澳門理工大學
Universidade Politécnica de Macau
Macao Polytechnic University

COMP412: Computer Security

Modern Block Ciphers

Dr. Kim, Song-Kyoo (Amang)
Associate Professor,

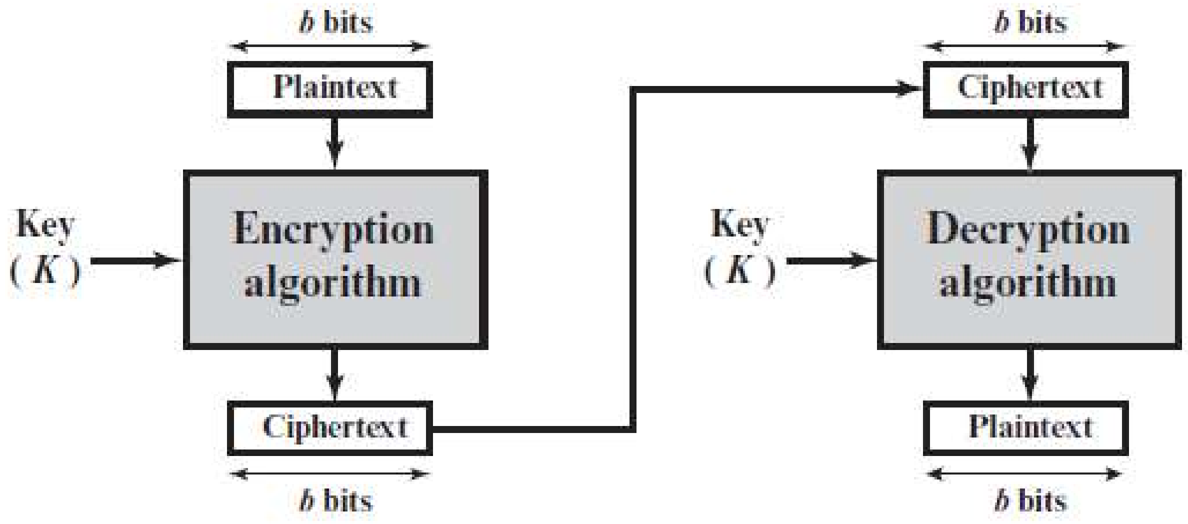
Faculty of Applied Sciences
MACAO POLYTECHNIC UNIVERSITY
Macau, SAR

Agenda

- Block ciphers
  - DES and its security
  - 2DES and 3DES
- Mode of use of block ciphers
  - ECB & CBC
  - CFB & OFB
- Stream ciphers
  - Random number generator
  - Stream cipher structure
  - RC4 algorithm

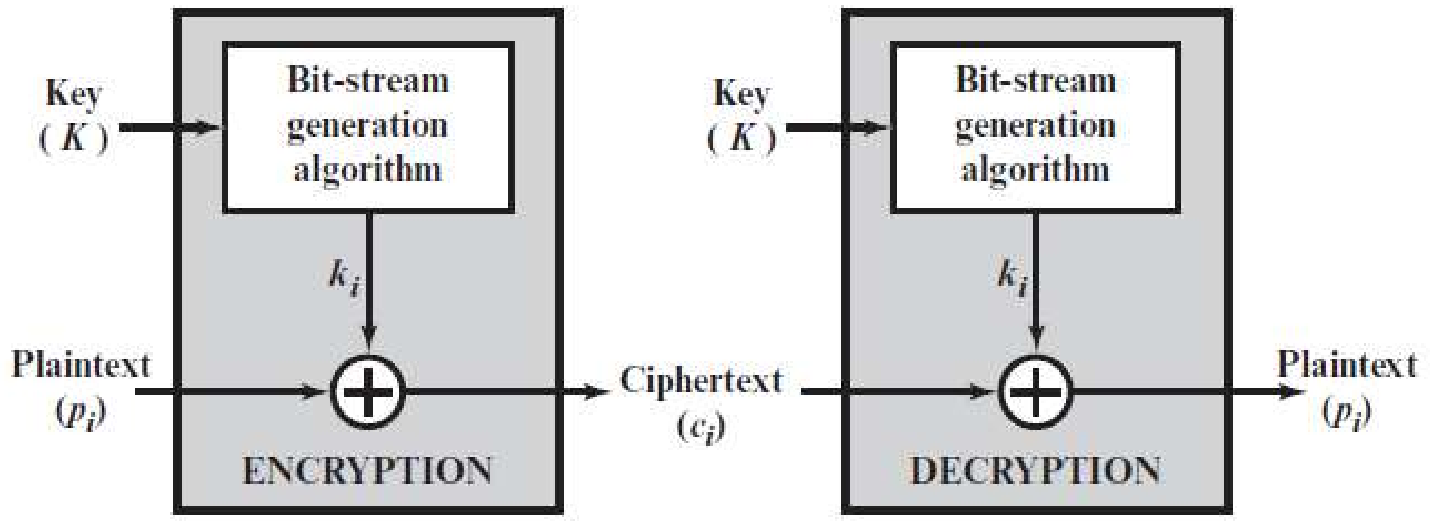
Stream Ciphers & Block Ciphers (1/2)

- Block cipher
  - Block of plaintext is treated as a whole and used to produce a ciphertext block of equal length (64 / 128 bits).



Stream Ciphers & Block Ciphers (2/2)

- Stream cipher
  - Encrypts a digital data stream one bit or one byte at a time.
  - classical stream ciphers are the autokeyed Vigenère cipher and the Vernam cipher.



Block Cipher Characteristics

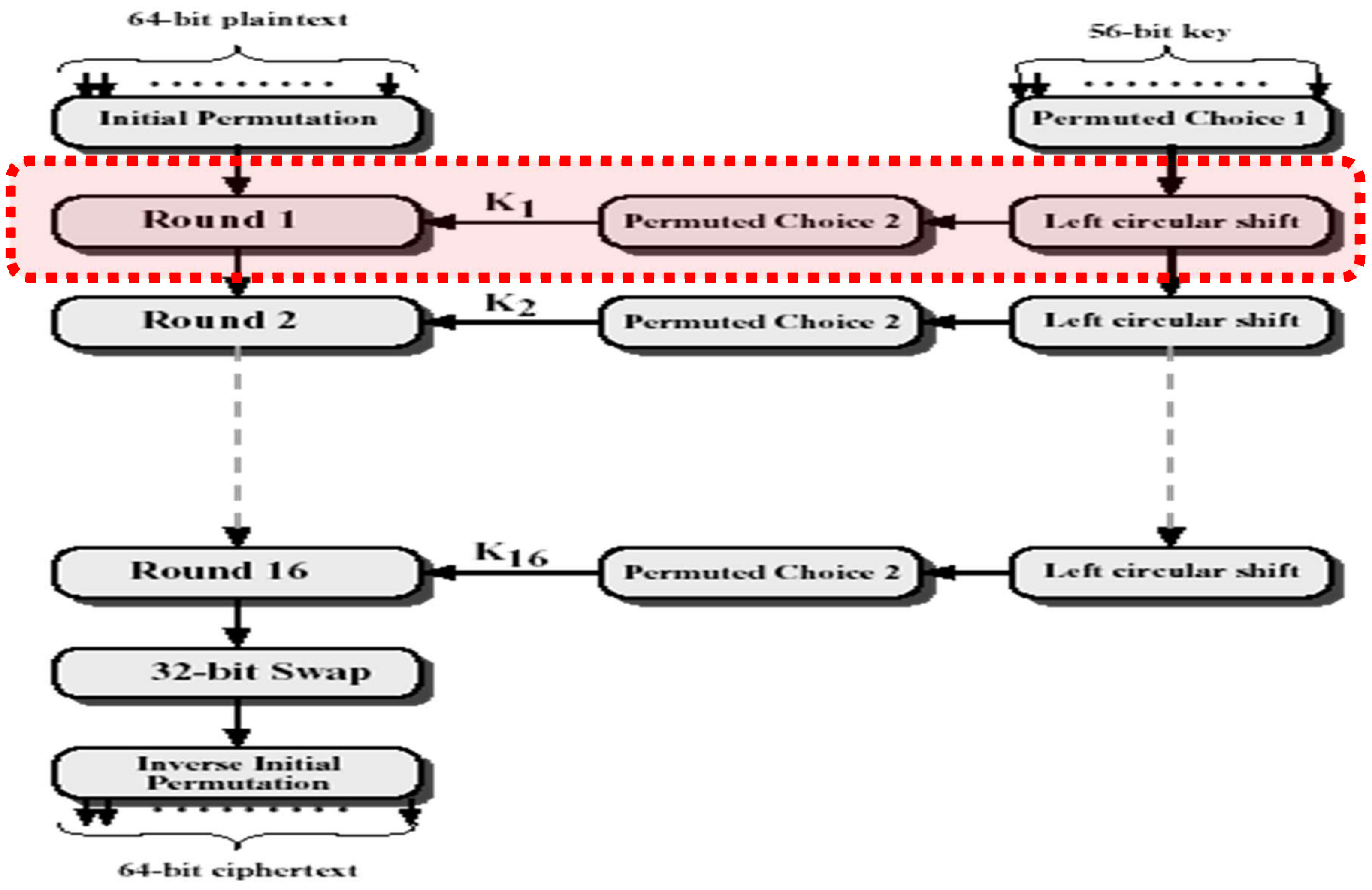
- Variable (or flexible) of
  - key length / block size / no. of rounds
- Mixed operators, data
- Key dependent rotation
- Key dependent S-boxes
- More complex key scheduling (Variable F)
- Operation of full data in each round
- Varying non-linear functions

Data Encryption Standard (1/3)

- **Data Encryption Standard (DES):**
  - A symmetric-key algorithm for the encryption of digital data.
  - Although its short key length of 56 bits makes it too insecure for applications,
  - it has been highly influential in the advancement of cryptography.
  - Key size – 56 bits; Block size – 64 bits
  - History
    - Developed in the early 1970s at IBM.
    - Submitted to the National Bureau of Standards (NBS).
    - In 1976, after consultation with the National Security Agency (NSA).

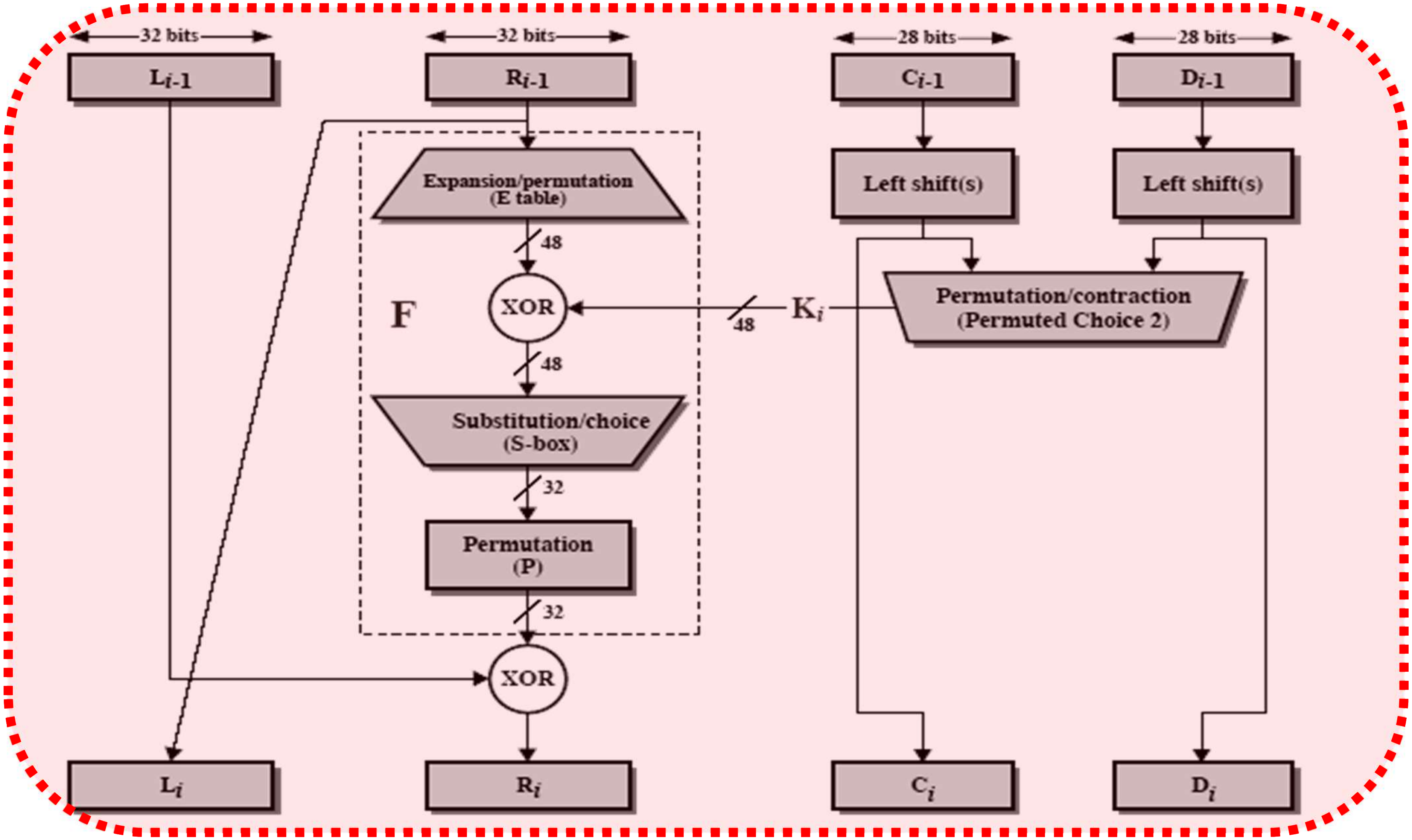
Data Encryption Standard (2/3)

- DES Algorithm



Data Encryption Standard (3/3)

- **DES Encryption (one round)**



澳門理工大學
Universidade Politécnica de Macau
Macao Polytechnic University

Digital Logic

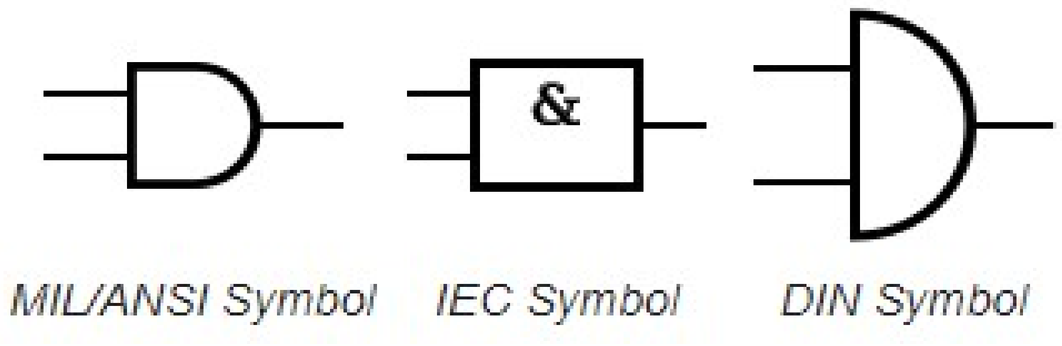
Digital Logic (1/5)

- Digital Electronics:
  - Field of electronics involving the study of digital signals and the engineering of devices that use or produce them.
  - This is in contrast to analog electronics and analog signals.
  - Complex devices may have simple electronic representations of Boolean logic functions.

- Boolean Logic (Boolean Algebra)
  - The branch of algebra in which the values of the variables are the truth values true and false, usually denoted 1 and 0, respectively → **Digital Logic.**
  - Truth Values
    - True → 1 / False → 0

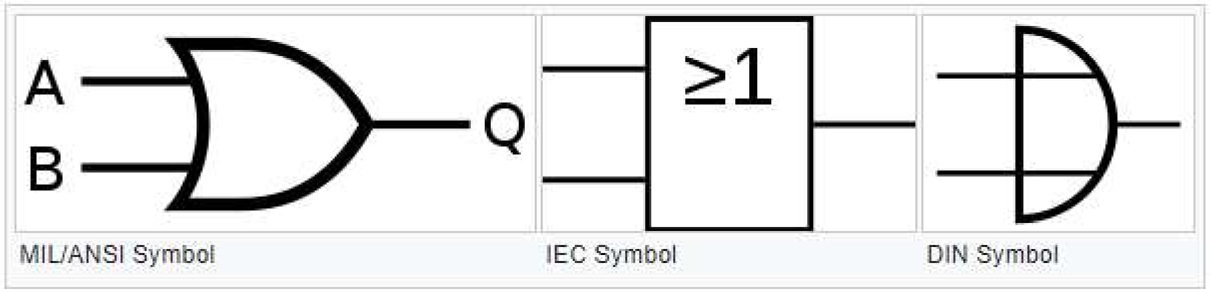
Digital Logic (2/5)

- AND Gate



INPUT		OUTPUT
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

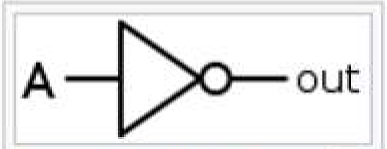
- OR Gate



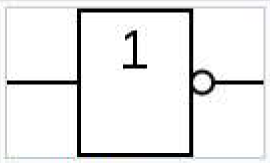
INPUT		OUTPUT
A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Digital Logic (3/5)

- NOT Gate (Inverter)



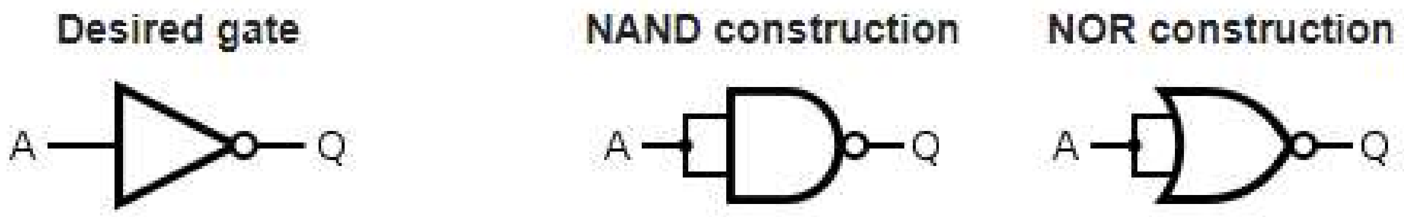
Traditional NOT Gate (Inverter) symbol



International Electrotechnical Commission NOT Gate (Inverter) symbol

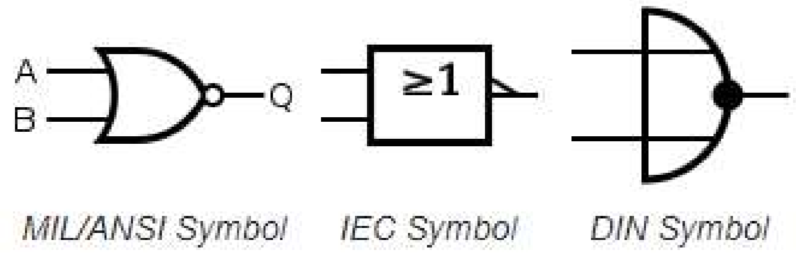
INPUT	OUTPUT
A	NOT A
0	1
1	0

- Variation of NOT Gate



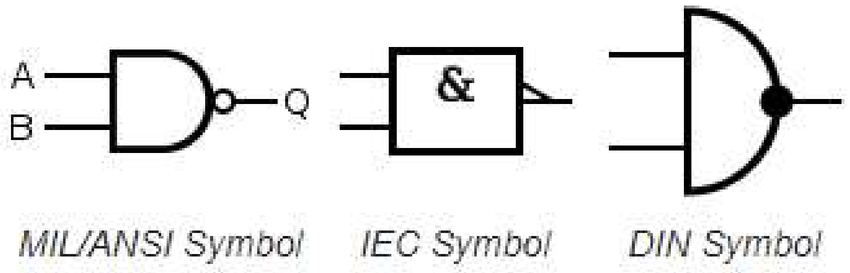
Digital Logic (4/5)

- NOR Gate



INPUT		OUTPUT
A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

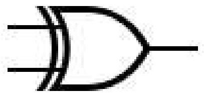
- NAND Gate

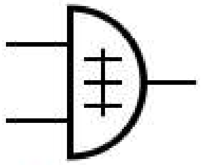


INPUT		OUTPUT
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

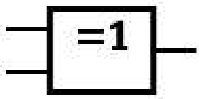
Digital Logic (5/5)

- XOR (Exclusive OR) Gate


ANSI XOR Schematic Symbol



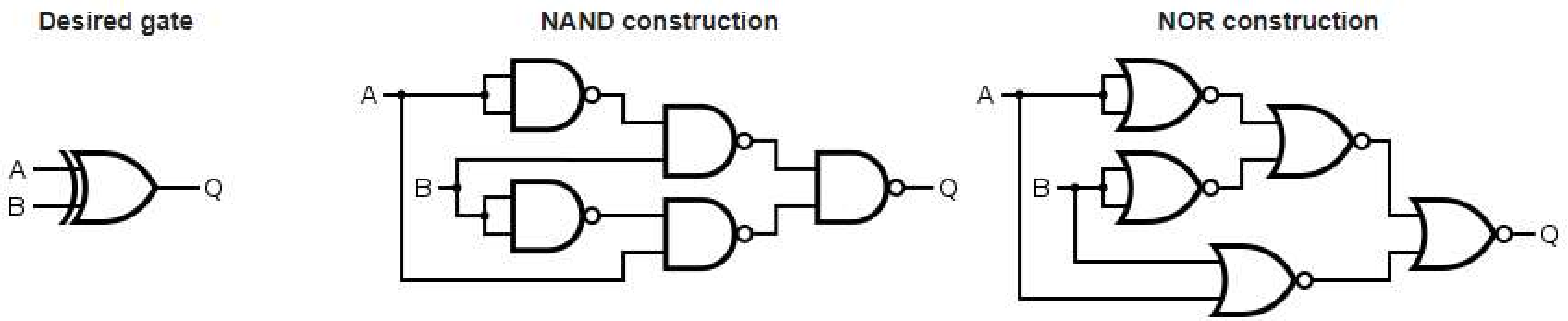
DIN XOR Schematic Symbol



IEC XOR Schematic Symbol

INPUT		OUTPUT
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

- XOR Composition



澳門理工大學
Universidade Politécnica de Macau
Macao Polytechnic University

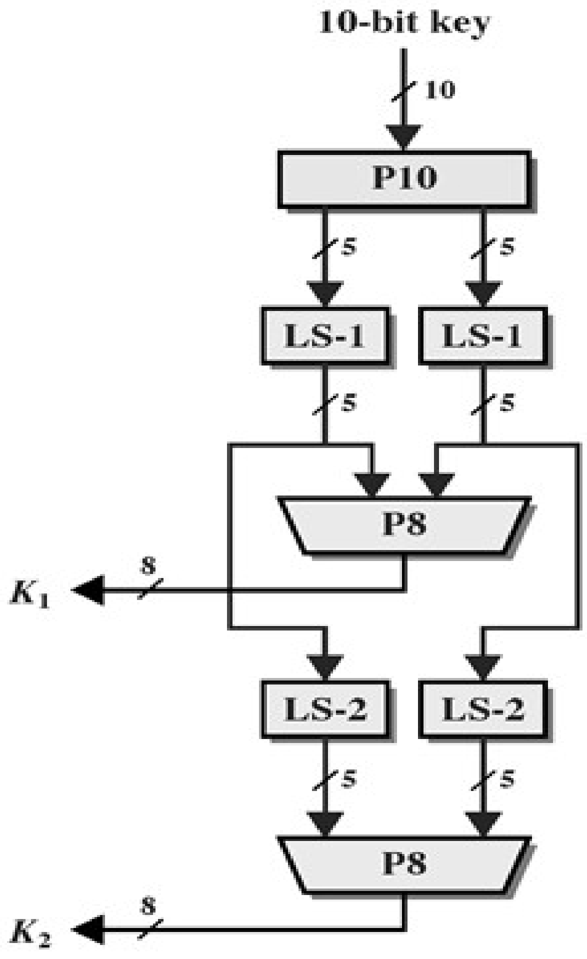
Simplified DES

Simplified DES (1/10)

- Input (plaintext) block: 8-bits
- Output (ciphertext) block: 8-bits
- Key: 10-bits
- Rounds: 2
- Round keys generated using permutations and left shifts
- **Encryption:**
  - Initial permutation, round function, switch halves
- **Decryption:**
  - Same as encryption, except round keys used in opposite order.

Simplified DES (2/10)

- **Key Generation**



P10 (10 bit Permutation)

(Position)

P10({1, 2, 3, 4, 5, 6, 7, 8, 9, 10})

= {3, 5, 2, 7, 4, 10, 1, 9, 8, 6}

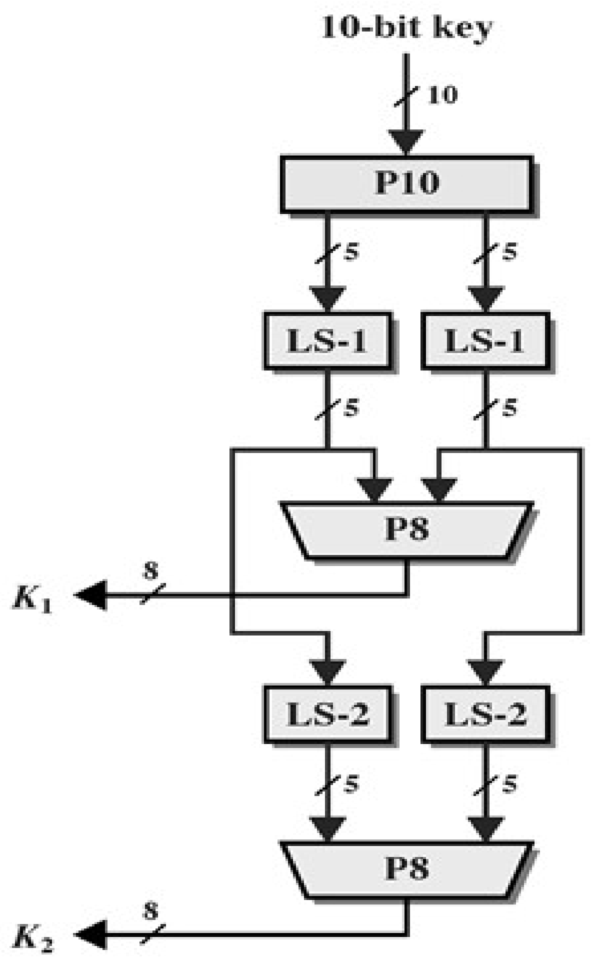
LS-1 (1 bit shift to left / 5 bit)

LS-1({1, 2, 3, 4, 5})

= {2, 3, 4, 5, 1}

Simplified DES (3/10)

- Key Generation



P8 (8 bit Permutation)

P8({1, 2, 3, 4, 5, 6, 7, 8, 9, 10})

= {6, 3, 7, 4, 8, 5, 10, 9}

→ **K_1**

LS-2 (2 bit shift to left / 5 bit)

LS-2({1, 2, 3, 4, 5})

= **LS-1** (**LS-1**{1, 2, 3, 4, 5})

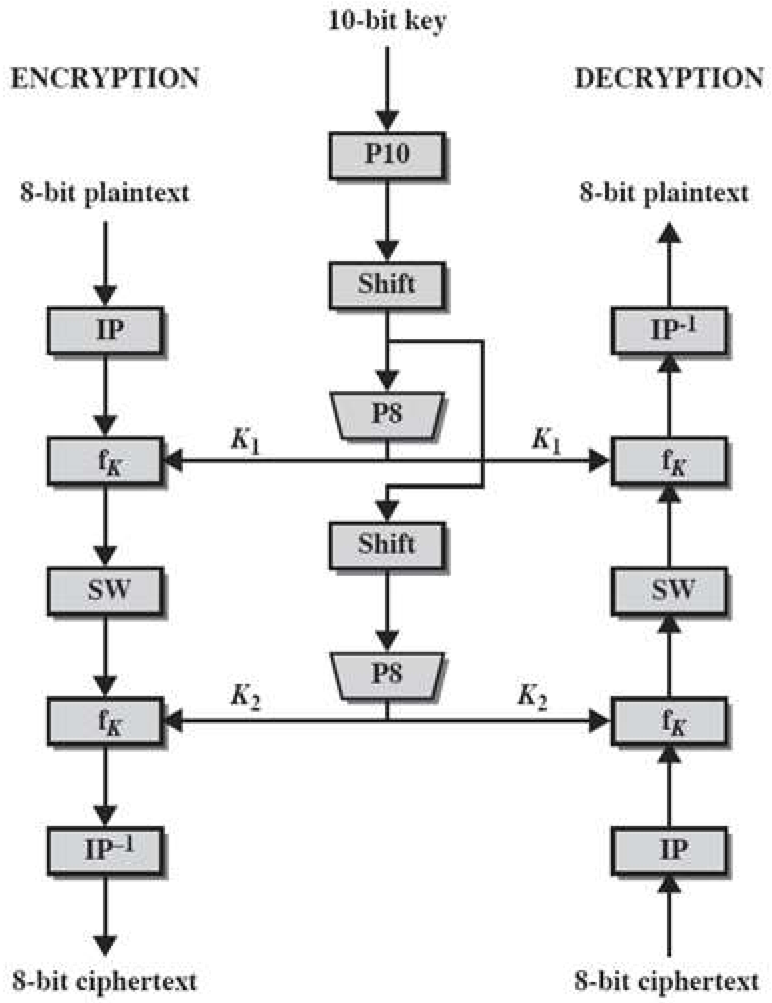
= **LS-1** ({2, 3, 4, 5,1})

= {3, 4, 5, 1, 2}

P8({LS-2_1, LS-2_2}) → **K_2**

Simplified DES (4/10)

- **S-DES Algorithm**



<u>IP (Initial Permutation)</u>

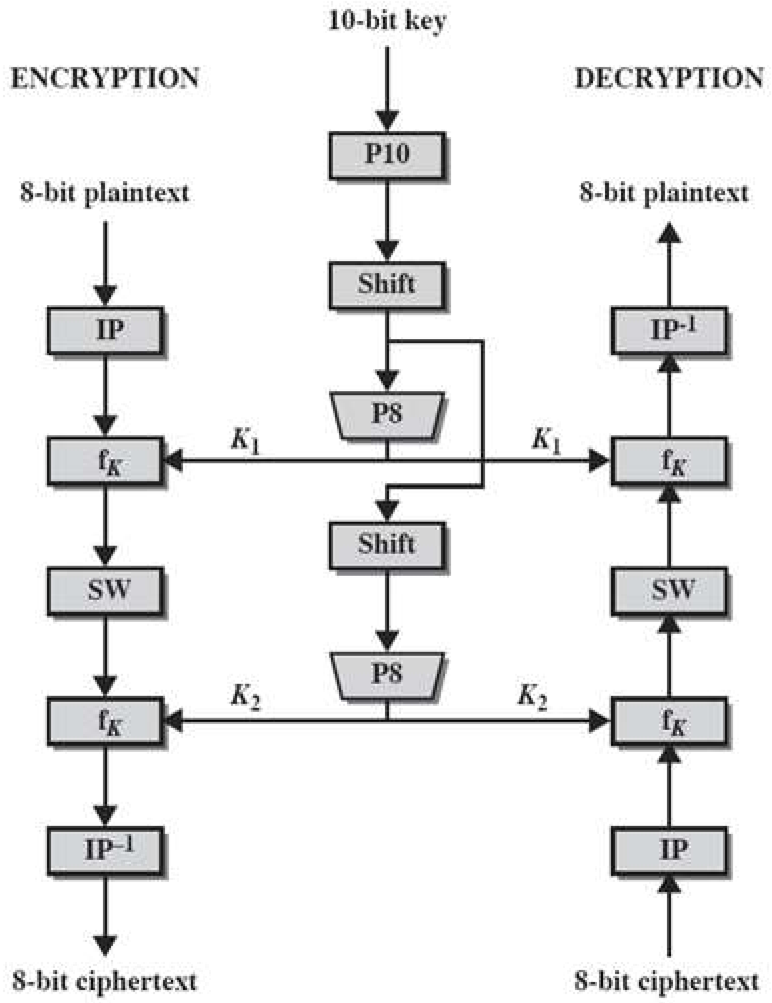
(Position)

IP = {2, 6, 3, 1, 4, 8, 5, 7}

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

Simplified DES (5/10)

- **S-DES Algorithm**



<u>Function f_k</u>

(Position)

IP = {2, 6, 3, 1, 4, 8, 5, 7}

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

<u>Function f_k</u>

$f_k(L , R) = (L \oplus F(R, SK), R)$

Simplified DES (6/10)

- **Function f_k**

IP({1, 2, 3, 4, 5, 6, 7, 8})

= {2, 6, 3, 1, 4, 8, 5, 7}

(Expansion/Permutation)

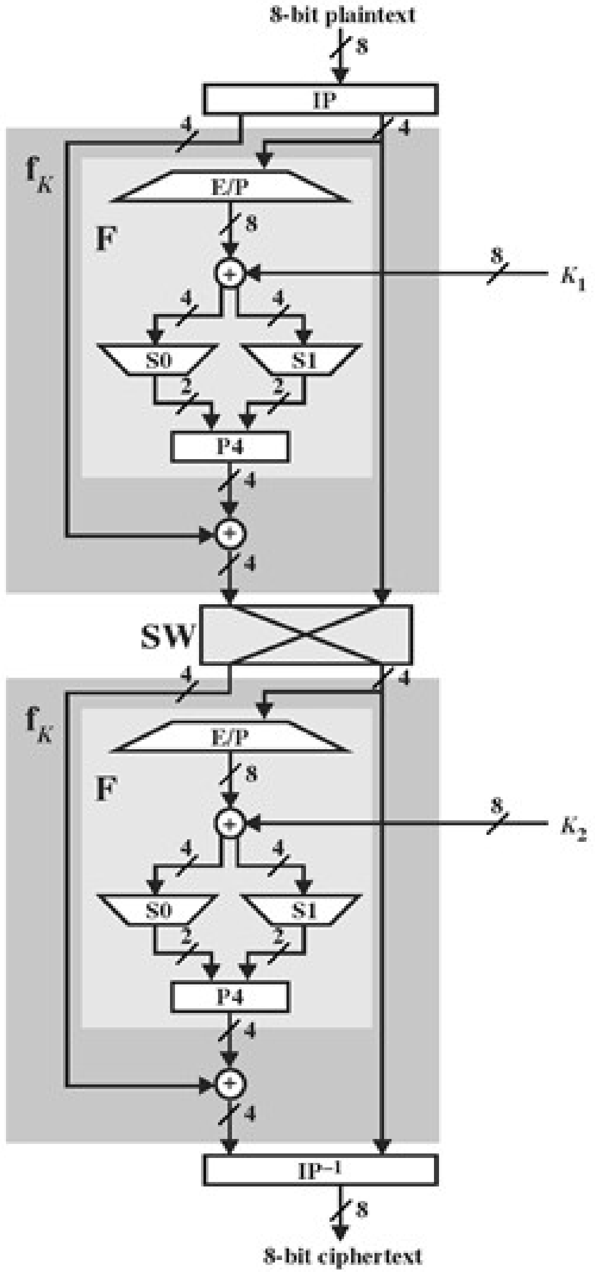
E/P({1,2,3,4})

= {{4,1,2,3}, {2,3,4,1}}

= {EP_1, EP_2}

P4({1, 2, 3, 4}) = {2, 4, 3, 1}

S0 & S1



Simplified DES (7/10)

- **S0 & S1**

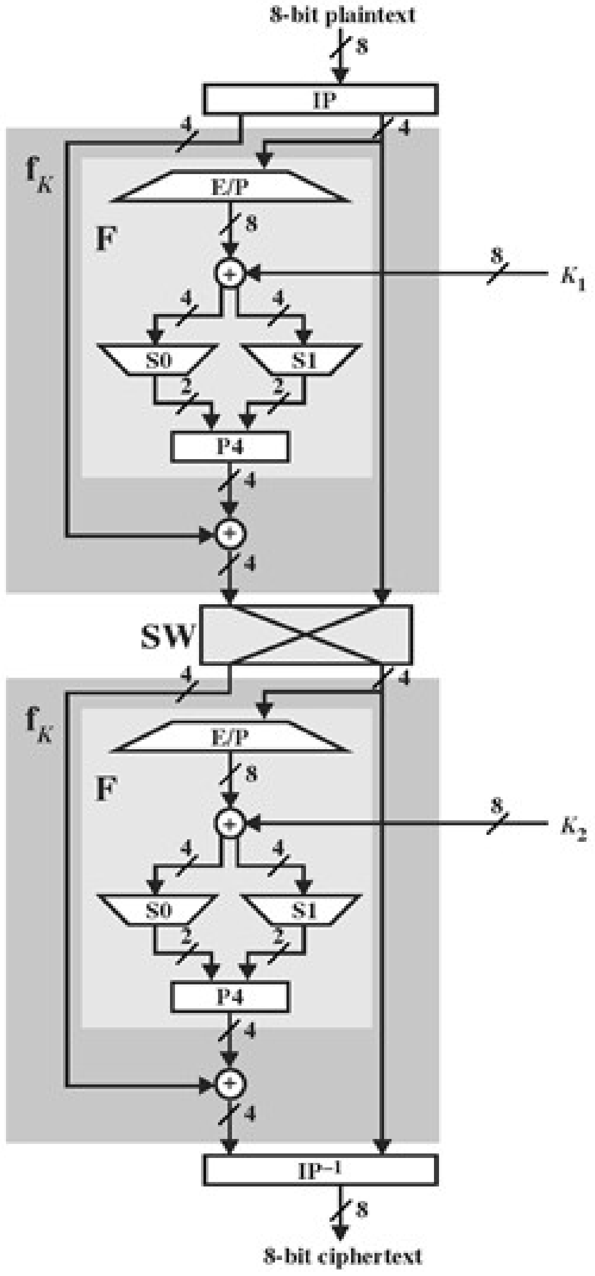
$\mathbf{K_1} = \{k_{11},\ldots,k_{18}\}$

(position)

$EP_1 = \{4,1,2,3\}$, $EP_2 = \{2,3,4,1\}$

$\{EP_1, EP_2\} \oplus \mathbf{K_1}$

$$= \begin{array}{c|cc|c} n_4 \oplus k_{11} & n_1 \oplus k_{12} & n_2 \oplus k_{13} & n_3 \oplus k_{14} \\ n_2 \oplus k_{15} & n_3 \oplus k_{16} & n_4 \oplus k_{17} & n_1 \oplus k_{18} \end{array}$$



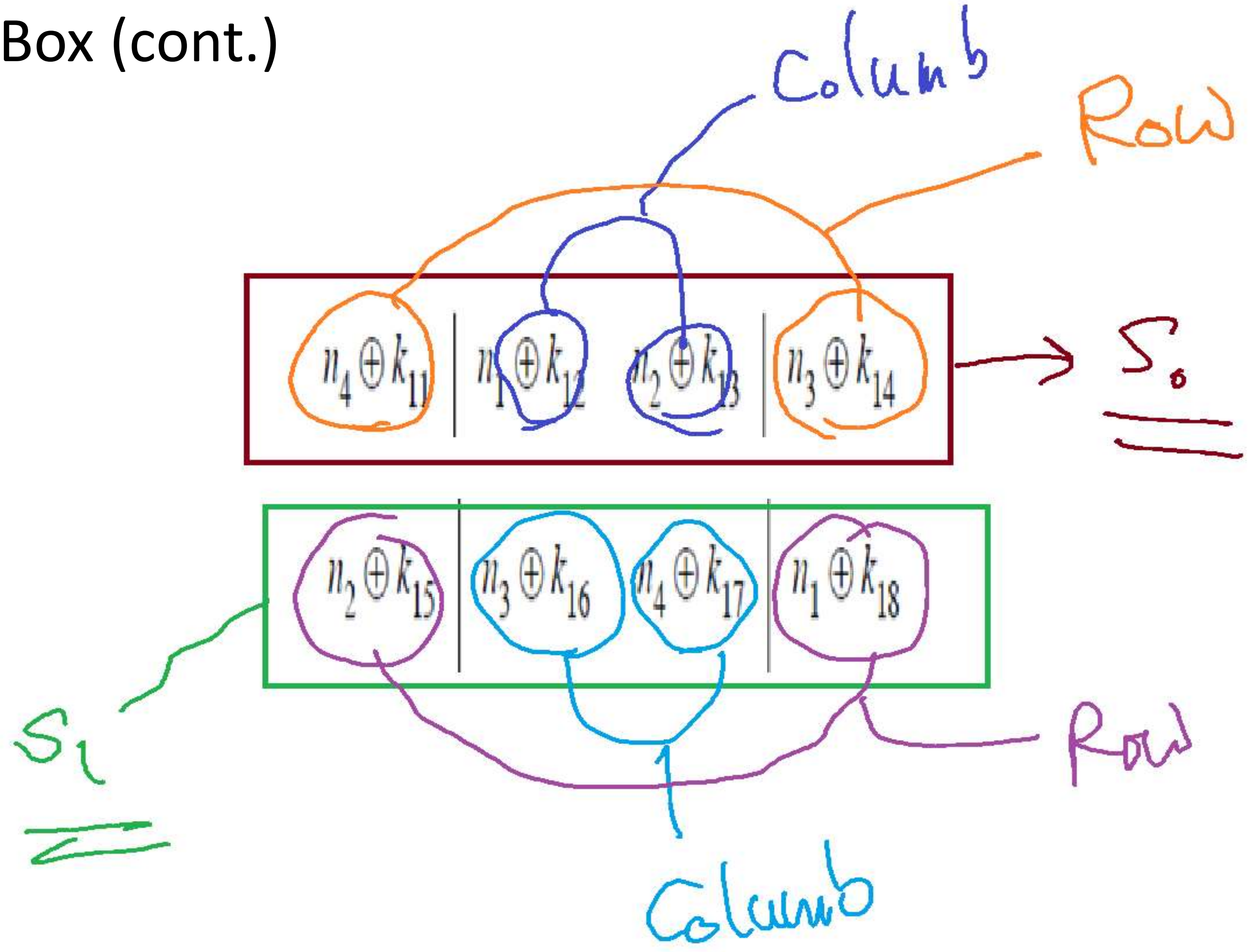
Simplified DES (8/10)

- S-Box
  - S-DES (and DES) perform substitutions using S-Boxes
  - S-Box considered as a matrix: input used to select row/column; selected element is output
  - 4-bit input: bit1; bit2; bit3; bit4
    - bit1 , bit4 species row (0, 1, 2 or 3 in decimal)
    - bit2, bit3 species column
    - 2-bit output

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

Simplified DES (9/10)

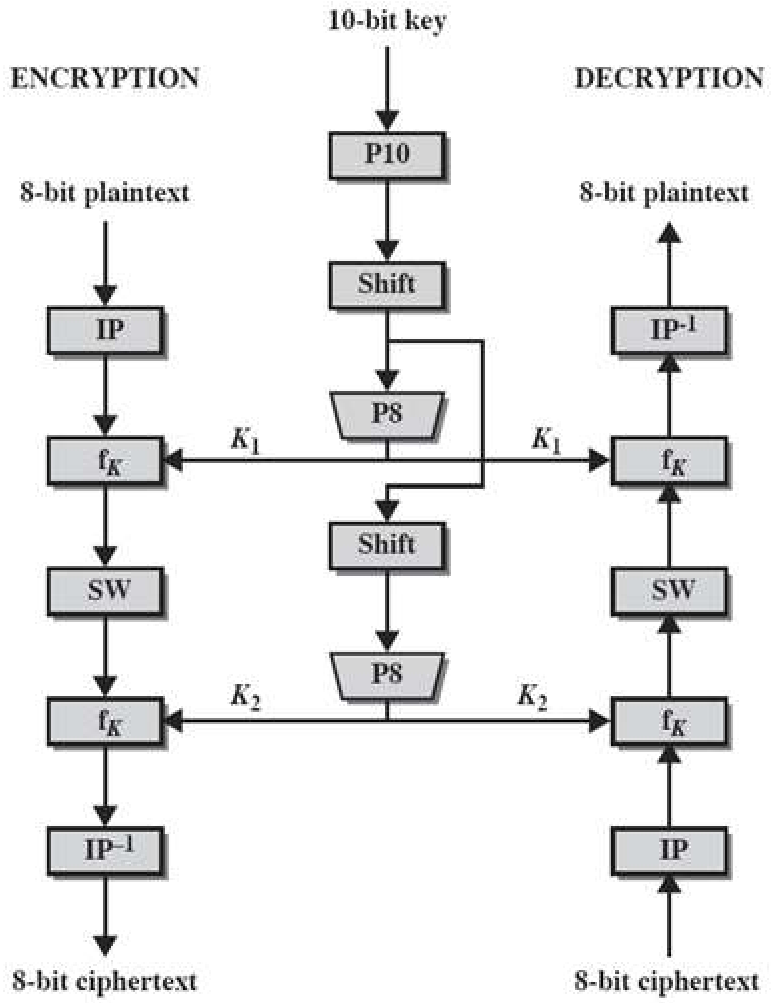
- S-Box (cont.)



Columb
Row
$n_4 \oplus k_{11}$ | $n_1 \oplus k_{12}$ | $n_2 \oplus k_{13}$ | $n_3 \oplus k_{14}$
S_0
$n_2 \oplus k_{15}$ | $n_3 \oplus k_{16}$ | $n_4 \oplus k_{17}$ | $n_1 \oplus k_{18}$
S_1
Row
Columb

Simplified DES (10/10)

- **S-DES Algorithm**



SW (Switch)

SW({**L**_IP_f_k, **R**_IP_f_k})

= {**R**_IP_f_k, **L**_IP_f_k}

Function f_k

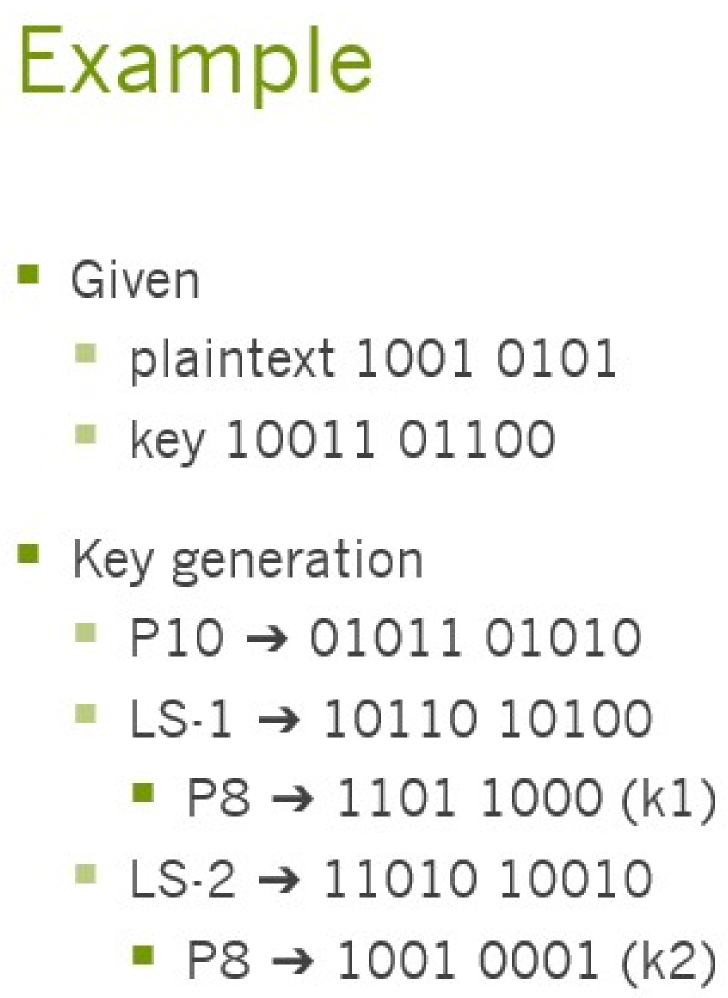
$f_k(L , R) = (L \oplus F(R, SK), R)$

where

SK = $\mathbf{K_2}$

$IP^{-1} = \{4, 1, 3, 5, 7, 2, 8, 6\}$

Simplified DES Example (1/3)



Example

- Given
  - plaintext 1001 0101
  - key 10011 01100
- Key generation
  - P10 → 01011 01010
  - LS-1 → 10110 10100
    - P8 → 1101 1000 (k1)
  - LS-2 → 11010 10010
    - P8 → 1001 0001 (k2)

Simplified DES Example (2/3)

- Encrypt: **1001 0101**
  - IP → 0101 1100 **L: 0101 R: 1100**
  - $f_k(L, R) = (L \oplus F(R, S_{K1}), R)$
    - EP → 0110 1001
    - $\oplus K_1$ → 1011 0001
    - S0 → row 11, col 01 → 01
    - S1 → row 01, col 00 → 10
    - P4 → 1010
    - $0101 \oplus 1010$ → 1111
  - SW → 1100 1111

L: 1100 R: 1111

- **$f_k(L, R) = (L \oplus F(R, S_{K2}), R)$**
  - **EP → 1111 1111**
  - **$\oplus K_2$ → 0110 1110**
  - **S_0 → row 00, col 11 → 10**
  - **S_1 → row 10, col 11 → 00**
  - **P4 → 0001**
  - **$1100 \oplus 0001$ → 1101**
- **IP^{-1} → 1101 1111**

Ciphertext: **1101 1111**

Simplified DES Example (3/3)

- Decrypt: **1101 1111**
  - IP → 1101 1111 **L: 1101 R: 1111**
  - $f_k(L, R) = (L \oplus F(R, S_{K2}), R)$
    - EP → 1111 1111
    - $\oplus K_2$ → 0110 1110
    - S0 → row 00, col 11 → 10
    - S1 → row 10, col 11 → 00
    - P4 → 0001
    - 1101 ⊕ 0001 → 1100
  - SW → 1111 1100

L: 1111 R: 1100

- $f_k(L, R) = (L \oplus F(R, S_{K1}), R)$
  - EP → 0110 1001
  - $\oplus K_1$ → 1011 0001
  - S_0 → row 11, col 01 → 01
  - S_1 → row 01, col 00 → 10
  - P4 → 1010
  - 1111 ⊕ 1010 → 0101
- IP^{-1} → 1001 0101

Plaintext: **1001 0101**

Simplified DES Summery

- Simplified DES expressed as functions:

$$\mathbf{CT} = IP^{-1} \circ f_{K2} \circ SW \circ f_{K1} \circ IP(\mathbf{PT})$$

$$\mathbf{PT} = IP^{-1} \circ f_{K1} \circ SW \circ f_{K2} \circ IP(\mathbf{CT})$$

- Security of Simplified DES:
  - 10-bit key, 1024 keys: brute force easy.
  - If know plaintext and corresponding ciphertext, can we determine key? → **Very hard**

澳門理工大學
Universidade Politécnica de Macau
Macao Polytechnic University

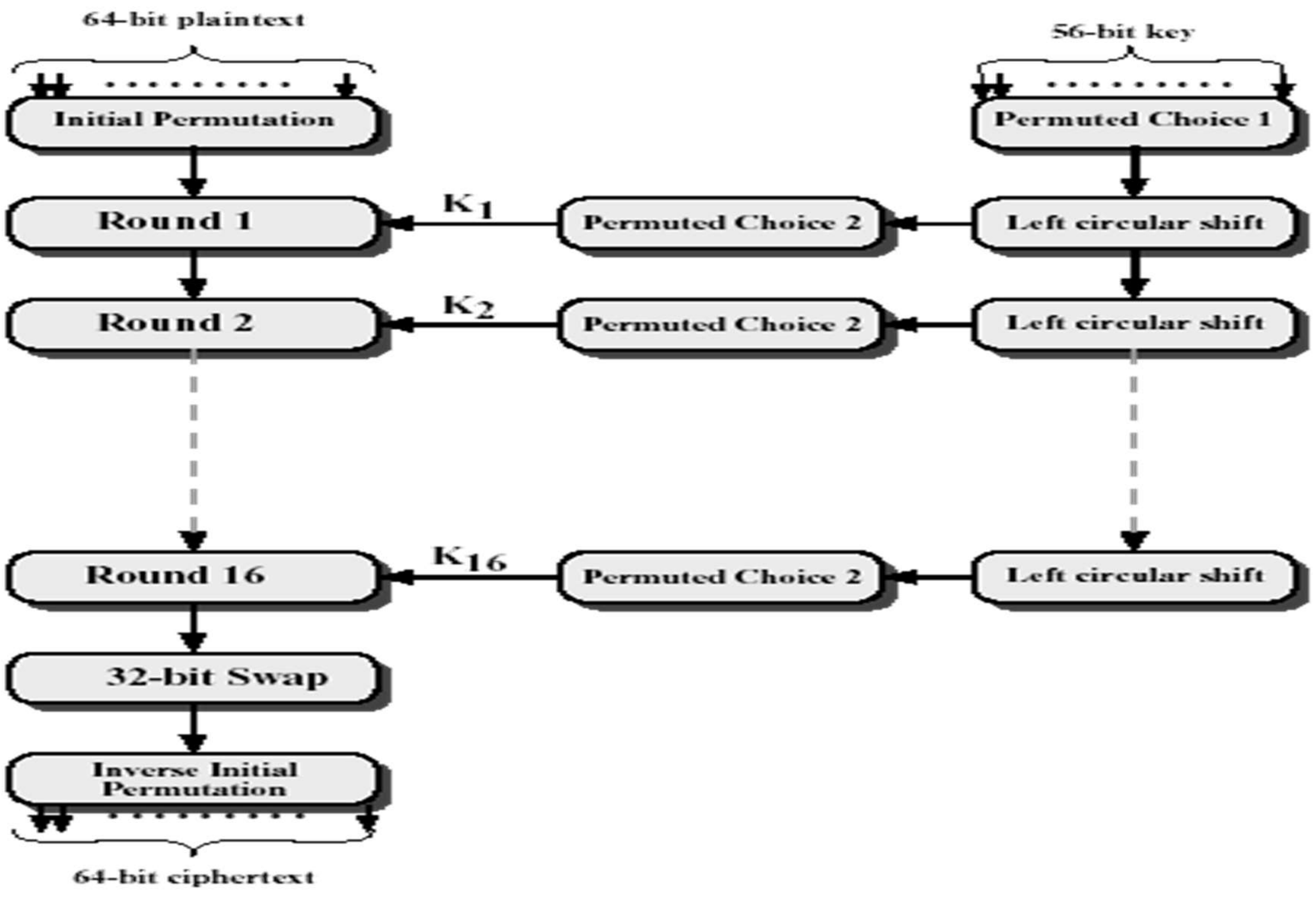
Data Encryption Standard (DES) Reloaded

Data Encryption Standard (1/22)

- Overview of the DES Encryption Algorithm
  - Works very similar to SDES but more complex and secured.
  - The basic process in enciphering a 64-bit data block with a 56-bits key using the DES consists of:
    - An initial permutation (IP)
    - 16 rounds of a complex key dependent calculation f
    - A final permutation, being the inverse of IP

Data Encryption Standard (2/22)

- DES Algorithm



Data Encryption Standard (3/22)

DES Key Schedule

- The subkeys used in each round are formed by the key schedule which has:
  - An initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - Selecting 24-bits from each half
    - Permuting them by PC2 for use in function f
    - Shifting each half separately either 1 or 2 places depending on the key left circularly shifting schedule (KS)
- SK_i = PC2(K_S(PC1(key), i))

Data Encryption Standard (4/22)

PC1

- PC1 is used to select 56 of 64 bits supplied as the key
- Every 8th bit is discarded (assumed to be parity)
- PC1 also splits the key bits into 2 halves (C and D)
- Note in DES number bits from 1 (left, MSB) to 32/64 (right, LSB)

-
  ```
  57, 49, 41, 33, 25, 17,  9,  C Half
   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,  D Half
   7, 62, 54, 46, 38, 30, 22,
  14,  6, 61, 53, 45, 37, 29,
  21, 13,  5, 28, 20, 12,  4
  ```

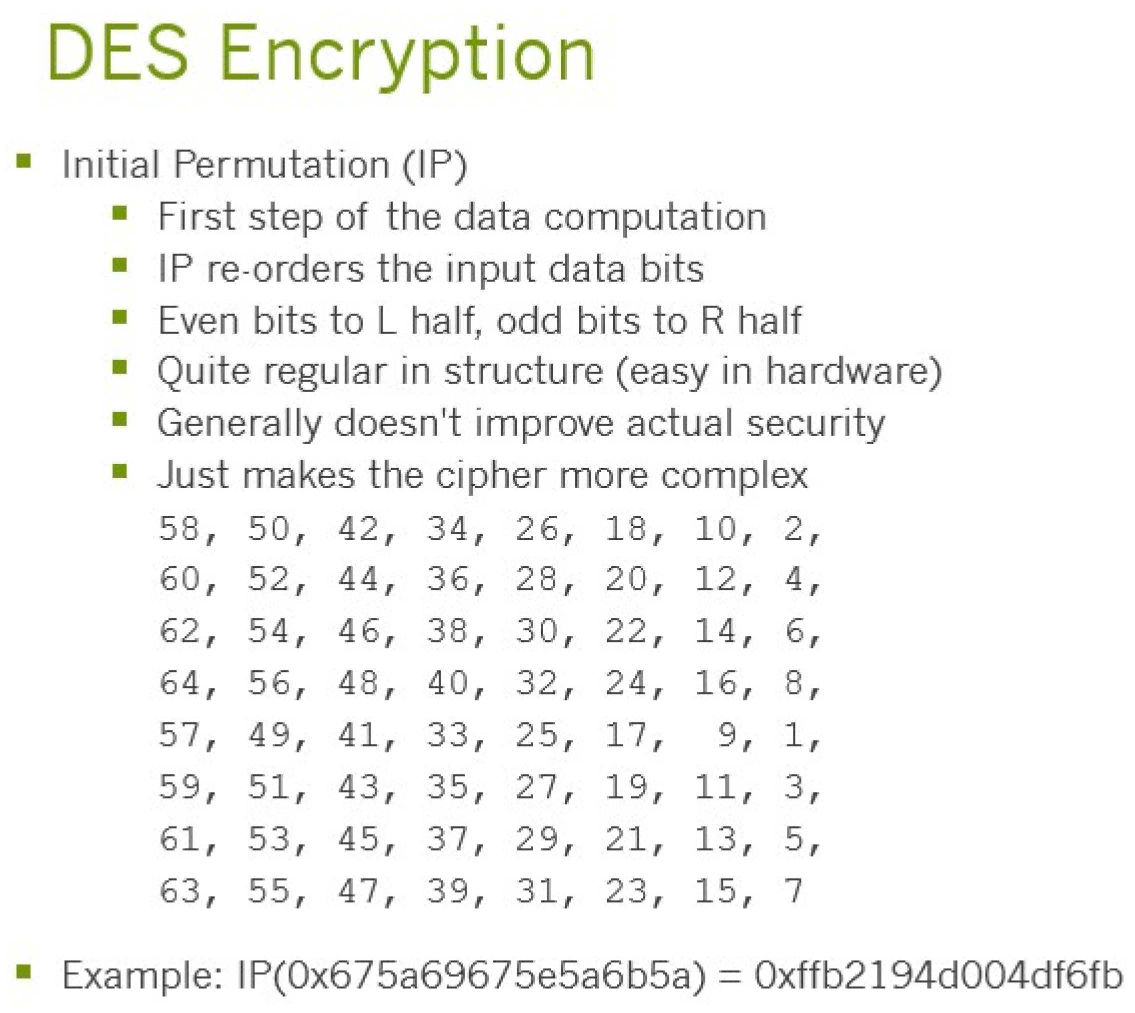
Data Encryption Standard (5/22)

Left Circular Shift & PC2

- Key shifting schedule is specified as:
  - Round: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
  - Shifts: 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1
- PC2 is used to select 48 bits (actually 2 lots of 24 bits) used in each round of the DES data computation

4 bits are taken off	14,	17,	11,	24,	1,	5,	C half
	3,	28,	15,	6,	21,	10,	(bits 1-28)
	23,	19,	12,	4,	26,	8,	
	16,	7,	27,	20,	13,	2,	
4 bits are taken off	41,	52,	31,	37,	47,	55,	D half
	30,	40,	51,	45,	33,	48,	(bits 29-56)
	44,	49,	39,	56,	34,	53,	
	46,	42,	50,	36,	29,	32	

Data Encryption Standard (6/22)



DES Encryption

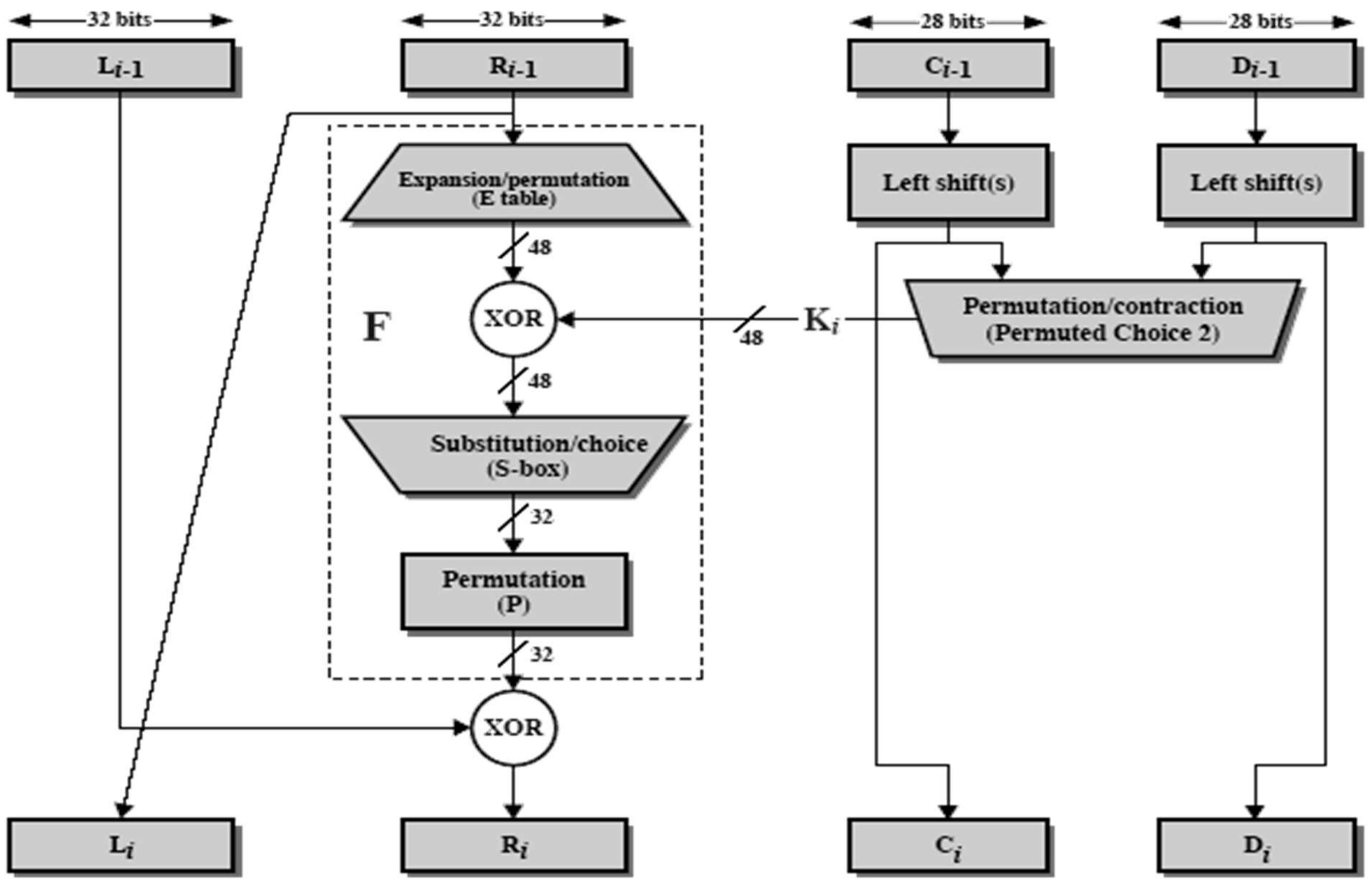
- Initial Permutation (IP)
  - First step of the data computation
  - IP re-orders the input data bits
  - Even bits to L half, odd bits to R half
  - Quite regular in structure (easy in hardware)
  - Generally doesn't improve actual security
  - Just makes the cipher more complex

```
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
```

- Example: IP(0x675a69675e5a6b5a) = 0xffb2194d004df6fb

Data Encryption Standard (7/22)

- **DES Encryption (one round)**



Data Encryption Standard (8/22)

DES Encryption (E)

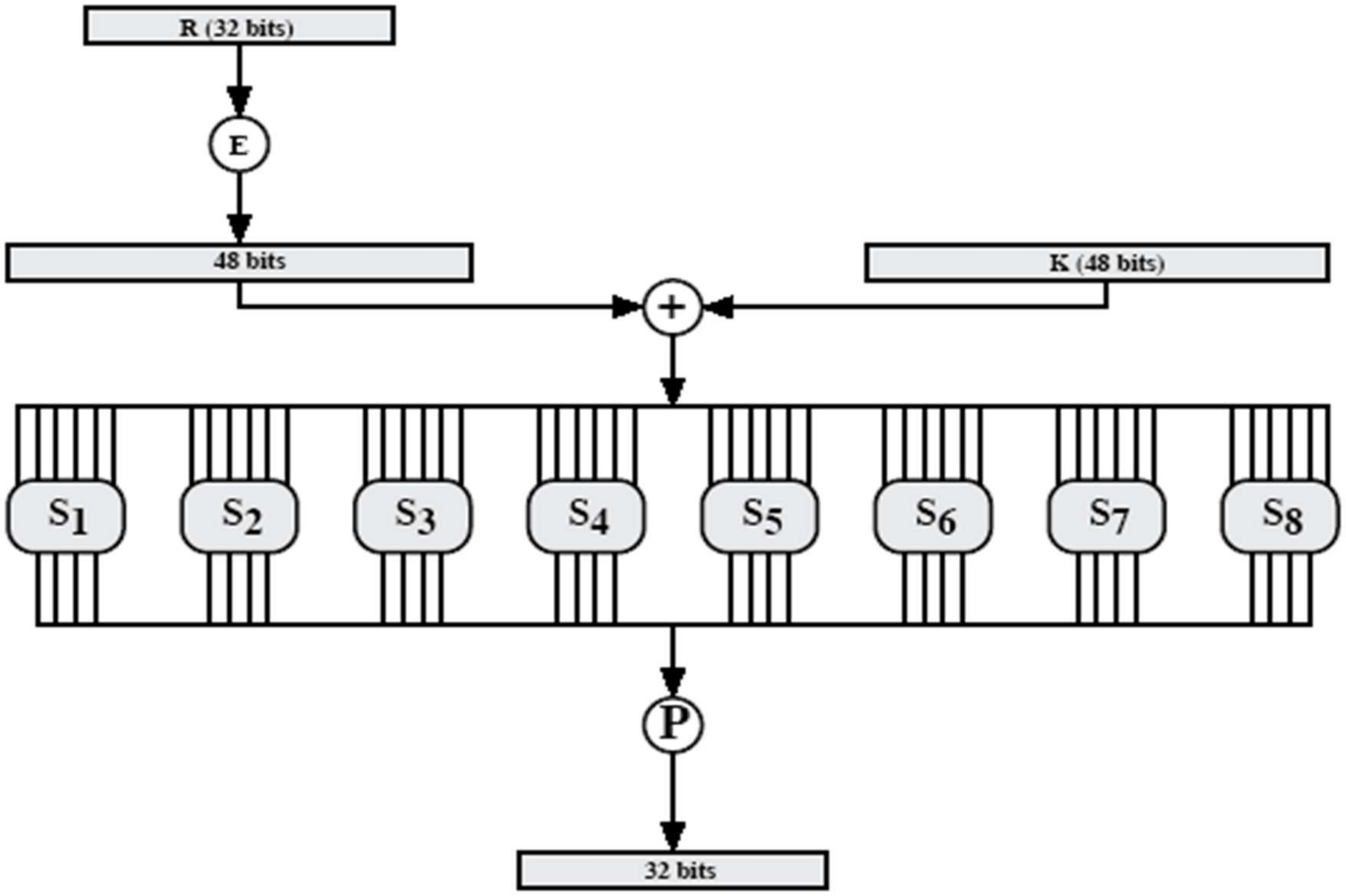
- Expansion Function E
  - Expands R side data input from 32 to 48 bits by duplicating some bits specifically split input into 8 groups of 4 bits then duplicate bits from either side to form groups of 6 bits
  - Provides auto-keying effect for following S-boxes

32,	1,	2,	3,	4,	5,	S1
4,	5,	6,	7,	8,	9,	S2
8,	9,	10,	11,	12,	13,	S3
12,	13,	14,	15,	16,	17,	S4
16,	17,	18,	19,	20,	21,	S5
20,	21,	22,	23,	24,	25,	S6
24,	25,	26,	27,	28,	29,	S7
28,	29,	30,	31,	32,	1	S8

- Example: E(0x004df6fb) = 0x80025bfad7f6

Data Encryption Standard (9/22)

- **DES Encryption (F)**



Data Encryption Standard (10/22)

- DES Encryption (S-Box): Substitution **S-Boxes**
  - There are 8 S-boxes, each of which maps 6 bits to 4 bits
  - Each S-box is actually of size 4 x 16
  - Outer bits 1 and 6 (row bits) select one of the 4 rows
  - Inner bits 2-5 (col bits) are substituted for 4 others
  - Result is 8 x 4 bits, or 32 bits
  - Example: 011001 ➔ row 01, col 1100 ➔ (1, 12) ➔ 9

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	12	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

Data Encryption Standard (11/22)

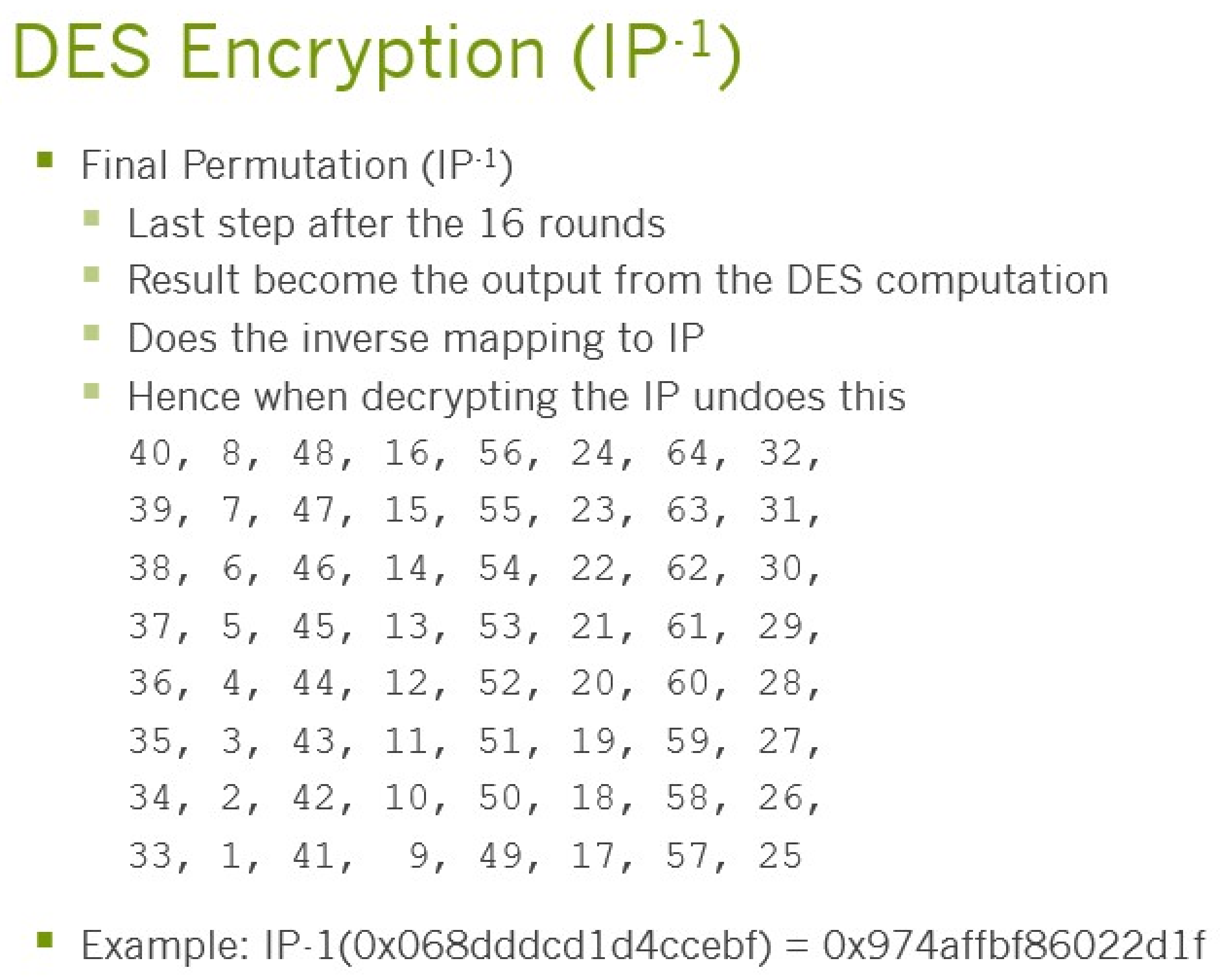
DES Encryption (P)

- Permutation P
  - P diffuses the S-box outputs across different S-box inputs in the next round
  - Ensures each S-Box outs affect both row and col bits

```
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
```

- Example: P(0x5fd25e03) = 0x746fc91a
  - 0101 1111 1101 0010 0101 1110 0000 0011
  - 0111 0100 0110 1111 1100 1001 0001 1010

Data Encryption Standard (12/22)



DES Encryption (IP^{-1})

- Final Permutation (IP^{-1})
  - Last step after the 16 rounds
  - Result become the output from the DES computation
  - Does the inverse mapping to IP
  - Hence when decrypting the IP undoes this

```
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
```

- Example: IP-1(0x068dddcd1d4ccebf) = 0x974affbf86022d1f

Data Encryption Standard (13/22)

DES Decryption

- To decrypt a block of data, we must repeat all steps of data computation
- Subkeys are used in reverse order (SK_{16} first, then SK_{15}, SK_{14}, SK_{13}, ... etc)
- Note that IP undoes the final IP^{-1} step of encryption
- First round with SK_{16} undoes 16th round of encryption, etc. till 16th round with SK_1 undoes 1st encryption round
- Final IP^{-1} undoes the initial encryption IP, thus recovering the original data value

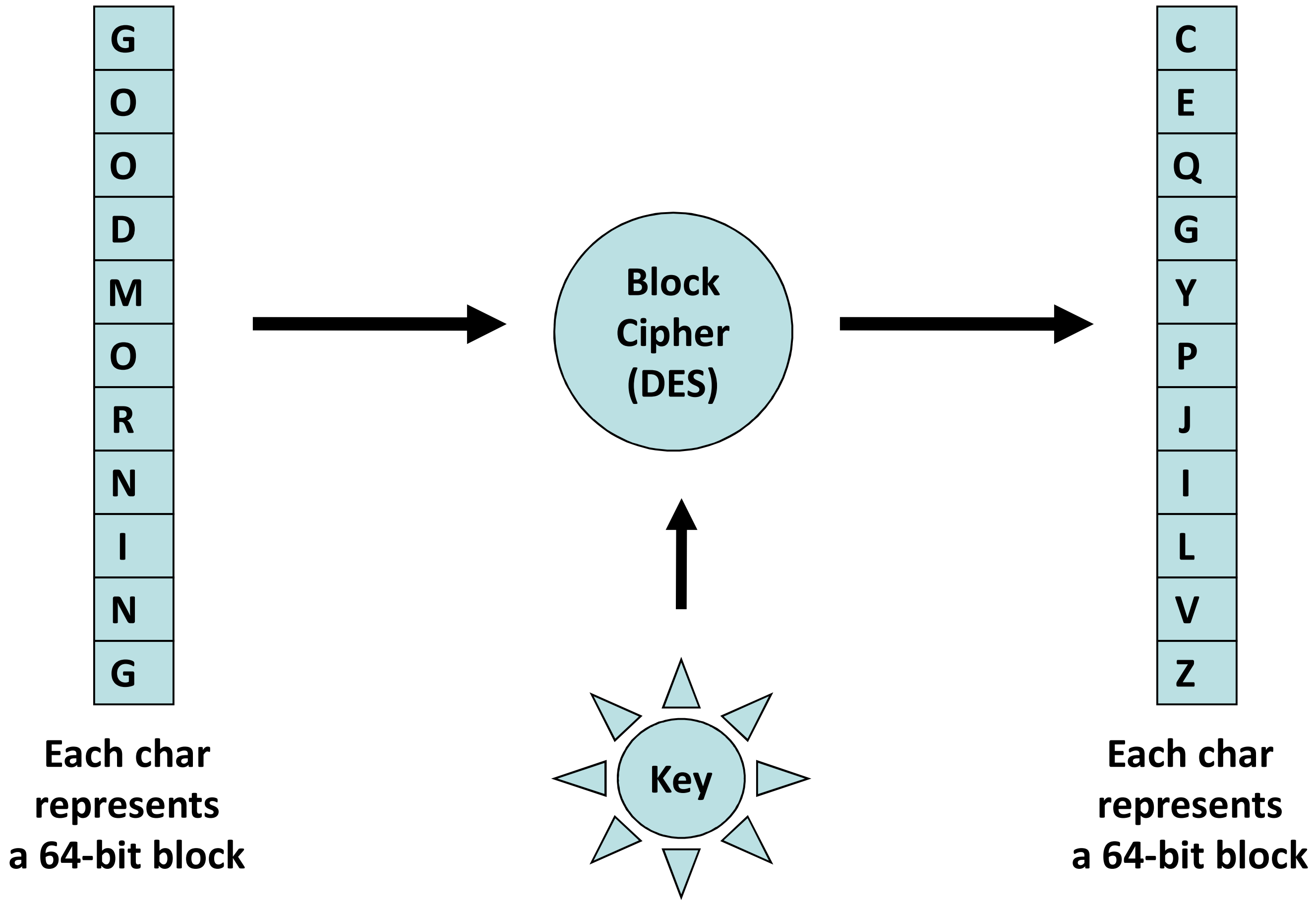
Data Encryption Standard (14/22)

Strength of DES

- 56-bit key size on a single machine performing one encryption per microsecond took a thousand years to break the cipher
- In 1977, 1 million encryption devices working on parallel took about 10 hours for a brute-force attack. (US$20 millions)
- In July 1998, DES cracker was built for less than US$250,000 and it took three days for search for the key provided that some degree of knowledge about the expected plaintext is needed
- Faster machine makes cracking DES easier
- It is worthy to note that design criteria for DES were not made public so a backdoor may be open for some people

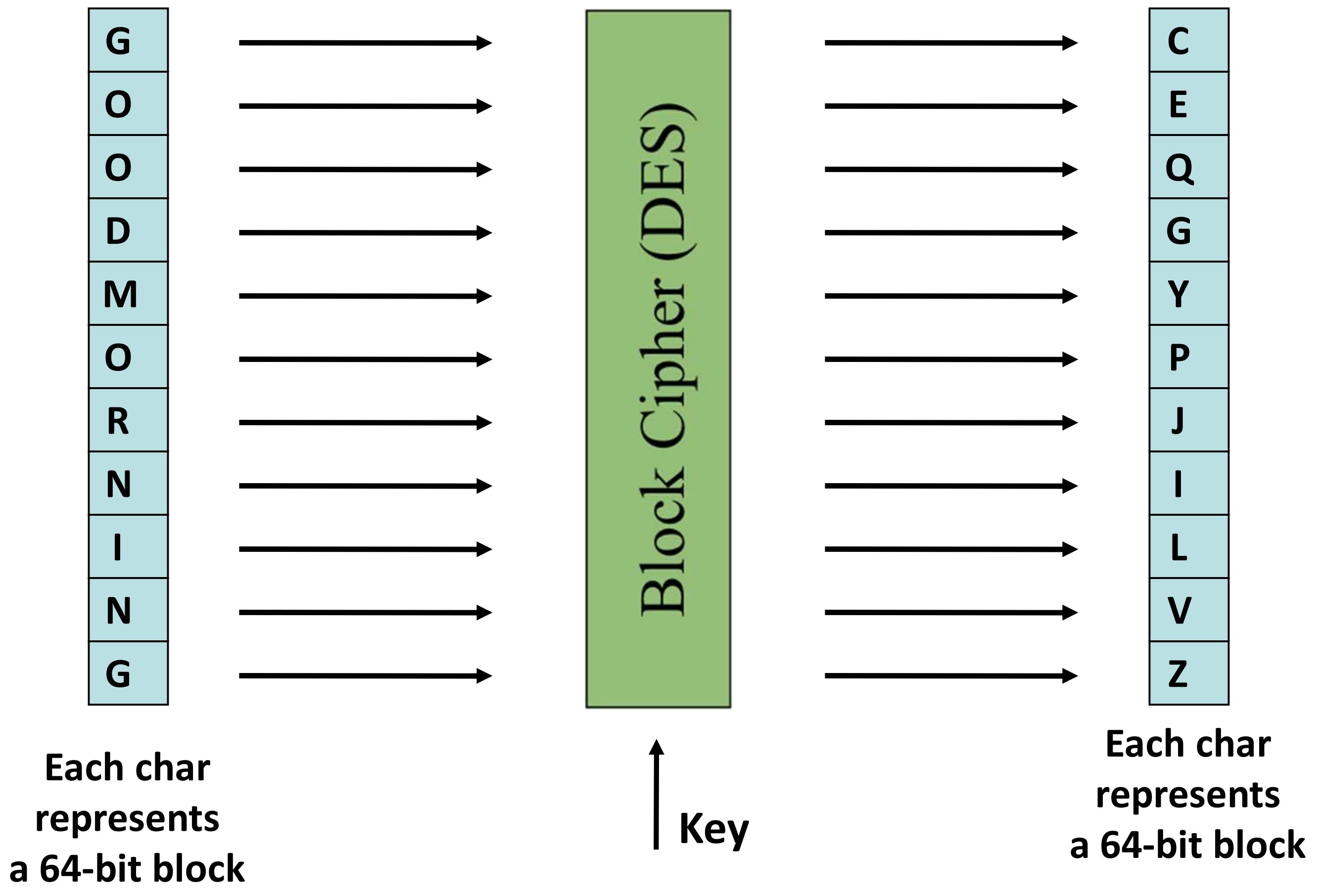
Data Encryption Standard (15/22)

- Mode of Use


G
O
O
D
M
O
R
N
I
N
G
Each char represents a 64-bit block
Block Cipher (DES)
Key
C
E
Q
G
Y
P
J
I
L
V
Z
Each char represents a 64-bit block

Data Encryption Standard (16/22)

- Mode of Use (cont.)

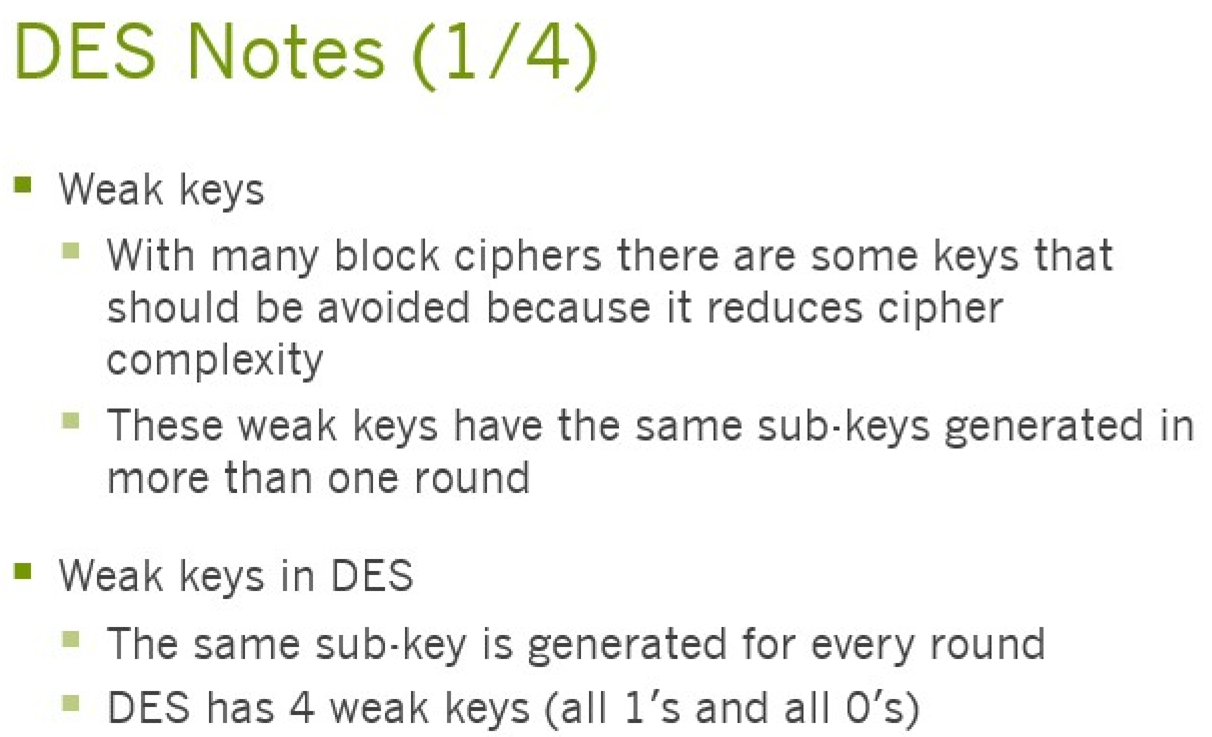

G
O
O
D
M
O
R
N
I
N
G
Block Cipher (DES)
C
E
Q
G
Y
P
J
I
L
V
Z
Each char represents a 64-bit block
Key
Each char represents a 64-bit block

Data Encryption Standard (17/22)

- Mode of Use (cont.)
  - Block ciphers encrypt a fixed size block of information.
  - DES encrypts 64-bit blocks of data, using a 56-bit key.
  - Need some way of specifying how to use it in practice, given that we usually have an arbitrary amount of information to encrypt.

- Modes are either:
  - **Block Modes** processes message in blocks (ECB, CBC)
  - **Stream Modes** processes message as a bit/byte stream (CFB, OFB)

Data Encryption Standard (18/22)



DES Notes (1/4)

- Weak keys
  - With many block ciphers there are some keys that should be avoided because it reduces cipher complexity
  - These weak keys have the same sub-keys generated in more than one round
- Weak keys in DES
  - The same sub-key is generated for every round
  - DES has 4 weak keys (all 1's and all 0's)

```
11 00   00 11
11 11   00 00
```

Data Encryption Standard (19/22)

DES Notes (2/4)

- Semi-weak keys in DES
  - Only two sub-keys are generated on alternate rounds
  - DES has 12 of these (all 1's, 0's and alternate 1's, 0's)

```
10 10   10 01
01 10   01 01
```

```
11 10   00 10
11 01   00 01
```

```
10 11   10 00
01 11   01 00
```

- Handling weak keys
  - Weak keys do not cause a problem but it is a hole
  - They are a tiny fraction of all available keys ($16/2^{56}$)
  - However they MUST be avoided by any key generation program

Data Encryption Standard (20/22)

DES Notes (3/4)

- DES S-box design
  - The input and output have a relationship that is nonlinear and difficult to approximate with linear functions.
  - Provide avalanche effect.
- DES key schedule
  - The rotations are used to present different bits of the key for selection on successive rounds.
  - To maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key.

Data Encryption Standard (21/22)

DES Notes (4/4)

- DES Permutations Design
  - 5 permutations: IP and IP^{-1}, P, E, PC1, PC2
  - IP and IP^{-1} and PC1 serve NO crypto logical function since they merely rearrange bits in a regular pattern. Searches may be constructed with their effect ignored.
  - E, P, and PC2 act with S-Boxes to ensure dependence of the output bits on the input bits and key bits, i.e., provide avalanche effect
- Although the standard for DES is public. The design criteria used are classified. A few has since been made public and we have seen reverse engineering to derive others

Data Encryption Standard (22/22)

- Simplified DES:

$$\mathbf{CT} = IP^{-1} \circ f_{K2} \circ \{SW\} \circ f_{K1} \circ IP(\mathbf{PT})$$

$$\mathbf{PT} = IP^{-1} \circ f_{K1} \circ \{SW\} \circ f_{K2} \circ IP(\mathbf{CT})$$

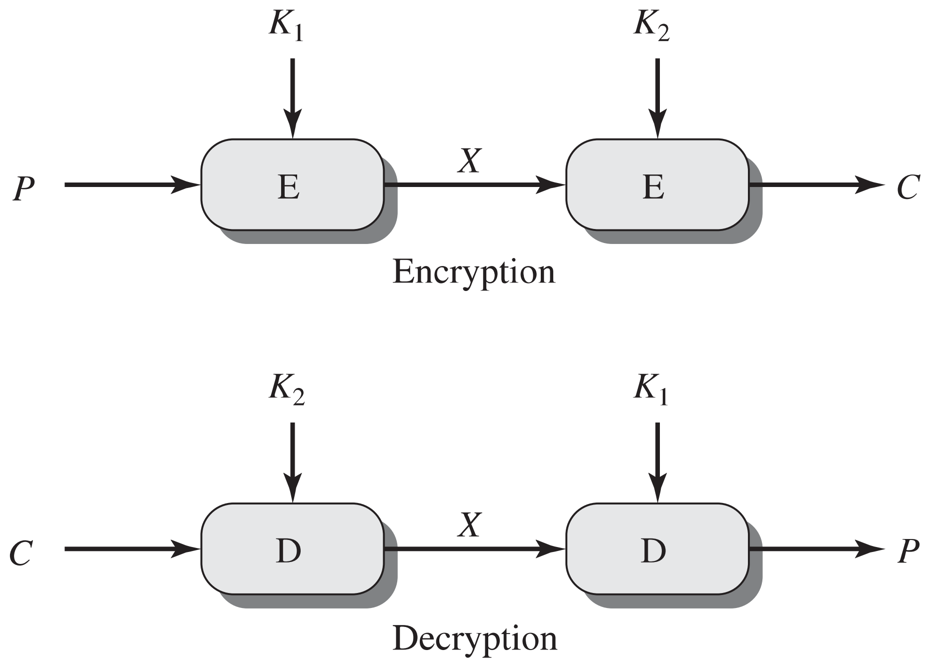
- DES

$$\mathbf{CT} = IP^{-1} \circ f_{K16} \circ \{SW \circ f_{K15} \circ \cdots \circ SW\} \circ f_{k1} \circ IP(\mathbf{PT})$$

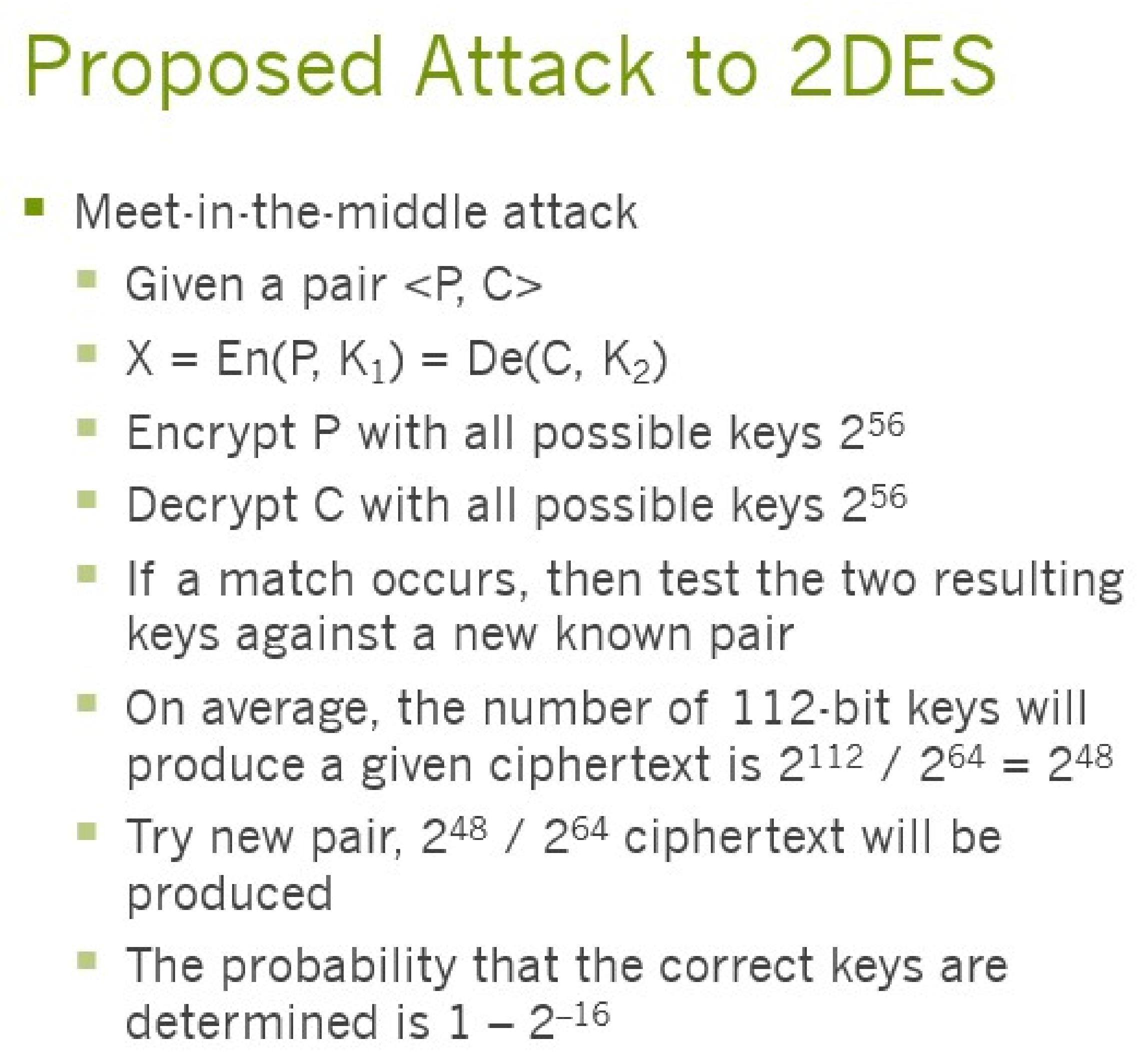
$$\mathbf{PT} = IP^{-1} \circ f_{K1} \circ \{SW \circ f_{K2} \circ \cdots \circ SW\} \circ f_{K16} \circ IP(\mathbf{CT})$$

Double DES (1/2)

- The idea is to do encryption twice with two keys:
  - $C = En(En(P, K_1), K_2)$
  - $P = De(De(C, K_2), K_1)$



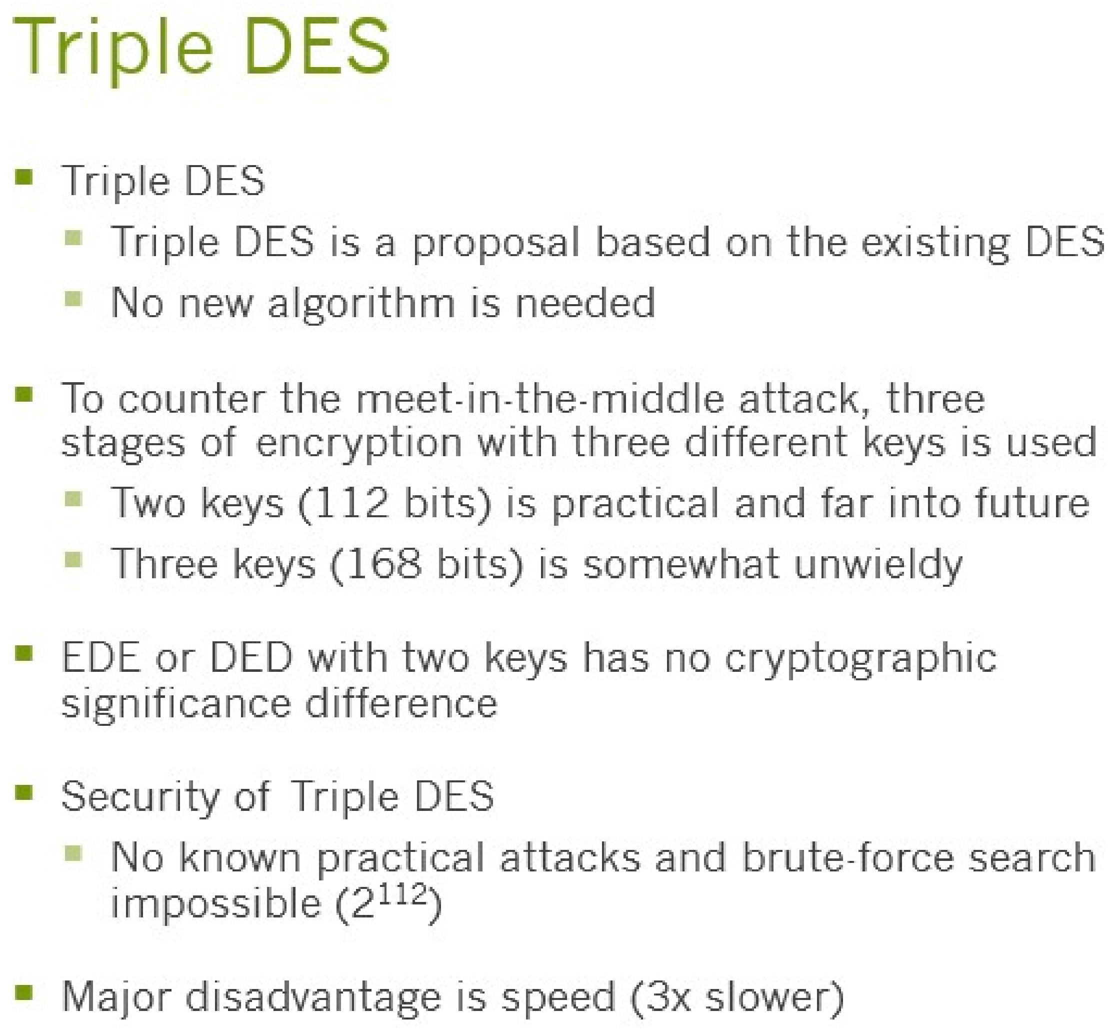
Double DES (2/2)



Proposed Attack to 2DES

- Meet-in-the-middle attack
  - Given a pair <P, C>
  - $X = En(P, K_1) = De(C, K_2)$
  - Encrypt P with all possible keys 2^{56}
  - Decrypt C with all possible keys 2^{56}
  - If a match occurs, then test the two resulting keys against a new known pair
  - On average, the number of 112-bit keys will produce a given ciphertext is $2^{112} / 2^{64} = 2^{48}$
  - Try new pair, $2^{48} / 2^{64}$ ciphertext will be produced
  - The probability that the correct keys are determined is $1 - 2^{-16}$

Triple DES (1/4)

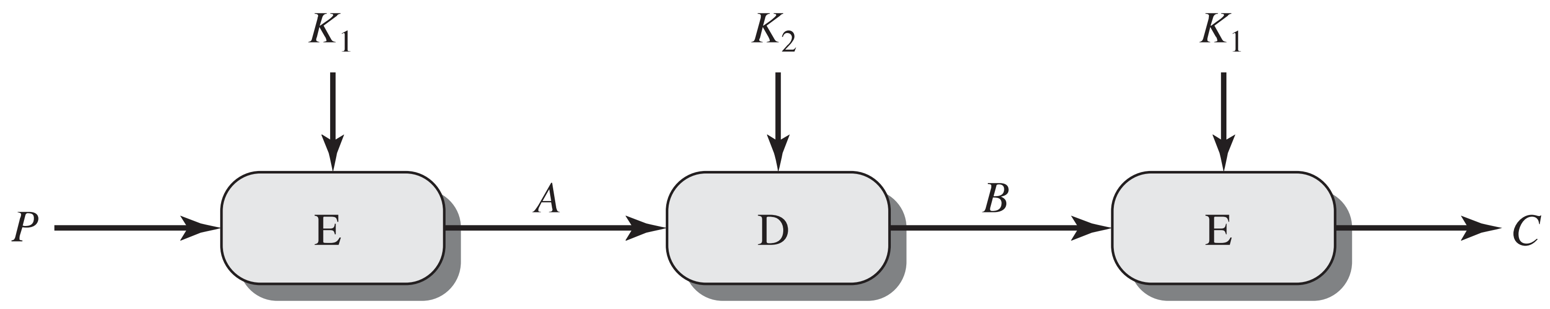


Triple DES

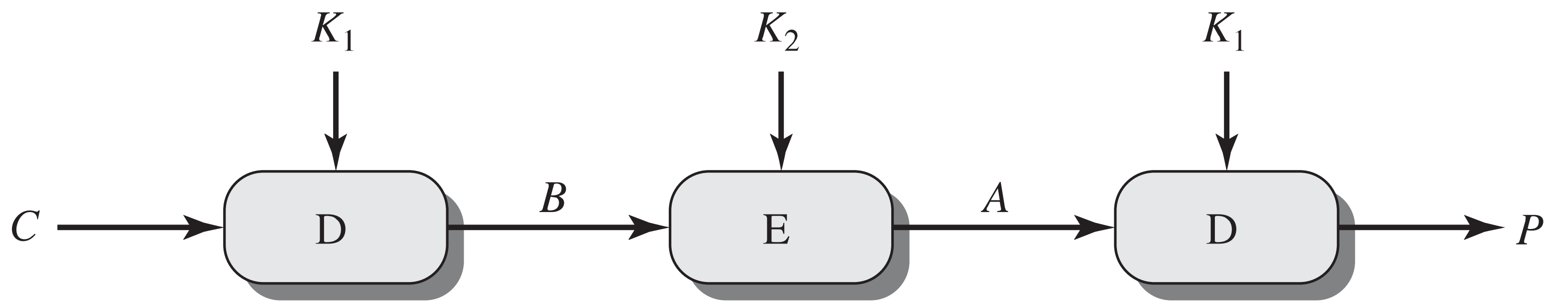
- Triple DES
  - Triple DES is a proposal based on the existing DES
  - No new algorithm is needed
- To counter the meet-in-the-middle attack, three stages of encryption with three different keys is used
  - Two keys (112 bits) is practical and far into future
  - Three keys (168 bits) is somewhat unwieldy
- EDE or DED with two keys has no cryptographic significance difference
- Security of Triple DES
  - No known practical attacks and brute-force search impossible (2^{112})
- Major disadvantage is speed (3x slower)

Triple DES (2/4)

- Proposed Attack to 3DES



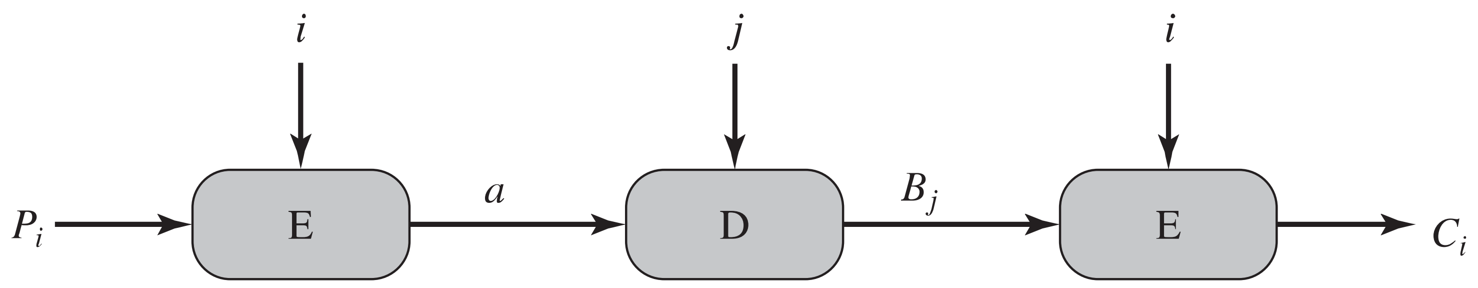
Encryption



Decryption

Triple DES (3/4)

- Proposed Attack to 3DES (cont.)
  - Prepare n pairs of <P, C> in Table 1
  - Pick up a random value for a and find all P with all possible 256 keys that produces a
  - Find all B by decrypting C with key *i* found above
  - For all 256 keys j, produces $B_j = D_j(a)$
  - If there is a match in Bj = Dj(a), test keys $[K_i, K_j]$ on other plaintext-ciphertext pairs



Triple DES (4/4)

- Proposed Attack to 3DES (cont.)
  - Probability of selecting the unique value that leads to success: $1/2^{64}$
  - Given n <P, C> pairs, the probability of success is n/264
  - The expected running time of the attack:
  - $(2^{56})(2^{64} / n) \rightarrow 2^{(120 - \log2\ n)}$

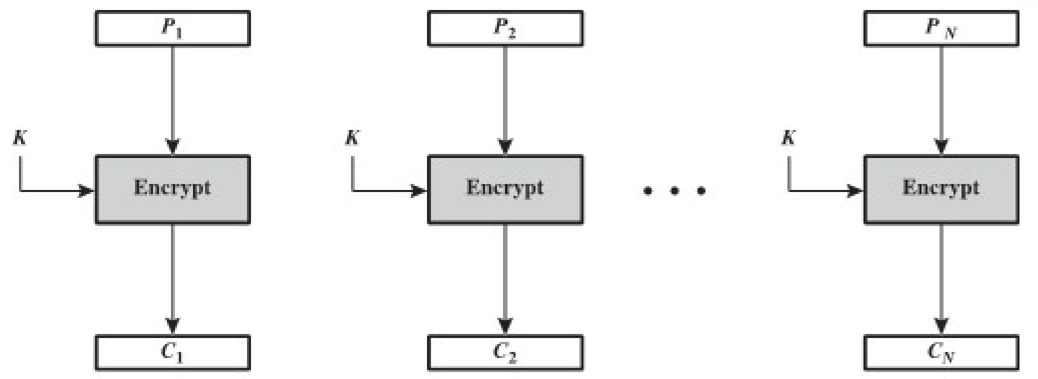
Other Block Ciphers

Electronic Codebook (ECB) (1/3)

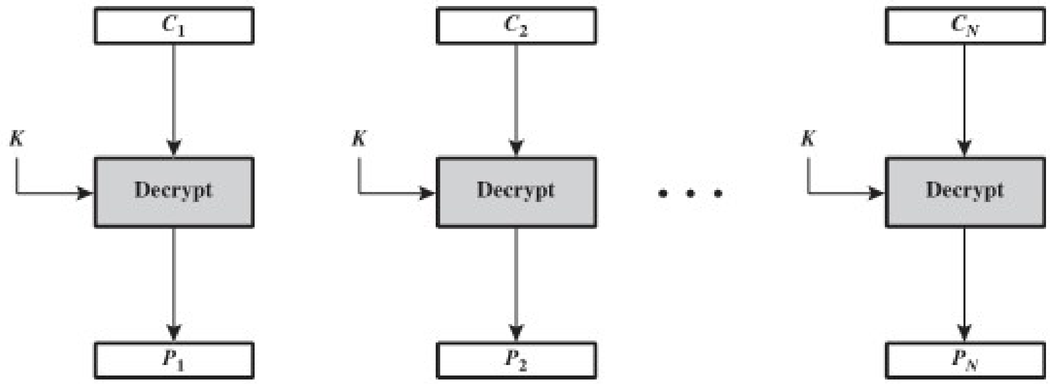
- Message is broken into independent blocks which are encrypted
- Each block is a value which is substituted, like a codebook, hence named
- Each block is independent of all others
- C_i = De(P_i, K_1)

Electronic Codebook (ECB) (2/3)

- Encryption



- Decryption



Electronic Codebook (ECB) (3/3)

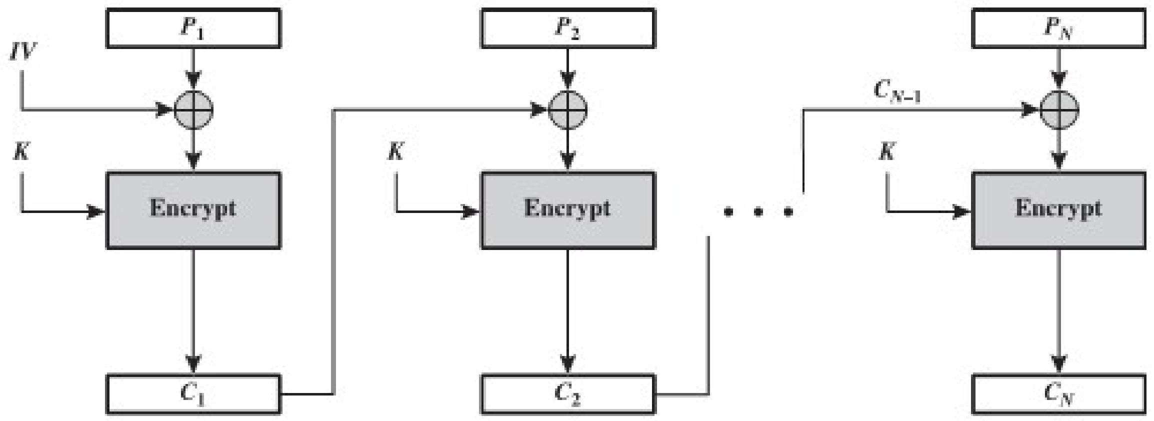
- Advantages and limitations of ECB
  - Repetitions in message can be reflected in ciphertext
  - If aligned with message block particularly with data such as graphics or with messages that change very little, which become a code-book analysis problem
  - Weakness due to encrypted message blocks being independent
  - Hence only really useful when sending a few blocks of data

Cipher Block Chaining (CBC) (1/3)

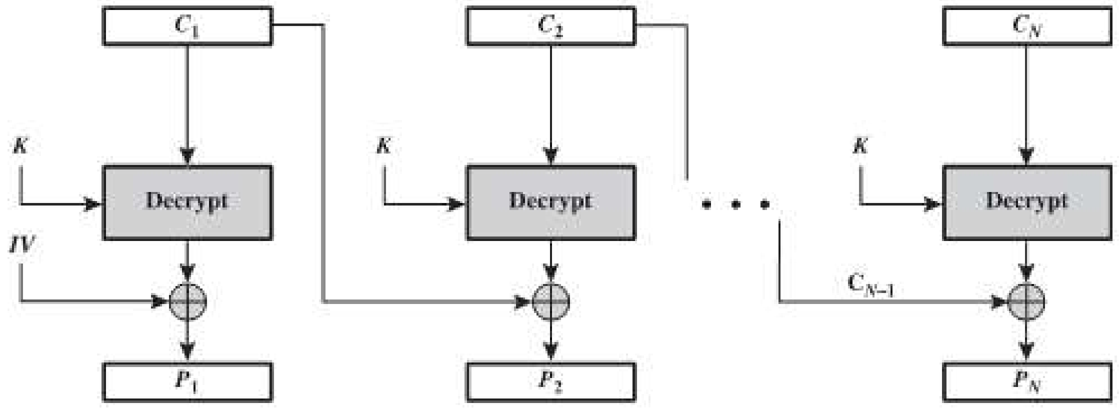
- Message is broken into blocks
- But these are linked together in the encryption operation.
- Cipher blocks are chained with plaintext, hence named.
- Use a known initial vector (IV) to start process
  - $C_i = En(P_i \oplus C_{i-1}, K_1)$
  - $C_0 = IV$

Cipher Block Chaining (CBC) (2/3)

- Encryption



- Decryption



Cipher Block Chaining (CBC) (3/3)

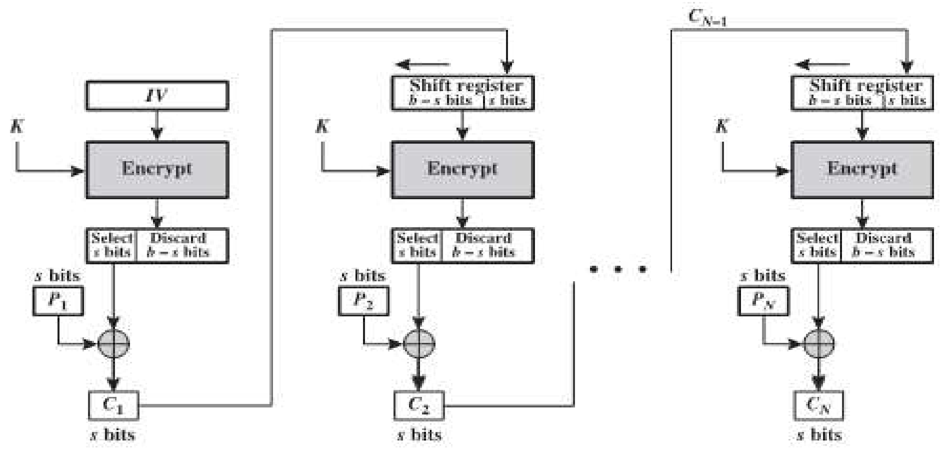
- Advantages and limitations of CBC
  - Each ciphertext block is dependent on all message blocks before it
  - Most common mode of use, a change in the message affects the ciphertext block as well as the original block
  - To start, need an initial value (IV) which must be known by both sender and receiver
    - IV must be a fixed value or it may be sent encrypted in ECB mode before rest of message
  - At the end of the message, have to handle a possible last short block
    - Either pad last block (usually with count of pad size)
    - Eg. [b1 b2 b3 0 0 0 0 5] ➔ 3 data bytes and 5 bytes pad+count

Cipher Feedback (CFB) (1/2)

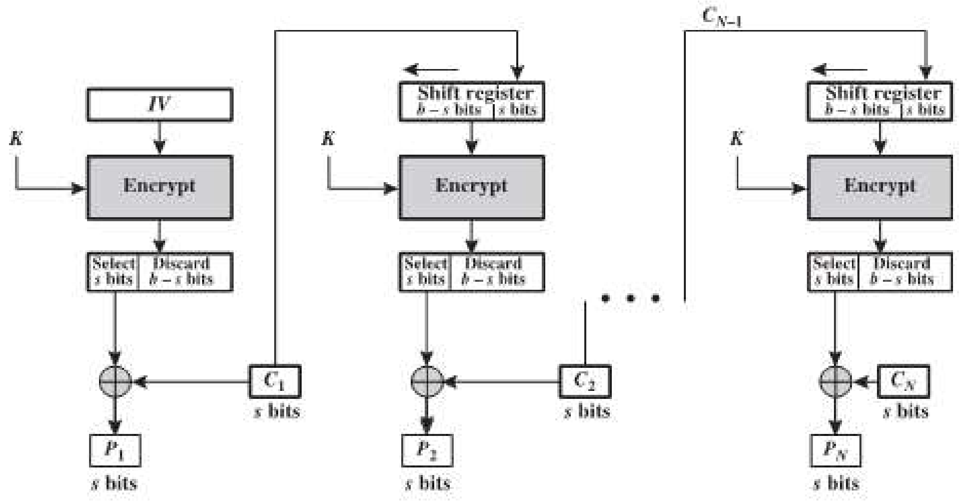
- The message is treated as a stream of bits which are added to the output of the block cipher
- Result is feed back for next stage (hence named)
- Standard allows any number of bit (1, 8 or 64 or whatever) to be feed back denoted CFB-1, CFB-8, CFB-64 etc
- In practice it is most efficient to consume all 64 bits, call this CFB-64
  - $C_i = P_i \oplus \text{Sig}(\text{En}(P_i , K_1))$
  - $C_0 = \text{IV}$

Cipher Feedback (CFB) (2/3)

- Encryption



- Decryption



Cipher Feedback (CFB) (3/3)

- Advantages and limitations of CFB
  - Appropriate when data inherently arrives in bits/bytes.
  - Most common stream mode.
  - Block cipher is use in encryption mode at both ends.
  - Errors propagate for several blocks after the error.

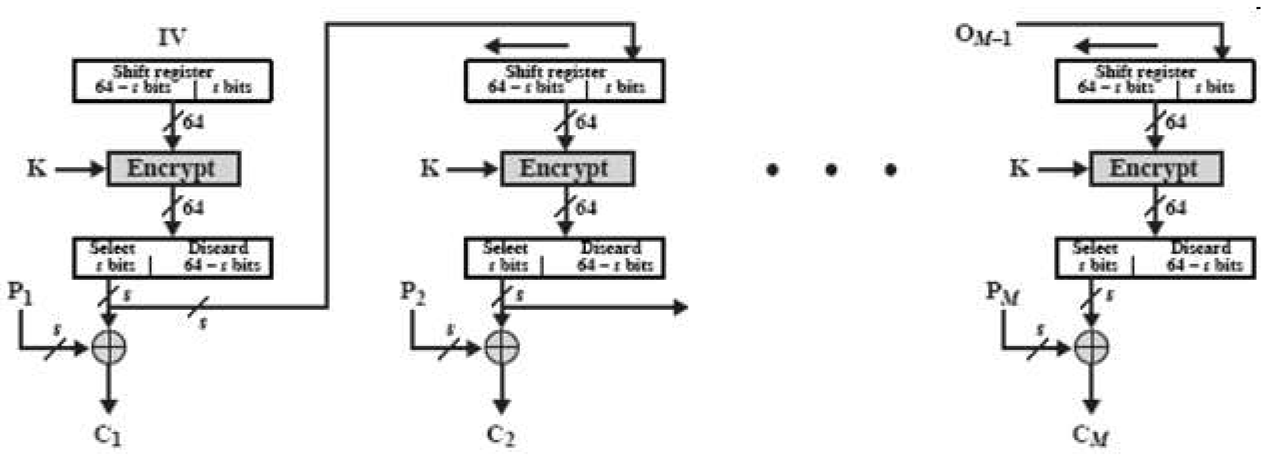
Output Feedback (OFB) (1/3)

- The message is treated as a stream of bits
- Output of block cipher is added to the message. and output if then feed back (hence named) thus feedback is independent of the message.

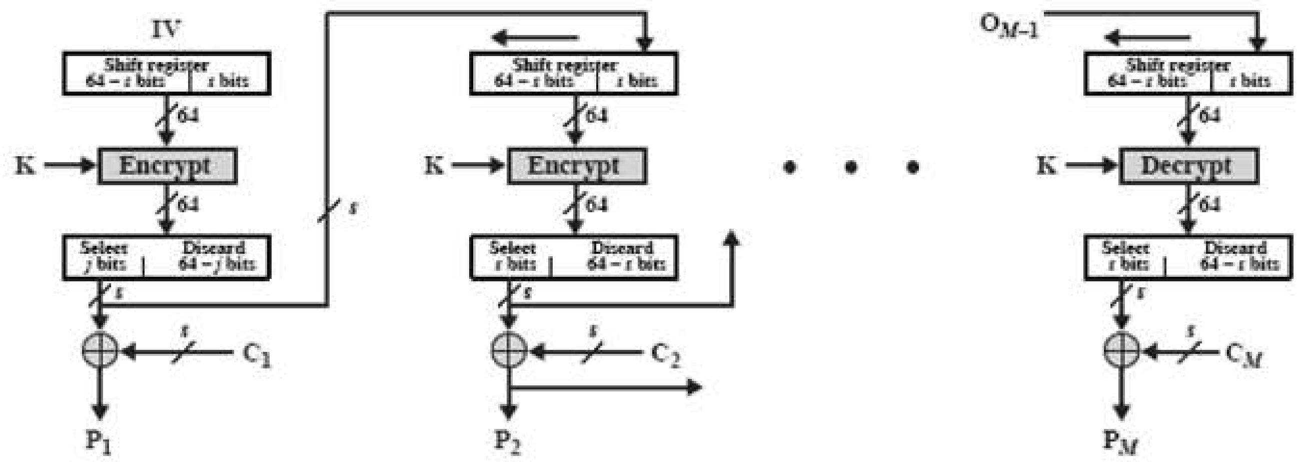
- Can be computed in advance
  - $C_i = P_i \oplus O_i$
  - $O_i = S_i\ (En(O_{i-1}, K_1))$
  - $O_0 = IV$

Output Feedback (OFB) (2/3)

- Encryption



- Decryption



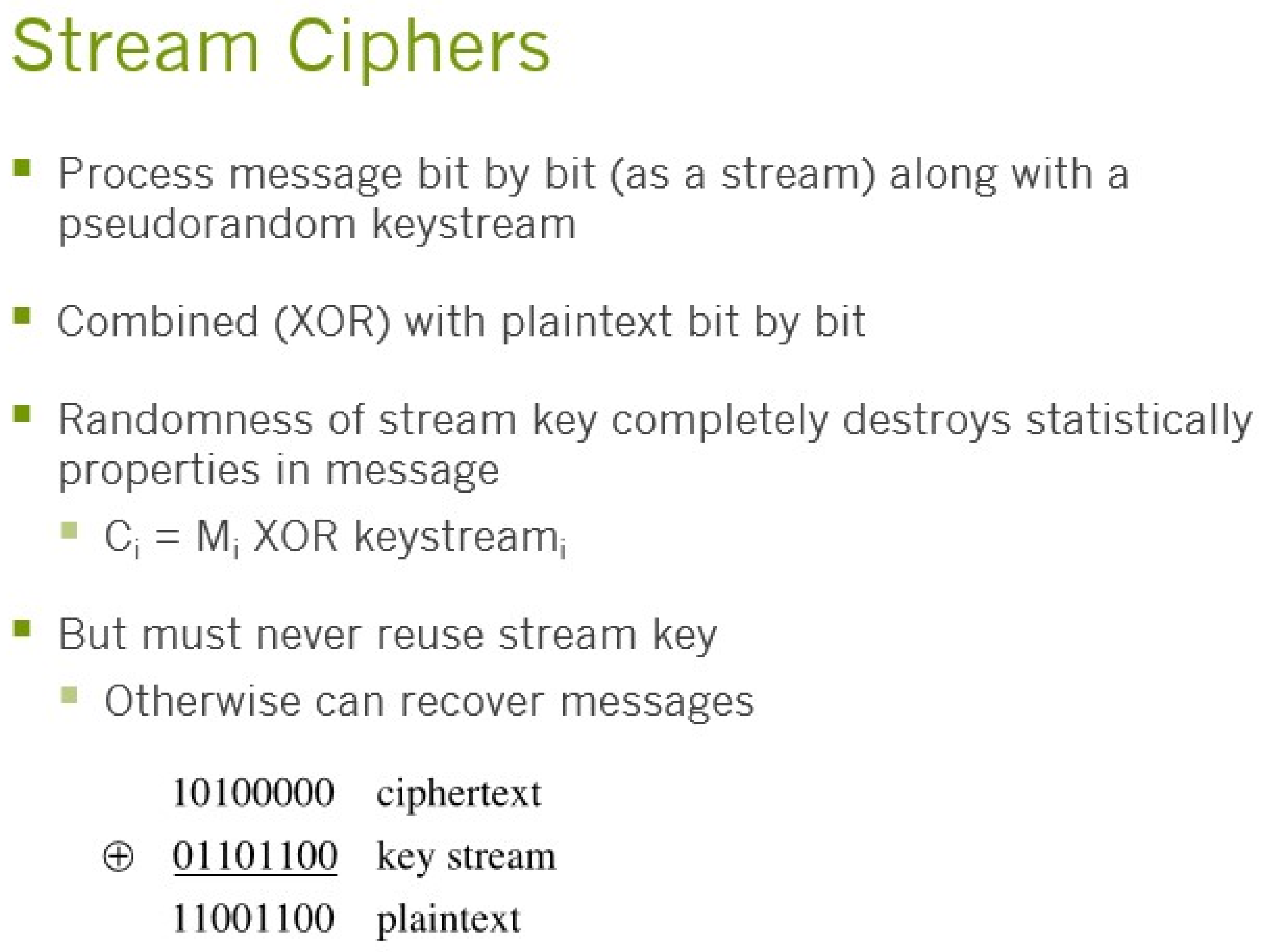
Output Feedback (OFB) (3/3)

- Advantages and limitations of OFB
  - Used where the error feedback is a problem.
  - No error propagation, or where the encryptions must be done before the message is available.
  - Similar to CFB but the feedback is from the output of the block cipher and is independent of the message.
  - never reuse the same sequence (key + IV), sender and receiver must remain in sync, and some recovery method is needed to ensure this occurrence
  - Although originally specified with m-bit feedback in the standards, subsequent research has shown that only 64-bit OFB should ever be used and this is the most efficient use anyway.

澳門理工大學
Universidade Politécnica de Macau
Macao Polytechnic University

Stream Ciphers

Stream Ciphers (1/7)



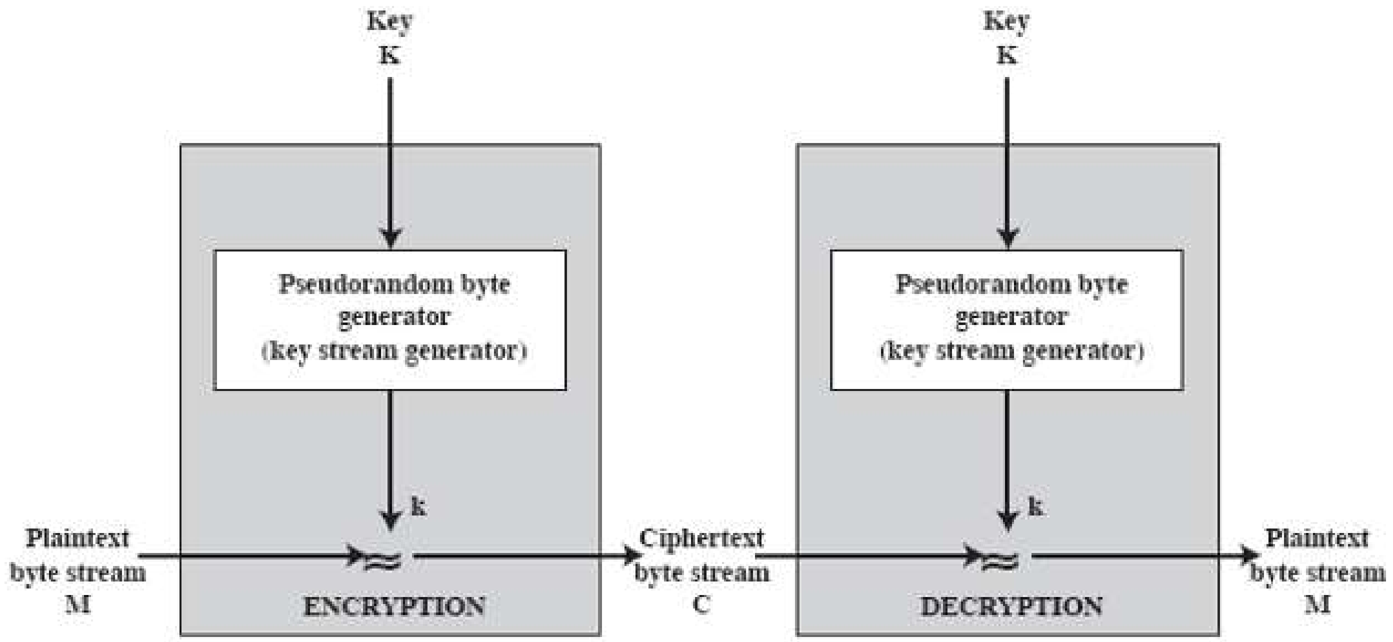
Stream Ciphers

- Process message bit by bit (as a stream) along with a pseudorandom keystream
- Combined (XOR) with plaintext bit by bit
- Randomness of stream key completely destroys statistically properties in message
  - $C_i = M_i$ XOR keystream$_i$
- But must never reuse stream key
  - Otherwise can recover messages

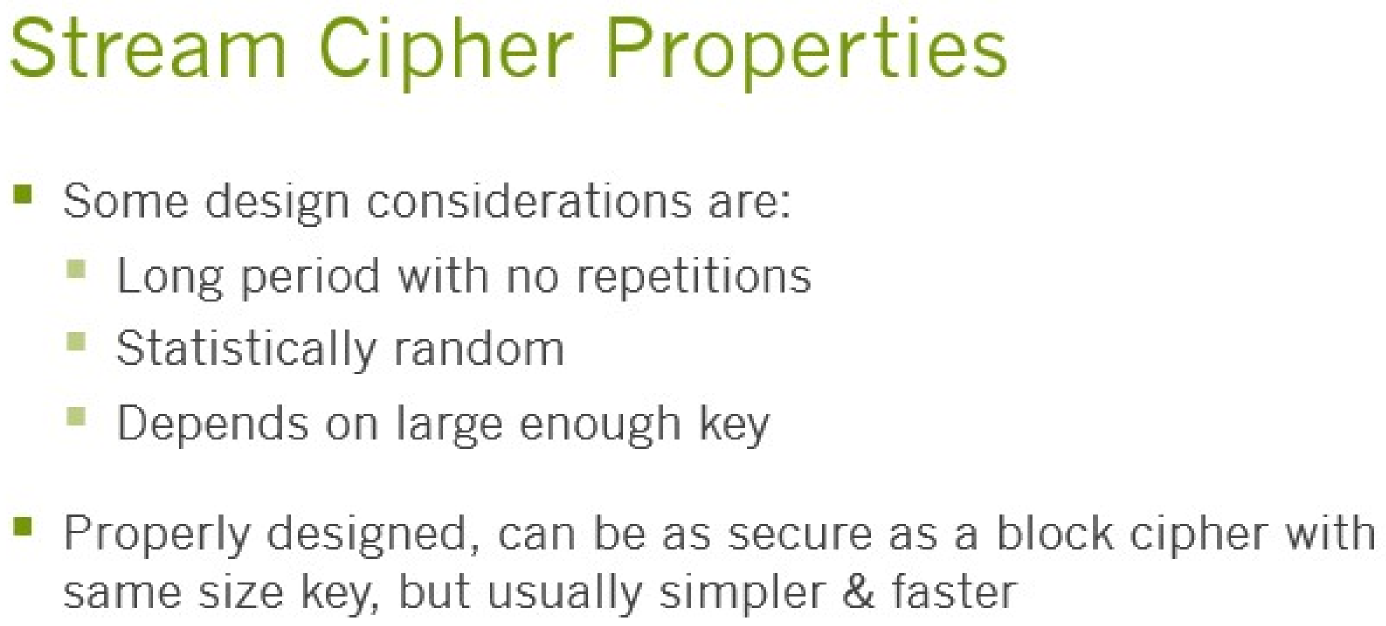
$$\begin{array}{rll} & 10100000 & \text{ciphertext} \\ \oplus & \underline{01101100} & \text{key stream} \\ & 11001100 & \text{plaintext} \end{array}$$

Stream Ciphers (2/7)

Stream Cipher Structure


Key
K
Pseudorandom byte generator (key stream generator)
k
Plaintext byte stream M
ENCRYPTION
Ciphertext byte stream C
Key
K
Pseudorandom byte generator (key stream generator)
k
DECRYPTION
Plaintext byte stream M

Stream Ciphers (3/7)



Stream Cipher Properties

- Some design considerations are:
  - Long period with no repetitions
  - Statistically random
  - Depends on large enough key
- Properly designed, can be as secure as a block cipher with same size key, but usually simpler & faster

Stream Ciphers (4/7)

- Stream ciphers and block ciphers are substitution ciphers.
- They use reversible mapping.

Reversible Mapping

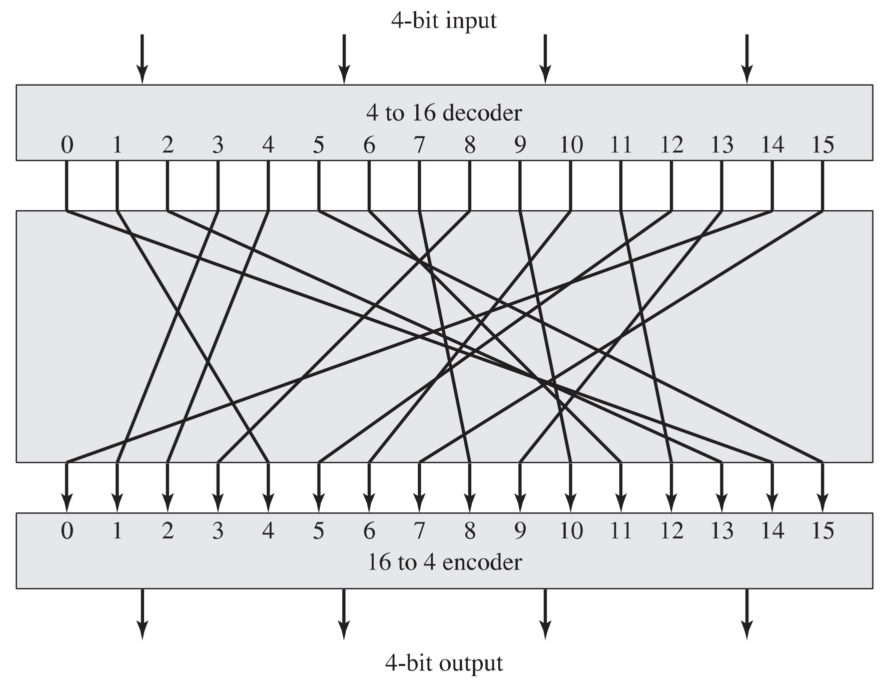
Plaintext	Ciphertext
00	11
01	10
10	00
11	01

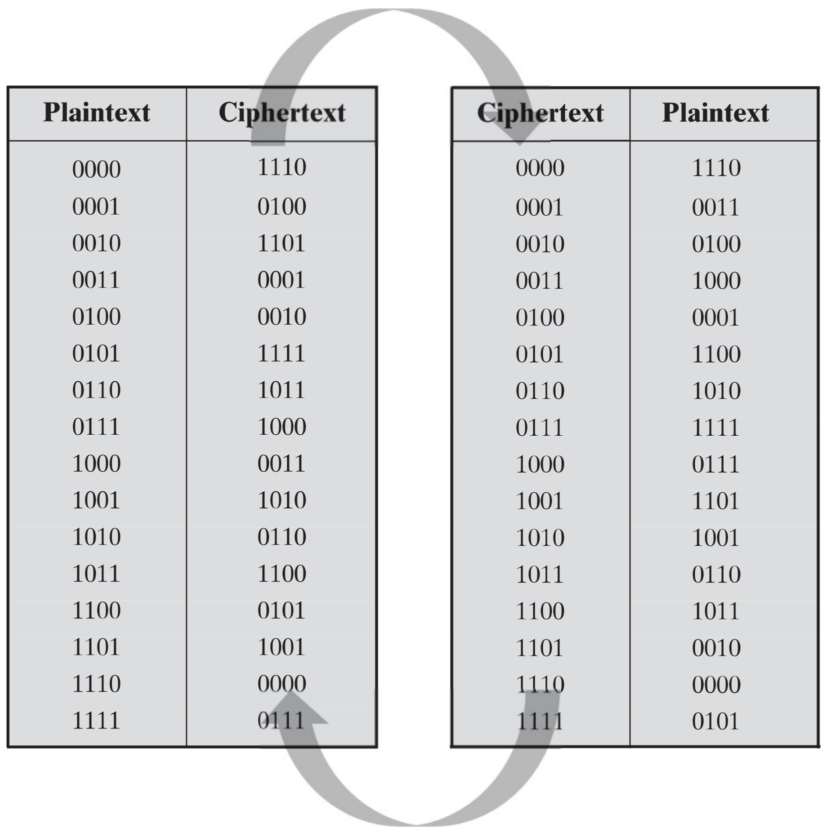
Irreversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	01
11	01

Stream Ciphers (5/7)

- Optimizing the Block size (*n* = 4)





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

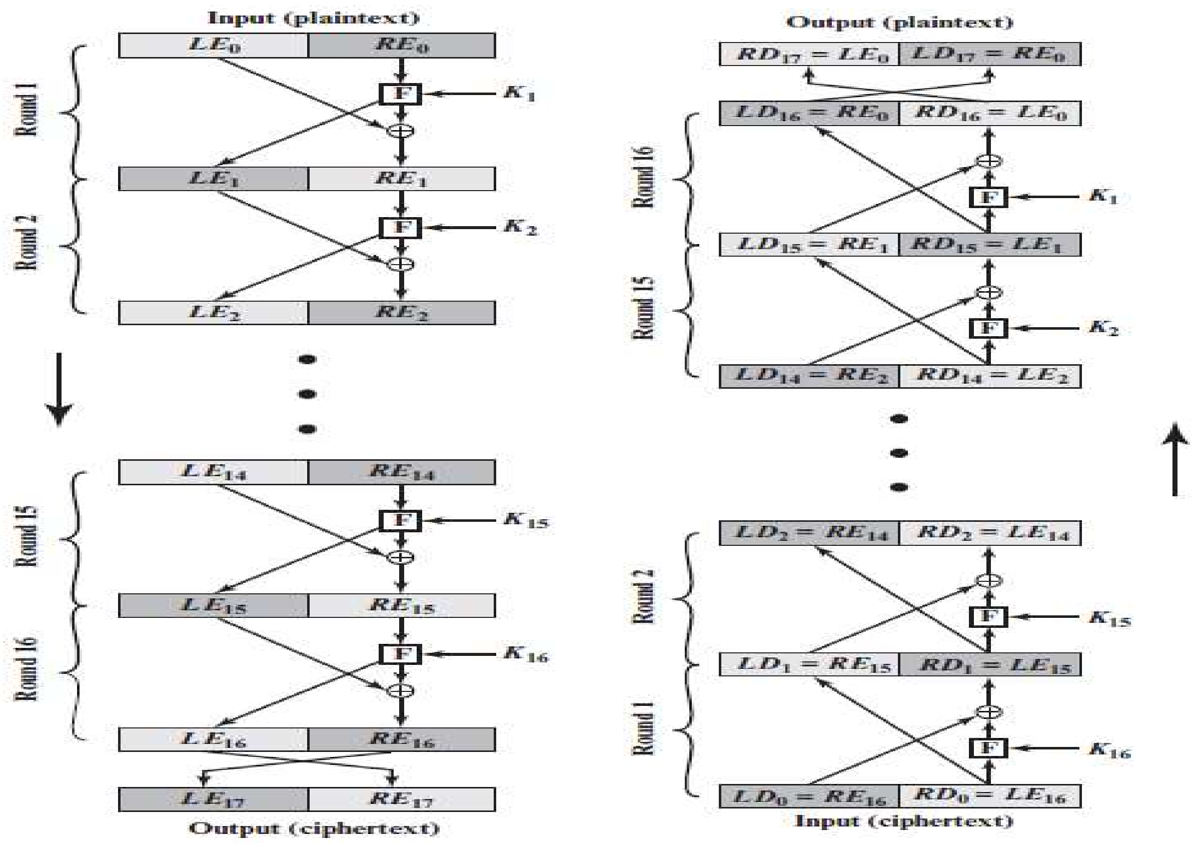
Ciphertext	Plaintext
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

Stream Ciphers (6/7)

- Feistel cipher
  - Symmetric structure used in the construction of block ciphers, named after Horst Feistel.
  - Feistel proposed that we can approximate the ideal block cipher by utilizing the concept of a product cipher, which is the execution of two or more simple ciphers in sequence.
  - Substitution:
    - 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. 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.

Stream Ciphers (7/7)

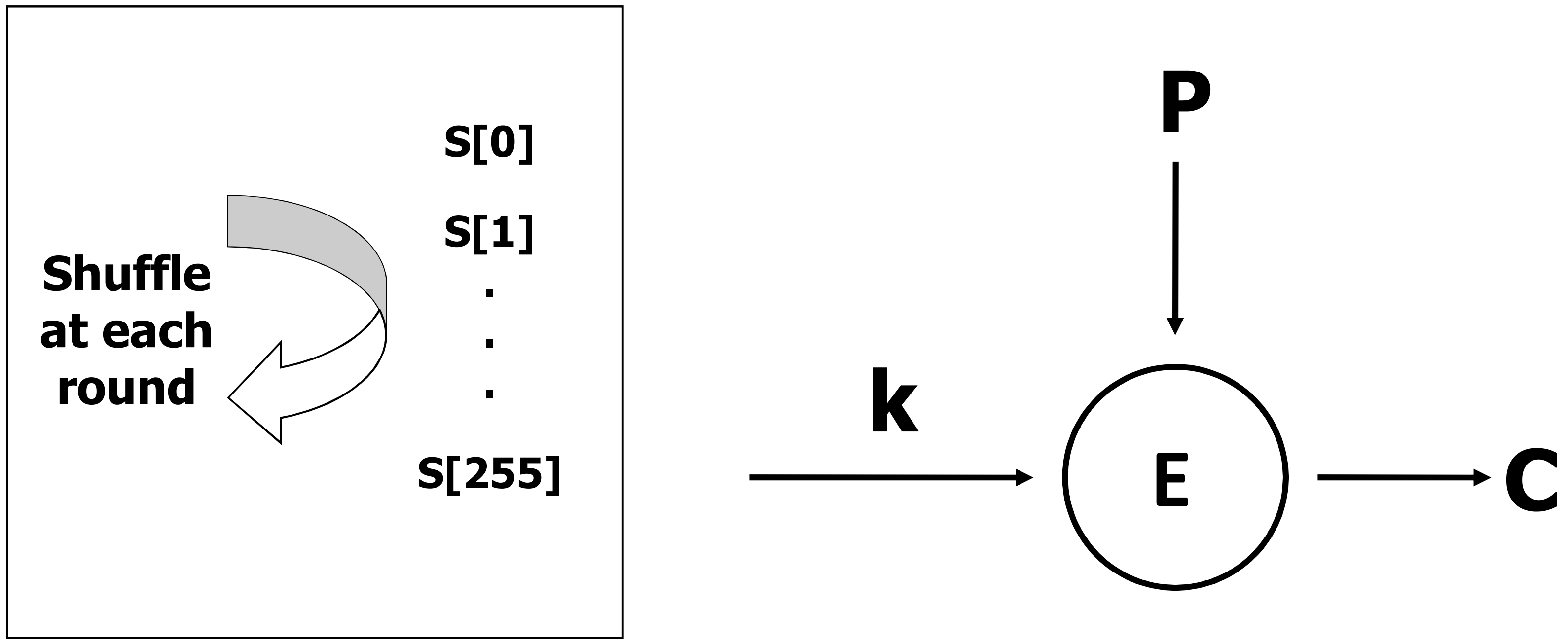
- Feistel Encryption & Decryption (16 round)



RC4 (1/6)

- A proprietary cipher owned by RSA DSI.
- Another Ron Rivest design, simple but effective.
- Variable key size, byte-oriented stream cipher.
- Widely used (web SSL/TLS, wireless WEP).
- Key forms random permutation of all 8-bit values.
- Uses that permutation to scramble input info processed a byte at a time

RC4 (2/6)



RC4 (3/6)

- RC4 Key Schedule
  - Starts with an array S of numbers: 0..255
  - Use key to well and truly shuffle
  - S forms internal state of the cipher
  - Given a key K of length keylen bytes

```
for i = 0 to 255 do
    S(i) = i
    T(i) = K(i mod keylen)
    j = 0
for i = 0 to 255 do
    j = (j + S(i) + T(i)) mod 256
    swap (S(i), S(j))
```

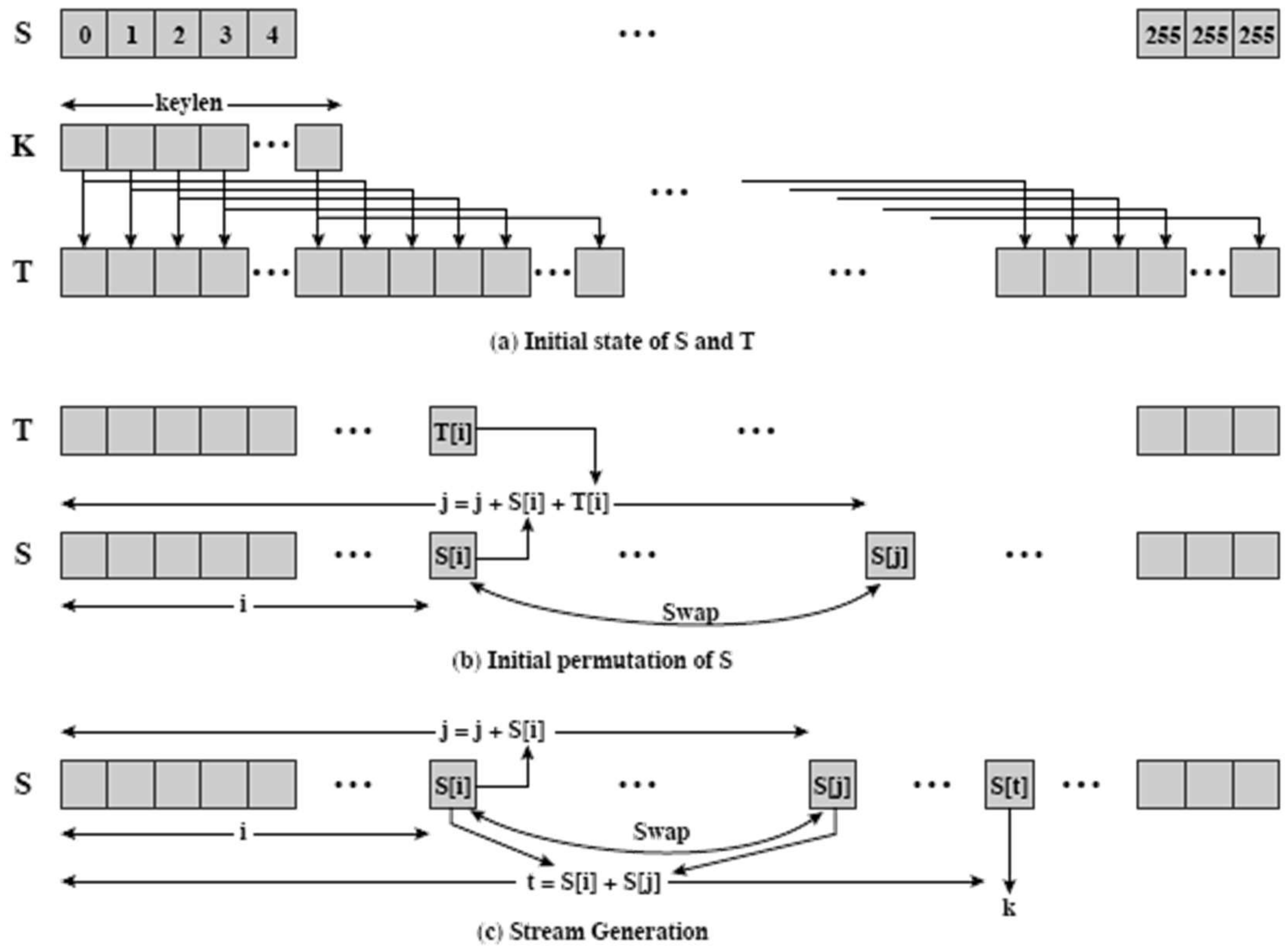
RC4 (4/6)

- RC4 Encryption / Decryption
  - Encryption continues shuffling array values
  - Sum of shuffled pair selects "stream key" value
  - XOR with next byte of message to en/decrypt

```
i = j = 0
for each message byte $M_i$
    i = (i + 1) mod 256
    j = (j + S(i)) mod 256
    swap(S(i), S(j))      //keep shuffling S()
    t = (S(i) + S(j)) mod 256
    $C_i = M_i \oplus S(t)$
```

RC4 (5/6)

- RC4 Operations



RC4 (6/6)

- RC4 Security
  - Claimed secure against known attacks.
    - Have some analyses, none practical
  - Result is very non-linear.
  - Since RC4 is a stream cipher, must never reuse a key.
  - Have a concern with WEP, but due to key handling rather than RC4 itself.

Exercises

- Try manually to encrypt your name using S-DES encryption algorithm.
- Generate the K1 and K2 from the 10-bit key 10011 00111 in SDES
- Do the decryption for the ciphertext above using the same key
- What are the design principles and parameters we need to consider when we design a secure block cipher?
- Explain and illustrate how you use a SDES cipher on modes ECB, CBC, CFB, and OFB. What are their characteristics?

澳門理工學院
INSTITUTO POLITÉCNICO DE MACAU
MACAO POLYTECHNIC INSTITUTE