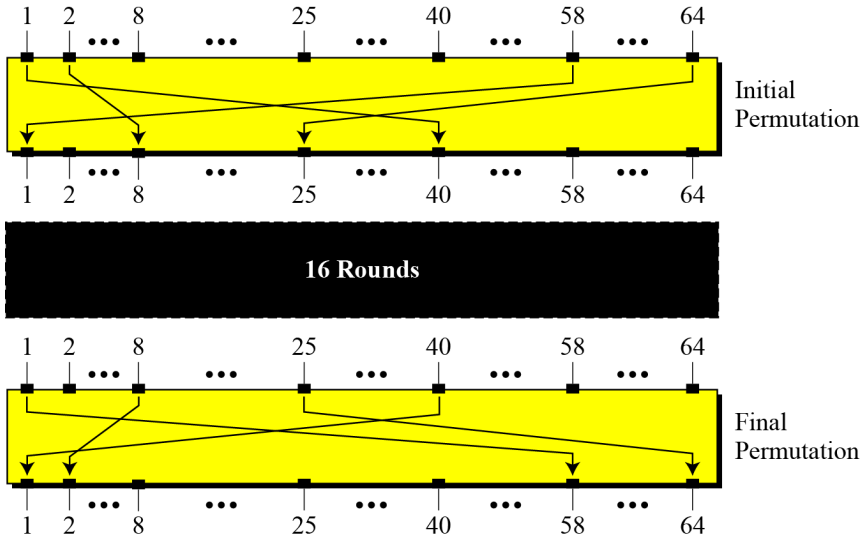
DES Encryption (Cont.)

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 the data and every bit of the 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: Initial and Final Permutations

- The initial and final permutations are straight P-boxes that are inverses of each other.
- They have **no cryptography significance in DES**.



Initial and final permutation steps in DES

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 permutation tables

DES Encryption: Initial and Final Permutations (Cont.)

Example 01: Find the output of the final permutation box when the input is given in hexadecimal as:

0x0000 0080 0000 0002

In binary -> 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 2

0000 0000 0000 0000 0000 0000 1000 0000 0000 0000 0000 0000 0000 0000 0000 0010

Solution: Only bit 25 and bit 63 are 1s; the other bits are 0s. In the final permutation, bit 25 becomes bit 64 and bit 63 becomes bit 15. The result is

0x0002 0000 0000 0001

In binary -> 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 1

0000 0000 0000 0010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0001

DES Encryption: Initial and Final Permutations (Cont.)

Example 02: Prove that the initial and final permutations are the inverse of each other by finding the output of the initial permutation if the input is

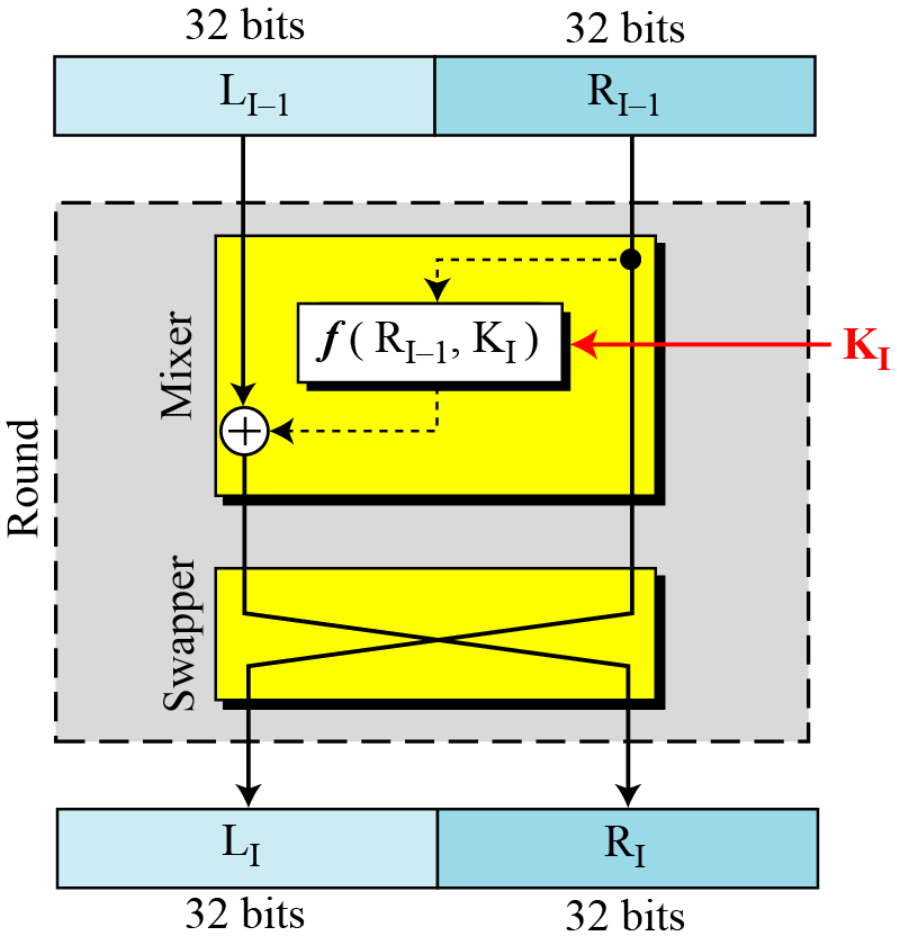
0x0002 0000 0000 0001

Solution: The input has only two 1s; the output must also have only two 1s. Using Table, we can find the output related to these two bits. Bit 15 in the input becomes bit 63 in the output. Bit 64 in the input becomes bit 25 in the output. So the output has only two 1s, bit 25 and bit 63. The result in hexadecimal is

0x0000 0080 0000 0002

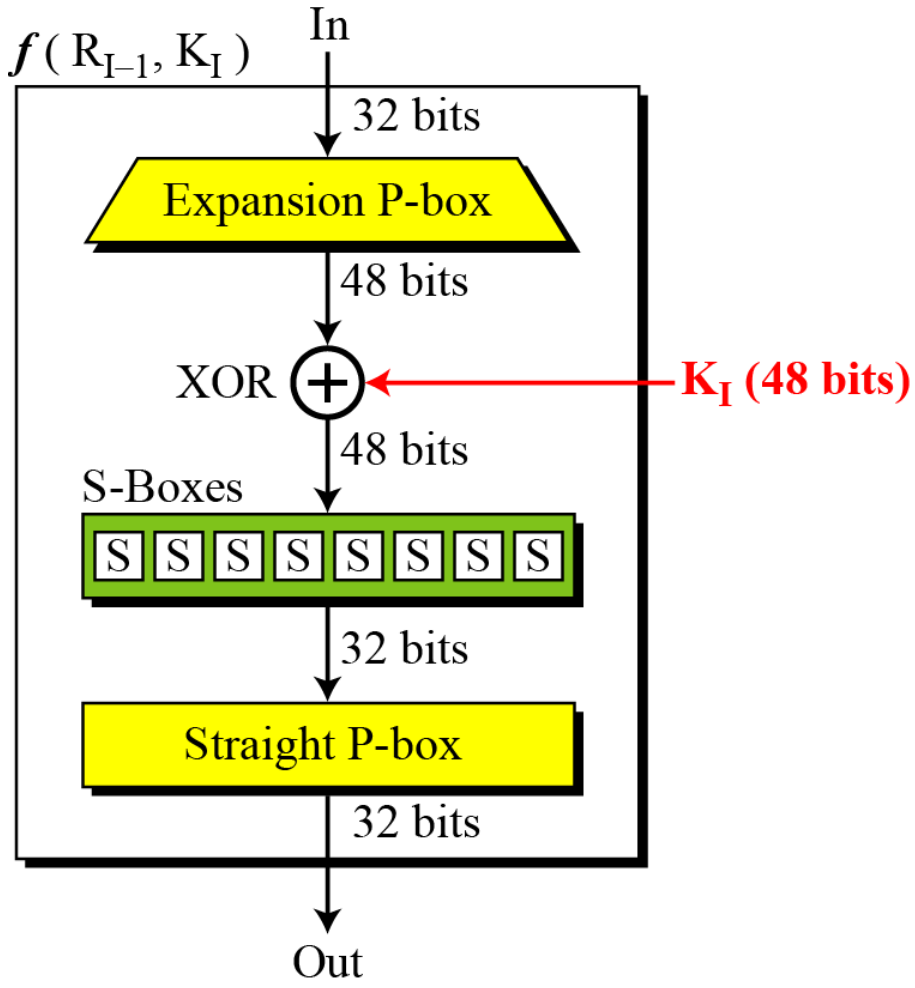
DES Encryption: Rounds

- DES uses 16 rounds.
- Each round of DES is a Feistel cipher.



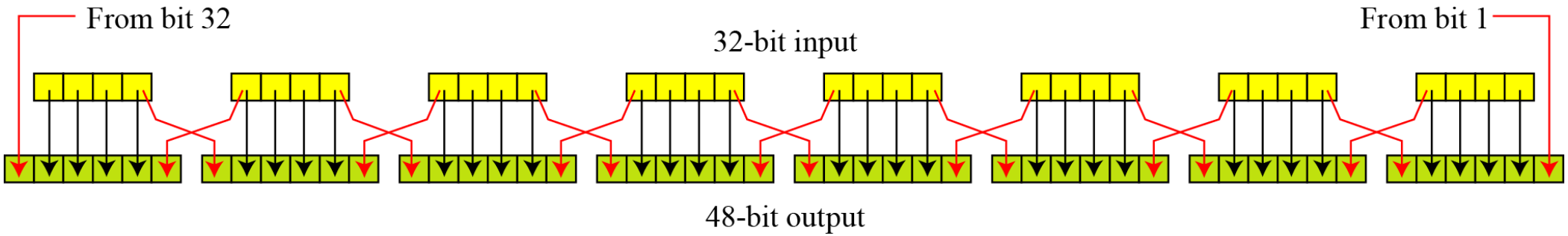
DES function

- The heart of DES is the DES function.
- The DES function applies a **48-bit key** to the rightmost **32 bits** to produce a **32-bit** output.



DES function: Expansion P-box

- Since R_{l-1} is a 32-bit input and K_l is a 48-bit key, we first need to expand R_{l-1} to 48 bits.



DES function: Expansion P-box (Cont.)

- Although the relationship between the input and output can be defined mathematically, DES uses Table to define this P-box.

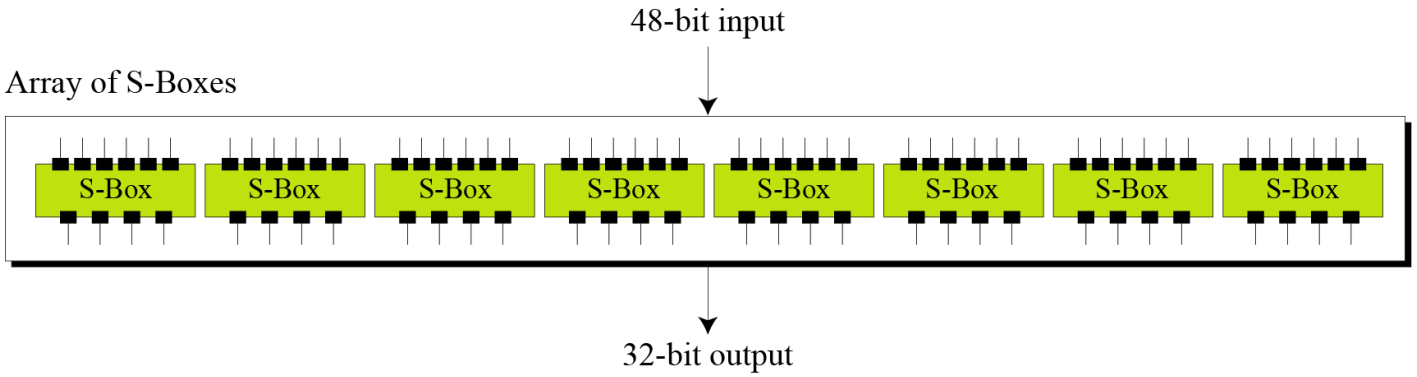
32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

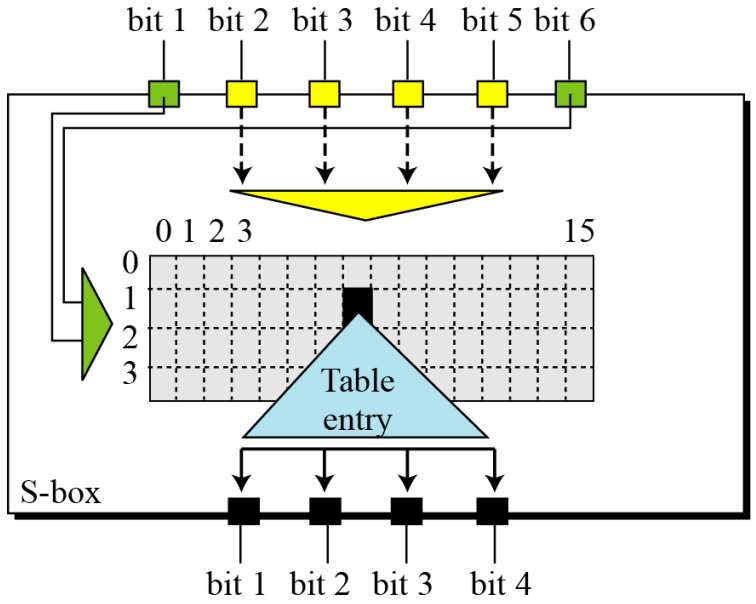
DES function: Whitener (XOR)

- After the expansion permutation, DES uses the XOR operation on the expanded right section and the round key.
- Note that both the right section and the key are 48-bits in length.
- Also note that the round key is used only in this operation.

DES function: S-Boxes

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





DES function: S-Boxes (Cont.)

- Table shows the permutation for **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

DES function: S-Boxes (Cont.)

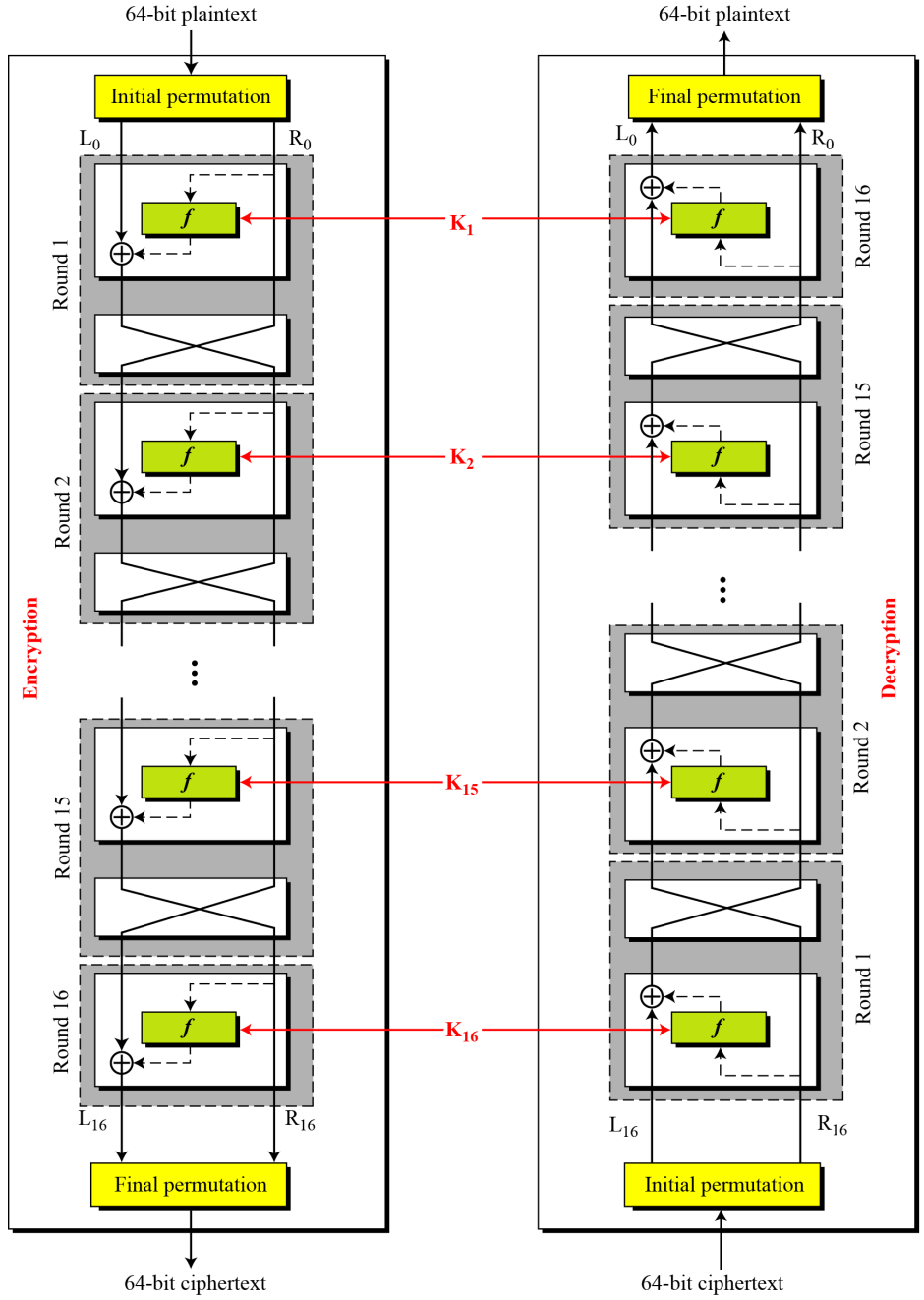
- **Example 01:** The input to S-box 1 is 100011. What is the output?
- **Solution:**
- If we write the first and the sixth bits together, we get **11 in binary**, which is **3 in decimal**.
- The remaining bits are **0001 in binary**, which is **1 in decimal**. We look for the value in **row 3, column 1, in Table (S-box 1)**.
- The result is **12 in decimal**, which in **binary is 1100**.
- So the input 100011 yields the output 1100.

DES function: Straight Permutation

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 Encryption

- DES cipher and reverse cipher for the first approach
- **Note:** there is ***no swapper in the last round.***



DES Decryption

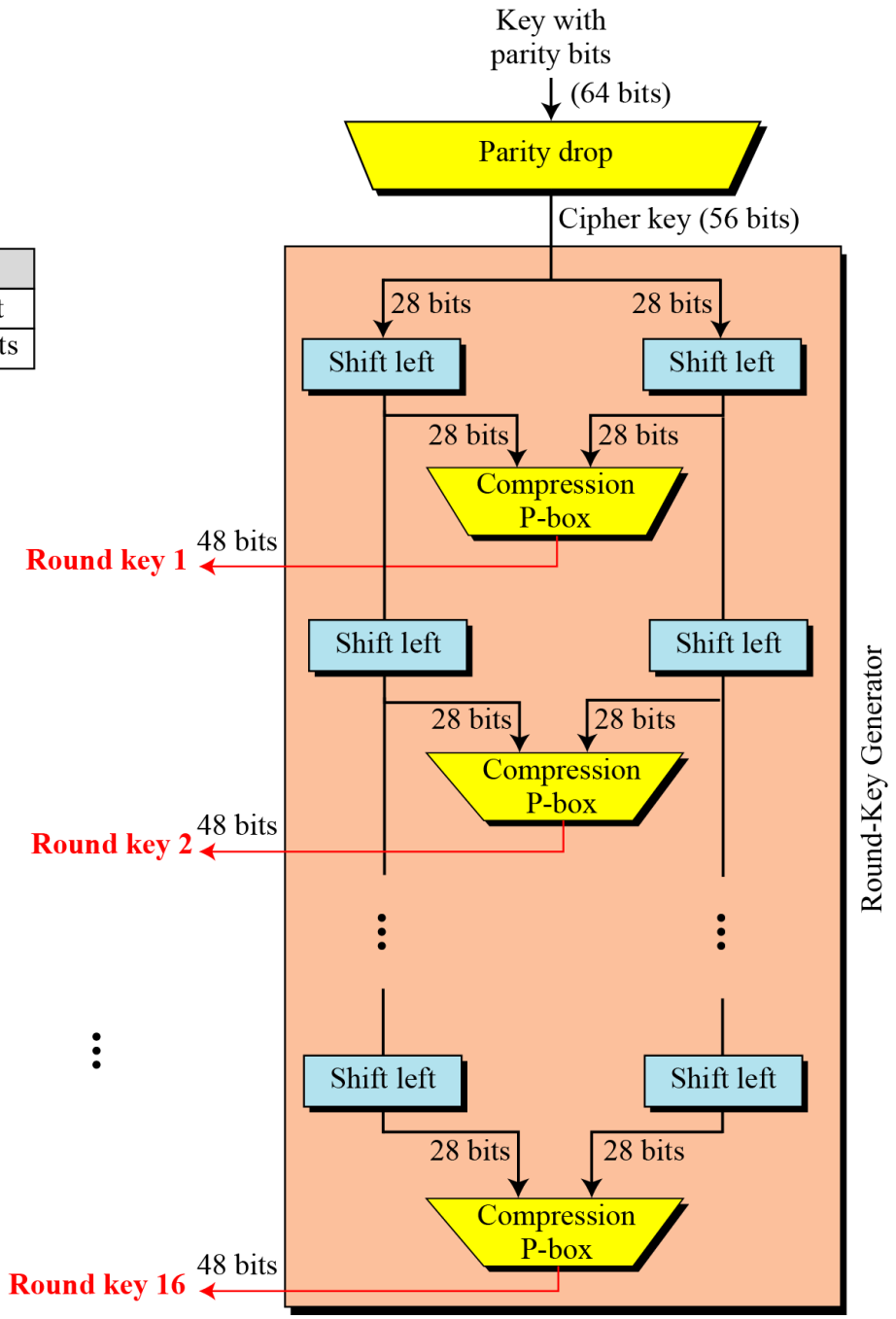
- There are two points to remember on DES decryption:
  - Decryption uses the same algorithm as encryption, except that the application of **subkeys** is reversed.
  - Additionally, the **initial** and **final** **permutations** are reversed.

DES Encryption

- **Key Generation:** The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.

Shifting

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



DES Encryption: Key Generation (Cont.)

- **It drops the parity bits (bits 8, 16, 24, 32, ..., 64)**

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

Parity-bit drop table

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bit shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Number of bits shifts

DES Encryption: Key Generation (Cont.)

14	17	11	24	01	05	03	28
15	06	21	10	23	19	12	04
26	08	16	07	27	20	13	02
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

Key-compression table

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 technology, it is not even necessary to use special purpose-built hardware.
  - 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 = 5.3×10^{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

DES Strength (Cont.)

Nature of DES Algorithm:

- Another concern is that 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 few **regularities** and **unexpected** behaviors of the **S-boxes** have been discovered.

Outline

- Data Encryption Algorithm Design Criteria
- Data Encryption Standard
- Multiple DES

Double-DES

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

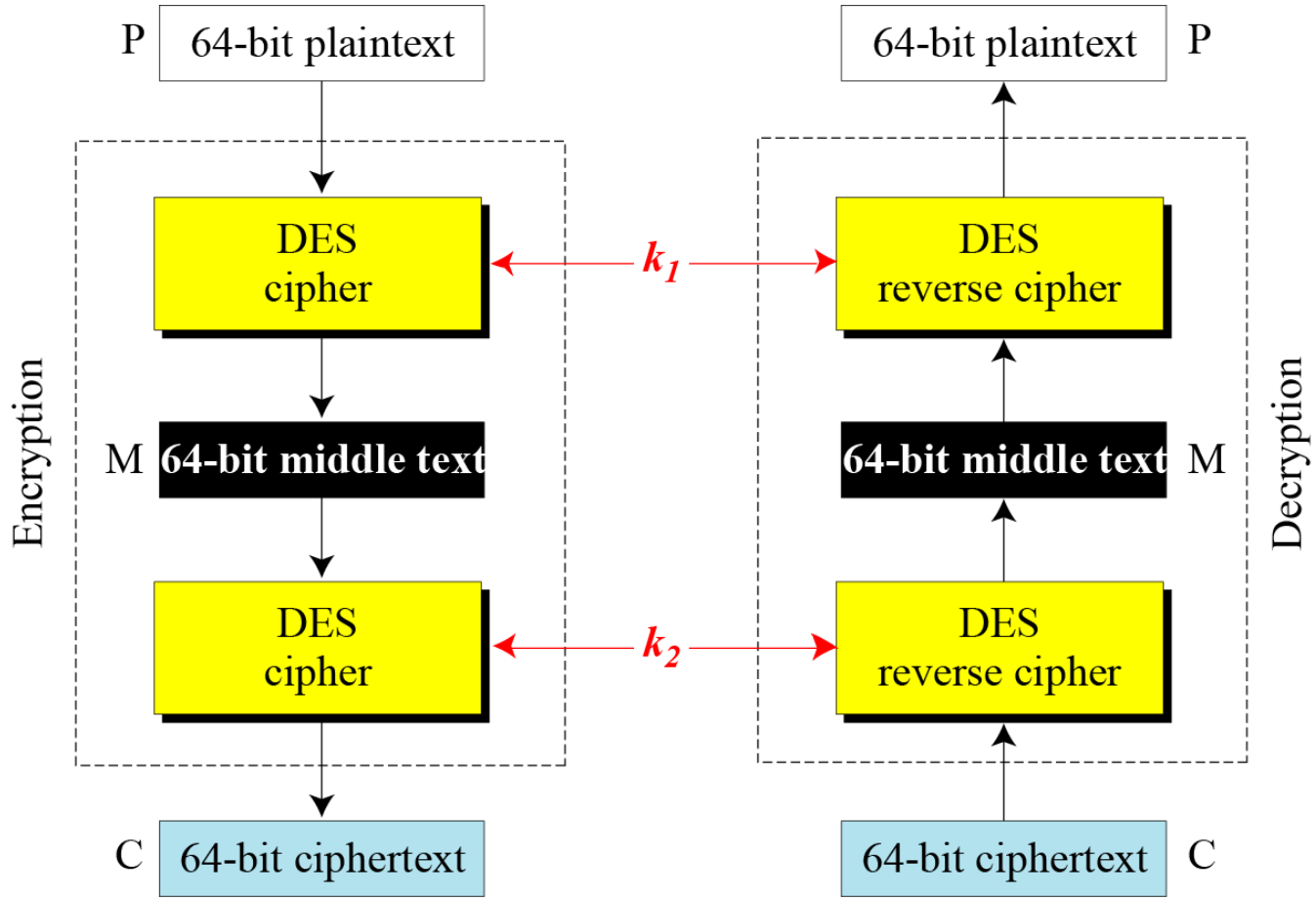
$$c = E_{K2}(E_{K1}(m))$$

- 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** attack.

Double-DES (Cont.)



Triple-DES

- In general, there are two flavors of 3-DES:

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

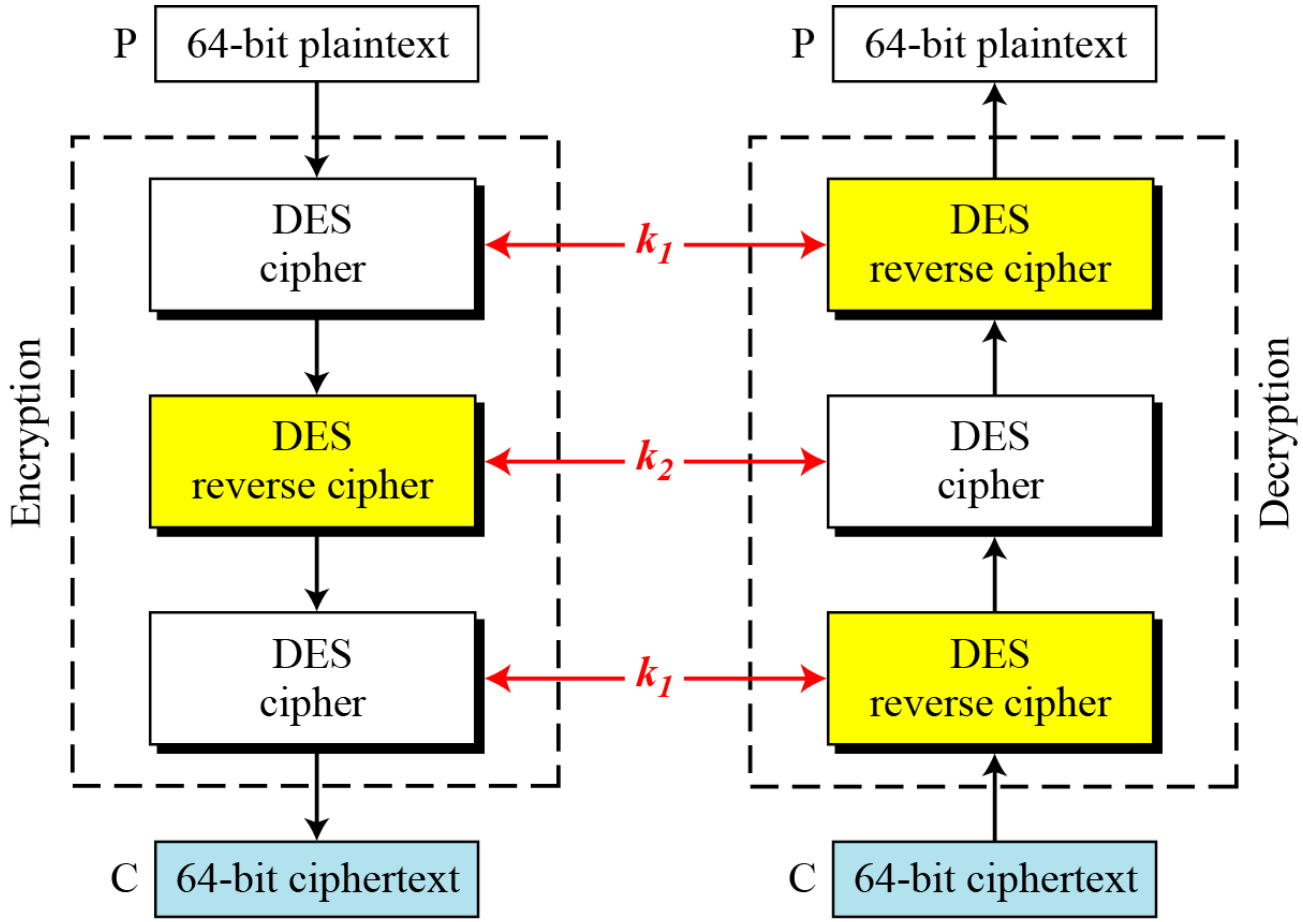
$$c = DES_{K3}(DES_{K2}(DES_{K1}(m)))$$

  b. Use two keys. Hence,

$$c = DES_{K1}(DES_{K2}^{-1}(DES_{K1}(m)))$$

- Triple DES with three keys is used by many applications such as **Pretty Good Privacy (PGP)**.

Triple-DES (Cont.)



DES SBoxes

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

Thank you