

RT 801 - Security in Computing

Module III **Cryptography**

Instructor: Dr. SABU M THAMPI, Rajagiri School of Engineering and Technology, Kochi, India

Module 3

Cryptography: Basic Encryption & Decryption – Transposition & substitution ciphers – Caesar substitution – Polyalphabetic substitutions – Crypt analysis

Symmetric key algorithms – Fiestel Networks – Confusion – Diffusion – DES Algorithm – Strength of DES – Comparison & important features of modern symmetric key algorithms

Public key cryptosystems – The RSA Algorithm – Diffie Hellman key exchange – comparison of RSA & DES – Message Authentication & Hash functions – Digital signature

Cryptography

- In Greek means “secret writing”
- An outsider (interceptor/intruder/adversary) can make following threats:
 - Block message (affecting availability)
 - Intercept message (affecting secrecy)
 - Modify message (affecting integrity)
- Cryptography is the fundamental technique to counter these threats because the outsider does not understand the meaning of messages

- **Cryptography:** Study of mathematical techniques related to certain aspects of information security, such as confidentiality, data integrity, entity authentication, and data origin authentication.
 - The basic component of cryptography is a **cryptosystem**
- **Cryptanalyst:** Person working for unauthorized interceptor
- **Cryptology:** Study of encryption and decryption, including cryptography and cryptanalysis.

Cryptanalysis

- Means “breaking the code”.
- Cryptanalysis relies on a knowledge of the encryption algorithm and some knowledge of the possible structure of the plaintext (such as the structure of a typical inter-bank financial transaction) for a partial or full reconstruction of the plaintext from ciphertext.
- Additionally, the goal is to also infer the key for decryption of future messages.

Cryptography issues

Confidentiality: only sender, intended receiver should “understand” message contents

- sender encrypts message
- receiver decrypts message

End-Point Authentication: sender, receiver want to confirm identity of each other

Message Integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection.

Cryptosystem

- A *cryptosystem* is a 5-tuple (E, D, M, K, C) , where M is the set of plaintexts, K is the set of keys, C is the set of ciphertexts,
 - $E: M \times K \rightarrow C$ is the set of encipher (encryption) functions, and
 - $D: C \times K \rightarrow M$ is the set of deciphering (decryption) functions.
 - Plaintext M : set of messages in original form
 - Ciphertext C : set of messages in encrypted form
- Security depends on the secrecy of the key, not the secrecy of the algorithm

Cryptosystem....

- **Encryption:** Process of encoding (enciphering) a message so that its meaning is not obvious.
 - Provide confidentiality
 - Principle of Encryption
 - Very hard (impossible) to find out the message without knowing the key
 - Very easy (and fast) to find out the message knowing the key
- **Decryption:** Process of decoding (deciphering or transforming) an encrypted message to its original form.

Conventional Encryption Principles

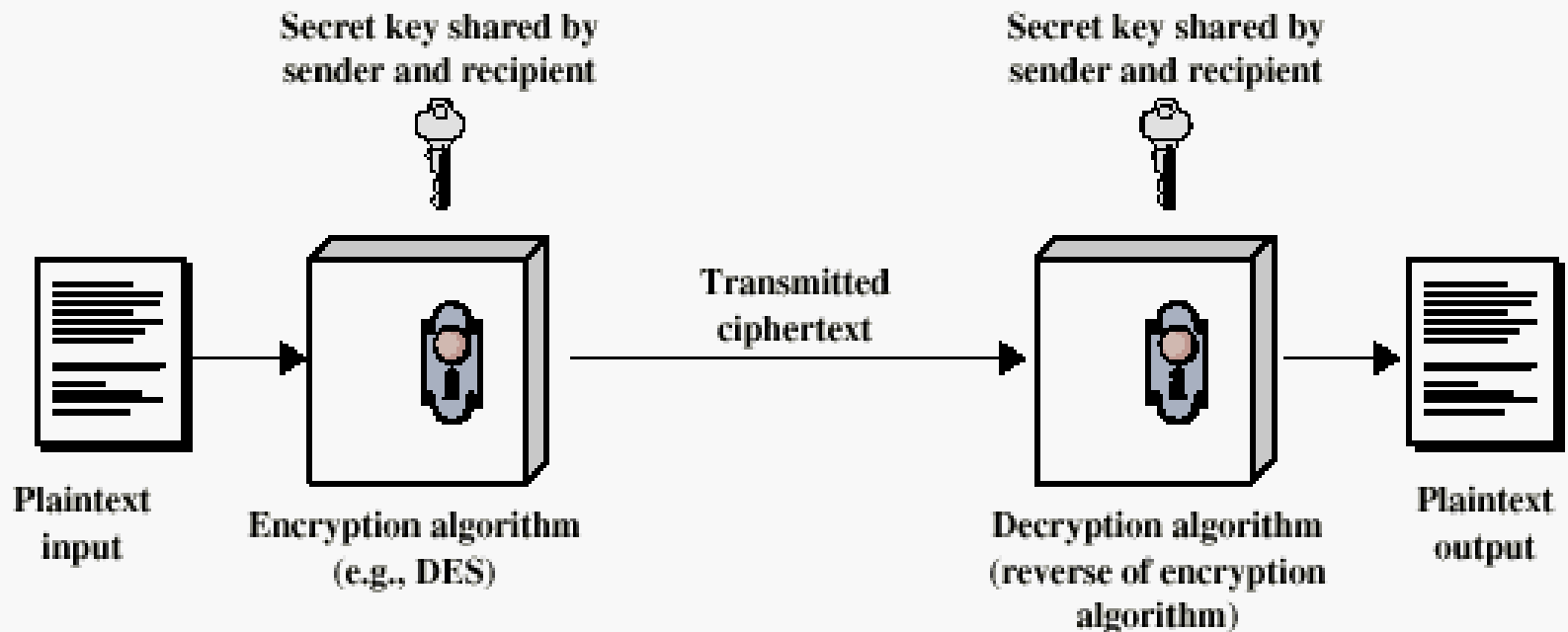


Figure 2.1 Simplified Model of Conventional Encryption

Brute-force attack

- When encryption and decryption algorithms are publicly available
 - a brute-force attack means trying every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained.

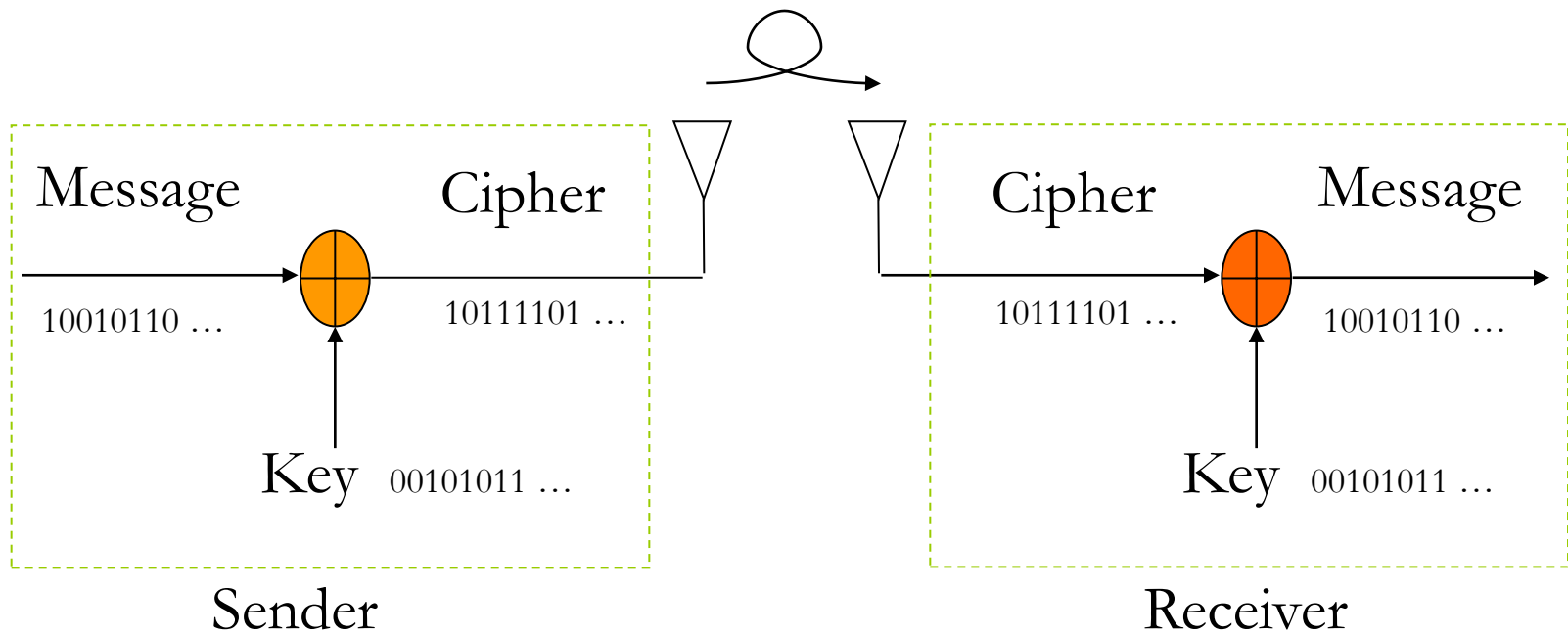
Key space

- The total number of all possible keys that can be used in a cryptographic system.
- For example, DES uses a 56-bit key.
 - So the key space is of size 2^{56} , which is approximately the same as 7.2×10^{16} .

One-Time Pad

- OTP takes a string of random numbers as long as the message
 - OTP uses a symmetric key, which should be statistically proven to be *random*
- Combines the random numbers with the message
 - Perform XOR arithmetic
- This produces ciphertext
- Because all random strings are equally likely, cryptanalysis is impossible

One-Time Pad



One Time Pad ...

- A and B wish to communicate privately using the one-time pad (it uses a symmetric key K)
- They have previously agreed upon secret key K which is a string of n randomly chosen bits
- If A wishes to send an n -bit message M to B, A sends to B the ciphertext $C = M \text{ XOR } K$,
- The received ciphertext can be decrypted by B to obtain M , since $M = C \text{ XOR } K$.
- When another message is to be sent, another key K must be used, hence the name “one-time pad”

One-Time Pad Encryption (example)

- Convert text message to N bits.
- Generate N random bits (one-time pad).
- Take bitwise XOR of two strings.
- Convert binary back into text.

S	E	N	D	M	O	N	E	Y	message
010010	000100	001101	000011	001100	001110	001101	000100	011000	binary
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
100000	010111	111011	111010	010110	110111	101111	111011	001010	XOR
g	X	7	6	W	3	v	7	K	encrypted

One-Time Pad Decryption

- Convert encrypted message to binary.
- Use same N random bits (one-time pad).
- Take bitwise XOR of two strings.
- Convert back into text.

g	X	7	6	W	3	v	7	K	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	binary
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
010010	000100	001101	000011	001100	001110	001101	000100	011000	XOR
S	E	N	D	M	O	N	E	Y	message

Why Does It Work?

- Crucial property: $(a \wedge b) \wedge b = a$.
- Decrypted message = original message.
- Use properties of XOR.

$$(a \wedge b) \wedge b = a \wedge (b \wedge b) = a \wedge 0 = a$$

Notation	Meaning
a	original message
b	one-time pad
\wedge	XOR operator
$a \wedge b$	encrypted message
$(a \wedge b) \wedge b$	decrypted message

Difficulties of OTP

- To ensure the security of the system, key size should not be less than message size
- Generating a fully random key is practically very difficult
- Sending an unrepeated key with the same size of the message through a secure channel to the receiver is not practical

Classical Cryptography

- Basic techniques for classical ciphers
 - **Substitution**: One letter is exchanged for another
 - **Transposition**: The order of the letters is rearranged (also referred to as permutation)
- Classical ciphers
 - **Mono-alphabetic**: Letters of the plaintext alphabet are mapped into other unique letters
 - **Poly-alphabetic**: Letters of the plaintext alphabet are mapped into letters of the ciphertext space depending on their positions in the text

Substitution

- Substitute each letter in the plaintext for another one
 - **Monoalphabetic cipher**: substitute one letter for another
 - **Ciphers in which the cipher alphabet remains unchanged throughout the message are called Monoalphabetic Substitution Ciphers.**
- **Goal**: Confusion – difficult to determine how a message and key were transformed into ciphertext.
- *Example* (**Caesar Cipher**)
 - a b c d e f g h i j k l m n o p q r s t u v w x y z
 - q e r y u i o p a s d f g w h j k l z x c v b n m t

under attack we need help

➔ cwyul qxxqrd bu wuuy pufj

Caesar Cipher

- This is the earliest known example of a substitution cipher.
- Each character of a message is replaced by a character three position down in the alphabet.

plaintext: are you ready

ciphertext: DUH BRX UHGGB

Caesar Cipher.....

- If we represent each letter of the alphabet by an integer that corresponds to its position in the alphabet, the formula for replacing each character 'p' of the plaintext with a character 'C' of the ciphertext can be expressed as

$$C = E(3, p) = (p + 3) \bmod 26$$

Caesar Cipher.....

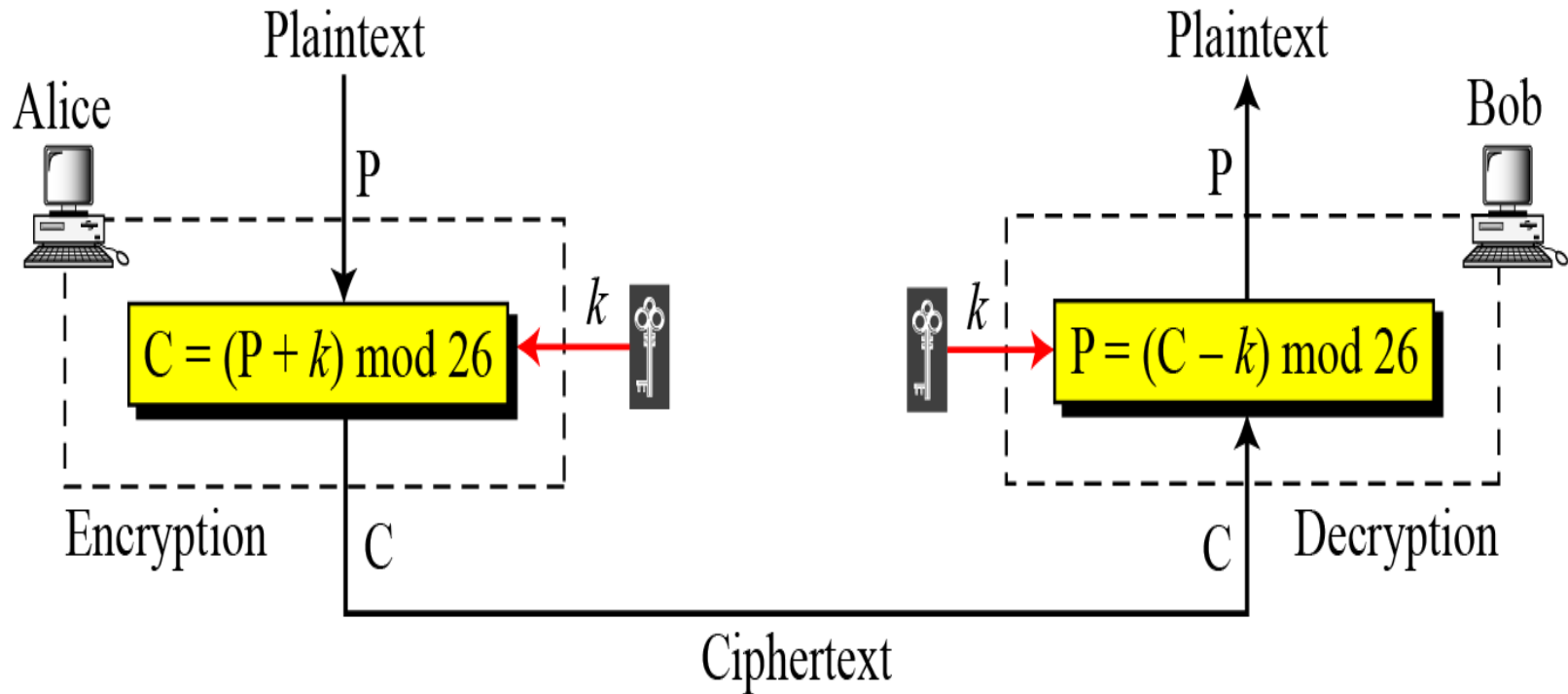
- A more general version of this cipher that allows for any degree of shift would be expressed by

$$C = E(k, p) = (p + k) \bmod 26$$

- The formula for decryption would be

$$p = D(k, C) = (C - k) \bmod 26$$

Caesar Cipher.....



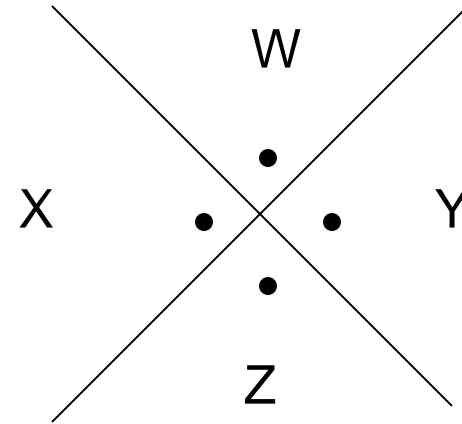
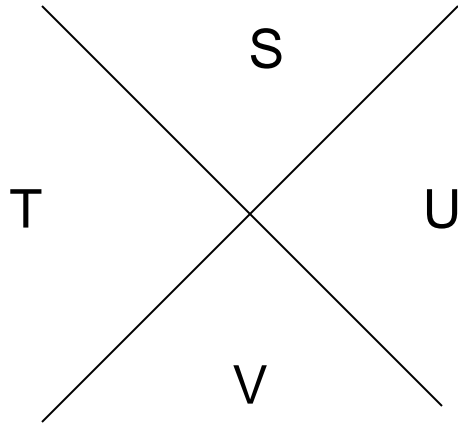
Pigpen Cipher

- Monoalphabetic cipher
- Pigpen cipher is a variation on letter substitution
- Alphabets are arranged as follows:

A	B	C
D	E	F
G	H	I

J •	K •	L •
M •	N •	O •
P •	Q •	R •

Pigpen Cipher diagram



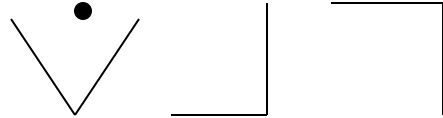
A =

C =

G =

W =

Pigpen Cipher

- Alphabets will be represented by the corresponding diagram
- E.g., WAG would be The diagram consists of three symbols: a triangle with a dot at its top vertex, a square with its top-right corner missing, and a square with its top-left corner missing.
- This is a weak cipher

In monoalphabetic substitution, the relationship between a symbol in the plaintext to a symbol in the ciphertext is always **one-to-one.**

Polyalphabetic Cipher

- In monoalphabetic cipher the problem was that each character was substituted by a single character.
- They are easy to break because they reflect the frequency data of the original alphabet.
- Polyalphabetic cipher's goal is to make this process difficult.

Polyalphabetic Cipher

- In polyalphabetic cipher, each plaintext character may be replaced by more than one character
- Since there are only 26 alphabets this process will require using a different representation than the alphabets
- Alphabets 'A' through 'Z' are replaced by 00, 01, 02, ..., 25

Polyalphabetic Cipher

- The most common method used is **Vigenère cipher**
- Vigenère cipher starts with a 26 x 26 matrix of alphabets in sequence. First row starts with 'A', second row starts with 'B', etc.
- This cipher requires a **keyword** that the sender and receiver know ahead of time.
- Each character of the message is combined with the characters of the keyword to find the ciphertext character.

Vigenère Cipher Table

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	A	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L

Vigenère Cipher Table

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Vigenère Cipher Table (Full)

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	v	v	w	x	y	z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Vigenère cipher

- E.g., Message = SEE ME IN MALL
- Take keyword as INFOSEC
- Vigenère cipher works as follows:

S E E M E I N M A L L

I N F O S E C I N F O

A R J A W M P U N Q Z

Vigenère cipher

- To **decrypt**, the receiver places the keyword characters below each ciphertext character
- Using the table, choose the row corresponding to the keyword character and look for the ciphertext character in that row
- Plaintext character is then at the top of that column

Vigenère cipher

- Decryption of ciphertext:

A R J A W M P U N Q Z

I N F O S E C I N F O

S E E M E I N M A L L

- Best feature is that same plaintext character is substituted by different ciphertext characters (i.e., polyalphabetic)

Polyalphabetic Ciphers Strength

- There are multiple ciphertext letters for each plaintext letter, one for each unique letter of the keyword. Thus, the letter frequency information is hidden.

TRANSPOSITION CIPHERS

A transposition cipher does not substitute one symbol for another, instead it changes the location of the symbols.

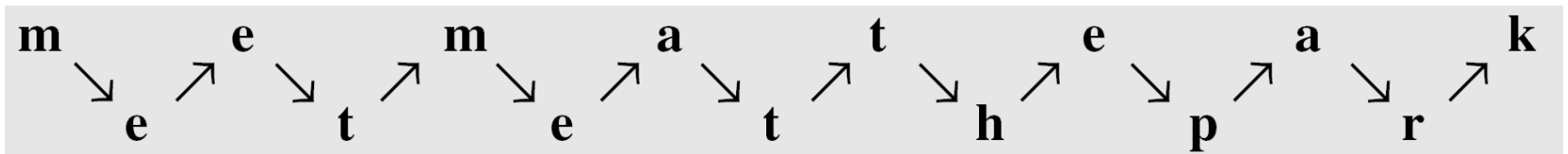
A transposition cipher reorders symbols.

Keyless Transposition Ciphers

Simple transposition ciphers, which were used in the past, are keyless.

A good example of a keyless cipher using the first method is the **rail fence cipher**. The ciphertext is created reading the pattern row by row.

For example, to send the message “Meet me at the park” to Bob, Alice writes



She then creates the ciphertext “**MEMATEAKETETHPR**”.

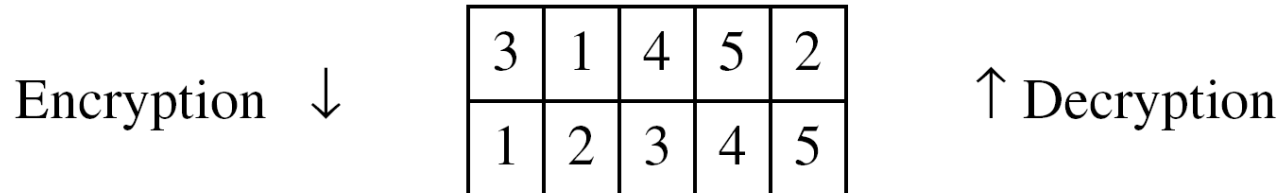
Alice and Bob can agree on the number of columns and use the second method. Alice writes the same plaintext, row by row, in a table of four columns.

m	e	e	t
m	e	a	t
t	h	e	p
a	r	k	

She then creates the ciphertext “**MMTAEEHREAEKTTP**”.

Transposition (using keys)

- Ciphertext: write out the **columns** in an order specified by a key.
- **Goal:** Diffusion – spread the information from the message or the key across the ciphertext.



Encryption ---3 (third column) becomes 1 (first column) and so on

Decryption – 1 (first column) becomes 3 (third column) and so on

The cipher can be made more secure by performing multiple rounds of such permutations.

Alice needs to send the message “**Enemy attacks tonight**” to Bob..

e n e m y a t t a c k s t o n i g h t z

The permutation yields

E E M Y N T A A C T T K O N S H I T Z G

Enemy Attack tonight

e	n	e	m	y
a	t	t	a	c
k	s	t	o	n
i	g	h	t	z

Encryption ↓

3	1	4	5	2
1	2	3	4	5

↑ Decryption

e	e	m	y	n
t	a	a	c	t
t	k	o	n	s
h	i	t	z	g

Cipher Text

etth eaki maot ycnz ntsg

Alice



Plaintext

e n e m y a t t a c k s t o n i g h t z

Write row by row

e	n	e	m	y
a	t	t	a	c
k	s	t	o	n
i	g	h	t	z

E	E	M	Y	N
T	A	A	C	T
T	K	O	N	S
H	I	T	Z	G

Read column by column

E T T H E A K I M A O T Y C N Z N T S G

Ciphertext

Encrypt

3	1	4	5	2
1	2	3	4	5

Key

Decrypt

Bob



Plaintext

e n e m y a t t a c k s t o n i g h t z

Read row by row

e	n	e	m	y
a	t	t	a	c
k	s	t	o	n
i	g	h	t	z

E	E	M	Y	N
T	A	A	C	T
T	K	O	N	S
H	I	T	Z	G

Write column by column

E T T H E A K I M A O T Y C N Z N T S G

Ciphertext

Transmission

Confusion and Diffusion

- **Confusion**
 - A technique that seeks to make the relationship between the statistics of the ciphertext and the value of the encryption keys as complex as possible. Cipher uses key and plaintext.
- **Diffusion**
 - A technique that seeks to obscure the statistical structure of the plaintext by spreading out the influence of each individual plaintext digit over many ciphertext digits.

Types of Cryptosystems

- *Symmetric cryptosystems* (also called *single-key* cryptosystems) are classical cryptosystems:

$$M = D(K, E(K, M))$$

- The encryption key and decryption key are the same.

- *Asymmetric cryptosystem*:

$$M = D(K_d, E(K_e, M))$$

- K_d is the decryption key and K_e is the encryption key
- $K_d \neq K_e$

Classical Cryptography: Secret-Key or Symmetric Cryptography

- Alice and Bob agree on an encryption method and a shared **key**.
- Alice uses the key and the encryption method to **encrypt** (or **encipher**) a message and sends it to Bob.
- Bob uses the same key and the related decryption method to **decrypt** (or **decipher**) the message.

Four Secure Key Distribution Strategies for Symmetric Cryptosystems

1. A key K can be selected by A to be shared with B, and K needs to be physically delivered to B
2. A third party can select the same key K and physically deliver K to A and B
3. If A and B have previously used a key K' , one party can transmit the new key K to the other, encrypted using the old key K'
4. If A and B each has an encrypted connection to a third party C, C can transmit the new key K on the encrypted links to both A and B

Public-Key Cryptography: Asymmetric Cryptography

- Alice generates a key value (usually a number or pair of related numbers) which she makes public.
- Alice uses her public key (and some additional information) to determine a second key (her ***private key***).
- Alice keeps her private key (and the additional information she used to construct it) secret.

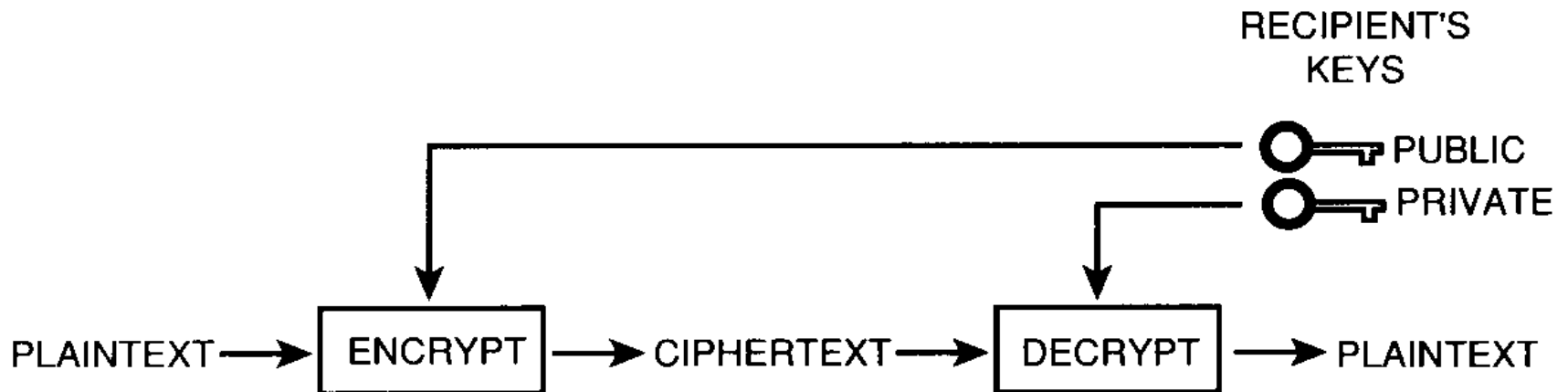
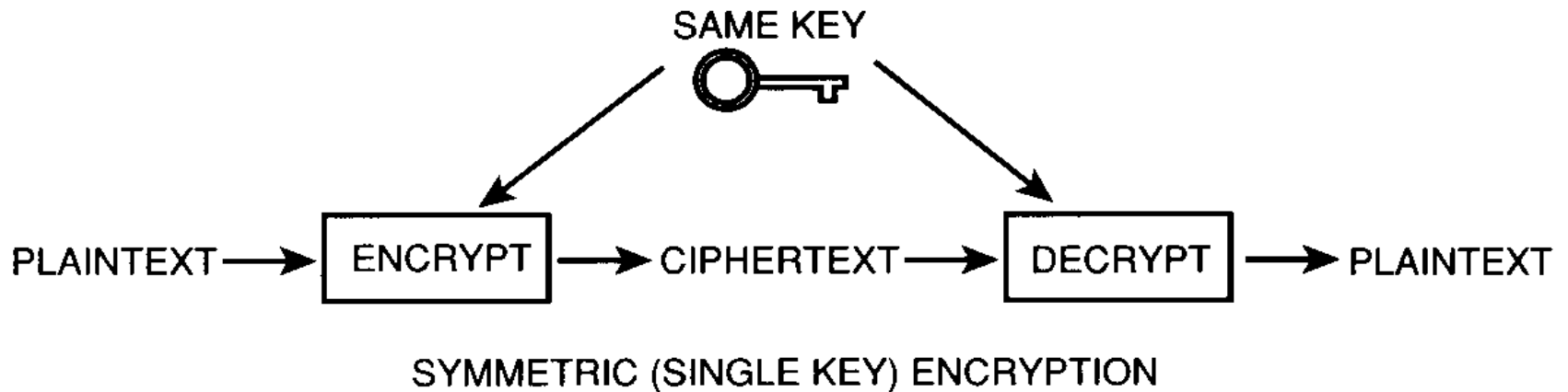
Public-Key Cryptography

- Bob (or Carol, or anyone else) can use Alice's public key to encrypt a message for Alice.
- Alice can use her private key to decrypt this message.
- No- one without access to Alice's private key can easily decrypt the message.

An Example: Internet Commerce

- Bob wants to use his credit card to buy some items from Alice over the Internet.
- Alice sends her public key to Bob.
- Bob uses this key to encrypt his credit-card number and sends the encrypted number to Alice.
- Alice uses her private key to decrypt this message (and get Bob's credit-card number).

Symmetric and Asymmetric Encryption

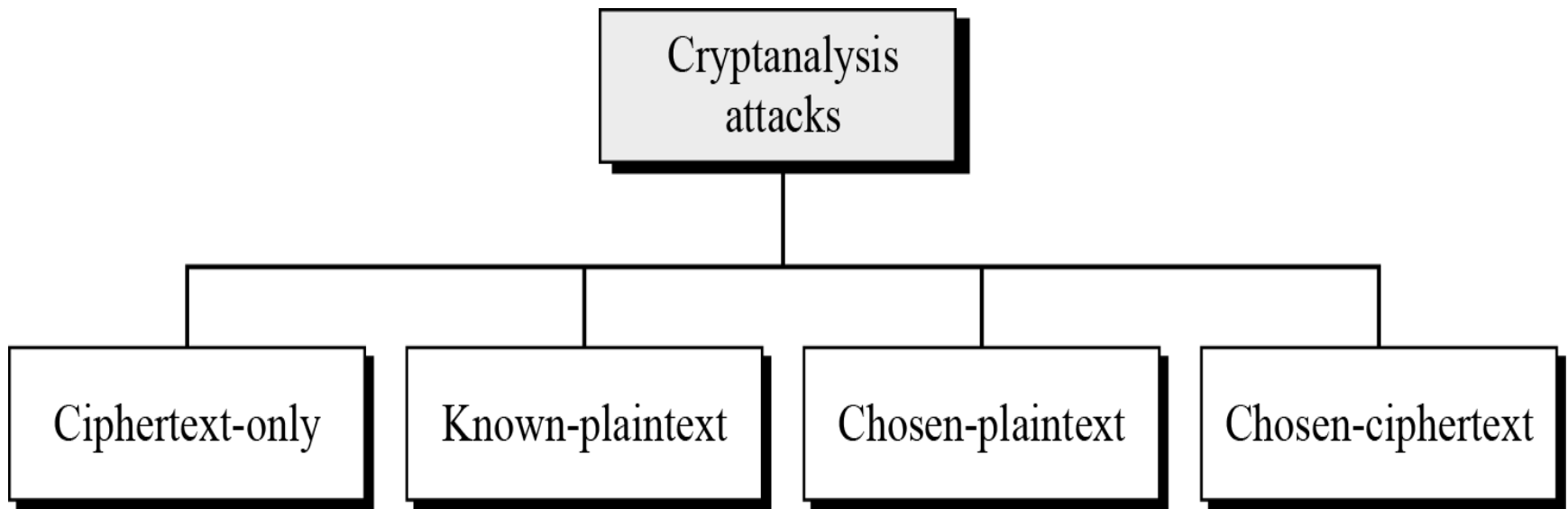


Kerckhoff's Principle

A cryptosystem should be secure even if everything about the system, except the key, is public knowledge.

Cryptanalysis

As cryptography is the science and art of creating secret codes, **cryptanalysis** is the science and art of breaking those codes.



May be classified by how much information needed by the attacker

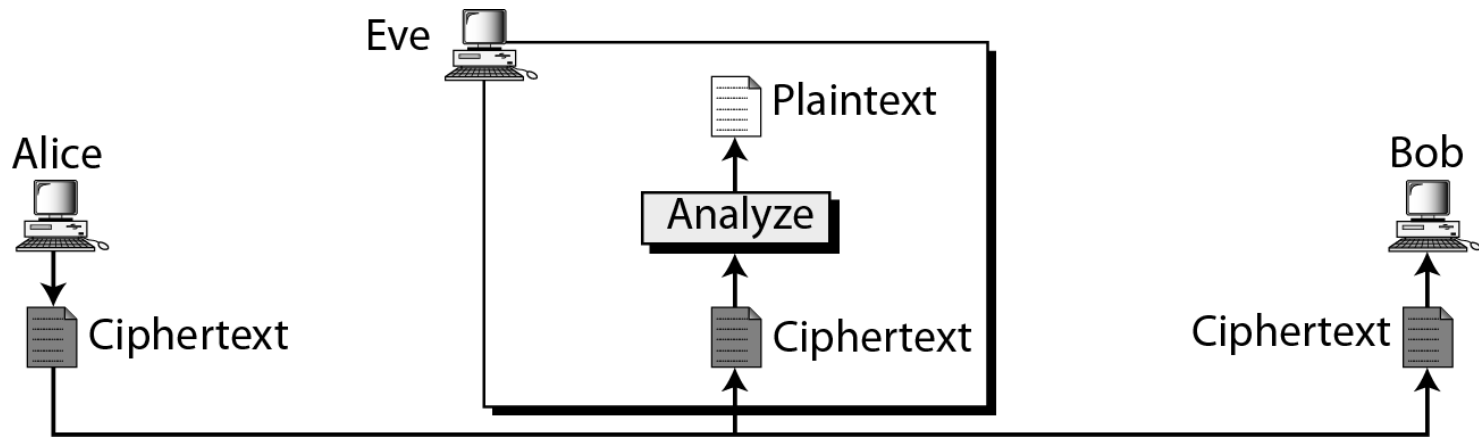
Ciphertext-Only Attack

- In *ciphertext only attack*, encryption algorithm and ciphertext are known to the cryptanalyst.
- With simple ciphers, such as the Caesar Cipher, frequency analysis can be used to break the cipher.
- Monoalphabetic substitution ciphers are vulnerable to ciphertext-only attacks.
 - Each character was substituted by a single character.
 - Easy to break because they reflect the frequency data of the original alphabet.

Ciphertext-Only Attack

Given: a ciphertext c

Q: what is the plaintext m ?



Known plaintext attack

- In *known plaintext attack*, information known includes: encryption algorithm, ciphertext, and one or more plaintext-ciphertext pairs formed with the secret key.
- Seeks to discover a correlation between plaintext and the corresponding ciphertext .
 - The attacker knows or can guess the plaintext for some parts of the ciphertext.
 - The task is to decrypt the rest of the ciphertext blocks using this information.
 - This may be done by determining the key used to encrypt the data.

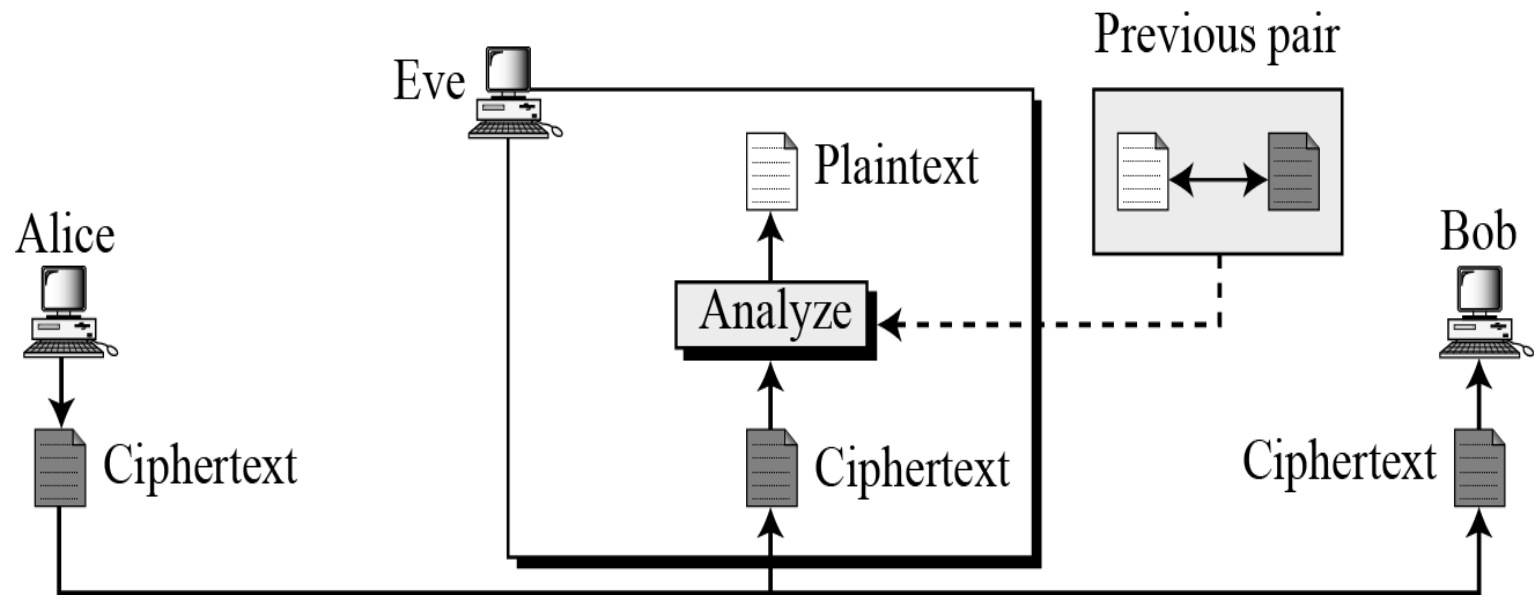
Known-Plaintext Attack

The attacker has samples of both the plaintext and its encrypted version (ciphertext).

Given: $(m_1, c_1), (m_2, c_2), \dots, (m_k, c_k)$ and a new ciphertext c .

Q: what is the plaintext of c ?

Q: what is the secret key in use?



- **Plain Text + Key = Cipher Text**
- **Cipher Text - Plain Text = Key**

Known-Plaintext Attack – example 1

- Alice sends Bob an email and attaches her favorite holiday snapshot. The email is **encrypted**.
- She sends the same holiday snapshot to her mother in **plain text**.
- Steve, who wishes to spy on Alice and Bob, was able to intercept her email to Mom and now has a copy of "myholiday.jpg".

Known-Plaintext Attack – example 1.....

- If the picture consisted of 200 Kilobytes of data and Alice included only a short personal message to Bob with the picture (say 50 letters)
 - then Steve already knows 99% of the message contents prior to encryption and now has greatly improved chances of breaking Alice's key if he comes into possession of the corresponding cipher text.

Known-Plaintext Attack – example 2

- An example with monoalphabetic substitution cipher.
- The following ciphertext is intercepted and is known to contain information about a person called “**ANDERSON**” and a place called “**MISSISSIPPI**”.

JZKGXAHZDAVWGZGBWGJKHUAIDGADZEDAADAADIID

JZKGXAHZDAVWGZGBWVGJKHUAIDGADZ**EDAADAADIID**

- We could use our information of the plaintext to devise stronger attack.
- MISSISSIPPI - we look for a sequence of 11 letters where the 3rd, 4th, 6th and 7th letters are the same and so are the 2nd, 5th, 8th and 11th.
 - The sequence ‘EDAADAADIID’ is the ciphertext for ‘MISSISSIPPI’.

JZKGXAHZDAVWGZGBWVGJKHUAIDGADZ**EDADAADIID**

- For ANDERSON

- Look for a sequence of 8 letters where all characters are different except for the 2nd and the 8th.
- Eventually the attacker will find that ‘JZKGXAHZ’ represents ‘ANDERSON’

ANDERSON **DA**VWGZGBWVGJKHUAIDGADZ MISSISSIPPI

ANDERSON **IS** VWGZGBWVGJKHUAIDGADZ MISSISSIPPI

JZKGXAHZDAVWGZGBWVGJKHUAIDGADZEDAADAADIID

- With the newly gained information so far, the ciphertext has been decrypted to

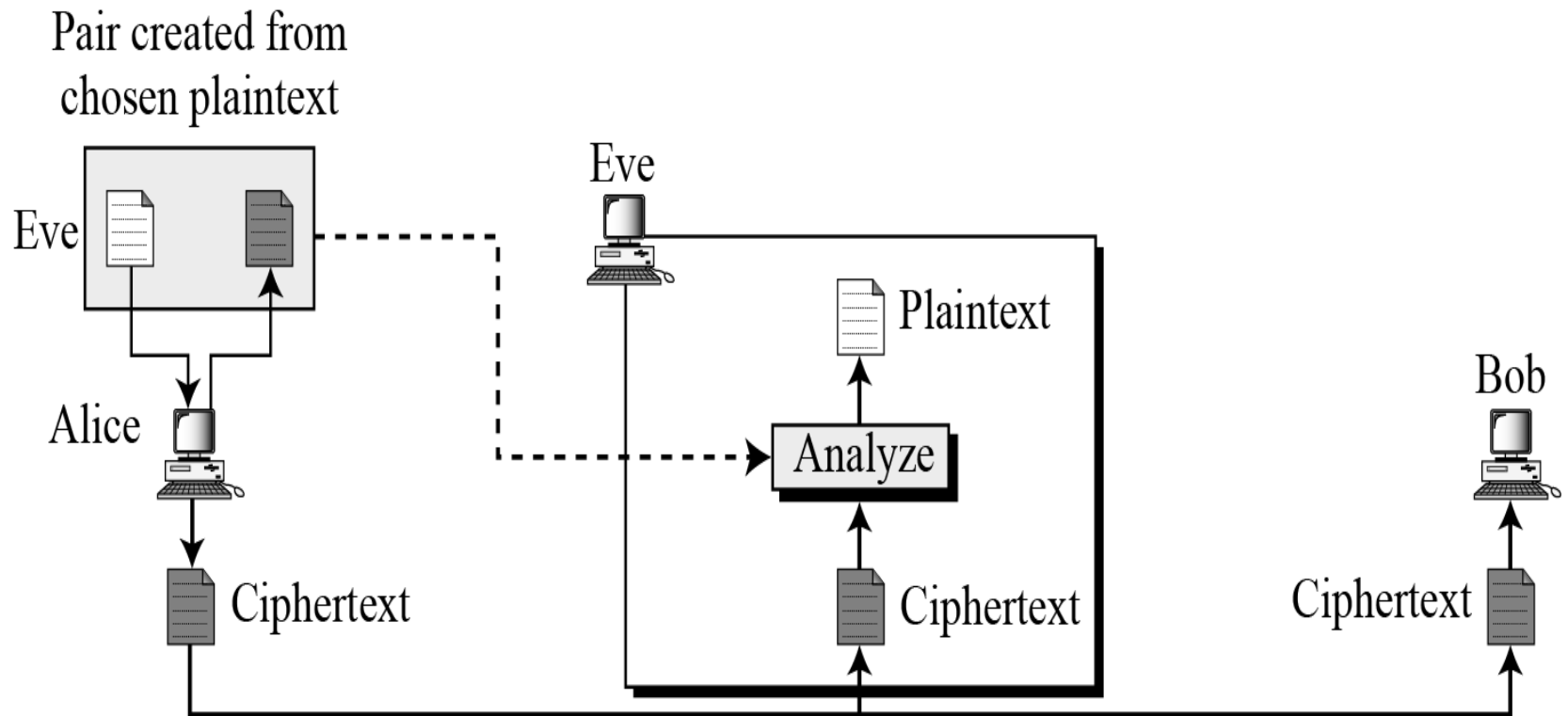
ANDERSON IS VWENEBWEADOU SPIES IN MISSISSIPPI

- Subsequent effort of cryptanalysis may eventually reveal the secret

ANDERSON IS THE NEW HEAD OF SPIES IN MISSISSIPPI

Chosen-Plaintext Attack

- In *chosen plaintext attack*, information known includes: encryption algorithm, ciphertext, and chosen plaintext and its corresponding ciphertext generated with the secret key.
- Given: (m_1, c_1) , (m_2, c_2) , ..., (m_k, c_k) , where m_1, m_2, \dots, m_k are chosen by the adversary; and a new ciphertext c .
 - Q: what is the plaintext of c , or what is the secret key?
- For a chosen plaintext attack, an attacker chooses plaintext and submits it to be encrypted.
- Attackers can then analyze the ciphertext that corresponds to the chosen plaintext, identify subtle differences and patterns, and quickly break the encryption.



Given: $P_1, C_1 = E_k(P_1), P_2, C_2 = E_k(P_2), \dots, P_t, C_t = E_k(P_t),$
 where the cryptanalyst gets to choose P_1, P_2, \dots, P_t

Chosen-Plaintext Attack

- For example, suppose we want to attack communication from Alice to Bob which is encrypted by monoalphabetic substitution cipher.
- Assume the intercepted messages so far could not be solved using frequency analysis.
- If we can get Alice to send an encrypted message to Bob which contains the word 'MISSISSIPPI'.
 - Here, we can send an email to Alice, “*Please tell Bob that saying Mississippi will take exactly one second*”.
 - Assume Alice simply forwards our written email.
 - We can intercept the message and obtain some information about the mapping of plaintext to ciphertext used in encryption of communication from Alice to Bob.

Example: chosen-plaintext attack

- In 1942, US Navy cryptanalysts discovered that Japan was planning an attack on “AF”.
- They believed that “AF” means Midway island.
- Pentagon didn’t think so.
- US forces in Midway sent a plain message that their *fresh water supplies were low*.
- Shortly, US intercepted a Japanese ciphertext saying that “AF” was low on water.
- This proved that “AF” is Midway.

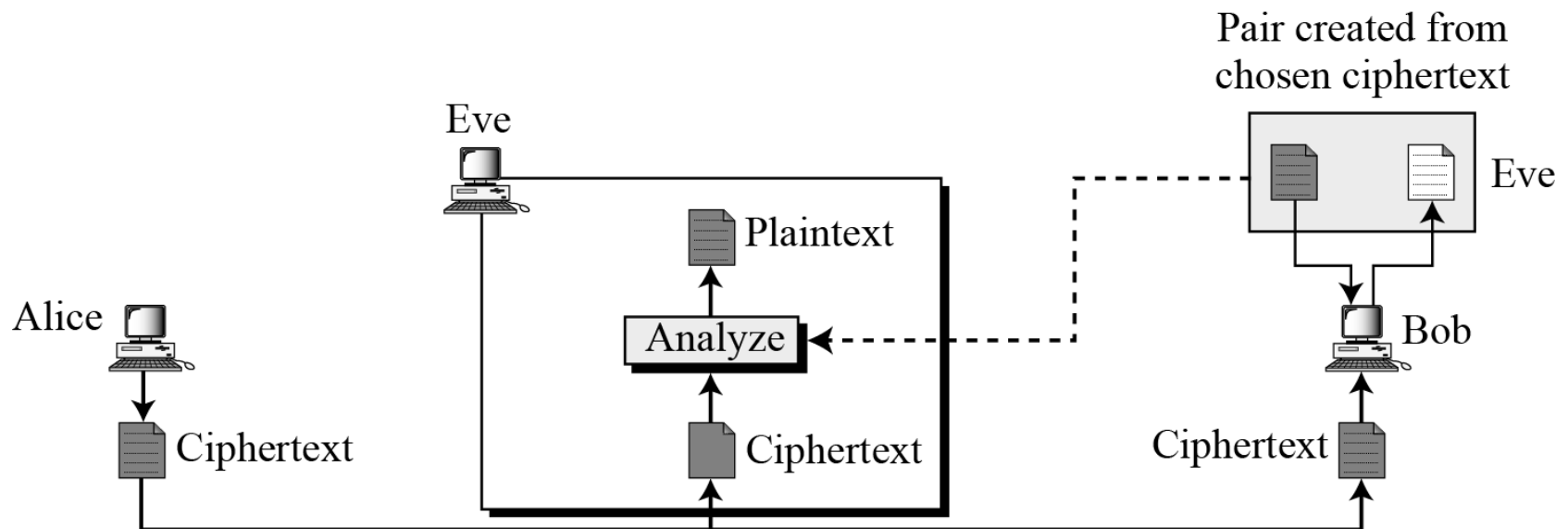
Chosen-Ciphertext Attack

Given: $(m_1, c_1), (m_2, c_2), \dots, (m_k, c_k)$, where c_1, c_2, \dots, c_k are chosen by the adversary; and a new ciphertext c .

Q: what is the plaintext of c , or what is the secret key of sender?

A chosen ciphertext attack is an attack where a cryptanalyst chooses a ciphertext and attempts to find a matching plaintext.

**An attacker gains access to an unattended decryption machine.
Decrypted with an unknown key**



- In *chosen ciphertext attack*, information known includes: encryption algorithm, ciphertext, and chosen ciphertext and its corresponding decrypted plaintext with the secret key.
- it has also been called a “*lunch-time*” or “*midnight*” attack.

Cryptographic Systems - Classification

- The type of operations used for transforming plaintext to ciphertext
 - Substitution
 - Transposition
 - Product systems – multiple stages of substitutions and transpositions
- The number of keys used
 - Symmetric
 - Asymmetric
- The way in which the plaintext is processed
 - Block cipher
 - Stream cipher

Block Ciphers

- Encrypt data one block at a time
- Operate on a plaintext block of n bits
- Produce a ciphertext block of n bits
- Typical block size 64 – 128 bits

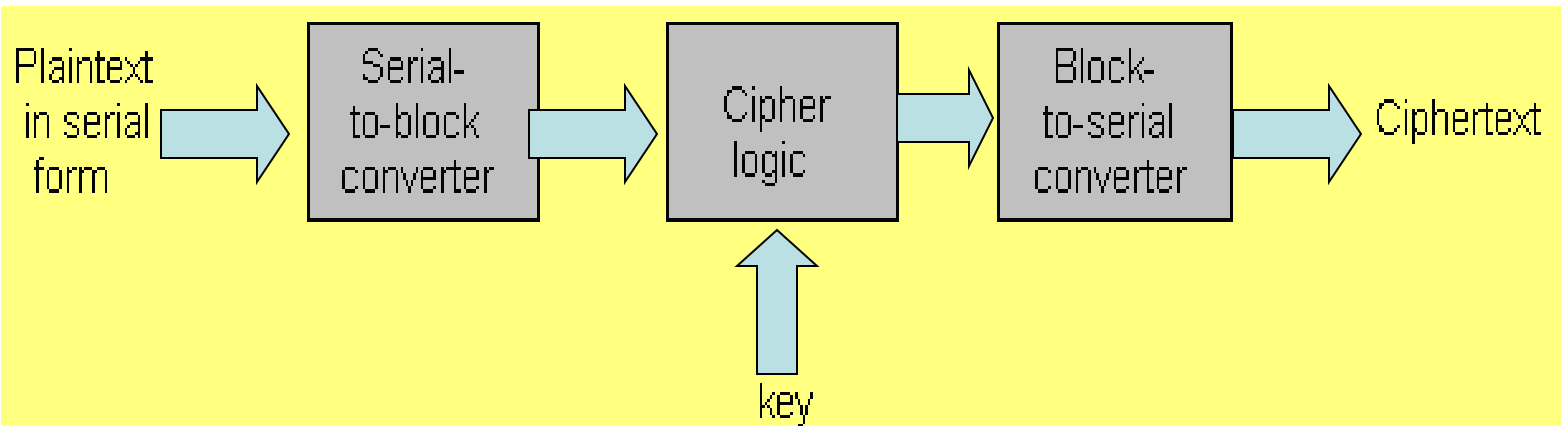
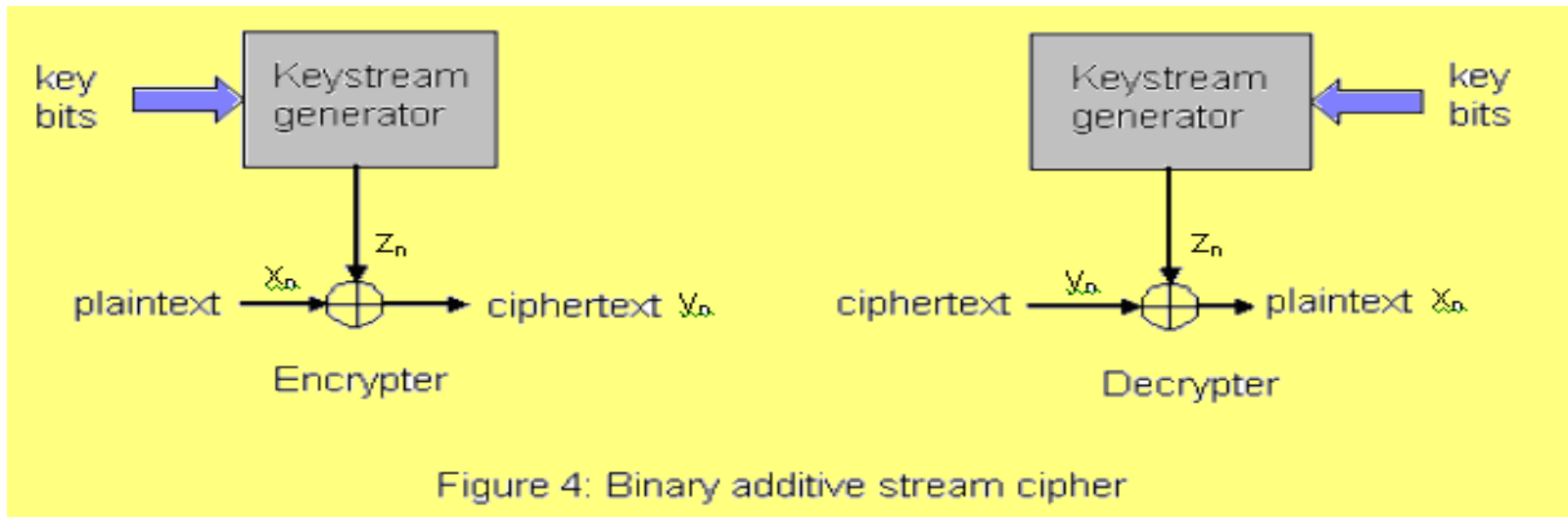


Figure 3: Block diagram of a block cipher

Stream ciphers

- Whereas block ciphers operate on large data on a block-by-block basis, stream ciphers operate on individual bits.



Let $x_n \rightarrow$ Plaintext bit; $y \rightarrow$ ciphertext bit; $z \rightarrow$ keystream bit at n^{th} instant

For encryption: $y_n = x_n \oplus z_n, \quad n=1, 2, \dots, N$

For decryption: $x_n = y_n \oplus z_n, \quad n=1, 2, \dots, N$

Steganography

Secure Communication

Two parties, Alice and Bob, can exchange information over an **insecure medium** in such a way that even if an intruder (Willie) is able to intercept, read and perform computation on the intercepted information, Willie will not be able to decipher the content of the exchanged information.

Encryption may not be enough

- Prisoners Problem:
 - Alice and Bob are in jail and wish to hatch an escape plan . All their communications pass through the warden, Willie, and if Willie detects any encrypted messages, he can simply stop the communication.
- So they must find some way of hiding their secret message in an innocuous looking text.

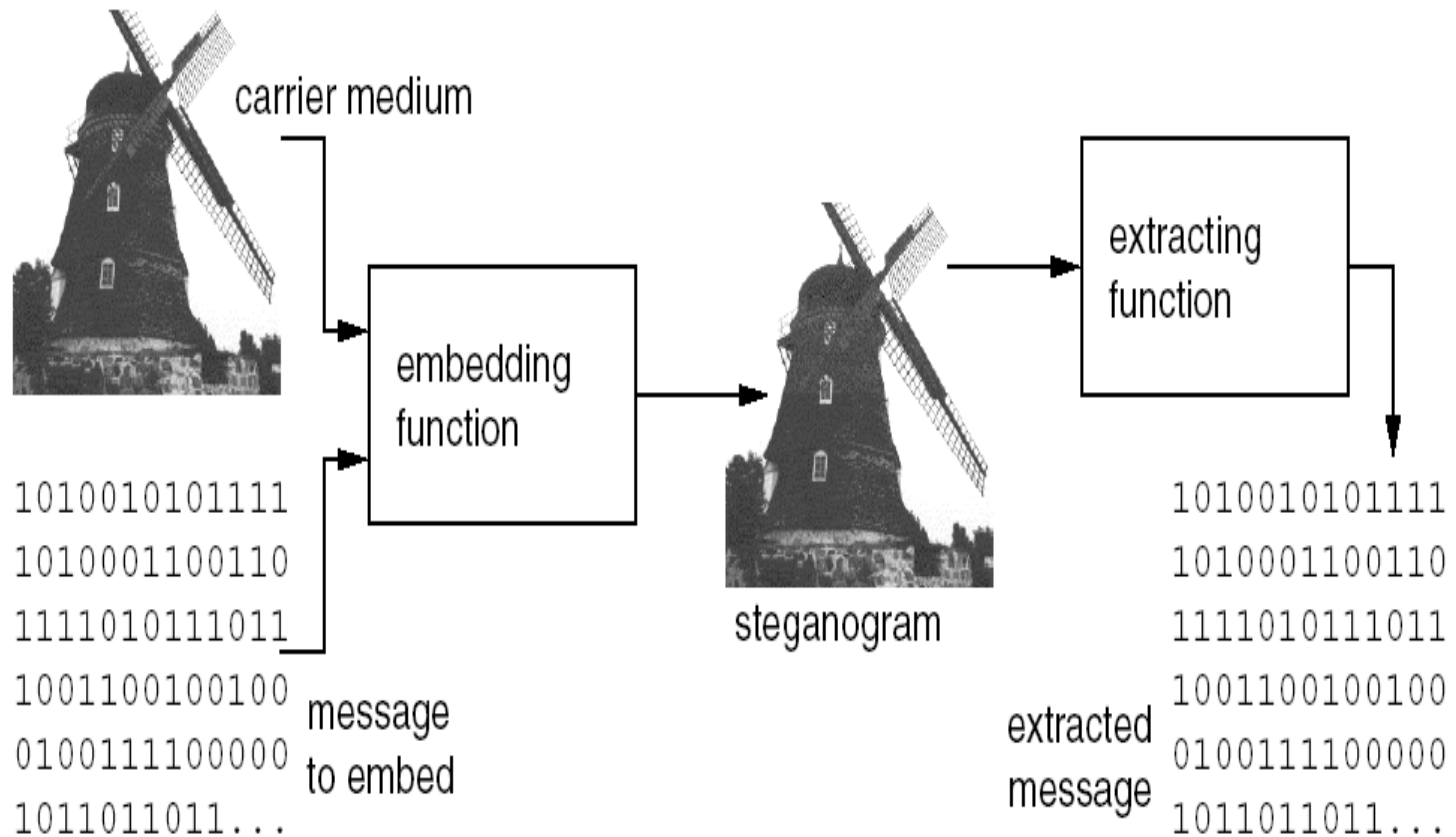
Steganography

- The art of **hiding** information in ways that prevent detection of hidden messages.
- In Greek means “covered writing”
- Steganography and cryptography are cousins in the spy craft family
- While the goal of the cryptography system is to conceal the content of the messages, the goal of information hiding or steganography is to conceal their existence

Steganography

- What to hide
 - Texts
 - Images
 - Sound
- How to hide
 - embed text in text/images/audio/video files
 - embed image in text/images/audio/video files
 - embed sound in text/images/audio/video files

Steganographic System

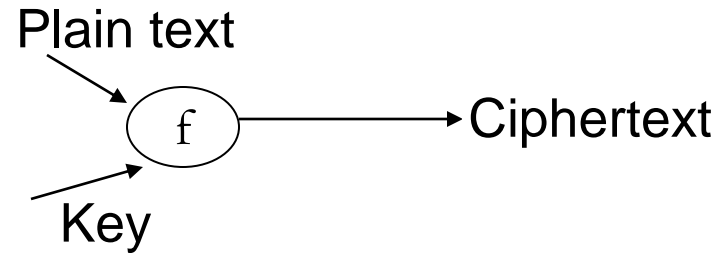


Comparison

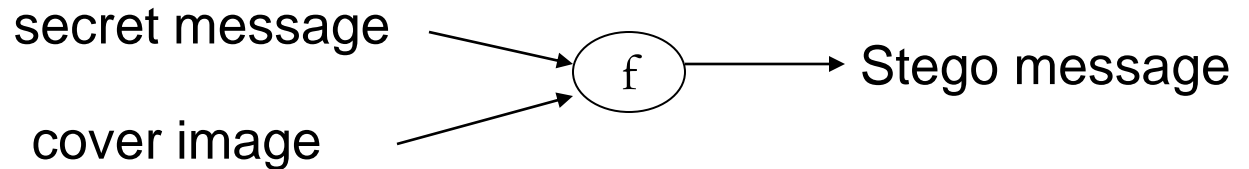
Cryptography

$$C = E_k (P)$$

$$P = D_k (C)$$



Steganography



A Real Example

- During WW2 the following cipher message was actually sent by a German spy
 - “Apparently neutral’s protest is thoroughly discounted and ignored. Isman hard hit. Blockade issue affects pretext for embargo on by-products, ejecting suets and vegetable oils”
- Hidden Message
 - “Pershing sails from NY June 1”
 - Can be obtained by extracting the second letter in each word of the message sent

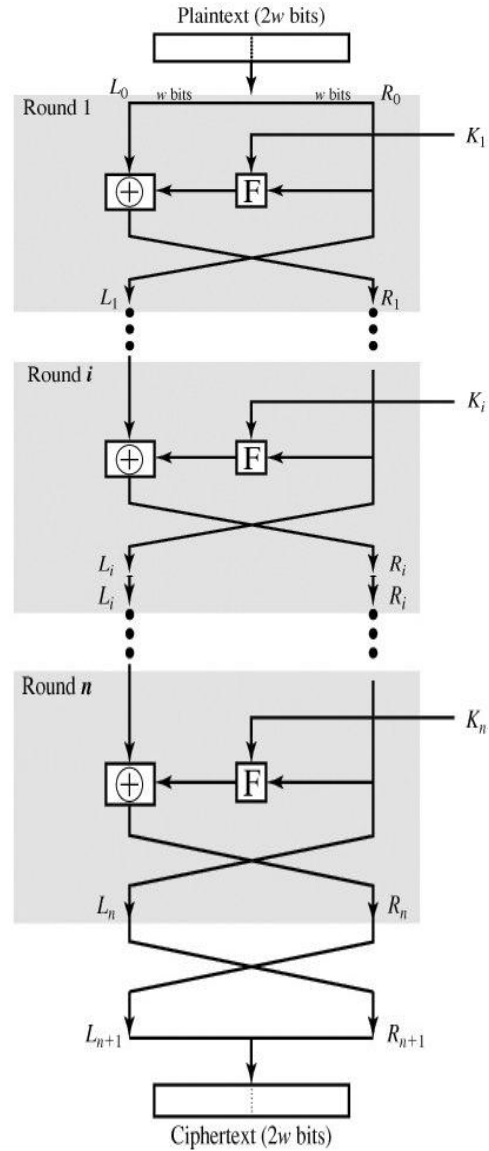
Fiestel Cipher

- A Feistel cipher is a symmetric structure used in the construction of block ciphers
- Named after the German-born physicist and cryptographer **Horst Feistel** who did pioneering research while working for IBM (USA).
- It is also commonly known as a *Feistel network*.
- A large proportion of block ciphers use the scheme, including the Data Encryption Standard (DES).

(from Wiki)

- The Feistel structure has the advantage that encryption and decryption operations are very similar, even identical in some cases, requiring only a reversal of the key schedule.
- The ciphertext is calculated from the plaintext by repeated application of same transformation or round function.

Fiestel Cipher



Fiestel Cipher

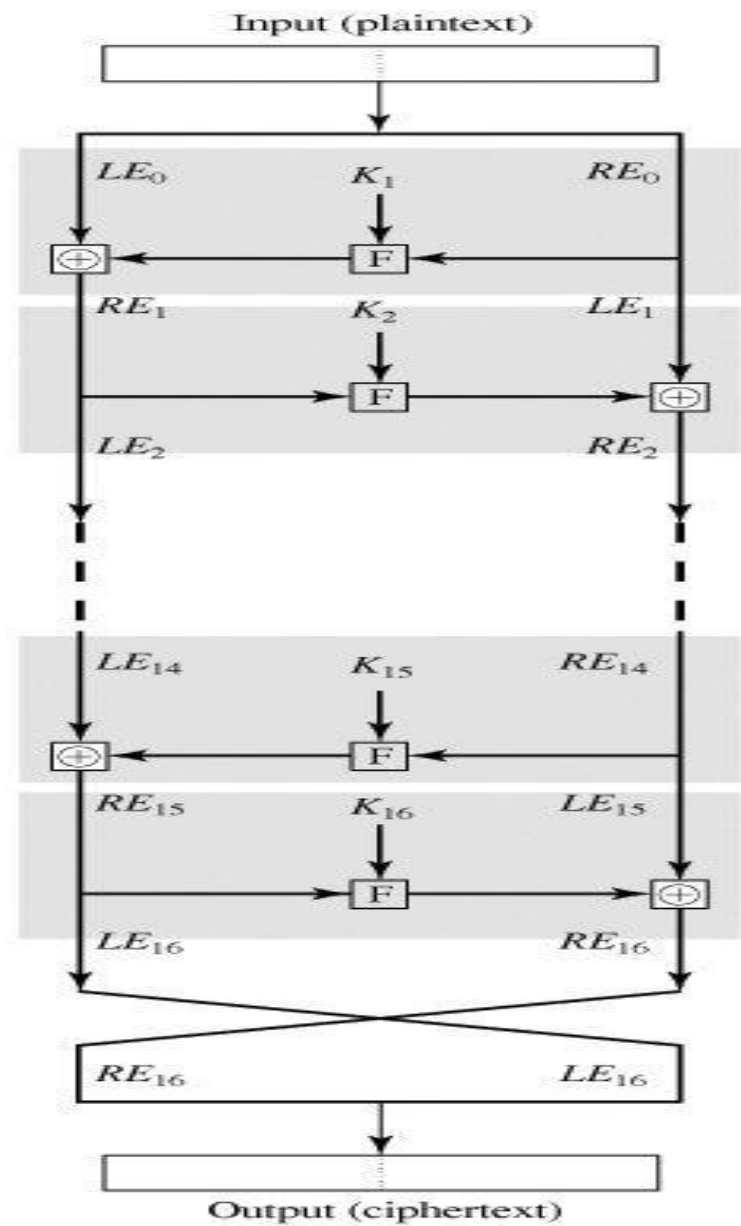
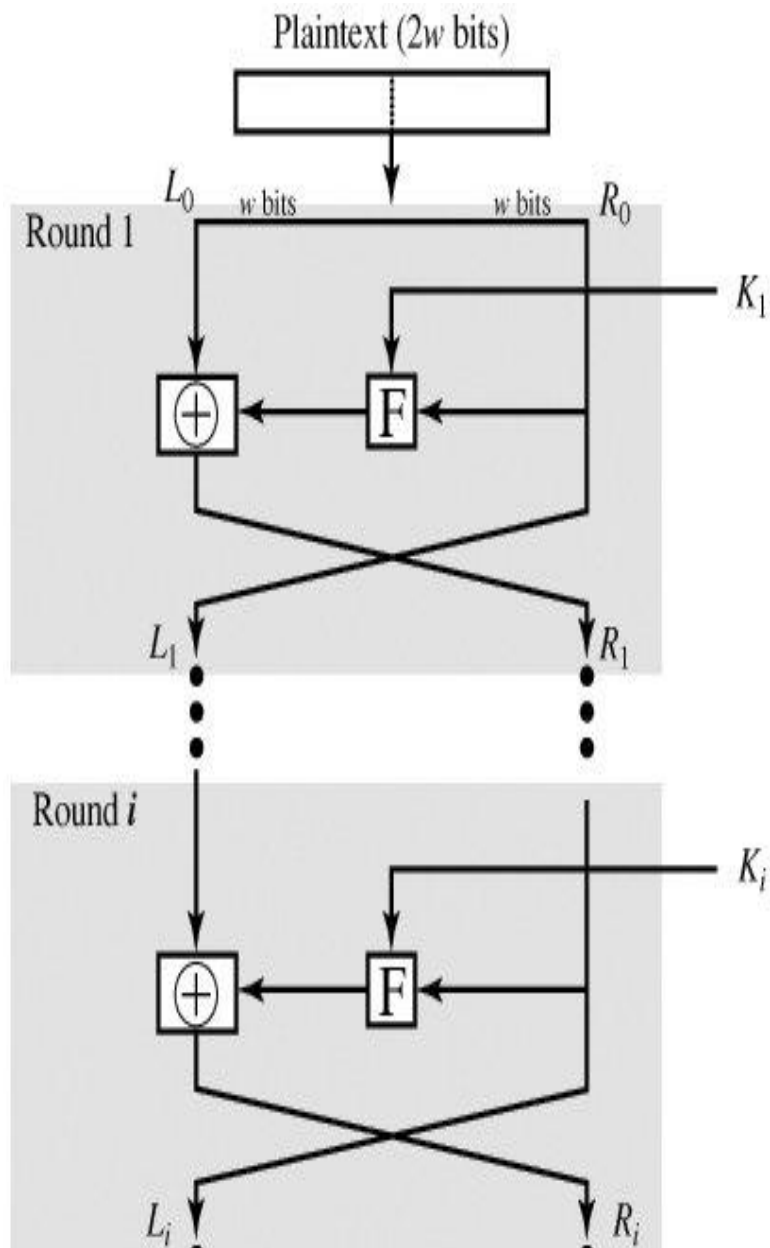
- Substitution performed to left half
 - apply round function F to right half
 - take XOR of output with left half
 - F is parameterized by round subkey K_i
- Permutation of left and right halves
 - interchange left and right halves (swapping)

How Feistel Cipher Scheme Works

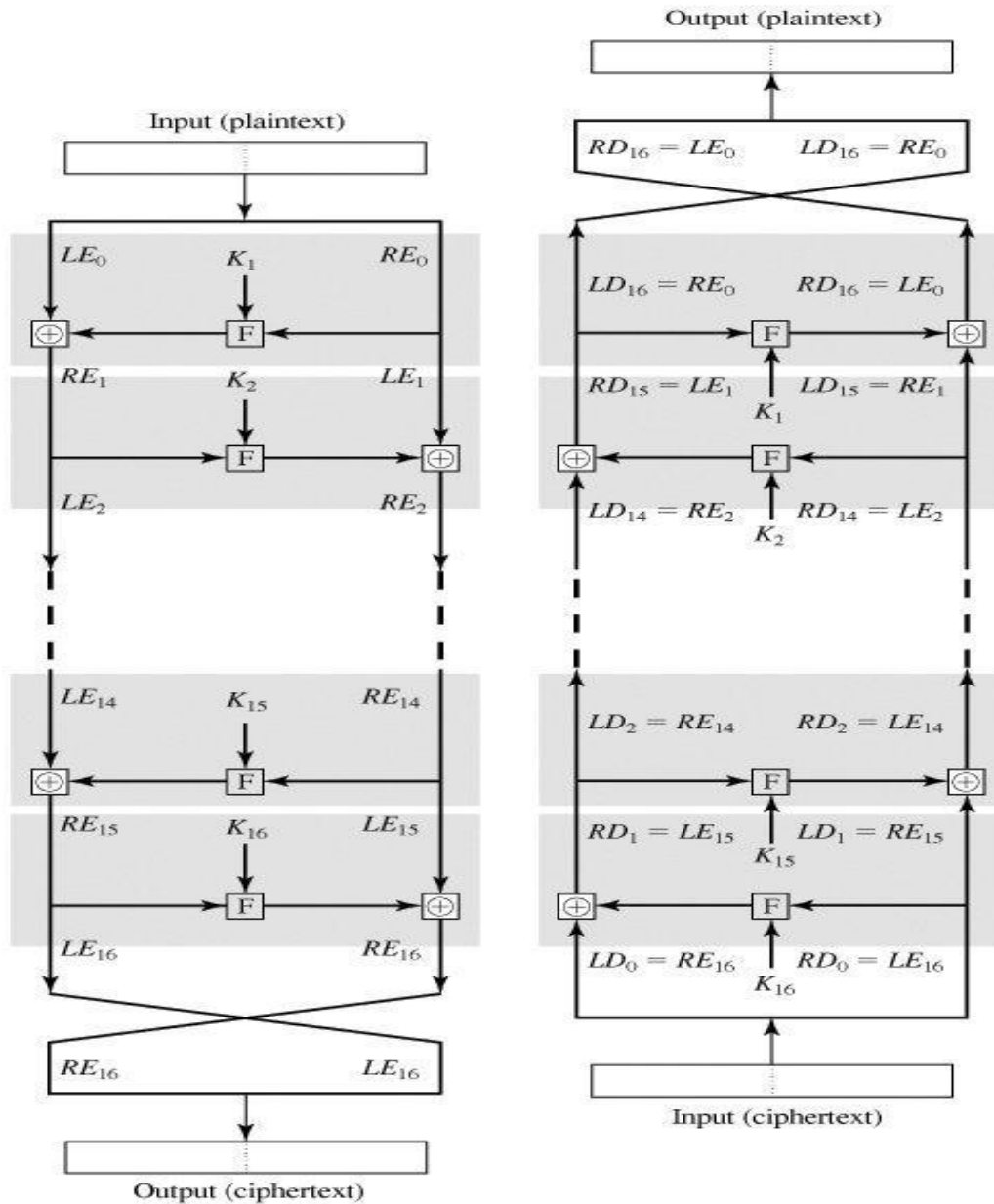
- The input to the encryption algorithm are a plaintext block of length $2w$ bits and a key k
- The plain text block is divided into two halves: L_i and R_i
- The two halves pass through **n rounds** of processing and then combine to produce the ciphertext block.
- All rounds have the same structure which involves substitution and transposition using a round function F and subkey K_i .

Fiestel Cipher

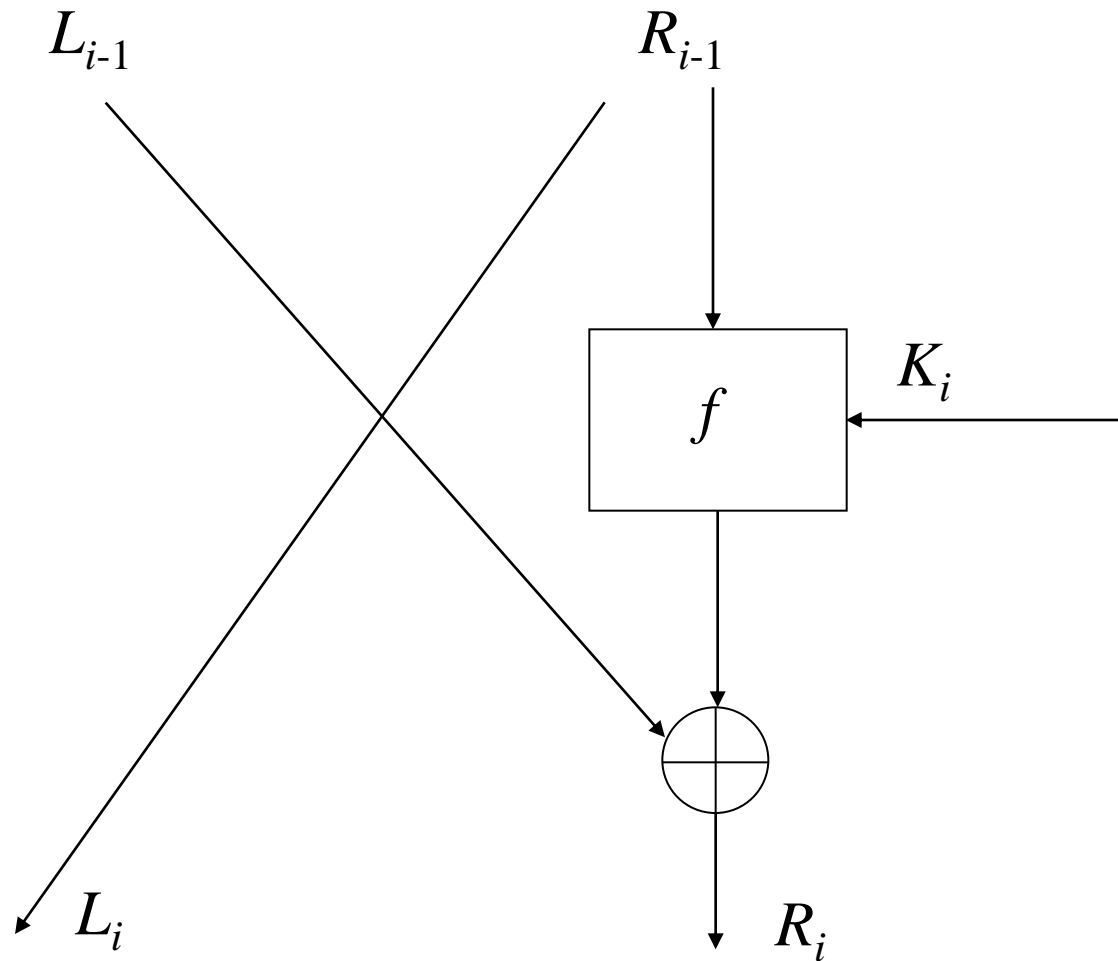
- Design Parameters
 - Block size
 - larger: greater security (diffusion)
 - smaller: faster encryption, decryption
 - typical: 64 bit, 128 bit AES
 - Key size
 - larger: greater security (brute-force resist)
 - smaller: faster encryption, decryption
 - typical: 128 bit
 - Number of rounds
 - multiple rounds increase security
 - typical: 16
 - Round function
 - complexity makes cryptanalysis difficult



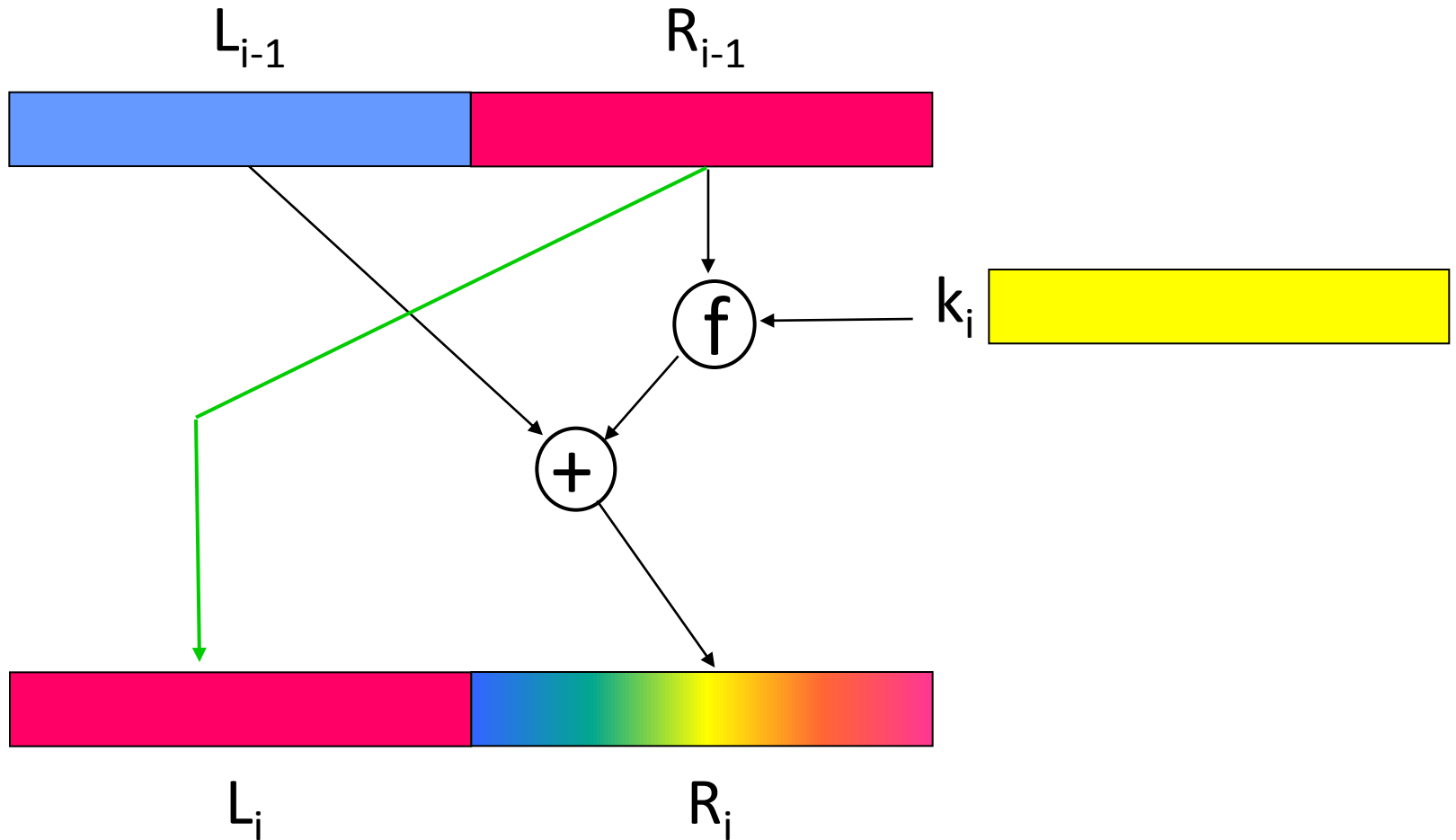
Feistel Cipher



One round of a Feistel system



Round i



Fiestel Cipher Decryption Algorithm

- Ciphertext is used as input
- Use subkeys K_i in reverse order
- Same algorithm is used
- Notation
 - LE_i : left half in encryption algorithm
 - RE_i : right half in encryption algorithm
 - LD_i : left half in decryption algorithm
 - RD_i : right half in decryption algorithm
- Output of i th encryption round input to $(16-i)$ th decryption round swapped

Fiestel Cipher Decryption Proof

- Encryption side

$$LE16 = RE15$$

$$RE16 = LE15 \oplus F(RE15, K16)$$

- Decryption side

$$LD1 = RD0 = LE16 = RE15$$

$$LD0 = RE16$$

$$RD1 = LD0 \oplus F(RD0, K16)$$

$$= RE16 \oplus F(RE15, K16)$$

$$= [LE15 \oplus F(RE15, K16)] \oplus F(RE15, K16)$$

Data Encryption Standard (DES)

- In 1974, IBM submitted an algorithm called LUCIFER for the National Bureau of Standards (NBS).
- The NBS forwarded it to the National Security Agency (NSA) , which reviewed it, and returned a version called the **Data Encryption Standard (DES) algorithm**.
- In 1977, NBS made it the official data encryption standard for use on all unclassified government communications.
- This was probably the result of a misunderstanding between NSA and NBS.
 - The NSA thought DES was hardware-only.
- But NBS published enough details so that people could write DES software.

- DES is a block cipher; it encrypts data in 64-bit blocks.
- A 64-bit block of plaintext goes in one end of the algorithm and a 64-bit block of Ciphertext comes out the other end.
- DES is a symmetric algorithm.
- The actual mechanics of how Encryption/Decryption is done is often called a Feistel system.
- DES uses 56-bit key.
- The overall procedure can be given as

$$P^{-1}\{F[P(X)]\}$$

where, $X \rightarrow$ plaintext

$P \rightarrow$ certain permutation

$F \rightarrow$ certain transposition & substitution

F is obtained by cascading a certain function f , with each stage of cascade referred as around.

- There are 16 rounds of operations.

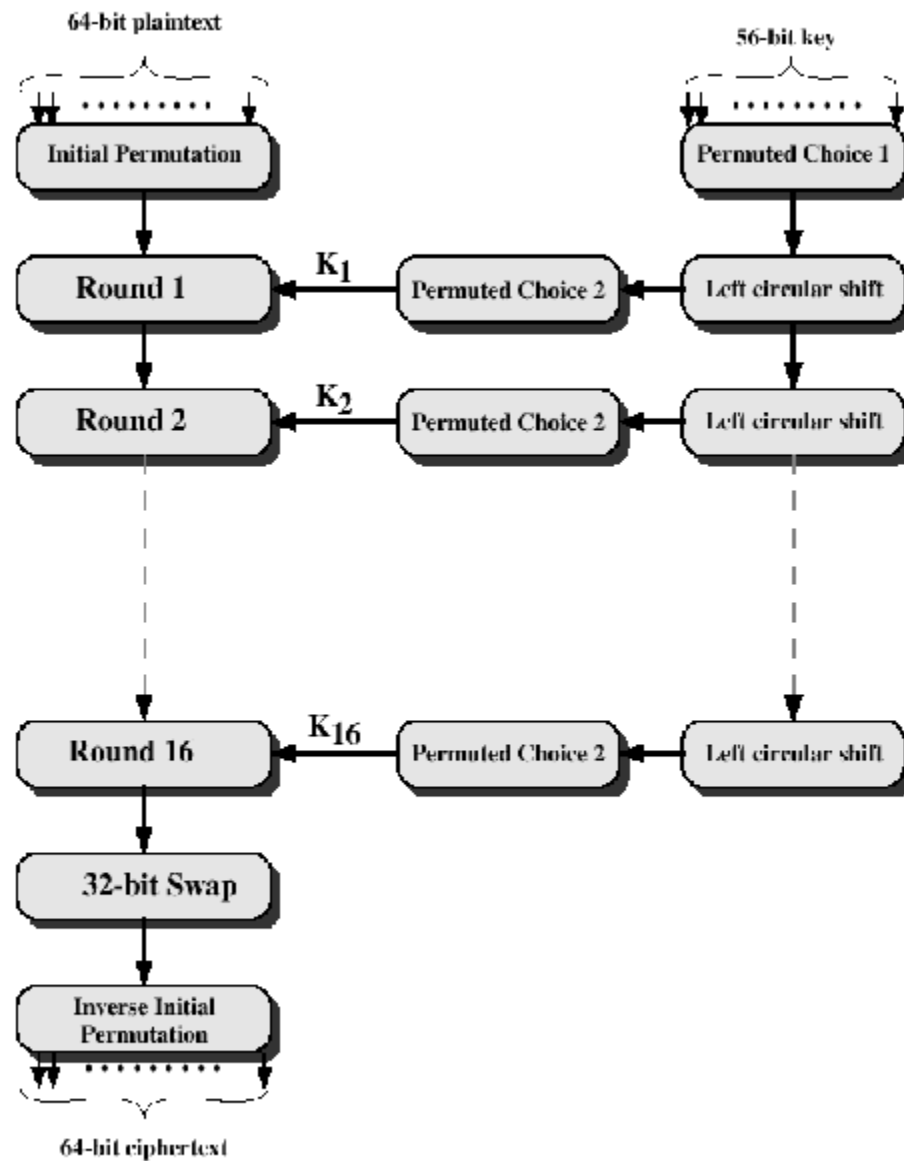
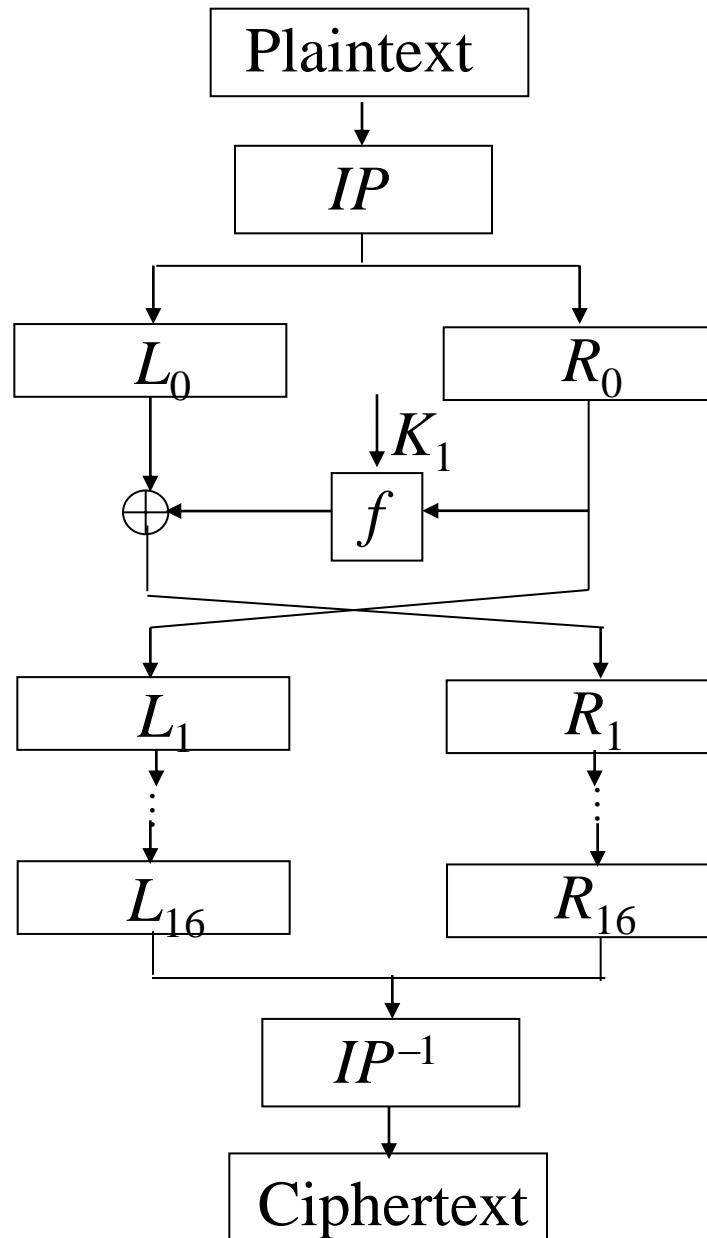


Figure 2.3 General Depiction of DES Encryption Algorithm

Description of DES Algorithm



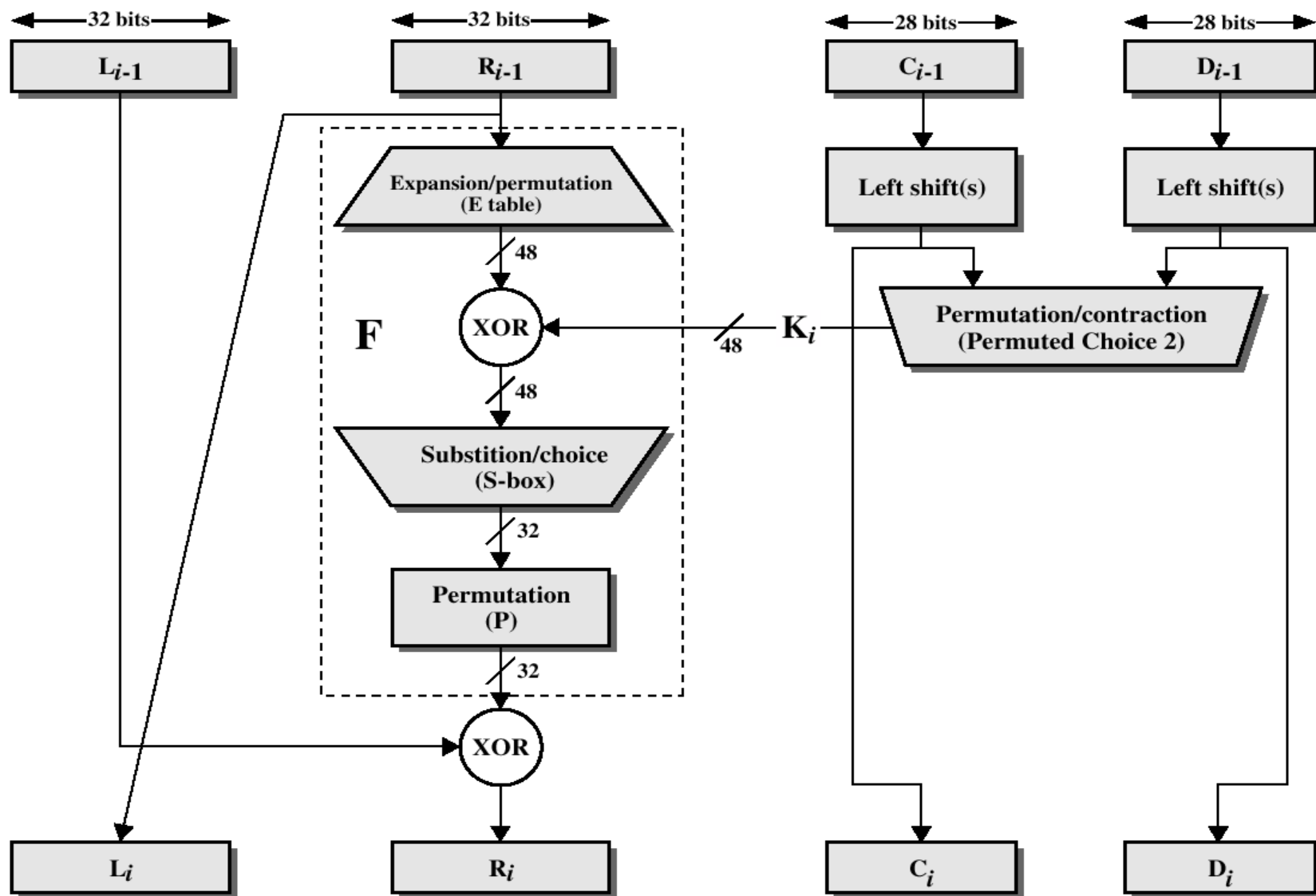


Figure 2.4 Single Round of DES Algorithm

How DES works?

- DES operates on 64-bit of data. Each block of 64 bits is divided into two blocks of **32 bits** each, a left half block **L** and a right half **R**.
 - **M** = 0123456789ABCDEF
 - **M** = 0000 0001 0010 0011 0100 0101 0110 0111
1000 1001 1010 1011 1100 1101 1110 1111
 - **L** = 0000 0001 0010 0011 0100 0101 0110 0111
 - **R** = 1000 1001 1010 1011 1100 1101 1110 1111

Encoding Data - *Initial Permutation*

- There is an *initial permutation*, **IP** of the 64 bits of the message data, **M**. This rearranges the bits according to the following table.

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

M = 0000 0001 0010 0011 0100 0101 0110 0111

1000 1001 1010 1011 1100 1101 1110 1111

IP = 1100 1100 0000 0000 1100 1100 1111 1111

1111 0000 1010 1010 1111 0000 1010 1010

- Next divide the permuted block **IP** into a left half **L₀** of 32 bits, and a right half **R₀** of 32bits.

L₀ = 1100 1100 0000 0000 1100 1100 1111 1111

R₀ = 1111 0000 1010 1010 1111 0000 1010 1010

Key Generation (K1....K16)

- 64-bit key is used as input to the algorithm
- Every eighth bit is ignored
- The key is first subjected to a permutation
Permuted Choice One table
- Resulting 56-bit key – treated as two 28-bit quantities labeled C0 and D0

Key Computation

- The 64-bit key is permuted according to the following table & 56-bit key is calculated from it.

Permuted Choice One table

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	25	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

LET

K = 00010011 00110100 01010111 01111001
10011011 10111100 11011111 11110001

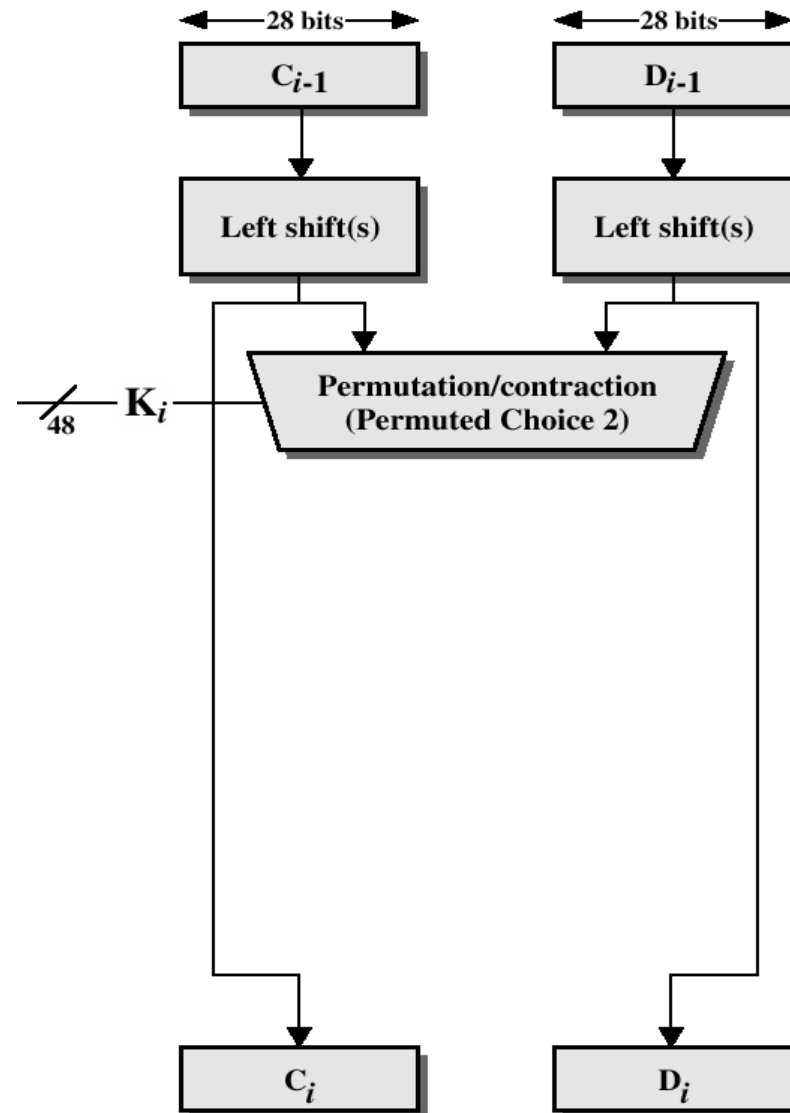
The 56-bit permutation:

K+ = 1111000 0110011 0010101 0101111
0101010 1011001 1001111 0001111

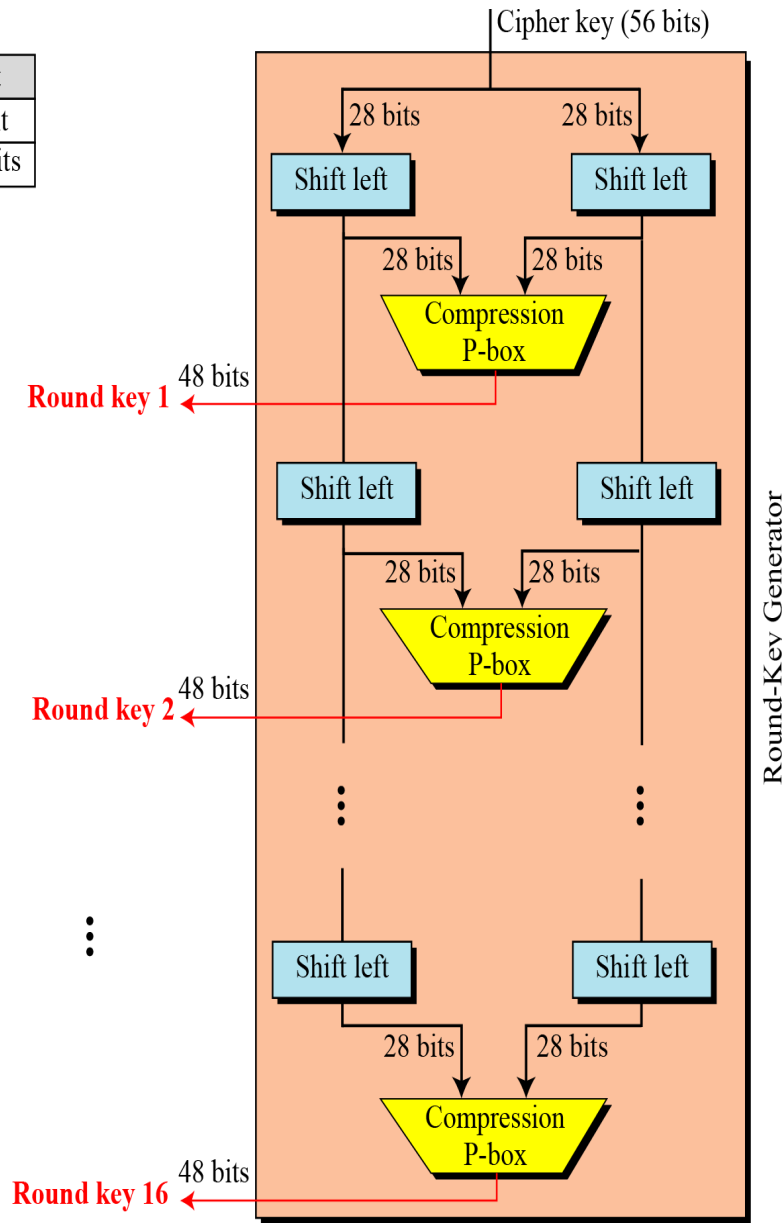
From the permuted key **K+**, we get

C₀ = 1111000 0110011 0010101 0101111
D₀ = 0101010 1011001 1001111 0001111

$(C_1, D_1) \dots (C_{16}, D_{16})$ - for each round



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



Key generation

Key Computation ...

- With \mathbf{C}_0 and \mathbf{D}_0 defined, we now create sixteen blocks \mathbf{C}_n and \mathbf{D}_n , $1 \leq i \leq 16$. Each pair of blocks \mathbf{C}_i and \mathbf{D}_i is formed from the previous pair \mathbf{C}_{i-1} and \mathbf{D}_{i-1} , respectively, for $i = 1, 2, \dots, 16$, using the following schedule of "left shifts" of the previous block.

Iteration Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Number of Left Shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

$\mathbf{C}_0 = 1111000011001100101010101111$

$\mathbf{D}_0 = 0101010101100110011110001111$

$\mathbf{C}_1 = 111000011001100101010101011111$

$\mathbf{D}_1 = 1010101011001100111100011110$

$\mathbf{C}_2 = 110000110011001010101010111111$

$\mathbf{D}_2 = 0101010110011001111000111101$

and so on upto \mathbf{C}_{16} & \mathbf{D}_{16} .

Key Computation...

- We now form the keys K_i , for $1 \leq i \leq 16$, by applying the following permutation table to each of the concatenated pairs $C_n D_n$.

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

Thus the 16, 48-bit subkeys are obtained.

$$C_1 D_1 = 1110000 \ 1100110 \ 0101010 \ 1011111 \\ 1010101 \ 0110011 \ 0011110 \ 0011110$$

$$K_1 = 000110 \ 110000 \ 001011 \ 101111 \\ 111111 \ 000111 \ 000001 \ 110010$$

Similarly,

$$K_2 = 011110 \ 011010 \ 111011 \ 011001 \\ 110110 \ 111100 \ 100111 \ 100101$$

$$K_3 = 010101 \ 011111 \ 110010 \ 001010 \\ 010000 \ 101100 \ 111110 \ 011001$$

and so on upto K_{16} .

Encoding Data ...

- We now proceed through 16 iterations, for $1 \leq i \leq 16$, using a function, F which operates on two blocks - a data block of 32 bits and a key K_i of 48 bits - to produce a block of 32 bits.

$$\begin{array}{l} L_i = R_{i-1} \\ R_i = L_{i-1} \oplus F(R_{i-1}, K_i) \end{array}$$

$$L_0 = 1100\ 1100\ 0000\ 0000\ 1100\ 1100\ 1111\ 1111$$

$$R_0 = 1111\ 0000\ 1010\ 1010\ 1111\ 0000\ 1010\ 1010$$

For $n = 1$, we have

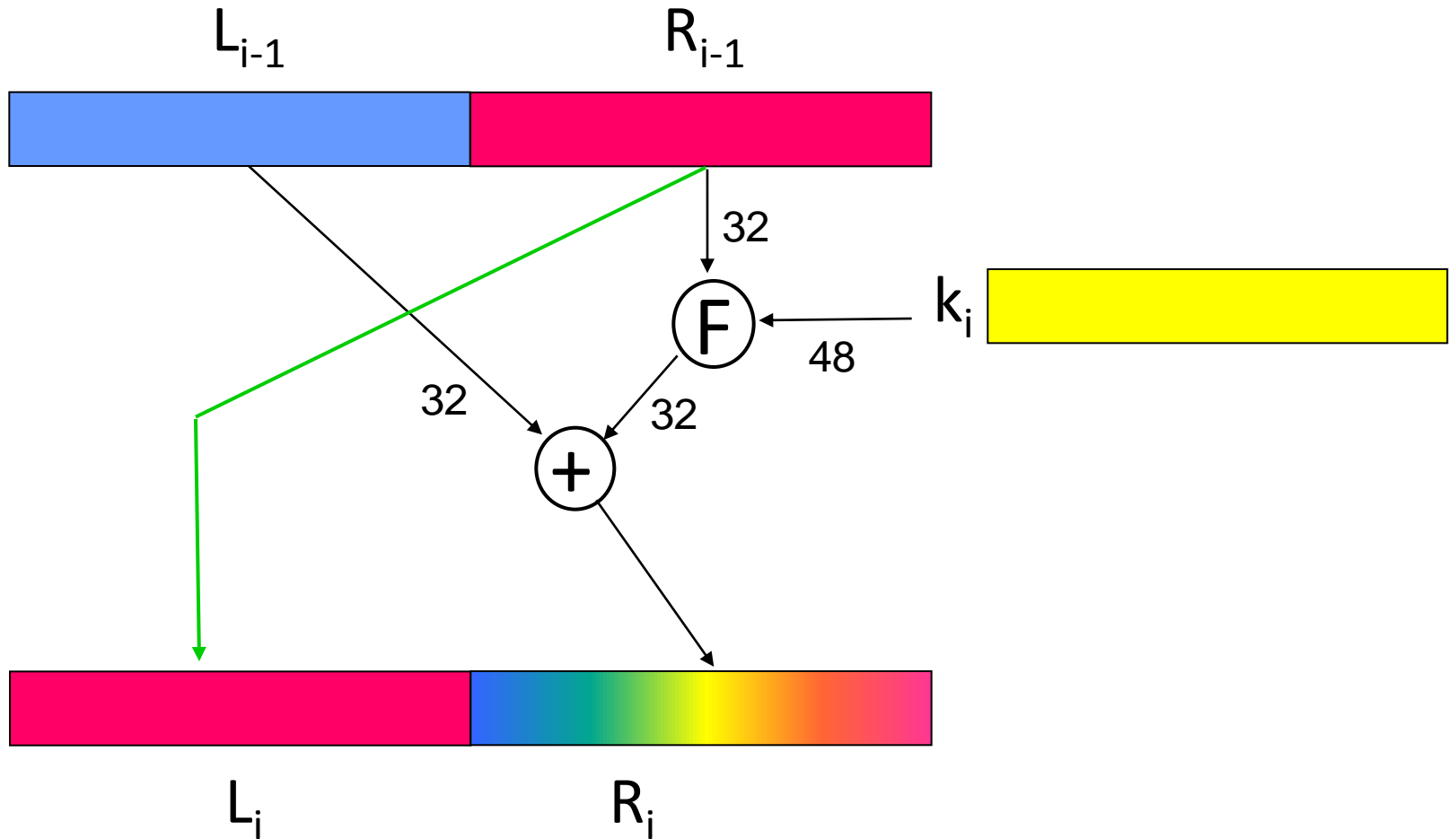
$$K_1 = 000110\ 110000\ 001011\ 101111\ 111111\ 000111\ 000001\ 110010$$

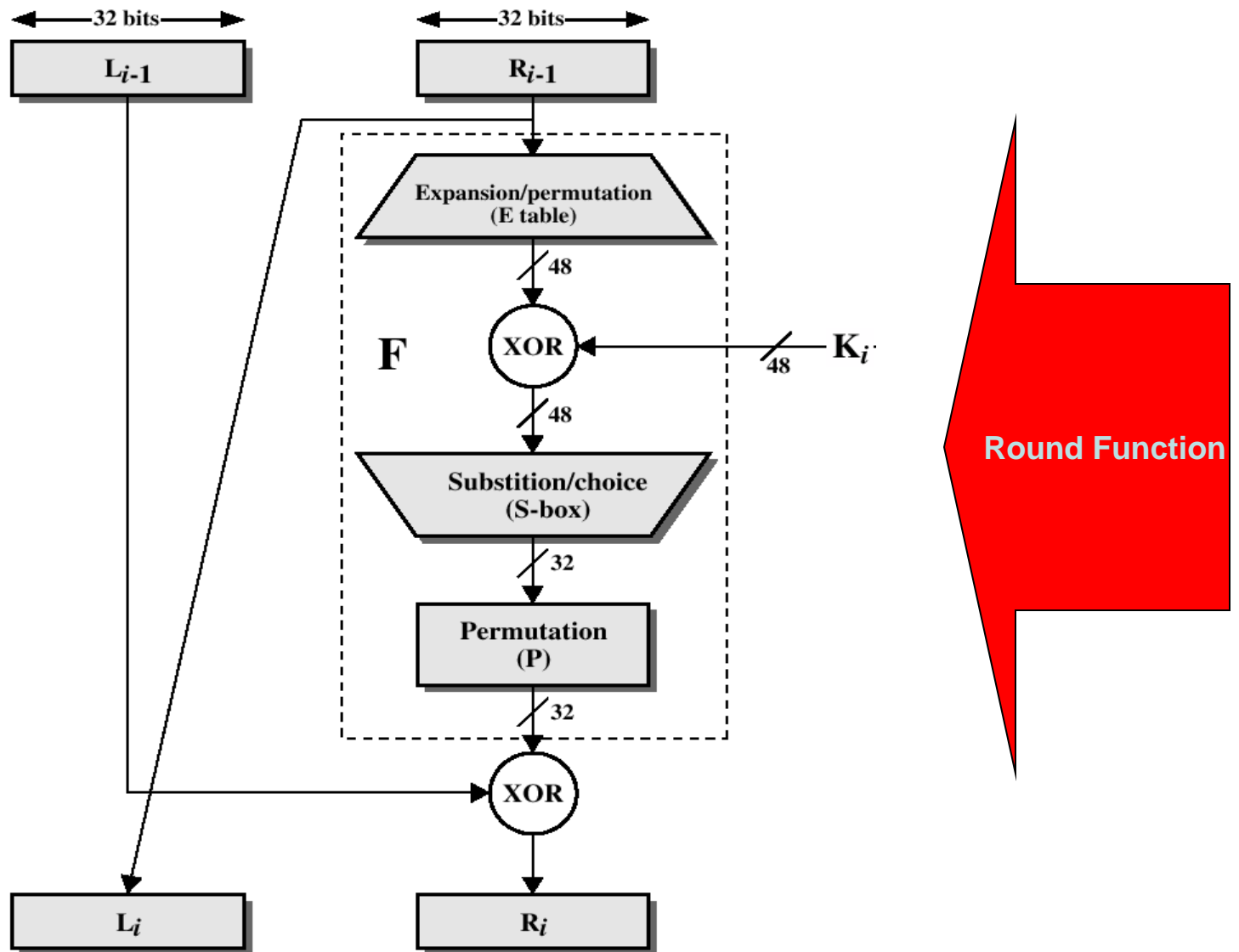
$$L_1 = R_0 = 1111\ 0000\ 1010\ 1010\ 1111\ 0000\ 1010\ 1010$$

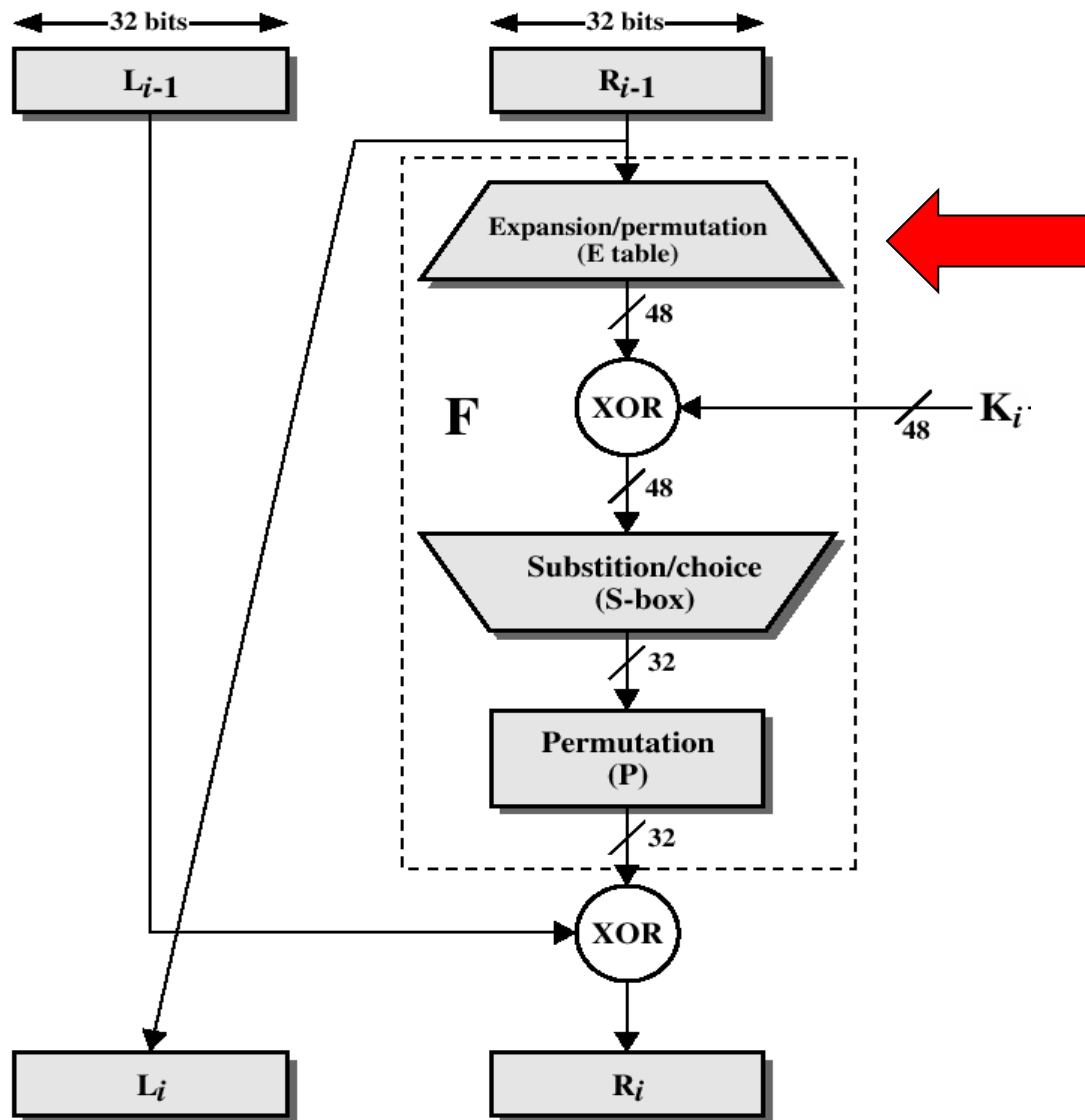
$$R_1 = L_0 \oplus f(R_0, K_1)$$

➤ It remains to explain how the function f works.

Round i of DES







Encoding Data ...

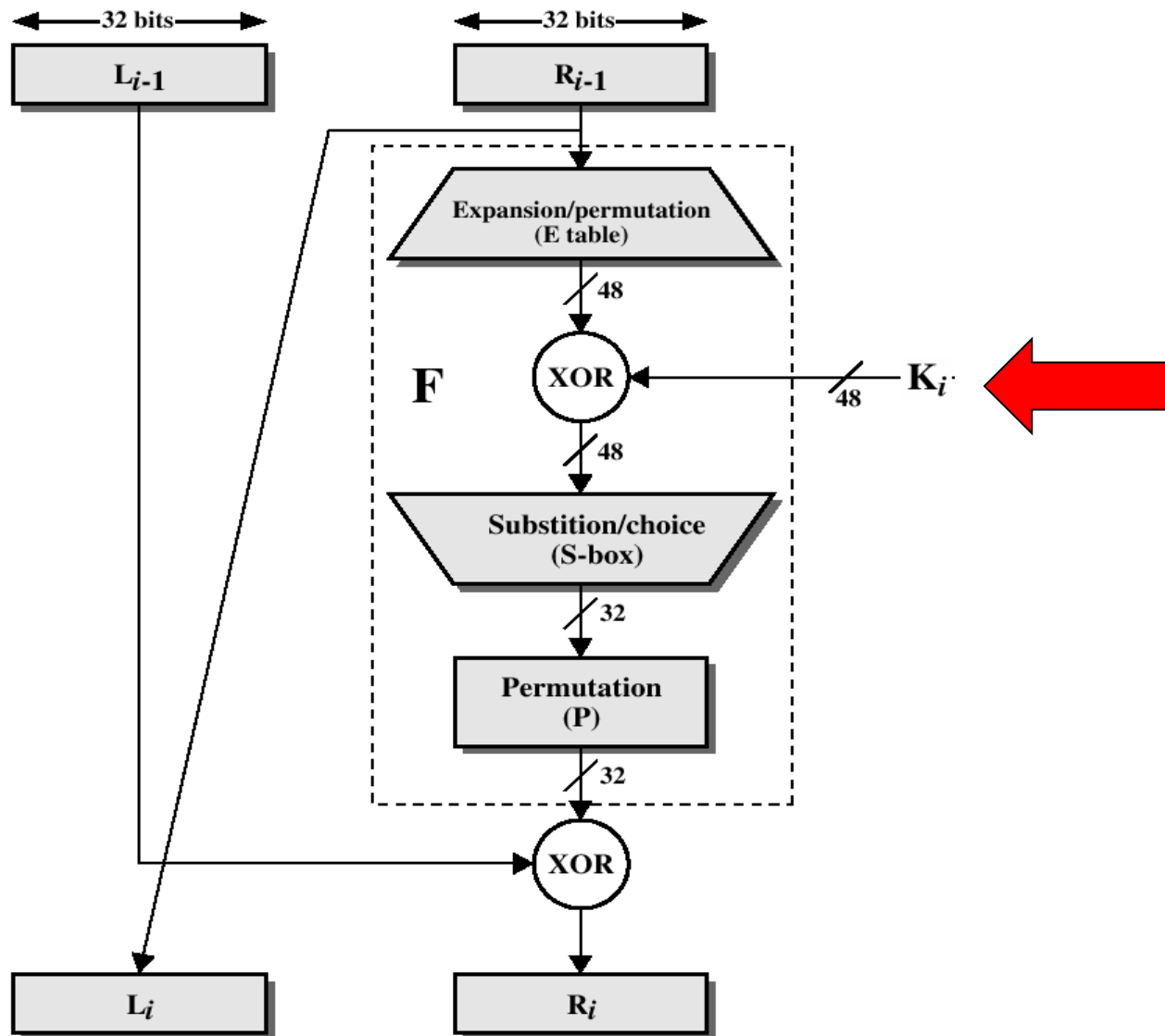
- To calculate F , we first expand each block R_{i-1} from 32 bits to 48 bits.
- This is done by using a selection table called **E-table** that repeats some of the bits in R_{i-1} .

E-table

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

We calculate $E(R_0)$ from R_0 as follows:

$$\begin{aligned} R_0 &= 1111 \quad 0000 \quad 1010 \quad 1010 \\ &\quad 1111 \quad 0000 \quad 1010 \quad 1010 \\ E(R_0) &= 011110 \ 100001 \ 010101 \ 010101 \\ &\quad 011110 \ 100001 \ 010101 \ 010101 \end{aligned}$$



Encoding Data...

- Next in the f calculation, we XOR the output $E(R_{i-1})$ with the key K_i :

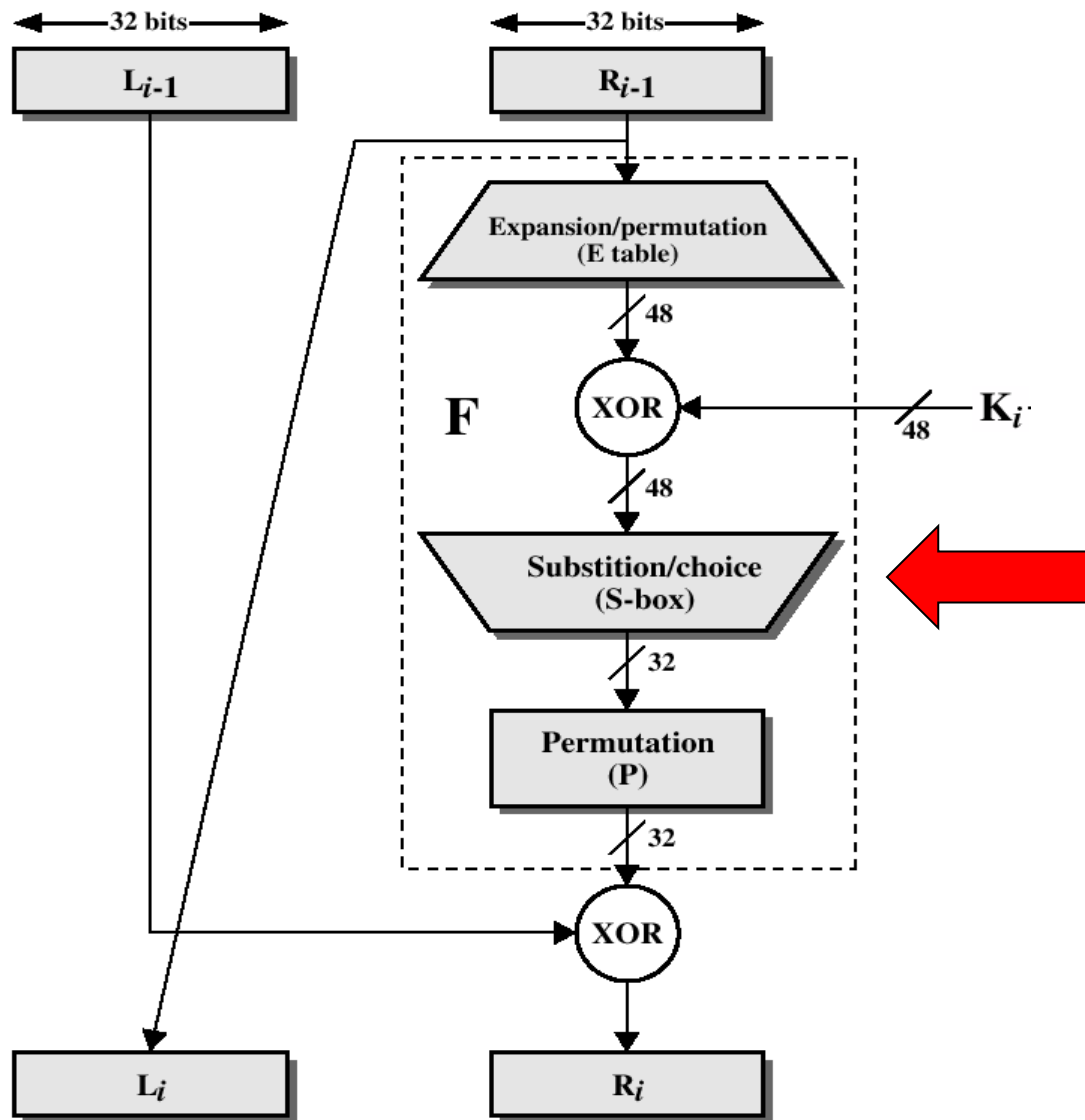
– For K_1 , $E(R_0)$, we have

$$K_i \oplus E(R_{i-1})$$

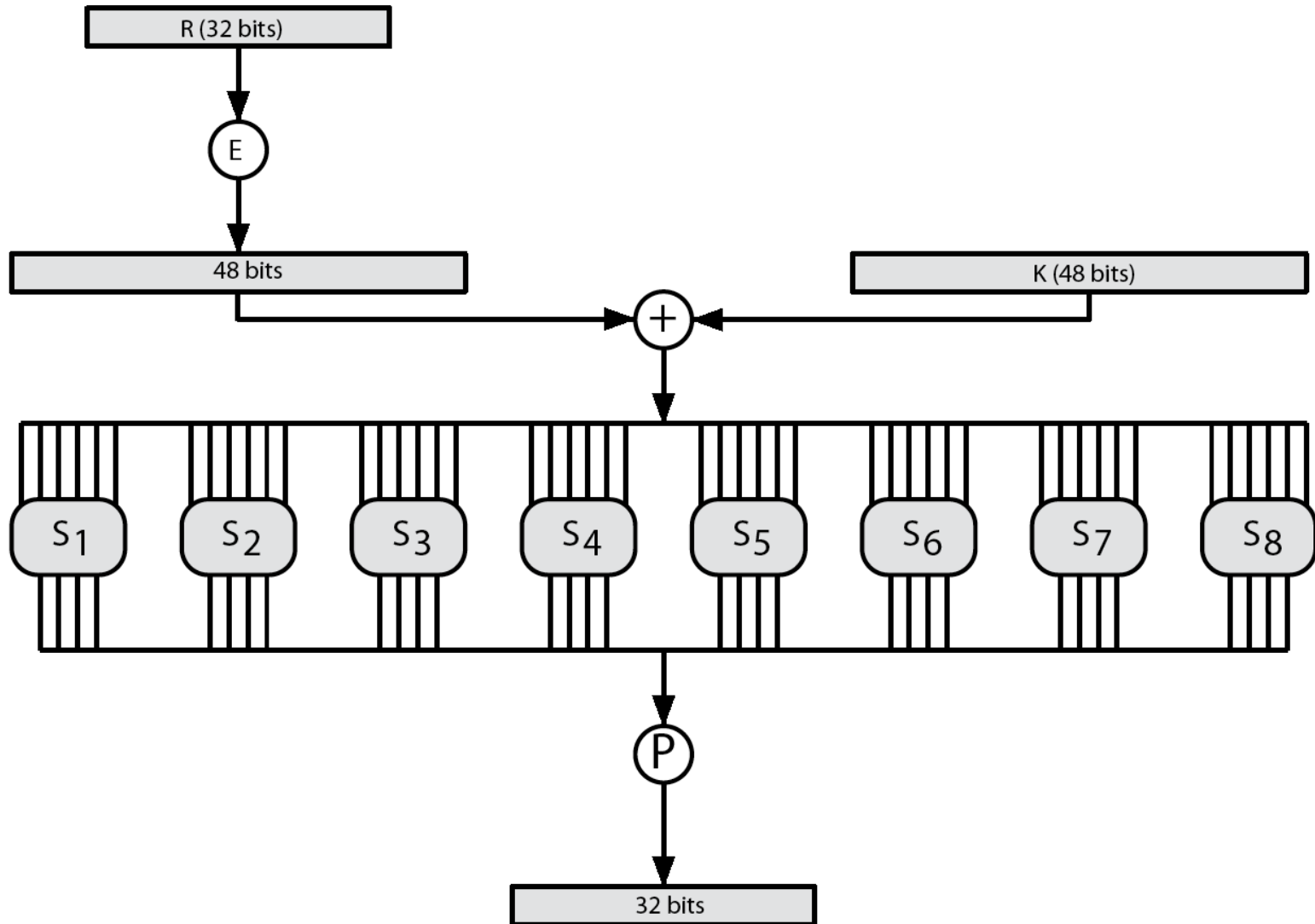
$K_1 = 000110\ 110000\ 001011\ 101111\ 111111\ 000111\ 000001\ 110010$

$E(R_0) = 011110\ 100001\ 010101\ 010101\ 011110\ 100001\ 010101\ 010101$

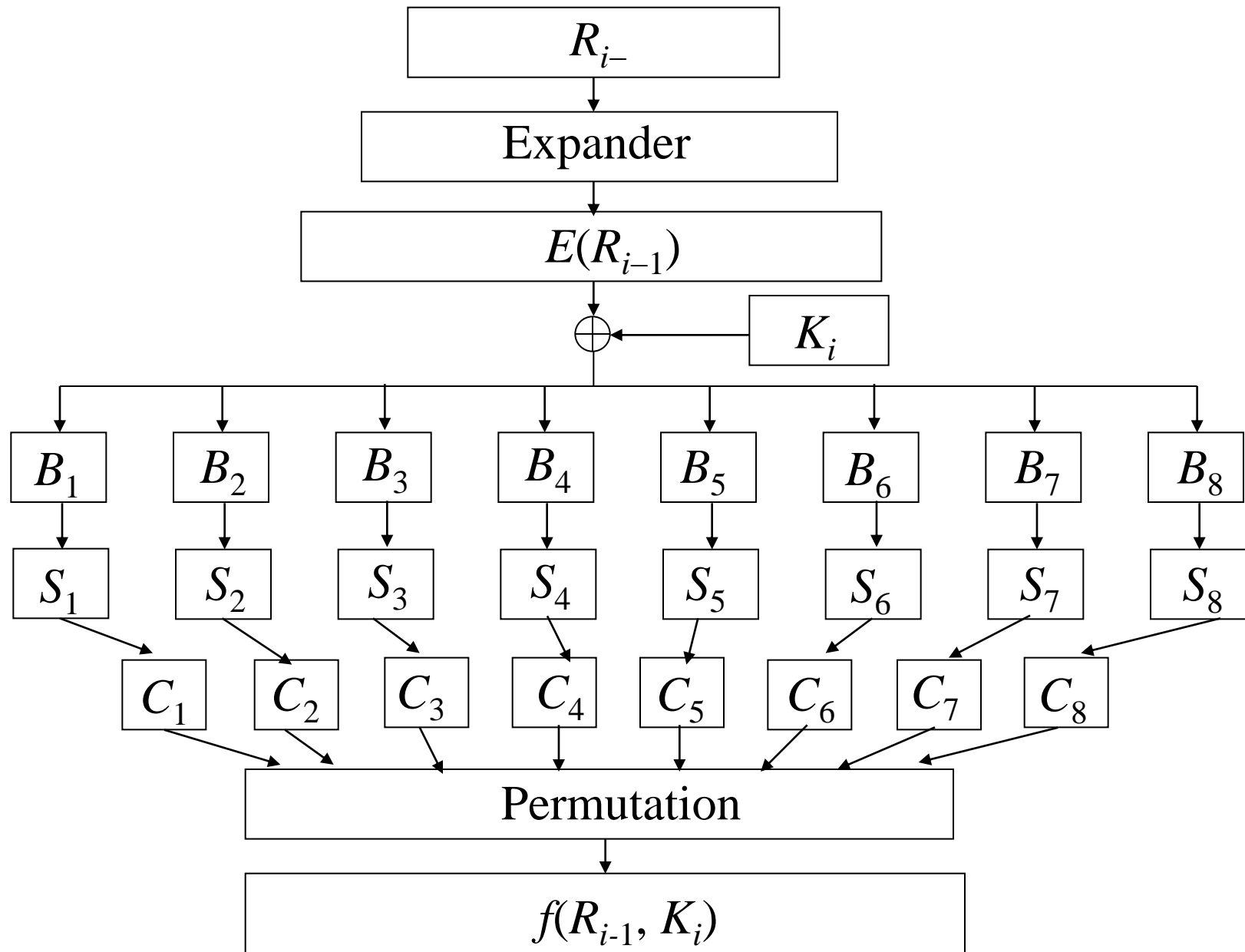
$K_1 \oplus E(R_0) = 011000\ 010001\ 011110\ 111010\ 100001\ 100110\ 010100$
 100111



The *F* function of DES



The Function $f(R_{i-1}, K_i)$



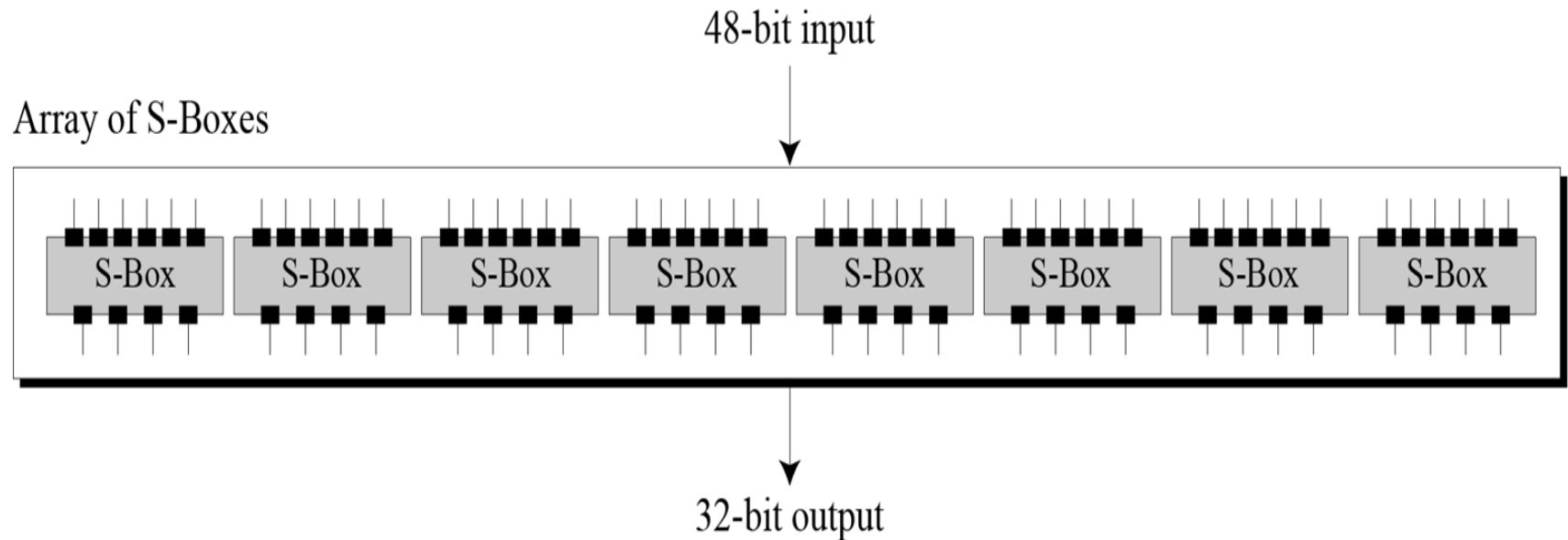
The S-Boxes

- There are eight S-boxes.
- Each S-box is specified as a 4 x 16 table.
 - each row is a permutation of 0-15
 - outer bits 1 & 6 of input are used to select one of the four rows
 - inner 4 bits of input are used to select a column
- All the eight boxes are different.
- The net result is that the eight groups of 6 bits are transformed into eight groups of 4 bits (the 4-bit outputs from the S boxes) for 32 bits total.

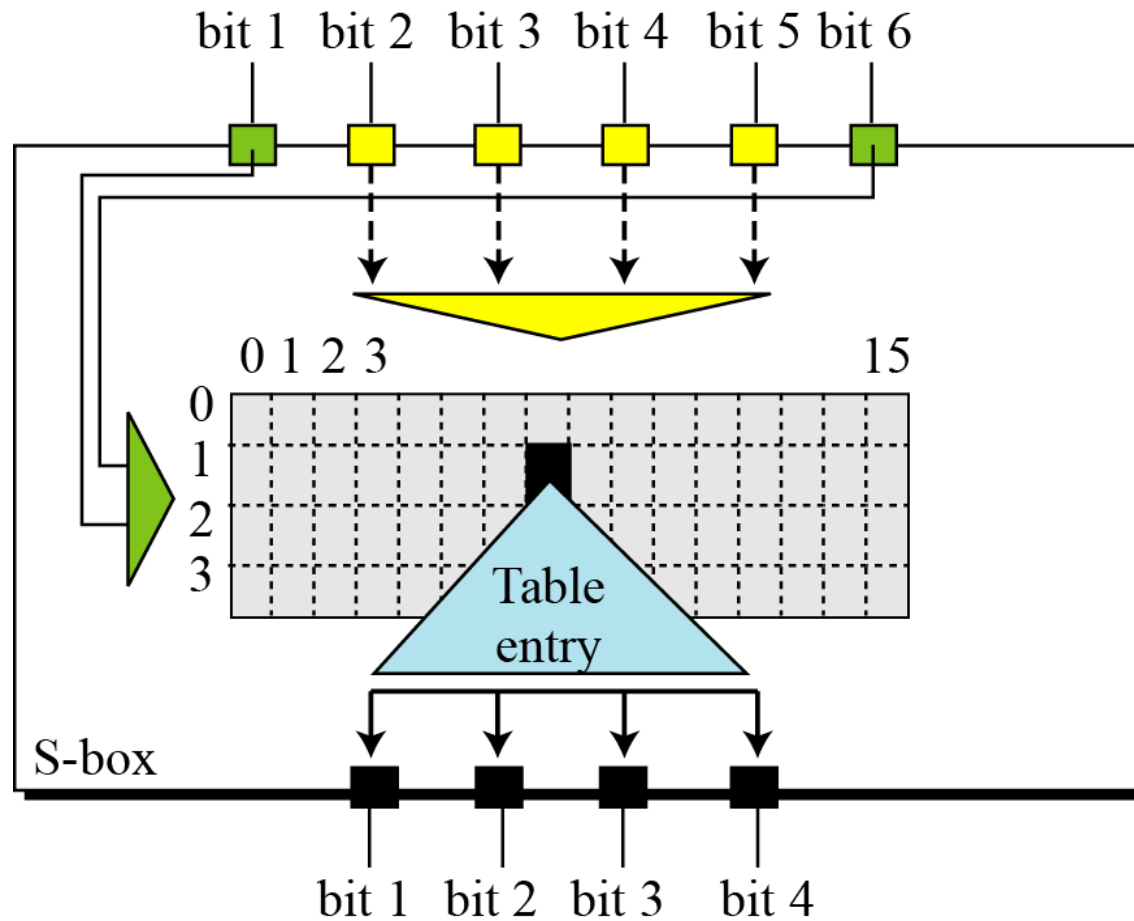
S-Boxes

The S-boxes do the real mixing (confusion).

8 S-boxes, each with a 6-bit input and a 4-bit output.



S-box rule



Encoding Data...

$$K1 \oplus E(R0) = \mathbf{011000\ 010001\ 011110\ 111010\ 100001\ 100110\ 010100\ 100111}$$

S1 S2 S3 S4 S5 S6 S7 S8

S₁ Box Column number

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	3	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Row number

Here $S_1(011011) = 0101$

Similarly, there exists S_1, S_2, \dots, S_8

For the first round, we obtain as the output of the **eight S boxes**:

S = 0101 1100 1000 0010 1011 0101 1001 0111

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 - Boxes (Continued)

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 - Boxes (Continued)

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

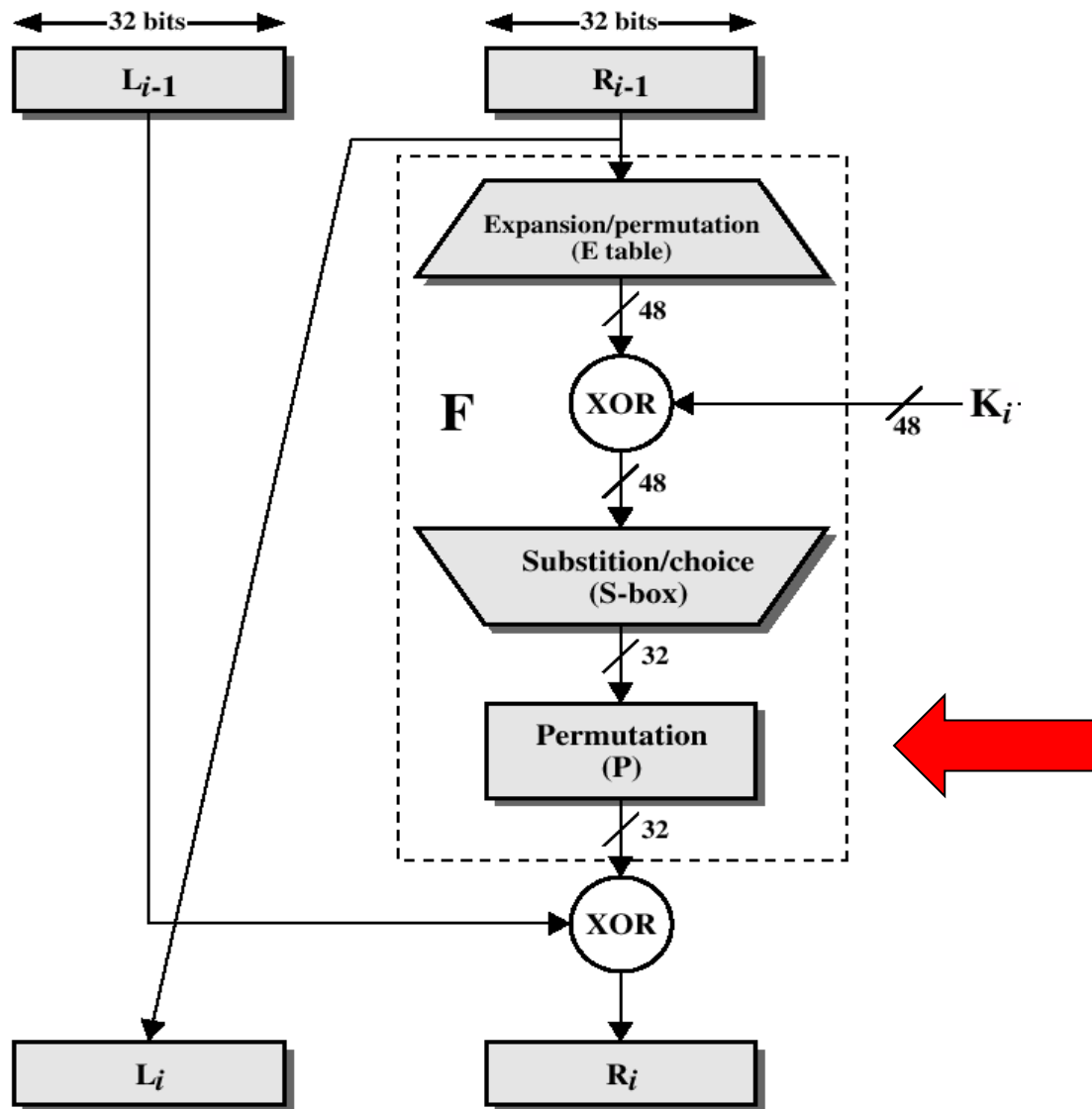
S-Boxes (Continued)

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



Encoding Data ...

- The final stage in the calculation of ***F*** is to do a permutation **P** of the **S**-box output to obtain the final value of ***f***:

$$f = P(S)$$

The permutation **P** is defined in the following table. **P** yields a 32-bit output from a 32-bit input by permuting the bits of the input block.

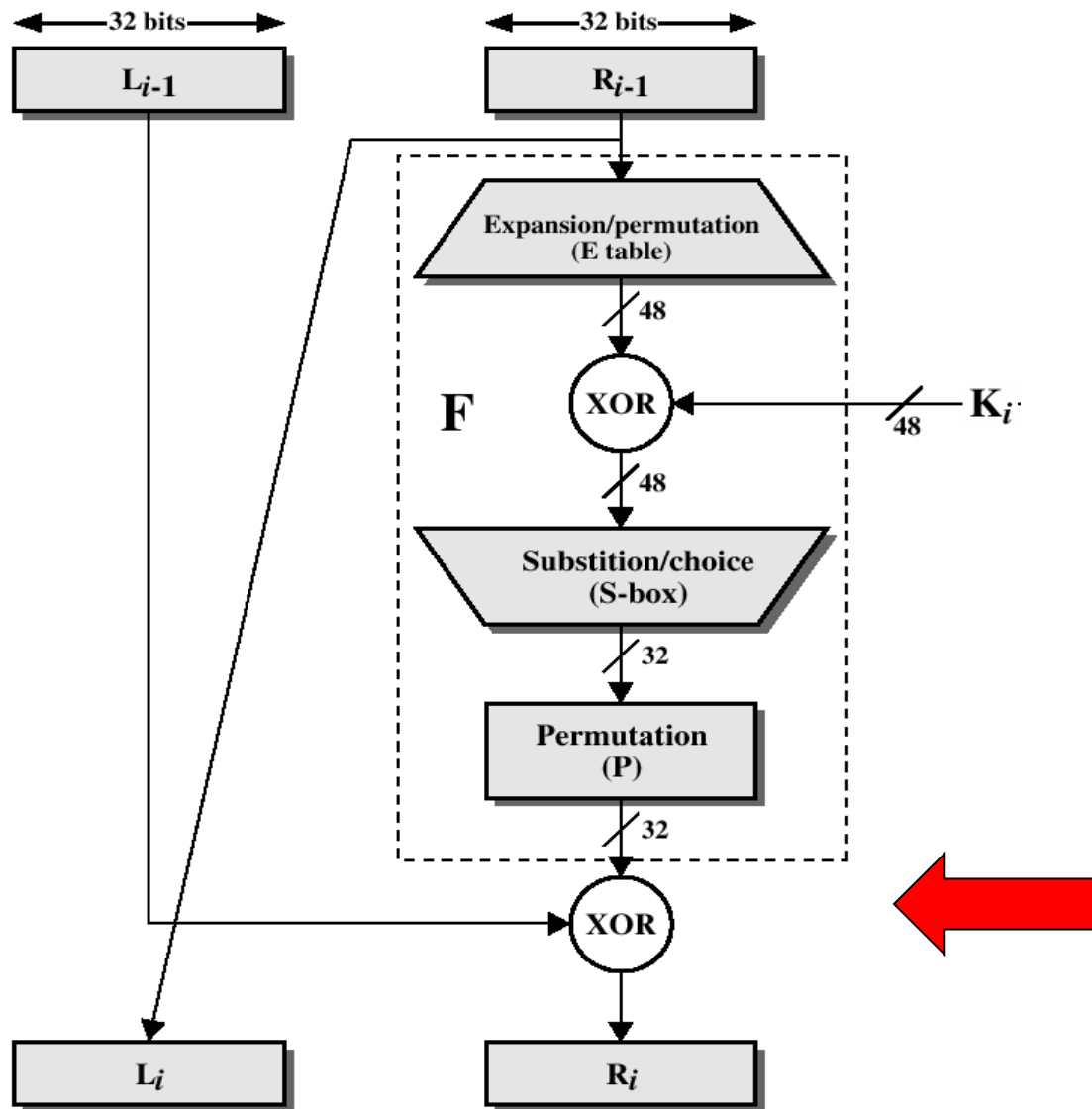
P

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

From

S = 0101 1100 1000 0010 1011 0101 1001 0111

f = 0010 0011 0100 1010 1010 1001 1011 1011

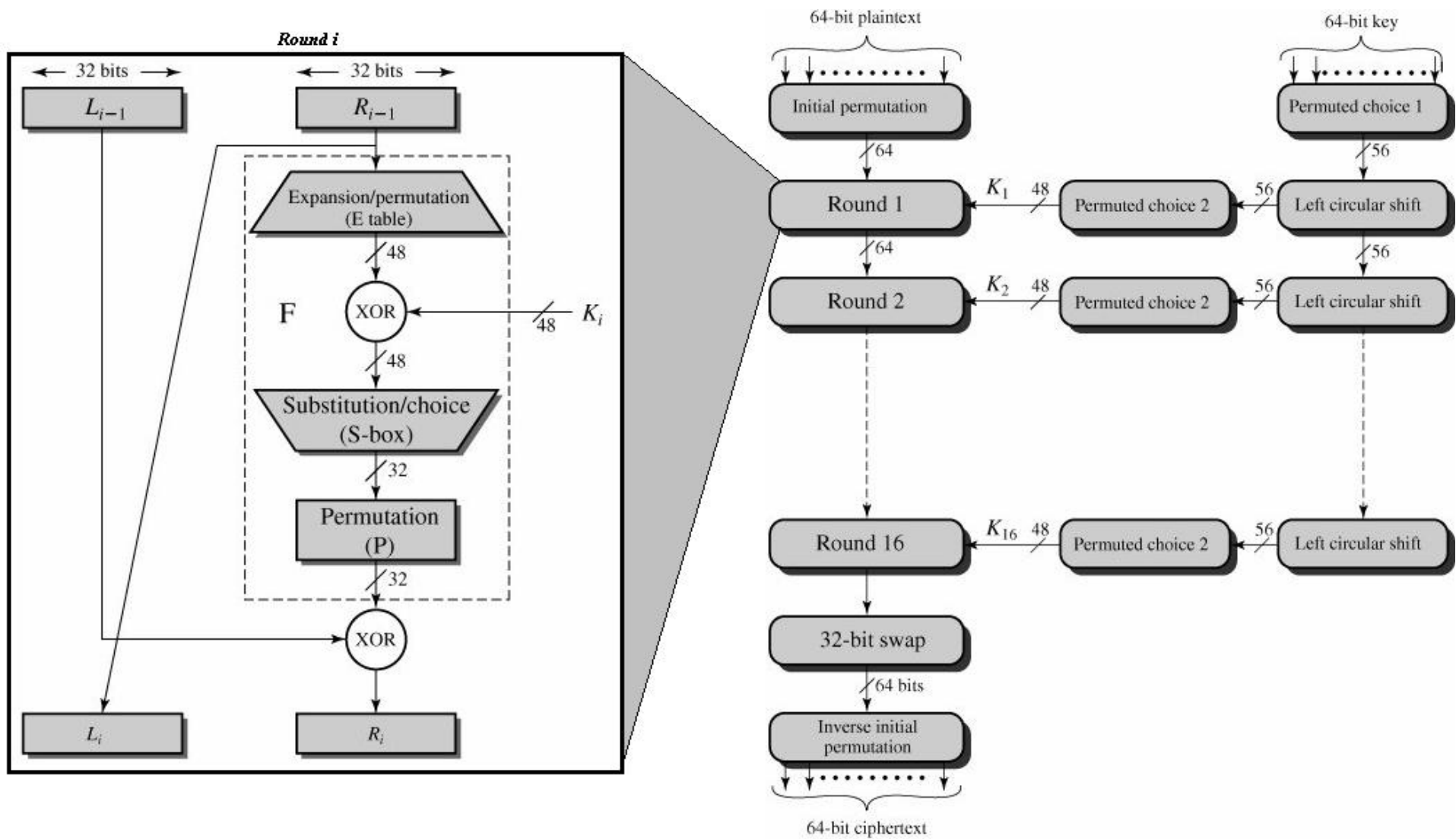


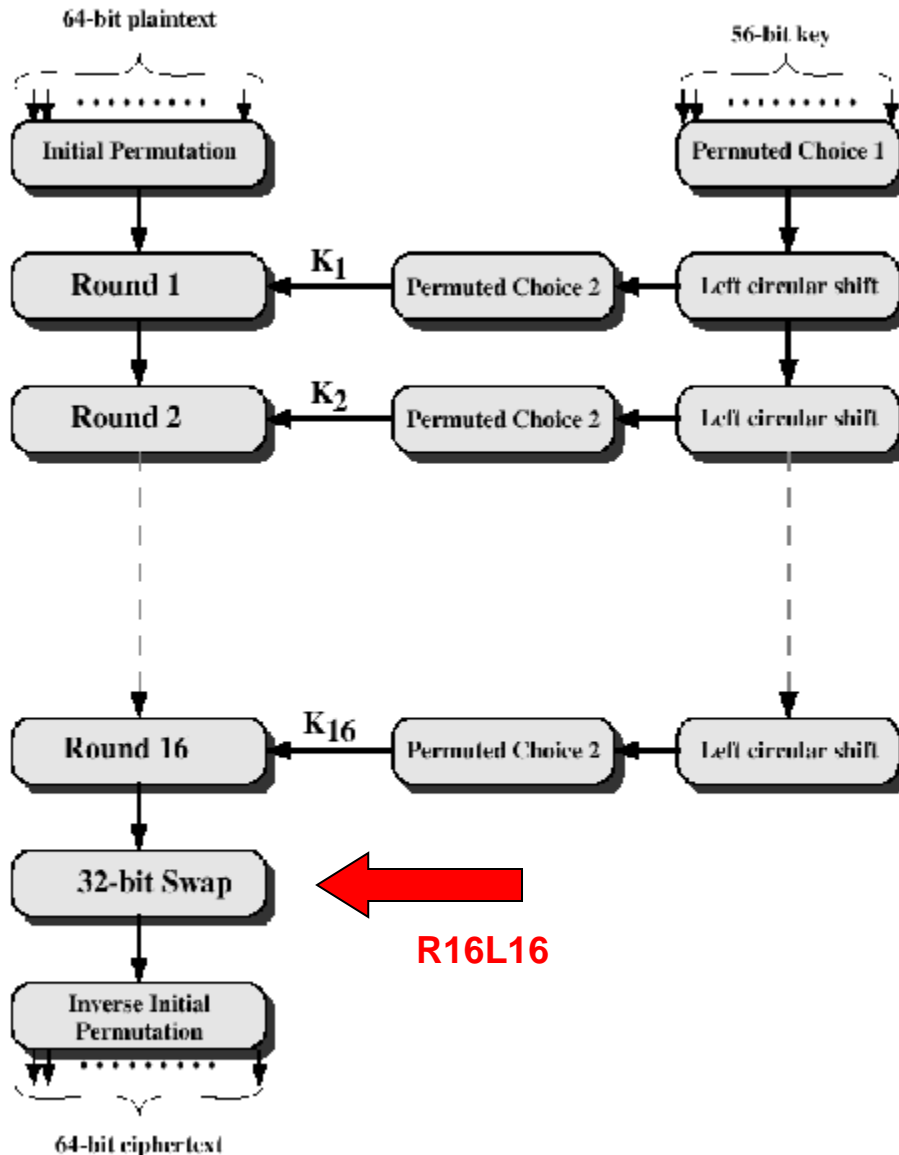
Encoding Data ...

- $R_1 = L_0 \oplus f(R_0, K_1)$

$$\begin{aligned} &= 1100\ 1100\ 0000\ 0000\ 1100\ 1100\ 1111\ 1111 \\ &\oplus 0010\ 0011\ 0100\ 1010\ 1010\ 1001\ 1011\ 1011 \\ &= 1110\ 1111\ 0100\ 1010\ 0110\ 0101\ 0100\ 0100 \end{aligned}$$

- Proceeding like this we obtain $L_1R_1, L_2R_2, \dots, L_{16}R_{16}$.
- At the end of the sixteenth round we have the blocks L_{16} and R_{16} . We then reverse the order of the two blocks into the 64-bit block $R_{16}L_{16}$ and apply a permutation IP^{-1} .





- At the end of the sixteenth round we have the blocks L16 and R16. We then reverse the order of the two blocks into the 64-bit block R16L16 and apply a permutation IP-1.

Figure 2.3 General Depiction of DES Encryption Algorithm

Encoding Data ... IP^{-1}

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

LET

$R_{16}L_{16} = 00001010\ 01001100$
 $11011001\ 10010101\ 01000011$
 $01000010\ 00110010\ 00110100$

$IP^{-1} = 1000\ 0101\ 1110\ 1000\ 0001$
 $0011\ 0101\ 0100\ 0000\ 1111\ 0000$
 $1010\ 1011\ 0100\ 0000\ 0101$
which in hexadecimal format is
 $85E813540F0AB405$.

Thus the encrypted form of $M = 0123456789ABCDEF$:

namely, $C = 85E813540F0AB405$

Decryption

As with any Feistel cipher, decryption uses the same algorithm as encryption, except that the application of the subkeys is reversed.

Avalanche Effect

- Avalanche effect:
 - A small change in the plaintext or key results in a significant change in the ciphertext.
 - This indicates a high degree of diffusion and confusion
 - Is a desirable property of any encryption algorithm
- DES exhibits a strong avalanche effect
 - Changing 1 bit in the plaintext affects 34 bits in the ciphertext on average.
 - 1-bit change in the key affects 35 bits in the ciphertext on average.

To check the avalanche effect in DES, let us encrypt two plaintext blocks (with the same key) that differ only in one bit and observe the differences in the number of bits in each round.

Plaintext: 0000000000000000

Key: 22234512987ABB23

Ciphertext: 4789FD476E82A5F1

Plaintext: 0000000000000001

Key: 22234512987ABB23

Ciphertext: 0A4ED5C15A63FEA3

Although the two plaintext blocks differ only in the rightmost bit, the ciphertext blocks differ in 29 bits. This means that changing approximately 1.5 percent of the plaintext creates a change of approximately 45 percent in the ciphertext.

Number of bit differences

<i>Rounds</i>	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>6</i>	<i>7</i>	<i>8</i>	<i>9</i>	<i>10</i>	<i>11</i>	<i>12</i>	<i>13</i>	<i>14</i>	<i>15</i>	<i>16</i>
Bit differences	1	6	20	29	30	33	32	29	32	39	33	28	30	31	30	29

The Strength of DES

1. The Use of 56-Bit Keys

With a key length of 56 bits, there are 2^{56} possible keys, which is approximately 7.2×10^{16}

Thus a brute-force attack appears impractical. Assuming that, on average, half the key space has to be searched, a single machine performing one DES encryption per microsecond would take more than a thousand years to break the cipher.

DES is now considered to be insecure for many applications. This is chiefly due to the 56-bit key size being too small

In July 1998, when the Electronic Frontier Foundation (EFF) announced that it had broken a DES encryption using a special-purpose "DES cracker" machine that was built for less than \$250,000. The attack took less than three days.

The Strength of DES

2. The Nature of the DES Algorithm

The focus of concern has been on the eight substitution tables, or S-boxes, that are used in each iteration.

No one has so far succeeded in discovering the fatal weaknesses in the S-boxes

The Strength of Des

3. Timing Attacks

A timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts.

A timing attack exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs.

DES appears to be fairly resistant to a successful timing attack.

Triple DES

- Triple DES (3DES[1]) is the common name for the Triple Data Encryption Algorithm (TDEA or Triple DEA) block cipher, which applies the Data Encryption Standard (DES) cipher algorithm three times to each data block.
- Because of the availability of increasing computational power, the key size of the original DES cipher was becoming subject to brute force attacks
- Triple DES was designed to provide a relatively simple method of increasing the key size of DES to protect against such attacks, without designing a completely new block cipher algorithm.

3DES

- Triple DES uses a "key bundle" which comprises three DES keys, K_1 , K_2 and K_3 , each of 56 bits (excluding parity bits).
- The encryption algorithm is:
 - ciphertext = $E_{K_3}(D_{K_2}(E_{K_1}(\text{plaintext})))$ i.e., DES encrypt with K_1 , DES *decrypt* with K_2 , then DES encrypt with K_3 .
- Decryption is the reverse:
 - plaintext = $D_{K_1}(E_{K_2}(D_{K_3}(\text{ciphertext})))$ I.e., decrypt with K_3 , *encrypt* with K_2 , then decrypt with K_1 .
- Each triple encryption encrypts one block of 64 bits of data.

3DES Keying options

- The standards define three keying options:
 - Keying option 1: All three keys are independent.
 - Keying option 2: K_1 and K_2 are independent, and $K_3 = K_1$.
 - Keying option 3: All three keys are identical, i.e. $K_1 = K_2 = K_3$.
- Keying option 1 is the strongest, with $3 \times 56 = 168$ independent key bits.
- Keying option 2 provides less security, with $2 \times 56 = 112$ key bits.
- Keying option 3 is equivalent to DES, with only 56 key bits.

Private-Key Cryptography

- Traditional **private/secret/single key** cryptography uses **one** key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is **symmetric**, parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender

Public-Key Cryptography

- **Public-key/two-key/asymmetric cryptography involves the use of two keys:**
 - a public-key, which may be known by anybody, and can be used to encrypt messages, and verify signatures
 - a private-key, known only to the recipient, used to decrypt messages, and sign (create) signatures
- **is asymmetric because**
 - those who encrypt messages cannot decrypt messages

Public-Key Characteristics

- Public-Key algorithms rely on two keys with the characteristics that it is:
 - computationally infeasible to find decryption key knowing only algorithm & encryption key
 - computationally easy to en/decrypt messages when the relevant (en/decrypt) key is know

Requirements for public key cryptography

- It is computationally easy for a party B to generate a public key and private key pair.
- It is computationally easy for a sender A, knowing the public key and the message to be encrypted M to generate the corresponding cipher text
- It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message.
- It is computationally infeasible for adversary knowing the public key to determine the private key.
- It is computationally infeasible for adversary knowing the public key and a ciphertext to recover the original message.
- The two keys can be applied in either order.

Public-Key Applications

- Can classify uses into 3 categories:
 - **encryption/decryption** (provide secrecy)
 - **digital signatures** (provide authentication)
 - **key exchange** (of session keys)

Rivest-Shamir-Adleman (RSA)

- Widely used public key encryption algorithm.
- by Rivest, Shamir & Adleman of MIT in 1977
- RSA is believed to be secure if its keys have a length of at least 1024-bits

RSA

It is a block cipher based upon the fact that finding a random prime number of **large size** (e.g., 100 digit) is computationally easy, but factoring the product of two such numbers is considered computationally infeasible.

- The security of the RSA cryptosystem is based on the widely believed difficulty of factoring large numbers.
- In 1999, a 512-bit number was factored in 4 months using the following computers:
 - 160 175-400 MHz SGI and Sun
 - 8 250 MHz SGI Origin
 - 120 300-450 MHz Pentium II
 - 4 500 MHz Digital/Compaq
- Estimated resources needed to factor a number within one year

Bits	PCs	Memory
430	1	128MB
760	215,000	4GB
1,020	342×10^6	170GB
1,620	1.6×10^{15}	120TB

- A factor is a number which divides INTO another number without leaving a remainder.
 - For example, 6 is a factor of 12, but 6 is not a factor of 13.
- A prime factor is a number such as 2, 3, 5, 7, 11, 13, or 17 (to name a few) --- a number that can only be divided by itself and 1.

RSA Key Generation Algorithm

1. Generate two large prime numbers, p and q
2. Let $n = p * q$
3. Let $\phi(n) = (p-1)*(q-1)$
4. Choose an integer e , $1 < e < \phi(n)$ such that:

$$\gcd(e, \phi(n)) = 1$$

[gcd – greatest common denominator]

5. Find d , $1 < d < \phi(n)$ such that

$$ed \equiv 1 \pmod{\phi(n)} \text{ i.e. compute } d = e^{-1} \pmod{\phi(n)}$$

Extended Euclidean algorithm

To be secure, **very large numbers must be used for p and q** - 100 decimal digits at the very least.

- The public key is (e, n) and the private key (d, n)
- The values p, q and $\phi(n)$ are private
- To encrypt a plaintext message block M , compute **$C = M^e \bmod n$**
- To decrypt the block,
 - compute **$M = C^d \bmod n$**

e - public exponent (give this to others)

d - private exponent (keep this secret)

Key Generation

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p - 1)(q - 1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod n$

Decryption

Ciphertext:	C
Plaintext:	$M = C^d \pmod n$

- Knowing C and e , it is “infeasible” to calculate m without knowing d and n
- Knowing d and n , it is easy to find the original plaintext message $m = C^d \bmod n$

RSA Factoring Challenge



<http://www.rsa.com>

- The RSA Factoring Challenge was a challenge put forward by RSA Laboratories on March 18, 1991 to encourage research into computational number theory and the practical difficulty of factoring large integers and cracking RSA keys used in cryptography.
- Let n be an RSA Number. RSA Laboratories states there are **prime numbers** p and q such that $n = p \times q$, the problem is to find these two primes, given only n .

RSA Number	Decimal digits	Binary digits	Cash prize offered	Factored on	Factored by
RSA-100	100	330	\$1,000 USD	April 1, 1991	Arjen K. Lenstra
RSA-110	110	364	\$4,429 USD	April 14, 1992	Arjen K. Lenstra and M.S. Manasse
RSA-120	120	397	\$5,898	June 9, 1993	T. Denny et al.
RSA-129	129	426	\$100 USD ^[4]	April 26, 1994	Arjen K. Lenstra et al.
RSA-130	130	430	\$14,527 USD	April 10, 1996	Arjen K. Lenstra et al.
RSA-140	140	463	\$17,226 USD	February 2, 1999	Herman te Riele et al.
RSA-150 ^[5]	150	496		April 16, 2004	Kazumaro Aoki et al.
RSA-155	155	512	\$9,383	August 22, 1999	Herman te Riele et al.
RSA-160	160	530		April 1, 2003	Jens Franke et al. , University of Bonn
RSA-170	170	563		December 29, 2009	D. Bonenberger and M. Krone
RSA-576	174	576	\$10,000 USD	December 3, 2003	Jens Franke et al. , University of Bonn
RSA-180	180	596		May 8, 2010	S. A. Danilov and I. A. Popovyan, Moscow State University
RSA-190	190	629		November 8, 2010	A. Timofeev and I. A. Popovyan
RSA-640	193	640	\$20,000 USD	November 2, 2005	Jens Franke et al. , University of Bonn
RSA-200	200	663		May 9, 2005	Jens Franke et al. , University of Bonn

RSA-768 is factored!

- A six-institution research team led by T. Kleinjung has successfully factored the RSA-768 (Decimal digits:232 Binary digits:768) challenge number.
- The effort took almost 2000 2.2GHz-Opteron-CPU years
- *The factors are:*

334780716989568987860441698482126908177047
949837137685689124313889828837938780022876
14711652531743087737814467999489

and

3674604366679959042824463379962795263227915
8164343087642676032283815739666511279233373
417143396810270092798736308917

Cash prize offered: \$30,000 USD

Example: Bob chooses his public key

- He randomly chooses two primes, 59 and 67, respectively ($p = 59$, $q = 67$)
- $N = p \times q = 3953$
- $\phi(n) = (58)(66) = 3828$
- Pick a random e , less than 3828 but > 1
 - try 2669, $\gcd(2669, 3828) = 1$
- $d = e^{-1} \bmod \phi(n)$
 - $d = 1625$
- Bob's private key (d, n) is $(1625, 3953)$, so now Bob publishes his public key (e, n) as $(2669, 3953)$

Alice wants to send Bob a message, m...

- Alice has plaintext 3128 to send. She will send $E(m)$:
 - Alice encrypts with public key (e,n) or $(2669,3953)$
 - $C=E(m) = 3128^{2669} \bmod 3953 = 3541$
- Bob receives the ciphertext 3541:
 - Bob decrypts with private key (d,n) or $(1625,3953)$
 - $M= 3541^{1625} \bmod 3953 = 3128$

• $C=M^e \bmod n$

• $M=C^d \bmod n$

Euclid's Algorithm

- To find the greatest common divisor of two numbers
- If you want to find the $\gcd(x,y)$, you take the larger of the two, so if $x > y$, we would write,
 - $x = m_1 * y + r_1$, where m_1 is our first multiplier and r_1 is our first remainder
 - If $r_1 = 1$ then we have found that $\gcd(x,y)=1$
 - if $r_1 = 0$ then $\gcd(x,y)=y$

- If $r_1 > 1$ then we must continue our process
- $y = m_2 * r_1 + r_2$, where m_2 and r_2 are our next multiplier and remainder
- Now if we have $r_2 = 1$ then $\gcd(x, y) = 1$
- If $r_2 = 0$, then $\gcd(x, y) = r_1$

- If $r_2 > 1$, then we must do another step, and continue this process until we reach a remainder of either 0 or 1.

$$x = m_1 * y + r_1$$

$$y = m_2 * r_1 + r_2$$

$$r_1 = m_3 * r_2 + r_3$$

$$r_2 = m_4 * r_3 + r_4$$

.....

$$r_i = m_{i+2} * r_{i+1} + r_{i+2}$$

.....

Euclid's Algorithm Example

$$p=7, q=11, n=pq=77$$

$$\phi(n)=(7-1)(11-1)=60, \text{ choose } k=13, \text{ find } \gcd(60,13)$$

$$\text{find } \gcd(60,13)$$

$$60=(4)(13)+8$$

$$13=(1)(8)+5$$

$$8=(1)(5)+3$$

$$5=(1)(3)+2$$

$$3=(1)(2)+1$$

$$2=(2)(1)+0$$

$$\gcd(60,13)=1$$

• If $r_2=0$, then $\gcd(x,y)=r_1$

$$ed \equiv 1 \pmod{\phi(n)}$$

$$d = e^{-1} \pmod{\phi(n)}$$

Inverses mod n

Extended Euclidean algorithm

Inverses mod n

We will number the steps of the Euclidean algorithm starting with step 0. The quotient obtained at step i will be denoted by q_i . As we carry out each step of the Euclidean algorithm, we will also calculate an auxiliary number, p_i . For the first two steps, the value of this number is given: $p_0 = 0$ and $p_1 = 1$. For the remainder of the steps, we recursively calculate $p_i = p_{i-2} - p_{i-1} q_{i-2} \pmod{n}$. Continue this calculation for one step beyond the last step of the Euclidean algorithm.

$$p_i = p_{i-2} - p_{i-1} q_{i-2} \pmod{n}$$

Example

Find the inverse of 15 mod 26.

$$\text{Step 0: } 26 = 1(15) + 11 \quad p_0 = 0$$

$$\text{Step 1: } 15 = 1(11) + 4 \quad p_1 = 1$$

$$\text{Step 2: } 11 = 2(4) + 3 \quad p_2 = 0 - 1(1) \pmod{26} = 25$$

$$\text{Step 3: } 4 = 1(3) + 1 \quad p_3 = 1 - 25(1) \pmod{26} = -24 \pmod{26} = 2$$

$$\text{Step 4: } 3 = 3(1) + 0 \quad p_4 = 25 - 2(2) \pmod{26} = 21$$

$$p_5 = 2 - 21(1) \pmod{26} = -19 \pmod{26} = 7$$

$$e=7, \phi(n)=160, d=?$$

decrypt(855) = (855²⁷⁵³) mod 3233

= 50432888958416068734422899127394466631453878360035509315554967564501 05562861208255997874424542811005438349865428933638493024645144150785

17209179665478263530709963803538732650089668607477182974582295034295 04079035818459409563779385865989368838083602840132509768620766977396
67533250542826093475735137988063256482639334453092594385562429233017 51977190016924916912809150596019178760171349725439279215696701789902
1343071464689712796102771813783945869772898693423652403116932170892 6961764372652131566583315871245975980304250314006837883246101784830
71758547454725206968892599589254436670143220546954317400228550092386 369424448559733306305160738530286321930291350374547194675776713579
54965202919790505781532871558392070303159585937493663238548602090830 63550704455658896319318011934122017826923344101330116480696334024075
04695258866987658669060224024102088466507530263958370526631933584734 81094876156227126037327597360375237388364148088948438061575757045380
08107946980066734877795883758289985132793070353355127509043994817897 90548993381217329458535447413268056981087263348285463816885048824346
5889783933466254454006619645218766694795528023088412465948239275105 77049113329025684306505229256142730389832089007051511055250618994177
231777951579429711795475296301837848629139777661298207389072796 762720235011399271581964273076407418989190486860748124549315795374377
12441601438765069145868196402276027766869530903951314968319097324505 45234594477256587887692693353918692354818518542420923064996406822184
49011913571088542442852112077371223831105455431265307394075927890822 60604317113339575226603445164525976316184277459043201913452893299321
61307440532227470572894812143586831978415592726496357090901215131304 15756920979851832104115596935784883366531595132734467524394087576977
78908490126915322842080949630792972471304442194243906590308142893930 29158483087368745078977086921845296741146321155667865528338164806795
455941891006950919658990854567980723923708463025534568691923556269 57157358790622745861957217211107882865756385970941907763205097832395
71346411902500470208485604082175094910771655317165297473803176765820 58767314028891032883431850884472116442719390374041315564986995913736
51621084511374022433518599576657753969362812542539006855262454561419 25880943740212888666974410972184534221817198089911953707545542033911
9645393664617929681653426522346399367423309181353390462367769367038 05342644821735823842192515904381485247388968642443703186654199615377
9139694900303958760654915244945043600135939277133952101251928572092 59788751160195962961569027116431894637342650023631004555718003693586
0552649100090724518378668956441716490727835628100970854524135469660 84481161338780654854515176167308605108065782936524108723263667228054
00387941086434822675009077826512101372819583165313969830908873174174 7453598868429855980718519221597004650810606844559536480892249405427
66329674592308898484868435865479850511542844016462352696931799377844 30217857019197098751629654665130278009966580052178208139317232379013
2324946826092008199810376848471678749891936949979148247163450609372 56541225019537951668976018558775993133677977939527822273233375295802
63122665358948205566515289466369032083287680432390611549350954590934 06676402258670848337605369986794102620470905715674470565311124286290
73548884929899835609996360921411284977458614696040287029670701478179 49024828290748416008368045866685507604619225209434980471574526881813
18508591501948527635965034581536416565493160130613304047344579651083 80304062240278898042825189094716292266898016684480963645198090510905
79651307507392459580744797523712667610114738787432144149154813591743 92794969565415563866883891715446305611805369728343470219206348999531
91764016110392490439179803398975491765395923608511807653184706473318 01578207412764787592739087492955716853665185912666373831235945891267
87095838000224515094244575648744840868775308453955217306366938917023 94037184780362774643171470855830491959895146776294392143100245613061
114299370005577513397172825491100560089408984196713197091118165542908 761090083242997831338240786961578492341986299168008677495934077593066
02207814943807854996798945399364063685722697422361858411425048372451 24465580270859179795591086523099756519838277952945756996572445578688
38354423268572236813990212613637440821314784832035636156113462870198 5142390184229097416538620232051039712184983355286308685184282634615027
44187358639504042281512399505995983653792227285847422071677836679451 34363807086579774219853595393166279988789721695963455346336497949221
13017661316207477266113107012321403713882270221723233085472679533016 07998062253835458948024820043144726191596190526034069061930939290724
1028498700167172969517703467909979440975063764929635675558007116218 27727603182921790350290486090976266285396627024392536890256337101471
68327404504583060228676314215815990079164262770005461232291921929971 69907690169025946468104141214204472402661658275680524166861473393322
65959127006456304474160852916721870070451446497932266687321463467490 411858867608368403619069578699096521390675205019744076776510438851
51941619318479919134924388152822038464729269446084915299958818598855 19514906630731177323813226751694588259363878610724302565980914901032
78384821401136556784934102431512482864529170314100400120163648299853 2516634905605379458508942440385525245547792240104614890752745163425
1399216373835681414904793203742633701987825405699619163520193896982 5447863130977349154478427634532593998741700138163189116645377208944
002854850026968582644562183794116702151847721909339232185087775790 95933267631141312961939849592613898790166971088102766386231676940572
95932538078643444100512138025081797622723797210352196773268441946486 16402961059899027710532570457016332613431076417700043237152474626393
9901189972784536294930363691490088106053123163009010150839331880116 6821516389310406665951378274989237455605110040164777168227162672078
3701224246551264878454923504185216742638318973332434674449039780017 84689726405462148024124125833843501704885320601475687862318094090012
63241969092252022679880113408073012216264404133887392600523096072386 158554965158001034746119792130767224543803671883257030860671331132581
9922797552277184864847532612430280417794309093899237083053652046462 5147267884961527773274119265709116613580084145421487687310394401054
79639308530896880365608504772144592172500126500717068969428154627563 70458838904219177398190648731908014828739058159462227867277418610111
0276324797290412221199411788204562353701759090678628159281519982214 57652796853892517218720090070389138562840007332258507590485348046564
5434983707328762595891427854318266587294608072389652291599021738887 95773647738726574610400822551124182720096168188828493894678810468847
31265541726209789056784581096517975300873063154649030211213352818084 76122990409576427857316364124880930949770739567588422963171158464569
8420245510902988238517953684125891446352791897307683834073696131409 7452298563868827269104335751767712889452788136862396506654089894394
95161912002160777898876864736481837825324846699168307281220310791935 64668640159148562699993734427677252275403853322196852298590851548110
40229657916338257385513314823459591633281445819843614596306024993617 53097925561238039014690665163673718859582772525683119989984646027216
46279764077057074816406450769779869955106180046741937808223250148934 07851137833251073738233403466269553292608813843895784099804170410417
7760846306286610614059615207066695243018483575031762939543026312673 77406936404705896083462601885911184367532529848040849710922999195
65539701911919188327308603766775339607722455632113506572191067587 5118681278634419757239219526333856538388240057910256494923394519
659392039923922174002472341471909709645621208299547746193228981812886 055565809385189881181290561427408580916876571191122476328868712755
38928438126611991937924624112632990739867854558756652453056197509891 1457811473577128360755400177426866096509330517210272306635739462334
13638045914237759965220309418558880039496755829711258361621890140359 54234930424749053693992776114261796407100127643280428706083531594582

305946326827861270203356980346143245697021484375 **mod 3233 = 123**

Assignments

1. Perform encryption and decryption using RSA algorithm, for the following:
 - ① $p = 3$; $q = 11$, $e = 7$; $M = 5$
 - ② $p = 5$; $q = 11$, $e = 3$; $M = 9$
2. In a public-key system using RSA, you intercept the ciphertext $C = 10$ sent to a user whose public key is $e = 5$, $\phi(n) = 35$. What is the plaintext M ?

RSA for digital signing and verification

Digital Signing

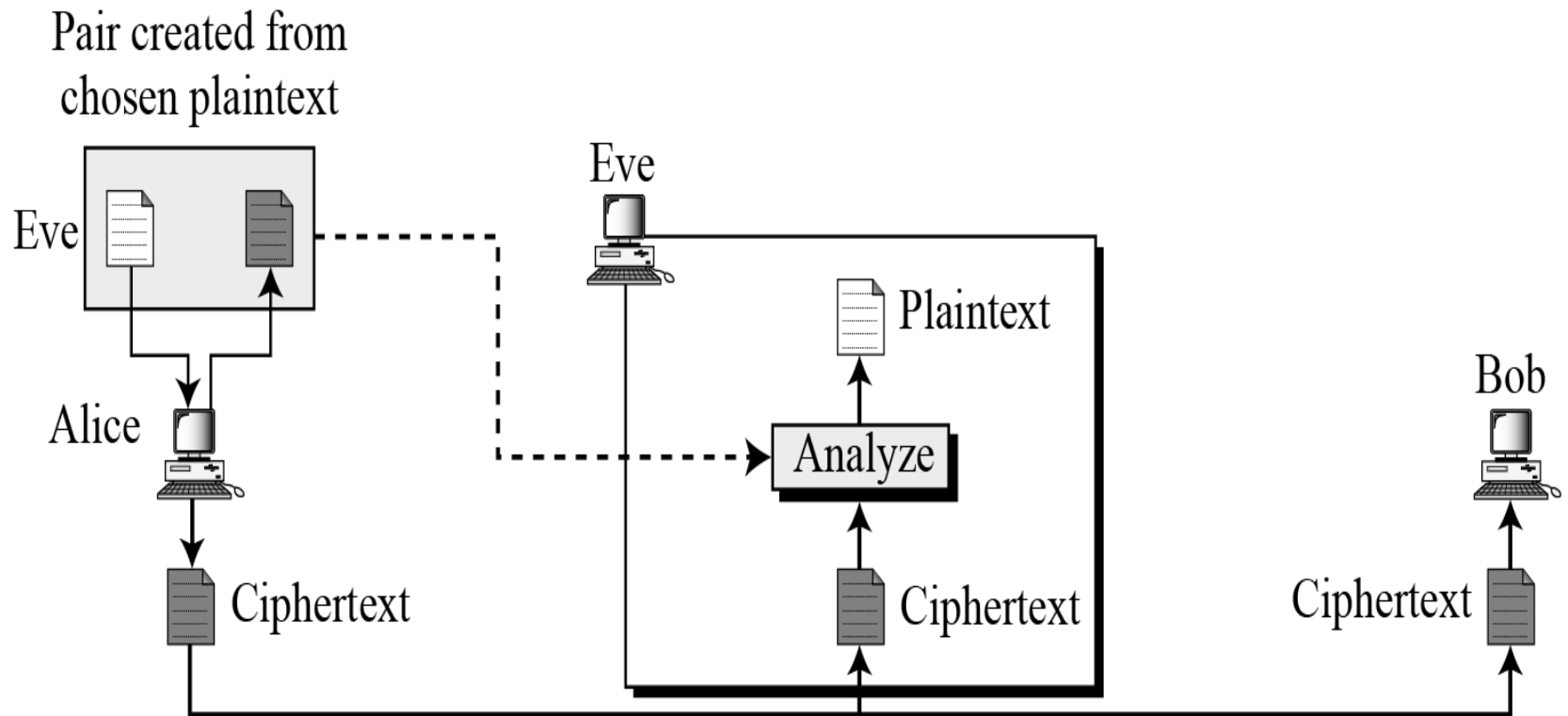
- In order to sign a message the sender does the following:
 - produces a hash value of the message
 - uses his/her private key (n,d) to compute the signature
 - **$S = M^d \bmod n$**
 - Sends the signature S to the recipient

- **Signature Verification**
- The recipient does the following in order to verify the message:
 - Uses the sender's public key (n,e) to compute the hash value **$V = S^e \bmod n$**
 - Extracts the hash value from the message
 - If both hash values are identical then signature is valid.

- Anyone else who wants to compute d , must first know $\varphi(n)$, but to know $\varphi(n)$, one must know p and q . In other words, they must factor n .

DES vs. RSA

- An RSA operation is can be performed by a series of modular multiplications.
 - DES several rounds – Feistel Structure
- By comparison, DES is much faster than RSA.
 - In software, DES is generally at least 100 times as fast as RSA.
 - Generation of numbers used in RSA can take time
 - In hardware, DES is between 1,000 and 10,000 times as fast, depending on the implementation.
- RSA can be used both for encryption and for digitally signing.
- RSA is generally considered to be secure when sufficiently long keys are used (512 bits is insecure, 768 bits is moderately secure, and 1024 bits is good, for now).
- The security of RSA relies on the difficulty of factoring large integers. Dramatic advances in factoring large integers would make RSA vulnerable.
- RSA is very vulnerable to chosen plaintext attacks.



Given: $P_1, C_1 = E_k(P_1), P_2, C_2 = E_k(P_2), \dots, P_t, C_t = E_k(P_t),$
 where the cryptanalyst gets to choose P_1, P_2, \dots, P_t

Advanced Encryption Standard

- AES was published by NIST (National Institute of Standards and Technology) in 2001.
- Symmetric block cipher
- Block length - 128 bits
- Key length - 128, 192 and 256 bits
- AES is not a Feistel structure
- AES process the entire data block in parallel during each round using substitutions and permutations.
- Rounds: 10,12 or 14 (depending on key size)

Key Management

- Public key encryption schemes are secure only if the authenticity of the public key is assured.
- Key management deals with the secure generation, distribution, and storage of keys. Secure methods of key management are extremely important.
- Once a key is randomly generated, it must remain secret to avoid unfortunate mishaps (such as impersonation).
- In practice, most attacks on public-key systems will probably be aimed at the key management level, rather than at the cryptographic algorithm itself.

- Users must be able to securely obtain a key pair suited to their efficiency and security needs.
- There must be a way to look up other people's public keys and to publicize one's own public key.
- Users must be able to legitimately obtain others' public keys; otherwise, an intruder can either change public keys listed in a directory, or impersonate another user.

Distribution of Public Keys

- *Public announcement of public key*
 - Public key made public
 - RSA – any participant can send his public key to any other participant or broadcast the key to the community at large
 - Disadv.
 - Any one can forge such a public announcement
 - i.e. some user could pretend to be user A and send a public key to another participant or broadcast such a public key

- *Public Key Authority*

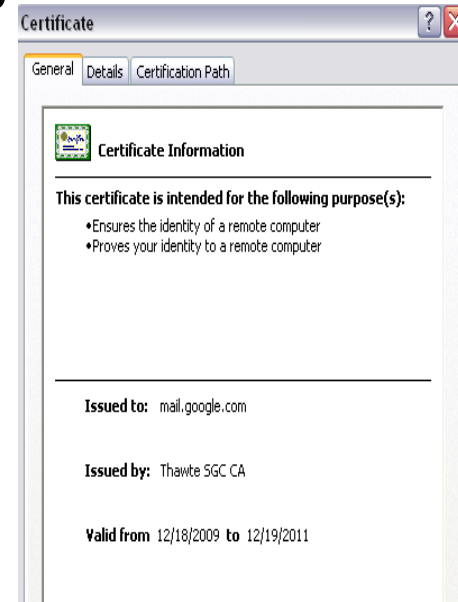
- Stronger control over the distribution of public keys from a directory
- A central authority maintains a dynamic directory of public keys of all participants
- Each participant reliably know a public key of the authority, with only the authority knows the corresponding private key.

- *Public Key Authority – steps*

- A send a time stamped message to the public key authority containing a request for the current public key of B
- The authority responds with a message that is encrypted using authorities private key PR_{auth} . Thus A is able to decrypt the message using the authority's public key
 - The message includes
 - B's public key, PUB which A can use to encrypt messages destined for B
 - The original time stamp, so A can determine that this is not an old message from authority.
 - The original request, sent by A to authority to ensure that the request was not altered.
- A stores B's public key and also uses it to encrypt a message to B containing an identifier of A(IDA) and a transaction id ($N1$)
- B retrieves A's public key from authority in the same manner as A retrieved B's public key

• Public-key Certificates

- Use certificates that can be used by participants to exchange keys without contacting a public key authority.
- A certificate consists of a public key plus an identifier of the key owner with the whole block signed by a trusted third party (certificate authority)
 - a Govt. agency, financial institution)
- A user can present his/her public key to the authority in a secure manner and obtain a certificate. The user can then publish the certificate.
- Anyone needed this users public key can obtain the certificate and verify that it is a valid by way of the attached trusted signature.



Diffie-Hellman Key Exchange Protocol

- **Diffie-Hellman key exchange protocol**
allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel.
- This key can then be used to encrypt subsequent communications using a symmetric key cipher.
- Developed by Diffie and Hellman in 1976 and published in the ground-breaking paper "New Directions in Cryptography."

- The protocol has two system parameters p and g .
- They are both public and may be used by all the users in a system.
- Parameter p is a prime number
- and parameter g (usually called a generator) is an integer less than p , with the following property:
 - for every number n between 1 and $p-1$ inclusive, there is a power k of g such that $n = g^k \bmod p$.

- Suppose Alice and Bob want to agree on a shared secret key using the Diffie-Hellman key agreement protocol.
- First, Alice generates a random private value a and Bob generates a random private value b .
- Both a and b are drawn from the set of integers \mathbb{Z}_p .
- Then they derive their public values using parameters p and g and their private values.
- Alice's public value is $g^a \bmod p$ and Bob's public value is $g^b \bmod p$.
- They then exchange their public values.

- Finally, Alice computes $g^{ab} = (g^b)^a \bmod p$,
i.e. $(g^b \bmod p)^a \bmod p = (g^b)^a \bmod p$
- Bob computes $g^{ba} = (g^a)^b \bmod p$
i.e. $(g^a \bmod p)^b \bmod p = (g^a)^b \bmod p$
- Since $g^{ab} = g^{ba} = k$, Alice and Bob now have a shared secret key k .

- It assumes that it is computationally infeasible to calculate the shared secret key $k = g^{ab} \bmod p$ given the two public values $g^a \bmod p$ and $g^b \bmod p$ when the prime p is sufficiently large.

1. Alice and Bob agree to use a prime number $p=23$ and base $g=5$.
2. Alice chooses a secret integer $a=6$, then sends Bob $A = g^a \bmod p$
 - $A = 5^6 \bmod 23$
 - $A = 15,625 \bmod 23$
 - $A = 8$
3. Bob chooses a secret integer $b=15$, then sends Alice $B = g^b \bmod p$
 - $B = 5^{15} \bmod 23$
 - $B = 30,517,578,125 \bmod 23$
 - $B = 19$
4. Alice computes $s = B^a \bmod p$
 - $s = 19^6 \bmod 23$
 - $s = 47,045,881 \bmod 23$
 - $s = 2$
5. Bob computes $s = A^b \bmod p$
 - $s = 8^{15} \bmod 23$
 - $s = 35,184,372,088,832 \bmod 23$
 - $s = 2$

Both Alice and Bob have arrived at the same value

Man-in-the-middle attack

- The Diffie-Hellman key exchange is vulnerable to a man-in-the-middle attack.
- An opponent Carol intercepts Alice's public value and sends her own public value to Bob.
- When Bob transmits his public value, Carol substitutes it with her own and sends it to Alice.
- Carol and Alice thus agree on one shared key and Carol and Bob agree on another shared key.
- After this exchange, Carol simply decrypts any messages sent out by Alice or Bob, and then reads and possibly modifies them before re-encrypting with the appropriate key and transmitting them to the other party.
- This vulnerability is present because Diffie-Hellman key exchange does not authenticate the participants.
- Possible solutions include the use of digital signatures.

Message Authentication

What is Authentication?

- A procedure to verify that received messages come from the alleged source and have not been altered.

- Message authentication is a mechanism or service used to verify the integrity of a message.
- Message authentication assures that data received are exactly as sent by (i.e., contain no modification, insertion, deletion, or replay) and that the purported identity of the sender is valid.

- Message authentication involves two aspects:
 - *Source authentication*, which verifies the identity of the source, prevents the acceptance of messages from a fraudulent source.
 - *Data integrity*, which protects the data from modification.

Authentication Requirements

- Disclosure
 - Release of message contents to any person or process not possessing the appropriate cryptographic key
- Traffic analysis
 - process of intercepting and examining messages in order to deduce information from patterns in communication
- Masquerade
 - Insertion of messages into the network from a fraudulent source.

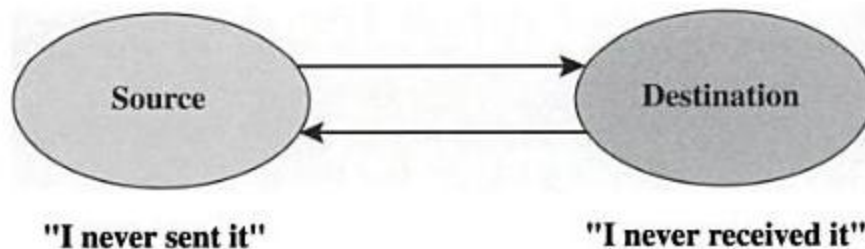
Authentication Requirements

- Content modification
 - Changes to the contents of a message, including insertion, deletion, transposition and modification
- Sequence modification
 - Any modification to a sequence of messages between parties, including insertion, deletion and reordering.

Authentication Requirements

- Timing modification
 - Delay or replay of messages
- Source repudiation
 - Denial of transmission of message by source.
- Destination repudiation
 - Denial of receipt of message by destination

Figure 4-11
An example of
repudiation



Nonrepudiation?

- Nonrepudiation means to ensure that a transferred message has been sent and received by the parties claiming to have sent and received the message.
- Nonrepudiation is a way to guarantee that the sender of a message cannot later deny having sent the message and that the recipient cannot deny having received the message.
- Nonrepudiation can be obtained through the use of digital signatures.

Measures to deal with the attacks

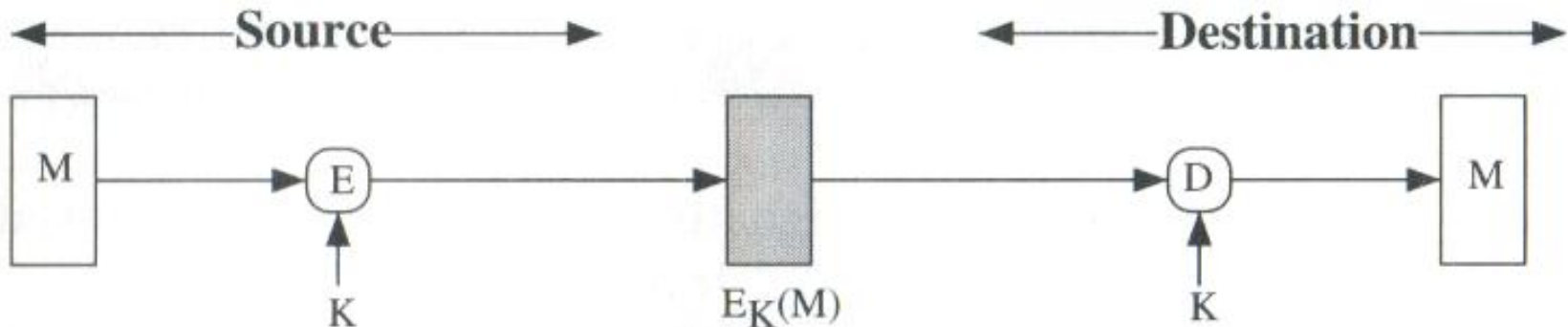
- Message confidentiality
 - Disclosure, Traffic analysis
- Message Authentication
 - Masquerade, Content modification, Sequence modification, Timing modification
- Digital Signature
 - Source and Destination repudiation

Authentication Functions

- 3 Types of cryptographic operations related to authentication:
 - **Message encryption:** The ciphertext of the entire message serves as its authenticator.
 - **Message authentication code (MAC):** A function of the message and a secret key that produces a fixed-length value that serves as the authenticator.
 - **Hash function:** A function that maps a message of any length into a fixed-length hash value, which serves as the authenticator.

Message Encryption

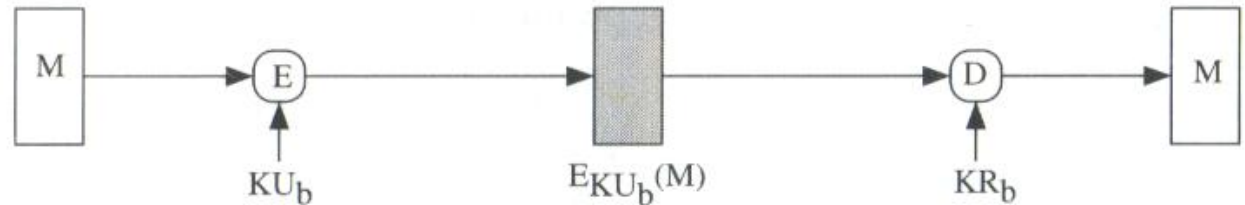
- if symmetric encryption is used then:
 - receiver know sender must have created it
 - since only sender and receiver know key used



(a) Conventional encryption: confidentiality and authentication

Message Encryption

- Public-key Encryption

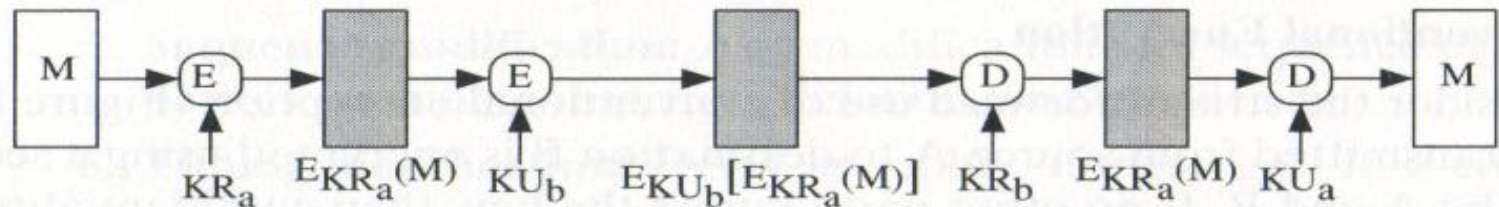


(b) Public-key encryption: confidentiality

Provides confidentiality but not authentication

No authentication – because any opponent could also use recipient's (B) public key to encrypt a message claiming to be the sender A

- if public-key encryption is used:
 - encryption provides no confidence of sender
 - since anyone potentially knows public-key
 - however if
 - sender **signs** message using their private-key (signature)
 - then encrypts with recipients public key (confidentiality)
 - have both secrecy and authentication
 - **Disadv:** public key algorithm - exercised four times rather than two in each communication



(d) Public-key encryption: confidentiality, authentication, and signature

Message Authentication Code

- An alternative authentication technique involves the use of a secret key to generate a small fixed-size block of data, known as a cryptographic checksum or MAC that is appended to the message.

Message Authentication Code

- generated by an algorithm that creates a small fixed-sized block
 - depending on both message and some key
 - like encryption though need not be reversible
- appended to message as a **signature**
- receiver performs same computation on message and checks it matches the MAC
- provides assurance that message is unaltered and comes from sender

MAC

- This technique assumes that two communicating parties, say A and B, share a common secret key K. When A has a message to send to B, it calculates the MAC as a function of the message and the key:

$MAC = C(K, M)$, where

M = input message

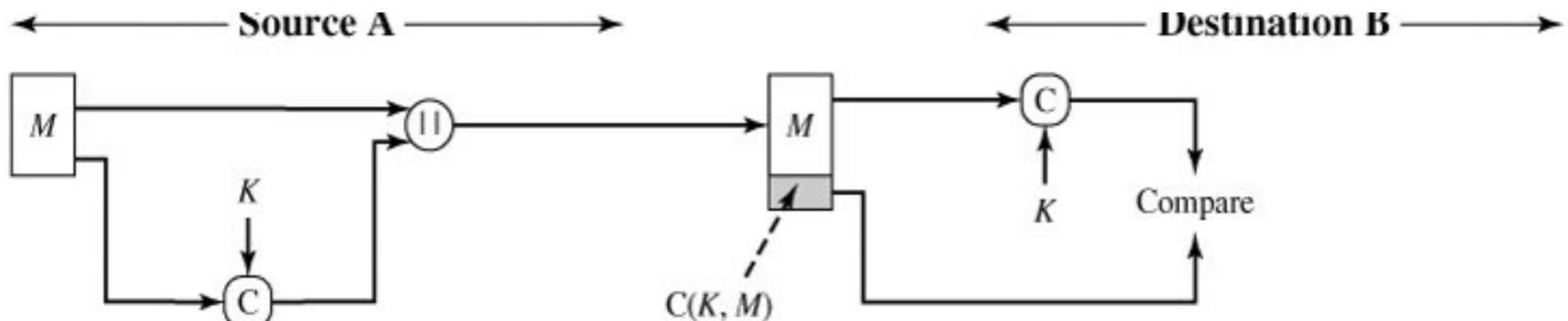
C = MAC function

K = shared secret key

MAC = message authentication code

Basic Uses of MAC

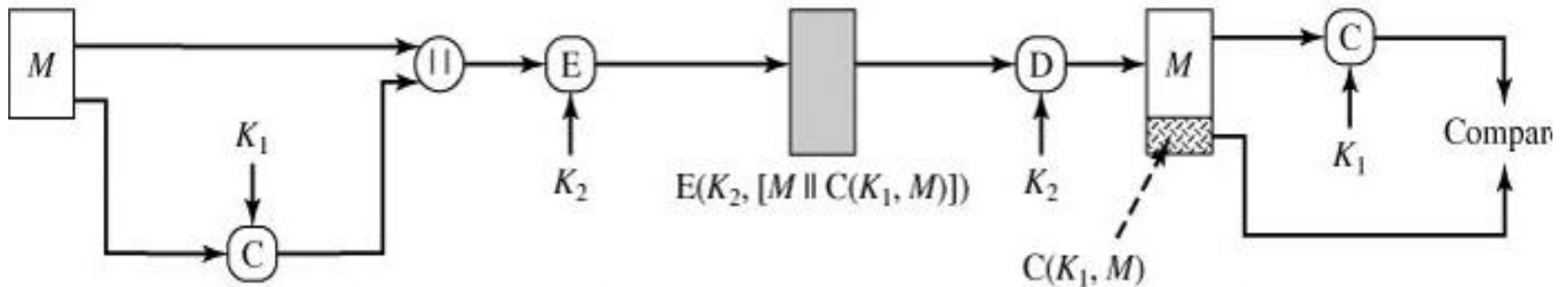
- a) The receiver is assured that the message has not been altered. If an attacker alters the message but does not alter the MAC, then the receiver's calculation of the MAC will differ from the received MAC. Because the attacker is assumed not to know the secret key, the attacker cannot alter the MAC to correspond to the alterations in the message.



(a) Message authentication

Basic Uses of MAC

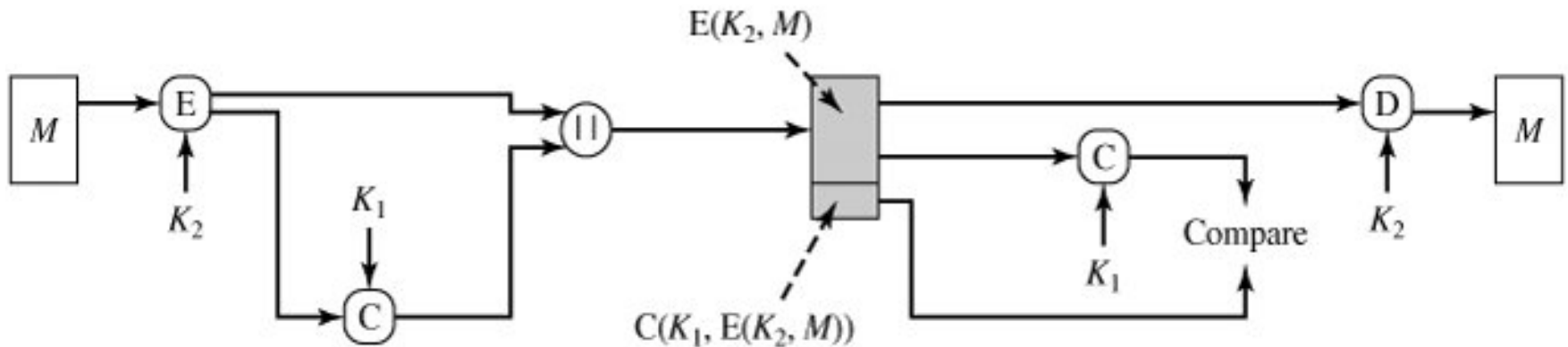
- b) The receiver is assured that the message is from the alleged sender. Because no one else knows the secret key, no one else could prepare a message with a proper MAC.



(b) Message authentication and confidentiality; authentication tied to plaintext

Basic Uses of MAC

c) authentication tied to ciphertext.



(c) Message authentication and confidentiality; authentication tied to ciphertext

(a) Message authentication

- $A \rightarrow B: M \parallel C(K, M)$
 - Provides authentication
Only A and B share K

(b) Message authentication and confidentiality: authentication tied to plaintext

- $A \rightarrow B: E(K_2, [M \parallel C(K_1, M)])$
 - Provides authentication
Only A and B share K_1
 - Provides confidentiality
Only A and B share K_2

(c) Message authentication and confidentiality: authentication tied to ciphertext

- $A \rightarrow B: E(K_2, M) \parallel C(K_1, E(K_2, M))$
 - Provides authentication
Using K_1
 - Provides confidentiality
Using K_2

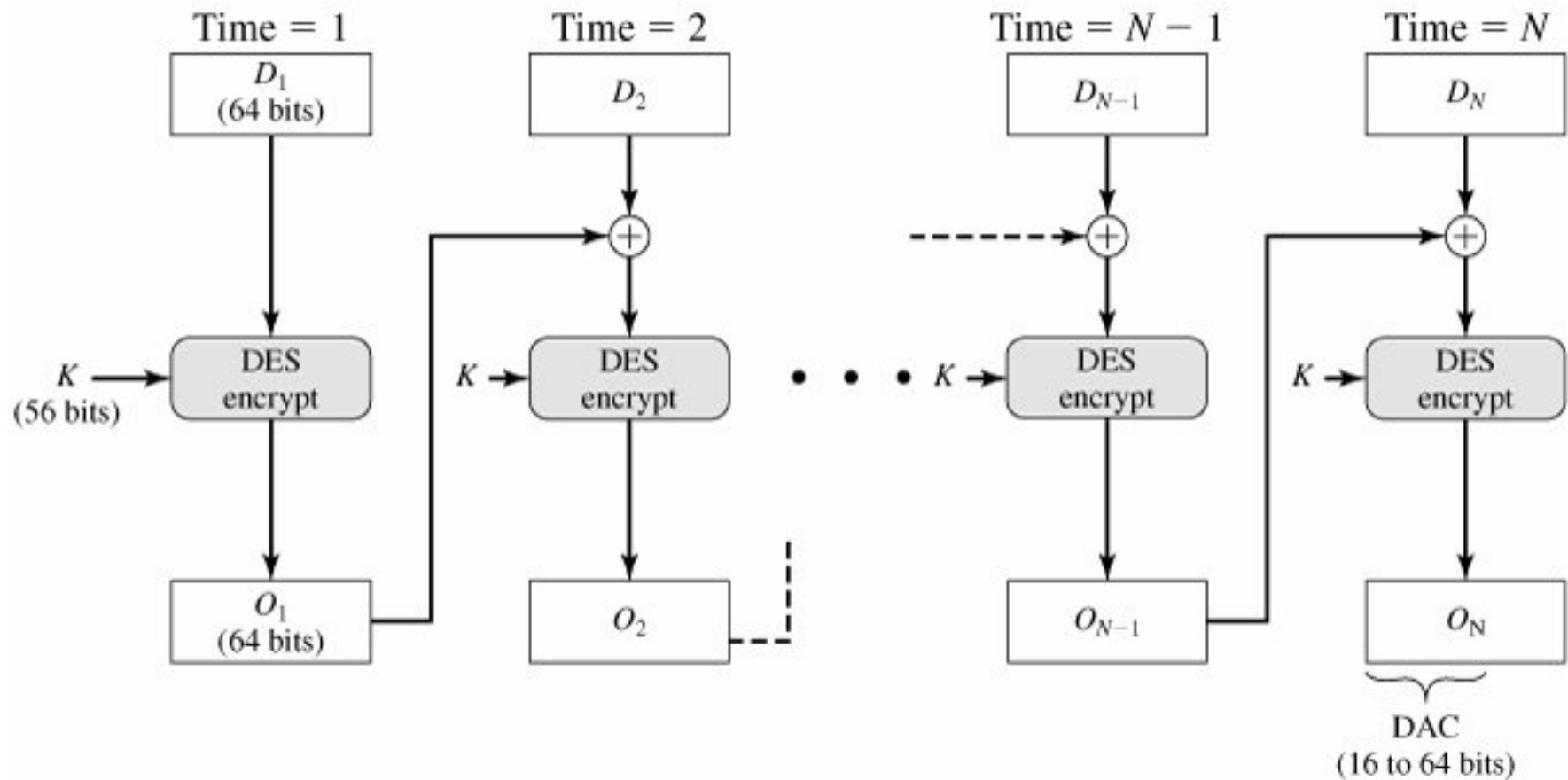
MAC Requirements

- The MAC function should satisfy the following requirements:
 1. If an opponent observes M and $C(K, M)$, it should be computationally infeasible for the opponent to construct a message M' such that $C(K, M') = C(K, M)$.
(it should infeasible that opponent is able to construct a new message to match a given MAC)
 1. $C(K, M)$ should be uniformly distributed in the sense that for randomly chosen messages, M and M' , the probability that $C(K, M) = C(K, M')$ is 2^{-n} , where n is the number of bits in the MAC.
 2. Let M' be equal to some known transformation on M . That is, $M' = f(M)$. For example, f may involve inverting one or more specific bits. In that case, $\Pr[C(K, M) = C(K, M')] = 2^{-n}$.

MAC based on DES

- The Data Authentication Algorithm, based on DES, has been one of the most widely used MACs for a number of years.
- The algorithm can be defined as using the cipher block chaining (CBC) mode of operation of DES with an initialization vector of zero.
- The message is encrypted with some block cipher algorithm in CBC mode to create a chain of blocks such that each block depends on the proper encryption of the previous block.

- The data (e.g., message, record, file, or program) to be authenticated are grouped into contiguous 64-bit blocks: D_1, D_2, \dots, D_N .
- If necessary, the final block is padded on the right with zeroes to form a full 64-bit block.
- Using the DES encryption algorithm, E , and a secret key, K , a data authentication code (DAC) is calculated.



$$\begin{aligned}
 O_1 &= E(K, D_1) \\
 O_2 &= E(K, [D_2 \oplus O_1]) \\
 O_3 &= E(K, [D_3 \oplus O_2]) \\
 &\dots\dots\dots \\
 O_N &= E(K, [D_N \oplus O_{N-1}])
 \end{aligned}$$

Hash Functions

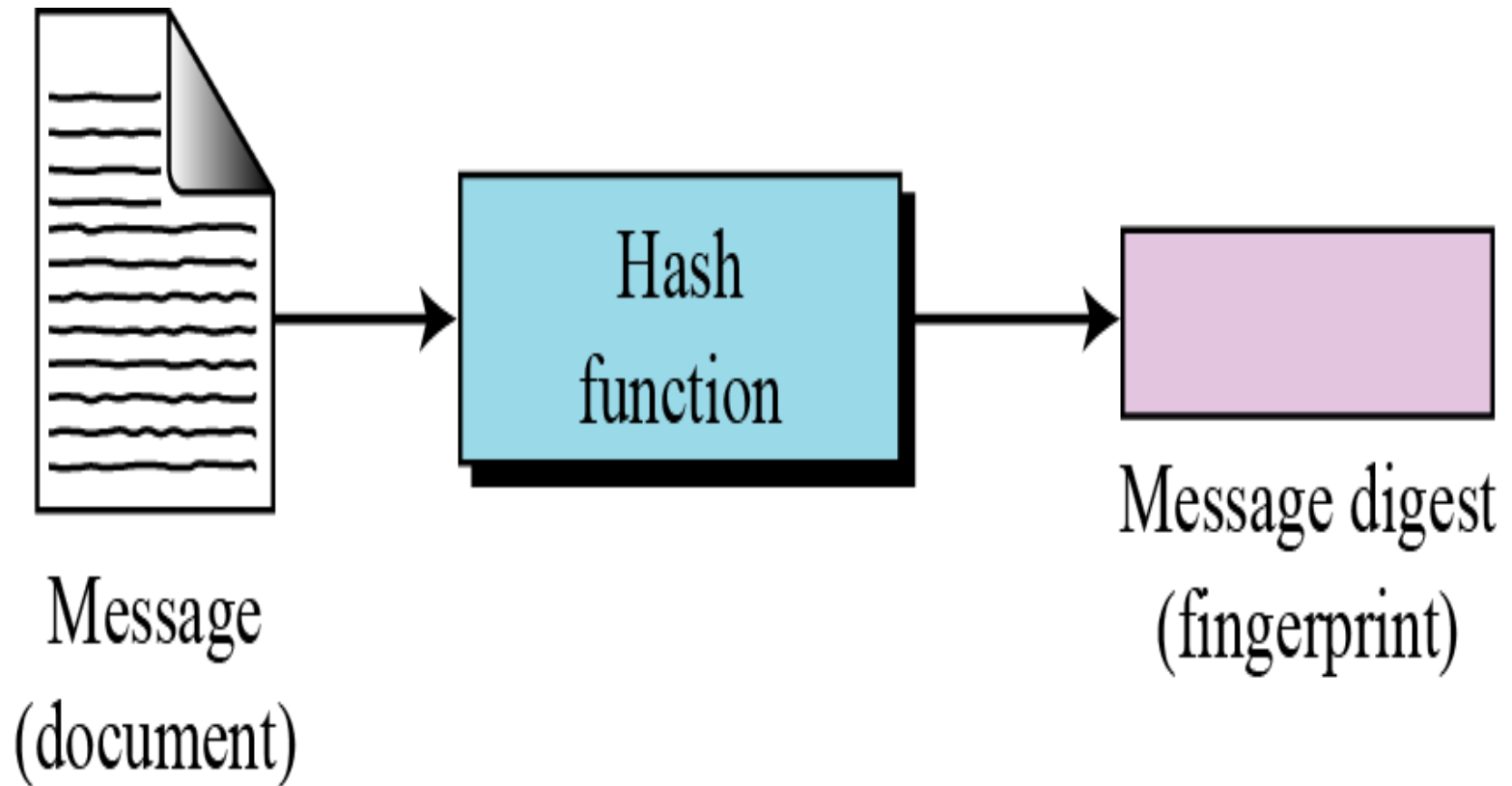
- Hash functions take a message as input and produce an output referred to as a hash code , hash value or simply hash.
 - More precisely, a hash function h maps bit strings of arbitrary finite length to strings of fixed length.

Hash Functions

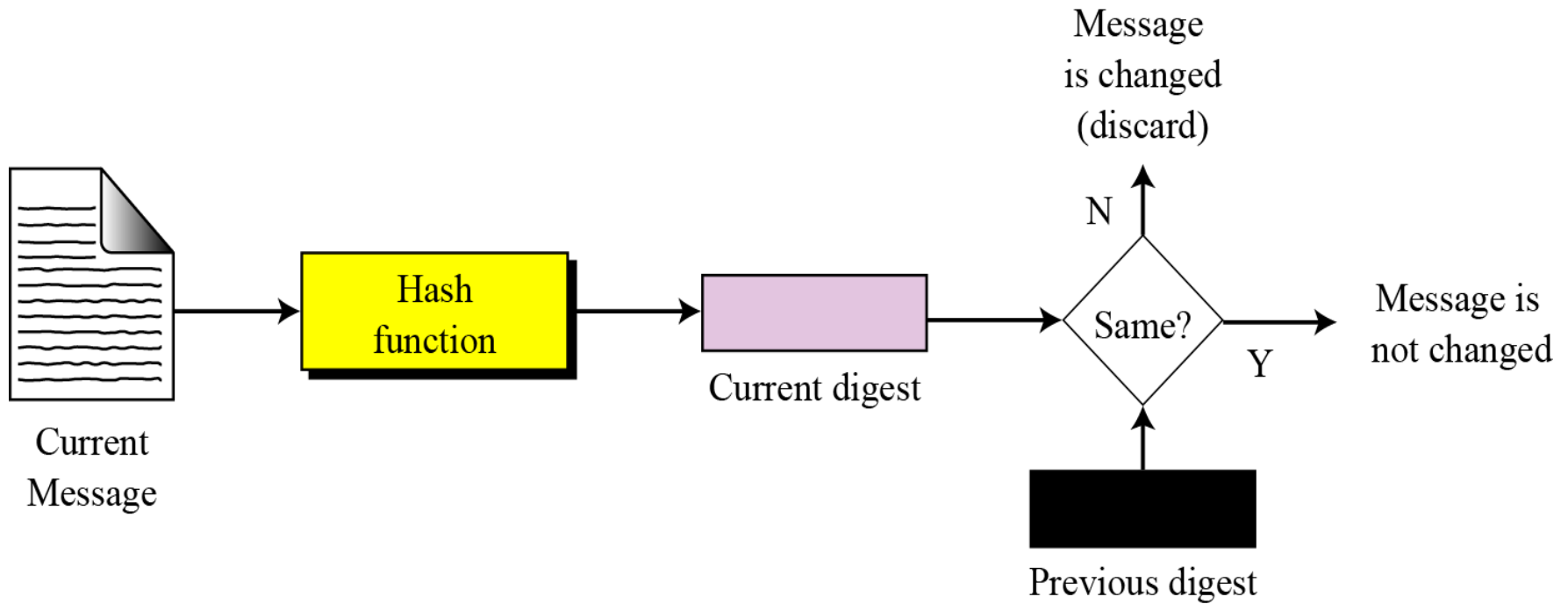
- A (one-way) hash function accepts a variable-size message M as input and produces a fixed-size hash code $H(M)$ as output (called Message Digest)
- Hash code provides error detection -> a change in one bit of message results in a change to the hash code.

- The purpose of a hash function is to produce a “fingerprint” of a file, message or other block of data.
- The hash value is appended to the message at the source.
- The receiver authenticates that message by recomputing the hash value.

Message and digest



Checking integrity



Requirements for a hash function

- H can be applied to a block of data of any size
- H produces a fixed length output
- It is easy to compute the hash value for any given message.
- It is infeasible to find a message that has a given hash. This is referred to as “one way property” (preimage resistance property)
- For any given block x , it is infeasible to find $y \neq x$ such that $H(y)=H(x)$, this is known as “weak collision resistance”
- It is infeasible to modify a message without changing its hash.

- Weak collision resistance
 - Guarantees that an alternative message hashing to the same value as a given message cannot be found.
- Strong Collision resistance
 - Refers to how resistant the hash function is to a type of attack known as birthday attack.

- Birthday attack

- If an encrypted hash code c is transmitted with the corresponding unencrypted message m the opponent can find m' such that $H(m')=H(m)$ and substitute another message and fool the receiver..i.e the opponent tries about 2^{63} messages to find one that matches the hash code of the intercepted message.

Simple Hash Functions

- based on XOR of message blocks
 - divide the message into equal size blocks
 - perform XOR operation block by block
 - final output is the hash
- not very secure

Simple Hash Function

- Bit-by-bit exclusive-OR (XOR)

$$C_i = b_{i1} \oplus b_{i2} \oplus \dots \oplus b_{im}$$

where C_i = i th bit of the hash code, $1 \leq i \leq n$

m = no. of n -bit blocks in the input

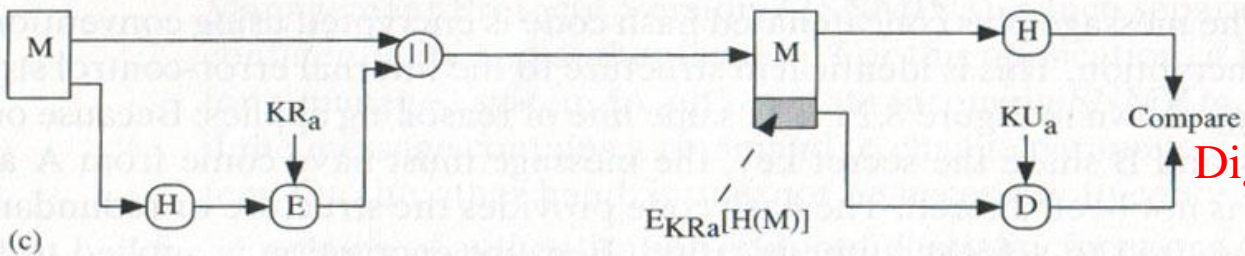
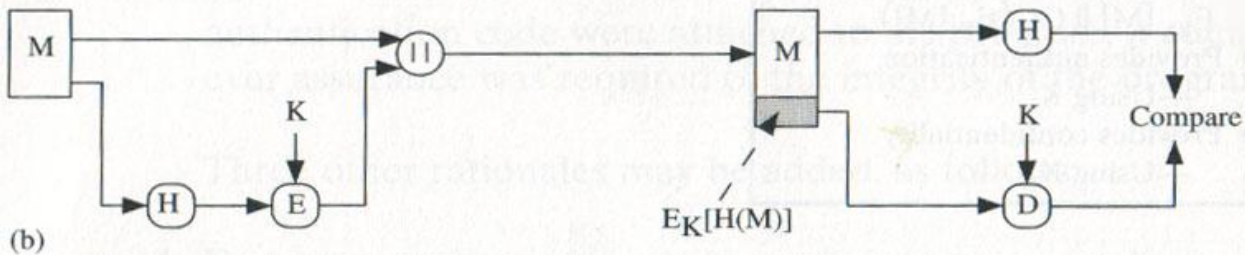
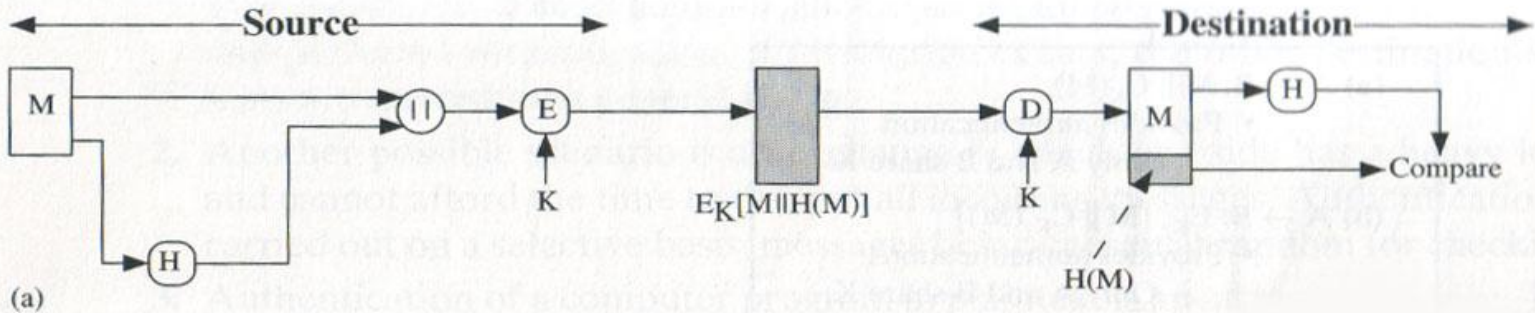
b_{ij} = i th bit in j th block

\oplus = XOR operation

	Bit 1	Bit 2	• • •	Bit n
Block 1	b_{11}	b_{21}		b_{n1}
Block 2	b_{12}	b_{22}		b_{n2}
	•	•	•	•
	•	•	•	•
	•	•	•	•
Block m	b_{1m}	b_{2m}		b_{nm}
Hash code	C_1	C_2		C_n

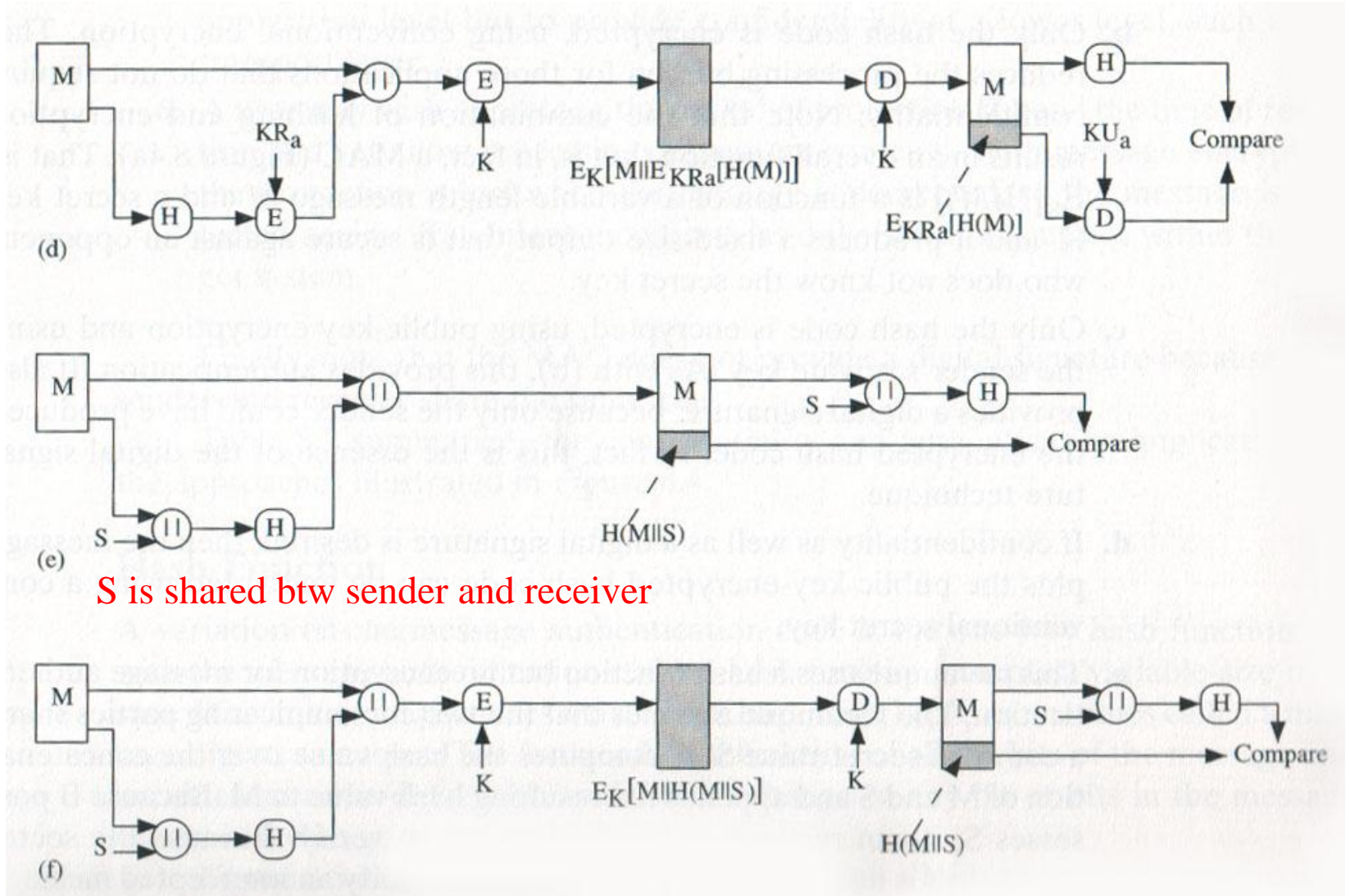
Figure 8.7 Simple Hash Function Using Bitwise XOR.

Basic Uses of Hash Functions

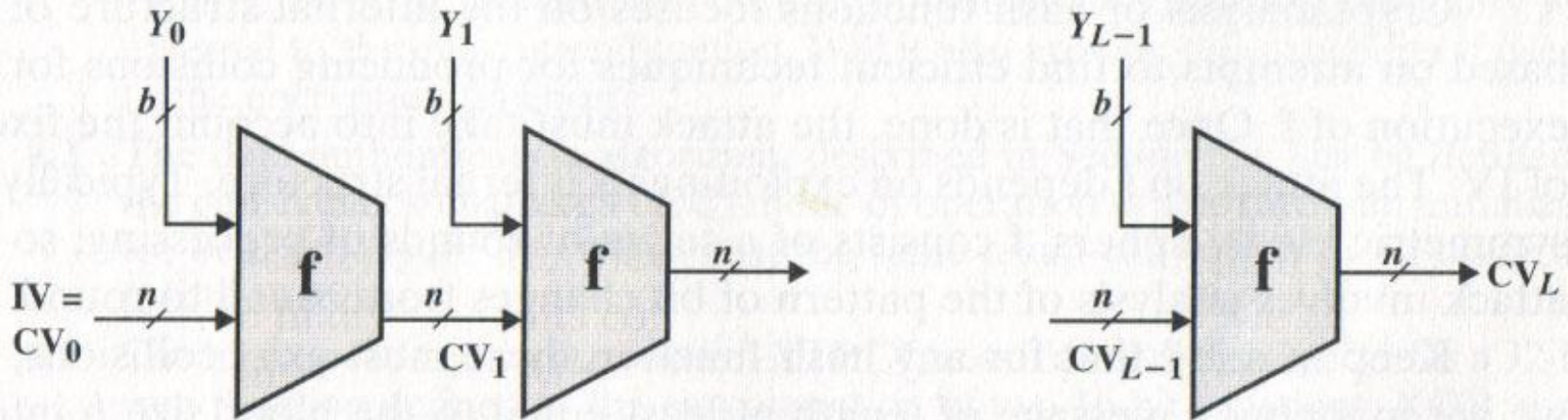


Digital Signature

Basic Uses of Hash Functions



General Structure of Hash Function



IV = initial value
CV = chaining variable
 Y_i = i th input block
 f = compression algorithm
 L = number of input blocks
 n = length of hash code
 b = length of input block

f : compression function taking two inputs and producing n -bit output
 $CV_0 = IV =$ initial n -bit value
 $CV_i = f(CV_{i-1}, Y_{i-1}), 1 \leq i \leq L$
 $H(M) = CV_L$

Figure 8.10 General Structure of Secure Hash Code.

DIGITAL SIGNATURE

- A sends a message to B with an authentication code, B modifies the message and claims that it has come from A. The modified message has an authentication code generated using the shared key of both A and B. There is no way to prove that A did in fact send the message.
- Hence, in situations where there is not complete trust between sender and receiver, something more than authentication is needed
 - The most attractive solution is **DIGITAL SIGNATURE**

Digital Signatures

- A signature is a technique for non-repudiation based on the public key cryptography.
- The creator of a message can attach a code, the signature, which guarantees the source and integrity of the message.

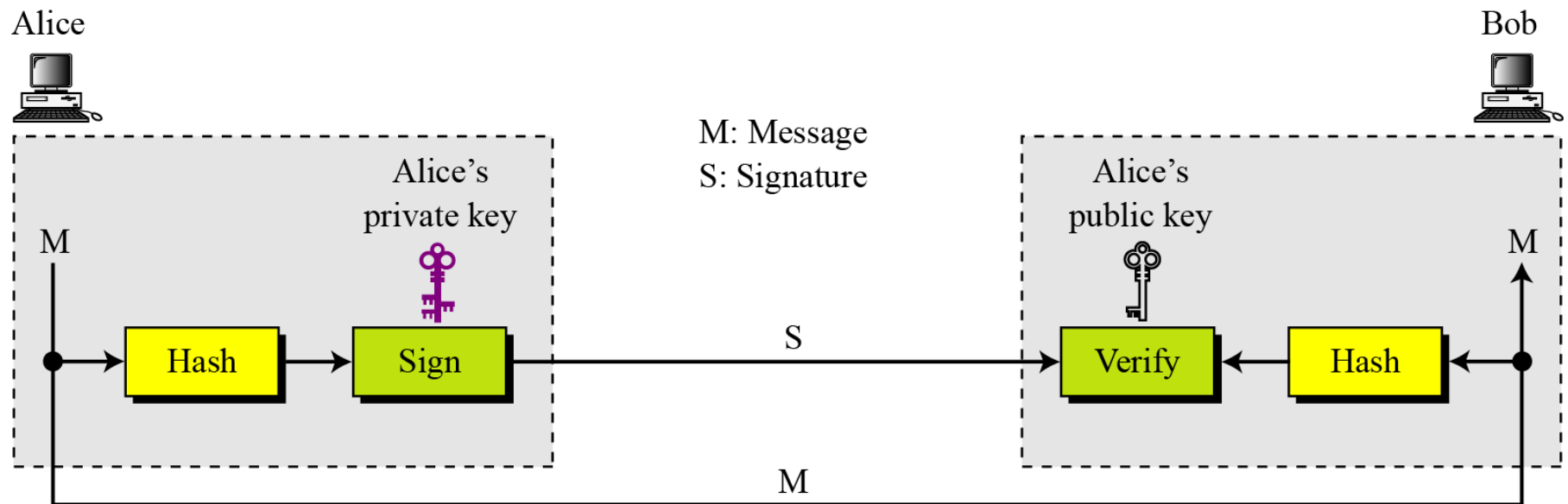
Properties of Signatures

- Similar to handwritten signatures, digital signatures must fulfill the following:
 - ✓ Must not be forgeable
 - ✓ Recipients must be able to verify them
 - ✓ Signers must not be able to repudiate them later
- In addition, digital signatures cannot be constant and must be a function of the entire document it signs

- The sender uses a **signing algorithm** to sign the message. The message and the signature are sent to the recipient.
- The recipient receives the message and the signature and applies the **verifying algorithm** to the combination. If the result is true, the message is accepted, otherwise it is rejected.

Signing the digest

Asymmetric-key cryptosystems are very inefficient when dealing with long messages. In a digital signature system, the messages are normally long, but we have to use asymmetric-key schemes. The solution is to sign a digest of the message, which is much shorter than the message itself.



Two categories of DS

- Direct DS
- Arbitrated DS

■ Direct DS

- involves only the communication parties
- The destination knows the public key of the source
- A DS may be formed by encrypting the entire message with the sender's private key or by encrypting a hash code of the message with the sender's private key.
- In case of dispute, some third party must view the message and its signature

■ Arbitrated DS

- Every signed message from a sender X to a receiver Y goes first to an arbiter A, who subjects the message and its signature to a number of tests to check the origin and content
- The message is then dated and sent to Y with an indication that it has been verified to the satisfaction of the arbiter.

The Digital Signature Standard

- Proposed by NIST – National Institute of Standards and Technology
- The DSA includes signature generation and verification.
- Generation makes use of a private key to generate a digital signature.
- Verification of the signature makes use of a public key.
- Each user possess a private and public key pair.
- Public keys are assumed to be known to all members of a group.
- Signature generation can be performed only by the possessor of the user's private key.

Digital Signature Algorithm (DSA)

- A hash function is used in the signature generation process to obtain a condensed version of data called message digest.
- The message digest is then signed.
- The digital signature is sent to the intended recipient along with signed data.
- The recipient of the message and signature verifies the signature by using the sender's public key.
- The same hash function must be used in the verification process.

Digital Signature Algorithm (DSA)

- creates a 320 bit signature, with 512-1024 bit security
- smaller and faster than RSA
- DSA authenticates the integrity of the signed data and the identity of the signer.
- The DSA may also be used in proving to a third party that data was actually signed by the generator of the signature.
- The DSA is intended for use in electronic mail, software distribution, electronic funds transfer, data storage etc...

DSA Key Generation

- have shared global public key values (p, q, g) :
 - choose q , a 160 bit
 - choose a large prime $p = 2^L$
 - where $L = 512$ to 1024 bits and is a multiple of 64
 - and q is a prime factor of $(p-1)$
 - choose $g = h^{(p-1)/q}$
 - where $h < p-1$, $h^{(p-1)/q} \pmod{p} > 1$
- users choose private & compute public key:
 - choose $x < q$
 - compute $y = g^x \pmod{p}$

DSA Signature Creation

- to **sign** a message M the sender:
 - generates a random signature key k , $k < q$
 - k must be random, destroyed after use, and never be reused
- then computes signature pair:
$$r = (g^k \pmod p) \pmod q$$
$$s = (k^{-1} \cdot H(M) + x \cdot r) \pmod q$$
- sends signature (r, s) with message M

DSA Signature Verification

- having received M & signature (r, s)
- to **verify** a signature, recipient computes:
$$w = s^{-1} \pmod{q}$$
$$u1 = (H(M) \cdot w) \pmod{q}$$
$$u2 = (r \cdot w) \pmod{q}$$
$$v = (g^{u1} \cdot y^{u2} \pmod{p}) \pmod{q}$$
- if $v=r$ then signature is verified

