Software Security and Dependability

ENGR5560G

Lecture 03

Block Ciphers and DES

Dr. Khalid A. Hafeez
Spring, 25

Lecture Outline

- Stream Cipher
- Block Cipher:
- Feistel Cipher
- Data Encryption Standard (DES)

Dimensions to Classify Cryptographic System

The type of operations used for transforming plaintext to ciphertext

Substitution

Each element in the plaintext is mapped into another element

Transposition

Elements in the plaintext are rearranged

The number of keys used

Symmetric, single-key, secret-key, conventional encryption

Asymmetric, two-key, or public-key encryption

The way in which the plaintext is processed

Block cipher

Process the input one block of elements at a time

Produce an output block for each input block

Stream cipher

Process the input elements continuously

Produce output one element at a time, as it goes along

The Way in Which the Plaintext is Processed

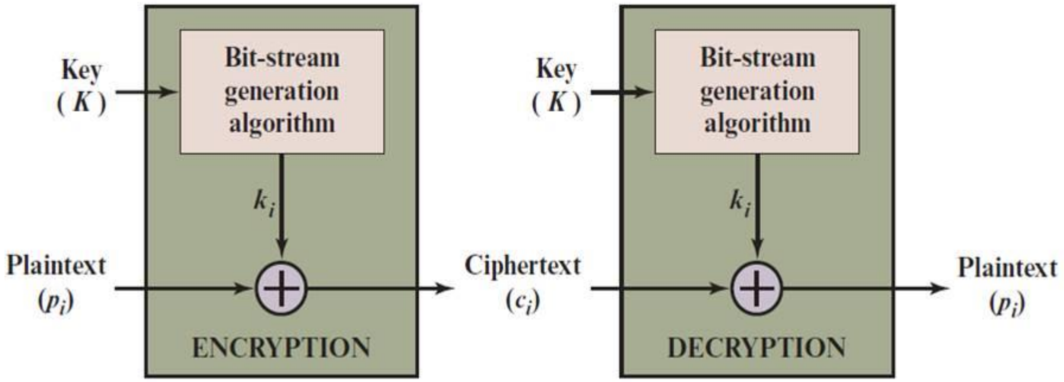
Stream Cipher:

- Encrypts a digital data stream one bit **or** one byte at a time

  Examples:

  Autokeyed Vigenère cipher

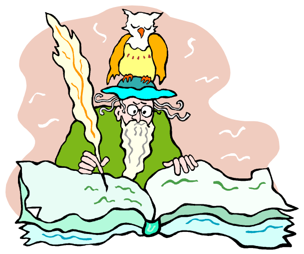
  Vernam cipher



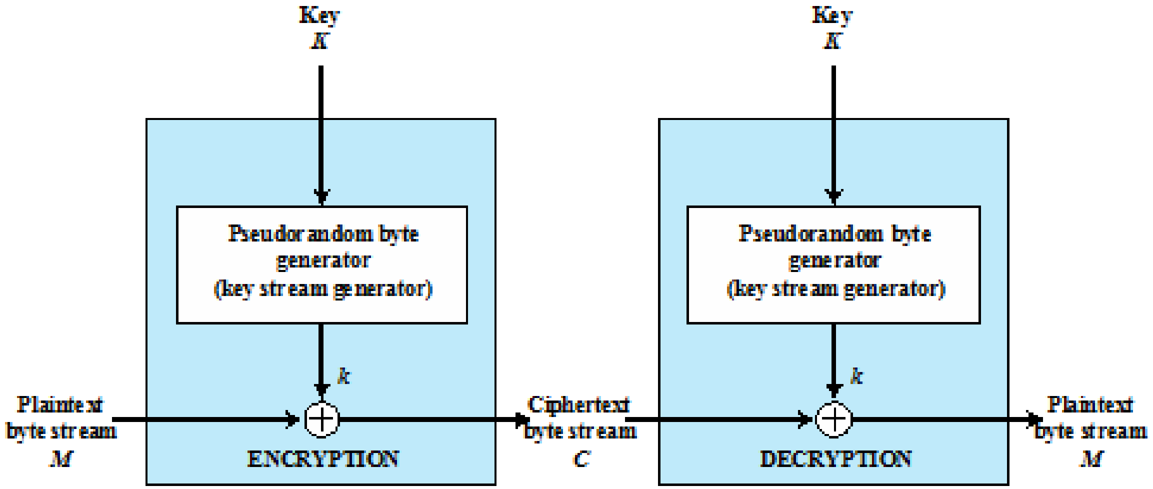
Keystream:

- Ideal case ← keystream is as long as the plaintext bit stream
- If the keystream is random ← cipher is unbreakable
- Keystream must be provided to both users in advance via some independent and secure channel
- 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 bit stream
  - The two users need only share the generating key, and each can produce the keystream

Stream Cyphers and RC4 (Rivest Cipher 4)



- Stream Cyphers
  - Processes the input elements continuously
  - Produces output one element at a time
  - Primary advantage is that they are almost always faster and use far less code
  - Encrypts plaintext one byte at a time
  - Pseudorandom stream is one that is unpredictable without knowledge of the input key
- Stream Cypher Structure



(b) Stream encryption

Figure 2.2 Types of Symmetric Encryption

Stream Cyphers and RC4 (Rivest Cipher 4)

- The RC4 Algorithm (stream cipher): used in SSL/TLS and WPA

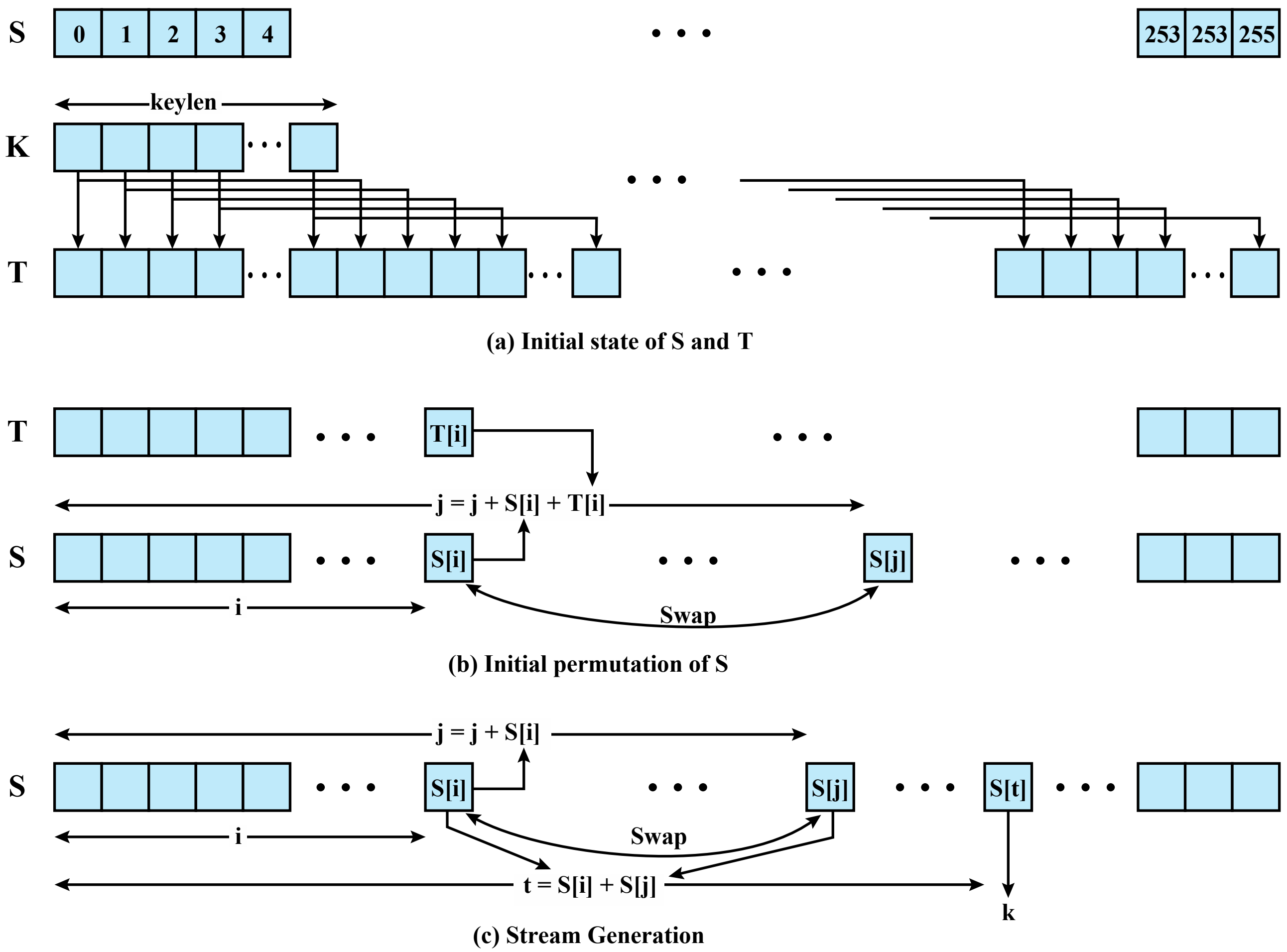


Figure 20.6 RC4

```
/* Initialization */
for i = 0 to 255 do
 S[i] =i;
T[i] = K[i mod keylen];
```

```
/* Initial Permutation of S */
j = 0;
for i = 0 to 255 do
 j = (j + S[i] + T[i]) mod 256;
Swap (S[i], S[j]);
```

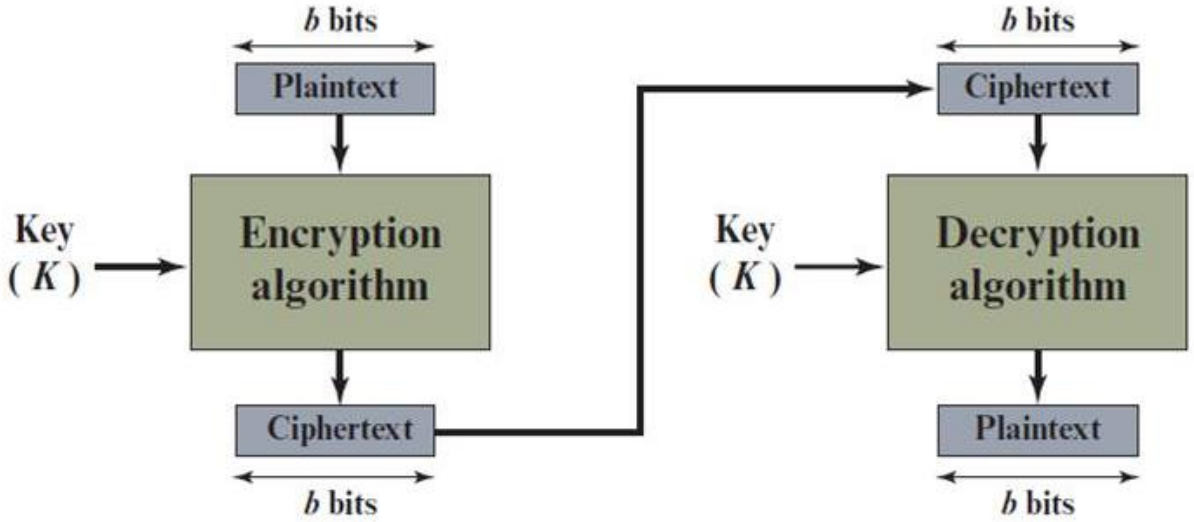
```
/* Stream Generation */
i, j = 0;
while (true)
i = (i + 1) mod 256;
j = (j + S[i]) mod 256;
Swap (S[i], S[j]);
t = (S[i] + S[j]) mod 256;
k = S[t];
```

To encrypt XOR the value k with the next byte of the plain text
To decrypt, XOR the value k with the next byte of ciphertext

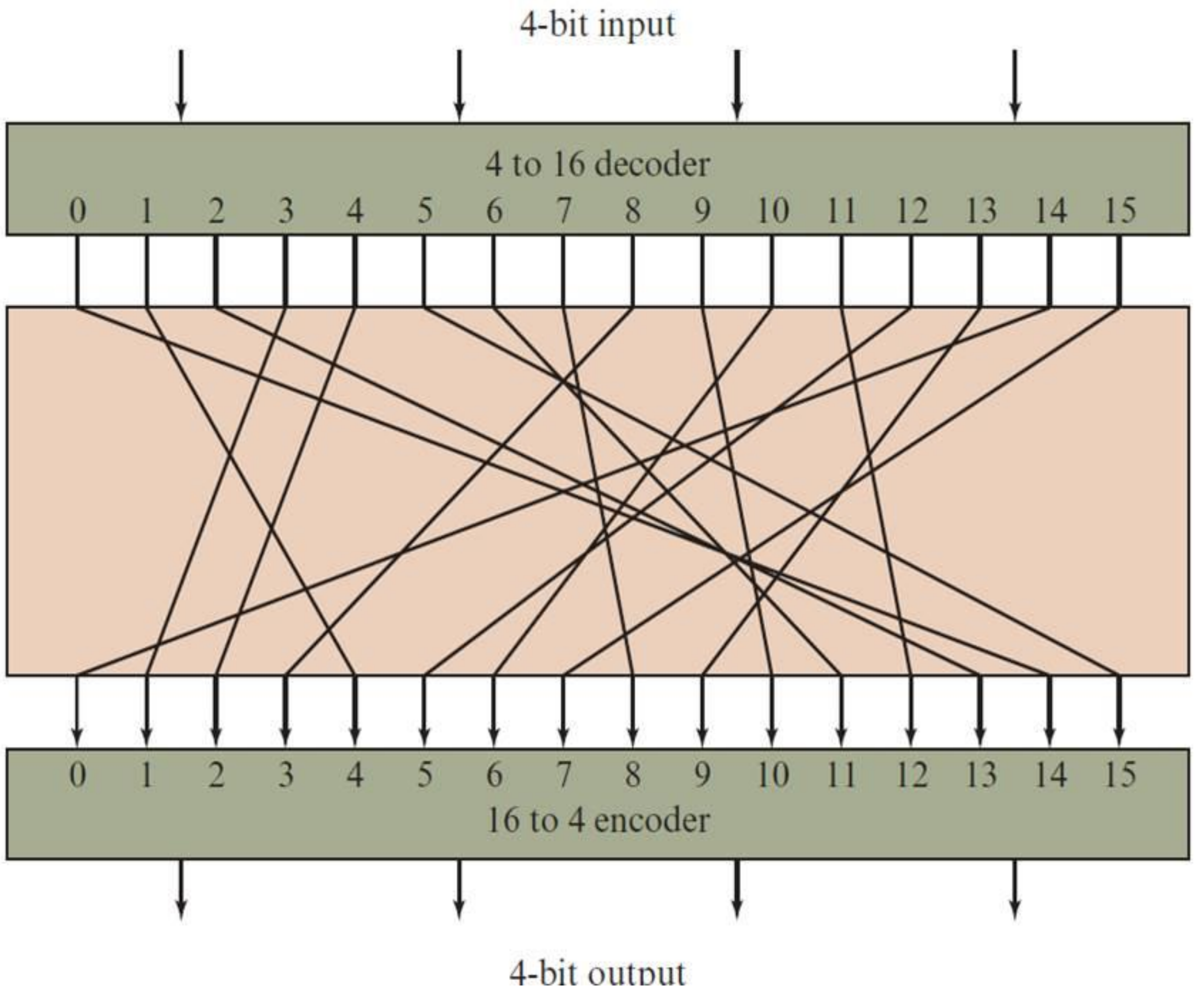
Block Cipher

Block Cipher:

- Plaintext convert to block of text
- A block of plaintext is treated as a whole and used to produce a ciphertext block
- Typically, a block size of 64 or 128 bits is used
- Two users share a symmetric encryption key
- Most network-based symmetric cryptographic applications make use of block ciphers



General 4-bit by 4-bit Block Substitution - Example



Encryption

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

Decryption

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

- Create cipher text by alternating substitutions and permutations

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

Permutation
- Order in which the elements appear in the sequence is changed

- It is a practical application of a proposal by Claude Shannon to develop a product cipher that alternates confusion and diffusion functions

- It is the structure used by many significant symmetric block ciphers currently in use

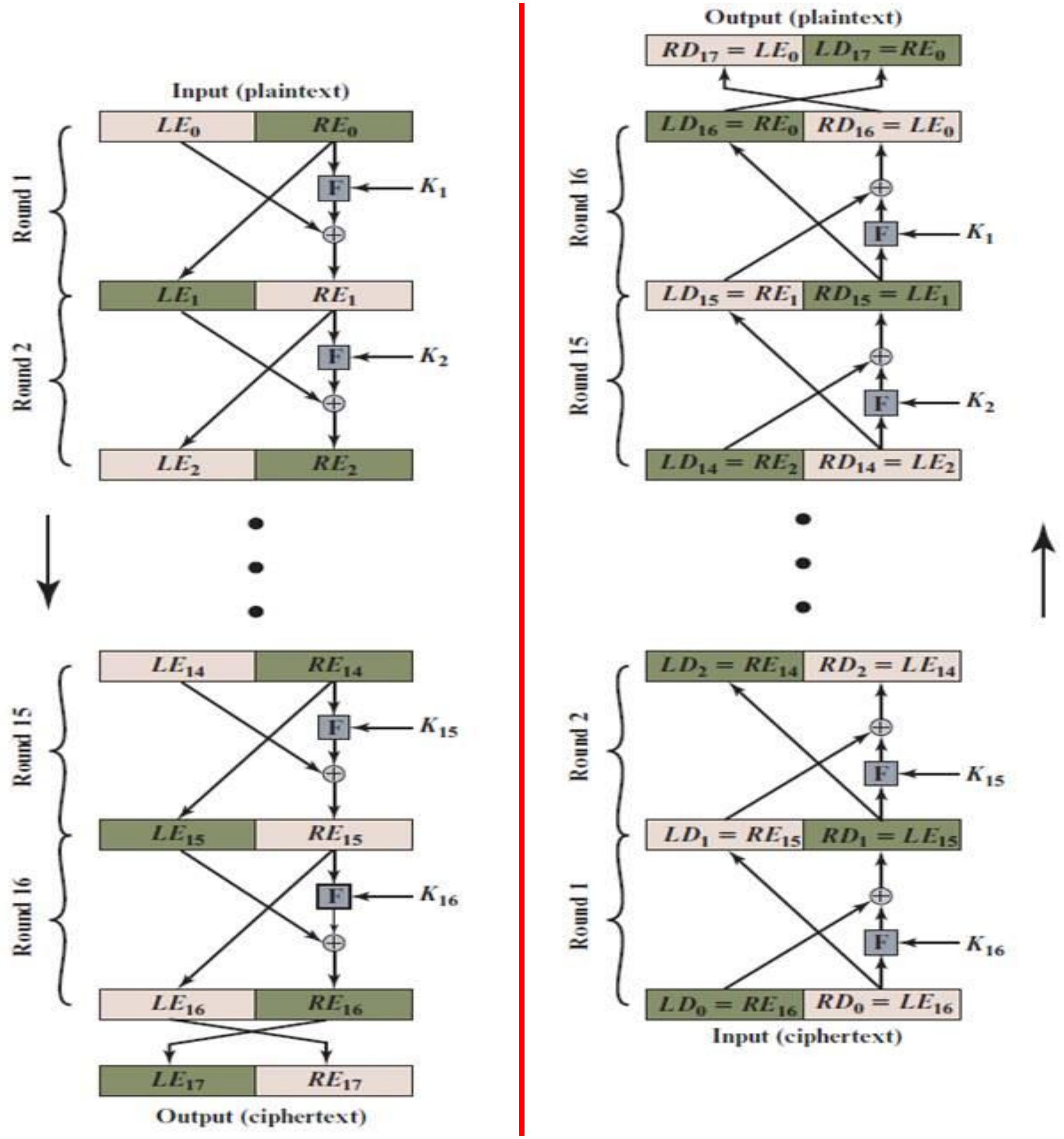
Feistel Cipher Design Features

- **Block size**
  - Larger block sizes mean greater security but reduced encryption/decryption speed for a given algorithm
- **Key size**
  - Larger key size means greater security but may decrease encryption/decryption speeds
- **Number of rounds**
  - The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security
- **Subkey generation algorithm**
  - Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis
- **Round function F**
  - Greater complexity generally means greater resistance to cryptanalysis
- **Fast software encryption/decryption**
  - In many cases, encrypting is embedded in applications or utility functions in such a way as to preclude a hardware implementation. Accordingly, the speed of execution of the algorithm becomes a concern.
- **Ease of analysis**
  - If the algorithm can be concisely and clearly explained, it is easier to analyze that algorithm for cryptanalytic vulnerabilities and therefore develop a higher level of assurance as to its strength

Feistel Encryption and Decryption (16 rounds)

- **Encryption:**
  - The plaintext block is divided into two halves, LE_0 and RE_0
  - Round i
    - $LE_i = RE_{i-1}$
    - $RE_i = LE_{i-1}$ XOR **F**(RE_{i-1}, K_i)

- **Decryption:**
  - $LD_0 = RE_n$
  - $RD_0 = LE_n$
  - Round i
    - $LD_i = RD_{i-1} = RE_{n-i}$
    - $RD_i = LD_{i-1}$ XOR **F**$(RD_{i-1}, k_i) = LE_{n-i}$



Data Encryption Standard (DES)

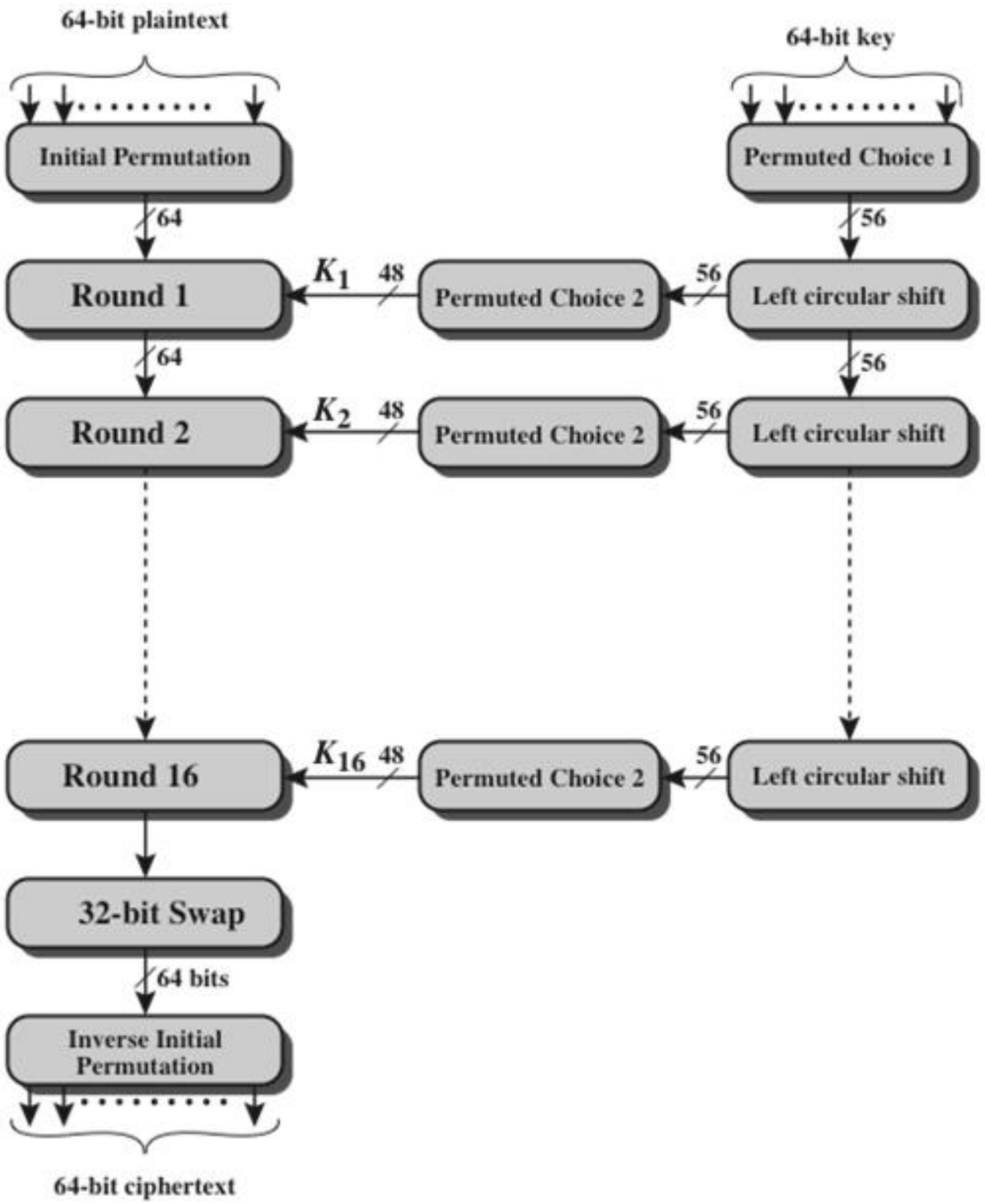
- Early 1970s ← demand for encryption for commercial applications
- In 1974 ← US National Bureau of Standards (NBS) received the most promising algorithm from a team of cryptographers working at IBM
- Issued in 1977 by the National Bureau of Standards (now NIST)
- Was the most widely used encryption scheme until the introduction of the Advanced Encryption Standard (AES) in 2001
- Algorithm itself is referred to as the Data Encryption Algorithm (**DEA**)
    - Data are encrypted in 64-bit blocks using a 56-bit key
    - The algorithm transforms 64-bit input in a series of steps into a 64-bit output
    - The same steps, with the same key, are used for Decryption
        - Decryption uses the same algorithm as encryption, except that the application of the subkeys is reversed.
        - Additionally, the initial and final permutations are reversed.

Internal Structure of DES

1. Initial and Final Permutation

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

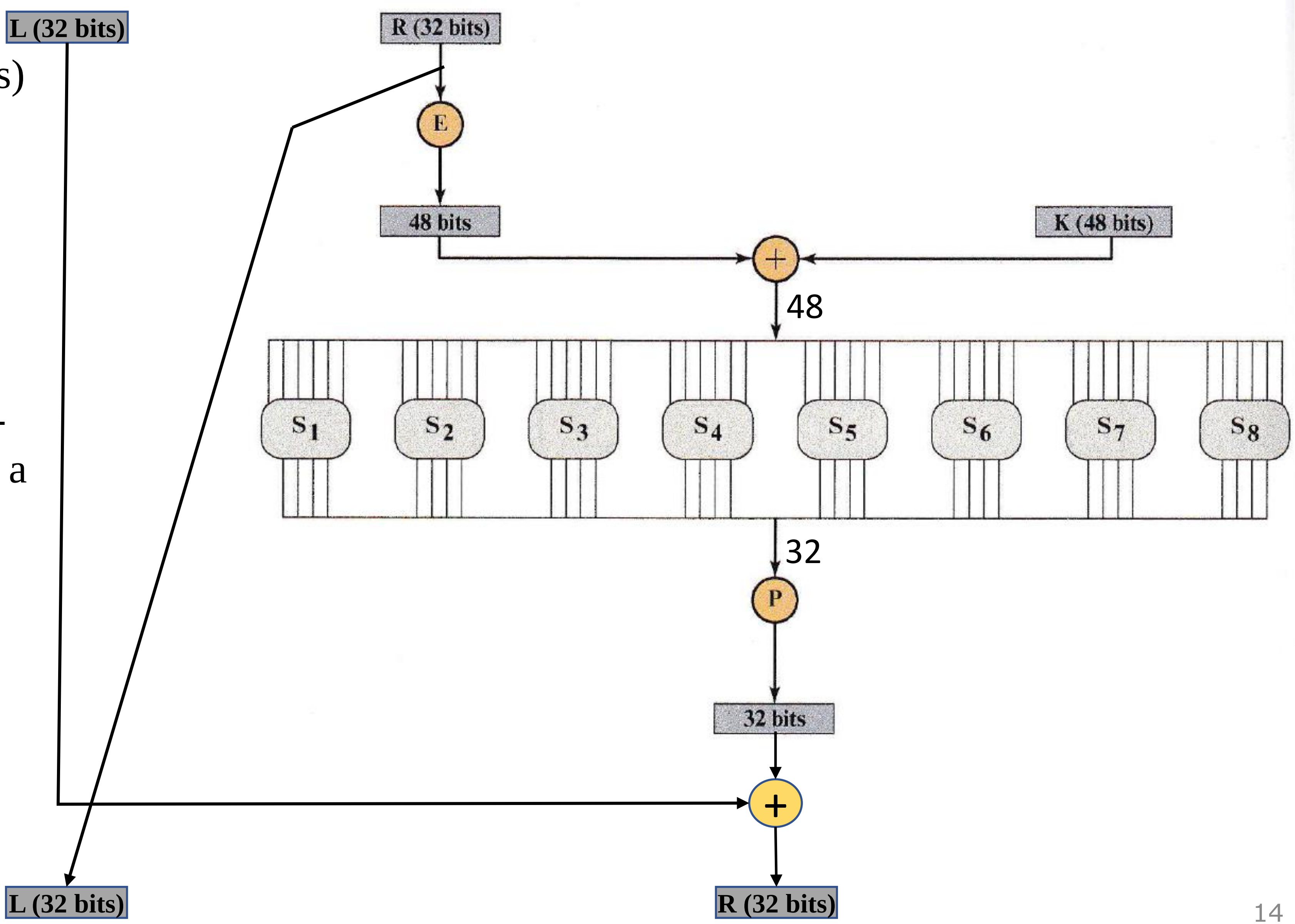
IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25



Internal Structure of DES

2. Round i:

- Takes the right half R_{i-1} (32 bits) of the output of the previous round
- Expand it to 48 bits
- XORed with the current round key k_i (48 bits)
- Eight 6-bit blocks are fed into eight different S-boxes, each S-box is a lookup table that maps a 6-bit input to a 4-bit output.
- The 32-bit output is permuted bitwise according to the P permutation
- XORed with the L_{i-1} (32 bits)

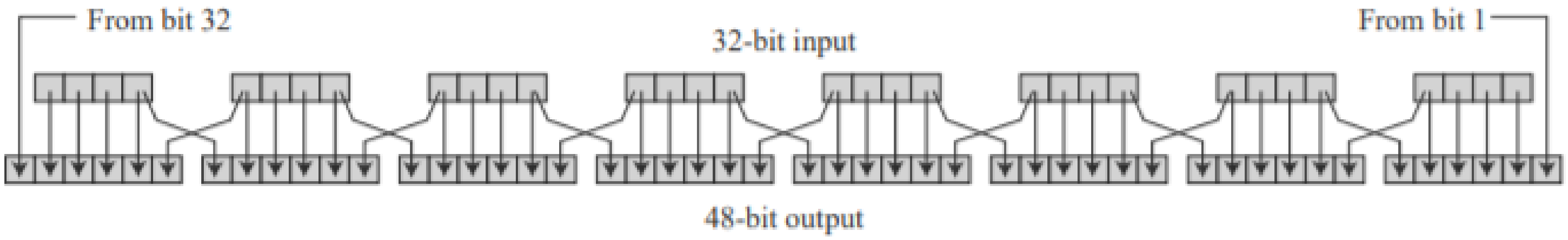


Internal Structure of DES

2. Round i:

- Takes the right half R_{i-1} (32 bits) of the output of the previous round
- Expand it to 48 bits

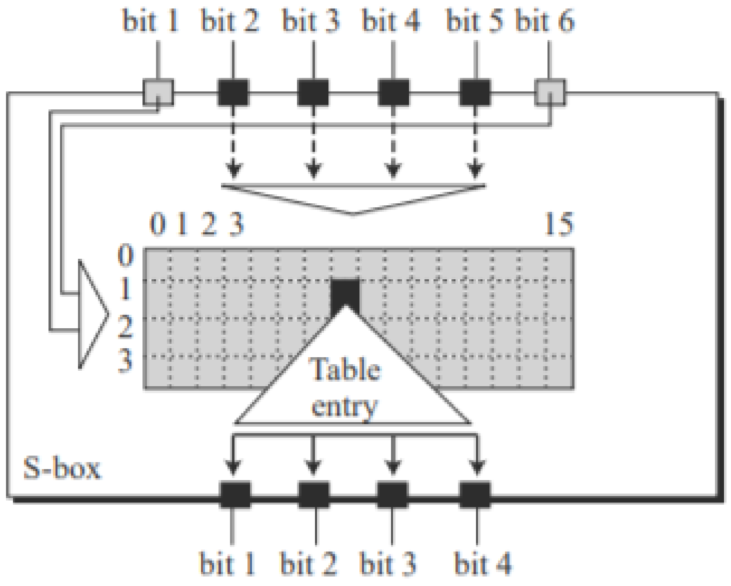
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1



Internal Structure of DES

2. Round i:

- Takes the right half R_{i-1} (32 bits) of the output of the previous round
- Expand it to 48 bits
- XORed with the current round key k_i (48 bits)
- Eight 6-bit blocks are fed into eight different S-boxes, each S-box is a lookup table that maps a 6-bit input to a 4-bit output. (**confusion**)



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

Internal Structure of DES

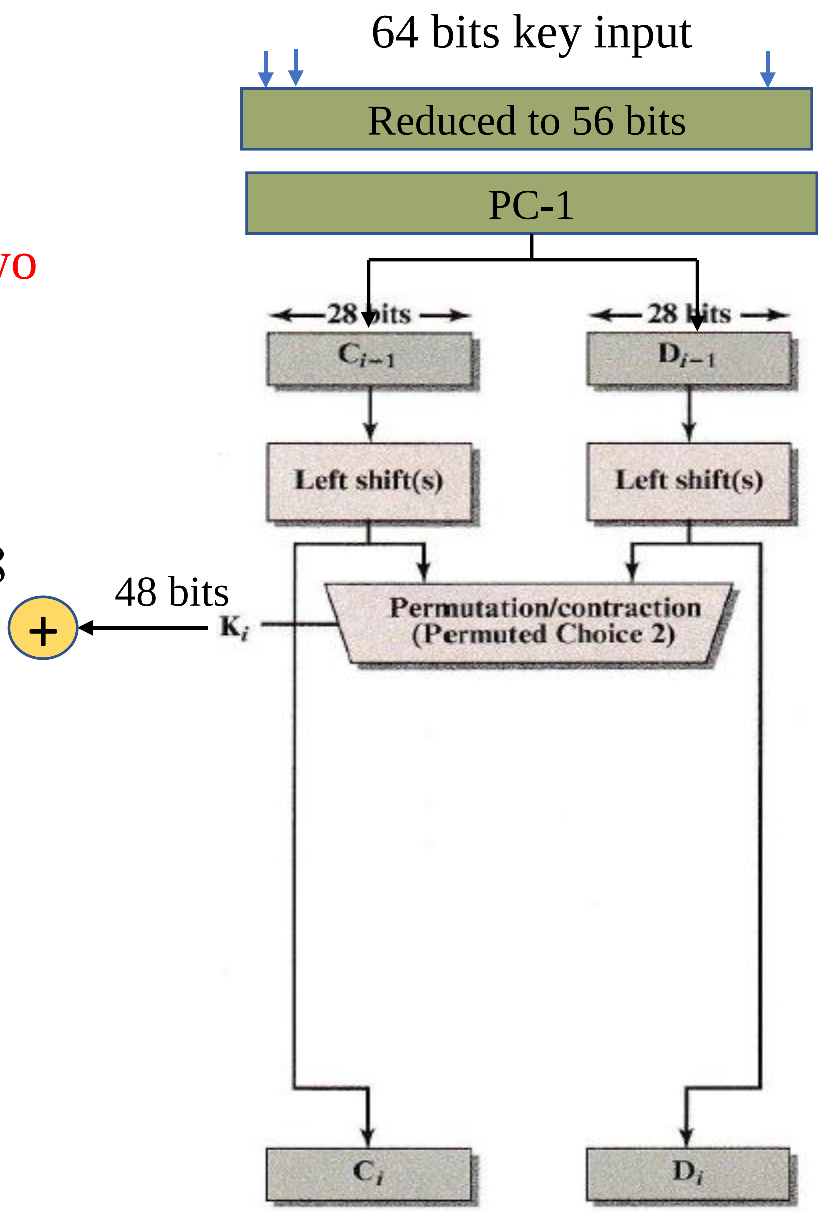
2. Round i:

- Takes the right half R_{i-1} (32 bits) of the output of the previous round
- Expand it to 48 bits
- XORed with the current round key k_i (48 bits)
- Eight 6-bit blocks are fed into eight different S-boxes, each S-box is a lookup table that maps a 6-bit input to a 4-bit output.
- The 32-bit output is permuted bitwise according to the P permutation

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

Key Schedule

1. 64 bit input → 56 bit key
2. Permutation (permuted choice 1: PC-1)
3. Resulting 56-bit key is split into two halves C_0 and D_0
4. The two 28-bit halves are cyclically left shifted (rotate) by one or two bit positions depending on the round i
    1. Rounds i = 1,2,9, 16, rotate left by one bit.
    2. Rounds i ≠ 1,2,9, 16, rotate left by two bits.
5. The two halves are permuted bitwise again with PC−2 and create 48 bits key k_i



Key Schedule

64 to 56

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

64 bit

PC-1

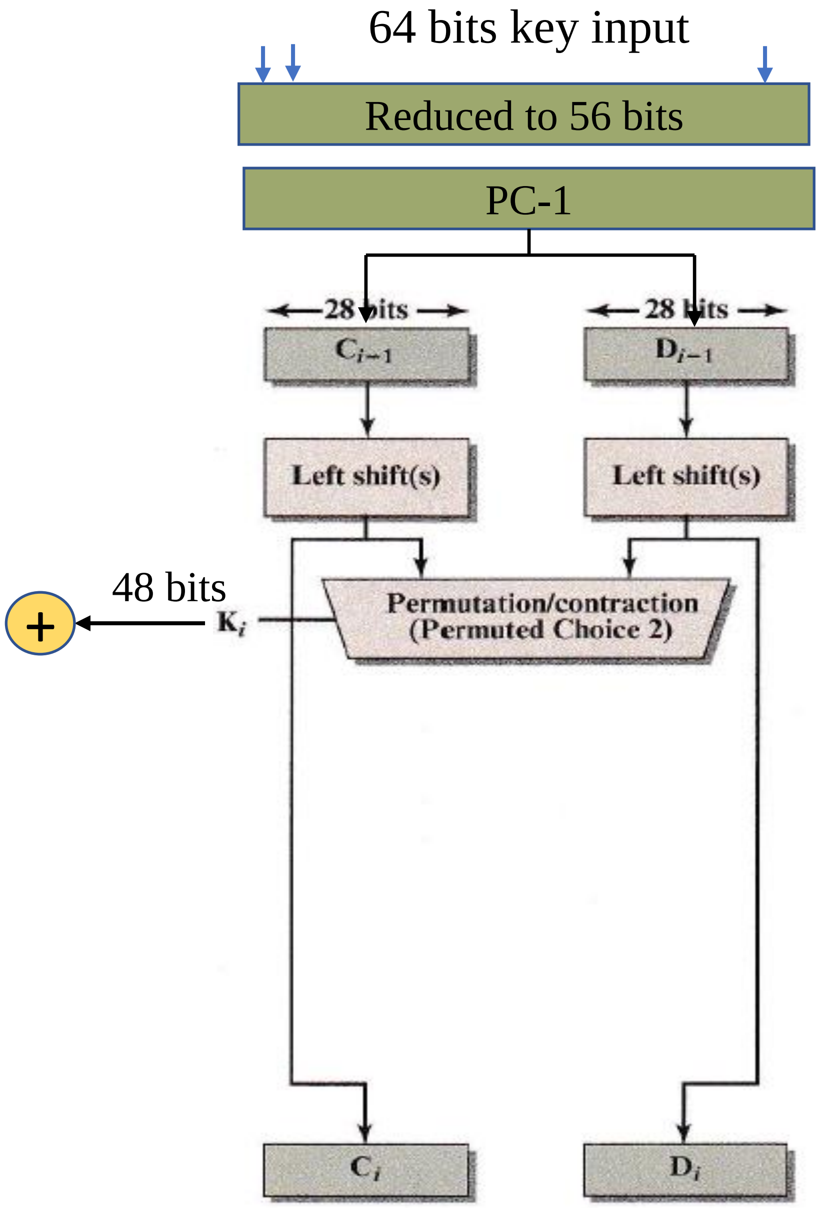
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

56 bit

PC-2

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

48 bit



DES Example

Plain text: 02468aceeca86420
Key: 0f1571c947d9e859
Cipher Text: da02ce3a89ecac3b

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

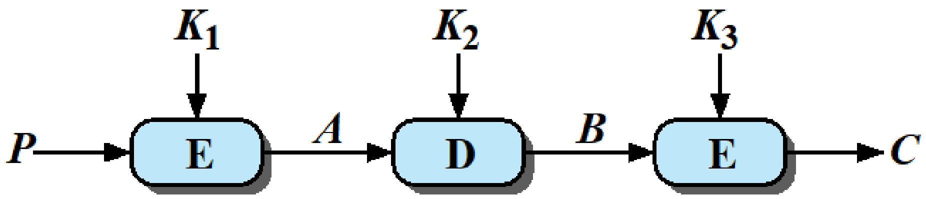
Triple DES (3DES)

- First standardized for use in financial applications in ANSI standard X9.17 in 1985
- Repeats basic DES algorithm three times using either 2 or 3 unique keys
- 3DES uses three keys and three executions of the DES algorithm.
- The function follows an encrypt-decrypt-encrypt (EDE) sequence:

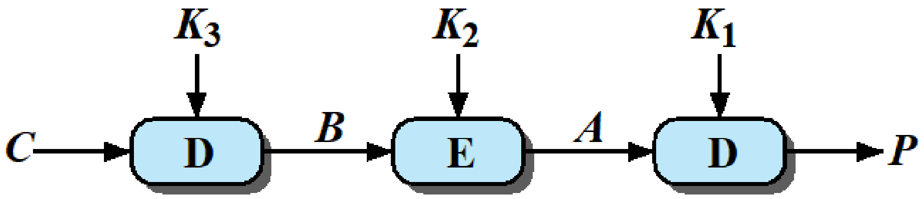
$$C = E(K3, D(K2, E(K1, P)))$$

where: C = ciphertext; P = plaintext; E[K, X] = encryption of X using key K, and D[K, Y] = decryption of Y using key K.

- Decryption is simply the same operation with the keys reversed: P = D(K1, E(K2, D(K3, C)))
- Attractions:
  - 168-bit key length overcomes the vulnerability to brute-force attack of DES
  - Underlying encryption algorithm is the same as in DES
- Drawbacks:
  - Algorithm is sluggish in software
  - Uses a 64-bit block size. A larger block size is more desirable.



(a) Encryption



(b) Decryption

Strength of DES

Brute-force attack:

- On average, half the key space must be searched
- Machine performing one DES encryption per microsecond would take more than a thousand years to break the cipher

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	$26! = 4 \times 10^{26}$	2×10^{26} ns = 6.3×10^{9} years	6.3×10^{6} years

Block Cipher Design Principles

1. Number of Rounds
2. Design of Function F
   - Nonlinear
   - Avalanche Property
     - a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext
3. Key Schedule Algorithm

Change in Plaintext

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

Block Cipher Design Principles

Change in Key

Key 1: 0f1571c947d9e859
Key 2: 1f1571c947d9e859

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

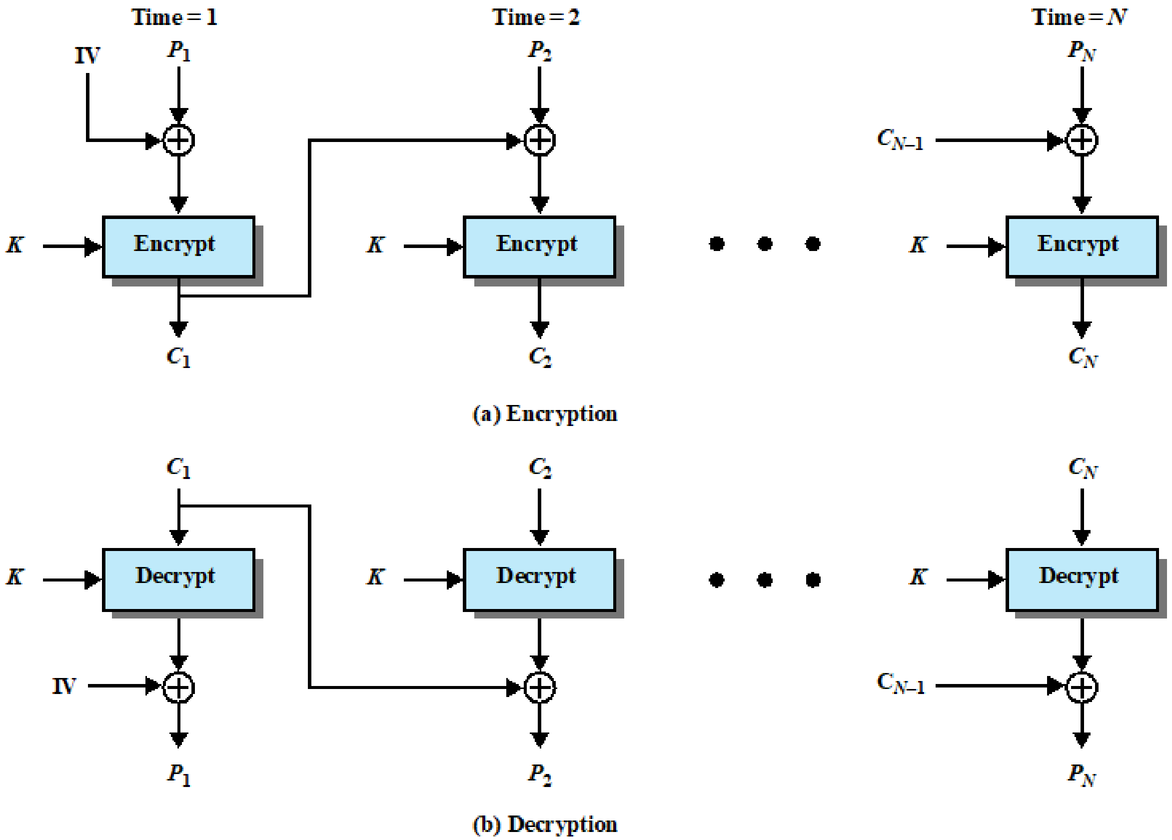
Block Cipher Modes of Operation

Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of 64 plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	•General-purpose block-oriented transmission •Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission •Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission •Useful for high-speed requirements

Block Cipher Modes of Operation

- Cipher block chaining (CBC)

$$C_j = \mathrm{E}(K, [C_{j-1} \oplus P_j])$$



$$D(K,C_j) = D(K,\mathrm{E}(K, [C_{j-1} \oplus P_j])) = C_{j-1} \oplus P_j$$

$$C_{j-1} \oplus D(K,C_j) = C_{j-1} \oplus C_{j-1} \oplus P_j = P_j$$

Block Cipher Modes of Operation

- Counter Mode (CTR)
  - The counter is initialized to some value and then incremented by 1 for each subsequent block (modulo 2^b, where b is the block size)
    - For encryption, the counter is encrypted and then XORed with the plaintext block
    - For decryption, the same sequence of counter values is used, with each encrypted counter XORed with a ciphertext block to recover the corresponding plaintext block.

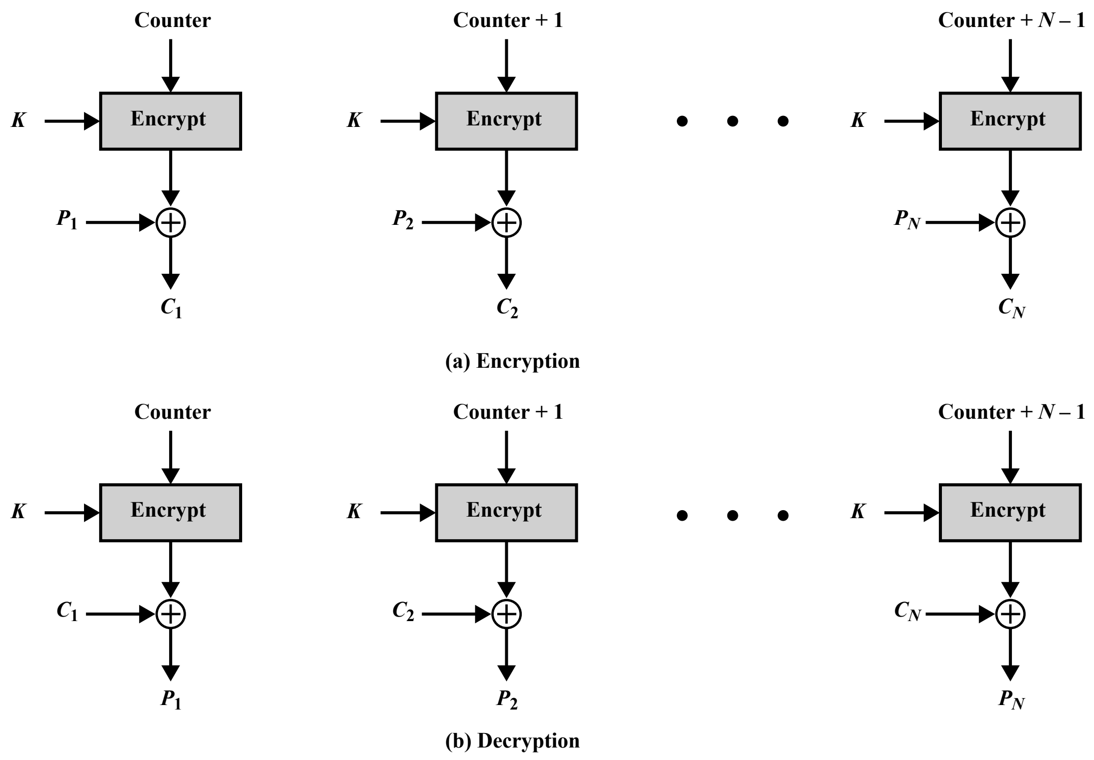


Figure 20.9 Counter (CTR) Mode

Message Authentication and Hash Functions

- Encryption protects against passive attack (eavesdropping).
- Message or data authentication is to protect against active attack (falsification of data and transactions).
  - **The two important aspects are:**
    - To verify that the contents of the message have not been altered
    - The source is authentic.
  - Also verify timely and in correct sequence of messages
  - Can use conventional encryption
    - Only sender & receiver share a key

Message Authentication and Hash Functions

- Authentication Using Symmetric Encryption
  - It would seem possible to perform authentication simply using symmetric encryption.
    - If we assume that only the sender and receiver share a key, then only the genuine sender would be able to encrypt a message successfully
    - If the message includes an error-detection code and a sequence number, the receiver is assured that no alterations have been made, and that sequencing is proper.
    - If the message also includes a timestamp, the receiver is assured that the message has not been delayed beyond that normally expected for network transit.
  - In fact, symmetric encryption alone is not a suitable tool for data authentication.
    - Block reordering is still a threat.

Message Authentication and Hash Functions

- Message Authentication without Message Encryption
  - Three situations where message authentication without confidentiality is preferable:
    - Message broadcast
      - Example:
        - Notification to users that the network is now unavailable,
        - An alarm signal in a control center
    - There is heavy load on one side that cannot afford the time to decrypt all incoming messages
      - Authentication is carried out on a selective basis
    - Computer program can be executed without having to decrypt it every time, which would be wasteful of processor resources
      - If a message authentication tag were attached to the program, it could be checked whenever assurance is required

Message Authentication and Hash Functions

- Message Authentication Code (MAC)
  - It uses a secret key to generate a small block of data, known as a message authentication code (MAC), that is appended to the message.
  - It assumes that two parties, say A and B, share a common secret key K_{AB}.
  - When A has a message (M) to send to B, it calculates the message authentication code as a complex function of the message and the key: $MAC_M = F(K_{AB}, M)$.

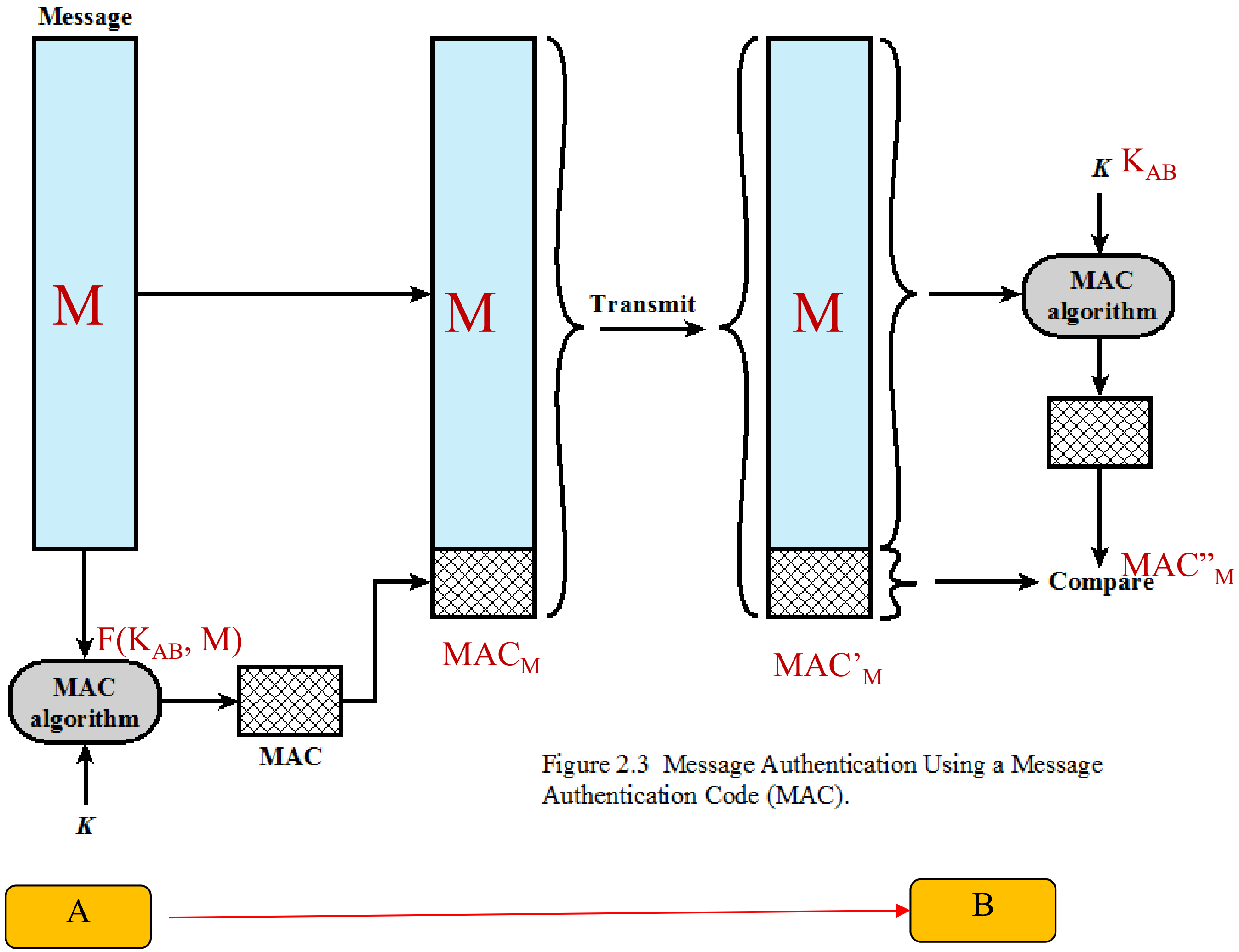


Figure 2.3 Message Authentication Using a Message Authentication Code (MAC).

- DES can be used to generate the MAC, and the last number of bits (16 or 32) of ciphertext are used as the code.

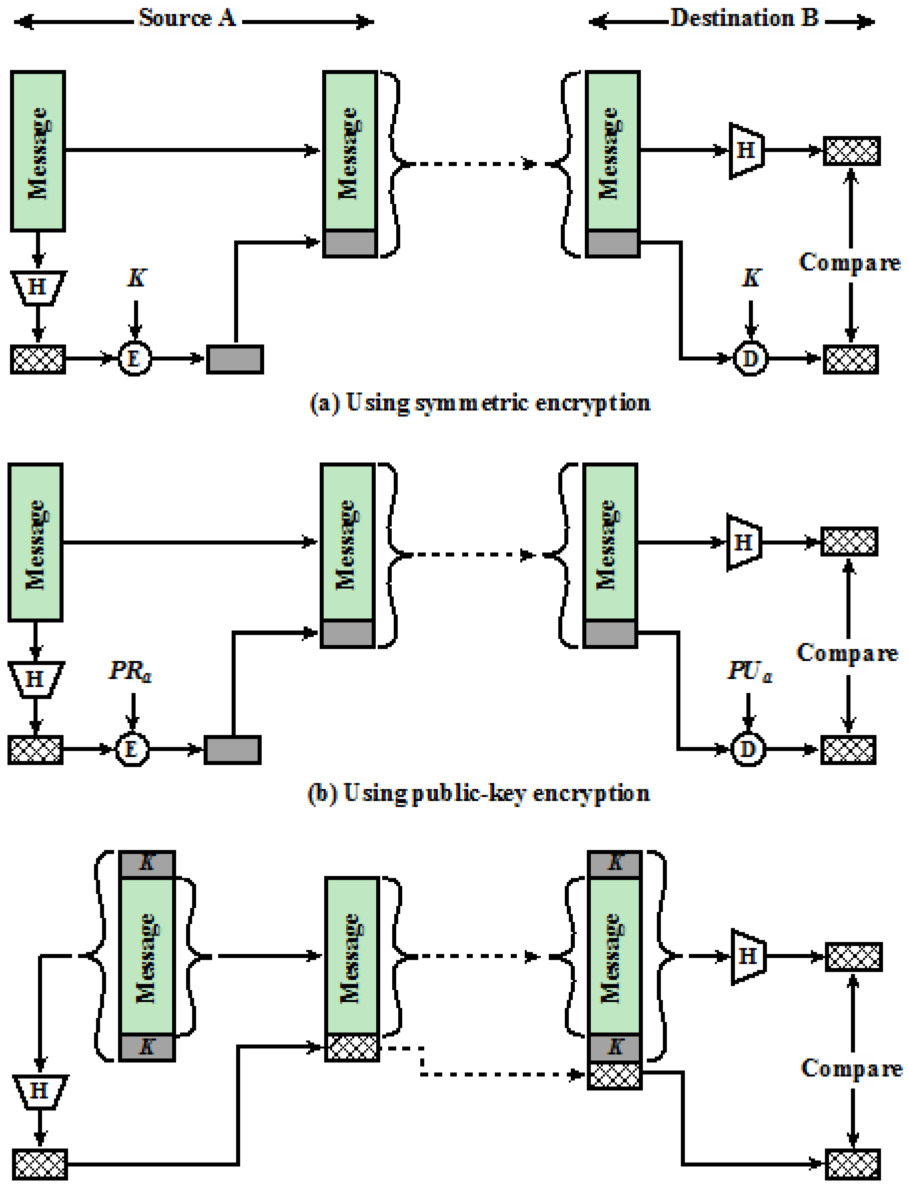
Message Authentication and Hash Functions

- One-Way Hash Function

- Unlike MAC, a hash function does not take a secret key as input.
- To authenticate a message, the message digest is sent with the message.
- Three ways, a message can be authenticated using a hash code:
  a) The message digest can be encrypted using symmetric encryption

  b) The message digest can also be encrypted using public-key encryption

  c) A keyed hash MAC: uses a common key K.
     - A calculates $MD_M = H(K \,ll\, M \,ll\, K)$,
     - then sends [M || MD_M] to B



(a) Using symmetric encryption

(b) Using public-key encryption

(c) Using secret value

Message Authentication and Hash Functions

- Security of Hash Functions

There are two approaches to attacking a secure hash function:

- Cryptanalysis
  - Exploit logical weaknesses in the algorithm
- Brute-force attack
  - Strength of hash function depends solely on the length of the hash code produced by the algorithm

SHA most widely used hash algorithm

- Secure Hash Algorithm
  - SHA-1 (160 bits)
  - SHA-2 (256, 384, 512bits)
  - SHA-3 different that SHA-2

Additional secure hash function applications:

- Passwords
  - Hash of a password is stored by an operating system
- Intrusion detection
  - Store H(F) for each file on a system and secure the hash values (e.g. on CD-R)

Message Authentication and Hash Functions

- Secure Hash Algorithm (SHA)
  - SHA was originally developed by NIST, published as FIPS 180 in 1993
  - Was revised in 1995 as SHA-1
    - Produces 160-bit hash values
  - NIST issued revised FIPS 180-2 in 2002
    - Adds 3 additional versions of SHA
    - SHA-256, SHA-384, SHA-512 with 256/384/512-bit hash values respectively
    - Same basic structure as SHA-1 but greater security
  - The most recent version is FIPS 180-4 which added two variants of SHA-512 with 224-bit and 256-bit hash sizes

Message Authentication and Hash Functions

- Secure Hash Algorithm (SHA)

	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512	SHA-512/224	SHA-512/256
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$	$< 2^{128}$	$< 2^{128}$
Word size	32	32	32	64	64	64	64
Block size	512	512	512	1024	1024	1024	1024
Message digest size	160	224	256	384	512	224	256
Number of steps	80	64	64	80	80	80	80
Security	80	112	128	192	256	112	128

Notes:
1. All sizes are measured in bits.
2. Security refers to the fact that a birthday attack on a message digest of size n produces a collision with a work factor of approximately $2^{n/2}$.