U-I Foundations of Cryptography

What is Cryptography?

Is art or science of Secret writing? It concerned with the developing algorithms to

- ✓ To conceal the content of messages from all except sender and recipient
- ✓ To verify the correctness of message or its sender and recipient

Cryptography is art or science of transforming intelligible message to unintelligible and again transforming that message back to the original form

Terminologies

- ✓ **Encryption(Enciphering) :**Process of encoding the message so that meaning is not obvious or not in understandable form
- ✓ **Decryption(Deciphering):** Reverse process of encryption
- ✓ **Plaintext:** The original form of the message
- ✓ **Cipher text:** Disguised(encrypted) message

P- plain-text

C- Cipher text

E- Encryption algorithm

D- Decryption algorithms

C= E (p)

P= D(C)

P= D(E (P))

- ✓ **Key :** Critical (secret) information used in cipher and known only to sender and receiver

  Symmetric – Shared key

  Asymmetric – Public key

- ✓ **Code:** Algorithm used for transforming the intelligible (plain text) to unintelligible (cipher text)

- ✓ **Cipher:** Is algorithm /Code used for transforming plaintext to cipher text

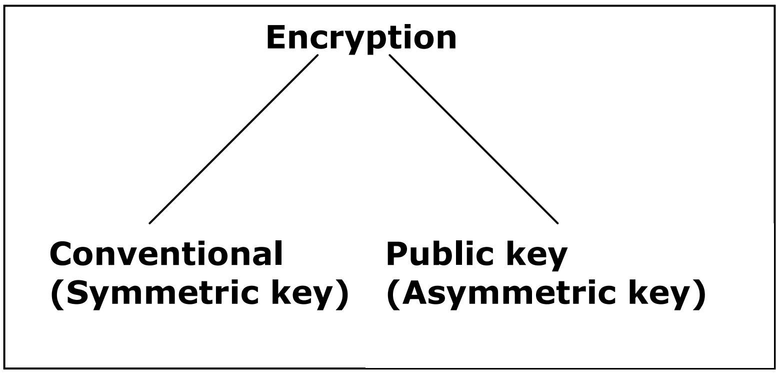
- ✓ **Cryptanalysis (Code breaking):** Study of method for transforming cipher text to plaintext without having knowledge of any key

- ✓ **Cryptology :** Area of cryptography and cryptanalysis together is called as cryptology

Types of ciphers:

There are two types of ciphers

1. Stream cipher : Converts plaintext to cipher text one bit at time
2. Block cipher : It takes a given length of data as input and produces different length of encrypted data



Conventional (Symmetric key) Cryptography:

Symmetric key cryptography Is also termed as private or secret key encryption because secret key is shared between sender and receiver

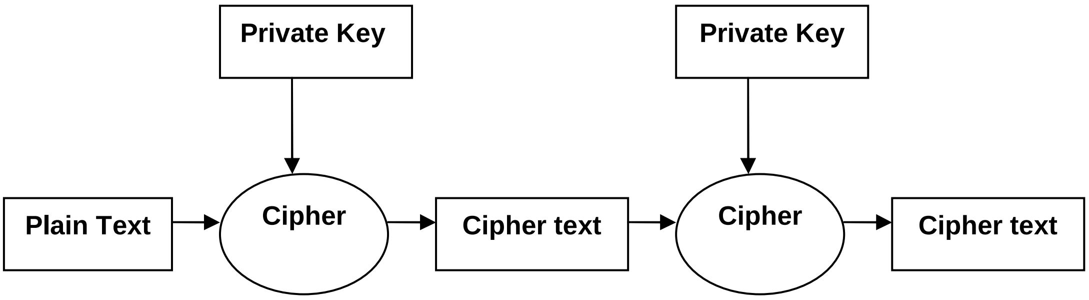


Fig Symmetric Encryption

Asymmetric cryptography:

- ✓ Developed in 1970
- ✓ Two keys are involved in asymmetric encryption
- ✓ One key is used by sender to encrypt the data and other by receiver to decrypt the data
- ✓ Both the keys are reversible also
- ✓ Generally public keys are used for encryption of data while private keys are used for decryption of data

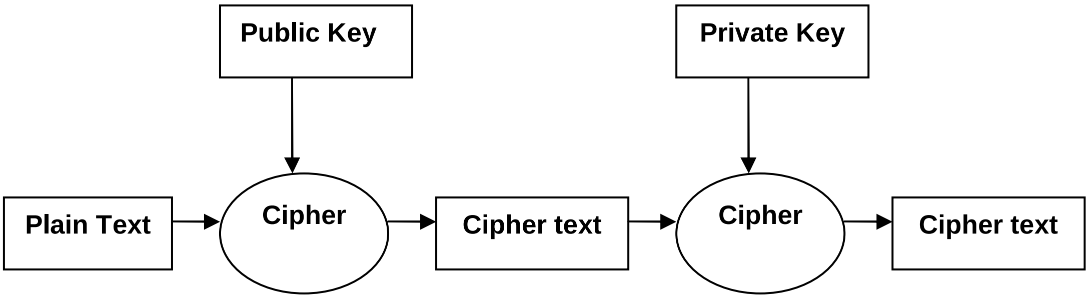


Fig Asymmetric Encryption

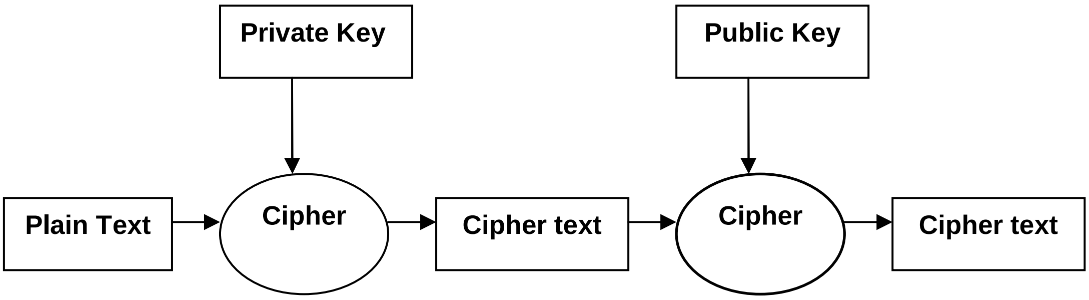


Fig Asymmetric Encryption

Why do we need Cryptography?

Computers are used by millions of people for many purposes

- ✓ Banking
- ✓ Shopping
- ✓ Tax returns
- ✓ Protesting
- ✓ Military
- ✓ Student records
- ✓ …

Privacy is a crucial issue in many of the above applications

Cryptography techniques would provide the solution to make sure that nosy people cannot read or secretly modify messages intended for other recipients

Objectives of Network Security:

- ✓ **Availability :** Ensures the availability of desired resource (i.e. when there is need of specific resource or service it must be available for access)
- ✓ **Confidentiality :** only sender and receiver can understand the message (to achieve this sender encrypts message with specific algorithm while receiver decrypts message)
- ✓ **Integrity:** Sender and receiver may have provision to check the integrity of data & get themselves ensured that message is not altered in transit(during communication)
- ✓ **Anonymity :** Ensures the privacy of the origin of data (i.e. receiver must have some mechanism to check that he is receiving data from a specific sender)
- ✓ **Authenticity :** Sender or receiver want to confirm the identity of each other and may be possible they would access some service after giving there authentication
- ✓ **Authorization:** Access to the resources are authorized after authentication

Security issues:

The world before computers was in some ways much simpler

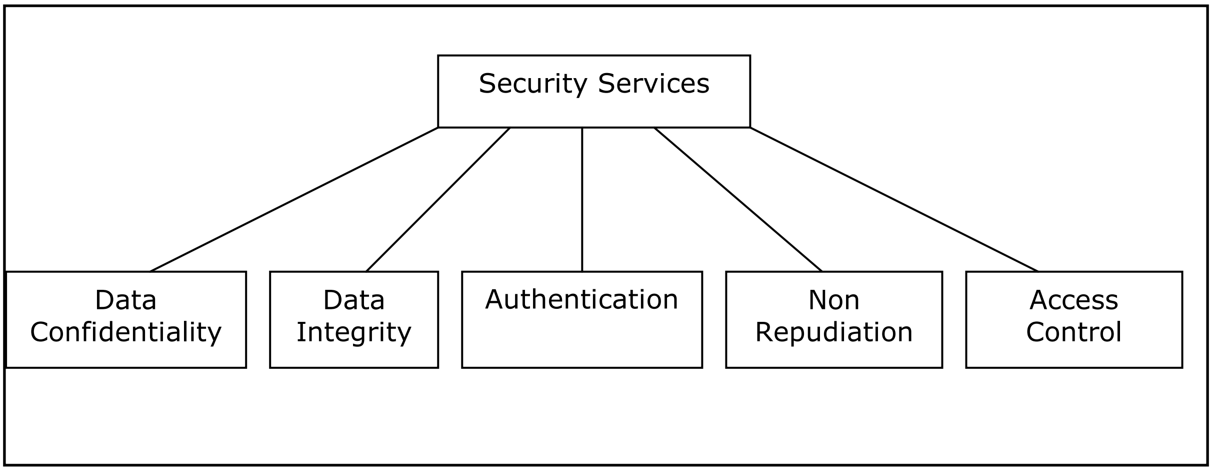
- ✓ Signing, legalizing a paper would authenticate it
- ✓ Photocopying easily detected
- ✓ Erasing, inserting, modifying words on a paper document easily detectable
- ✓ Secure transmission of a document: seal it and use a reasonable mail carrier (hoping the mail train does not get robbed)
- ✓ One can recognize each other's face, voice, hand signature, etc.

Electronic world: the ability to copy and alter information has changed dramatically

- ✓ No difference between an "original" file and copies of it
- ✓ Removing a word from a file or inserting others is undetectable
- ✓ Adding a signature to the end of a file/email: one can impersonate it –add it to other files as well, modify it, etc.
- ✓ Electronic traffic can be (and is!) monitored, altered, often without noticing
- ✓ How to authenticate the person electronically communicating with you

Security attack: Any action that comprises the security of information owned by an organization

Security Services: A service that enhances the security of data processing system and information transfer of organization



Data Confidentiality: Designed to protect data from disclosure attack. The service is defined by X.800 and it provides confidentiality for the whole message or the part of message and also offers protection against traffic analysis i.e. designed to prevent sniffing and traffic analysis

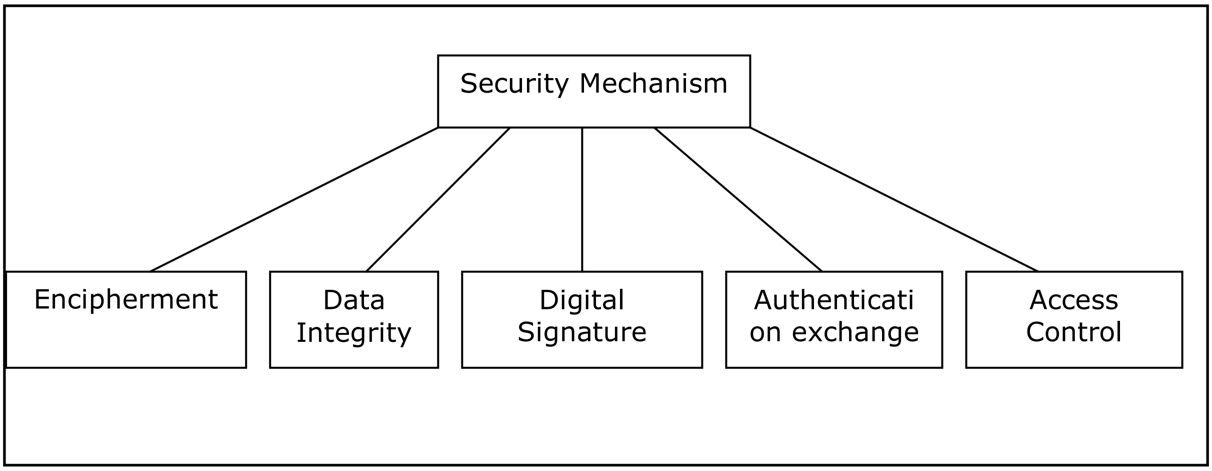
Data Integrity: Is designed to ensure the integrity of data as it protects data from modification, insertion, deletion and replaying by intruder or hacker

Authentication: This service checks authenticity of communicating parties

Nonrepudiation: This service protects against repudiation by either sender of receiver

Access Control: Provides protection against unauthorized access to data

Security Mechanism: A mechanism that is designed to detect, prevent or recover from security attack



Encipherment: Hiding or covering data can provide confidentiality. Today two techniques cryptography and Steganography are used for enciphering

Data Integrity: Sender and receiver ensures integrity of data on the basis of checksum values

Digital Signature: is means by which sender can electronically sign the data and the receiver can electronically verify the signature

Authentication Exchange: Two end users exchange some message to prove their identity

Access Control: Uses method to prove that a user has access right to data or resource owned by the system

Possibilities of Network Security attack:

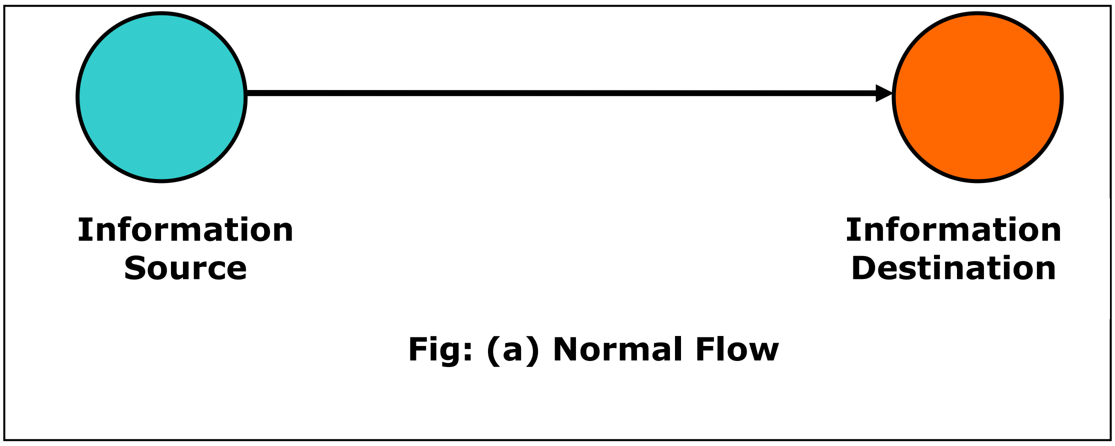


Figure: Shows normal flow of data from Information Source to Information Destination

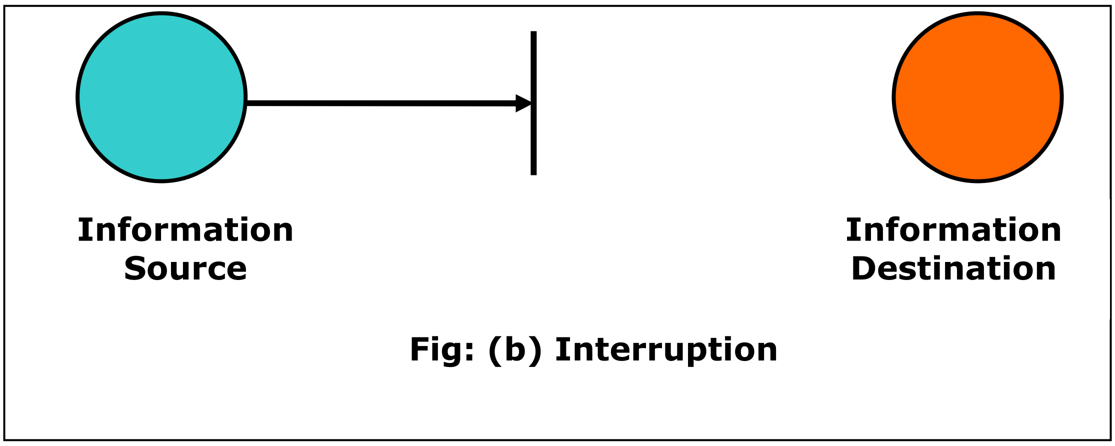


Figure: Shows Interruption of channel between source & Destination

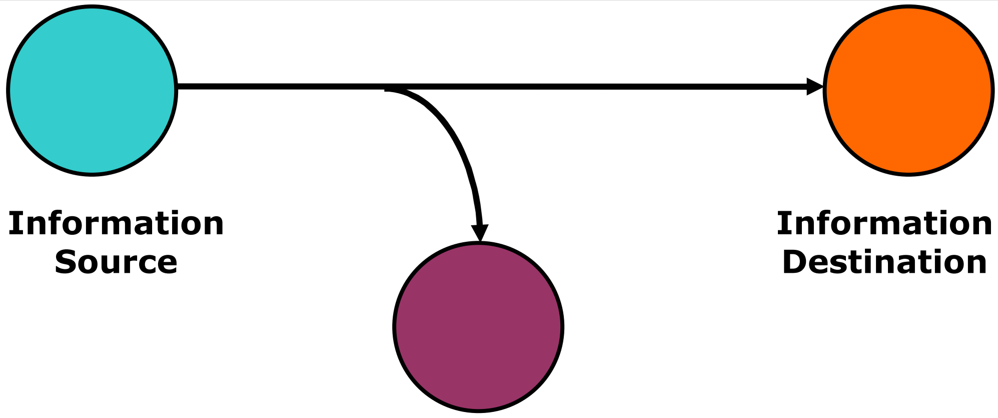


Fig: (c) Interception

Figure: Shows Interception of Data between source & Destination where some intruder is listening ongoing channel

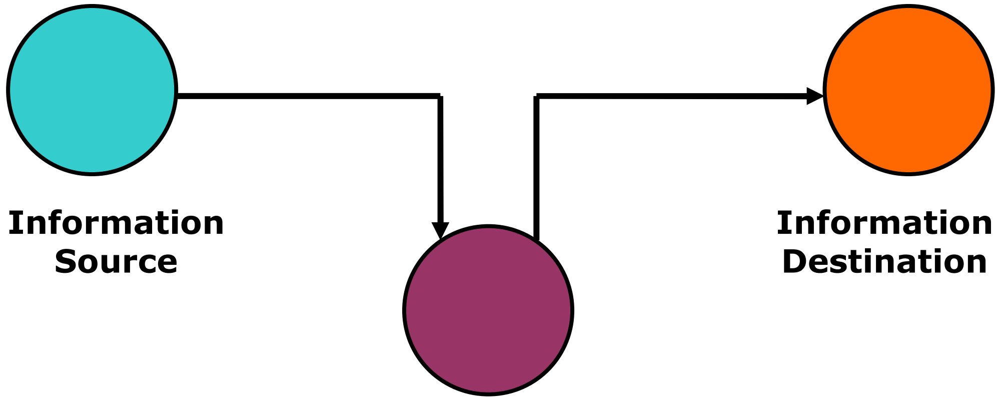


Fig: (d) Modification

Figure: Shows Modification of Data between source & Destination where intruder is modifying the channel

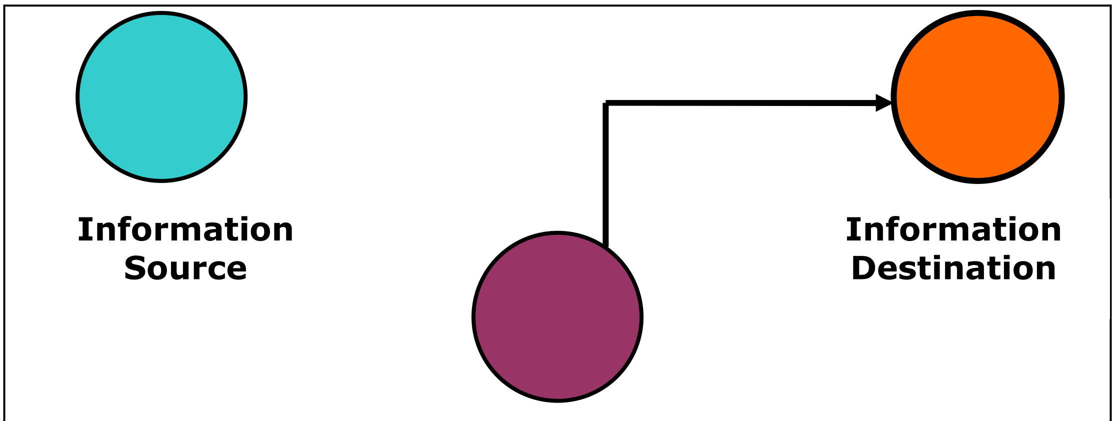


Fig: (e) Fabrication

Figure: Shows Fabrication of Data between source & Destination where intruder fabricates data and divert it towards receiver

Possible attackers:

1. Student: to have fun snooping on other people's email
2. Cracker: to test out someone's security system, to steal data
3. Businessman: to discover a competitor's strategic marketing plan
4. Ex-employee: to get revenge for being fired
5. Accountant: to embezzle money from a company
6. Stockbroker: to deny a promise made to a customer by email
7. Convict: to steal credit card numbers for sale
8. Spy: to learn an enemy's military or industrial secrets
9. Terrorist: to steal germ warfare secrets

Security issues: Some Practical Situations

1. A sends a file to B: E intercepts it and reads it

   How to send a file that looks gibberish to all but the intended receiver?

2. A send a file to B: E intercepts it, modifies it, and then forwards it to B

   How to make sure that the document has been received in exactly the form it has been sent

3. E sends a file to B pretending it is from A

   How to make sure your communication partner is really who he claims to be

4. A sends a message to B: E is able to delay the message for a while

   How to detect old messages?

5. A sends a message to B. Later A (or B) denies having sent (received) the message

   How to deal with electronic contracts?

6. E learns which user accesses which information although the information itself remains secure

   E prevents communication between A and B: B will reject any message from A because they look unauthentic

Friends and Enemies: Alice, Bob, Trudy:-

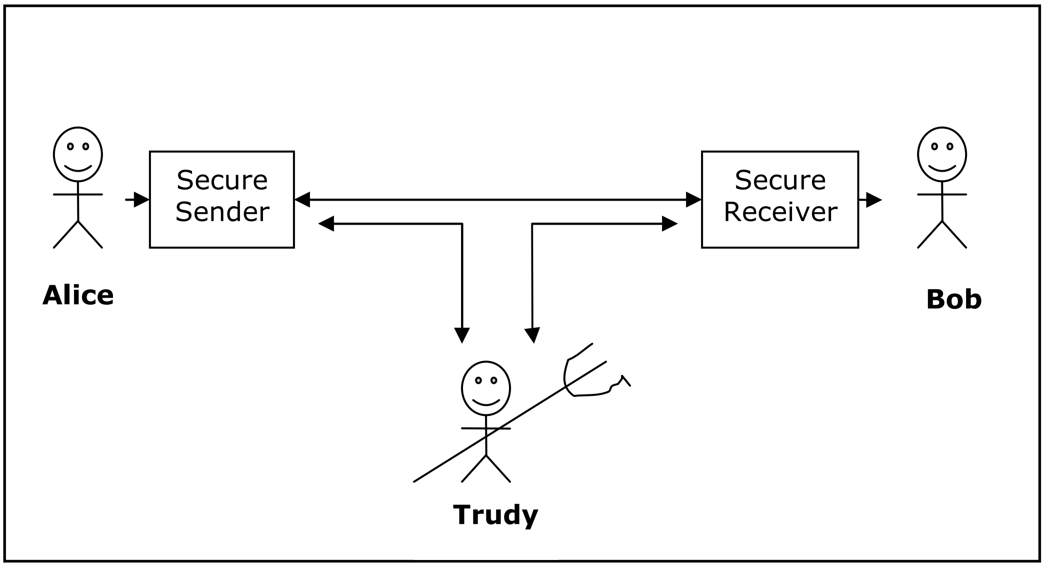


Figure: Shows well known example of Network security world

- ✓ Alice, Bob and Trudy are well known in network security world
- ✓ Bob and Alice are lovers and want to communicate securely with each other
- ✓ Trudy is the intruder(cruel lady) may intercept , delete ,modify and fabricate message

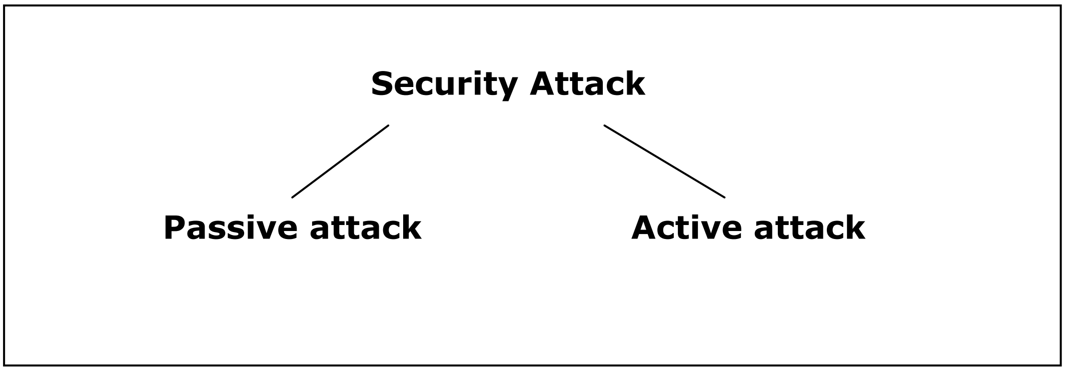
What can a bad guy (intruder)/ cruel lady (Trudy) do?

- ✓ **Eavesdrop:** intercepts message
- ✓ Actively insert message or data in ongoing connection
- ✓ **Impersonation:** Can fake(spoof) source address in packet(or any field in packet)
- ✓ **High jacking:** “Take Over” ongoing connection by removing sender or receiver inserting himself in place
- ✓ **Denial of service :** Prevent service from being used by others(i.e. by overloading the resources)

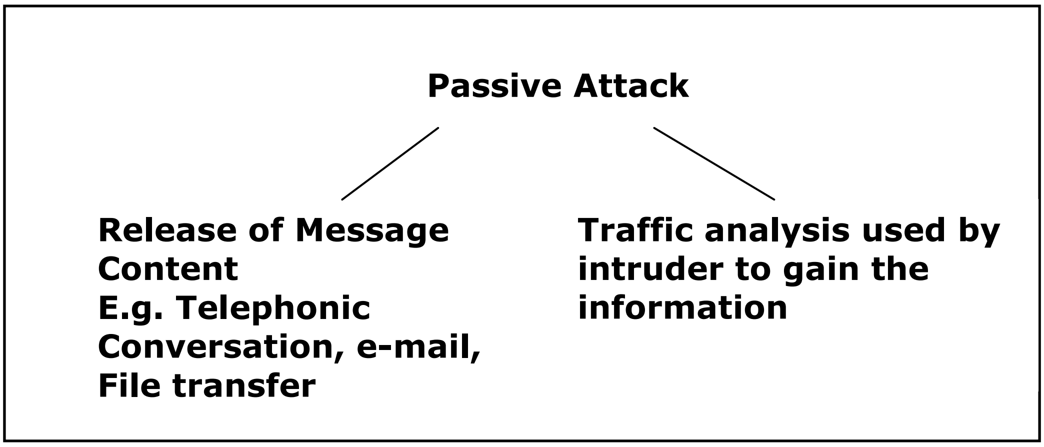
Types of Security attacks:

Security attacks are categorized in main two categories

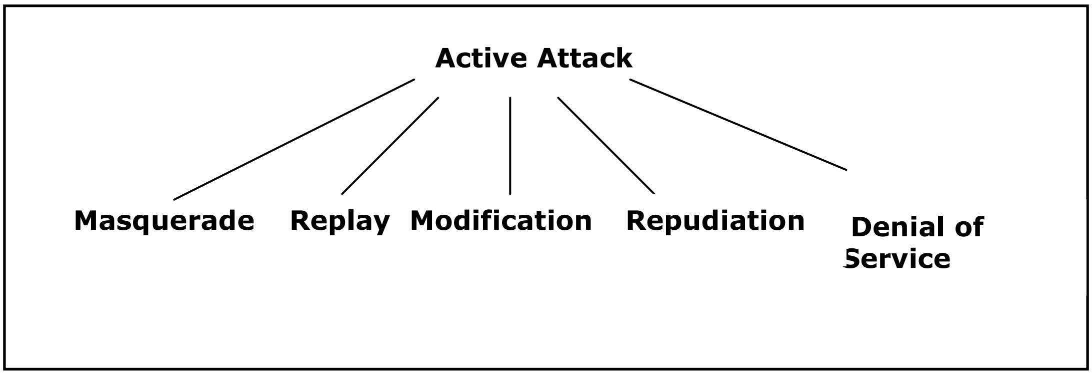
- ✓ Passive attack
- ✓ Active attack



Passive attacks: they are having the nature of eavesdropping or monitoring of transmitting channel or packet sniffing (Here intruder simply listen the ongoing channel and grab important information later on makes use of grabbed data for analysis)


Passive Attack
Release of Message Content
E.g. Telephonic Conversation, e-mail, File transfer
Traffic analysis used by intruder to gain the information

Active attacks: Involves some modification of data stream or creation of false stream



Masquerade: Takes place when one entity pretends to be different entity

Reply: Involves passive capture of data unit and its subsequent transmission to produce unauthorized effect

Modification: Portion of message is altered

Repudiation : This type of attack is different from others as its not performed by third party but it is performed by one of the two parties in the communication i.e. sender and receiver

In this case either sender of message may deny that he has sent message; or the receiver of the message might later deny that he has received message

Denial of service: Disruption of entire network by overloading the different services

Attacks	Passive/Active	Threatening
Sniffing/Traffic Analysis	Passive	Confidentiality
Modification Masquerading Replaying Repudiation	Active	Integrity
Denial of service	Active	Availability

Table: Categorization of passive/Active attacks

Model for Network Security:

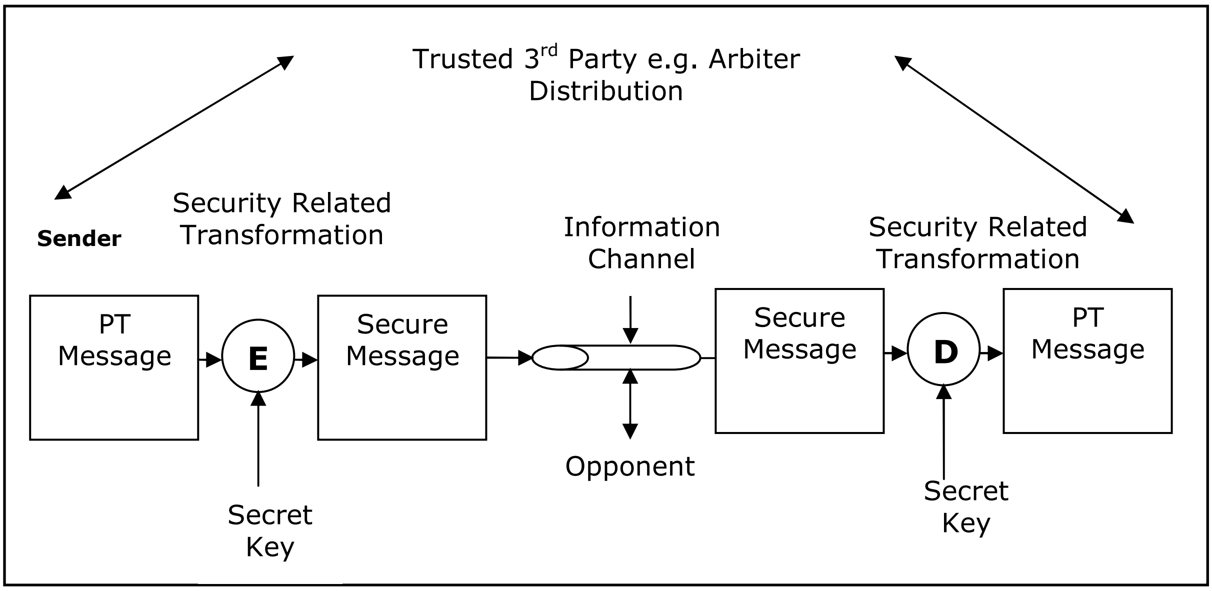


Fig: Model for Network Security

A message to be transformed from one party to another across network, the two parties who are the principals in the transaction must have to cooperate for the exchange to take place through logical information channel

General Model shows that there are basic four tasks

- ✓ Design an algorithm for performing the security related transformation
- ✓ Generate the secret information to be used with algorithms
- ✓ Develop methods for distribution and sharing of secret information
- ✓ Specify the protocols to be used

Symmetric Cipher Model:

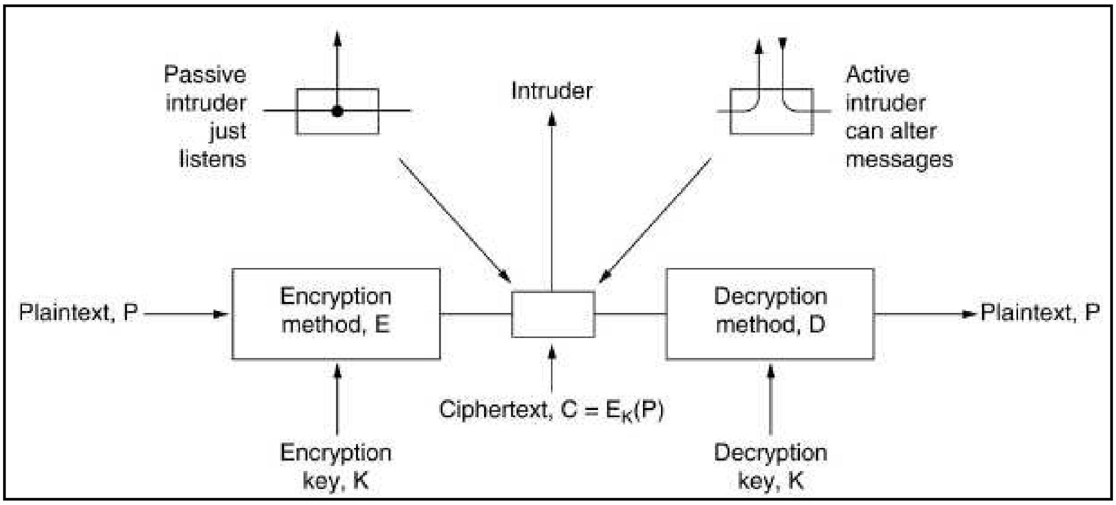


Fig: Symmetric Cipher Model

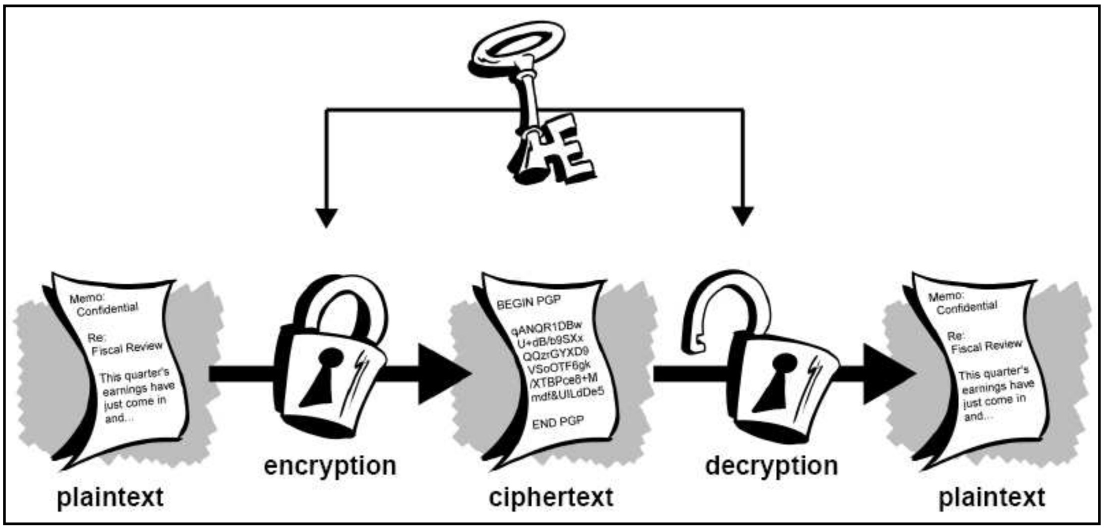


Fig: Simplified Symmetric Cipher Model

Symmetric cipher model has five ingredients

1. Plaintext
2. Encryption algorithms
3. Secret Key
4. Cipher text
5. Decryption algorithms

There are major two requirements for secure use of conventional cryptosystem

- ✓ Opponent should not be able to decipher the ciphertext or discover the key even if he/she is having the ciphertext

- ✓ Sender and receiver must have obtained the secret key in secure fashion

We assume that it’s impractical to decrypt(decipher) the message on the basis of algorithmic knowledge and ciphertext i.e. no need to keep secrecy of algorithm

So with the use of symmetric encryption principle security problem lies in to maintain the secrecy of the secret key

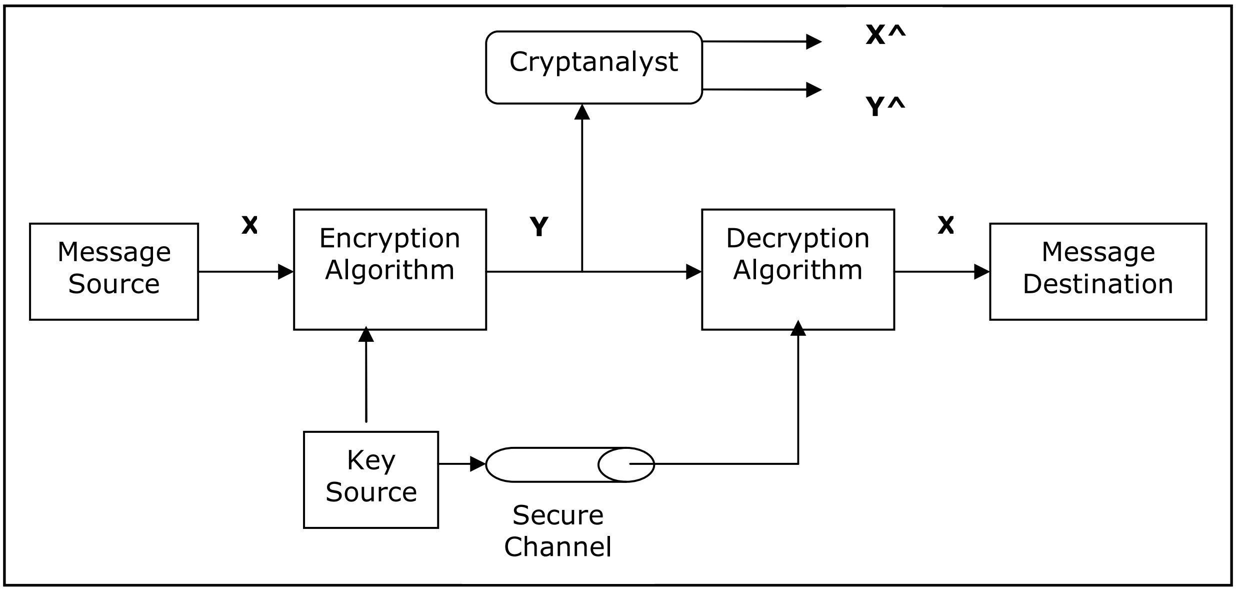


Fig: Modified Model of conventional Cryptosystem

As shown in Fig Source produces message in plaintext

X= [x1,x2,x3, - - - - - X_m] where m- is element of X are letters in some finite alphabet

For encryption a key of the form

K= [k1, K2, - - - - - - - - -k_m] is generated

If the key is generated at the message source then it must also be provided to the destination by means of some secure channel

So with message X and encryption key K as input encryption algorithm produces ciphertext

Y=[y1, y2, - - - - - - - -y_n]

So we can write $Y=E_k(X)$

i.e. ciphertext Y is produced with encryption and at Receiver end ciphertext is inverted to produce plaintext

$X= D_k(Y)$

Packet sniffing/snooping:

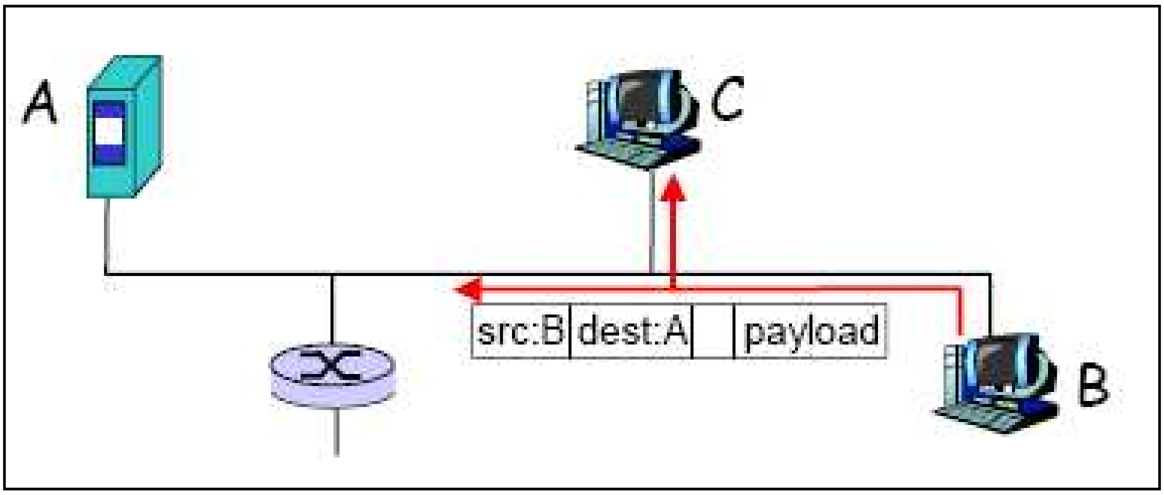


Fig: Shows Packet sniffing where C sniffs packets of A

- ✓ As shown in fig Computer **A** and **B** are genuine users , **B** is diverting data to **A** but in between them intruder **C** is listening the ongoing communication
- ✓ Our channel acts as broadcast media i.e. Packet intended from B to A also passes through C
- ✓ Promiscuous NIC reads all packets passing by, it grabs the important information passing through it
- ✓ can read all unencrypted data (e.g. passwords)
- ✓ e.g.: C sniffs B's packets

IP Spoofing:

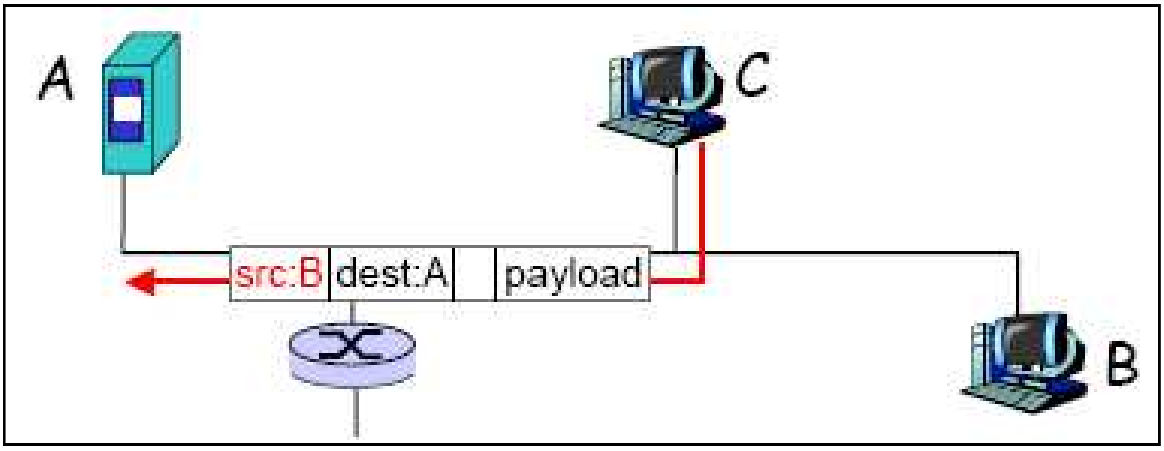


Fig: Shows Packet spoofing where C pretends himself as B

- ✓ Based on sniffed information C fabricates a packet but in packet it writes source address as computer B
- ✓ i.e. C can generate “raw” IP packets directly from application, putting any value into IP source address field
- ✓ receiver can’t tell if source is spoofed
- ✓ e.g.: C pretends to be B

Denial of service (DOS):

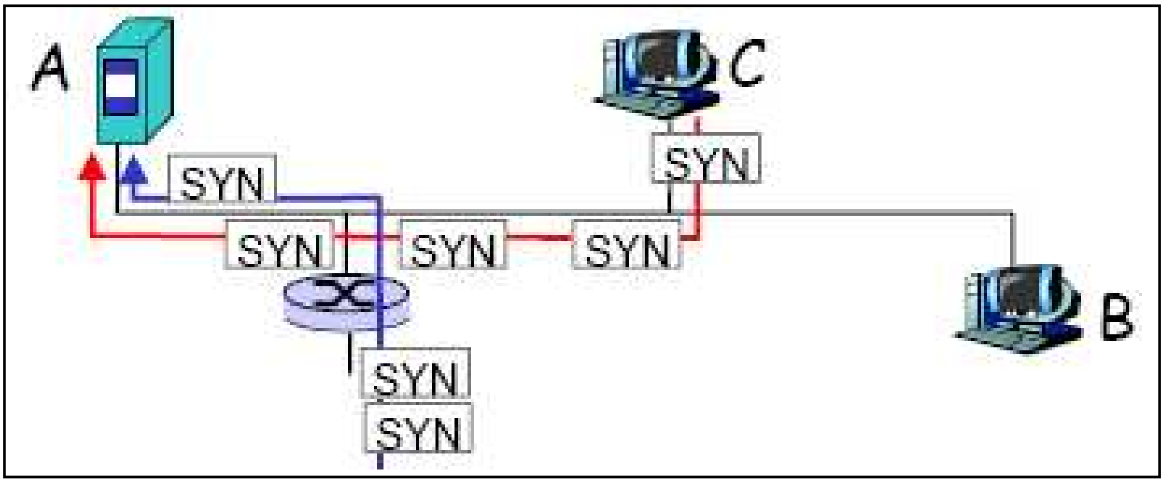


Fig: Shows Denial of service attack

- ✓ Here major objective of intruder is to overload the service/server so that it would deny to provide the service
- ✓ For overloading the service generally intruders are writing some sort of script/code that would divert maliciously generated packets in the form of request
- ✓ Flood of maliciously generated packets “swamp” receiver
- ✓ Distributed DOS (DDOS): multiple coordinated sources Swamp receiver
- ✓ e.g., C and remote host SYN-attack A

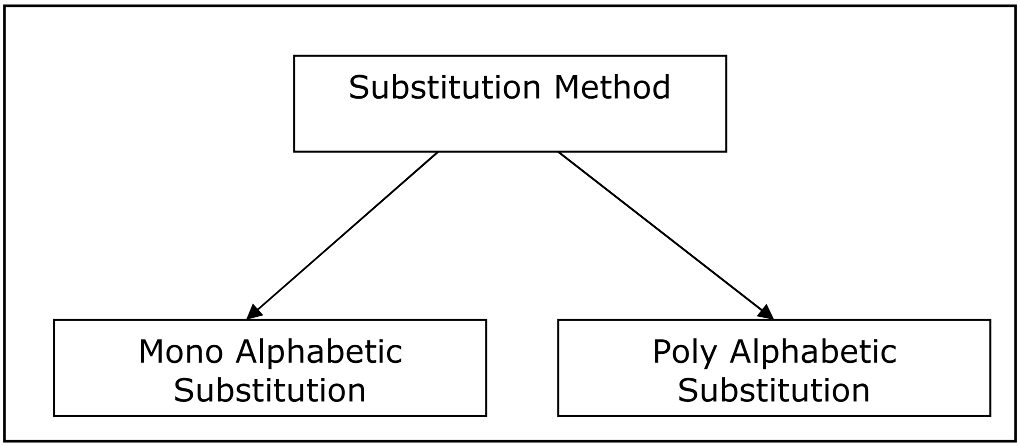
Cryptographic Techniques:

All cryptographic algorithms are based on following two techniques

- ✓ Substitution
- ✓ Transposition (Permutation)

Substitution Technique: Is one in which the letters of the plaintext are replaced by other letters (i.e. Fixed symbols or alphabets)

Transposition Technique: Method of disguising text or alphabet by shuffling or exchanging their position



Monoalphabetic Substitution: Here substitution of an alphabet takes place with the fixed alphabet throughout the PT

Poly Alphabetic Substitution: Here substitution of an alphabet takes place with more than one alphabet (i.e. not with specific fixed alphabet)

Caesar’s Cipher

The earliest known use of substitution cipher was given by Julius Caesar for exchanging military secret information before 2000 years

An extremely simple example of conventional cryptography is a substitution cipher. A substitution cipher substitutes one piece of information for another.

The Caesar cipher involved in replacing each letter of alphabet with the letter standing three places further down the alphabet

For example, if we encode the word “SECRET” using Caesar’s key value of 3, we offset the alphabet so that the 3rd letter down (D) begins the alphabet.

Where D=A, E=B, F=C, and so on.

So starting with ABCDEFGHIJKLMNOPQRSTUVWXYZ and sliding everything up by 3, you get DEFGHIJKLMNOPQRSTUVWXYZABC

i.e.

P.T.: A	B	C	D	E	F	G	……..	Z
C.T.: D	E	F	G	H	I	J	………	C

Now let’s assign numerical value (NV) to each letter

P.T.: A	B	C	D	E	F	G	……..	Z
N.V.: 0	1	2	3	4	5	6	……..	25

The algorithms can be expressed as

For plaintext letter p, substitute the ciphertext letter c3

C= E (p) = (p+3) mod 26

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

C=E(P) = (P+K) mod(26)

Where K takes a value in the range of 1 to 25 and decryption algorithm is

P = D(C) = (C – K) mod 26

Drawback of Caesar Cipher:

Major problem of Caesar cipher is language regularity due to which there is possibility that cryptanalysis may guess the message present in CT

Language regularity is based on the frequency of letter occurrence

- ✓ Letter **E** is more frequent then
- ✓ T R I O A S Then
- ✓ Rarely used is J K Q X Z
- ✓ Letter **E** is **25** times more frequent than the **Q**

Example of language Regularity: (Caesar Monoalphabetic Substitution)

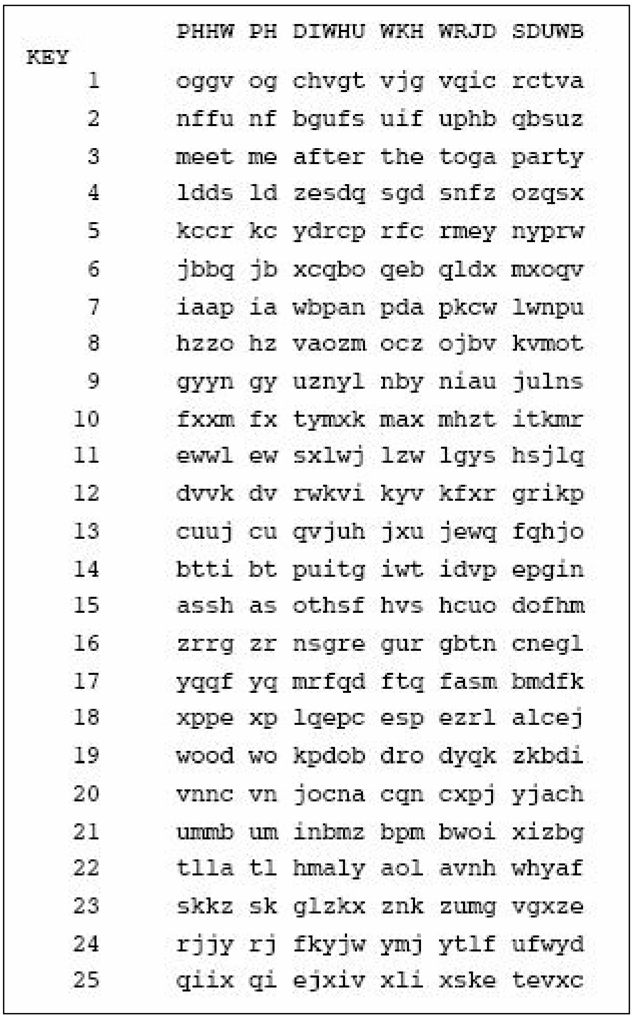
P.T.:	A	B	C	D	E	F	G	……..	Z
C.T.:	D	E	F	G	H	I	J	………	C

C.T.:	WTIGMEP	WTIEOIV	GSQMRR
P.T.:	SPECIAL	SPEAKER	COMING

As shown appearance frequencies of letters words and pairs of letters accelerates the identification of certain letters

Attacking Caesar Cipher:

- ✓ Caesar can be broken if we only know one pair (plain letter, encrypted letter)
- ✓ The difference between them is the key
- ✓ Caesar can be broken even if we only have the encrypted text andno knowledge of the plaintext
- ✓ *Brute-force attack*is easy: there are only 25 keys possible
- ✓ Try all 25 keys and check to see which key gives an intelligible message



KEY	PHHW PH DIWHU WKH WRJD SDUWB
1	oggv og chvgt vjg vqic rctva
2	nffu nf bgufs uif uphb qbsuz
3	meet me after the toga party
4	ldds ld zesdq sgd snfz ozqsx
5	kccr kc ydrcp rfc rmey nyprw
6	jbbq jb xcqbo qeb qldx mxoqv
7	iaap ia wbpan pda pkcw lwnpu
8	hzzo hz vaozm ocz ojbv kvmot
9	gyyn gy uznyl nby niau julns
10	fxxm fx tymxk max mhzt itkmr
11	ewwl ew sxlwj lzw lgys hsjlq
12	dvvk dv rwkvi kyv kfxr grikp
13	cuuj cu qvjuh jxu jewq fqhjo
14	btti bt puitg iwt idvp epgin
15	assh as othsf hvs hcuo dofhm
16	zrrg zr nsgre gur gbtn cnegl
17	yqqf yq mrfqd ftq fasm bmdfk
18	xppe xp lqepc esp ezrl alcej
19	wood wo kpdob dro dyqk zkbdi
20	vnnc vn jocna cqn cxpj yjach
21	ummb um inbmz bpm bwoi xizbg
22	tlla tl hmaly aol avnh whyaf
23	skkz sk glzkx znk zumg vgxze
24	rjjy rj fkyjw ymj ytlf ufwyd
25	qiix qi ejxiv xli xske tevxc

Fig: Brute Force Cryptanalysis of Caesar Cipher

Why is Caesar easy to break?

- ✓ Only 25 keys to try
- ✓ The language of the plaintext is known and easily recognizable
- ✓ What if the language is unknown?
- ✓ What if the plaintext is a binary file of an unknown format?

Playfair Cipher:

- Multiple letter encryption method
- Invented by Sir Charles Wheatstone in 1854, but named after his friend Baron Playfair who championed the cipher at the British foreign office
- Encrypts pair of letters at each step
- Use words in language as key and build a 5*5 matrix (table of letters) in the key and other letters(I is considered the same as J)
- This is called **key matrix**

A 5X5 matrix of letters based on a keyword

- Fill in letters of keyword (no duplicates)
- Left to right, top to bottom
- Fill the rest of matrix with the other letters in alphabetic order E.g. using the keyword **MONARCHY**, we obtain the following matrix

  Key: - MONARCHY

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

Rules of Substitution:

The plaintext is encrypted **two letters at a time**:

1. Repeated letters in plaintext are replaced with filler letter such as Z

E.g. "BALLOON" is treated as "BALZLOZON" & SUNNY is treated as SUNZNY

2. Form the pair of alphabets if letters are not having even alphabet then add filler alphabet Z at end

3. If both letters fall in the same row of the key matrix, replace each with the letter to its right (wrapping back to start from end), e.g. “AR" encrypts as "RM"

4. If both letters fall in the same column, replace each with the letter below it (again wrapping to top from bottom), e.g. “MU" encrypts to "CM"

5. Otherwise each letter is replaced by the one in its row in the column of the other letter of the pair, e.g. “HS" encrypts to "BP", and “EA" to "IM" or "JM" (as desired)

6. Decryption works in the reverse direction

7. The examples above are based on this key matrix

PT: SUNNY **PAIRS:** SU NZ NY **CT:** LX RW YG

PT: BALLOON **PAIRS:** BA LZ LO ZO NZ **CT:** IB TU PM VR RZ

Decryption works in the reverse direction

✓ The examples above are based on this key matrix:

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

Security much improved over Monoalphabetic

- ✓ There are 26 x 26 = 676 diagrams
- ✓ Needs a 676 entry diagram frequency table to analyze (vs. 26 for a Monoalphabetic) and correspondingly more ciphertext
- ✓ Widely used for many years (e.g. US & British military in WW I, other allied forces in WW II)
- ✓ Can be broken, given a few hundred letters
- ✓ Playfair cipher may attack based on appearance frequency of letters but still subject to an attack

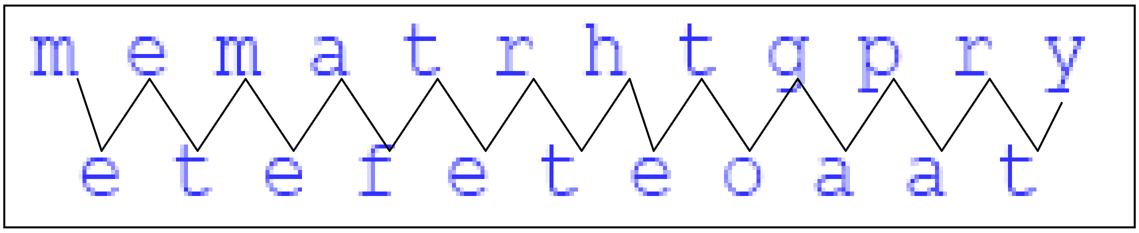
Transposition Method:

- ✓ Perform some sort of permutation on the plaintext letters
- ✓ Hide the message by rearranging the letter order without altering the actual letters used
- ✓ The simplest such technique: *rail fence technique*

Rail fence Cipher

- ✓ Got the name from the structure of Rail fence
- ✓ Idea: write plaintext letters diagonally over a number of rows, and then read off cipher row by row

  E.g., with a rail fence of depth 2, to encrypt the text “meet me after the toga party”, write message as:



m e m a t r h t g p r y

e t e f e t e o a a t

Ciphertext is read from the above row-by-row

CT: MEMATRHTGPRYETEFETEOAAT

Attack: this is easily recognized because it has the same frequency distribution as the original text

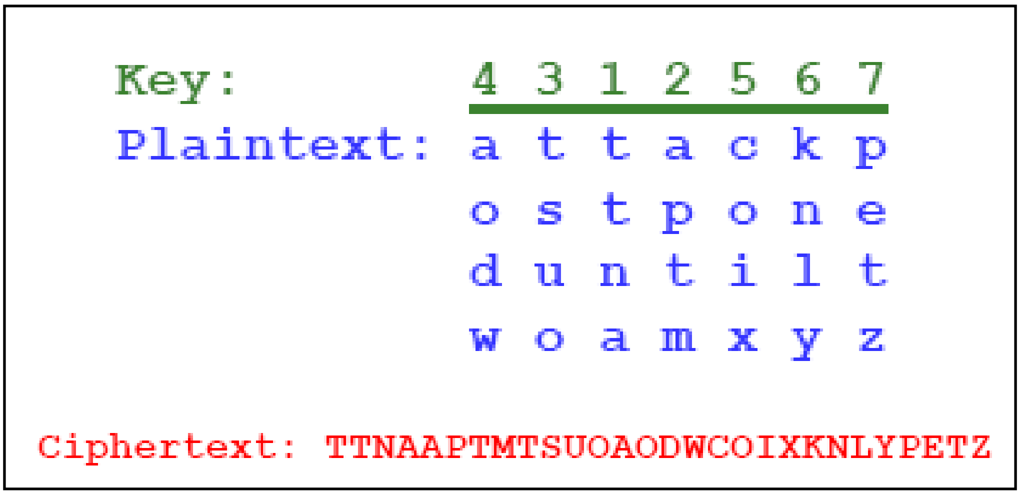
Row Column Cipher:

More complex scheme: **row transposition**

Write letters of message in rows over a specified number of columns

Reading the crypto text column-by-column, with the columns permuted according to some key

Example: "attack postponed until two am" with key 4312567: first read the column marked by 1, then the one marked by 2, etc**.**



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

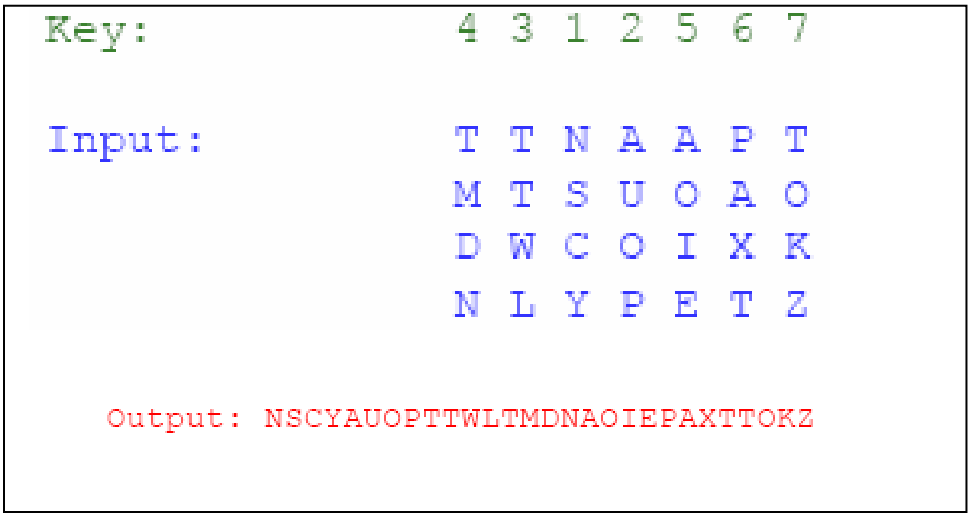
Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

If we number the letters in the plaintext from 1 to 28, then the result of the first encryption is the following permutation of letters from plaintext: 03 10 17 24 04 11 18 25 02 09 16 23 01 08 15 22 05 12 19 26 06 13 20 27 07 14 21 28

- ✓ Note the regularity of that sequence!
- ✓ Easily recognized!

Repeated Row Column

Idea: use the same scheme once more to increase security



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

After the second transposition we get the following sequence of letters:

17 09 05 27 24 16 12 07 10 02 22 20 03 25 15 12 04
23 19 14 11 01 26 21 18 08 06 28

This is far less structured and so, more difficult to cryptanalyze

Basics of Abstract Algebra

Group(G, •,e): a set G with a binary operation •and an element e∈G satisfying the following laws:

Associativity: a •(b •c)=(a •b) •c for any a,b,c∈G

Identity element: a •e=e •a=a, for any a∈G

Inverse element: for each a∈G, there exists an element a’∈G such that a •a’= a’•a=e.

a’is usually denoted as -a and is called the *inverse*of a

***Example*of a group:** the set of integers with the addition (Z,+,0)

Note that the set of integers with the multiplication (Z,x,1) isnot a group: the inverse element does not exist for all integers (it exists only for 1 and −1)

A group (G,+,e) is called:

Commutative(or *abelian*) if a •b=b •a for all a,b in G
Finite if set G is finite
Infinite if set G is infinite

Example:

(Z,+,0) is a commutative group

The set of nxn matrices over integers, with the addition, is a commutative group

The set of permutations of the set {1,2,…,n} with the composition, is a finite non-commutative group

Rings:

Ring(R,+,•,0): a set R with two binary operations + and •satisfying the following laws:

(R,+,0) is a commutative group

Associative multiplication: a•(b•c)=(a•b)•c for any a,b,c∈R

Distributive: a•(b+c)=a•b + a•c; (a+b)•c=a•c + b•c

A ring (R,+,•,0) is called:

Commutative if the multiplication •is commutative

Unitary (or with *unity element*) if operation •has an identity element 1: a•1=1•a=a, for all a in R. We denote it as (R,+,•,0,1)

Integral domain if

- ✓ It is commutative
- ✓ It has unity element
- ✓ It has no zero divisors: if a•b=0, then either a=0, or b=0

Example:

(Z,+,•,0,1) is an integral domain

The set of nxnmatrices over integers with addition and multiplication is a commutative unitary ring, but not an integral domain

(Z26, +,•,0,1) is a commutative unitary ring, but not an integral domain:2•13=0 (mod 26)

Fields:

Field (F, +, •, 0, 1):
(F, +, •, 0, 1) is an integral domain

Multiplicative inverse: for any nonzero element *a i*n F there exists an element *a'* in F such that *a*•*a'*=*a'*•*a*=1
a' is usually denoted as a^{-1} and it is called the multiplicative inverse of *a*

Example:
The set of rational numbers (Q, +, •, 0, 1), the set of real numbers (R, +, •, 0, 1) are fields

The set of integers (Z, +, •, 0, 1) is not field: only 1 and −1 have multiplicative inverses

(Z_{26}, +, •, 0, 1) is not field

(Z_3, +, •, 0, 1) is a finite field: the inverse of 1 is 1 and the inverse of 2 is 2
(Z_5, +, •, 0, 1) is a finite field: 1•1=1 mod 5, 2•3=1 mod 5, 4•4=1 mod 5
Inverse of 1 is 1, inverse of 2 is 3, inverse of 3 is 2, and inverse of 4 is 4

Finite Fields:

It can be proved that if a field is finite then it has p^n elements, for some prime number p

We also say that it has order p^n

We denote GF(p^n) –GF stands for Galois field

For n=1 we have GF(p) which is Z_p

If p is prime, then any element in Z_p has a multiplicative inverse

For n>1 the field has a different structure

Start from Z_p and build a field with p^n elements

Modular Arithmetic:

Consider the set of integers: fix a positive integer n

For any integer a, there exists integers q and r such that a=qn+r and r is from 0 to n-1

q is the *largest integers* less than or equal to a/n

r is called the *residue* of a modulo n

Define the operator *mod*: a mod n=r

Define the operator *div*: a div n=q

Example:

7 mod 5 = 2, 11 mod 7 =4,

-11 mod 7 =3: -11=(-2).7+3

Congruence modulo n: a≡b mod n if a mod n = b mod n

Example: 73 ≡4 mod 23, 21 ≡-9 ≡1mod 10

Greatest Common Divisor:

The positive integer d is the greatest common divisor of integers a and b, denoted d=gcd(a, b) if

It is a divisor of both a and b

Any other divisor of a and b is a divisor of d

Example: gcd (8, 12) =4, gcd(24,60)=12

Integers a and b are called *relatively prime* if gcd (a, b) =1

Computing gcd (a, b): Euclid’s algorithm

Based on the following fact:

- ✓ gcd(a, 0) then gcd=a
- ✓ gcd (a, b) =gcd (b, a mod b)

Euclid's Algorithm to compute gcd (a, b) –

```
Euclid (a, b)

If b=0 then return a

Else return Euclid (b, a mod b)
```

Steps:

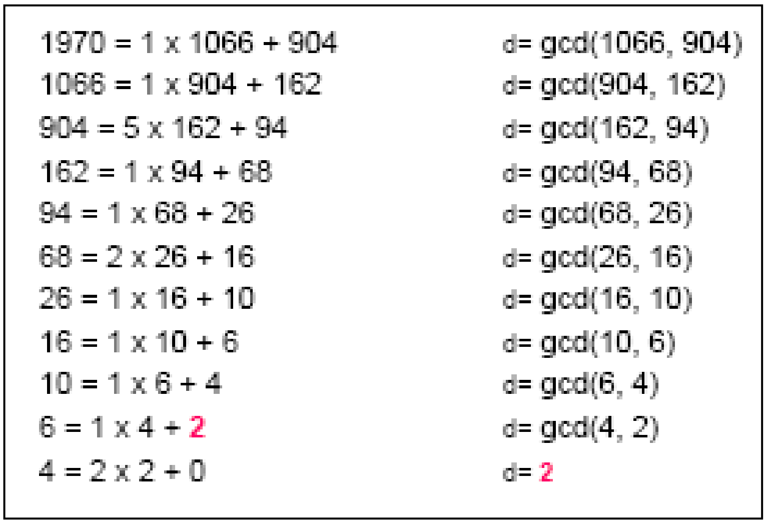
```
    gcd (a, b)

1. A =a, B=b
2. if B=0
3. return A=gcd(a, b)
4. R=A mod B
5. A=B
6. B=R
7. go to step 2
```

Note: the algorithm always terminates with solution i.e. gcd

Example:

D=gcd (1970, 1066)



1970 = 1 x 1066 + 904	d= gcd(1066, 904)
1066 = 1 x 904 + 162	d= gcd(904, 162)
904 = 5 x 162 + 94	d= gcd(162, 94)
162 = 1 x 94 + 68	d= gcd(94, 68)
94 = 1 x 68 + 26	d= gcd(68, 26)
68 = 2 x 26 + 16	d= gcd(26, 16)
26 = 1 x 16 + 10	d= gcd(16, 10)
16 = 1 x 10 + 6	d= gcd(10, 6)
10 = 1 x 6 + 4	d= gcd(6, 4)
6 = 1 x 4 + 2	d= gcd(4, 2)
4 = 2 x 2 + 0	d= 2

Result: *gcd (1970, 1066) =2*

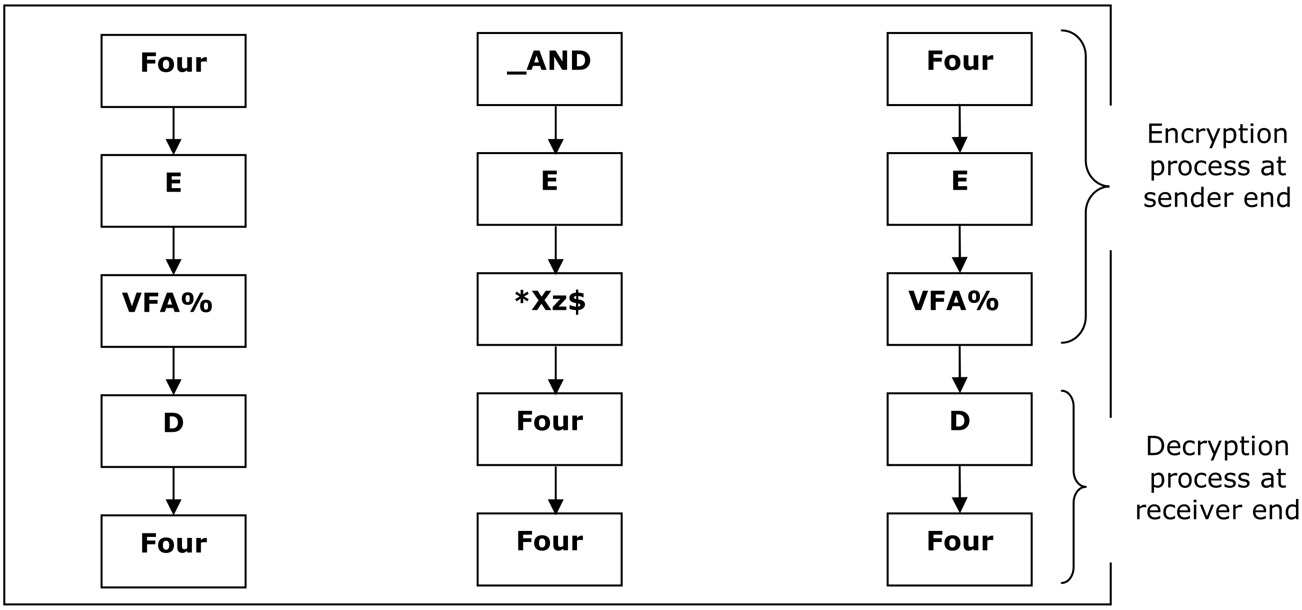
Design Issues of Block Cipher:

1. **S-Box(Substitution box or confusion box) , T-Box(Transposition box or diffusion box) :**
    - ✓ Output bits produced by these boxes should not be closed to a linear function of input bits.
    - ✓ Input that differ in one bit should generate output that differ in many bits
    - ✓ Each row of the S-Box should be a permutation of the possible input / output values
    - ✓ Output bits of S-Box should be distributed such that they affect other S-Boxes in the following round
2. **No. Of Rounds :**
    - ✓ More rounds are generally better, but they cost in reduced performance
    - ✓ Number of rounds should maximize the Avalanche Effect (about 50% of output bits should change for any change in input bit)
    - ✓ Number of rounds should be selected to make the effects of advanced attacks (differential / linear / etc) be similar to exhaustive search (when taking into account the overhead required to run the attacks)
3. **F-Function :**
    - ✓ Must be difficult to unscramble
    - ✓ Should be non-linear
    - ✓ SAC (Strict Avalanche Criteria) – any output bit should be inverted with probability ½ when some input bit is changed
    - ✓ BIC (Bit Independence Criteria) – any two output bits should change independently when some input bit is changed

Block Cipher Principles:

Rather than encrypting one bit at a time a block of bit is encrypted at one go

E.g. we have to encrypt **FOUR _AND FOUR** using block cipher


Four
E
VFA%
D
Four
_AND
E
*Xz$
Four
Four
Four
E
VFA%
D
Four
Encryption process at sender end
Decryption process at receiver end

FOUR would be encrypted first, followed by **_AND** and finally **FOUR**
Thus one block of character gets encrypted at a time

During decryption each block is translated back to the original form
In actual practice communication takes place in bits therefore FOUR actually means binary equivalent of ACII characters later on any algorithm encrypts/decrypts resultant bits are converted in to ASCII equivalent then in their original format

Major problem with block cipher is repeating blocks for which same cipher block is generated which gives cryptanalyst clue regarding the original data

Even if the cryptanalyst cannot guess the complete word, consider he makes the changes like debit to credit and credit to debit in the fund transfer message it would create havoc /great problem

To deal with above problem of block cipher, block ciphers are used in chaining mode

Fiestel Cipher:

- ✓ Fiestel was one of the designer of early cryptographic algorithms at IBM in 1970
- ✓ Fiestel cipher is scheme or template for specifying the algorithms of block cipher
- ✓ Fiestel scheme allows encryption and decryption with the same hardware ckt. or piece of software
- ✓ Fiestel scheme is used by algorithms like DES, IDEA, RC5

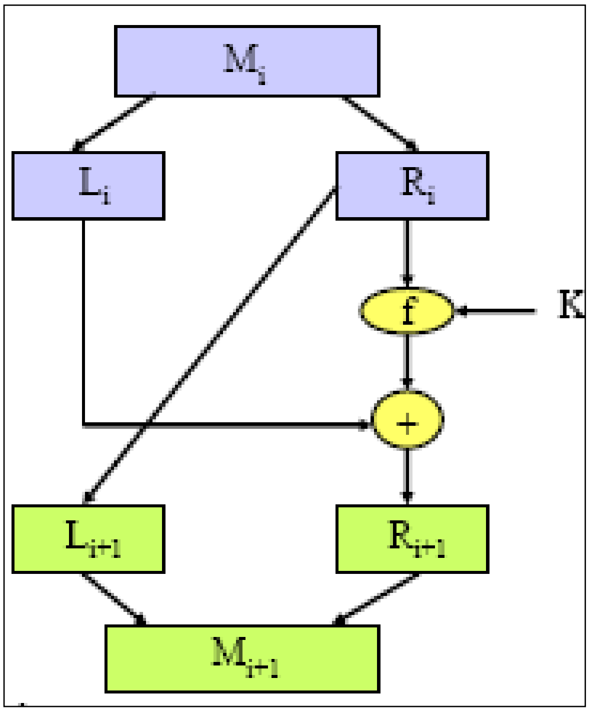
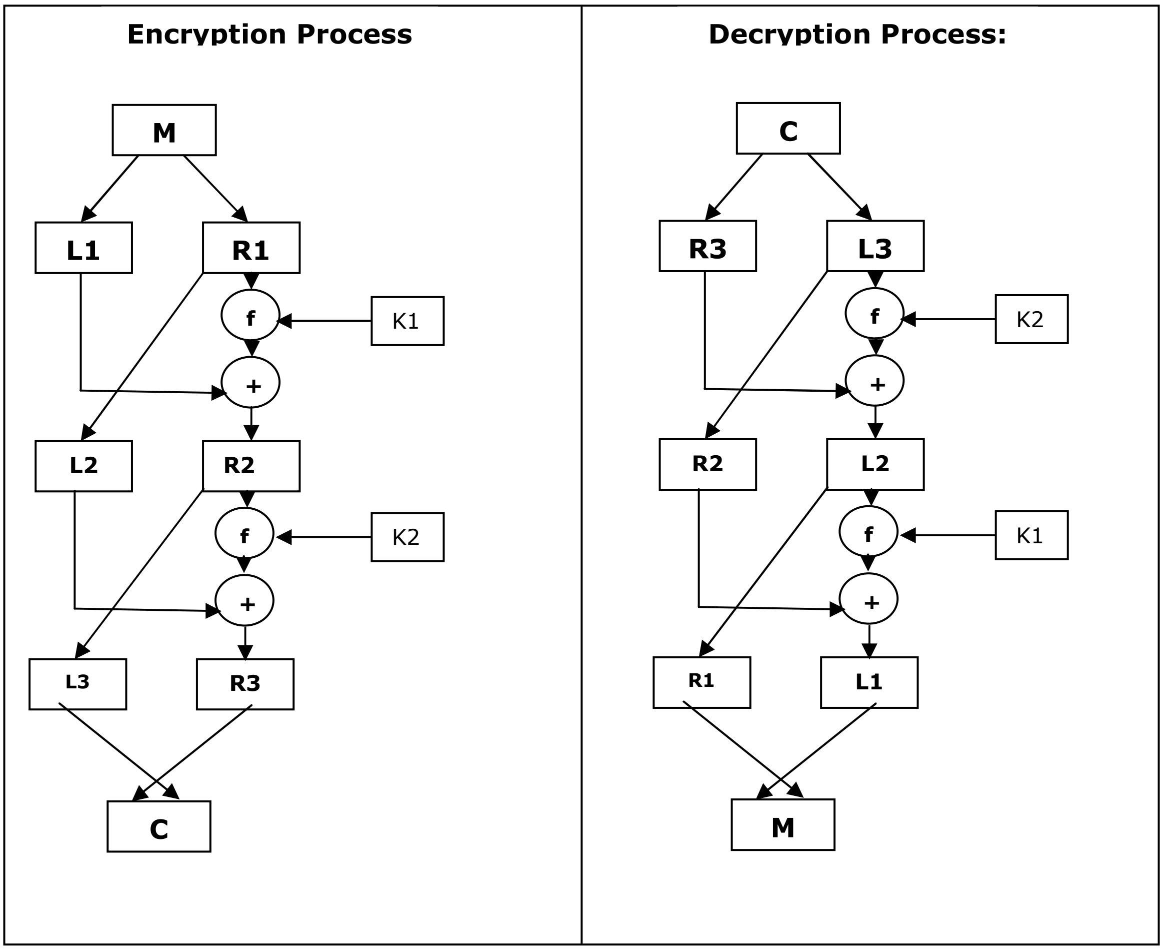


Figure: Shows the Fiestel Structure

- ✓ There is r –rounds in Fiestel cipher
- ✓ In round i input block Mi is broken in two half blocks Li and Ri
- ✓ Input half block Ri is copied to output half block Li+1(to be used as input in round i+1)

- ✓ Input half block Ri and round key Ki are scrambled by function f
- ✓ The scrambled result is XORed with input half block Li to create Output half block (Ri+1) that can be used as input in round i+1



- ✓ In Fiestel cipher decryption is same as encryption process simply we have to reverse the order of round keys
- ✓ Function f can be any function usually it is function that is easy to compute but hard to reverse
- ✓ Function f only serves to generate a pad to be XORed with the left half block function f creates this pad from R half block and from round key K

Fiestel Cipher scheme doesn’t specify the following

- ✓ Block Size
- ✓ Key Size
- ✓ No. of Rounds
- ✓ Round key generation algorithms
- ✓ Scrambling Function

DES (Data Encryption Standard):

DES History

- ✓ In 1973, NBS (National Bureau of Standards) came out with an RFP (Request for Proposals) for a commercial encryption standard

- ✓ IBM proposed its strong Lucifer algorithm (developed by Fiestel and others)

- ✓ NSA (National Security Agency) requested to weaken the strength of Lucifer (by shortening the key)

- ✓ NSA also made changes to IBM’s Lucifer algorithm

- ✓ Data Encryption Standard (DES) accepted in 1976

DES Design Criteria

- ✓ NBS had set the following design criteria for DES:

- ✓ Algorithm must provide high level of security

- ✓ Algorithm must be completely specified

- ✓ Security of the algorithm must reside in the key

- ✓ Algorithm must be available to all users

- ✓ Algorithm must be adaptable for use in diverse applications

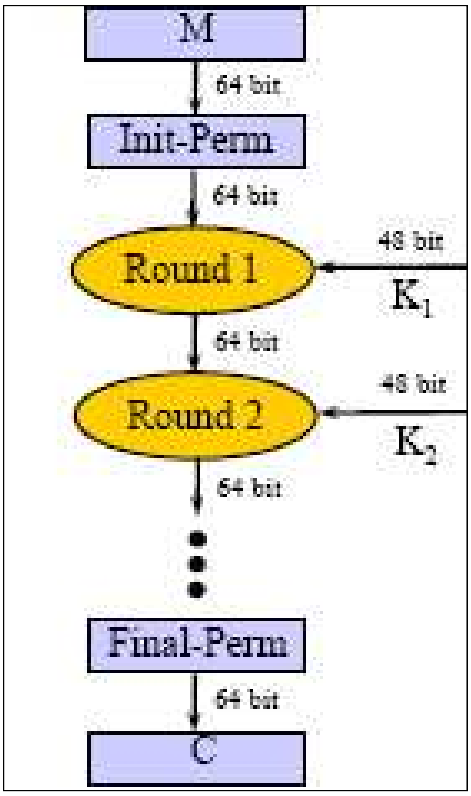
- ✓ Algorithm must be efficiently implemented in hardware

- ✓ Algorithm must be efficient to use
- ✓ Algorithm must be able to be validated
- ✓ Algorithm must be exportable

DES Structure

- ✓ Block size – 64 bits
- ✓ Key size – 56 bits (in a 64-bit buffer)
- ✓ Fixed initial permutation on input block (64 bits)
- ✓ 16 round keys (48 bits) derived from key (56 bits)
- ✓ Key scheduling scheme for 16 round keys
- ✓ 16 iterations each consisting of scrambling the round-block (64 bits) with the round-key (48 bits)
- ✓ Scrambling function detailed later
- ✓ Fixed inverse initial permutation on output block

Overall structure of DES is as follows :



Initial Permutation (64 inputs / 64 outputs):

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

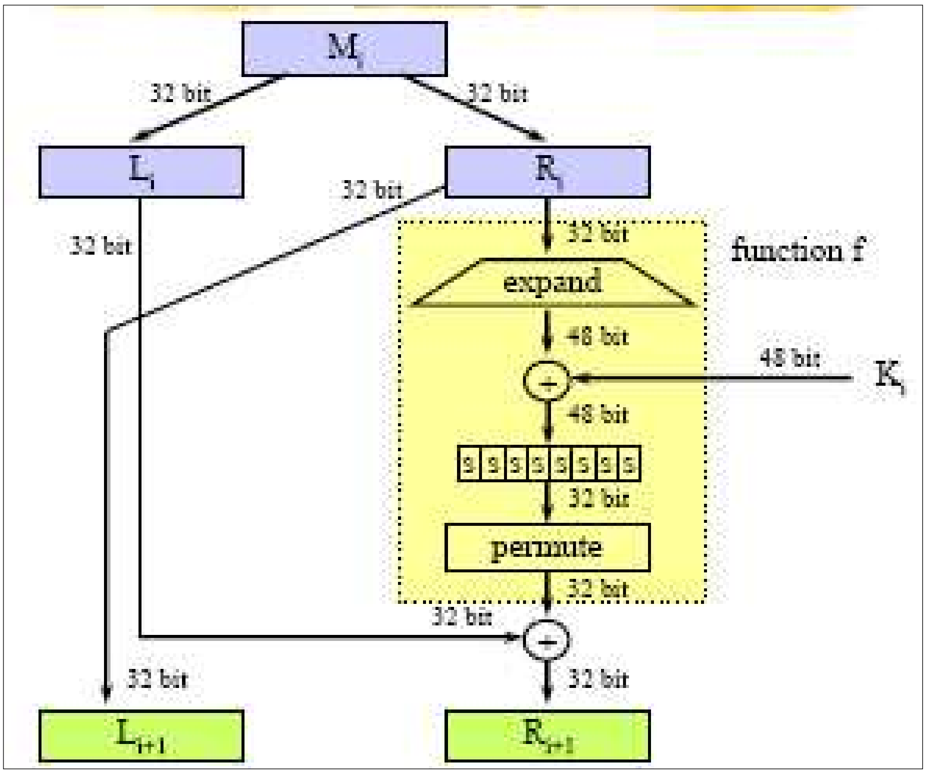
Initial and Final Permutations:

- ✓ "Final Permutation" is inverse of "Initial Permutation"
- ✓ "Initial Permutation" and "Final Permutation" are fully specified and do not add to the security of DES
- ✓ Purpose of "Initial Permutation" and "Final Permutation" is to make software implementations of DES slow

Question: Why not use only "Initial Permutation"?

Answer: To support encryption/decryption with the Feistel Scheme!

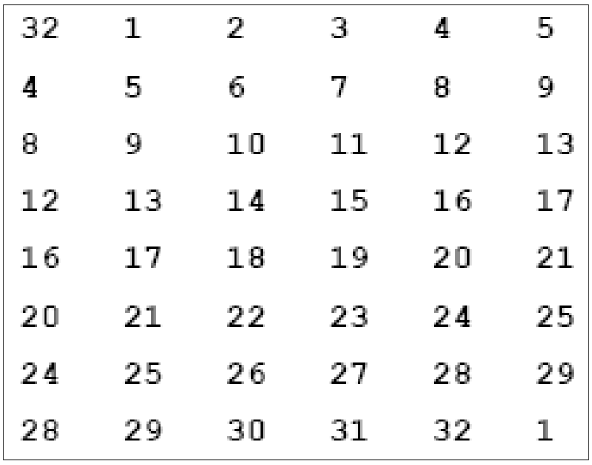
Each round of DES consists of the following operations



- ✓ Message block Mi (64 bits) is split into left half-block Li (32 bits) and right half block Ri (32 bits)
- ✓ Right half block Ri is copied to become left half-block Li+1

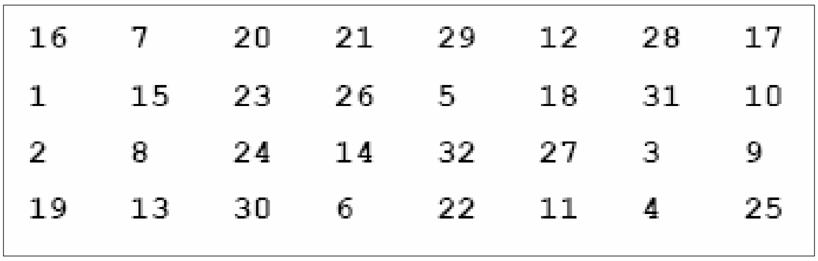
- ✓ Right half block Ri is expanded to 48 bits and is XORed with round key Ki (48 bits)
- ✓ The eight S-Boxes – each takes 6 bits (of the 48 bits above) and generates 4 bits (resulting in 32 bits)
- ✓ The resulting 32 bits are permuted and XORed with the left half-block Li to create right half-block Ri+1

Expand Function (32 inputs / 48 outputs):



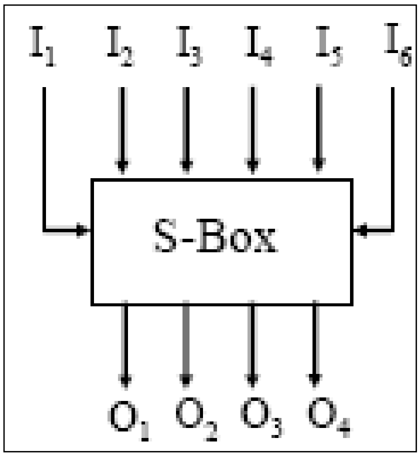
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Internal Permute (32 inputs / 32 outputs):

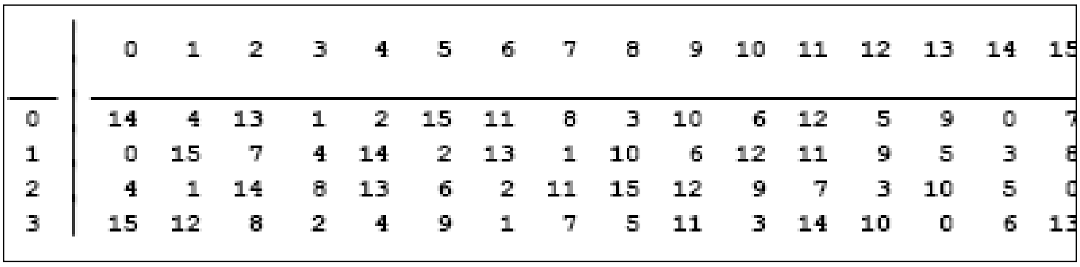


16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

S-Box-1 (6 inputs / 4 outputs):



- ✓ First and sixth input bits select row (between 0 and 3 in table below)
- ✓ Four middle input bits select column (between 0 and 15 in table below)
- ✓ Value of four output bits is depicted in decimal (between 0 and 15) in table entry below



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

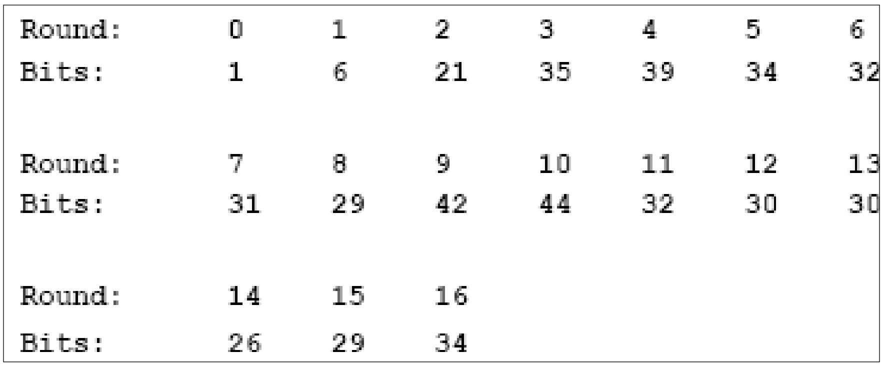
DES Design Issues

- ✓ Unofficial requirement to make DES slow in software
- ✓ The NSA reduction of the key-size to 56 bits
- ✓ NSA changes to the S-Boxes
- ✓ “Initial Permutation” and “Inverse Initial Permutation”

- ✓ Effects of the “Expand” and “Permute” operations
- ✓ Effects of the “f” scrambling function
- ✓ Effects of the S-Boxes
- ✓ Exportability of cryptographic algorithms, software, and hardware

Avalanche Effect in DES – Change in Plaintext:

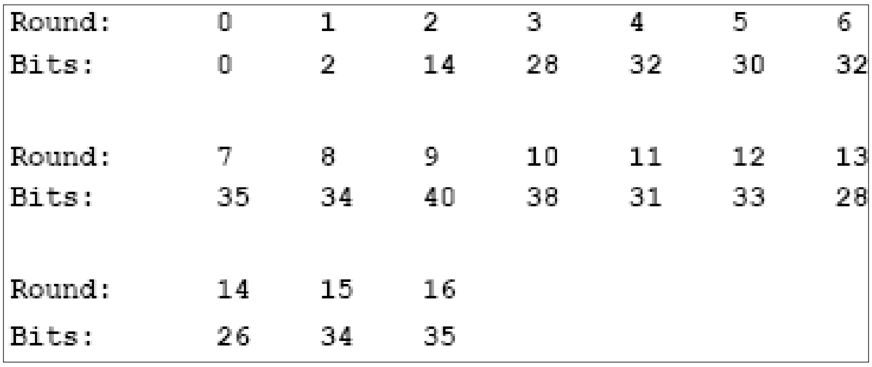
- ✓ Number of output bits that change when one input bit is changed



Round:	0	1	2	3	4	5	6
Bits:	1	6	21	35	39	34	32
Round:	7	8	9	10	11	12	13
Bits:	31	29	42	44	32	30	30
Round:	14	15	16				
Bits:	26	29	34				

Avalanche Effect in DES – Change in Key:

- ✓ Number of output bits that change when one key bit is changed



Round:	0	1	2	3	4	5	6
Bits:	0	2	14	28	32	30	32
Round:	7	8	9	10	11	12	13
Bits:	35	34	40	38	31	33	28
Round:	14	15	16				
Bits:	26	34	35				

Avalanche Effect in DES and Number of Rounds:

- ✓ For output to appear random - number of bits that change should be around 50% (that is – 32 bits)
- ✓ With 16 DES rounds – the Avalanche Effect DES is about optimal
- ✓ Also 16 rounds is large enough to withstand certain cryptanalytical attacks

Weak DES Keys:

- ✓ 4 keys in which each half of the key (after PC-1) is either all 0’s or all 1’s
- ✓ For these keys: EK(EK(X)) = X Semi-Weak DES Keys:
- ✓ 12 keys in which each half of the key (after PC-1) is one of the following: all 0’s, all 1’s, alternating 0’s and 1’s, and alternating 1’s and 0’s
- ✓ For pairs of keys: EK1(EK2(X)) = X

DES strength:

- ✓ Since 1975, there was a debate regarding the selection of only 56 bits for the DES key size
- ✓ Exhaustive Search Attack: Requires searching O(256) keys
- ✓ Differential Cryptanalysis: Requires analyzing O(247) chosen plaintexts
- ✓ Linear Cryptanalysis: Requires analyzing O(247) known plaintexts
- ✓ In 1990’s – DES was declared not secure enough by the technical community (IETF)

Exhaustive Search Attack:

- ✓ Search space of O(256) = O(1017) keys
- ✓ In the 1970's, Diffie and Hellman suggested a $20M-machine that will crack DES in about one day
- ✓ In the 1990's, Wiener suggested a $1M-machine that will crack DES in 3.5 hours
- ✓ Assume about 109 encryptions per second on today's computers. Then about 108 computers seconds are required to crack DES
- ✓ In 1990's, DES challenges were broken in matter of days using distributed clusters of computers
- ✓ Presumably, most national security agencies have the hardware and software to crack DES in hours

Differential Cryptanalysis Attack:

- ✓ Study the differences between two encryptions of two different plaintext blocks M and M*
- ✓ Study the probability of output differences in each S-Box
- ✓ Trace back differences to specific S-Boxes
- ✓ Estimate the likelihood of key-bits involved in the XOR operation before the S-Boxes
- ✓ Continue developing estimates for key until one key emerges as the only ultimate option
- ✓ Chosen space of O(247) plaintexts
- ✓ Not practical – but theoretically important

Linear Cryptanalysis Attack:

- ✓ Approximate the DES key as a linear transformation of the plaintext bits and the ciphertext bits
- ✓ Change the coefficients based on multiple values of pairs of <plaintext,ciphertext>
- ✓ Requires known space of O(247) <plaintext,ciphertext> pairs
- ✓ Not practical – but theoretically important

DES Strength – Summary

- ✓ Since early 1990's – DES is considered not secure enough for technical and commercial use
- ✓ Several approaches:
- ✓ Strengthening DES – 2-DES
- ✓ Strengthening DES – 3-DES
- ✓ Strengthening DES – DES-X
- ✓ Other Algorithms

DES Variants:

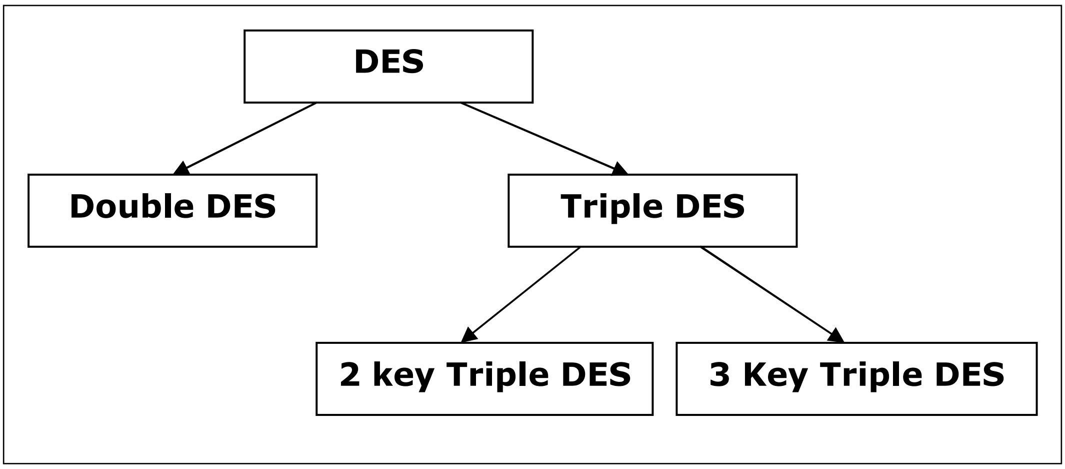


Figure: Shows the variants of DES

Double DES:

- ✓ Apply two iterations of DES with two keys K1 and K2

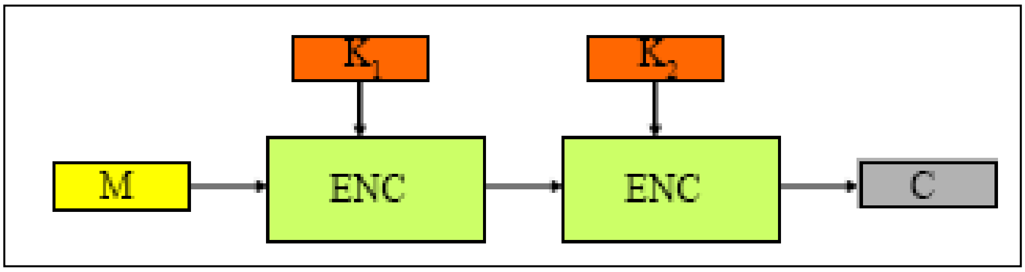


Figure: Double DES Encryption process

C=Ek1 (Ek2 (M))

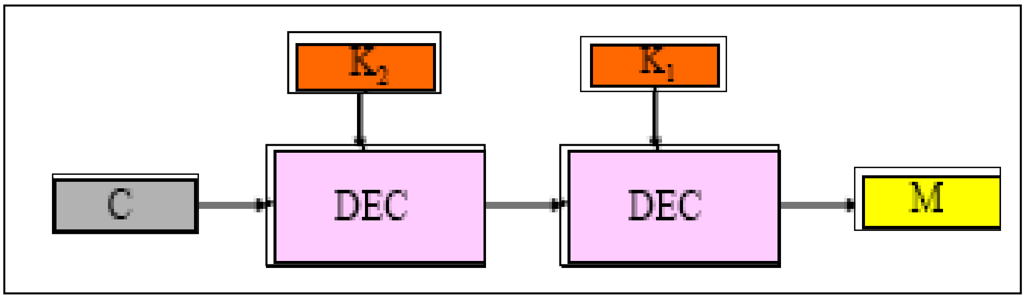


Figure: Double DES Decryption process

M=DK2(Dk1(C))

Where

- ✓ M- Plain text block
- ✓ C- Cipher Text
- ✓ ENC- Encryption Process
- ✓ DEC- Decryption Process

2-key Triple-DES

DES Encrypt-Decrypt-Encrypt with two keys K1, and K2

Properties:

- ✓ Two keys (112 bits)
- ✓ Strength about O(2110) against Meet-in-the-Middle
- ✓ Compatible with regular DES when K1= K2

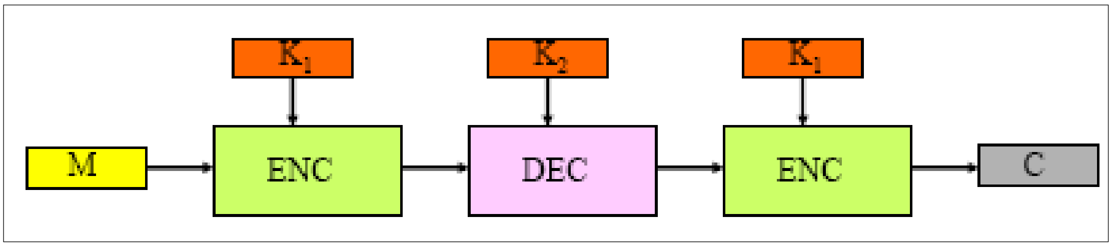


Figure: 2-key Triple DES Encryption process

C=Ek1 (Dk2 (EK1 (M)))

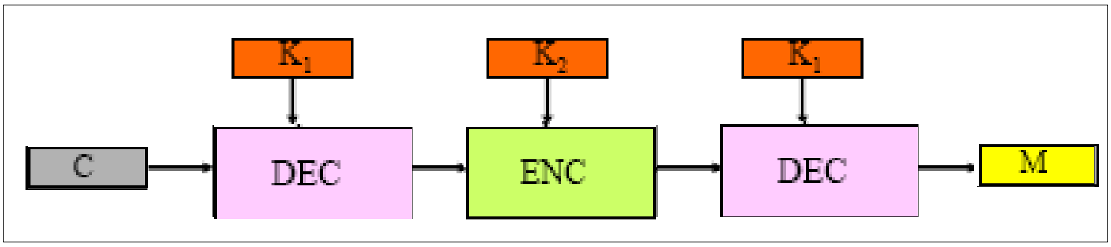


Figure: 2-key Triple DES Decryption process

M=DK1(Ek2(DK1(C)))

3-KEY TRIPLE-DES

EEE Mode:

DES Encrypt-Encrypt-Encrypt with three keys K1, K2, and K3

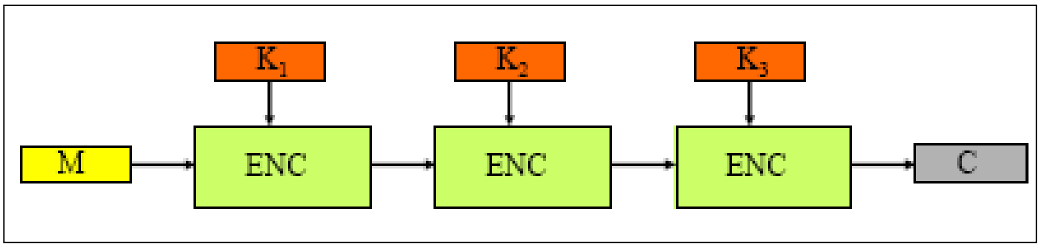


Figure: 3-key Triple DES Encryption process

C=Ek1 (Ek2 (EK3 (M)))

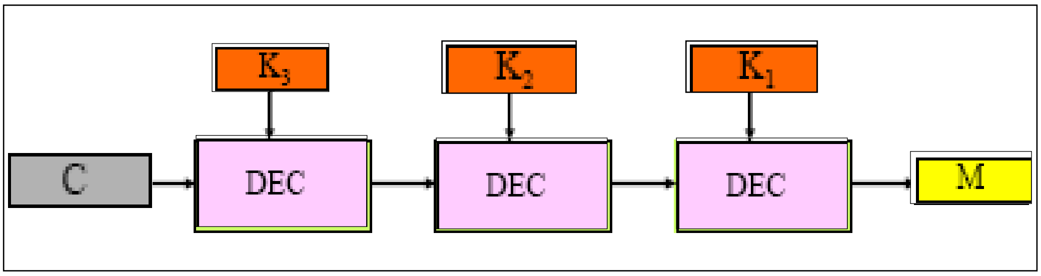
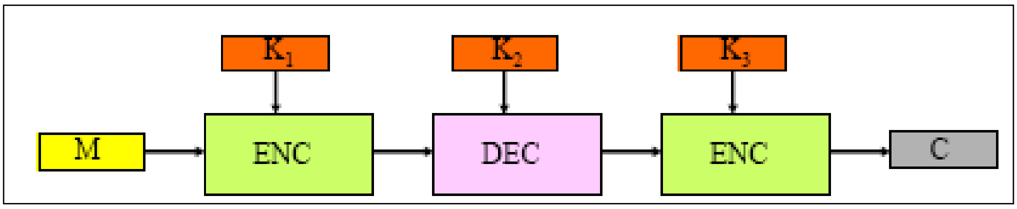


Figure: 3-key Triple DES Decryption process

M=DK3(Dk2(DK1(C)))

EDE Mode:

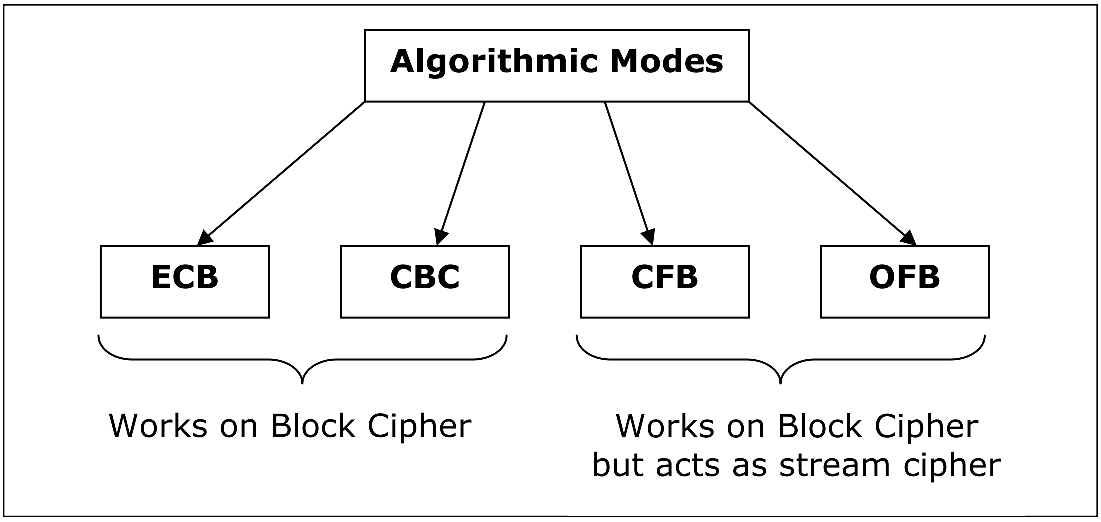


C=Ek1 (Dk2 (EK3 (M))) and **M=DK3 (Dk2 (DK1(C)))**

Algorithmic Modes (Modes of operation):

An algorithmic mode is combination of series of algorithmic steps on a block cipher and some form of feedback from the previous steps

There are four algorithmic modes namely
1. Electronic Code Book (ECB)
2. Cipher block Chaining (CBC)
3. Cipher Feedback(CFB)
4. Output Feedback(OFB)



Electronic Code Book (ECB):

- ✓ ECB is simplest mode of operation
- ✓ Incoming plaintext message is divided in blocks of 64 or 32 bit
- ✓ All the plaintext blocks then encrypted independently
- ✓ For encrypting all the block of message same key and algorithm is used

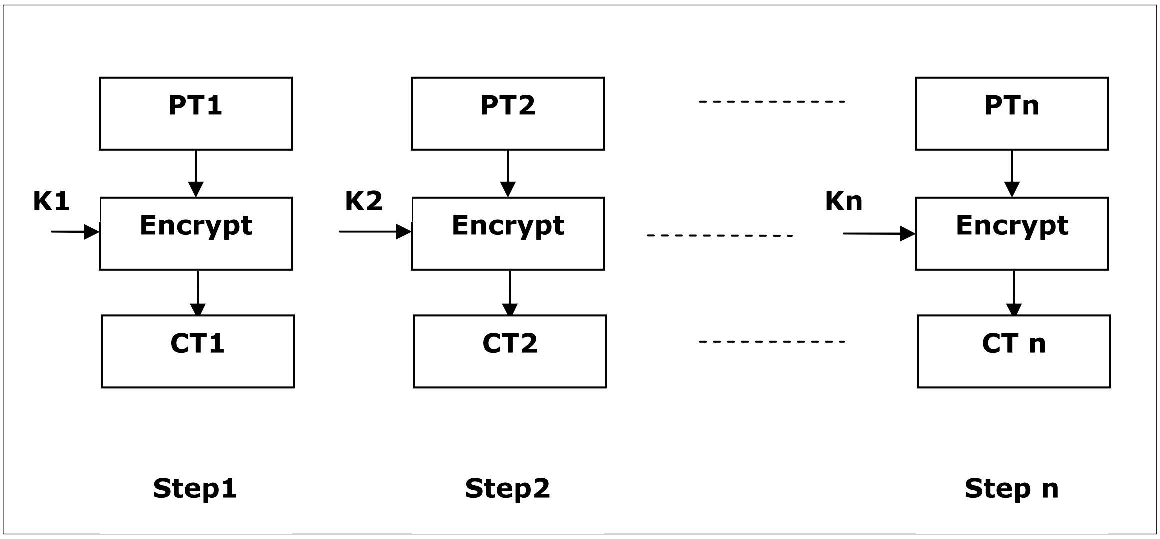


Fig: ECB Mode – Encryption Process

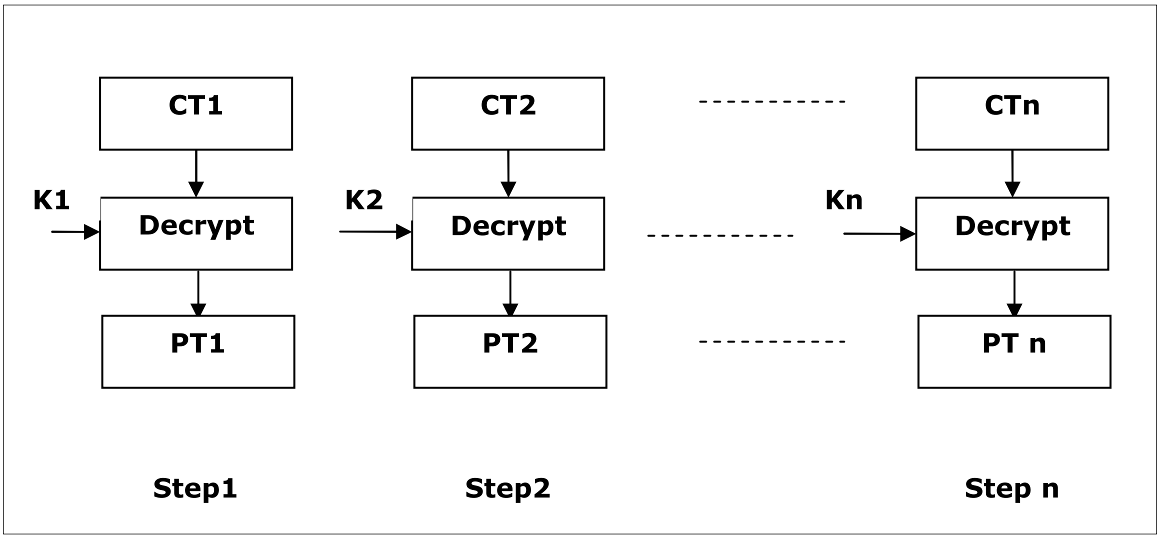


Fig: ECB Mode –Decryption Process

At receivers end incoming data is divided into 64 or 32-bit blocks and by using same key as was used for encryption each block is decrypted to produce corresponding PT block

In ECB single key is used for encryption so if input message block repeats then output CT block also repeated that’s why ECB is not suitable for encrypting secure data

Cipher Block Chaining (CBC):

- ✓ CBC ensures even if PT blocks repeats in input there would not be two identical blocks in CT they would differ from each other
- ✓ CBC adds feedback mechanism to block ciphers.
- ✓ Here result of encryption of previous block is fed back as input for the current block i.e. each block is used to modify the encryption of the next block

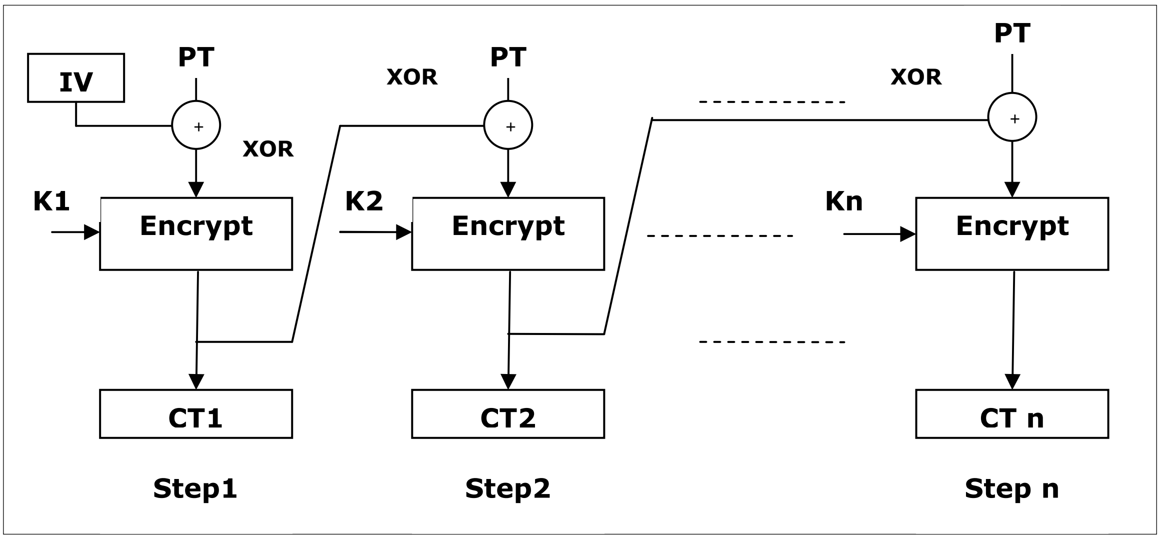


Fig. : CBC Mode Encryption Process

- ✓ As shown in figure first step receives two inputs i.e. first PT block and random vector IV (Initialization Vector)
- ✓ IV has no special meaning it simply used to make each message unique since its value is randomly generated
- ✓ First block of CT and IV are combined using XOR and then encrypted using a key to produce the first cipher text block CT1 and this block is provided as input to next plaintext block
- ✓ Now PT2 or 2nd PT block is XORed with output of step1 it then encrypted with the same key as used in step1 and produced CT is passed to next step …..

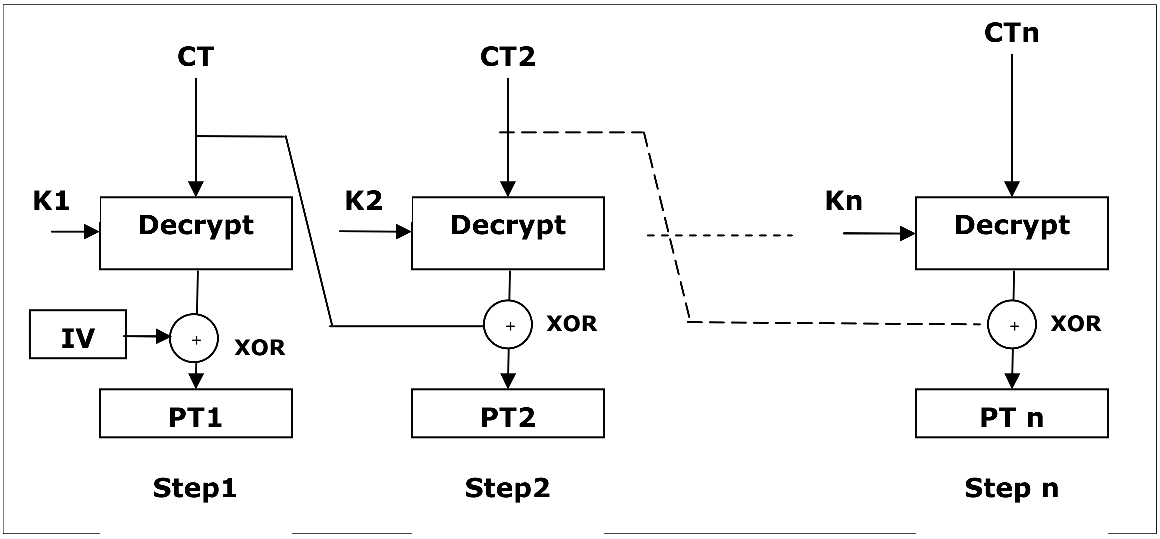


Fig: CBC Mode Decryption Process

- ✓ As shown in figure while decryption CT block1 is passed through the decryption algorithm using the same key which was used during encryption for all the blocks?
- ✓ Output of the above step is XORed with IV and it produces PT block
- ✓ In step2 CT block 2 is decrypted and XORed with CT block1to produce PT block 2 PT2

CFB (Cipher feedback):

All applications cannot work on block of data. Security is also required in applications that are character oriented

e.g. a operator is typing keystroke at terminal which need to be immediately transmitted across communication link in secure manner and CFB is useful in such cases . In this mode data is encrypted in units that are smaller

Let's understand CFB mode by assuming we are dealing with j bits at a time (as we have usually j=8 not always)

CFB is slightly complicated we see the step by step details

Step 1: Like CBC a 64 –bit initialization vector is used in CFB mode

- ✓ IV is kept in shift register
- ✓ It is encrypted in first step to produce 64 bit IV-Cipher

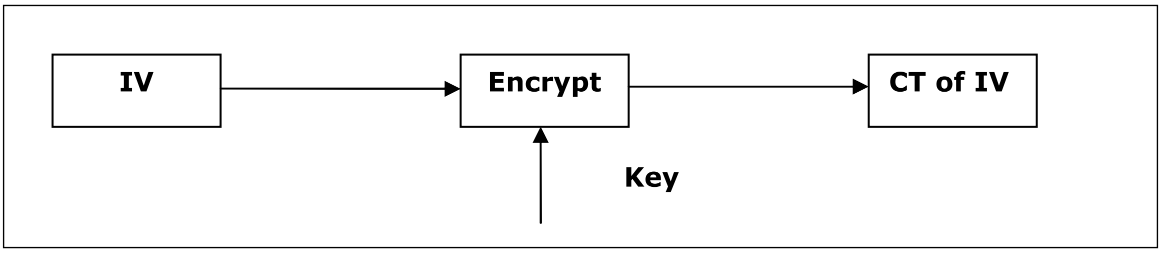


Fig: CFB Step1

Step 2:

- ✓ Now MSB (i.e. leftmost bit) j-bit of encrypted
- ✓ IV are XORed with first j-bits of plaintext
- ✓ This produces first portion of ciphertext say (c)
- ✓ C is transmitted to receiver

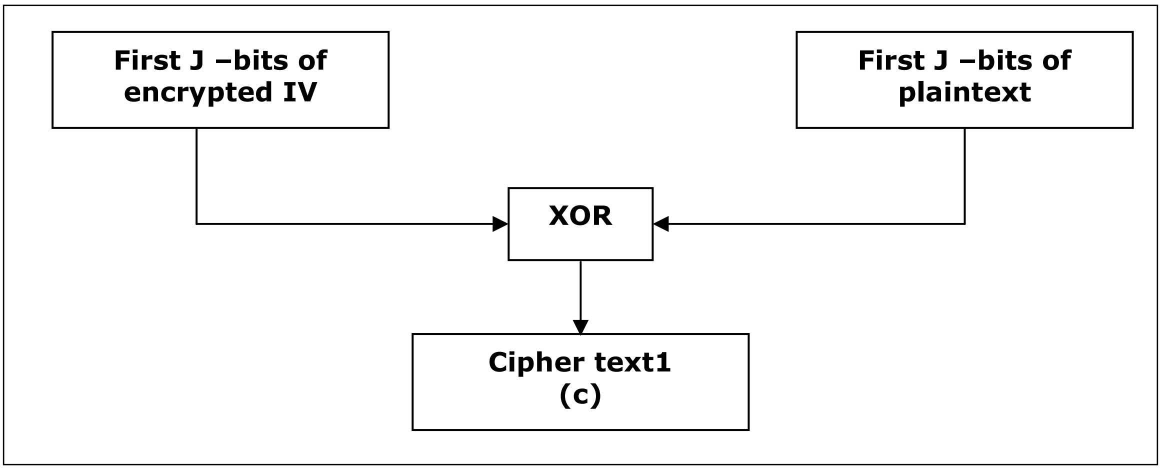


Fig: CFB Step2

Step 3:

- ✓ In this step IV bits (Contents of shift register containing IV) are shifted left by j-positions
- ✓ Thus the rightmost j positions of the shift register now contain unpredictable data
- ✓ These rightmost j-positions are now filled with C obtained in previous step

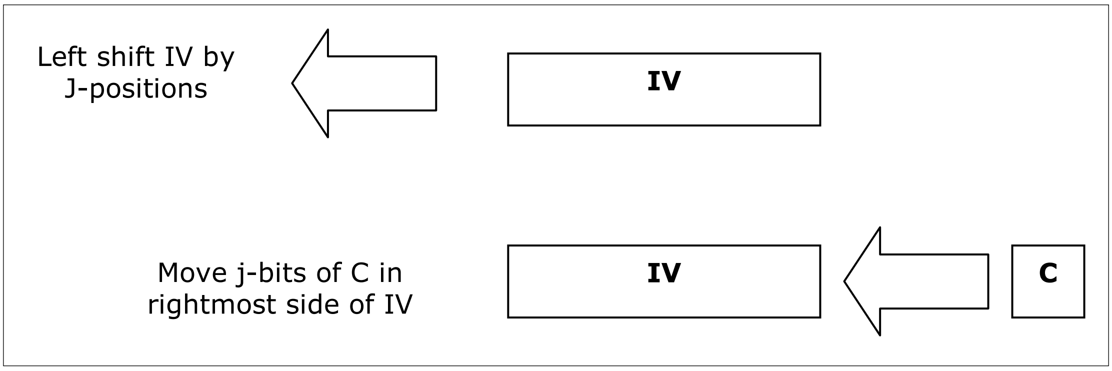

Left shift IV by J-positions
IV
Move j-bits of C in rightmost side of IV
IV
C

Fig: CFB Step2

Step 4:

- ✓ Now step1 to step 3 continues until all the plain text block are encrypted i.e. following steps repeats
- ✓ IV is encrypted
- ✓ Leftmost j-bits of encrypted IV are XORed with next j-bits of Pt
- ✓ Resulting j-bits of CT is send to receiver
- ✓ The shift register containing IV is left shifted by j-bits
- ✓ J- bits of CT are inserted from right into shift register containing IV

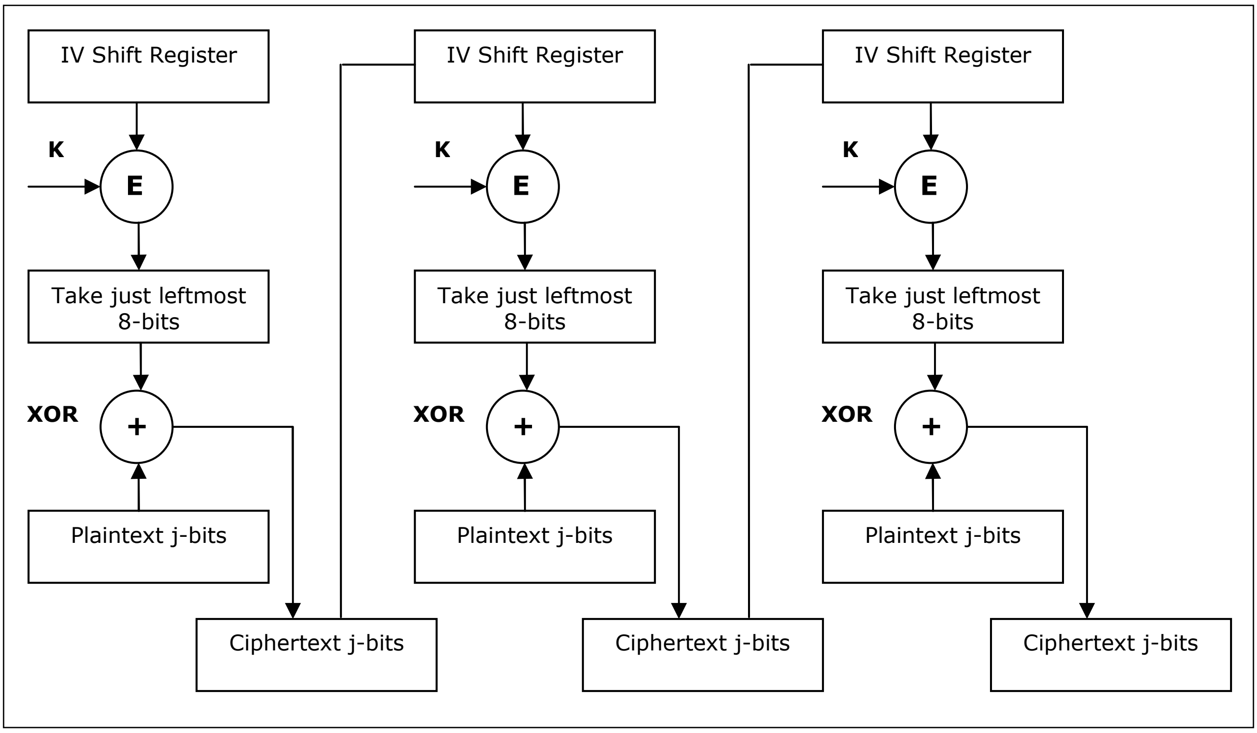


Fig: CFB – Overall Encryption Process

Output feedback Mode (OFB):

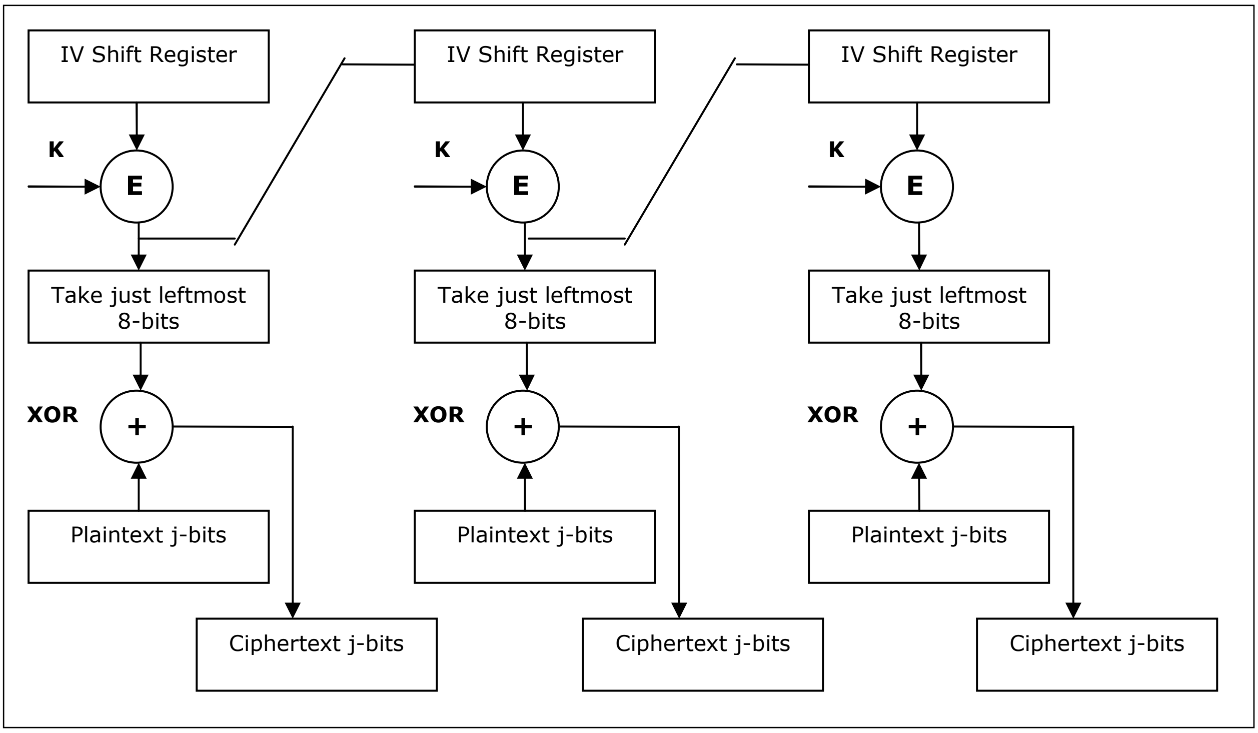


Fig: OFB – Overall Encryption Process

- ✓ OFB is extremely similar to CFB
- ✓ In OFB output of IV encryption process is fed as input to the next stage encryption process

Question:

1. What are essential ingredients of a symmetric cipher?

2. What are two basic functions used in encryption algorithm

3. How many keys are used by two peoples to communicate via cipher

4. What difference is between block and stream cipher

5. Write down two general approaches of attacking ciphers

6. Define and explain Caesar cipher, Playfair cipher, Monoalphabetic cipher

7. Write a note on transposition cipher & Steganography

8. Write down the difference between diffusion and confusion

9. Why it’s important to study Fiestel cipher

10. Explain different algorithmic modes

11. Working of DES and its variants along with advantages

12. Explain Euclid’s algorithm with suitable example for generating GCD

13 Are all block Ciphers polyalphabetic? Explain

14 Alice can use only additive cipher (Caesar) on her computer to Send a message to a friend she thinks that the message is More secure if she encrypts the message two times each time, Each time with different key. Is she right? Defend your answer.

15. Define Key,Code,Encryption,Cryptanalysis,Symmetric key cipher

16. List and Explain different kinds of cryptanalysis attacks

17. Define greatest common divisor of two integers. Which algorithm can effectively find the greatest common divisor

18. Explain why modern block ciphers are designed as substitution ciphers instead of transposition ciphers.

19. Need of cryptography and its applications

20. Encrypt “ MEET ME AFTER TOGA PARTY “ by using caesar cipher, Rail fence cipher , Playfair cipher and row column method