Security definition:

Security is referred to protect data during the transmission. It is concerned with making sure that noisy people cannot read message ,or worse yet ,secretly modify messages intended for other recipients. It is concerned with people trying to accesses the remote services that they are not authorized to use. Security also deals with people trying to deny that they sent certain message.

The OSI Security Architecture

ITU-T Recommendation X.800, *Security Architecture for OSI* (open systems interconnection) , defines such a systematic approach.

The OSI security architecture is useful to managers as a way of organizing the task of providing security. The OSI security architecture focuses on security attacks, mechanisms, and services. These can be defined briefly as follows:

- **Security attack:** Any action that compromises the security of information owned by an organization.
- **Security mechanism:** A process that is designed to detect, prevent, or recover from a security attack.
- **Security service:** A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

Security Attacks

A useful means of classifying security attacks is in terms of ***passive attacks*** and ***active attacks*** . A passive attack attempts to learn or make use of information from the system but does not affect system resources. An active attack attempts to alter system resources or affect their operation.

Passive Attacks:

Passive attacks are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted. Two types of passive attacks are release of message contents and traffic analysis.

The **release of message contents**. A telephone conversation, an electronic mail message, and a transferred file may contain sensitive or confidential information. We would like to prevent an opponent from learning the contents of these transmissions.

A second type of passive attack, **traffic analysis**. Suppose that we had a way of masking the contents of messages or other information traffic so that opponents, even if they captured the message, could not extract the information from the message. The common technique for masking contents is encryption. If we had encryption protection in place, an opponent might still be able to observe the pattern of these messages. The opponent could determine the location and identity of communicating hosts and could observe the frequency and length of messages being exchanged. This information might be useful in guessing the nature of the communication that was taking place.

Active Attacks

Active attacks involve some modification of the data stream or the creation of a false stream and can be subdivided into four categories: masquerade, replay, modification of messages, and denial of service.

A **masquerade** takes place when one entity pretends to be a different entity .Masquerade attack usually includes one of the other forms of active attack.

Replay involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized.

Modification of messages simply means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect.

The **denial of service** prevents or inhibits the normal use or management of communications facilities.

Security Services:

X.800 defines a security service as a service provided by a protocol layer of communicating open systems, which ensures the systems or of data transfers. Perhaps a clearer definition is found in RFC 2828, which provides the following definition: a processing or communication service that is provided by a system to give a specific kind of protection to system resources; security services implement security policies and are implemented by security mechanisms.

X.800 divides these services into five categories and fourteen specific services .

Security Services (X.800)

AUTHENTICATION

The assurance that the communicating entity is the one that it claims to be.

Peer Entity Authentication Used in association with a logical connection to provide confidence in the identity of the entities connected.

Data Origin Authentication In a connectionless transfer, provides assurance that the source of received data is as claimed.

ACCESS CONTROL

The prevention of unauthorized use of a resource (i.e., this service controls who can have access to a resource, under what conditions access can occur, and what those accessing the resource are allowed to do).

DATA CONFIDENTIALITY

The protection of data from unauthorized disclosure.

Connection Confidentiality The protection of all user data on a connection.

Connectionless Confidentiality The protection of all user data in a single data block

Selective-Field Confidentiality The confidentiality of selected fields within the user data on a connection or in a single data block.

Traffic Flow Confidentiality The protection of the information that might be derived from observation of traffic flows.

DATA INTEGRITY

The assurance that data received are exactly as sent by an authorized entity (i.e., contain no modification, insertion, deletion, or replay).

Connection Integrity with Recovery Provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data within an entire data sequence, with recovery attempted.

Connection Integrity without Recovery As above, but provides only detection without recovery.

Selective-Field Connection Integrity Provides for the integrity of selected fields within the user data of a data block transferred over a connection and takes the form of determination of whether the selected fields have been modified, inserted, deleted, or replayed.

Connectionless Integrity Provides for the integrity of a single connectionless data block and may take the form of detection of data modification. Additionally, a limited form of replay detection may be provided.

Selective-Field Connectionless Integrity Provides for the integrity of selected fields within a single connectionless data block; takes the form of determination of whether the selected fields have been modified.

NONREPUDIATION

Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

Nonrepudiation, Origin Proof that the message was sent by the specified party.

Nonrepudiation, Destination Proof that the message was received by the specified party.

Security mechanisms

The lists of security mechanisms defined in X.800 are:

Security Mechanisms (X.800)

SPECIFIC SECURITY MECHANISMS

May be incorporated into the appropriate protocol layer in order to provide some of the OSI security services.

Encipherment :The use of mathematical algorithms to transform data into a form that is not readily intelligible. The transformation and subsequent recovery of the data depend on an algorithm and zero or more encryption keys.

Digital Signature :Data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient).

Access Control A variety of mechanisms that enforce access rights to resources.

Data Integrity A variety of mechanisms used to assure the integrity of a data unit or stream of data units.

Authentication Exchange A mechanism intended to ensure the identity of an entity by means of information exchange.

Traffic Padding The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts.

Routing Control Enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected.

Notarization The use of a trusted third party to assure certain properties of a data exchange.

PERVASIVE SECURITY MECHANISMS

Mechanisms that are not specific to any particular OSI security service or protocol layer.

Trusted Functionality That which is perceived to be correct with respect to some criteria (e.g., as established by a security policy).

Security Label The marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource.

Event Detection Detection of security-relevant events.

Security Audit Trail Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities.

Security Recovery Deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions.

Cryptography

When people initially tried to communicate over distances, they tried to ensure the secrecy of their communications. **Cryptography** is a technique for securing the secrecy of communication .

Cryptography is a well-known method for securing the secrecy of communication in which the secret message is transformed to another form such that it does not make any sense to the intruder. The word Cryptography was evolved from two Greek words − Crypto (hidden, secret), and Graphein (writing). **Cryptography** is a techniques used for deciphering a message without any knowledge of the enciphering details fall into the area of **cryptanalysis**. Cryptanalysis is what the layperson calls "breaking the code." The areas of cryptography and cryptanalysis together are called **cryptology**.

Symmetric Cipher Model

A symmetric encryption scheme has five ingredients :

- **Plaintext(P):** This is the original intelligible message or data that is fed into the algorithm as input.
- **Encryption algorithm (E):** The encryption algorithm performs various substitutions and transformations on the plaintext.
- **Secret key(K) :** The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time. The exact substitutions and transformations performed by the algorithm depend on the key.

- **Ciphertext (C) :** This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts.The ciphertext is an apparently random stream of data and, as it stands, is unintelligible.
- **Decryption algorithm (D) :** This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext.

The security of the cryptosystem often depends on keeping the key secret to some set of parties.

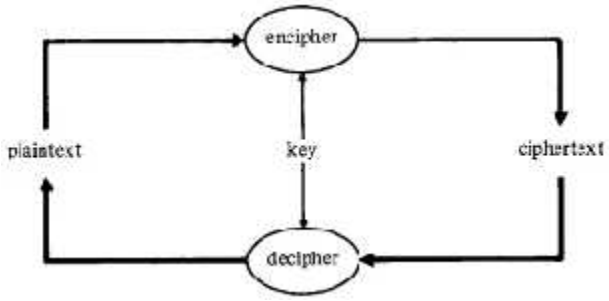


Figure (1) Secret writing (cryptography)

There are two requirements for secure use of conventional encryption:

a- A strong encryption algorithm.

b- Sender and receiver must have same secret key.

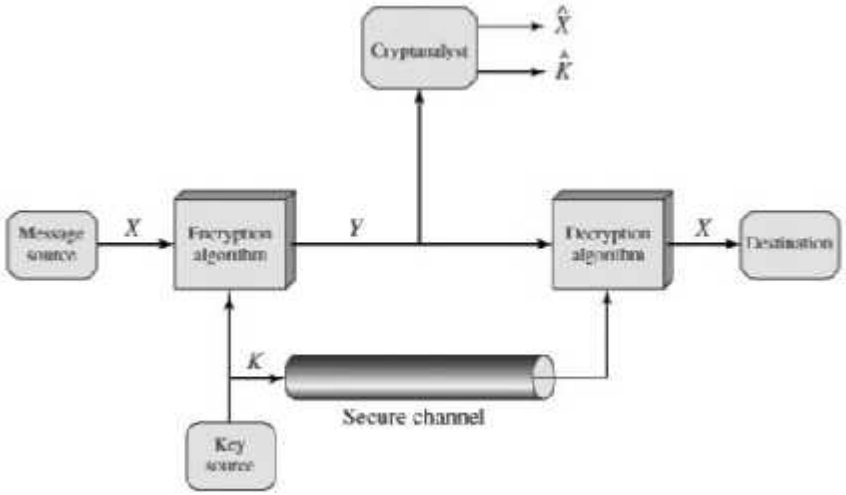


Figure (2) Model of Conventional Cryptosystem

If $X = [X1, X2, ..., XM]$ is a source produces a message in plaintext, and $K = [K1, K2, \ldots KJ]$ is the encryption key.

The encryption algorithm forms the ciphertext $Y = [Y1, Y2, ..., YN]$. We can write this as

$$Y = E(K, X)$$

The intended receiver, in possession of the key, is able to invert the transformation:

$$X = D(K, Y)$$

Cryptography systems are characterized along three independent dimensions :

1- The type of operations used for transforming plain text to cipher text:

a-Transposition algorithm: in which elements of the plain text are rearranged. The key is permutation of symbols , figure 3 explains the Transposition algorithm.

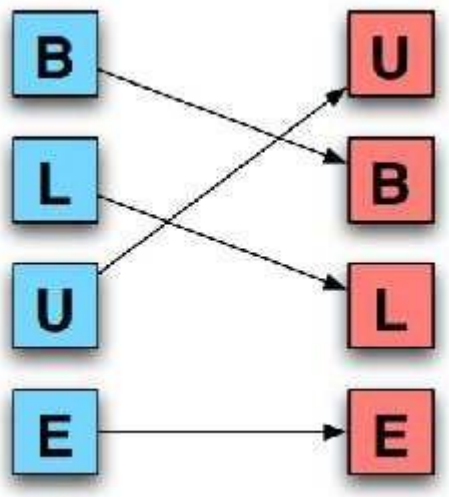


Figure (3)

b-Substitution algorithm: in which each element in the plain text (bit, letter, and group of bits or letters) is mapped into other element. The key is the permutation. Figure (4) explains the Substitution algorithm.

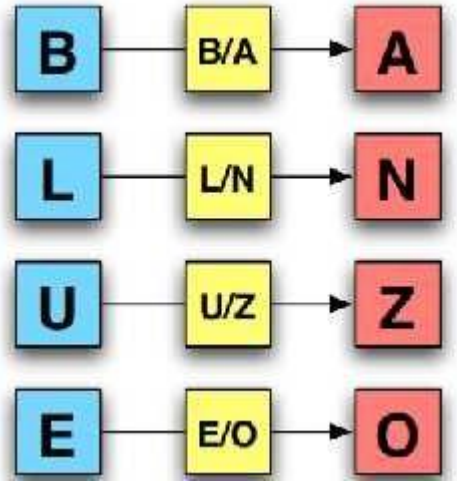


Figure (4)

c-Product system: in which multiple stages of substitution and transposition operations are done.

2- The number of key used:

a-Symmetric encryption: if both sender and receiver used the same key,the system is referred to as symmetric ,single-key, secret-key, or conventional encryption.

b-Asymmetric encryption : if the sender and receiver used different keys, the system is referred to as asymmetric, two-key, or public- key encryption.

3- the way in which the plain text is processed :

a-Block cipher : that process the input one block of elements at a time ,producing an output block for each input block.

b- Stream cipher: that process the input elements continuously, producing output elements at a time as it goes along.

Cryptanalysis

Is the science and study of methods of breaking ciphers. The whole point of cryptography is to keep the plaintext or the key or both secret from the eavesdroppers. There are two general approaches to attacking a conventional encryption scheme:

A- **Cryptanalysis:** Cryptanalytic attacks rely on the nature of the algorithm plus perhaps some knowledge of the general characteristics of the plaintext or even some sample plaintext - ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used.

There are several types of **cryptanalytic attacks**, based on the amount of information known to the cryptanalyst.

1- **Ciphertext-only attack**, the cryptanalyst has the ciphertext of several messages, all of which have been encrypted using the same encryption algorithm.
2- **a known-plaintext attack**, a cryptanalyst has not only the ciphertext of several message, but also have the plaintext of those message
3- **chosen-plaintext attack**, it is same as know-plain text attack, but here the cryptanalyst choose the plain text that gets encrypted.
4- **Chosen-ciphertext attack**: the cryptanalyst can choose different ciphertexts to be decrypted and has access to the decrypted plain text .
5- **Chosen-key attack** : this attack does not mean that the cryptanalyst can choose the key ,but it mean that the cruptanalyst

has some knowledge about the characteristics of the key ,or the relationship between different keys.

Types of Attacks on Encrypted Messages

Type of Attack	**Known to Cryptanalyst**
1- Ciphertext only	● Encryption algorithm ● Ciphertext
2- Known plaintext	● Encryption algorithm ● Ciphertext ● One or more plaintext-ciphertext pairs formed with the secret key.
3- Chosen plaintext	● Encryption algorithm ● Ciphertext ● Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key.
4- Chosen ciphertext	● Encryption algorithm ● Ciphertext ● Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key.
5- Chosen text	● Encryption algorithm ● Ciphertext ● Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key.

- Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key.

B- brute-force attack involves trying every possible key until an intelligible translation of the ciphertext into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.

Table 1. Average Time Required for Exhaustive Key Search

Key size (bits)	Number of alternative keys		Time required at 1 decryption/*ms*		Time required at 10^6 decryption/*ms*
32	2^{32}	= 4.3 x 10^9	2^{31} *ms*	= 35.8 minutes	2.15 milliseconds
56	2^{56}	= 7.2 x 10^{16}	2^{55} *ms*	= 1142 years	10.01 hours
128	2^{128}	= 3.4 x 10^{38}	2^{127} *ms*	= 5.4 x 10^{24} years	5.4 x 10^{18} years
168	2^{168}	= 3.7 x 10^{50}	2^{167} *ms*	= 5.9 x 10^{36} years	5.9 x 10^{30} years
26 characters (permutation)	26!	= 4 x 10^{26}	2 x 10^{26} *ms*	= 6.4 x 10^{12} years	6.4 x 10^6 years

Substitution Techniques

A substitution technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols. If the plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with ciphertext bit patterns.

Some type of substitution cipher:

1. Caesar cipher:

The earliest known use of a substitution cipher, and the simplest, was by Julius Caesar. The Caesar cipher involves replacing each letter of the alphabet with the letter standing three places further down the alphabet. For example,

```
plain:  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
cipher: 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
```

So, K=3 and E = small later ➡ Capital letters

Example :

Plain = meet me later ,key=3

Cipher=PHHW PH ODWHU

The assign a numerical equivalent to each letter:

a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12

n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

Then the algorithm can be expressed as follows. For each plaintext letter *p*, substitute the ciphertext letter *C*:

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

A shift may be of any amount, so that the general Caesar algorithm is

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

where *k* takes on a value in the range 1 to 25. The decryption algorithm is simply

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

A brute-force attack is easly performed with Caesar cipher because:

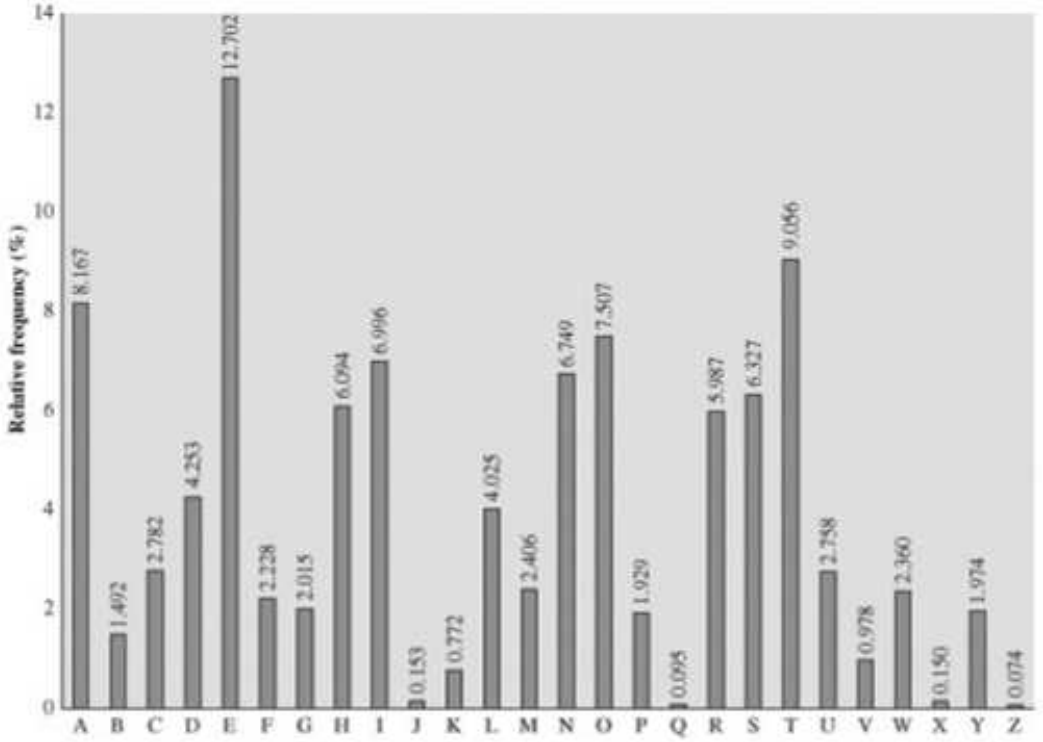
1. The encryption and decryption algorithms are known.
2. There are only 25 keys to try.
3. The language of the plaintext is known and easily recognizable.

2- **Monoalphabetic Ciphers**:

With only 25 possible keys, the Caesar cipher is far from secure.A dramatic increase in the key space can be achieved by allowing an arbitrary substitution. The "cipher" line can be any permutation of the 26 alphabetic characters, then there are 26! possible keys.

If the cryptanalyst knows the nature of the plaintext (e.g.,noncompressed English text), then the analyst can exploit the regularities of the language.

As a first step, the relative frequency of the letters can be determined and compared to a standard frequency distribution for English, such as is shown in below:



Relative Frequency of Letters in English Text

Example:

Cipher text= F W Z EZ W Q Z F Z I F

Total no. of elements=12

Z=4/12 , F=3/12 ,W=2/12

Z > F > W > E >Q >I Compare with the previous figure

Z=e , F=t , W=a ,

From the above analysis we can conclude that:

F W Z …. =t a e …. But it will be only estimation not certain and may by wrong

3- **Playfair Cipher**:

The best-known multiple-letter encryption cipher is the Playfair, which treats digrams in the plaintext as single units and translates these units into ciphertext digrams.

The Playfair algorithm is based on the use of a 5 x 5 matrix of letters constructed using a keyword.

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

In this case, the keyword is *monarchy*. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter.

Plaintext is encrypted two letters at a time, according to the following rules:

1. Repeating plaintext letters that are in the same pair are separated with a filler letter, such as x, so that balloon would be treated as ba lx lo on.
2. Two plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.
3. Two plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the column circularly following the last. For example, mu is encrypted as CM.
4. Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM (or JM, as the encipherer wishes).

4- Hill cipher:

Hill cipher is a multiletter cipher . The encryption algorithm takes *m* successive plaintext letters and substitutes for them *m* ciphertext letters. The substitution is determined by *m* linear equations in which each character is assigned a numerical value (a = 0, b = 1 ... z = 25). For *m* = 3, the system can be described as follows:

$$c_1 = (k_{11}p1 + k_{12}p2 + k_{13}p3) \bmod 26$$

$$c_2 = (k_{21}p1 + k_{22}p2 + k_{23}p3) \bmod 26$$

$$c_3 = (k_{31}p1 + k_{32}p2 + k3_{3}p3) \bmod 26$$

This can be expressed in term of column vectors and matrices:

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \bmod 26$$

Or

$$\mathbf{C} = \mathbf{KP} \bmod 26$$

C is cipher letter (3x1)

K is key (3x3).

P is plaintext letter (3x1).

In general terms, the Hill system can be expressed as follows:

$$\mathbf{C} = E(\mathbf{K}, \mathbf{P}) = \mathbf{KP} \bmod 26$$

$$\mathbf{P} = D(\mathbf{K}, \mathbf{P}) = \mathbf{K}^{-1}\mathbf{C} \bmod 26 = \mathbf{K}^{-1}\mathbf{KP} = \mathbf{P}$$

Ex: plaintext = paymoremoney

and use the encryption key is

$$\mathbf{K} = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$$

Find the ciphertext?

Sol:

pay mor emo ney

first three letters p=15 , a=0 , y=24

this can be represented in vector as :

$$P = \begin{pmatrix} 15 \\ 0 \\ 24 \end{pmatrix}$$

Then;

$$C = K\,P \bmod 26 = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \begin{pmatrix} 15 \\ 0 \\ 24 \end{pmatrix} \bmod 26$$

$$= \begin{pmatrix} 375 \\ 819 \\ 486 \end{pmatrix} \bmod 26 = \begin{pmatrix} 11 \\ 13 \\ 18 \end{pmatrix}$$

11 = L , 13 = N , 18 = S

So , pay ⟶ LNS .continuing in this fashion,

The ciphertext for the entire plaintext is LNSHDLEWMTRW.

Ex :

Decrypt the above ciphertext ?

Sol.

Decryption requires using the inverse of the matrix **K**. The inverse **K**

of a matrix **K** is defined by the equation **KK** = **K** **K** = **I**,

$$\mathbf{K}^{-1} = \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix}$$

This is demonstrated as follows:

$$\begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}\begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix} = \begin{pmatrix} 443 & 442 & 442 \\ 858 & 495 & 780 \\ 494 & 52 & 365 \end{pmatrix} \bmod 26 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

P= D(K,C)= K . C mod 26

$$\begin{vmatrix} P1 \\ P2 \\ P3 \end{vmatrix} = \begin{vmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{vmatrix} \begin{vmatrix} 11 \\ 13 \\ 5 \end{vmatrix} \quad \text{mod26=} \quad \begin{vmatrix} 15 \\ 0 \\ 24 \end{vmatrix} \begin{matrix} p \\ a \\ y \end{matrix}$$

H.W

The plaintext "**friday**" is encrypted using a 2 x 2 Hill cipher to yield the ciphertext **PQCFKU**, find the encrypted key?

Note: to find additive inverse modulo n of an integer we use the table below ,as example modulo 5

	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

The negative of integer x is the integer y ,such that

$$(x+ y) \bmod n =0$$

$$(x+y) \bmod 5=0$$

If x=2 ⇒ y=3 because (2+3)mod 5=0

If x=4 ⇒ y=1 because (4+1) mod 5=0

While the multiplicative inverse of an integer x is y such that

$$(x* y) \bmod n=1$$

Example of modulo 5 multiplicative invers of an integer x is

	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

(x*y) mod 5 =1

If x= 2 ⇒ y=3 because (2*3) mode 5=1

If x= 4 ⇒ y=4 because (4*4) mod 5=1

Example :

Find $\begin{pmatrix} 5 & 8 \\ 17 & 3 \end{pmatrix}$ for modulo 26 ?

solution :

5- Polyalphabetic Ciphers:

Another way to improve on the simple monoalphabetic technique is to use different monoalphabetic substitutions as one proceeds through the plaintext message. The general name for this approach is **polyalphabetic substitution cipher**.

The best known, and one of the simplest, such algorithm is referred to as the **Vigenère cipher.**

1- Vigenère cipher:

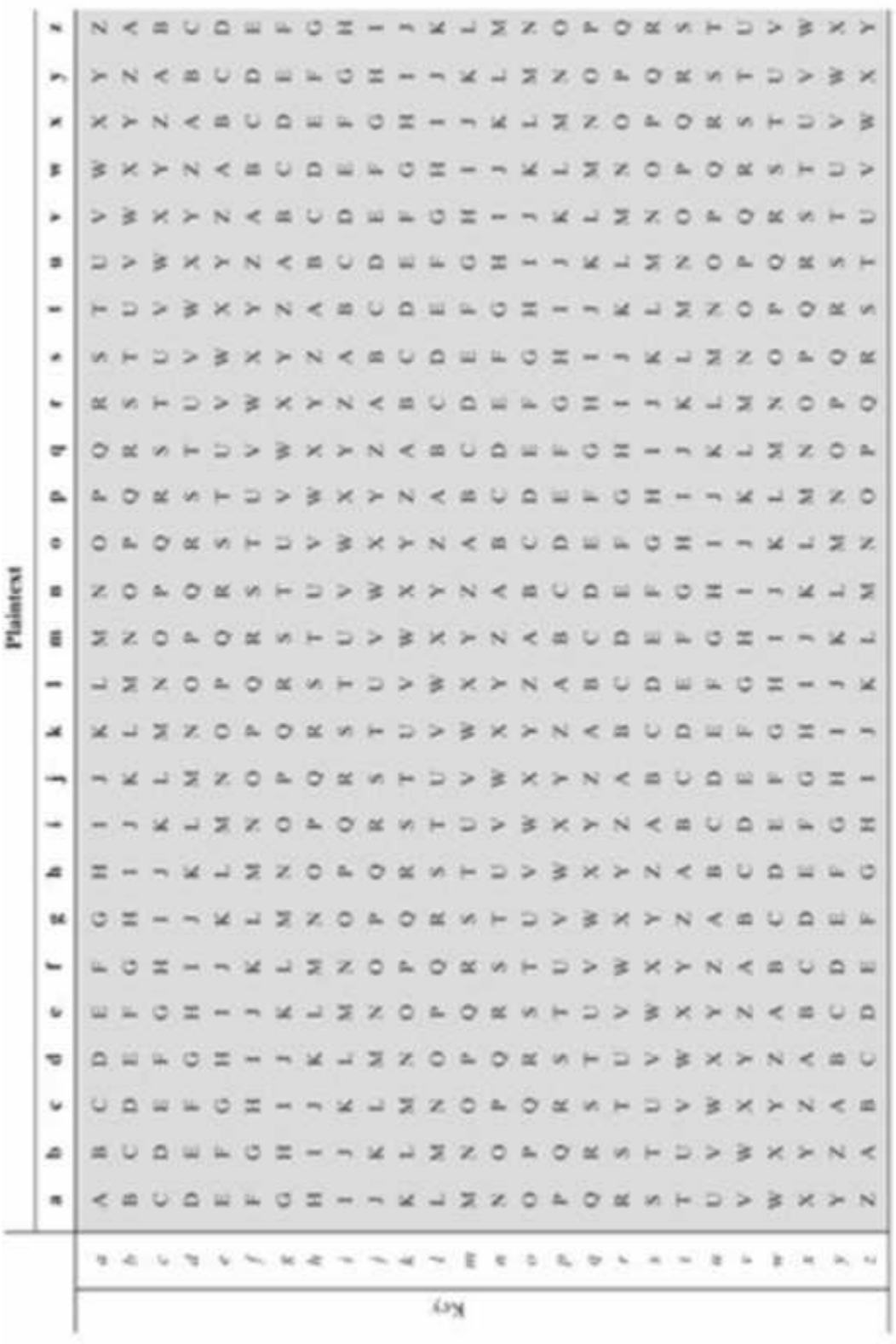
use Vigenère tableau is constructed (Table below). Each of the 26 ciphers is laid out horizontally, with the key letter for each cipher to its left. The process of encryption is simple: Given a key letter ***x*** and a plaintext letter **y**, the ciphertext letter is at the intersection of the row labeled *x* and the column labeled y; in this case the ciphertext is V.

A key is needed that is as long as the message. Usually, the key is a repeating keyword.

Example:

```
key:          deceptivedeceptivedeceptive
plaintext:    wearediscoveredsaveyourself
ciphertext:   ZICVTWQNGRZGVTWAVZHCQYGLMGJ
```

Decryption is equally simple. The key letter again identifies the row. The position of the ciphertext letter in that row determines the column, and the plaintext letter is at the top of that column.



The Modern Vigenère Tableau

2- Autokey system

The periodic nature of the keyword can be eliminated by using anonrepeating keyword that is as long as the message itself.

Example :

```
key:          deceptivewearediscoveredsav
plaintext:    wearediscoveredsaveyourself
ciphertext:   ZICVTWQNGKZEIIGASXSTSLVVWLA
```

3- Vernam cipher:

The system works on binary data rather than letters , the keyword is chosen as long as the plaintext and has no statistical relationship to it Vernam systems work with very long message but repeating keywrd. The system can be expressed succinctly as follows:

$$c_i = p_i \oplus k_i$$

where

p_i = ith binary digit of plaintext

k_i = ith binary digit of key

c_i = ith binary digit of ciphertext

$\oplus$ = exclusive or (XOR) operation

The decryption simply involves the same bitwise operation:

$$p_i = c_i \oplus k_i$$

Example :

Pi ⊕	Ki	= Ci	Ci ⊕	Ki =	Pi
0	0	0	0	0	0
0	1	1	1	1	0
1	0	1	1	0	1
1	1	0	0	1	1

6- One-Time Pad

It using a random key that is as long as the message, so that the key need not be repeated. In addition, the key is to be used to encrypt and decrypt a single message, and then is discarded. Each new message requires a new key of the same length as the new message. length as the new message.

One-time pad, is unbreakable. It produces random output that bears no statistical relationship to the plaintext.

In fact, given any plaintext of equal length to the ciphertext, there is a key that produces that plaintext.

The one-time pad offers complete security but, in practice, has two fundamental difficulties:

1- There is the practical problem of making large quantities of random keys. Any heavily used

2- Even more daunting is the problem of key distribution and protection. For every message to be sent, a key of equal length is needed by both sender and receiver.

Because of these difficulties, the one-time pad is of limited utility, and is useful primarily for low bandwidth channels requiring very high security.

Transposition Techniques:

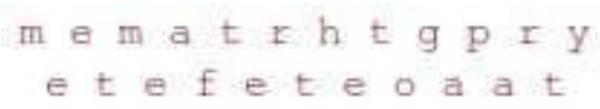
This technique is achieved by performing some sort of permutation on the plaintext letters.

1-Rail fence technique:

The plaintext is written down as a sequence of diagonals and then read off as a sequence of rows.

Example:

To encipher the message "meet me after the toga party" with a rail fence of depth 2, we write the following:



The encrypted message is:

MEMATRHTGPRYETEFETEOAAT

For depth 3 we write:

m t a e o p t
e m f r g a y
e e t t a r

The encryption message is :

MTAEOPTEMFRGAYEETTAR

2- Columnar transposition :

is to write the message in a rectangle, row by row, and read the message off, column by column, but permute the order of the columns. The order of the columns then becomes the key to the algorithm.

Example :

```
Key:            4 3 1 2 5 6 7
Plaintext:      a t t a c k p
                o s t p o n e
                d u n t i l t
                w o a m x y z
Ciphertext:     TTNAAPTMTSUOAODWCOIXKNLYPETZ
```

3- Double columnar transposition:

The transposition cipher can be made significantly more secure by performing more than one stage of transposition. The result is a more complex permutation that is not easily reconstructed. Thus, if the foregoing message is re-encrypted using the same algorithm.

Example :

```
Key:            4 3 1 2 5 6 7
Input:          t t n a a p t
                m t s u o a o
                d w c o i x k
                n l y p e t z
Output:         NSCYAUOPTTWLTMDNAOIEPAXTTOKZ
```

Block Cipher Principles

A **block cipher** is an encryption/decryption scheme in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length.

- Many block ciphers have a Feistel structure. Such a structure consists of a number of identical rounds of processing. In each round, a substitution is performed on one half of the data being processed, followed by a permutation that interchanges the two halves. The original key is expanded so that a different key is used for each round.

Most symmetric block encryption algorithms in current use are based on a structure referred to as a Feistel block cipher.

Stream Ciphers and Block Ciphers

A **stream cipher** is one that encrypts a digital data stream one bit or one byte at a time. Examples of classical stream ciphers are the autokeyed Vigenère cipher and the Vernam cipher.

A **block cipher** is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. Typically, a block size of 64 or 128 bits is used. A block cipher can be used to achieve the same effect as a stream cipher.

The idea of block cipher :

A block cipher operates on a plaintext block of ***n* bits** to produce a ciphertext block of ***n* bits**.

There are 2^n possible different plaintext blocks and, for the encryption to be reversible (i.e., for decryption to be possible), each must produce a unique ciphertext block. Such a transformation is called

reversible, or nonsingular. The following examples illustrate nonsingular and singular transformation for $n = 2$.

Reversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	00
11	01

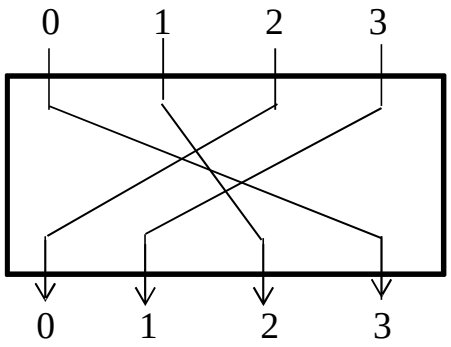
Irreversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	01
11	01

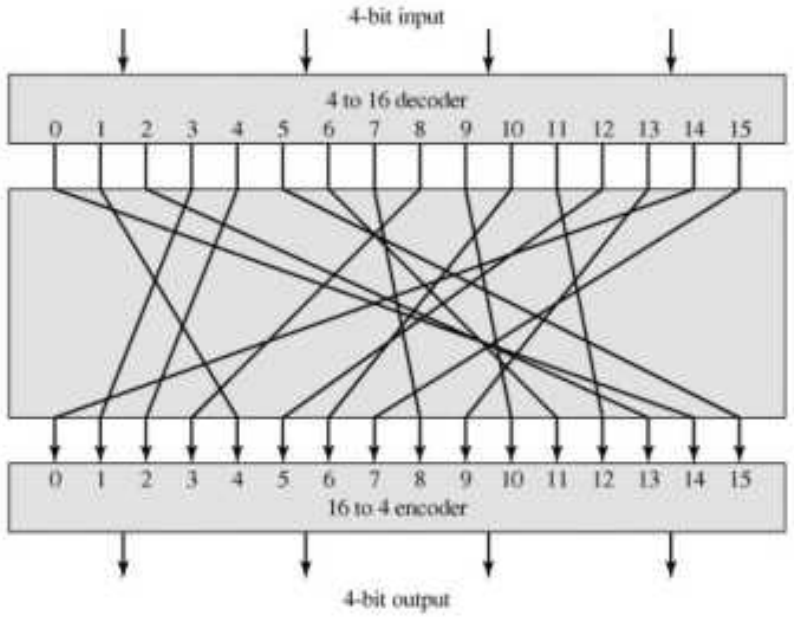
In the latter case, a ciphertext of 01 could have been produced by one of two plaintext blocks.

So if we limit ourselves to reversible mappings, the number of different transformations is $2^n!$

The circuit diagram of the table (Reversible) above is shown below (n=2)



While with n=4, the circuit diagram (Reversible) is shown below:



General *n*-bit-*n*-bit Block Substitution (shown with *n* = 4)

The key is used here to determine the specific mapping from among all possible mapping (show the value of the ciphertext for each plaintext block).

For n=2 the key length is :

Key length = (2bits) x (4-row)= 8 bits.

Generally , for an n-bit ideal block cipher the key length is (n x 2^{n}).

If n is large ,the key will be very large ,as example:

If n= 64 $\Longrightarrow$ K= 64 x $2^{64} = 2^{70} \approx 10^{21}$bits.

The Feistel Cipher

Feistel proposed , can approximate the ideal block cipher by utilizing the concept of a product cipher(use of a cipher that alternates substitutions and permutations) , which is the execution of two or more simple ciphers in sequence in such a way that the final result or product is cryptographically stronger than any of the component ciphers.

In fact, this is a practical application of a proposal by Claude Shannon to develop a product cipher that alternates confusion and diffusion functions.

The essence of the approach is to develop a block cipher with a key length of k bits and a block length of n bits, allowing a total of 2 possible transformations, rather than the 2 ! transformations available with the ideal block cipher.

Diffusion and Confusion

The terms diffusion and confusion were introduced by Claude Shannon to capture the two basic building blocks for any cryptographic system. Shannon's concern was to thwart cryptanalysis based on statistical analysis.

If these statistics are in any way reflected in the ciphertext, the cryptanalyst may be able to deduce the encryption key, or part of the key, or at least a set of keys likely to contain the exact key. In what Shannon refers to as a strongly ideal cipher, all statistics of the ciphertext are independent of the particular key used.

In **diffusion**, the statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext.

Diffusion can be achieved by repeatedly performing some permutation on the data followed by applying a function to that permutation; the effect is that bits from different positions in the original plaintext contribute to a single bit of ciphertext.

The mechanism of diffusion seeks to make the statistical relationship between the plaintext and ciphertext as complex as possible in order to thwart attempts to deduce the key.

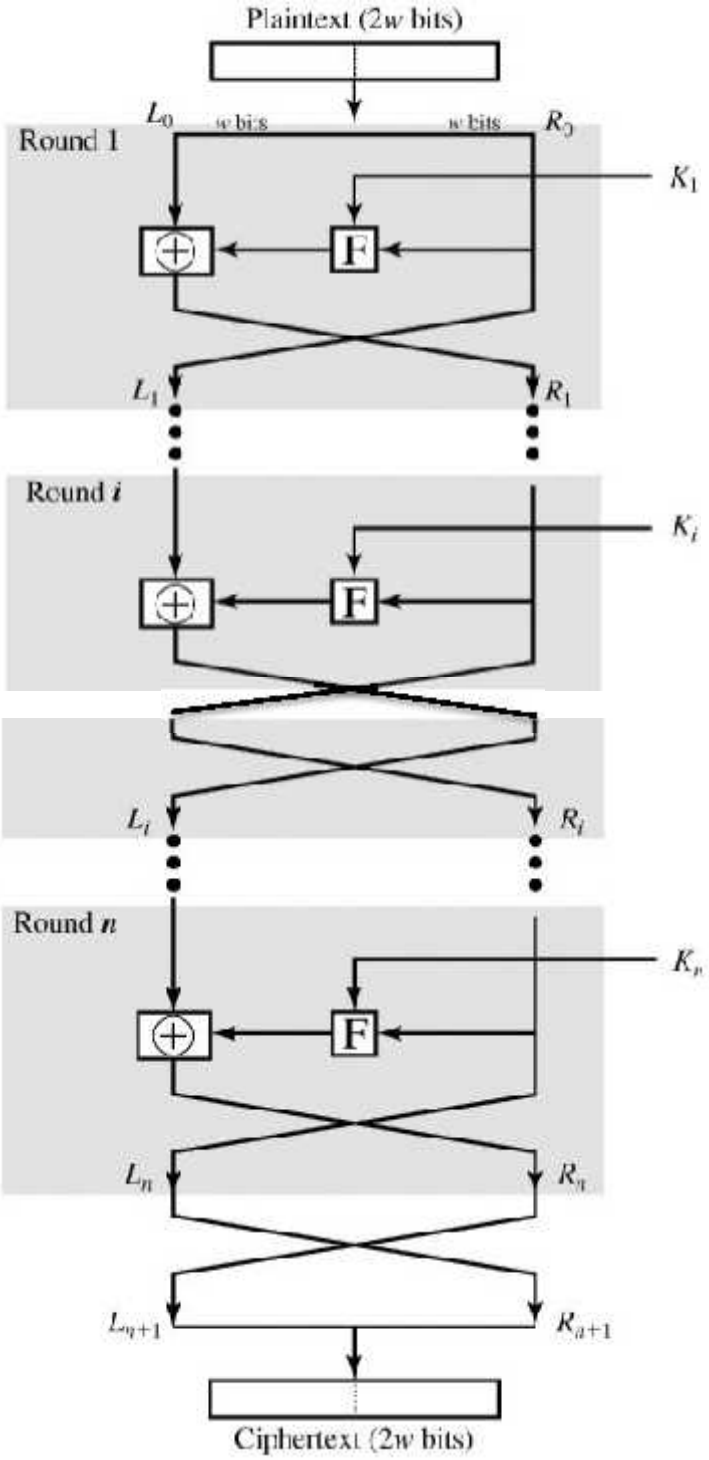
On the other hand, **confusion** seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible, again to thwart attempts to discover the key.

This is achieved by the use of a complex substitution algorithm.

Feistel Cipher Structure(encryption):

The inputs to the encryption algorithm are a plaintext block of length 2w bits and a key K. The plaintext block is divided into two halves, L(left) and R (right) . The two halves of the data pass through n rounds of processing and then combine to produce the ciphertext block. Each round i has as inputs L_{i-1} and R_{i-1}, derived from the previous round, as well as a subkey K_i, derived from the overall K. In general, the subkeys K_i are different from K and from each other.

Figure below depicts the structure proposed by Feistel.



Classical Feistel Network

All rounds have the same structure.

A- **Substitution** is performed on the left half of the data.

B- **Permutation** is performed that consists of the interchange of the two halves of the data.

This structure is a particular form of the substitution-permutation network (SPN) proposed by Shannon.

The exact realization of a Feistel network depends on the choice of the following parameters and design features:

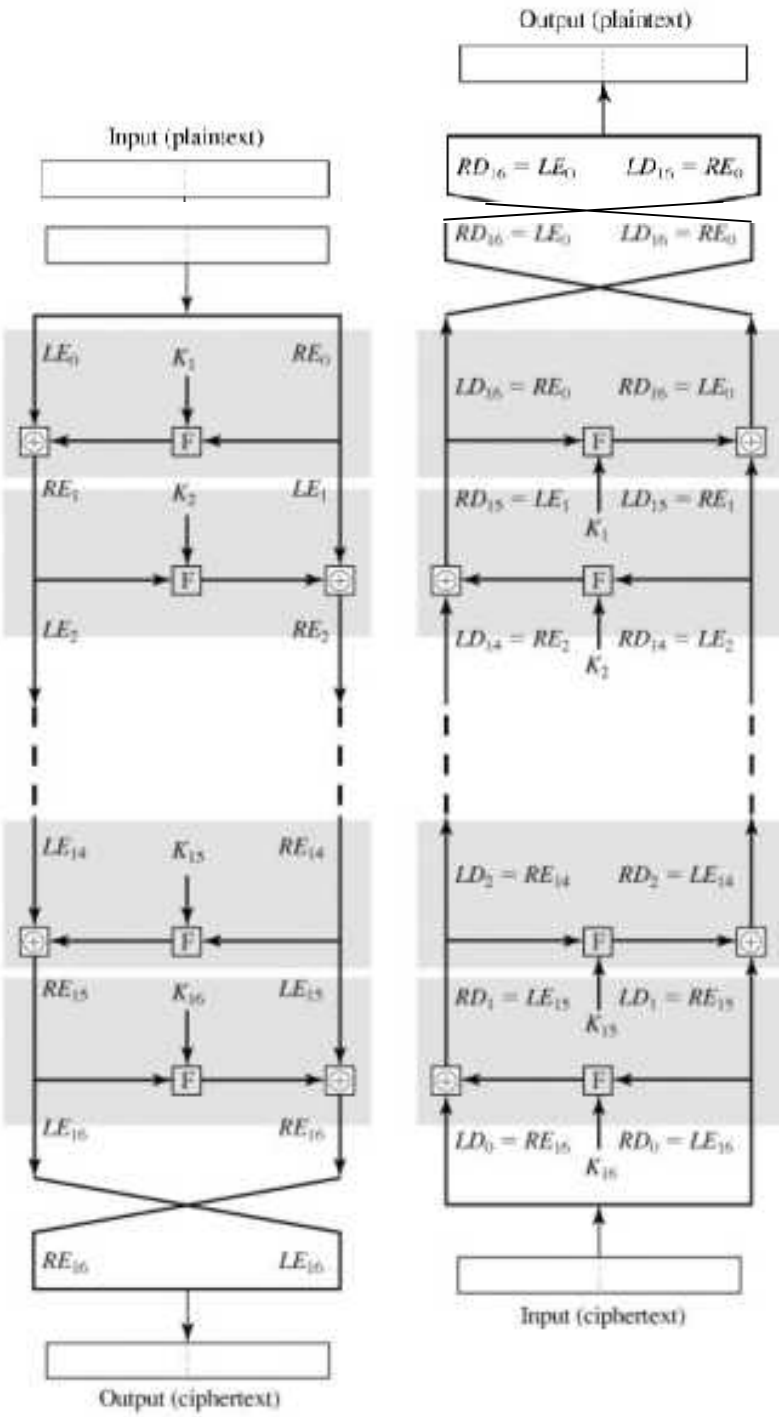
- **Block size:** Larger block sizes mean greater security (all other things being equal) but reduced encryption/decryption speed for a given algorithm.
- **Key size:** Larger key size means greater security but may decrease encryption/decryption speed. Key sizes of 64 bits or less are now widely considered to be inadequate, and 128 bits has become a common size.
- **Number of rounds:** The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security. A typical size is 16 rounds.
- **Subkey generation algorithm:** Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis.
- **Round function:** Again, greater complexity generally means greater resistance to cryptanalysis.
- **Fast software encryption/decryption:** The speed of execution of the algorithm becomes a concern.
- **Ease of analysis:** Although we would like to make our algorithm as difficult as possible to cryptanalyze.

Feistel Decryption Algorithm

The process of decryption with a Feistel cipher is essentially the same as the encryption process. The rule is as follows: Use the ciphertext as input to the algorithm, but use the subkeys K in reverse order.

That is, use K_n in the first round, K_{n-1} in the second round, and so on until K is used in the last round.

This is a nice feature because it means we need not implement two different algorithms, one for encryption and one for decryption.



Encryption Decryption

Feistel Encryption and Decryption

First, consider the encryption process. We see that:

$$LE_{16} = RE_{15}$$

$$RE_{16} = LE_{15} \times F(RE_{15}, K_{16})$$

On the decryption side,

$$LD_1 = RD_0 = LE_{16} = RE_{15}$$

$$RD_1 = LD_0 \times F(RD_0, K_{16})$$

$$= RE_{16} \times F(RE_{15}, K_{16})$$

$$= [LE_{15} \times F(RE_{15}, K_{15})] \times F(RE_{15}, K_{16})$$

The XOR has the following properties:

$[A \times B] \times C = A \times [B \times C]$

$D \times D = 0$

$E \times 0 = E$

Thus, we have $LD_1 = RE_{15}$ and $RD_1 = LE_{15}$.

Therefore, the output of the first round of the decryption process is LE15||RE15 , which is the 32-bit swap of the input to the sixteenth round of the encryption. in general terms. For the ith iteration of the encryption algorithm,

$$LE_i = RE_{i-1}$$

$$RE_i = LE_{i-1} \times F(RE_{i-1}, K_i)$$

Rearranging terms,

$$RE_{i-1} = LE_i$$

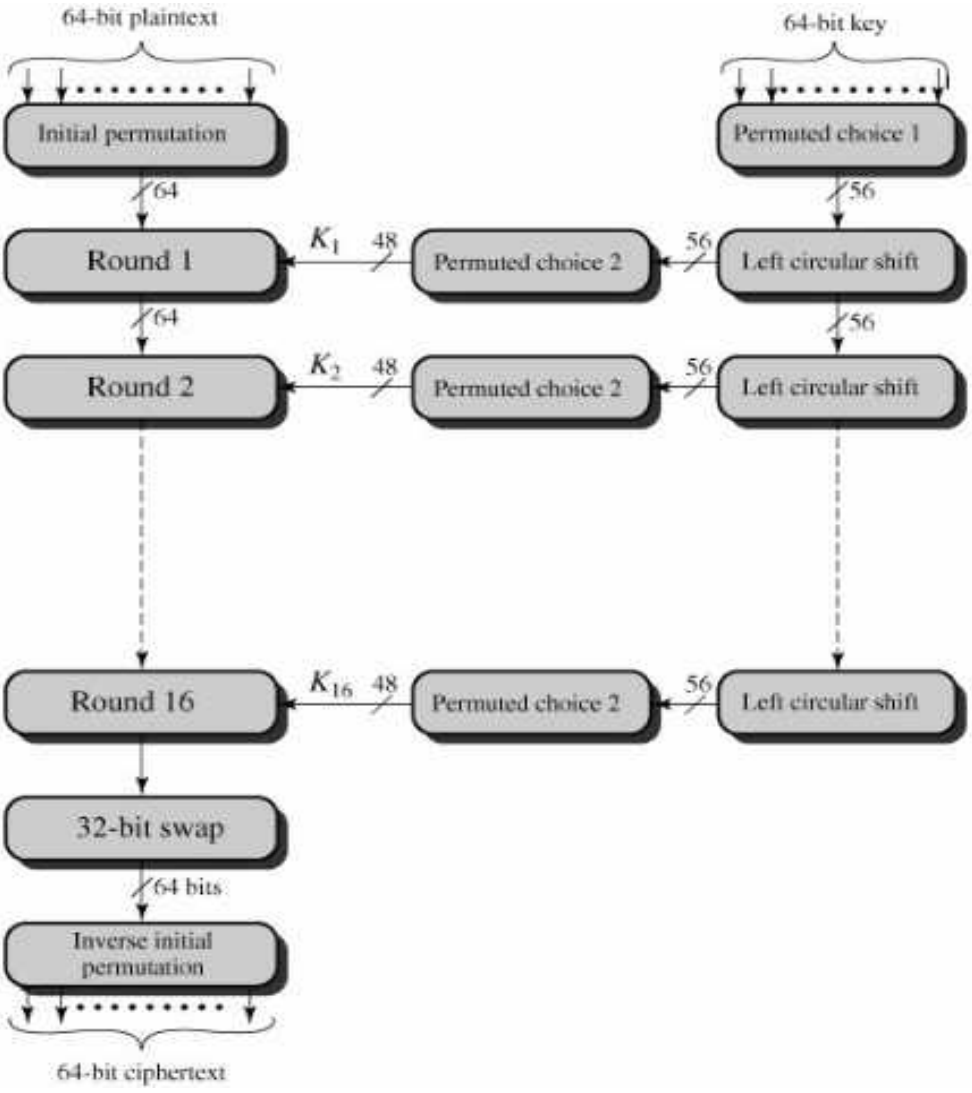
$$LE_{i-1} = RE_i \times F(RE_{i-1}, K_{i2} = RE_i \times F(LE_i, K_i)$$

The Data Encryption Standard:

For DES, data are encrypted in 64-bit blocks using a 56-bit key. The algorithm transforms 64-bit input in a series of steps into a 64-bit output. The same steps, with the same key, are used to reverse the encryption.

DES Encryption

The overall scheme for DES encryption is illustrated in Figure below. As with any encryption scheme, there are two inputs to the encryption function: the plaintext to be encrypted and the key. In this case, the plaintext must be 64 bits in length and the key is 56 bits in length.



General Depiction of DES Encryption Algorithm

Looking at the left-hand side of the figure up, we can see that the processing of the plaintext proceeds in three phases:

First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input. This is followed by **Second,** a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions. The output of the last (sixteenth) round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the **preoutput**.

Third: the preoutput is passed through a permutation (IP^{-1}) that is the inverse of the initial permutation function, to produce the 64-bit ciphertext.

The right-hand portion of Figure up shows the way in which :

First ,the 56-bit key is used. Initially, the key is passed through a permutation function.

Second, Then, for each of the 16 rounds, a *subkey* (*K*) is produced by the combination of a left circular shift and a permutation. The permutation function is the same for each round, but a different subkey is produced because of the repeated shifts of the key bits.

Initial Permutation

The initial permutation and its inverse are defined by tables, as shown in Tables 1b and 2b, respectively. The tables are to be interpreted as follows.

- The input to a table consists of 64 bits numbered from 1 to 64.
- The 64 entries in the permutation table contain a permutation of the numbers from 1 to 64.
- Each entry in the permutation table indicates the position of a numbered input bit in the output, which also consists of 64 bits.

Consider the following 64-bit input *M*:

M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}
M_{17}	M_{18}	M_{19}	M_{20}	M_{21}	M_{22}	M_{23}	M_{24}
M_{25}	M_{26}	M_{27}	M_{28}	M_{29}	M_{30}	M_{31}	M_{32}
M_{33}	M_{34}	M_{35}	M_{36}	M_{37}	M_{38}	M_{39}	M_{40}
M_{41}	M_{42}	M_{43}	M_{44}	M_{45}	M_{46}	M_{47}	M_{48}
M_{49}	M_{50}	M_{51}	M_{52}	M_{53}	M_{54}	M_{55}	M_{56}
M_{57}	M_{58}	M_{59}	M_{60}	M_{61}	M_{62}	M_{63}	M_{64}

Here *Mi* is a binary digit. Then the permutation $X = \mathrm{IP}(M)$ is as follows:

M_{58}	M_{50}	M_{42}	M_{34}	M_{26}	M_{18}	M_{10}	M_2
M_{60}	M_{52}	M_{44}	M_{36}	M_{28}	M_{20}	M_{12}	M_4
M_{62}	M_{54}	M_{46}	M_{38}	M_{30}	M_{22}	M_{14}	M_6
M_{64}	M_{56}	M_{48}	M_{40}	M_{32}	M_{24}	M_{16}	M_8
M_{57}	M_{49}	M_{41}	M_{33}	M_{25}	M_{17}	M_9	M_1
M_{59}	M_{51}	M_{43}	M_{35}	M_{27}	M_{19}	M_{11}	M_3
M_{61}	M_{53}	M_{45}	M_{37}	M_{29}	M_{21}	M_{13}	M_5
M_{63}	M_{55}	M_{47}	M_{39}	M_{31}	M_{23}	M_{15}	M_7

If we then take the inverse permutation $Y = IP^{-1}(X) = IP^{-1}(\mathrm{IP}(M))$, it can be seen that the original ordering of the bits is restored.