الحاسبات

4th Class

Computers & Data Security

أمنية الحاسوب والبيانات

أستاذ المادة

م . د . انوار عباس حطاب

Chapter 1

Basic Data Security Concepts

Definitions

- **Computer Security** - generic name for the collection of tools designed to protect data and to thwart hackers.
- **Information systems security** is the ability to provide the services required by the user community while simultaneously preventing unauthorized use of system resources
- **Network Security** - measures to protect data during their transmission
- **Internet Security** - measures to protect data during their transmission over a collection of interconnected networks
- Aspects of Security: - 3 aspects of information security:
  - **security attack**
  - **security service**
  - **security mechanism**

Security Attack

- any action that compromises the security of information owned by an organization
- information security is about how to prevent attacks, or failing that, to detect attacks on information-based systems
- often *threat* & *attack* used to mean same thing
- have a wide range of attacks and can focus of generic types of attacks
  - passive
  - active

Passive Attacks

Darth
read contents of message from Bob to Alice
Internet or other comms facility
Bob
Alice

Active Attacks

Darth
Capture message from Bob to Alice; later replay message to Alice
Internet or other comms facility
Bob
Alice

Security Services

1. **Confidentiality:** - The concept of *Confidentiality* relate to the protection of information and prevention of unauthorized access or disclosure. The ability to keep data confidential, or secret, is critical to staying competitive in today’s business environments.

Examples of Confidentiality

- Student grade information is an asset whose confidentiality is considered to be very high
  - The US FERPA Act: grades should only be available to students, their parents, and their employers (when required for the job)
- Student enrollment information: may have moderate confidentiality rating; less damage if enclosed
- Directory information: low confidentiality rating; often available publicly

2. **Integrity:** - Integrity deals with prevention of unauthorized modification of intentional or accidental modification.

- **Data integrity**: assures that information and programs are changed only in a specified and authorized manner
- **System integrity**: Assures that a system performs its operations in unimpaired manner

Examples of Integrity

- A hospital patient's allergy information (high integrity data): a doctor should be able to trust that the info is correct and current
- If a nurse deliberately falsifies the data, the database should be restored to a trusted basis and the falsified information traced back to the person who did it
- An online newsgroup registration data: moderate level of integrity
- An example of low integrity requirement: anonymous online poll (inaccuracy is well understood)

3. Availability: - assures that the resources that need to be accessed are accessible to authorized parties in the ways they are needed. Availability is a natural result of the other two concepts (confidentiality and integrity).

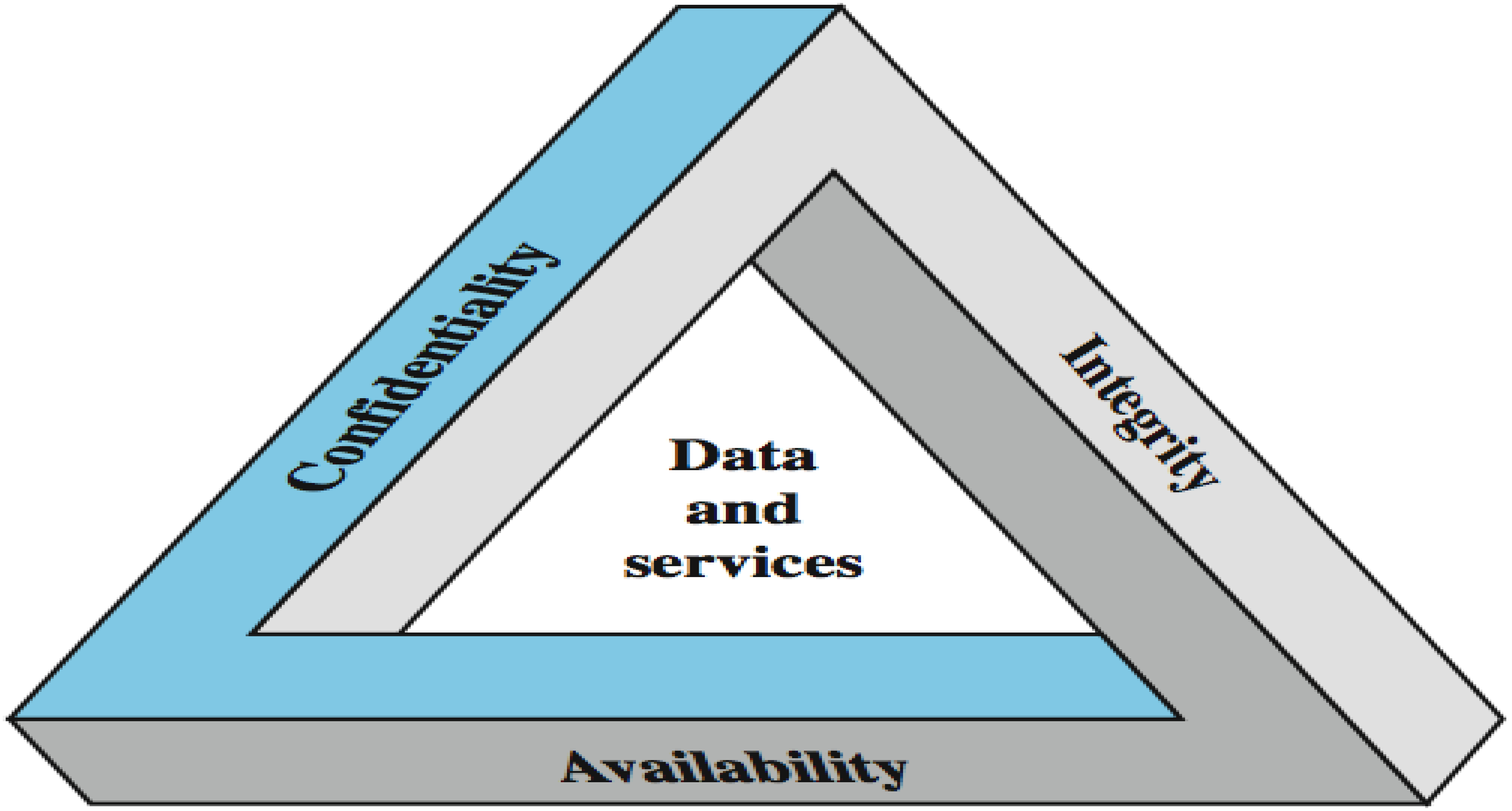
Examples of Availability

- A system that provides authentication: high availability requirement
  - If customers cannot access resources, the loss of services could result in financial loss
- A public website for a university: a moderate availably requirement; not critical but causes embarrassment
- An online telephone directory lookup: a low availability requirement because unavailability is mostly annoyance (there are alternative sources)

4. **Authentication** is the process by which the information system assures that you are who you say you are; how you prove your identity is authentic. Methods of performing authentication are:

- User ID and passwords. The system compares the given password with a stored password. If the two passwords match then the user is authentic.
- Swipe card, which has a magnetic strip embedded, which would already contain your details, so that no physical data entry takes place or just a PIN is entered.
- Digital certificate, an encrypted piece of data which contains information about its owner, creator, generation and expiration dates, and other data to uniquely identify a user.
- key fob, small electronic devices which generate a new random password synchronized to the main computer
- Biometrics - retinal scanners and fingerprint readers. Parts of the body are considered unique enough to allow authentication to computer systems based on their properties.

5. **Accountability (Non-Repudiation):** - The security goal that generates the requirement for actions of an entity to be traced uniquely to that entity. This supports nonrepudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action.



Security Mechanism

- feature designed to detect, prevent, or recover from a security attack
- no single mechanism that will support all services required
- however one particular element underlies many of the security mechanisms in use:
  - **cryptographic techniques**
- hence our focus on this topic

Chapter 2
Terminology and Background

- **Cryptography** is the art or science of keeping messages secret.
- **Cryptanalysis** is the art of **breaking** ciphers, i.e. retrieving the plaintext without knowing the proper key.
- People who do cryptography are **cryptographers**, and practitioners of cryptanalysis are **cryptanalysts**.
- Cryptography deals with all aspects of secure messaging, authentication, digital signatures, electronic money, and other applications.
- **Cryptology** is the branch of mathematics that studies the mathematical foundations of cryptographic methods.

The various components of a basic cryptosystem are as follows : -

- **Plaintext.** It is the data to be protected during transmission.
- **Encryption Algorithm.** It is a mathematical process that produces a ciphertext for any given plaintext and encryption key. It is a cryptographic algorithm that takes plaintext and an encryption key as input and produces a ciphertext.
- **Ciphertext.** It is the scrambled version of the plaintext produced by the encryption algorithm using a specific the encryption key. The ciphertext is not guarded. It flows on public channel. It can be intercepted or compromised by anyone who has access to the communication channel.
- **Decryption Algorithm,** It is a mathematical process, that produces a unique plaintext for any given ciphertext and decryption key. It is a cryptographic algorithm that takes a ciphertext and a decryption key as input, and outputs a plaintext. The decryption algorithm essentially reverses the encryption algorithm and is thus closely related to it.
- **Encryption Key.** It is a value that is known to the sender. The sender inputs the encryption key into the encryption algorithm along with the plaintext in order to compute the ciphertext.
- **Decryption Key.** It is a value that is known to the receiver. The decryption key is related to the encryption key, but is not always identical to it. The receiver inputs the decryption key into the decryption algorithm along with the ciphertext in order to compute the plaintext.
- For a given cryptosystem, a collection of all possible decryption keys is called a **key space**

Basic Cryptographic Algorithms

- A **cipher** is the method of encryption and decryption.
- Some cryptographic methods rely on the secrecy of the algorithms. **Keyless Cipher** is a cipher that does not require the use of a key.
- All modern algorithms use a **key** to control encryption and decryption; a message can be decrypted only if the key matches the encryption key.
- The key used for decryption can be different from the encryption key, but for most algorithms they are the same.

$E_K(m)$
ciphertext
m
plaintext
E
D
$D_{K'}(E_K(m)) = m$
K
encryption key
eavesdropping
adversary
K'
decryption key

Classical model of encryption

Basic classification of encryption key-based algorithms

1. Symmetric-key or (or secret-key) encryption algorithm.

- Symmetric algorithms use the same key for encryption and decryption (or the decryption key is easily derived from the encryption key)
- two main types:
  - **stream ciphers** – operate on individual characters of the plaintext
  - **block ciphers** – process the plaintext in larger blocks of characters

Symmetric Encryption
Key
Plaintext
Encryption
Ciphertext
Decryption
Original Plaintext

2. Asymmetric (or public-key) encryption algorithms.

- algorithms use a different key for encryption and decryption, and the decryption key cannot be derived from the encryption key.
- permit the encryption key to be public (it can even be published in a newspaper), allowing anyone to encrypt with the key, whereas only the proper recipient (who knows the decryption key) can decrypt the message. The encryption key is also called the **public key** and the decryption key the **private key** or **secret key**.
-
Asymmetric Encryption
Encryption Key
K_E
Decryption Key
K_D
Plaintext
Encryption
Ciphertext
Decryption
Original Plaintext

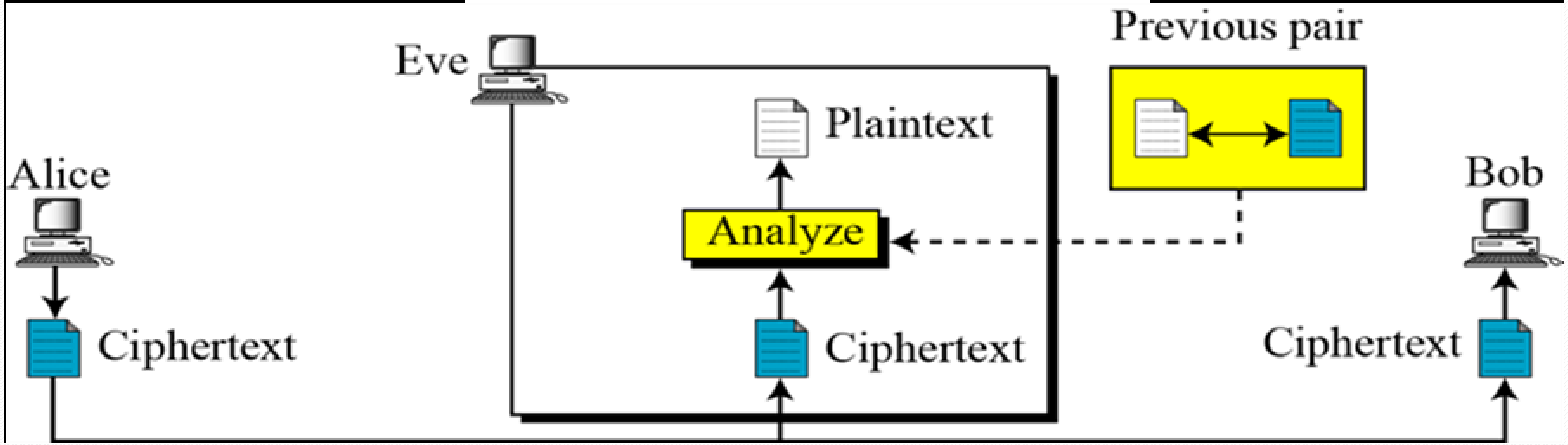
Cryptanalysis and Attacks on Cryptosystems

- There are many cryptanalytic techniques. Some of the more important ones for a system implementer are
  - **Ciphertext-only attack** (Only know algorithm / ciphertext, statistical, can identify plaintext): This is the situation where the attacker does not know anything about the contents of the message, and must work from ciphertext only. It is very hard.

• Ciphertext-only attack
Eve
Plaintext
Analyze
Ciphertext
Alice
Ciphertext
Bob
Ciphertext

- **Known-plaintext attack** (know/suspect plaintext & ciphertext to attack cipher): 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, or via some shortcut.

- **Known-plaintext attack**



- **Chosen-plaintext attack** (selects plaintext and obtain ciphertext to attack cipher): The attacker is able to have any text he likes encrypted with the unknown key. The task is to determine the key used for encryption.

• Chosen-plaintext attack
Pair created from chosen plaintext
Eve
Alice
Ciphertext
Eve
Plaintext
Analyze
Ciphertext
Bob
Ciphertext

- **Chosen Ciphertext Attacks** (select ciphertext and obtain plaintext to attack cipher): Attacker obtains the decryption of any ciphertext of its choice (under the key being attacked)

• Chosen Ciphertext Attacks
Alice
Ciphertext
Eve
Plaintext
Analyze
Ciphertext
Pair created from chosen ciphertext
Eve
Bob
Ciphertext

Chapter three
Mathematics

- **Modular Arithmetic**
  - several important cryptosystems make use of modular arithmetic. This is when the answer to a calculation is always in the range 0 – *m* where m is the **modulus**.
  - (a mod n) means the remainder when a is divided by n.
  - a mod n = r
  - a div n=q
  - a = qn + r
  - r = a – q * n

Example :- if a=13 and n=5, find q and r.

q=13 div 5=2 and r=13-2 *5=3 which is equivalent to (13 mod 5)

Example :- find (-13 mod 5).

This can be found by find the number (b) where 5*b >13 then let b=3 and 5*3=15 which is less than 13 so

-13 mod 5=5*3-13=2

- **Properties of Congruences.**
  - Two numbers ***a*** and ***b*** are said to be "*congruent modulo **n***" if

$$(a \bmod n) = (b \bmod n) \rightarrow a \equiv b(\bmod n)$$

  - The difference between ***a*** and ***b*** will be a multiple of ***n*** So ***a-b*** = ***kn*** for some value of ***k***
  - **Examples** $4 \equiv 9 \equiv 14 \equiv 19 \equiv -1 \equiv -6 \bmod 5$

$73 \equiv 4(\bmod 23$

- **Properties of Modular Arithmetic.**

1. $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
2. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
3. $[(a \bmod n) \text{ x } (b \bmod n)] \bmod n = (a \text{ x } b) \bmod n$

- Examples

$11 \bmod 8 = 3;\ 15 \bmod 8 = 7$

$[(11 \bmod 8) + (15 \bmod 8)] \bmod 8 = 10 \bmod 8 = 2$

$(11 + 15) \bmod 8 = 26 \bmod 8 = 2$

$[(11 \bmod 8) - (15 \bmod 8)] \bmod 8 = -4 \bmod 8 = 4$

$(11 - 15) \bmod 8 = -4 \bmod 8 = 4$

$[(11 \bmod 8) \times (15 \bmod 8)] \bmod 8 = 21 \bmod 8 = 5$

$(11 \times 15) \bmod 8 = 165 \bmod 8 = 5$

- **Exponentiation** is done by repeated multiplication, as in ordinary arithmetic.
- Example

$$\textit{To find } (11^{7} \bmod 13) \textit{ do the followings}$$

$$11^{2} = 121 \equiv 4 \pmod{13}$$

$$11^{4} \, (11^{2})^{2} \equiv 4^{2} \equiv 3 \pmod{13}$$

$$11^{7} \equiv 11 \times 4 \times 3 \equiv 132 \equiv 2 \pmod{13}$$

- **Greatest Common Divisor(GCD).**
  - Let *a* and *b* be two non-zero integers. The greatest common divisor of *a* and *b*, denoted gcd(a,b) is the largest of all common divisors of *a* and *b*.
  - When gcd(*a*,*b*) = 1, we say that *a* and *b* are *relatively prime*.
  - It can be calculated using the following equation: -

$$\mathbf{GCD(a,b)=GCD(b,a\ mod\ b)}$$

- Example :- find the GCD(72,48).

GCD(89,25)=GCD(25, 89 mod 25)= GCD(25, 14)

GCD(25, 14)=GCD(14, 25 mod 14)= GCD(14,11)

GCD(14,11)=GCD(11, 14 mod 11)= GCD(11,3)

GCD(11,3)=GCD(3, 11 mod 3)=GCD(3, 2)

GCD(3,2)=GCD(2, 3 mod 2)=GCD(2,1)

GCD(2,1)=GCD(1, 2 mod 1)=GCD(1,0) so the GCD(89,25)=1

Least Common Multiple (LCM).

- The least common multiple of the positive integers a and b is the smallest positive integer that is divisible by both a and b.
- The least common multiple of a and b is denoted by LCM(a, b).
- It can be calculated using the following equation: -

$$LCM(\mathrm{a, b}) = |a * b| \,/\, GCD(\mathrm{a, b})$$

- Example :- find the LCM(354,144).

GCD(354,144)=GCD(144,354 mod 144)=GCD(144,66)

GCD(144,66)=GCD(66, 144 mod 66)= GCD(66,12)

GCD(66,12) =GCD(12, 66 mod 12)=GCD(12,6)

GCD(12,6)=GCD(6, 127 mod 6)=GCD(6,0)=6

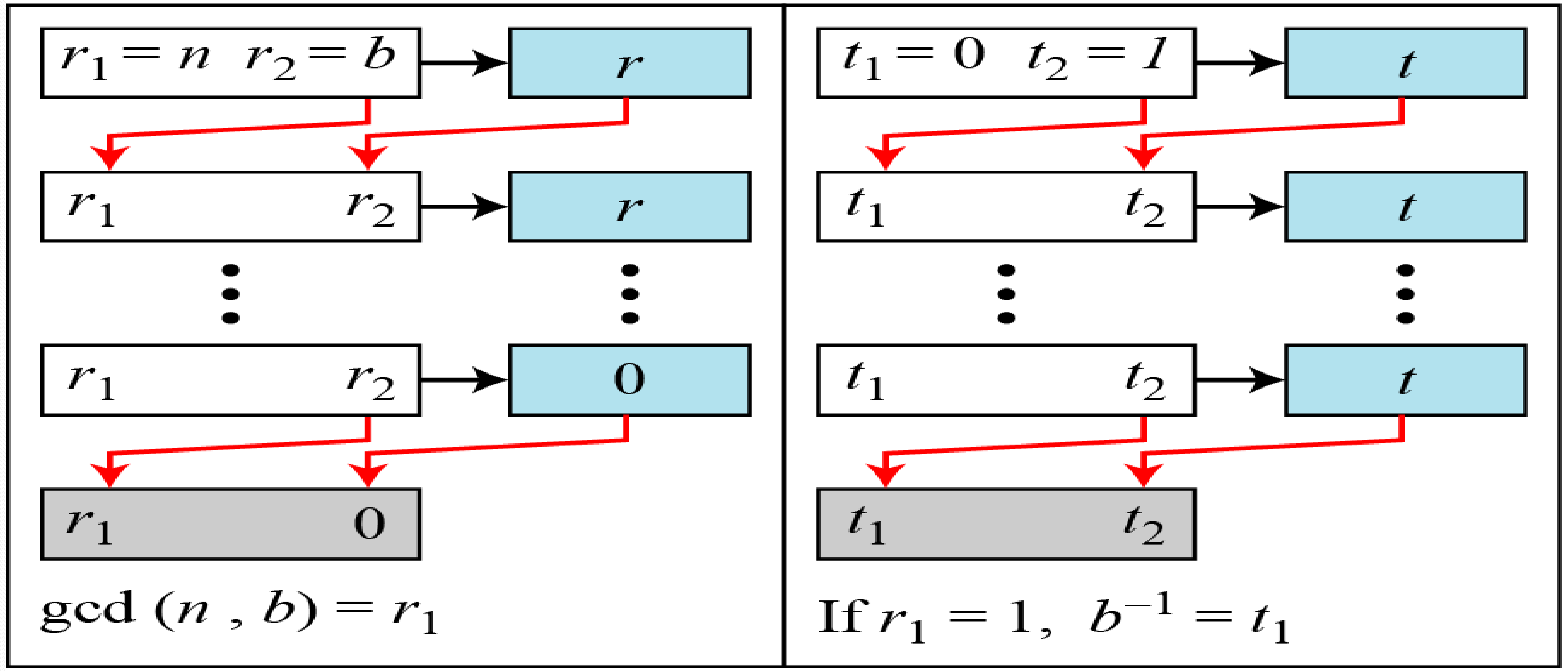
LCM(354,143)=(354*144)/6=8496

Multiplicative Inverse

- In Z_n, two numbers a and b are the multiplicative inverse of each other if

$$a \times b \equiv 1 \pmod{n}$$

- The extended Euclid nds the multiplicative inverses of b in Zn when n and b are given and gcd (n, b) = 1 as shown in this figure:

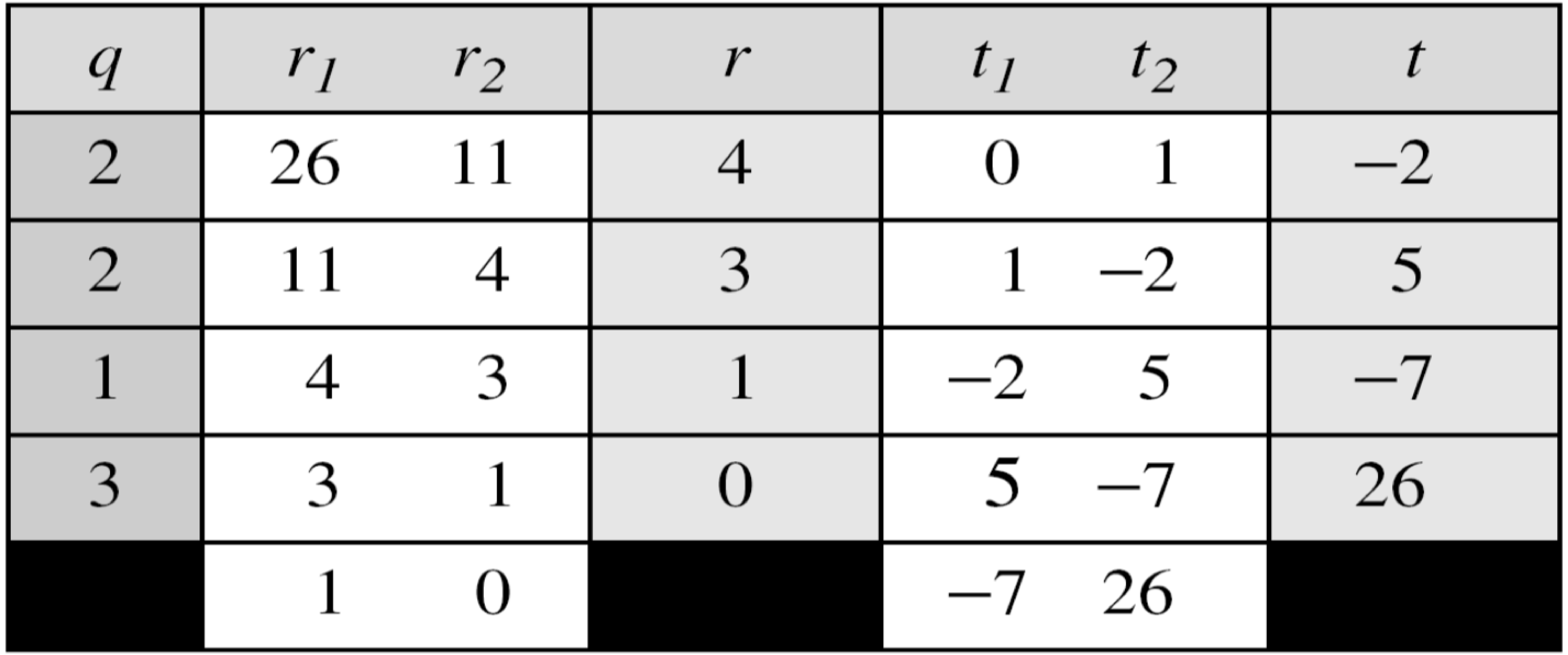


a. Process

```
$r_1 \leftarrow n$;        $r_2 \leftarrow b$;
$t_1 \leftarrow 0$;        $t_2 \leftarrow 1$;

while ($r_2 > 0$)
 {
  $q \leftarrow r_1 / r_2$;

  $r \leftarrow r_1 - q \times r_2$;
  $r_1 \leftarrow r_2$;        $r_2 \leftarrow r$;

  $t \leftarrow t_1 - q \times t_2$;
  $t_1 \leftarrow t_2$;        $t_2 \leftarrow t$;
 }
  if ($r_1 = 1$) then $b^{-1} \leftarrow t_1$
```

b. Algorithm

- Example: - Find the multiplicative inverse of 11 in Z_{26}.
- The GCD(26,11)must be 1 in order to find the inverse. Bu using the extended Euclidean algorithm, we can use this table



q	r_1	r_2	r	t_1	t_2	t
2	26	11	4	0	1	−2
2	11	4	3	1	−2	5
1	4	3	1	−2	5	−7
3	3	1	0	5	−7	26
	1	0		−7	26	

- the inverse of 11 is –7 mod 26=19.
- Or we can find the inverse based on using the equation $n = qn + r$

Example: - Find the multiplicative inverse of 11 in Z_{26}.

- $26=11*2+4$
- $11=4*2+3$
- $4=3*1+1$
- $3=3*1+0$
- We are now in reverse compensation starting from one as shown
- $1=4-(3*1)$
- $1=4-(11-(4*2))$
- $1=\underline{4}-11+\underline{4*2}$
- $1=3*4-11$
- $1=3*(26-11*2)-11$
- $1=3*26-6*11-11= 3*26\underline{-7}*11$ so the multiplicative inverse of 11 is -7

- Example :- Find the multiplicative inverse of 23 in Z_{100}.
- 100=23*4+8
- 23=8*2+7
- 8=7*1+1
- 7=1*7+0
- Now in revers way
- 1=8-(7*1)
- 1=8-(23-8*2)
- 1=8-23+8*2
- 1=3*8-23
- 1=3*(100-23*4)-23=3*100-12*23-23=3*100-$\underline{13}$*23 So the multiplicative inverse of 23 in Z_{100} is -23 or 87(-23 mod 100).

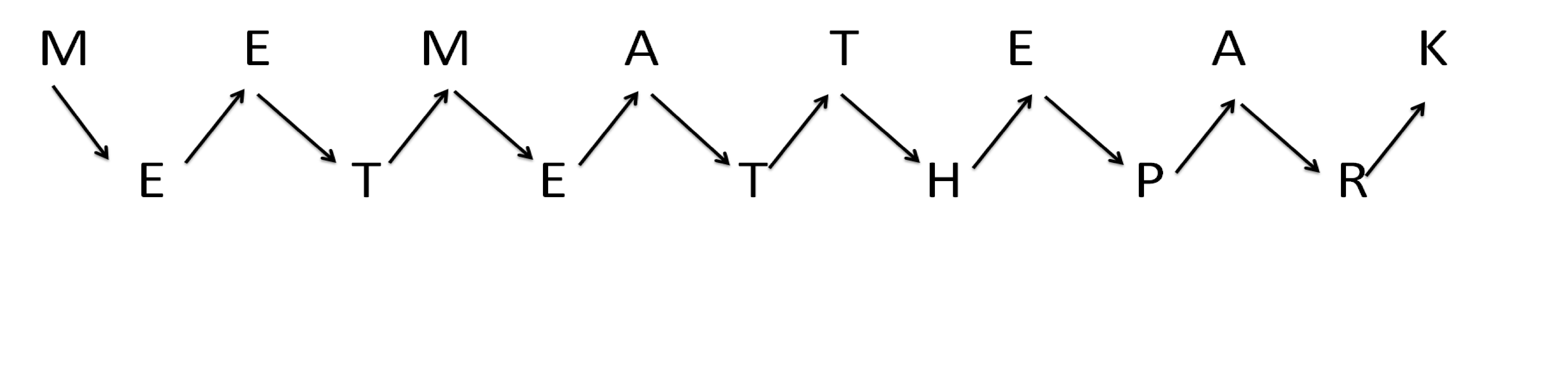
Chapter Three
Classical Symmetric Cipher

The forms of Encryption

- **Transposition (or permutation) cipher:** Transposition cipher keeps the letters the same, but rearranges their order according to a specific algorithm.
- **Substitution cipher:** replacing each element of the plaintext with another element.
- **Product cipher:** using multiple stages of substitutions and transpositions

Transposition cipher

1. **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**).

2. Columnar Transposition Ciphers.

- Write the message in rows of a fixed length, and then read out again column by column.
- The columns are chosen in some scrambled order.
- Both the length of the rows and the permutation of the columns are usually defined by a key.

Example: Let the plaintext is (WE ARE DISCOVERED FLEE AT ONCE) the key word be: ZEBRA.

Z	E	B	R	A
W	E	A	R	E
D	I	S	C	O
V	E	R	E	D
F	L	E	E	A
T	O	N	C	E

- The ciphertext:

EODAEASRENEIELORCEECWDVFT

Double Columnar Transposition.

Alice

enemyattackstonightz

Plaintext

Write row by row

Permute columns

Read column by column

ettheakimaotycnzntsg

Middle-text

Write row by row

Permute columns

Read column by column

Ciphertext

TIYTEAOZHMCSEANGYKTN

Send

Bob

TIYTEAOZHMCSEANGYKTN

Ciphertext

Write column by column

Permute columns

Read row by row

Middle-text

ettheakimaotycnzntsg

Write column by column

Permute columns

Read row by row

Plaintext

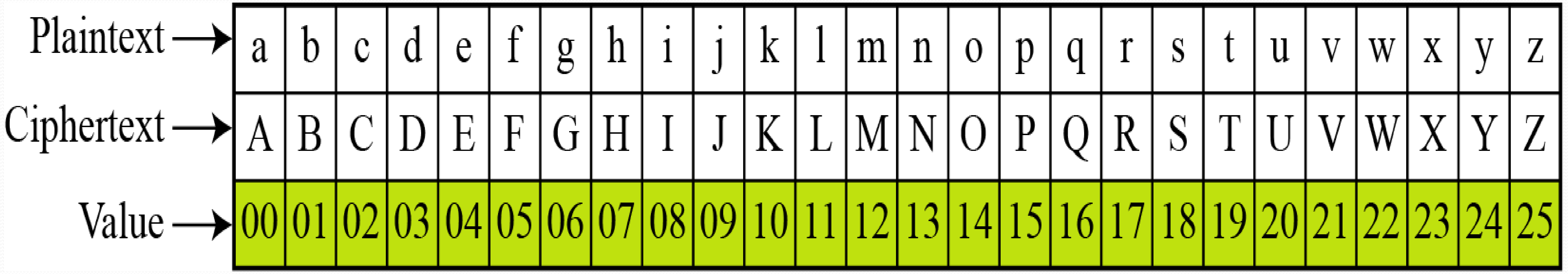
enemyattackstonightz

Substitution cipher

1. **Monoalphabetic Ciphers.**
   - It is simple substitution
   - involves replacing each letter in the message with another letter of the alphabet.
   - In monoalphabetic substitution, the relationship between a symbol in the plaintext to a symbol in the ciphertext is always one-to-one.

- **Additive Cipher:-** is the simplest monoalphabetic cipher. It is sometimes called a shift cipher and sometimes **a Caesar cipher**, but the term additive cipher better reveals its mathematical nature. When the cipher is additive, the plaintext, ciphertext, and key are integers in Z26.

Plaintext and ciphertext in Z26



Plaintext →	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
Ciphertext →	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
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Additive Cipher

Plaintext
Alice
P
$C = (P + k) \bmod 26$
k
Encryption
C
Ciphertext
Plaintext
Bob
P
k
$P = (C - k) \bmod 26$
C
Decryption

- ***Example***
- Use the additive cipher with key = 15 to encrypt the plain text (hello).
- We apply the encryption algorithm to the plaintext, character by character:

Plaintext h e l l o

7 4 11 11 14

Encryption

(7+15) mod 26=22→ W, (4+15) mod 26=19 →T, (11 +15) mod 26=0 →A, (11+15) mod 26=0 →A,(14+15) mod 26=3 →D

Ciphertext WTAAD

- We apply the decryption algorithm to the plaintext character by character:

Ciphertext

W T A A D

22 19 0 0 3

Decryption

(22-15) mod 26=7→ h, (19-15) mod 26=4 →e, (0-15) mod 26=11 →l, (0-15) mod 26=11 →l,(3-15) mod 26=14 →o

Ciphertext h e l l o

- **Caesar Cipher: -** Named for Julious Caesar. Caesar used a key of 3 for his communications.

Plaintext	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
Ciphertext	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

- **Cryptanalysis of the Caesar cipher: -**

- Example : - decrypt the following ciphertext:-
  wklv phvvdjh lv qrw wrr kdug wr euhdn
- By using the above table, replace the characters as show
  ciphertext = wklv phvvdjh lv qrw wrr kdug wr euhdn
  plaintext = THIS MESSAGE IS NOT TOO HARD TO BREAK

- `

- ***Example:*** Eve has intercepted the ciphertext (UVACLYFZLJBYL). Show how she can use a brute-force attack to break the cipher.
- Eve tries keys from 1 to 7. With a key of 7, the plaintext is (not very

Ciphertext: UVACLYFZLJBYL

K = 1 → **Plaintext:** tuzbkxeykiaxk

K = 2 → **Plaintext:** styajwdxjhzwj

K = 3 → **Plaintext:** rsxzivcwigyvi

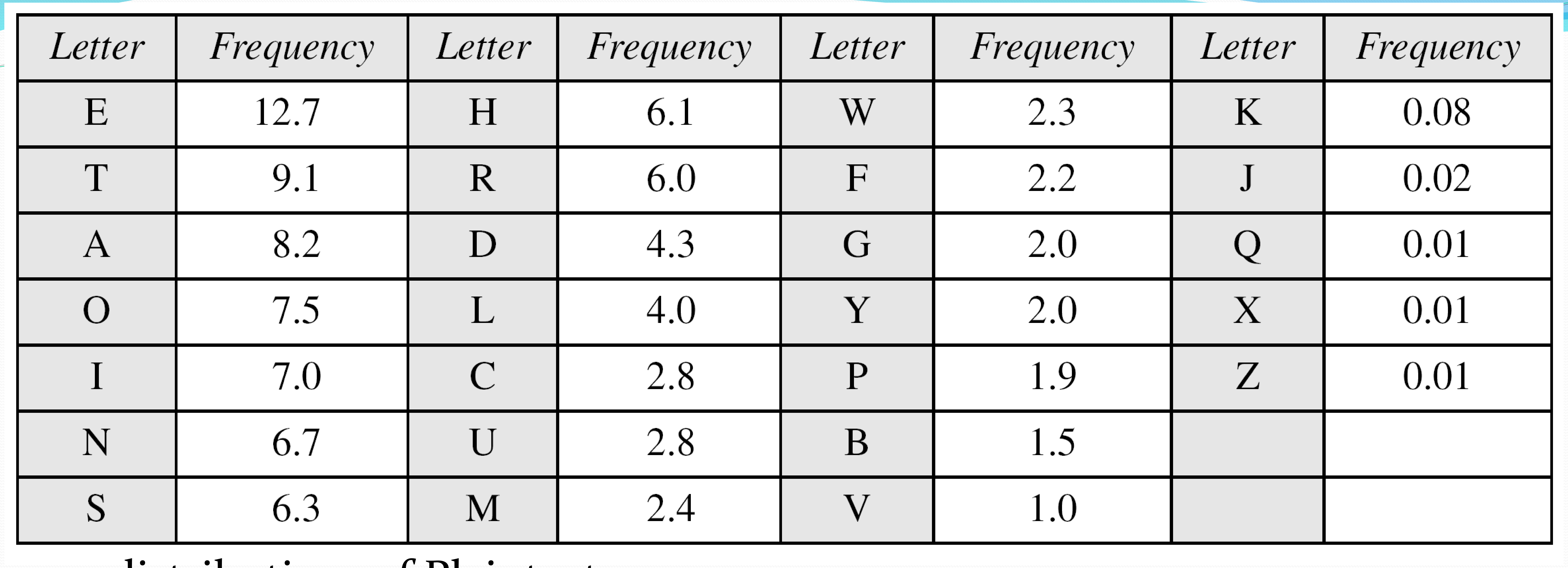
K = 4 → **Plaintext:** qrwyhubvhfxuh

K = 5 → **Plaintext:** pqvxgtaugewtg

K = 6 → **Plaintext:** opuwfsztfdvsf

K = 7 → **Plaintext:** notverysecure

Table of Frequency of characters in English



Letter	*Frequency*	*Letter*	*Frequency*	*Letter*	*Frequency*	*Letter*	*Frequency*
E	12.7	H	6.1	W	2.3	K	0.08
T	9.1	R	6.0	F	2.2	J	0.02
A	8.2	D	4.3	G	2.0	Q	0.01
O	7.5	L	4.0	Y	2.0	X	0.01
I	7.0	C	2.8	P	1.9	Z	0.01
N	6.7	U	2.8	B	1.5		
S	6.3	M	2.4	V	1.0		

Frequency distributions of Plaintext :-

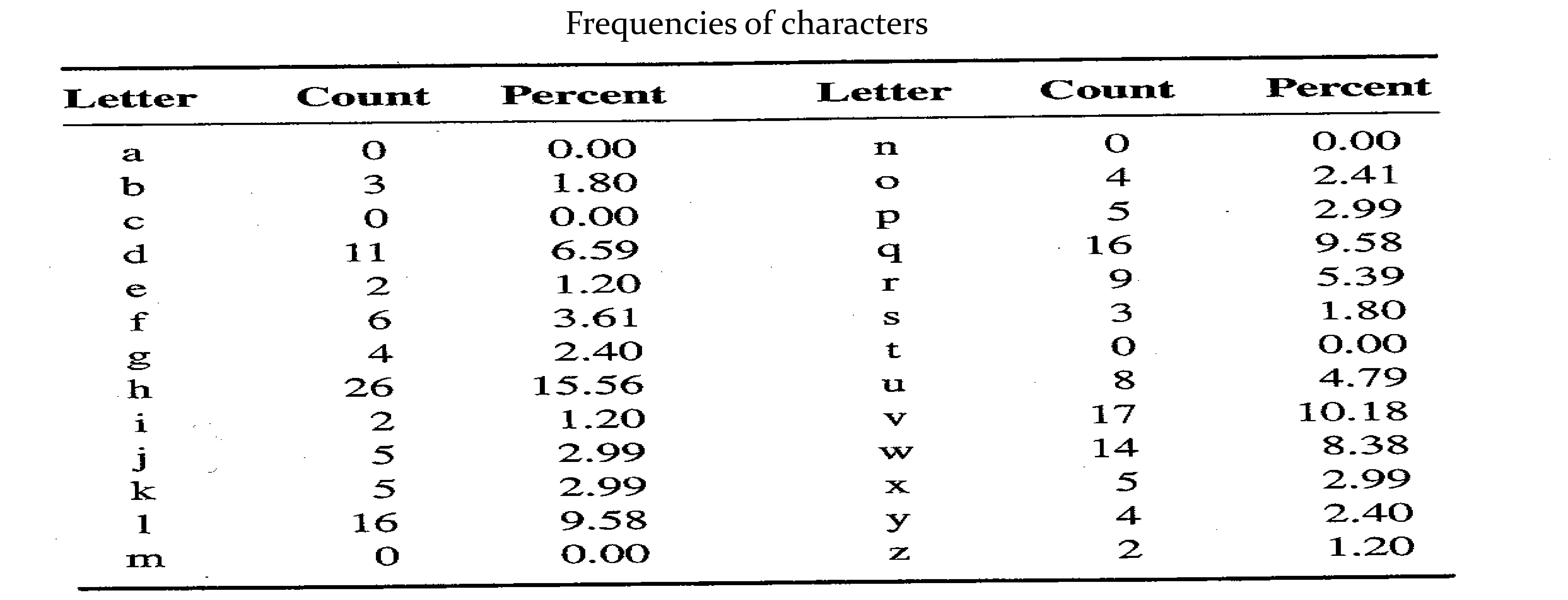
- E
- T
- A, O, R, N , I
- H , C , D , L, M
- .
- .
- X , J ,Z , Q

- Example : - Eve has intercepted the following ciphertext. Using a statistical attack, find the plaintext.

Ciphertext= hqfubswlrq lv d phdqv ri dwwdlqlqj vhfxuh frppxulfdwlrq

- When Eve tabulates the frequency of letters in this ciphertext, she gets:

h=26, v=17 and so on.



Frequencies of characters

Letter	Count	Percent	Letter	Count	Percent
a	0	0.00	n	0	0.00
b	3	1.80	o	4	2.41
c	0	0.00	p	5	2.99
d	11	6.59	q	16	9.58
e	2	1.20	r	9	5.39
f	6	3.61	s	3	1.80
g	4	2.40	t	0	0.00
h	26	15.56	u	8	4.79
i	2	1.20	v	17	10.18
j	5	2.99	w	14	8.38
k	5	2.99	x	5	2.99
l	16	9.58	y	4	2.40
m	0	0.00	z	2	1.20

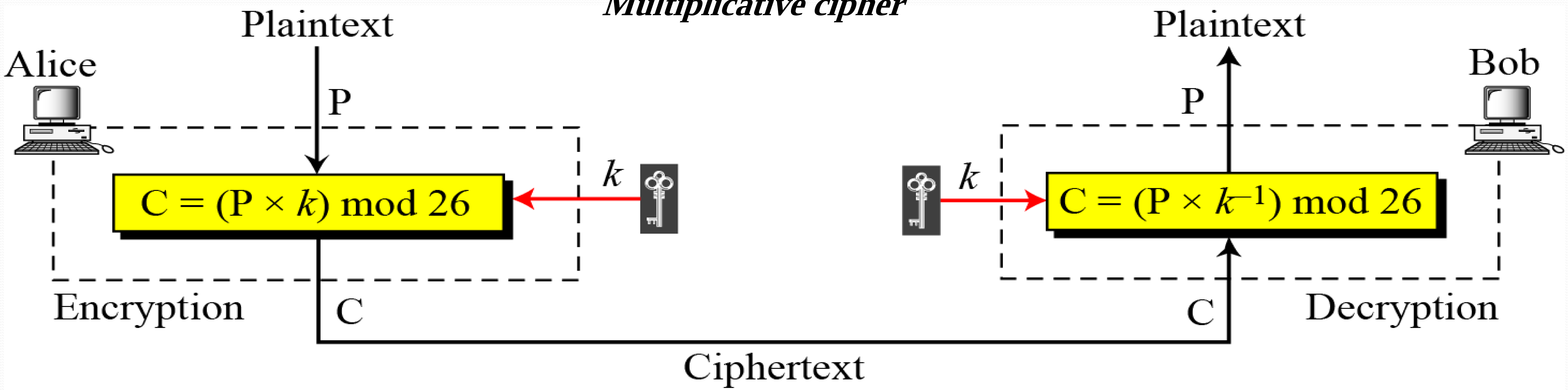
- So we will replace each character with the corresponding high frequency in plaintext as shown: -

Plaintext = ENCRYPTION IS A MEANS OF ATTAINING SECURE COMMUNICATION

Which means that the key is =3 ? How?

- **Multiplicative Ciphers:** - In a multiplicative cipher, the plaintext and ciphertext are integers in Z_{26}; the key is an integer in Z_{26}^*.

Multiplicative cipher



- The key domain for any multiplicative cipher which must be in Z26*, is the set that has only 12 members: 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25.(**why**)
- Example: - We use a multiplicative cipher to encrypt the message “hello” with a key of 7. The ciphertext is “XCZZU”.

Plaintext: h → 07	Encryption: (07 × 07) mod 26	ciphertext: 23 → X
Plaintext: e → 04	Encryption: (04 × 07) mod 26	ciphertext: 02 → C
Plaintext: l → 11	Encryption: (11 × 07) mod 26	ciphertext: 25 → Z
Plaintext: l → 11	Encryption: (11 × 07) mod 26	ciphertext: 25 → Z
Plaintext: o → 14	Encryption: (14 × 07) mod 26	ciphertext: 20 → U

- Cryptanalyses of the multiplicative cipher based on finding the multiplication inverse of the key (where the multiplication inverse of **7 is 15**) as shown

Ciphertext X → 23	Decryption: (23 * 15) mod 26	plaintext= 7→h
Ciphertext C → 2	Decryption: (2 * 15) mod 26	plaintext= 4→e
Ciphertext Z → 25	Decryption: (25 * 15) mod 26	plaintext=11→l
Ciphertext Z → 25	Decryption: (25 * 15) mod 26	plaintext=11→l
Ciphertext U → 20	Decryption: (20 * 15) mod 26	plaintext=14→o

- Affine Ciphers

$$C = (P \times k_1 + k_2) \bmod 26 \qquad P = ((C - k_2) \times k_1^{-1}) \bmod 26$$

where k_1^{-1} is the multiplicative inverse of k_1 and $-k_2$ is the additive inverse of k_2

Alice
Plaintext
P
$T = (P \times k_1) \bmod 26$
$C = (T + k_2) \bmod 26$
Encryption
C
k_1
k_2
key
key
k_1
k_2
Plaintext
Bob
P
$P = (T \times k_1^{-1}) \bmod 26$
$T = (C - k_2) \bmod 26$
C
Decryption
Ciphertext

- The affine cipher uses a pair of keys in which the first key is from Z_{26}^* and the second is from Z_{26}. The size of the key domain is 26 × 12 = 312.
- The additive cipher is a special case of an affine cipher in which $k_1 = 1$. The multiplicative cipher is a special case of affine cipher in which $k_2 = 0$.

- Example: - Use an affine cipher to encrypt the message “hello” with the key pair (7, 2).

P: h → 07	Encryption: $(07 \times 7 + 2) \bmod 26$	C: 25 → Z
P: e → 04	Encryption: $(04 \times 7 + 2) \bmod 26$	C: 04 → E
P: l → 11	Encryption: $(11 \times 7 + 2) \bmod 26$	C: 01 → B
P: l → 11	Encryption: $(11 \times 7 + 2) \bmod 26$	C: 01 → B
P: o → 14	Encryption: $(14 \times 7 + 2) \bmod 26$	C: 22 → W

- To decrypt the message “ZEBBW” with the key pair (7, 2) in modulus 26. where where the multiplication inverse of **7 is 15**

C: Z → 25	Decryption: $((25 - 2) \times 7^{-1}) \bmod 26$	P:07 → h
C: E → 04	Decryption: $((04 - 2) \times 7^{-1}) \bmod 26$	P:04 → e
C: B → 01	Decryption: $((01 - 2) \times 7^{-1}) \bmod 26$	P:11 → l
C: B → 01	Decryption: $((01 - 2) \times 7^{-1}) \bmod 26$	P:11 → l
C: W → 22	Decryption: $((22 - 2) \times 7^{-1}) \bmod 26$	P:14 → o

2. Polyalphabetic Ciphers

- In polyalphabetic substitution, each occurrence of a character may have a different substitute.
- The relationship between a character in the plaintext to a character in the ciphertext is one-to-many.
- **Autokey Cipher: -**

$P = P_1P_2P_3 \ldots$ $C = C_1C_2C_3 \ldots$ $k = (k_1, P_1, P_2, \ldots)$

Encryption: $C_i = (P_i + k_i) \bmod 26$ Decryption: $P_i = (C_i - k_i) \bmod 26$

- Assume that Alice and Bob agreed to use an autokey cipher with initial key value k1 = 12. Now Alice wants to send Bob the message “Attack is today”. Enciphering is done character by character as shown :-

Plaintext:	a	t	t	a	c	k	i	s	t	o	d	a	y
P's Values:	00	19	19	00	02	10	08	18	19	14	03	00	24
Key stream:	*12*	*00*	*19*	*19*	*00*	*02*	*10*	*08*	*18*	*19*	*14*	*03*	*00*
C's Values:	12	19	12	19	02	12	18	00	11	7	17	03	24
Ciphertext:	**M**	**T**	**M**	**T**	**C**	**M**	**S**	**A**	**L**	**H**	**R**	**D**	**Y**

- **Vigenere Cipher: -**

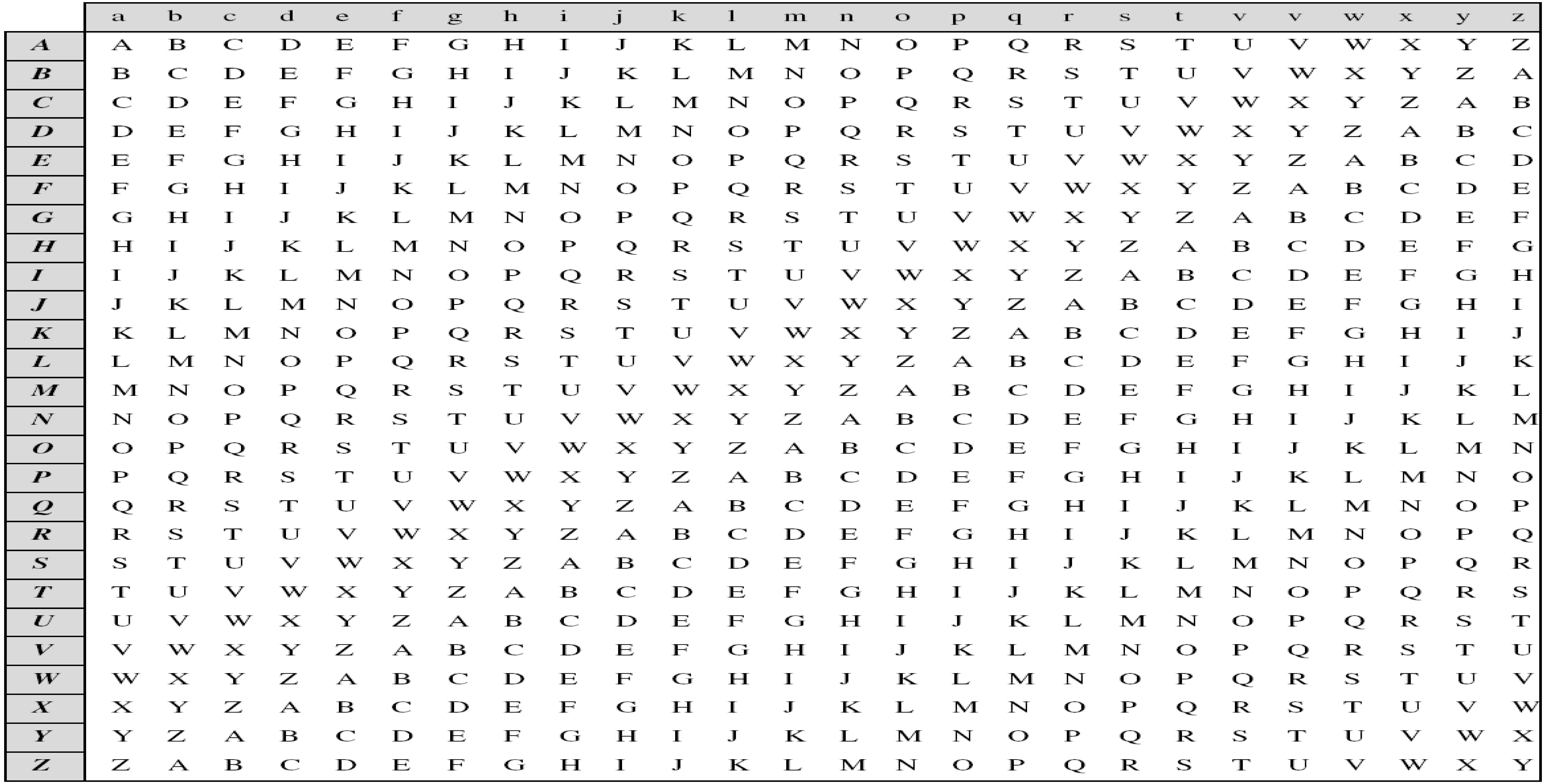
$P = P_1P_2P_3 \ldots$ $\quad C = C_1C_2C_3 \ldots$ $\quad K = [(k_1, k_2, \ldots, k_m), (k_1, k_2, \ldots, k_m), \ldots]$

Encryption: $C_i = P_i + k_i$ $\quad$ Decryption: $P_i = C_i - k_i$

- **Example:** - We can encrypt the message "She is listening" using the 6-character keyword "PASCAL".

Plaintext:	s	h	e	i	s	l	i	s	t	e	n	i	n	g
P's values:	18	07	04	08	18	11	08	18	19	04	13	08	13	06
Key stream:	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***	***18***	***02***	***00***	***11***	***15***	***00***
C's values:	**07**	**07**	**22**	**10**	**18**	**22**	**23**	**18**	**11**	**6**	**13**	**19**	**02**	**06**
Ciphertext:	**H**	**H**	**W**	**K**	**S**	**W**	**X**	**S**	**L**	**G**	**N**	**T**	**C**	**G**

- Vigenere cipher can be seen as combinations of m additive ciphers. As shown in a Vigenere Tableau which can be used to find ciphertext which the intersection of a row and column.



	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

- **Running Key:** -Exactly Vigenère Cipher but the key length is exactly same length of the plaintext, usually keys are determined from books known from both sender and receiver.

3. Polygraphic Ciphers

- Instead of substituting one letter for another letter, a polygraphic cipher performs substitutions with two or more groups of letters.
- This has the advantage of masking the frequency distribution of letters, which makes frequency analysis attackes much more difficult.
- **Playfair Cipher:-** You create 5x5 matrix based on a keyword with the reset of the alphabets characters. For example a keyword (without repetition) such as "PROBLEMS":

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

- In this cipher, we will encipher letters pairs at a time. Consider the following plaintext:

  SHE WENT TO THE STORE

- When we pair up the letters they get grouped as follows:

  SH EW EN TT OT HE ST OR E

- But, we are not allowed to encipher any double letters. So, in this case, we will insert an Q into the plaintext. (If Q is a double letter, then insert another infrequent letter, say X.)

  SH EW EN TQ TO TH ES TO RE

- To encipher pairs of letters, adhere to the following rules:

1. If the two letters are on the same row of the chart, like "ES", then replace each letter by the letter to the right. (If necessary, wrap around to the left end of the row. So "ES" encrypts to "MA".

2. If the two letters are on the same column of the chart, like, "TH", then replace each letter by the letter below it. (If necessary, wrap around to the top end of the column.) So "TH" encrypts to "YT".

3. If two letters are on a different row and column, like, "SH", then replace each letter by another letter on its same row, but in the column of the other letter. So "SH" encrypts to "AG".

- Using these rules, here is the encryption of the plaintext above:

  Plaintext : SH EW EN TQ TO TH ES TO RE

  Ciphertext: AG MV MK UT QB YT MA QB PM

- To decipher, ignore rule 1. In rules 2 and 3 shift up and left instead of down and right. Rule 4 remains the same. Once you are done, drop any extra Xs that don't make sense in the final message and locate any missing Qs or any Is that should be Js.

- **Hill Cipher:** - The Hill Cipher uses matrix multiplication to encrypt a message.
- First, you need to assign two numbers to each letter in the alphabet and also assign numbers to space, . , and ? or !.
- The key space is the set of all invertible matrices over Z_{26}. 26 was chosen because there are 26 characters, which solves some problems later on.

$$K = \begin{bmatrix} k_{11} & k_{12} & \dots & k_{1m} \\ k_{21} & k_{22} & \dots & k_{2m} \\ \vdots & \vdots & & \vdots \\ k_{m1} & k_{m2} & \dots & k_{mm} \end{bmatrix}$$

$$\begin{aligned} C_1 &= P_1 k_{11} + P_2 k_{21} + \cdots + P_m k_{m1} \\ C_2 &= P_1 k_{12} + P_2 k_{22} + \cdots + P_m k_{m2} \\ &\dots \\ C_m &= P_1 k_{1m} + P_2 k_{2m} + \cdots + P_m k_{mm} \end{aligned}$$

- The key matrix in the Hill cipher needs to have a multiplicative inverse as shown :
- For example, the plaintext “code is ready” can make a 3 × 4 matrix when adding extra bogus character “z” to the last block and removing the spaces.

- For example, the plaintext “code is ready” can make a 3 × 4 matrix when adding extra bogus character “z” to the last block and removing the spaces.

$$\overset{\mathbf{C}}{\begin{bmatrix} 14 & 07 & 10 & 13 \\ 08 & 07 & 06 & 11 \\ 11 & 08 & 18 & 18 \end{bmatrix}} = \overset{\mathbf{P}}{\begin{bmatrix} 02 & 14 & 03 & 04 \\ 08 & 18 & 17 & 04 \\ 00 & 03 & 24 & 25 \end{bmatrix}} \overset{\mathbf{K}}{\begin{bmatrix} 09 & 07 & 11 & 13 \\ 04 & 07 & 05 & 06 \\ 02 & 21 & 14 & 09 \\ 03 & 23 & 21 & 08 \end{bmatrix}}$$

a. Encryption

$$\overset{\mathbf{P}}{\begin{bmatrix} 02 & 14 & 03 & 04 \\ 08 & 18 & 17 & 04 \\ 00 & 03 & 24 & 25 \end{bmatrix}} = \overset{\mathbf{C}}{\begin{bmatrix} 14 & 07 & 10 & 13 \\ 08 & 07 & 06 & 11 \\ 11 & 08 & 18 & 18 \end{bmatrix}} \overset{\mathbf{K}^{-1}}{\begin{bmatrix} 02 & 15 & 22 & 03 \\ 15 & 00 & 19 & 03 \\ 09 & 09 & 03 & 11 \\ 17 & 00 & 04 & 07 \end{bmatrix}}$$

b. Decryption

- The ciphertext is “OHKNIHGKLISS”.
-

- **One-Time Pad**: - One of the goals of cryptography is perfect secrecy. A study by Shannon has shown that perfect secrecy can be achieved if each plaintext symbol is encrypted with a key randomly chosen from a key domain. This idea is used in a cipher called one-time pad, invented by **Vernam.**
- **Example**:- Plaintext VERNAMCIPHER
- **Key** **76 48 16 82 44 3 58 11 60 5 48 88**
- **Encryption**

Plaintext **21 4 17 13 0 12 2 8 15 7 4 17**

+

Key **76 48 16 82 44 3 58 11 60 5 48 88**

Ciphertext **97 52 33 95 44 15 60 19 75 12 52 105 mod 26**

19 0 7 17 18 15 8 19 23 12 0 1

t a h r s p i t x m a b

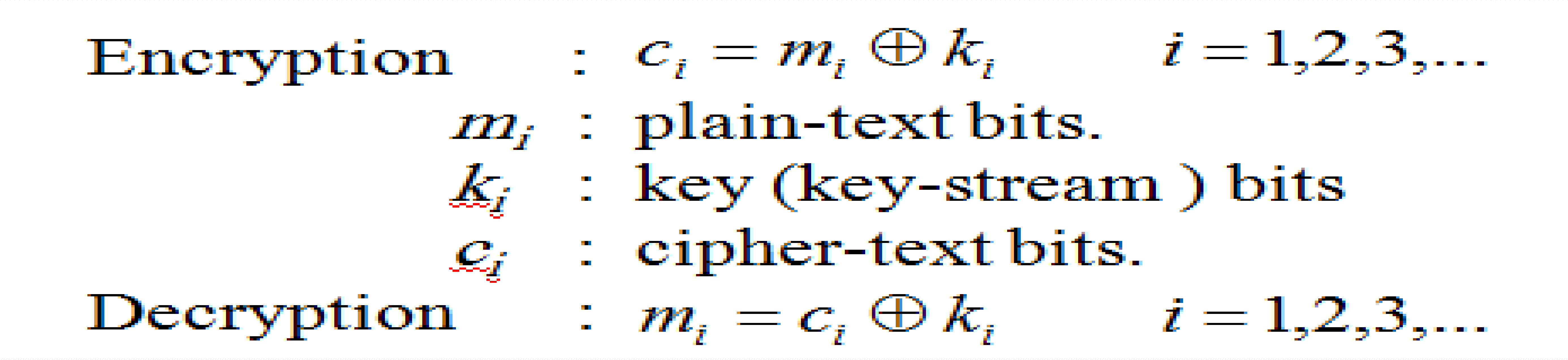
- **Decryption**

Ciphertext	t	a	h	r	s	p	i	t	x	m	a	b	
	19	0	7	17	18	15	8	19	23	12	0	1	
												-	
Key	76	48	16	82	44	3	58	11	60	5	48	88	
	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
plaintext	-57	-48	-9	-65	-26	12	-50	8	-37	7	-48	-87	mod 26
	21	4	17	13	0	12	2	8	15	7	4	17	
	V	E	R	N	A	M	C	I	P	H	E	R	

Chapter Four
Modern Symmetric Ciphers
(Stream Cipher and Block Cipher)

Stream cipher

- Basic Idea of stream cipher comes from One-Time-Pad cipher: -



Encryption : $c_i = m_i \oplus k_i \qquad i = 1,2,3,\ldots$

m_i : plain-text bits.

k_i : key (key-stream) bits

c_i : cipher-text bits.

Decryption : $m_i = c_i \oplus k_i \qquad i = 1,2,3,\ldots$

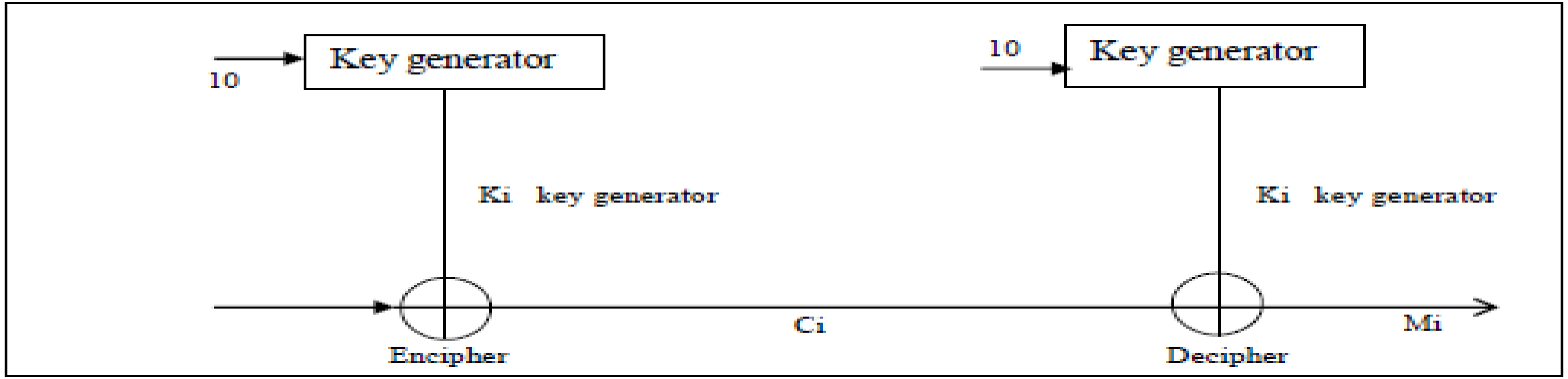
- The **drawback of** One-Time-Pad cipher is that the key-stream should be as long as plain-text. Key distribution & Management difficult.
- **Stream Cipher** is the solution (in which key-stream is generated in pseudo-random fashion from relatively short *secret key*.
- **Pseudo-randomness :** sequences appears random to a computationally bounded adversary.

(i) Encryption
(ii) Decryption
Plaintext m_i
Ciphertext c_i
Key k
Keystream z_i
State σ_i
σ_{i+1}
σ_i
f
g
h
h^{-1}
k
z_i
m_i
c_i

There are two different approaches to stream encryption they are; **synchronous methods** and **self-synchronous methods**.

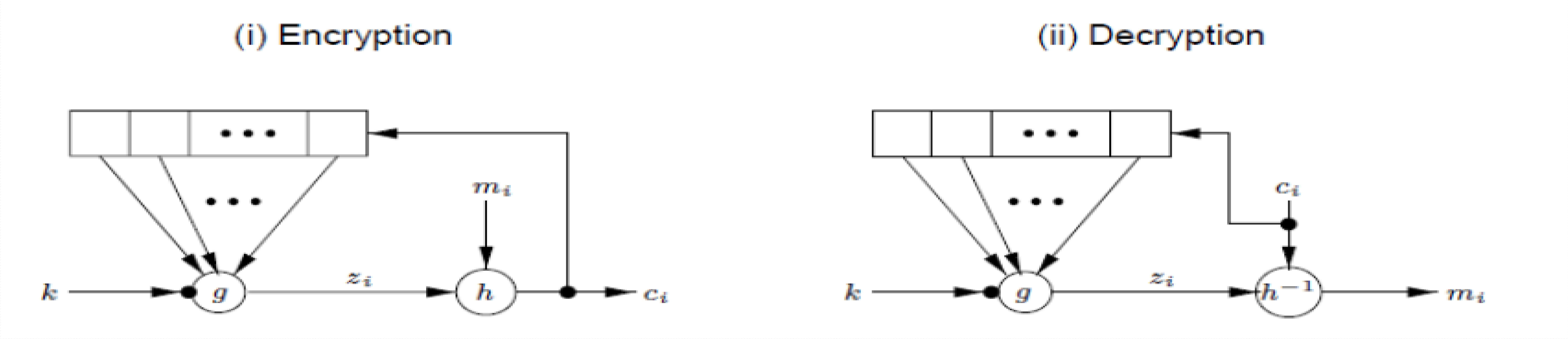
1. Synchronous Stream Ciphers

- Key-stream is independent of plain and cipher-text.
- Both sender &receiver must be synchronized.
- Resynchronization can be needed (This means that if a ciphertext is lost during transmission, the sender and receiver must resynchronize their key generators before they can proceed).
- Synchronous stream ciphers have the advantage of not propagating errors. A transmission error effecting one character will not affect subsequent characters. From another point of view; this is a disadvantage in that it is easier for an opponent to modify (with out detection) a single ciphertext character.
- Active attacks can easily be detected (disadvantage)

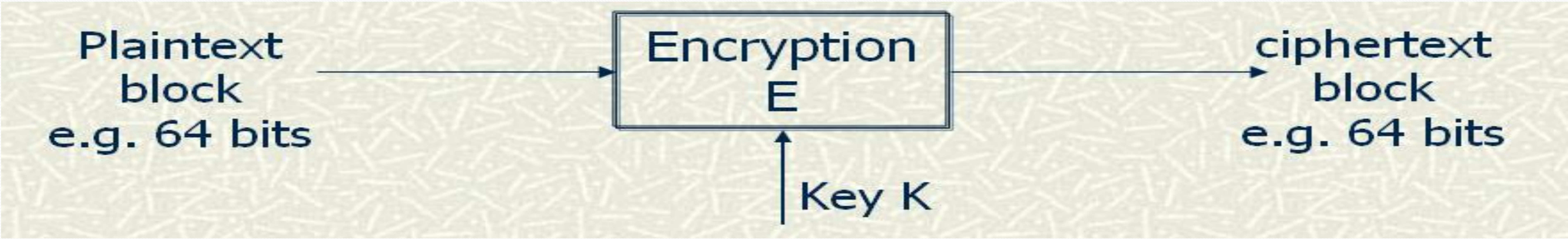


2. Self-Synchronizing Stream Ciphers

- Key-stream is a function of fixed number *t* of cipher-text bits. This is done by using a cipher feed back mode (CFB) because the ciphertext characters participate in the feed back loop.
- It is some times called **chaining**, because each ciphertext character depend on preceding cipher-text character (chain) the feed back
- Limited error propagation (up to *t* bits).
- Active attacks cannot be detected.
- At most *t* bits later, it resynchronizes itself when synchronization is lost.
- It helps to diffuse plain-text statistics**.**



- **Block cipher :** - an encryption scheme that encrypts a block of clear text into a block of cipher text of the same length. In this case, a block cipher can be viewed as a simple substitute cipher with character size equal to the block size.



Plaintext block e.g. 64 bits

Encryption E

Key K

ciphertext block e.g. 64 bits

- **Block cipher operation modes: -**

1. **ECB Operation Mode.**

- ECB stands for **Electronic Code Book.** Blocks of clear text are encrypted independently.
- Strength: it's simple.
- Weakness :

  1- Repetitive information contained in the plaintext may show in the ciphertext, if aligned with blocks.

  2. If the same message is encrypted (with the same key) and sent twice, their ciphertext are the same.
- Typical application: secure transmission of short pieces of information (e.g. a temporary encryption key)

Encryption: $C_i = E_K (P_i)$ Decryption: $P_i = D_K (C_i)$

E: Encryption D: Decryption

P_i: Plaintext block i C_i: Ciphertext block i

K: Secret key

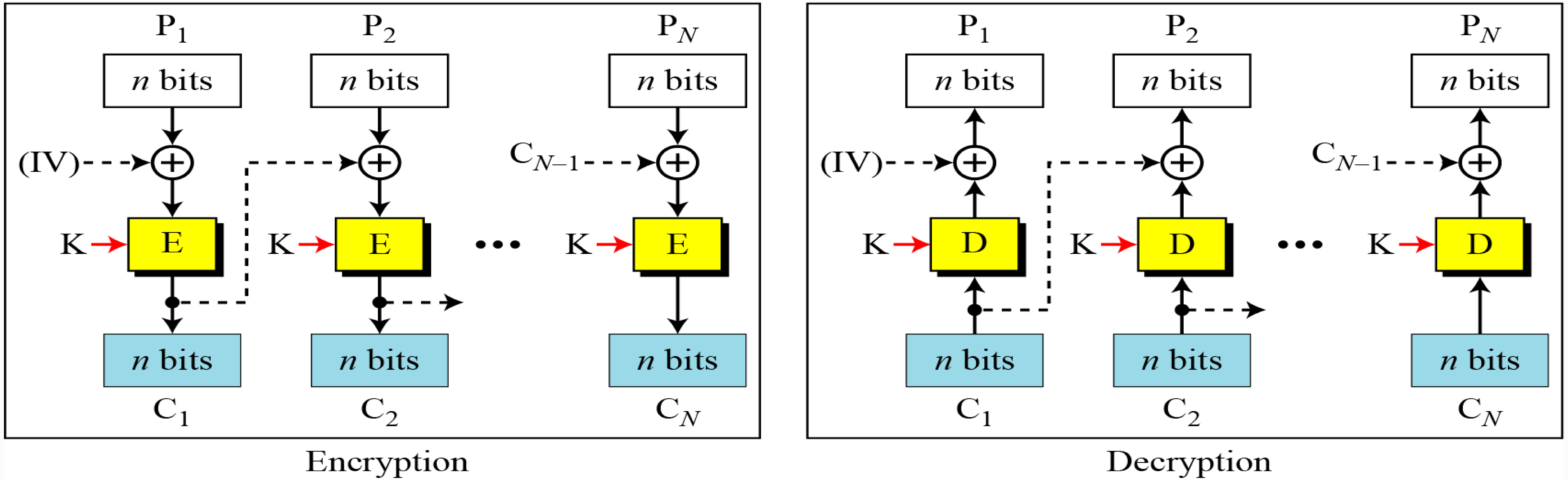
P_1
P_2
P_N
n bits
K
E
D
•••
C_1
C_2
C_N
Encryption
Decryption

2. CBC Operation Mode.

- **CBC** stands **for Cipher-Block Chaining** The previous cipher text block is XORed with the clear text block before applying the encryption mapping.
- Solve security deficiencies in ECB where Repeated same plaintext block result different ciphertext block
- Use Initial Vector (IV) to start process

Ci = EK (Pi XOR Ci-1)

E: Encryption D : Decryption
P_i: Plaintext block *i* C_i : Ciphertext block *i*
K: Secret key IV: Initial vector (C_0)



3. Cipher FeedBack (CFB).

- Message is treated as a stream of bits , Bitwise-added to the output of the block cipher , Result is feedback for next stage (hence name).its Uses for stream data encryption, authentication
- Use Initial Vector to start process.
- Plaintext is treated as a stream of bits. Any number of bit (1, 8 or 64 or whatever) to be feed back (denoted CFB-1, CFB-8, CFB-64)

IV
Shift register
64 − s bits
s bits
64
K
Encrypt
64
Select
s bits
Discard
64 − s bits
s
P1
s
C1
s
P2
C2
CM−1
PM
CM
• • •
IV
Shift register
64 − s bits
s bits
K
Encrypt
Select
j bits
Discard
64 − j bits
s
C1
P1
P2
C2
Decrypt
CM
PM

(a) Encryption

(b) Decryption

4. Output Feedback Mode (OFM).

- The block cipher is used as a stream cipher, it produces the random key stream.
- Very similar to CFB But output of the encryption function output of cipher is fed back (hence name), instead of ciphertext.

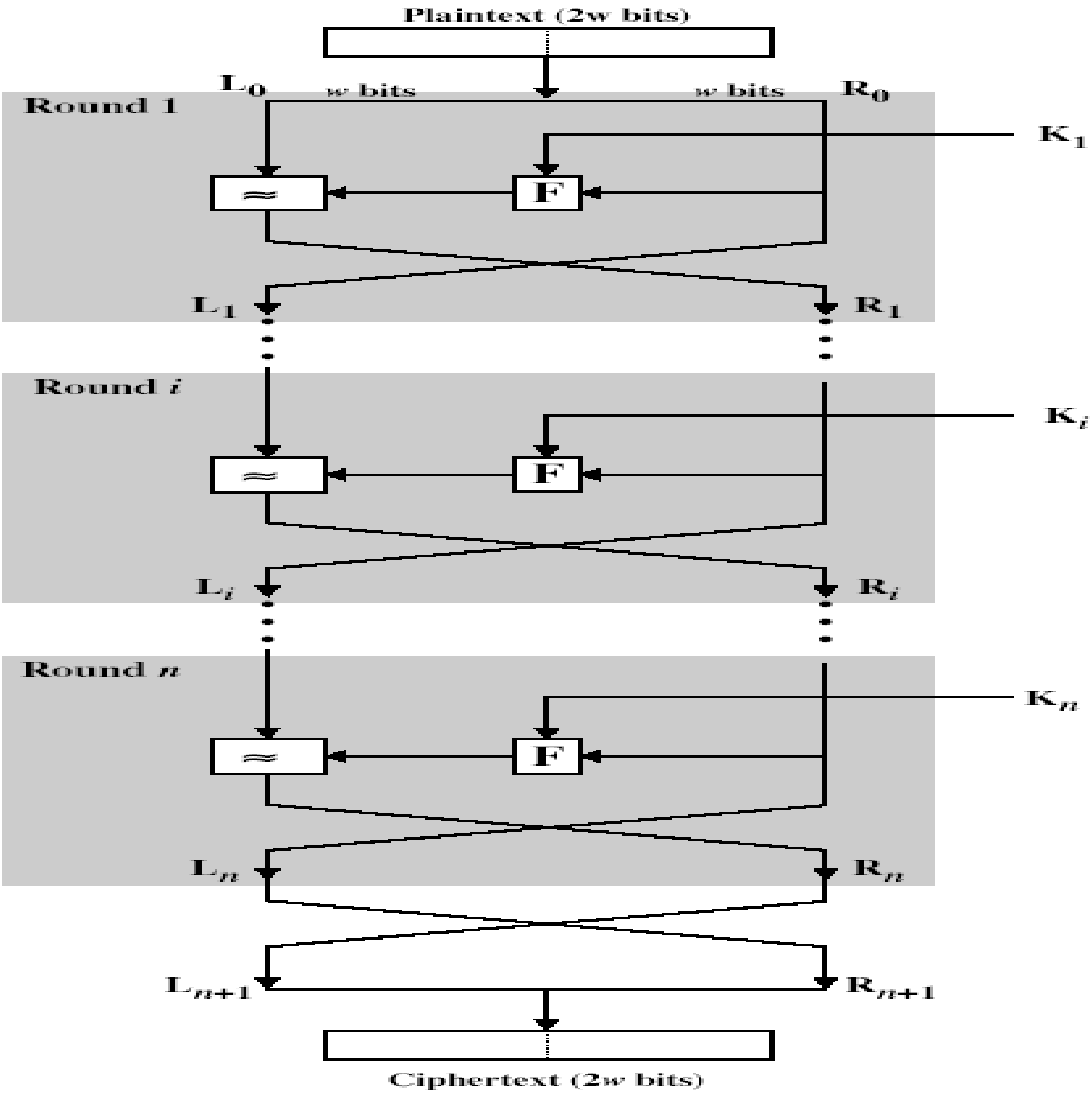
IV
Shift register
64 – s bits
s bits
64
K
Encrypt
64
Select
s bits
Discard
64 – s bits
P1
s
C1
P2
C2
O_{M–1}
P_M
C_M
• • •
IV
Shift register
64 – s bits
s bits
K
Encrypt
Select
j bits
Discard
64 – j bits
s
C1
P1
C2
P2
O_{M–1}
Decrypt
C_M
P_M

(a) Encryption

(b) Decryption

- **Product Cipher** - An encryption scheme that "uses multiple ciphers in which the cipher text of one cipher is used as the clear text of the next cipher". Usually, substitution ciphers and transposition ciphers are used alternatively to construct a product cipher.
- **Iterated Block Cipher** - A block cipher that "iterates a fixed number of times of another block cipher, called round function, with a different key, called round key, for each iteration".
- Most symmetric block ciphers are based on a **Feistel Cipher Structure.**
- **Feistel Cipher** - An iterate block cipher that Process through multiple rounds which
  - partitions input block into two halves
  - perform a substitution on left data half
  - based on round function of right half & sub key
  - then have permutation swapping halves

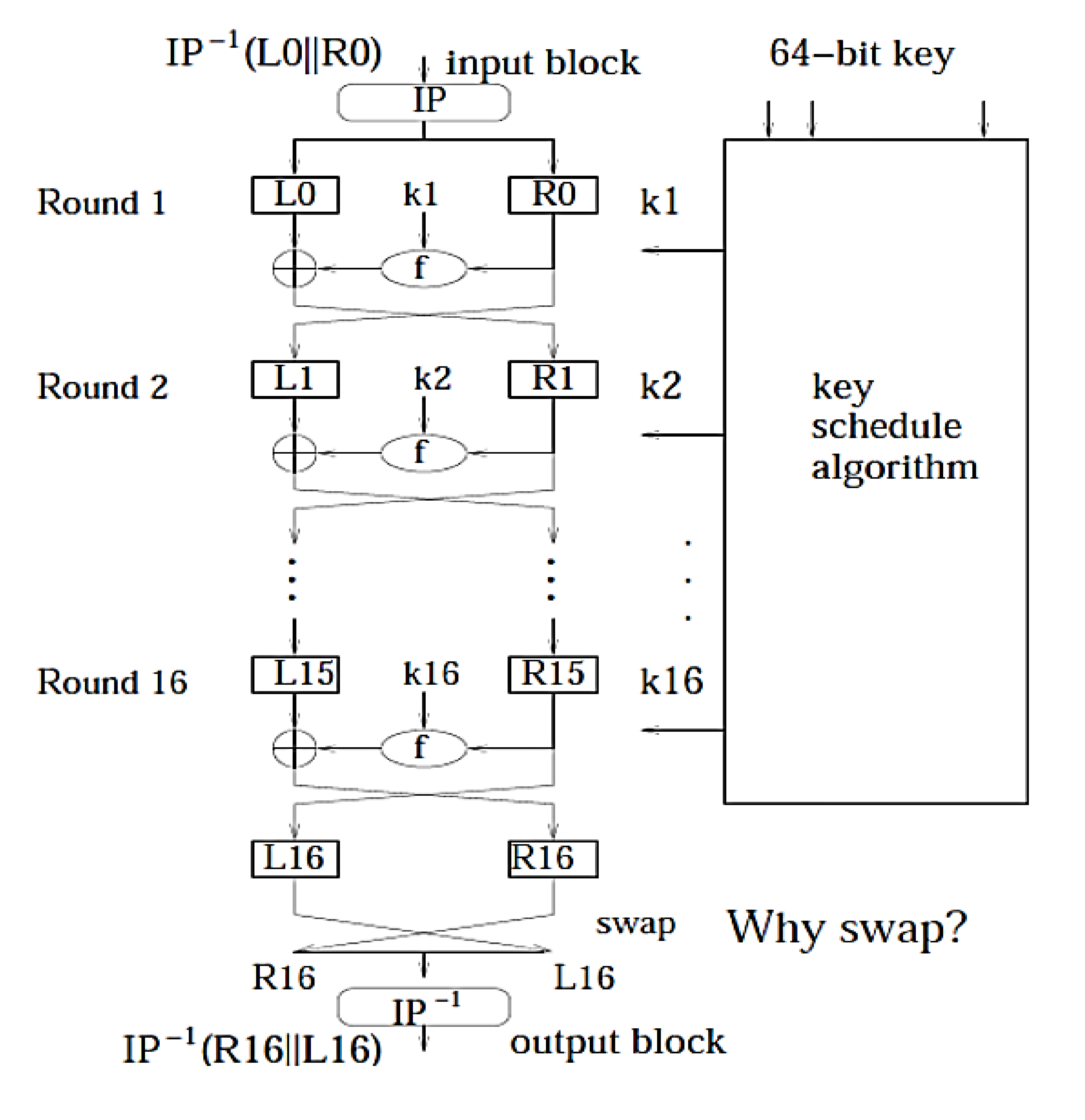
Feistel Cipher Structure



Data Encryption Standard (DES)

- A 16-round Feistel cipher with block size of 64 bits.
- Published in 1977, standardized in 1979.
- Key: 64 bit quantity = 8-bit parity+56-bit key
  - every eighth bit is used for parity checking and is ignored.
- It encrypts 64-bit data, and uses 56-bit key with 16 48-bit sub-keys.
- DES is a symmetric algorithm: The same algorithm and key are used for both encryption and decryption (except for minor differences in the key schedule).
- DES is a single combination of these techniques (a substitution followed by a permutation) on the text, based on the key.
- All security rests within the key.
- The algorithm is nothing more than a combination of the two basic techniques of encryption: **confusion** and diffusion.
  - **Diffusion** – dissipates statistical structure of plaintext over bulk of ciphertext(spreading of the effect of a change in the plaintext to many parts of the ciphertext).
  - **Confusion** – makes relationship between ciphertext and key as complex as possible (difficulty in determining how a change in the plaintext will affect the ciphertext).

DES Algorithm



Why swap?

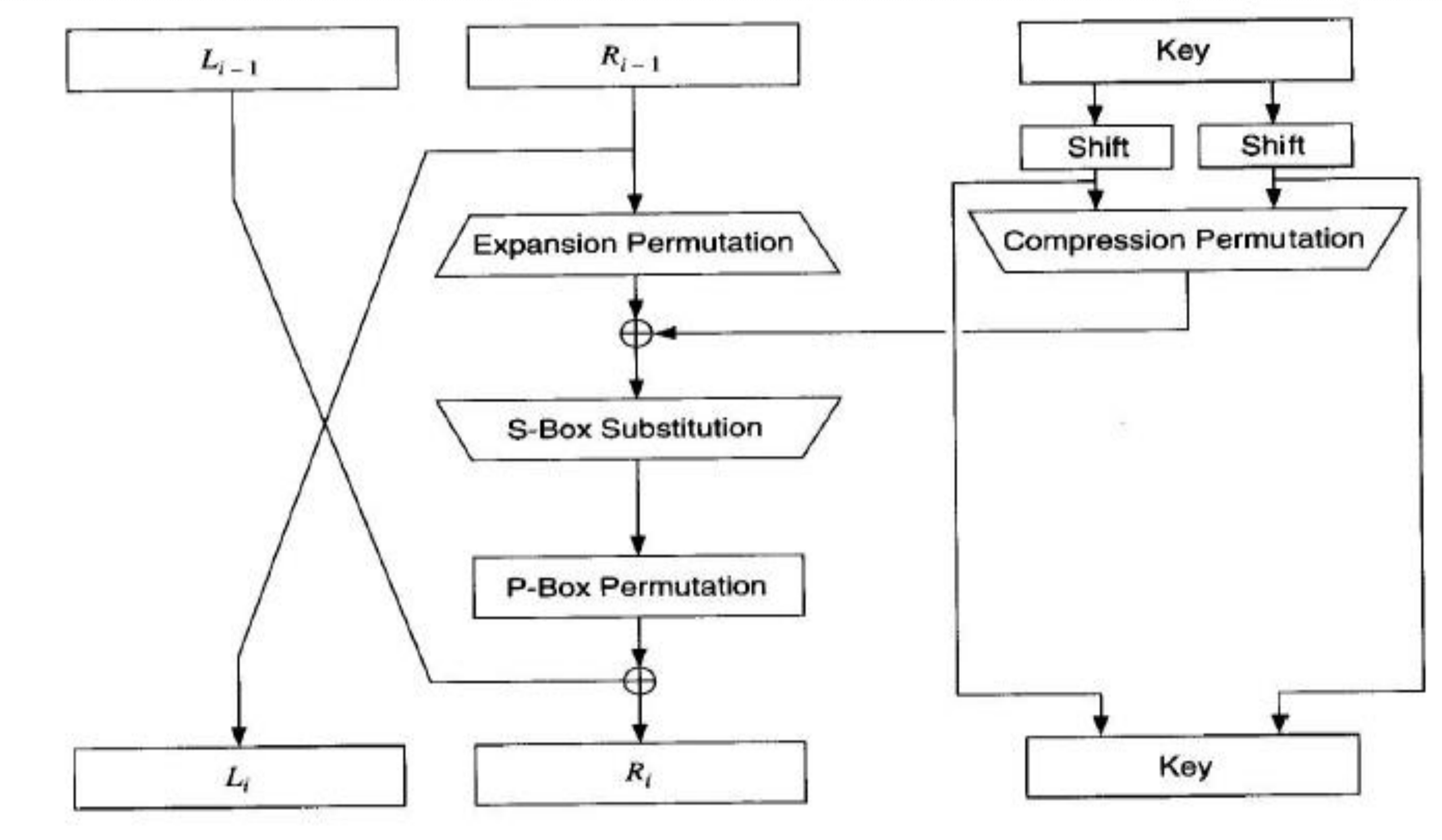
Outline of the Encryption Algorithm

- The basic process in enciphering a 64-bit data block using the DES consists of:
  - an initial permutation (IP)
  - 16 rounds of a complex key dependent calculation f
  - final permutation, being the inverse of IP
- In each round : -
  - the key bits are shifted, and then 48 bits are selected from the 56 bits of the key.
  - The right half of the data is expanded to 48 bits via an **expansion permutation**, combined with 48 bits of a shifted and permuted key via **an XOR**, sent through **8 S-boxes** producing 32 new bits, and **permuted** again.
  - These four operations make up **Function F**.
  - The output of Function F is then combined with the left half via another XOR.
  - The result of these operations becomes the new right half; the old right half becomes the new left half.
  - If Bi is the result of the ith iteration, Li and Ri are the left and right halves of Bi, Ki is the 48-bit key for round i, and F is the function that does all the substituting and permuting and XORing with the key, then a round looks like:

$$Li = R_{j-1}$$

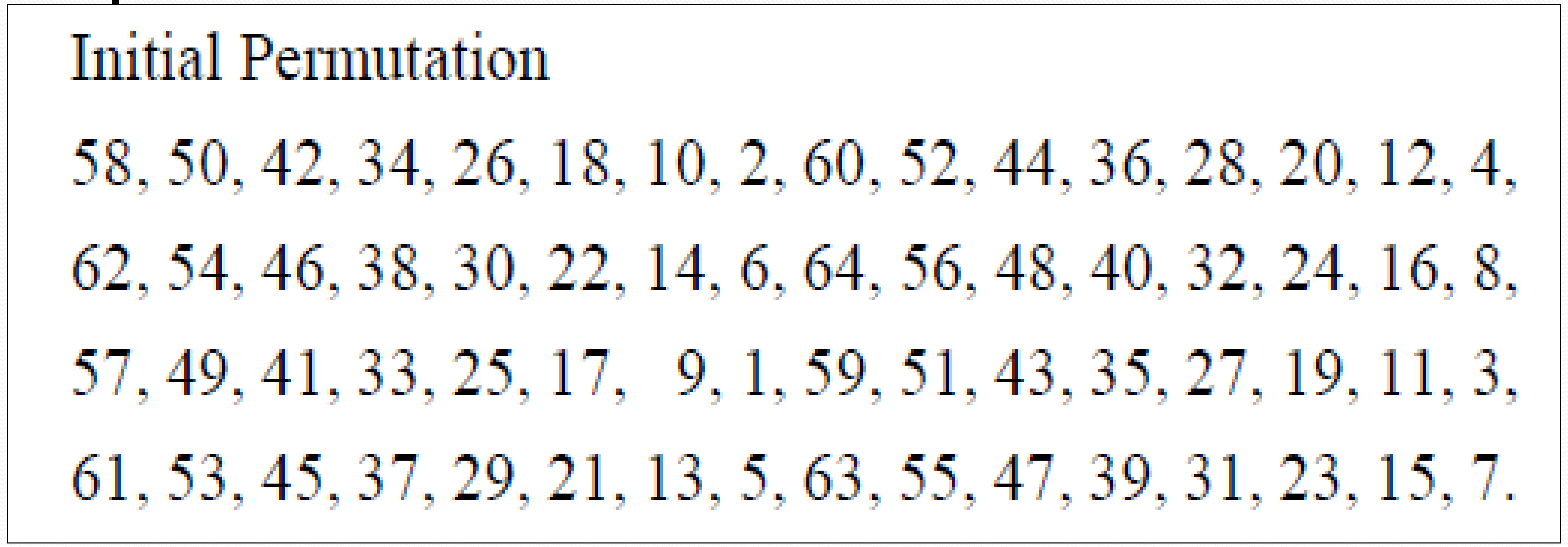
$$Ri = L_{i-1} \text{ Xor } f\,(R_{i-1}, K_i)$$

One round of DES



The Initial Permutation

- The initial permutation occurs before round 1.
- it transposes the input block as described in this Table



Initial Permutation

58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44, 36, 28, 20, 12, 4,

62, 54, 46, 38, 30, 22, 14, 6, 64, 56, 48, 40, 32, 24, 16, 8,

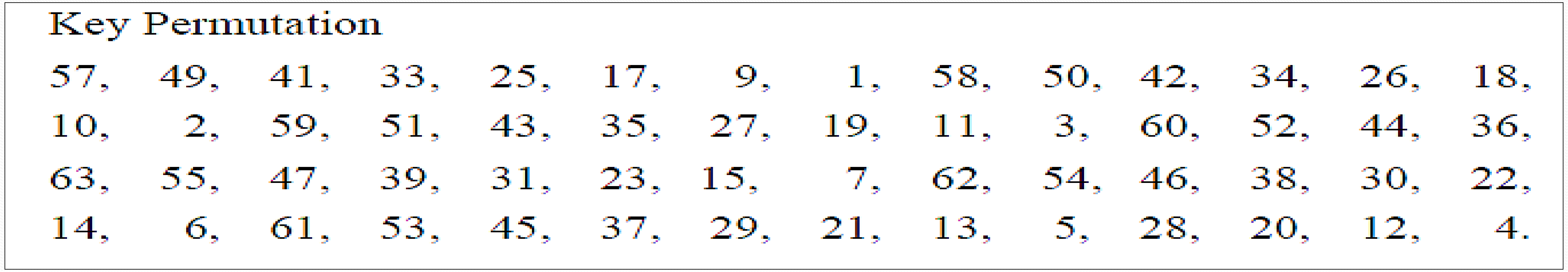
57, 49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35, 27, 19, 11, 3,

61, 53, 45, 37, 29, 21, 13, 5, 63, 55, 47, 39, 31, 23, 15, 7.

- This table The initial permutation and the corresponding final permutation do not improve DES's security, just make DES more complex should be read left to right, top to bottom.
- For example, the initial permutation moves bit 58 of the plaintext to bit position 1, bit 50 to bit position 2, bit 42 to bit position 3, and so forth.
- Example: IP(675a6967 5e5a6b5a) = (ffb2194d 004df6fb)

The Key Transformation: -

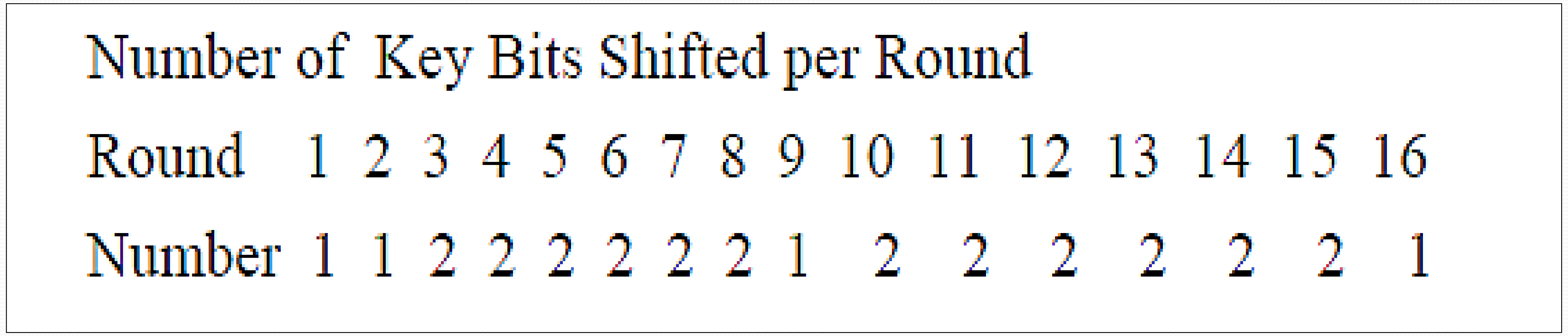
- Initially, the 64-bit DES key is reduced to a 56-bit key by ignoring every eighth bit using this table: -



Key Permutation

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

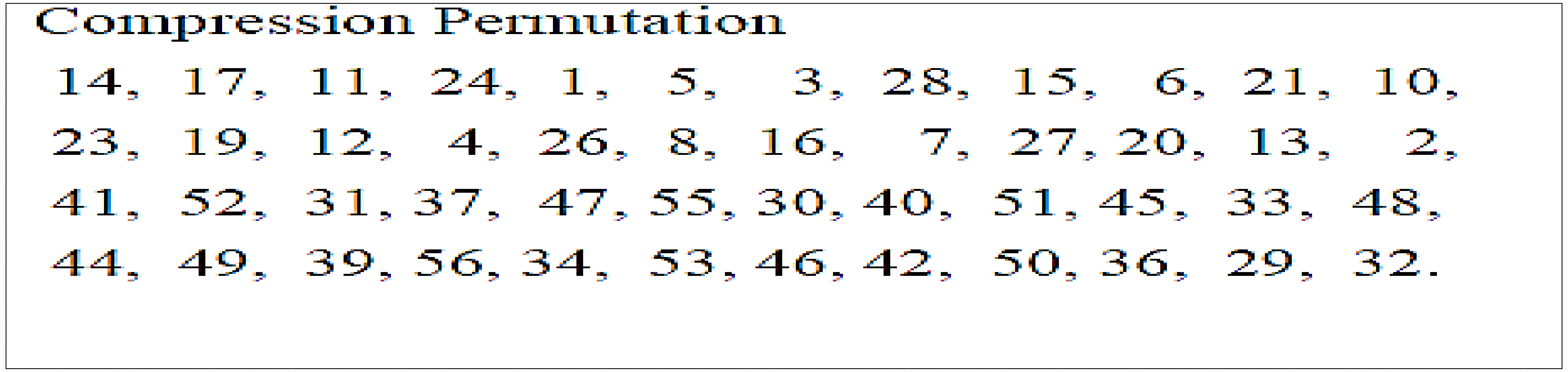
- Next the 56-bits key is reduced to a 48-bits subkey for each of the 16 rounds of DES. These subkeys, Ki, are determined in the following manner: -
  - splits the 56-bits key bits into 2 halves (C and D), each 28-bits
  - The halves C and D are circularly shifted left by either one or two bits, depending on the round. This shift is given in this table



Number of Key Bits Shifted per Round

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

- After being shifted, 48 out of the 56 bits are selected. This is done by an operation called **compression permutation**, it permutes the order of the bits as well as selects a subsets of bits.



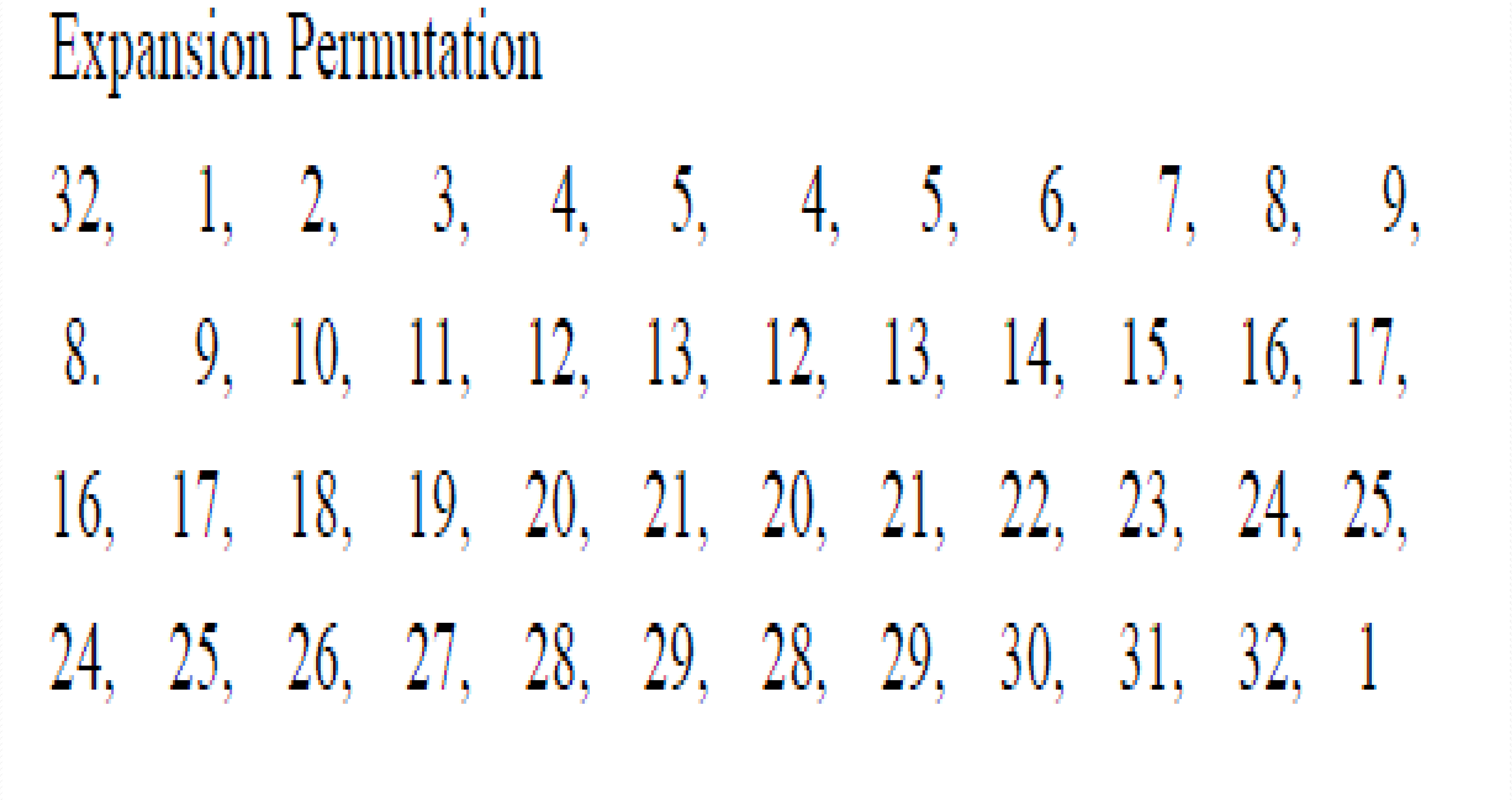
Compression Permutation

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.

- Example: keyinit(5b5a5767, 6a56676e)

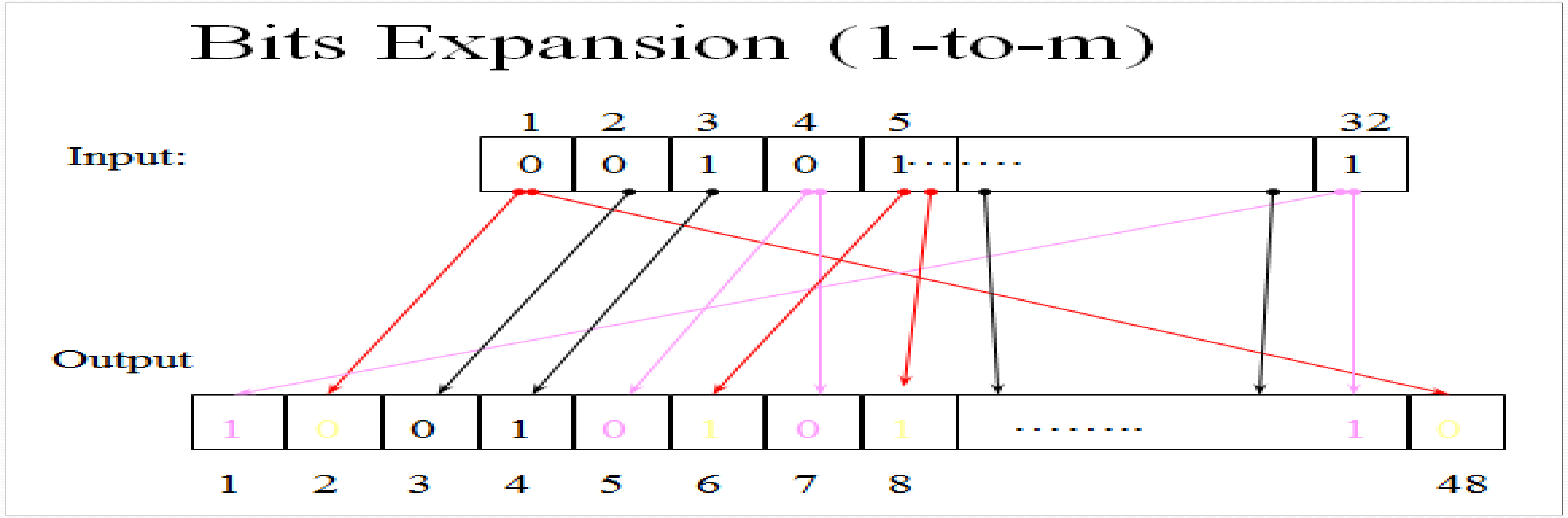
The Expansion Permutation: -

- This operation expands the right half of the data, Ri, from 32 bits to 48 bits.
- Because this operation changes the order of the bits as well as repeating certain bits, it is known as an expansion permutation
- This operation has two purposes:
  - It makes the right half the same size as the key for the XOR operation
  - and it provides a longer result that can be compressed during the substitution operation.
- For each 4-bit input block, the first and fourth bits each represent two bits of the output block, while the second and third bits each represent one bit of the output block as shown : -



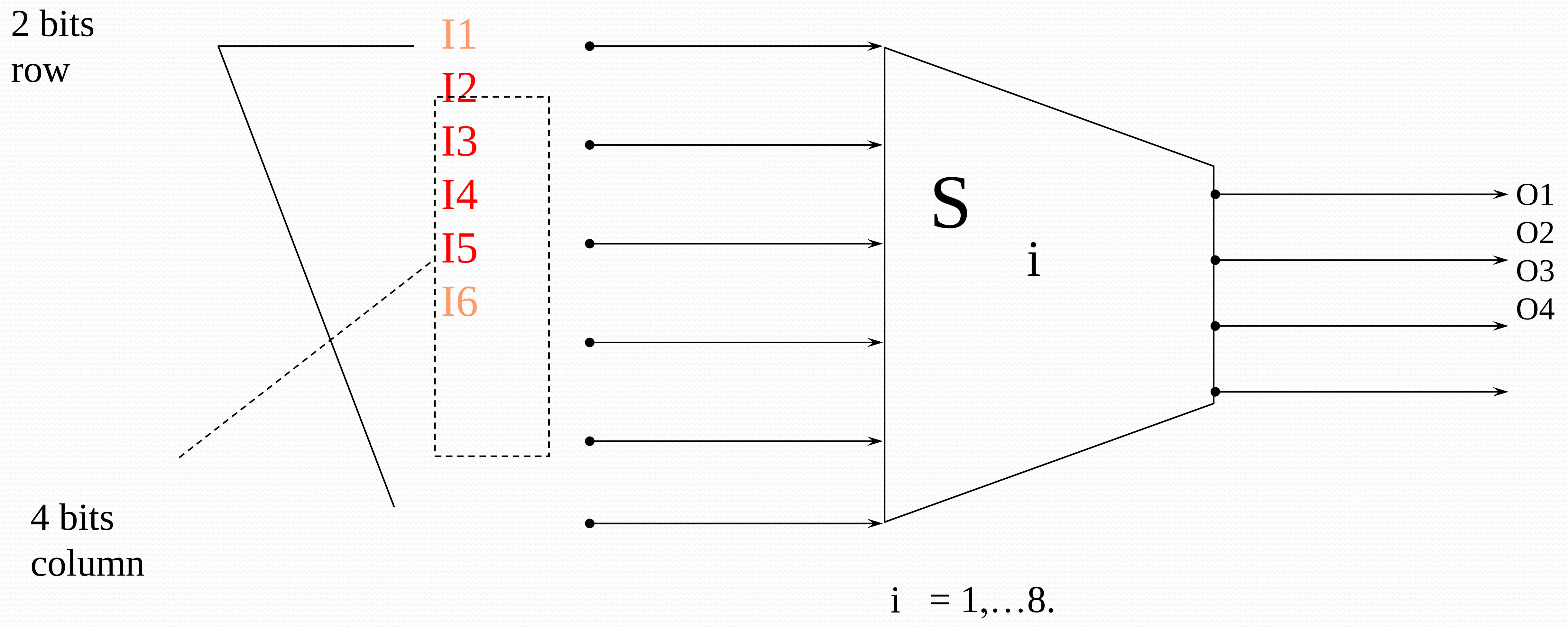
Expansion Permutation

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



The S-Box Substitution:-

- After the compressed key is XORed with the expanded block, the 48-bit result moves to a substitution operation.
- The substitutions are performed by eight substitution boxes, or S-boxes.
- There are eight different S-boxes.
- Each S-box has a 6-bit input and a 4-bit output.



- The 48 bits are divided into eight 6-bit sub-blocks.
- Each separate block is operated on by a separate S-box: The first block is operated on by S-box 1, the second block is operated on by S-box 2, and so on.
- Each S-box is a table of 4 rows and 16 columns. Each entry in the box is a 4-bit number.
- The 6 input bits of the S-box specify under which row and column number to look for the output. All eight S-boxes tables are :

S-box 1:

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

S-box 2:

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

S-box 3:

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

S-box 4:

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

S-box 5:

2,	12,	4,	1,	7,	10,	11,	6,	8,	5,	3,	15,	13,	0,	14,	9,
14,	11,	2,	12,	4,	7,	13,	1,	5,	0,	15,	10,	3,	9,	8,	6,
41,	2,	1,	11,	10,	13,	7,	8,	15,	9,	12,	5,	6,	3,	0,	14,
11,	8,	12,	7,	1,	14,	2,	13,	6,	15,	0,	9,	10,	4,	5,	3,

S-box 6:

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

S-box 7:

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

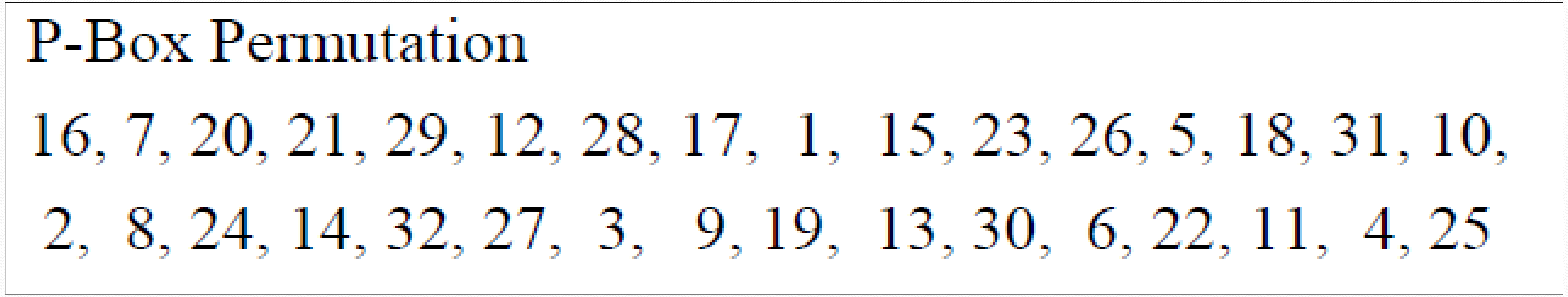
S-box 8:

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

- For example, assume that the input to the sixth S-box (i.e., bits 31 through 36 of the XOR function) is 110011.
- The first and last bits combine to form 11, which corresponds to row 3 of the sixth S-box.
- The middle 4 bits combine to form 1001, which corresponds to the column 9 of the same S-box.
- The entry under row 3, column 9 of S-box 6 is 14. (Remember to count rows and columns from 0 and not from 1.)
- The value 1110 is substituted for 110011.
- Example: S(18 09 12 3d 11 17 38 39) = 5fd25e03 (?)

The P-Box Permutation: -

- The 32-bit output of the S-box substitution is permuted according to a P-box.
- This permutation maps each input bit to an output position; no bits are used twice and no bits are ignored.



P-Box Permutation

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

- For example, bit 21 moves to bit 4. while bit 4 moves to bit 3 1.
- Finally, the result of the P-box permutation is XORed with the left half of the initial 64-bit block.
- Then the left and right halves are switched and another round begins.

The Final Permutation: -

- The final permutation is the inverse of the initial permutation.

Final Permutation

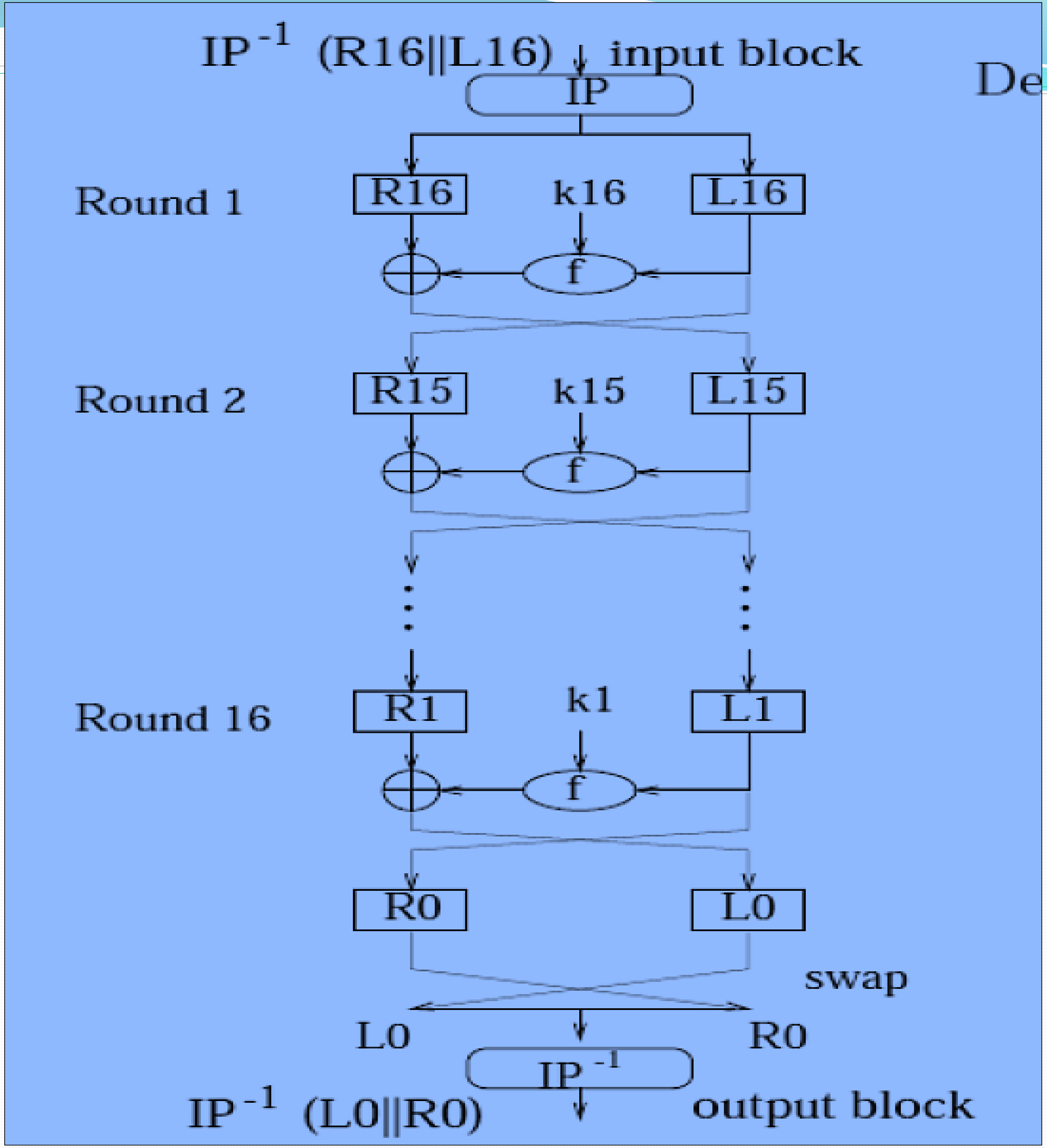
40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47, 15, 55, 23, 63,
31,
38, 6, 46, 14, 54, 22, 62 30, 37, 5, 45, 13, 53, 21, 61,
29,
36, 4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11, 51, 19, 59,
27,
34, 2, 42, 10, 50, 18, 58, 26, 33, 1, 41, 9, 49, 17, 57,
25.

- Note that the left and right halves are not exchanged after the last round of DES; instead the concatenated block R16L16 is used as the input to the final permutation.

Decrypting DES

- same function to encrypt or decrypt a block.
- The only difference is that the keys must be used in the reverse order. That is, if the encryption keys for each round are K1, K2, K3, . . . , K16, then the decryption keys are K16, K15, K14, . . . , K1,.
- The algorithm that generates the key used for each round is circular as well.
- The key shift is a right shift and the number of positions shifted is 0, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1.

De


IP -1 (R16||L16)
input block
IP
Round 1
R16
k16
L16
f
Round 2
R15
k15
L15
f
Round 16
R1
k1
L1
f
R0
L0
swap
L0
R0
IP -1
IP -1 (L0||R0)
output block

Block vs. Stream Ciphers

Block cipher	Stream cipher
process messages in into blocks, each of which is then en/decrypted	process messages a bit or byte at a time when en/decrypting
Error propagation	Low error propagation
Slowness	Speed of transformation
High Diffusion	Low diffusion
Immunity to insertions	Susceptibility to attacks on integrity

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 or verify signatures cannot decrypt messages or create signatures

Bobs's public key ring
Joy
Mike
Alice
Ted
Alice's public key
Alice 's private key
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., RSA)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output
(a) Encryption

Public-Key Characteristics: -

- it is computationally infeasible to find decryption key knowing only algorithm & encryption key
- it is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
- either of the two related keys can be used for encryption, with the other used for decryption (for some algorithms)

Public-Key Applications: -

- can classify uses into 3 categories:
  - encryption/decryption (provide secrecy)
  - digital signatures (provide authentication)
  - key exchange (of session keys)
- some algorithms are suitable for all uses, others are specific to one

Security of Public Key Schemes: -

- like private key schemes brute force exhaustive search attack is always theoretically possible
- but keys used are too large (>512bits)

Diffie-Hellman

- first public-key type scheme proposed by Diffie & Hellman in 1976 along with the exposition of public key concepts.
- Based on the difficulty of computing discrete logarithms of large numbers.

Public Parameter Creation

A trusted party chooses and publishes a (large) prime p
and an integer g having large prime order in $\mathbb{F}_p^*$.

Private Computations

Alice	Bob
Choose a secret integer a. Compute $A \equiv g^a \pmod{p}$.	Choose a secret integer b. Compute $B \equiv g^b \pmod{p}$.

Public Exchange of Values

Alice sends A to Bob $\longrightarrow$ A

B $\longleftarrow$ Bob sends B to Alice

Further Private Computations

Alice	Bob
Compute the number $B^a \pmod{p}$.	Compute the number $A^b \pmod{p}$.

The shared secret value is $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod{p}$.

- Where g is a primitive root of p.
- Let p be a prime. Then g is a *primitive root* for p if the powers of g, 1, g, g^2, g^3, ... include all of the residue classes mod p (except 0)
- **Examples:**

If p=7, then 3 is a primitive root for p because the powers of 3 are

1, 3, 2, 6, 4, 5---that is, every number mod 7 occurs except 0.

But 2 isn't a primitive root because the powers of 2 are

1, 2, 4, 1, 2, 4, 1, 2, 4...missing several values.

- **Example:**

If p=13, then 2 is a primitive root because the powers of 2 are
1, 2, 4, 8, 3, 6, 12, 11, 9, 5, 10, 7---which is all of the classes mod 13 except 0.

There are other primitive roots for 13 (?).

Diffie-Hellman

Eve
ALICE
BOB

g

p

$$A = g^{x} \bmod p$$

$$B = g^{y} \bmod p$$

A

$$k = B^{x} \bmod p$$

B

$$k' = A^{y} \bmod p$$

$$k' = k = g^{xy} \bmod p$$

- Example : -
- Alice and Bob agree on **p = 23** and **g = 5**.(show that 5 is primitive root of 23)
- Alice chooses **a= 6** and sends **5 6 mod 23 = 8**.
- Bob chooses **b = 15** and sends **515 mod 23 = 19**.
- Alice computes **19 6 mod 23 = 2**.
- Bob computes **815 mod 23 = 2**. Then **2** is the shared secret.
- Clearly, much larger values of **a**, **b**, and **p** are required.

Rivest, Shamir and Adleman (RSA)

- RSA stands for Rivest, Shamir, and Adleman, they are the inventors of the RSA cryptosystem. RSA is one of the algorithms used in PKI (Public Key Infrastructure), asymmetric key encryption scheme. RSA is a block chiper, it encrypt message in blocks (block by block). The common size for the key length now is 1024 bits for P and Q, therefore N is 2048 bits, if the implementation (the library) of RSA is fast enough, we can double the key size.
- **Key Generation Algorithm**
- Generate two large random primes, *p* and *q*, of approximately equal size such that their product n = pq is of the required bit length, e.g. 1024 bits.
- Compute n = pq and (ϕ) phi = (p-1)(q-1).
- Choose an integer *e*, 1 < e < phi, such that gcd(e, phi) = 1.
- Compute the secret exponent *d*, 1 < d < phi, such that ed ≡ 1 (mod phi).
- The public key is (n, e) and the private key is (n, d). Keep all the values d, p, q and phi secret.
- n is known as the *modulus*.
- e is known as the *public exponent* or *encryption exponent* or just the *exponent*.
- d is known as the *secret exponent* or *decryption exponent*.`

- In encryption, represents the plaintext message as a positive integer *m and* computes the ciphertext $C = m^e \bmod n$.
- In decryption compute $m = c^d \bmod n$

Example : let p=17 & q=11 then

- Compute n = pq =17×11=187.
- Compute ø(n) or (ϕ) phi =(p–1)(q-1)=16×10=160.
- choose e=7 (1 < e < 160) where gcd(7,160)=1.
- d=23 where 1 < d < 160 and ed ≡ 1 (mod 160).(multiplication inverse).
- The public key is (187, 7) and the private key is (187, 23).
- given message M = 88 (88<187)
- encryption: $C = m^e \bmod n$: $C = 88^7 \bmod 187 = 11$.
- Decryption: $m = c^d \bmod n$: $m = 11^{23} \bmod 187 = 88$.
-

- **Ex**/ p=3,q=11,e=7,m=2 encrypt and decrypt using RSA Algorithm?
- Choose p = 3 and q = 11
- Compute n = p * q = 3 * 11 = 33
- Compute $\phi(n) = (p - 1) * (q - 1) = 2 * 10 = 20$
- Let e = 7
- Compute d = 3 [(3 * 7) mod 20 = 1]
- Public key is (e, n) => (7, 33)
- Private key is (d, n) => (3, 33)
- The encryption of *m* = 2 is *c* = 2^7 *mod 33 = 29*
- The decryption of *c = 29* is *m* = 29^3 *mod 33* = 2