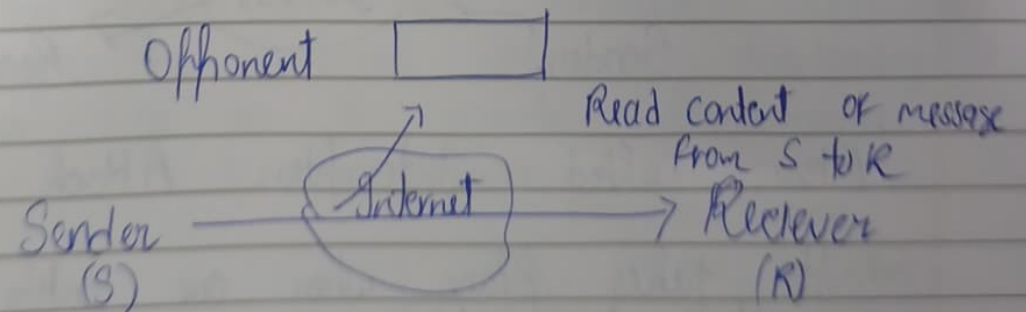
Ch-1 Symmetric Cipher Model

Q.1 List and explain various types of attacks on encrypted messages [It is Q-4]
Q. List & explain various types of attacks

Ans Two types of attack :-
1) Passive Attack
2) Active Attack

1) Passive Attack
→ A Passive attack attempts to learn or make use of information from the system but does not affect the system resources
→ Attacker aims to obtain info that is in transit
→ No intention to perform any modifications to the data
Two types:-
1) Release of Message Contents
2) Traffic Analysis

1) Release of message contents
→ A telephone conversation, an electronic mail message, and transferred file may contain sensitive or confidential information



2) Traffic Analysis
→ Mask the contents of message so that opponents could not extract the information from the message

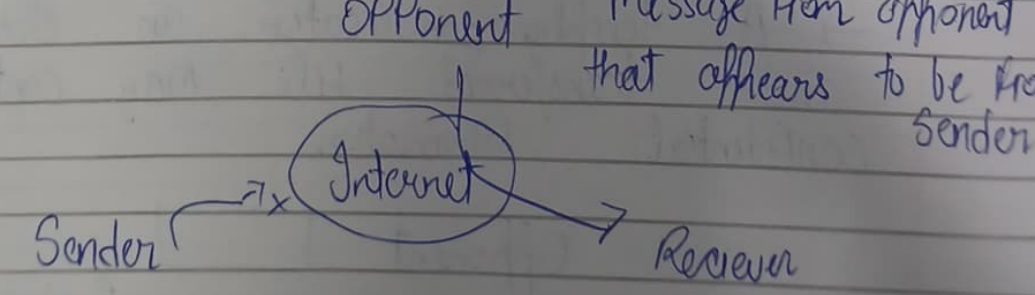
→ Passive attacks are very difficult to detect because they do involve any alteration of data
→ It is feasible to prevent the sucess of attack usually by means of encryption

2) Active Attack

→ Involves some modification of data stream or the creation of a false stream. These attacks can not be prevented easily

Types
(1) Masquerade
(2) Replay
(3) Modification of Message
(4) Denial of Service

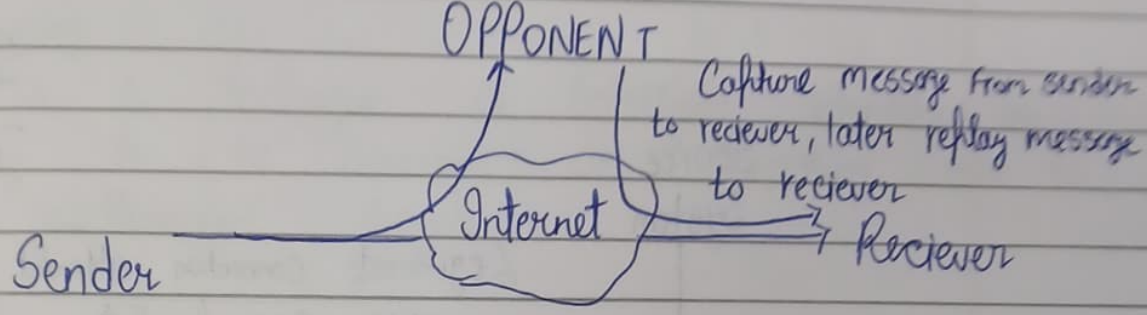
(1) Masquerade
→ Impersonating hota hai



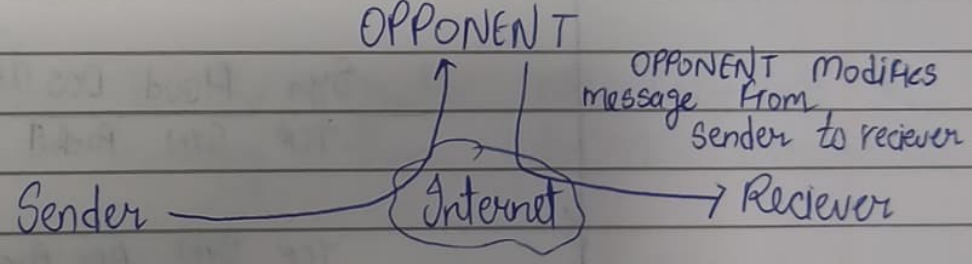
Also called Interuption Attack

→ It takes place when one entity pretends to be a different entity

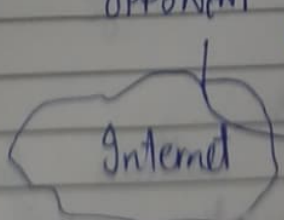
2) Replay
→ It involves passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect



3) Modification of message
→ It involves some change to og message.
It produces an unauthorized effect

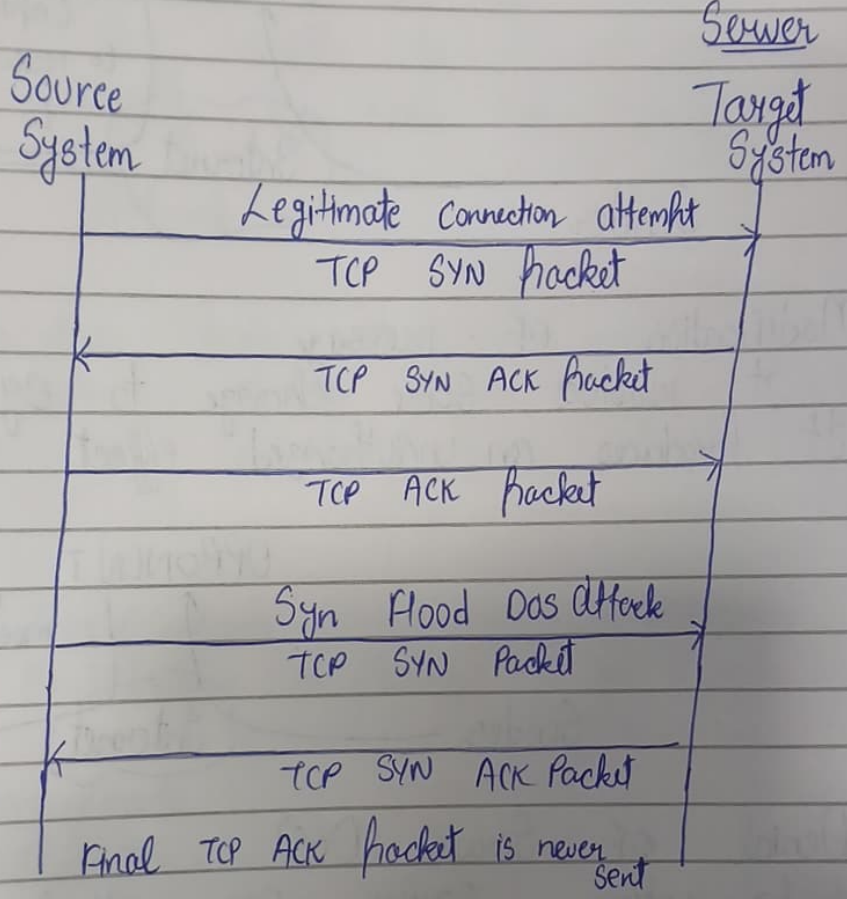


4) Denial of Service (DoS)
→ Fabrication causes DoS attacks
→ DoS prevents the normal use or management of communication facilities
→ Another form of service denial is the disruption of an entire network, either by disabling the network or by overloading it with messages so as to degrade performance



7 The first type of DOS attacks were single source attacks

SYN Flood is most widely used DOS attack



Q-2 Define them

Ans (i) Cryptography:-
It is an art or science encompassing the principles and methods of transforming a plaintext message into one that is unintelligible, and then that message back to its orignal form

(ii) Cryptanalysis
The study of methods for obtaining the meaning of encrypted information without accesing the secret information

(iii) Sniffing
→ It refers to monitoring and capturing network traffic (data packets) using hardware or software tools

(iv) Spoofing
→ It is a technique of falsifying identity by pretending to be someone or something else to gain unauthorized access

(v) Interception
→ It occurs when an unauthorized entity gains access to data being transmitted between two legitimate parties

(vi) Fabrication
→ It refers to the creation of false data or messages and inserting them into a communication system

(vii) Masquerade
→ It is a type of attack where an unauthorized entity pretends to be an authorized user to gain access to a system

Q.3 What are the key principles of security?

Ans (1) Confidentiality
(2) Integrity
(3) Availability

🔑 1. Confidentiality

Ensures that **information is accessible only to authorized users** and not disclosed to others.

- **Goal:** Prevent unauthorized access or exposure.
- **Techniques:** Encryption, access control, authentication.
- **Example:** Password protection, SSL/TLS encryption.

🔑 2. Integrity

Ensures that **data remains accurate, complete, and unaltered** during storage, transmission, and processing.

- **Goal:** Prevent unauthorized modification or corruption of data.
- **Techniques:** Hash functions, digital signatures, checksums.
- **Example:** Using SHA-256 to verify data integrity.

🔑 3. Availability

Ensures that **authorized users have reliable and timely access** to information and resources when needed.

- **Goal:** Keep systems operational and accessible.
- **Techniques:** Backup systems, redundancy, DDoS protection.
- **Example:** Using load balancers and failover servers.

1. Confidentiality

- Confidentiality refers to limiting information access and disclosure to authorized users and preventing access by or disclosure to unauthorized ones.
- Sensitive information should be kept secret from individuals who are not authorized to see the information.
- Underpinning the goal of confidentiality are authentication methods like user-IDs and passwords that uniquely identify a data system's users, and supporting control methods that limit each identified user's access to the data system's resources.
- Confidentiality is not only applied to storage of data but also applies to the transmission of information.
- Confidentiality means that people cannot read sensitive information, either while it is on a computer or while it is traveling across a network.

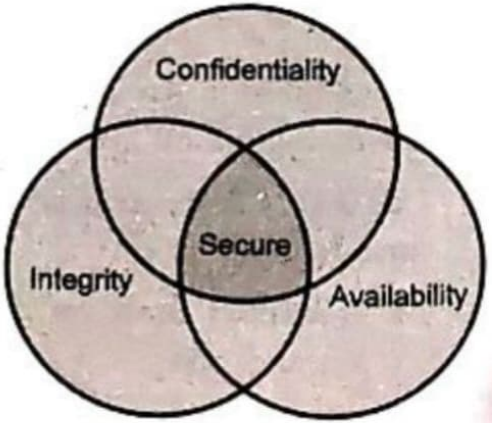


Fig. 1.1.1 Relationship between confidentiality integrity and availability

2. **Integrity**

- Integrity refers to the trustworthiness of information resources.
- Integrity should not be altered without detection.
- It includes the concept of "data integrity" namely, that data have not been changed inappropriately, whether by accident or deliberately malign activity.
- It also includes "origin" or "source integrity" that is, that the data actually came from the person or entity you think it did, rather than an imposter.
- Integrity ensures that information is not changed or altered in transit. Under certain attack models, an adversary may not have to power to impersonate an authenticated party or understand a confidential communication, but may have the ability to change the information being transmitted.
- On a more restrictive view, however, integrity of an information system includes only preservation without corruption of whatever was transmitted or entered into the system, right or wrong.

3. **Availability**

- Availability refers, to the availability of information resources. An information system that is not available when you need it is at least as bad as none at all.
- Availability means that people who are authorized to use information are not prevented from doing so. It may be much worse, depending on how reliant the organization has become on a functioning computer and communications infrastructure.
- Almost all modern organizations are highly dependent on functioning information systems. Many literally could not operate without them.
- Availability, like other aspects of security, may be affected by purely technical issues (e.g. a malfunctioning part of a computer or communications device), natural phenomena (e.g. wind or water), or human causes (accidental or deliberate).

Q-4 What is Cryptanalysis ? Explain 5 techniques or how attackers can launch attacks

Ans Cryptanalysis is the process of trying to break any cipher text message to obtain the original plain text message itself is called Cryptanalysis

1) Cipher-text only Attack
2) Known - Plaintext Attack
3) Chosen - Plaintext " "
4) Adaptive Chosen ~~Plaintext~~ Cipher attack
5) Chosen text attack

(1) Ciphertext only Attack
→ The Cryptanalyst has ciphertext of several messages, all of which have been encrypted using the same encryption algorithm
→ The analyst may be able to capture one or more plaintext messages as well as their encryptions

Given: $C_1 = E_K(P_1)$, $C_2 = E_K(P_2)$ ___ $C_i = E_K(P_i)$

Deduce: Either P_1, P_2 ___ P_i, K or an algorithm to infer P_{i+1} from $C_{i+1} = E_K(P_{i+1})$

(2) Known - Plaintext Attack
→ Attacker knows some combinations of P_i, C_i and based on these, he try to decrypt the messages
→ Attacker has pairs of plaintext and corresponding ciphertext and uses them to derive the key or a way to decrypt other ciphertext

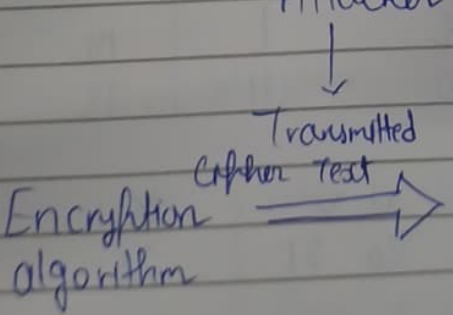
Given: P_1, $C_1 = E_K(P_1)$, P_2, $C_2 = E_K(P_2)$ ... P_i, $C_i = E_K(P_i)$

Deduce: Either K or an algorithm to infer P_{i+1} from $C_{i+1} = E_K(P_{i+1})$

3) Chosen Plaintext attack
→ More powerful than a known plaintext attack because the cryptanalyst can choose specific plaintext blocks to encrypt
→ Attacker can request ciphertexts for plaintexts of their choice and uses the results to break the scheme

Given: P_1, $C_1 = E_K(P_1)$, P_2, $C_2 = E_K(P_2)$ ... P_i, $C_i(E_K(P_i)$
where the cryptanalyst gets to choose $P_1, P_2 \ldots P_i$
Deduce: Either K or an algorithm to infer P_{i+1} from $C_{i+1} = E_K(P_{i+1})$

4) Chosen Ciphertext attack
The attacker can choose a ciphertext and get its decrypted plaintext (eg. by exploiting a decryption service)
→ They then use that information to determine the key or decrypt other ciphertexts



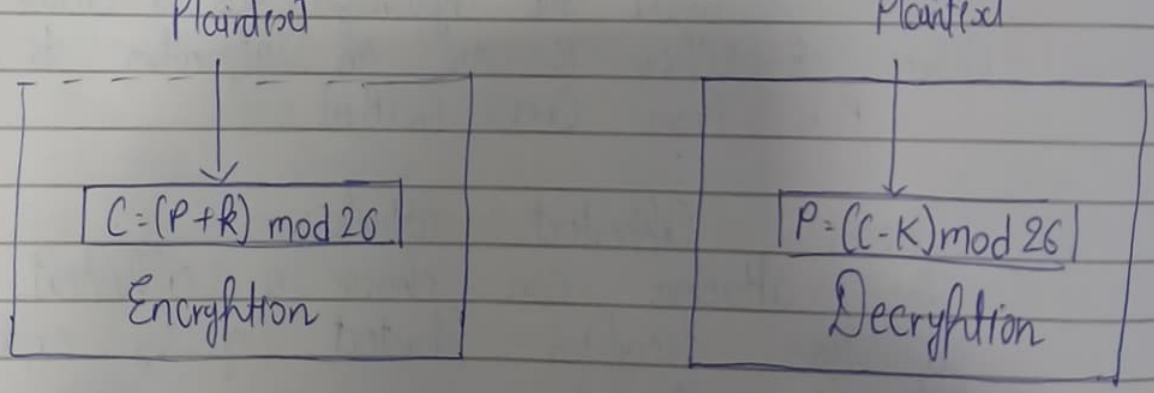
5) Chosen Text Attack

→ Combination of Chosen Plaintext & Chosen Ciphertext

The attacker can both encrypt chosen plaintexts and decrypt chosen ciphertexts

Q-5 Explain Caesar Cipher with example

Ans → It is encryption algorithm technique
→ Key = 3



a	b	c	d	e	f	g	h
0	1	2	3	4	5	6	7

i	j	k	l	m	n	o	p	q
8	9	10	11	12	13	14	15	16

r	s	t	u	v	w	x	y	z
17	18	19	20	21	22	23	24	25

1.5 Cryptanalysis

GTU : May-11, 12, 14, Winter-15

- The process of trying to break any cipher text message to obtain the original plain text message itself is called as cryptanalysis.
- Cryptanalysis is the art of deciphering encrypted communications without knowing the proper keys
- Cryptanalysis is the breaking of codes. The person attempting a cryptanalysis is called as a cryptanalyst.
- Brute force attack : The attacker tries every possible key on a piece of cipher text until an intelligible translation into plaintext is obtained.

- Types of Attacks on Encrypted Messages :

Sr. No.	Types of attack	Known to cryptanalyst
1.	Ciphertext only	1. Encryption algorithm 2. Cipher text
2.	Known plaintext	1. Encryption algorithm 2. Cipher text 3. One or more plaintext ciphertext pairs formed with the secret key.
3.	Chosen plaintext	1. Encryption algorithm 2. Ciphertext 3. Plain text message chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key.
4.	Chosen ciphertext	1. Encryption algorithm 2. Cipher text 3. Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key.
5.	Chosen text	1. Encryption algorithm 2. Cipher text 3. Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key. 4. Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key.

→ In Caesar Cipher, each alphabet in a message is replaced by an alphabet three places down the line

Plain Text	Hello	world
Cipher Text	KHOOR	ZRUOG

Q-25 Explain Hill Cipher with example

Ans $C = KP \mod 26$

Eg: 'exam'

Key $\begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}$

Plaintext (P) = $\begin{bmatrix} e \\ x \end{bmatrix} \begin{bmatrix} a \\ m \end{bmatrix} = \begin{bmatrix} 4 & 0 \\ 23 & 12 \end{bmatrix}$

(K) Key = $\begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}$

$C = KP = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix} \begin{bmatrix} 48 \\ 84 \end{bmatrix}$

$= \begin{bmatrix} 9(41) + 4(84) \\ 5(41) + 7(84) \end{bmatrix}$

$= \begin{bmatrix} 432 + 336 \\ 240 + 588 \end{bmatrix} = \begin{bmatrix} 768 \\ 828 \end{bmatrix}$

$KP \mod 26 = \begin{bmatrix} 128 \\ 181 \end{bmatrix} \mod 26 \quad \begin{bmatrix} 48 \\ 84 \end{bmatrix} \mod 26$

$= \begin{bmatrix} 24 \\ 25 \end{bmatrix} \begin{bmatrix} 22 \\ 6 \end{bmatrix} = \begin{bmatrix} y \\ z \end{bmatrix} \begin{bmatrix} w \\ g \end{bmatrix}$

Ciphertext = "yzwg"

Eg: Plaintext = "hi"
key = "jefh"

Key = $\begin{bmatrix} 9 & 5 \\ 4 & 7 \end{bmatrix} \times \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}$ ✓

$P_2 = \begin{bmatrix} 7 \\ 8 \end{bmatrix}$

$KP_2 = \begin{bmatrix} 9 & 5 \\ 4 & 7 \end{bmatrix} \begin{bmatrix} 7 \\ 8 \end{bmatrix}$

$KP \text{ mod} = \begin{bmatrix} 9\times7 + 5\times8 \\ 4\times7 + 7\times8 \end{bmatrix}$

$= \begin{bmatrix} 9\times7 + 8\times4 \\ 35 + 7\times8 \end{bmatrix} = \begin{bmatrix} 63 + 40 \\ 28 + 56 \end{bmatrix}$

$= \begin{bmatrix} 63 + 32 \\ 35 + 56 \end{bmatrix} = \begin{bmatrix} 103 \\ 84 \end{bmatrix}$

$= \begin{bmatrix} 95 \\ 91 \end{bmatrix} = \begin{bmatrix} 25 \\ 9 \end{bmatrix}$

$$KP \bmod 26 = \begin{bmatrix} 95 \\ 91 \end{bmatrix} \bmod 26$$

$$= \begin{bmatrix} 17 \\ 13 \end{bmatrix}$$

$$= \begin{bmatrix} r \\ n \end{bmatrix} \{ \text{Ciphertext} \}$$

Plaintext = "hi"
key = "jefh"
Ciphertext = "rn"

Play Fair Cipher

Keyword: MONARCHY
Plaintext: bu33 (buzz)

Rules of PlayFair Cipher
1) Pairs
2) Importance of x
3) going down
4) always right
5) Same row different column
6) i/j

5×5 matrix

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

Eg: text : "MISSION" Key: "IMPOSSIBLE"

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

text: "mission"

mi sx si on
pm pz im it

Ciphertext: PMPZIMIT

Eg: text: "you keep smiling"

Keyword: "happiness"

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

text: you keep smiling

yo uk ex ep sm il in gx

VT RM BV BH OG CT NH KW

Ciphertext: VTRMBVBHOGCTNHKW

- Monoalphabetic Cipher

- Polyalphabetic Cipher (Vigenere Process)

Encryption Process

$$C_i = (P_i + K_{i \bmod m}) \bmod 26$$

Key: deceptive
Plaintext: wearedisc

Key	3	4	2	4	15	19	8	21	4
PT	22	4	0	17	4	3	8	18	2
CT	25	8	2	21	19	22	16	13	6

$$C = (Key + PT) \bmod 26$$

Q-7 Explain One-Time Pad Substitution technique with help of example

Ans
→ Improvement to the Vernam Cipher
→ It yields the ultimate in security
→ Random Key that is as long as the message
→ 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

PT "come today"
Key NCBTZ@ARX

PT	C	O	M	E	T	O	D	A	Y
	2	14	12	4	19	14	3	0	24
Key	N	C	B	T	Z	@	A	R	X
	13	2	1	19	25	16	0	17	23
Total	15	16	13	23	44	30	3	17	47
Subtract 26 if >25	15	16	13	23	18	4	3	17	21
Ciphertext	P	@	N	X	S	E	D	R	V

Q-1 ~~Mart~~ Man - in - the - middle attack Explain

Ans Man - in - the Middle Attack is an attack in which an attacker is able to read, insert and modify at will, messages between two parties without either party knowing that the link between them has been compromised

- The MTM attack may include one or more of
1) Eavesdropping, including traffic analysis and possibly
2) Known - plaintext attack
3) Chosen ciphertext attack, depending on what the receiver does with a message that it decrypts
3) Substitution attack
4) Replay attacks
5) Denial of service attack

Ch-2 Steam Ciphers and Block Ciphers

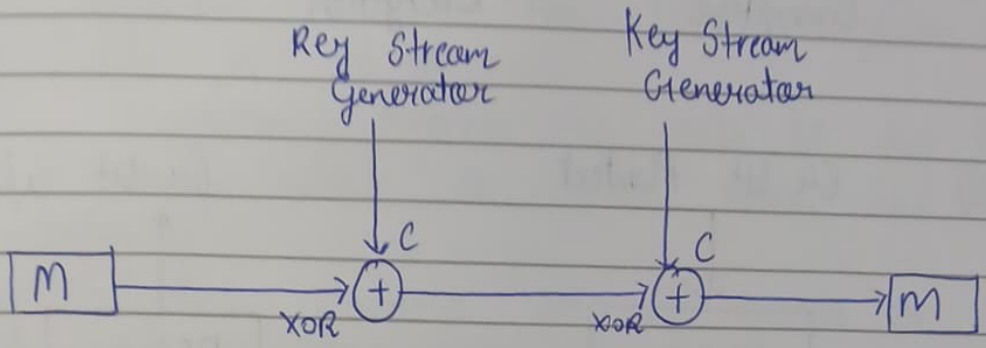
Q-1 Differentiate between Block Cipher and Steam Cipher

Ans

Stream Steam Cipher	Block Cipher
1) Steam ciphers operate on smaller units of plaintext	1) Operate on larger block of data
2) Faster than Block Cipher	2) Slower than steam cipher
3) Requires less code	3) Requires more code
4) Only one time of key use	4) Reuse of key is possible
5) Eg: One time Pad	5) Eg: DES
6) Application: SSL	6) Application: Database, File encryption
7) Steam Cipher is more suitable for hardware implementation	7) Easier to implement in software

Q.2 Stream Cipher

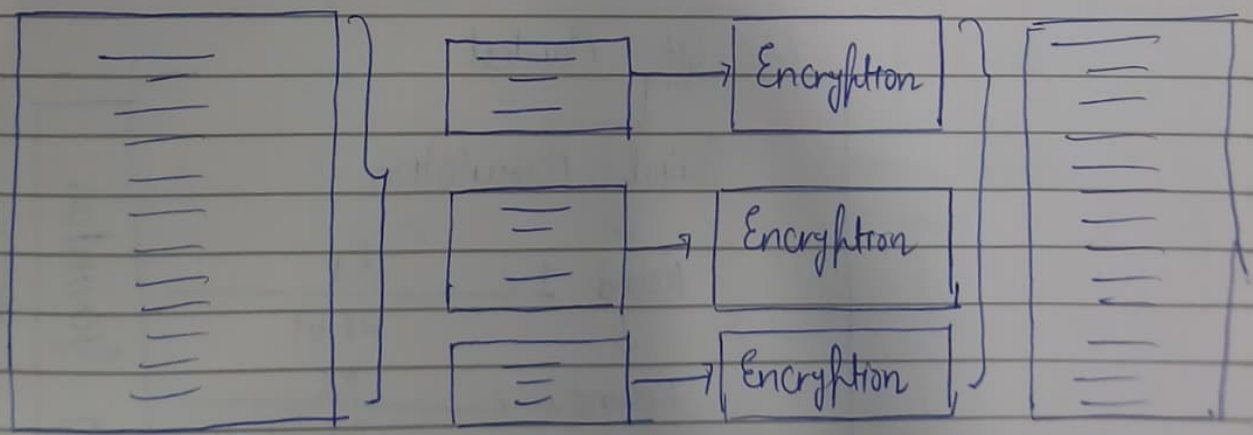
Ans Stream Cipher



1 1 0 0 1 1 1 0 0 1 → M
⊕ 0 1 0 1 0 1 1 1 0 0 → K
1 0 1 1 1 0 0 1 1 0 1 → C
⊕ 0 1 0 1 0 1 1 1 1 0 0
1 1 0 0 1 1 1 1 0 0 1 → M

Q.3 Block Ciphers

Ans

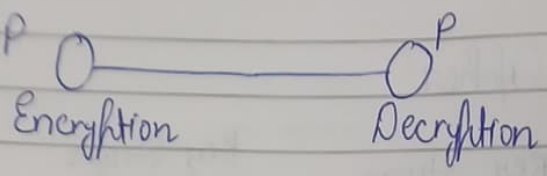


→ Block same size like n characters

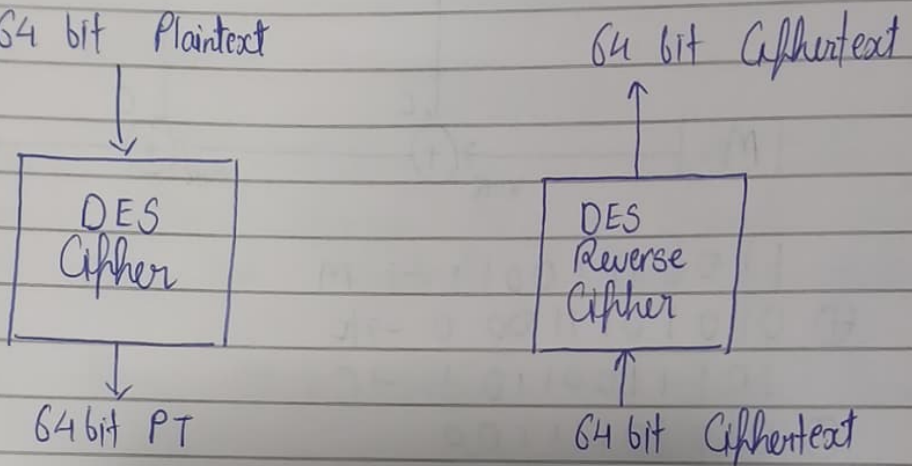
Q-4 DES Encryption

Ans

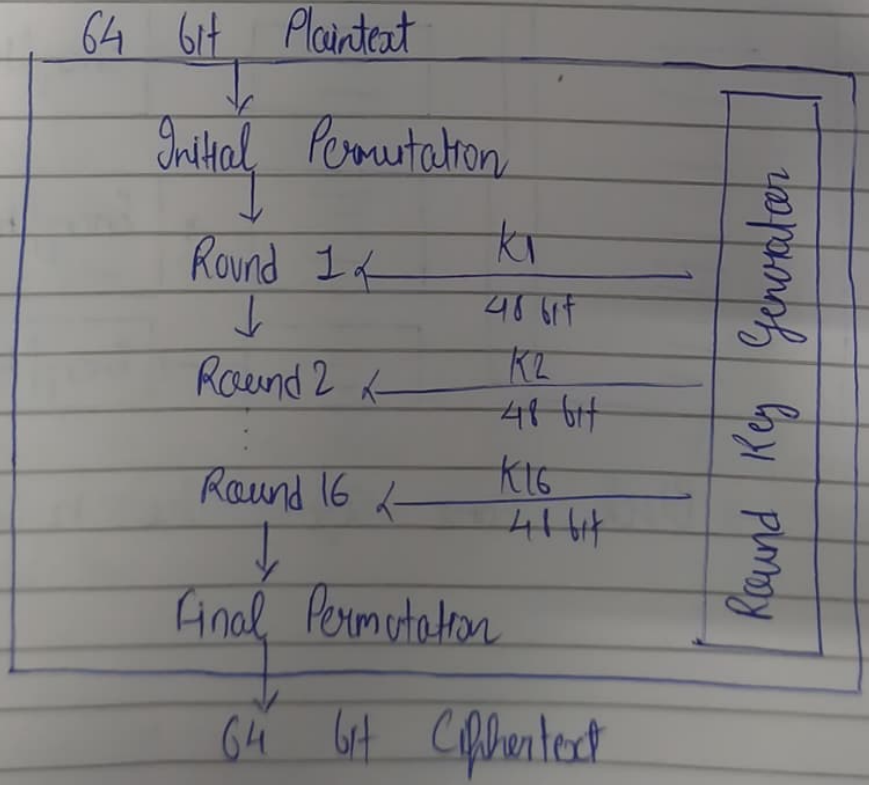
1)



2)



3) Initial Permutation

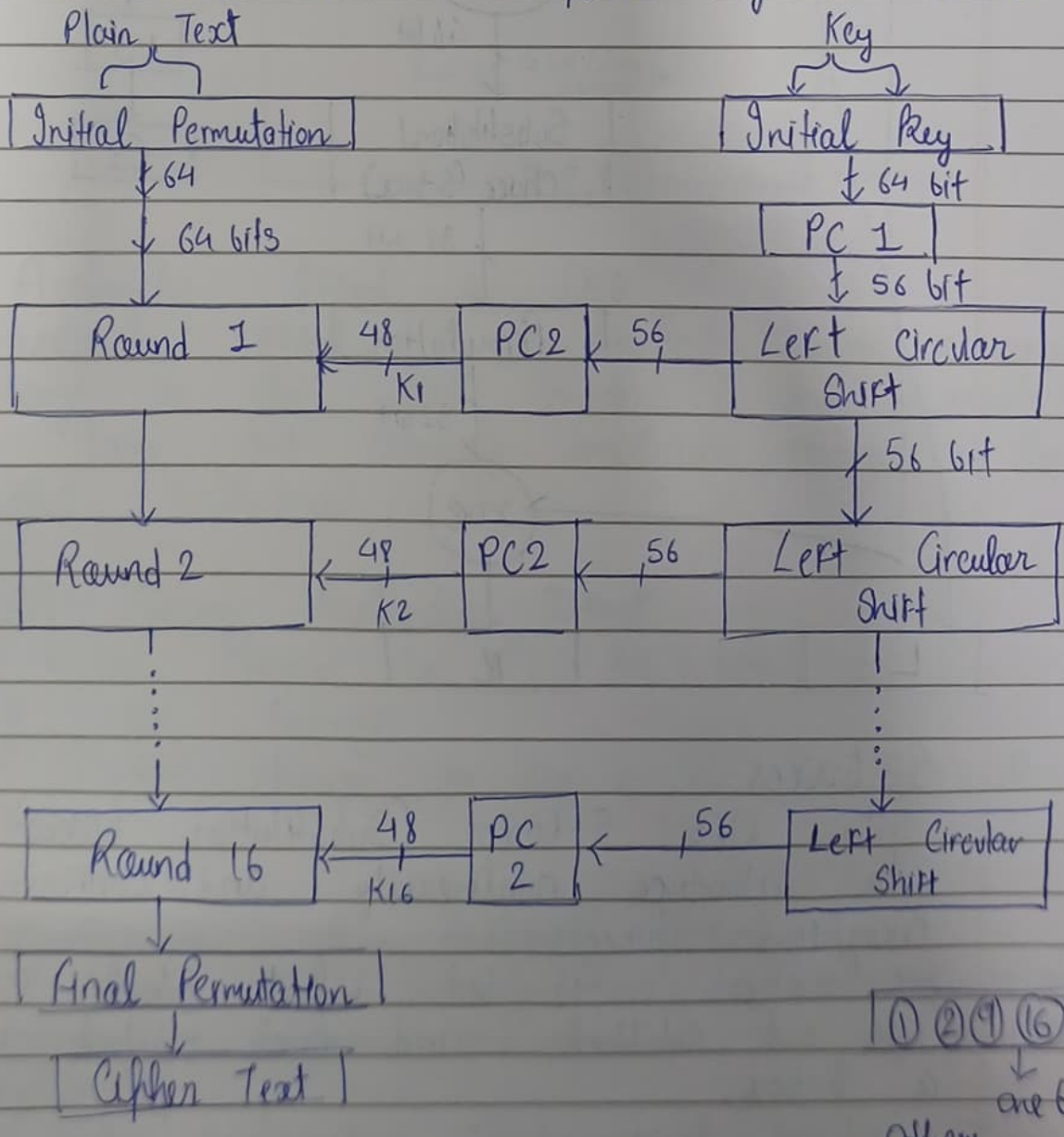


PC1 — 64 bit → 56 bit
every 8th bit remove

56 bit → 48 bit — PC2

Page No.
Date

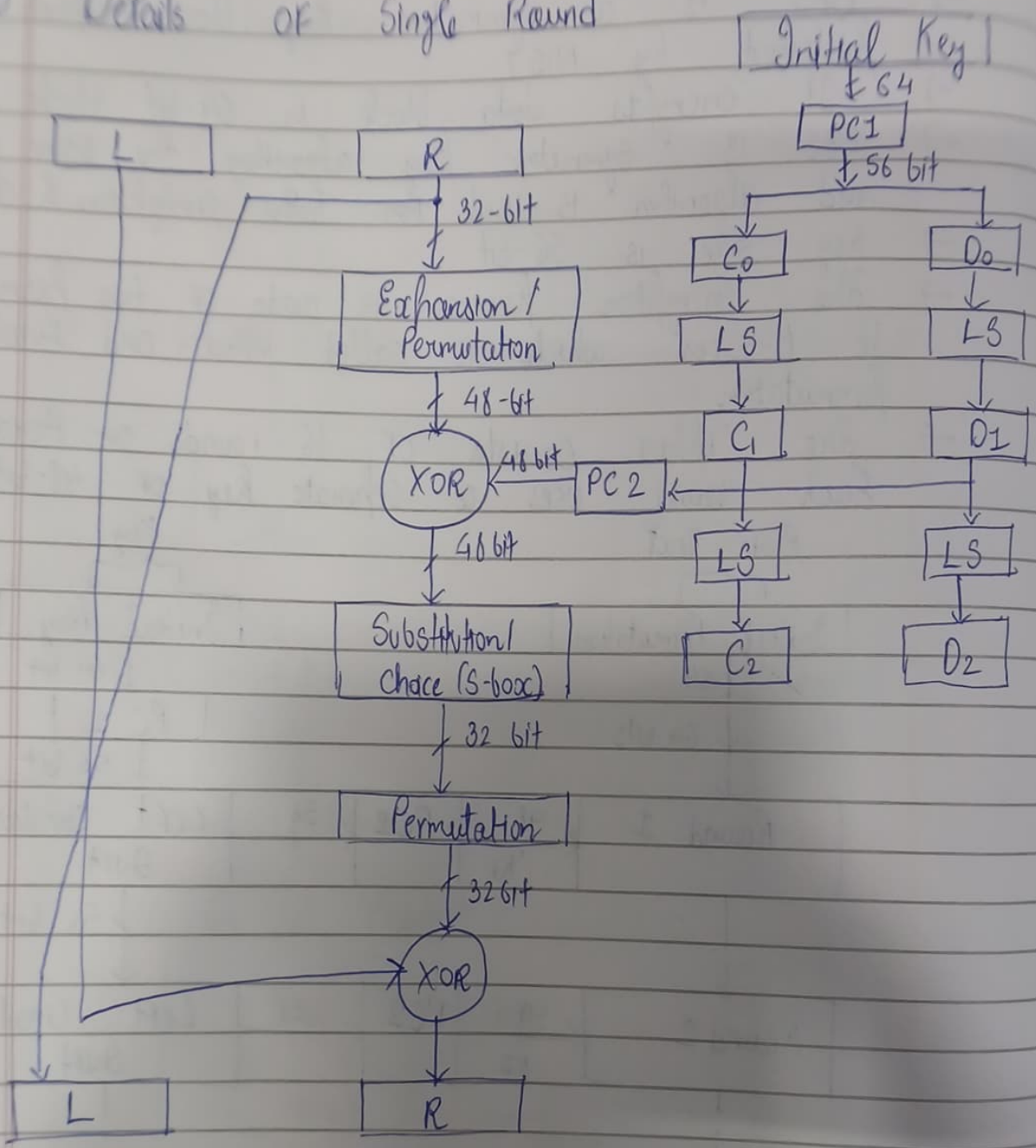
→ DES is a Symmetric Key block cipher published by NIST
→ It encrypts data block in 64-bit block
→ DES is Symmetric Key algorithm : the same Key and algorithm is used for both encryption & decryption.
→ Key size is 56-bit
→ The encryption process is made of two permutations je P-boxes, which is called initial and final permutation
→ The cipher consists of 16 rounds or iterations. Each round uses a seperate key of 48-bits



① ② ⑨ ⑯
↓
one bit

Others
↓
two bit

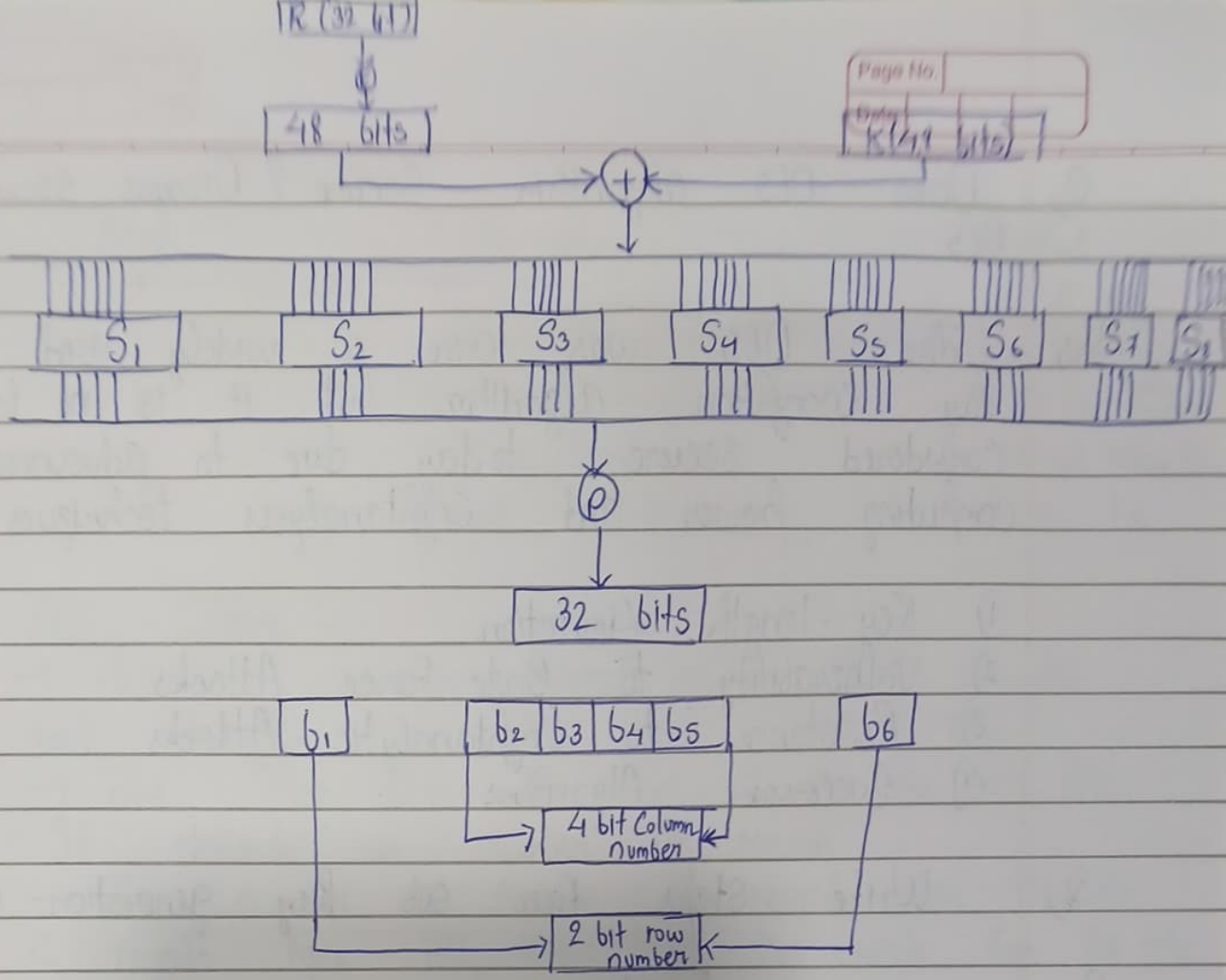
- Details of Single Round



- S-boxes

In DES, S-boxes (Substitution boxes) are used to introduce nonlinearity and confusion in the encryption process

→ The 48 bit input block is divided into 8 sub blocks and each subblock is given to a S-box.



- Avalanche Effect in DES

Avalanche effect is a property where a small change in the input (plaintext or key) results in a large and seemingly unrelated change in the output (ciphertext)

DES - Strong Avalanche Effect

→ 1 bit change in PT :- 34 bits change in CT on Average

→ 1 bit change in Key :- 35 bits changes in Average

Significance of Avalanche Effect

1) Ensure high security and unpredictability in encryption
2) Provides confusion and diffusion, two key principles of cryptography
3) Makes cryptanalysis attacks much harder

Page No.
Date

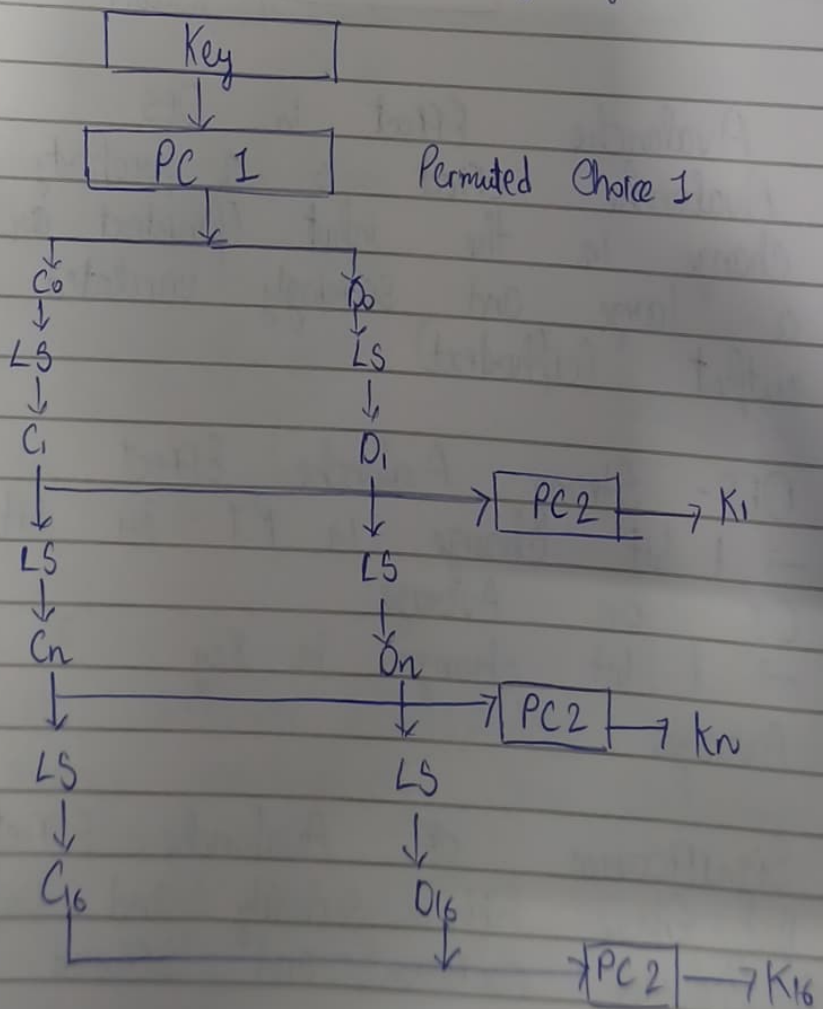
Q. Does DES algorithm Secure ? Discuss Security of DES

Ans The DES was once a widely used Symmetric key encryption algorithm, but it is no longer considered secure today due to advances in computing power and cryptanalysis techniques

1) Key - length Limitation
2) Vulnerability to Brute - Force Attacks
3) Resistance to Cryptanalytic Attacks
4) Successor Algorithms

Q. Write steps for sub key generation in DES

Ans



Step	Operation	Output
1	PC-1 (Drop parity bits)	56-bit Key
2	Split into C_0, D_0	Two 28-bit halves
3	Left Shift (1 or 2 bits)	C_i, D_i for each round
4	PC-2 (Compression)	48-bit subkey K_i
5	Repeat for 16 rounds	$K_1, K_2 \ldots K_{16}$

0 AES Cipher

→ AES is a non-Feistal cipher that encrypts and decrypts a data block of 128 bits

→ The key size can be 128, 192 or 256 bits. It depends on number of rounds

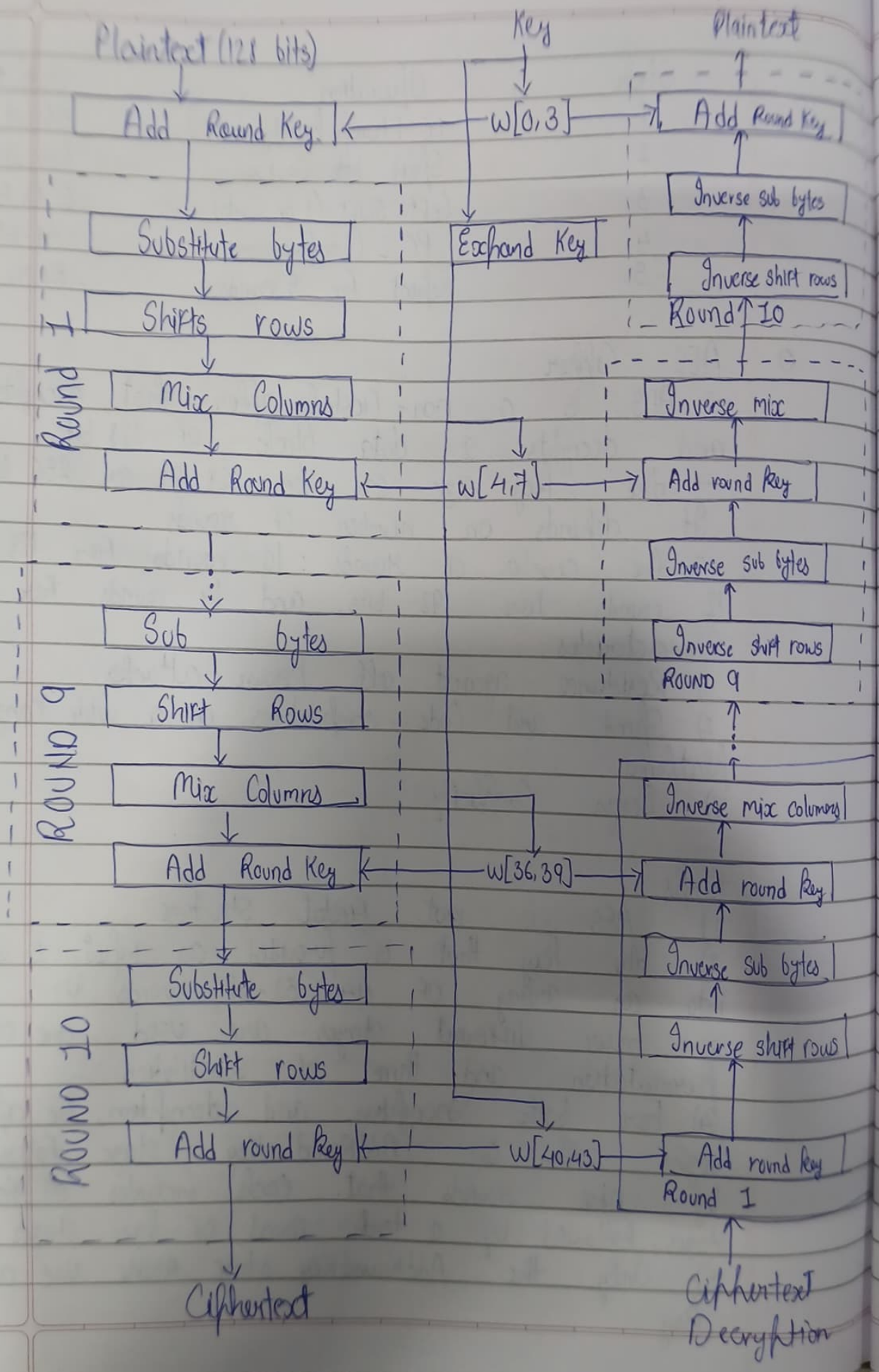
→ The number of rounds : 10 rounds for 128 bits, 12 rounds for 192 bits, and 14 rounds for 256 bits

Characteristics

1) Resistance against all known attacks
2) Speed and Code compactness on a wide range of platforms
3) Design Simplicity

Steps

(1) AES is not Feistal structure
2) The key that is provided as input is expanded into an array of 44 32-bit words w
(3) Four different stages are used, one of permutation and three of substitution
(4) For both encryption and decryption, the cipher begins with an AddRound Key stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages
5) Only the AddRoundKey stage make use of the key


Plaintext (128 bits)
Add Round Key
Substitute bytes
Shifts rows
Mix Columns
Add Round Key
Round 1
Sub bytes
Shift Rows
Mix Columns
Add Round Key
ROUND 9
Substitute bytes
Shift rows
Add round Key
ROUND 10
Ciphertext
Key
W[0,3]
Exchand Key
W[4,7]
W[36,39]
W[40,43]
Plaintext
Add Round Key
Inverse sub bytes
Inverse shift rows
Round 10
Inverse mix
Add round Key
Inverse sub bytes
Inverse shift rows
ROUND 9
Inverse mix columns
Add round Key
Inverse sub bytes
Inverse shift rows
Add round Key
Round 1
Ciphertext
Decryption

(6) The Add Round Key stage is, in effect, a form of Vernan Cipher and by itself would not be Formidable

(7) Each stage is easily reversible

(8) The Decryption algo makes use of expanded key in reverse order

9) Once it is established that all four stages are reversible, it is easy to verify that decryption does recover the plaintext

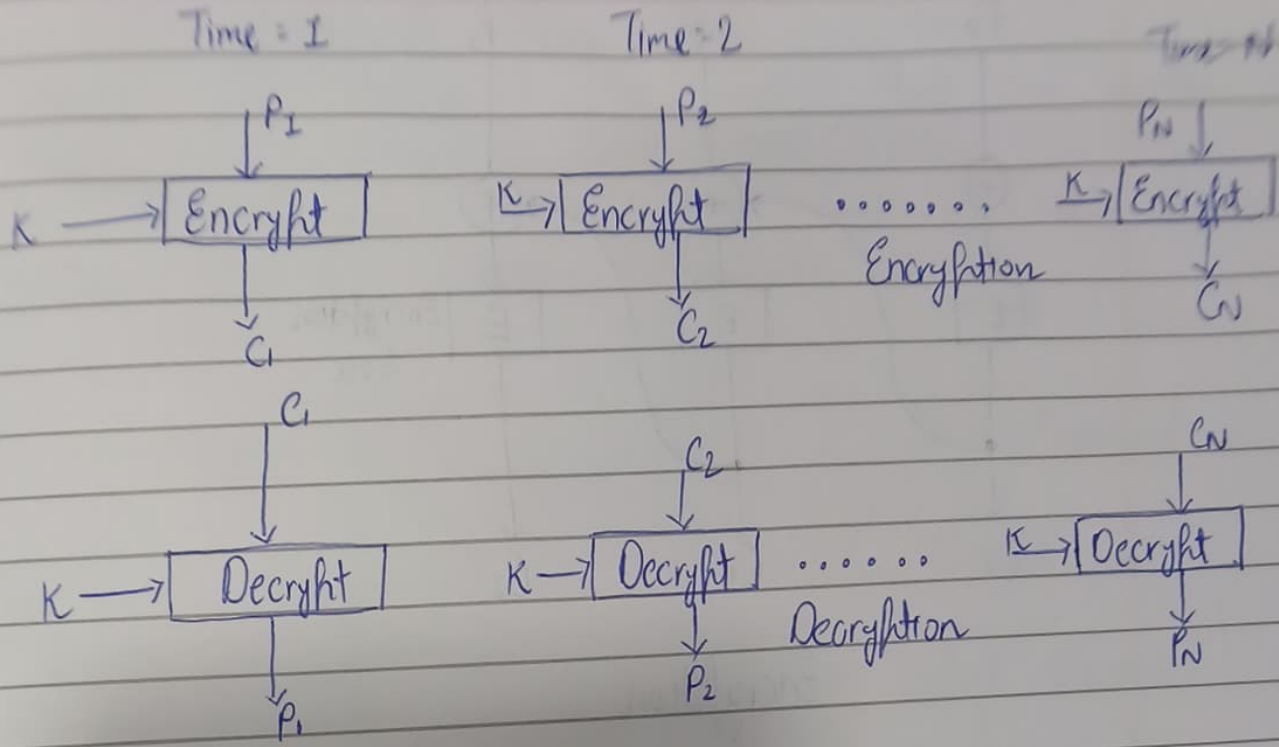
10) The final round of both encryption & decryption consists of only three stages

Ch-3 Multiple Encryption & Triple DES

List various mode of Cryptographic operation Explain any one with diagrams

1) Electronic Code book
2) Cipher Block Chaining Mode (CBC)
3) Cipher Feedback Mode

1) Electronic Code Book
→ A block of plaintext encrypts into a block of Ciphertext. Block size is 64 bits
→ Each Block is encrypted independently
→ It is not necessary to encrypt the File linearly
→ User can encrypt the 10 blocks in the middle first, then the blocks at the end, and finally the blocks in the beginning
→ Because of this, encrypted Files are accessed randomly like a database
→ It is very easy to parallelize the process
→ Pad the last block with some regular pattern ie zeroes, ones to make it a complete block
→ ECB method is ideal for a short amount of data, such as an encryption key
→ ECB not secure for lengthy messages
→ ECB has security problems that limit its usability



2) Cipher Block Chaining Mode (CBC)

→ The Plaintext is XORed with the previous ciphertext block if is encrypted

→ The CBC mode is iterative mode

→ After a plaintext block is encrypted, the resulting ciphertext is also stored in a Feedback register

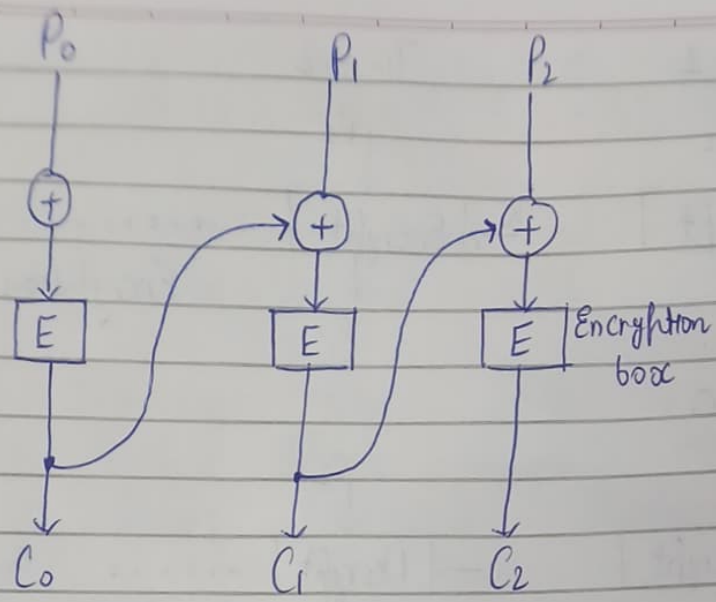
→ The encryption of each block depends on all the previous blocks.

→ After the next block is decrypted, it is XORed with the results of Feedback register

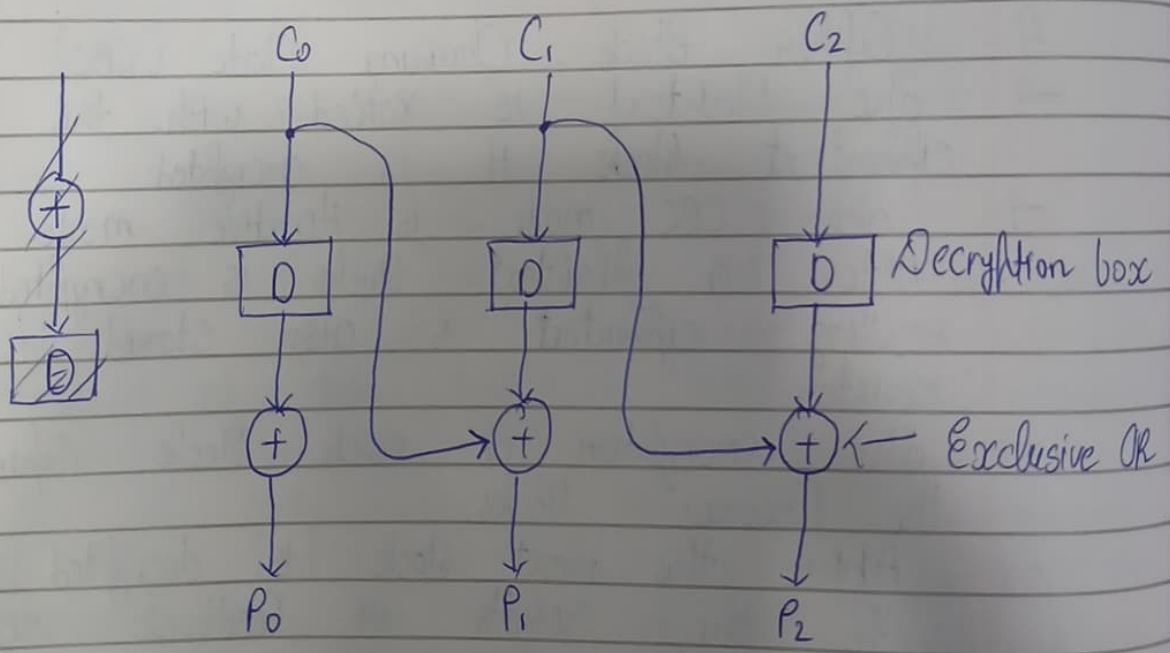
o Mathematically it is

$C_i = E_K (P_i \oplus C_{i-1})$

$P_i = C_{i-1} \oplus D_K (C_i)$



Encryption



Decryption

→ A single bit error in a plaintext block will affect that ciphertext block and all subsequent ciphertext blocks

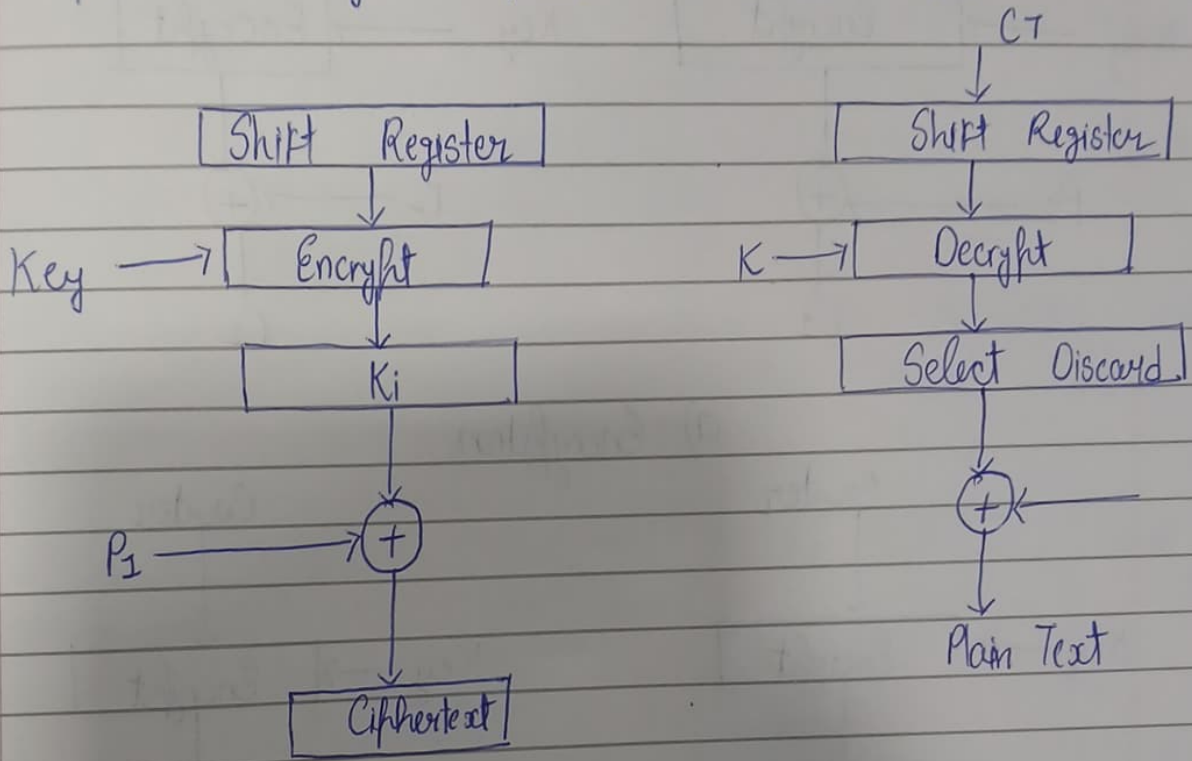
→ CBC mode is self recovering

→ Encryption is not parallelizable

3) Cipher Feedback Mode (CFB)

→ Data is encrypted in units that are smaller than a defined block size

→ It is possible to convert to DES into stream cipher using cipher feedback mode



→ More than one message can be encrypted with same key, provided that a diff. initialization vector is used

→ CFB speed is same as block cipher

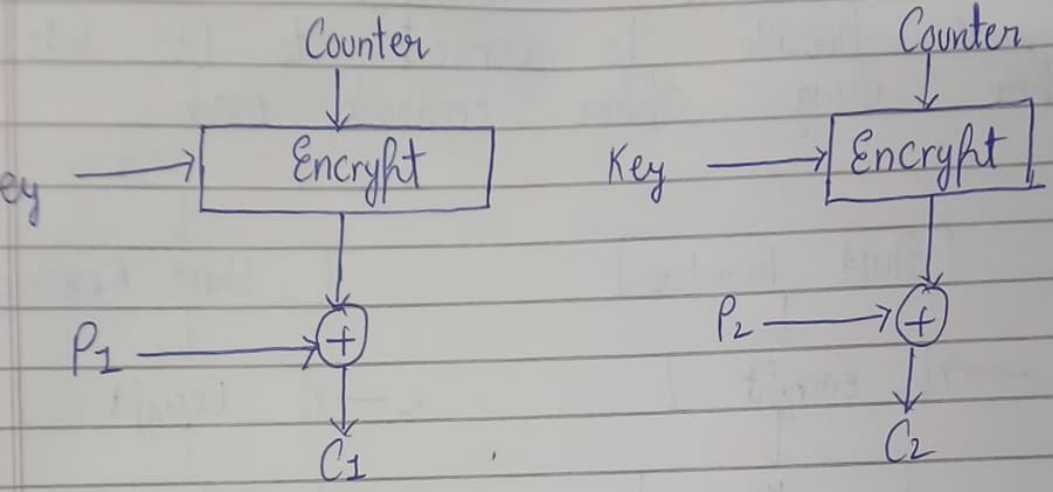
→ CFB is self recovering wrt synchronization errors as well

→ Encryption is not parallelizable, decryption is and has random access property

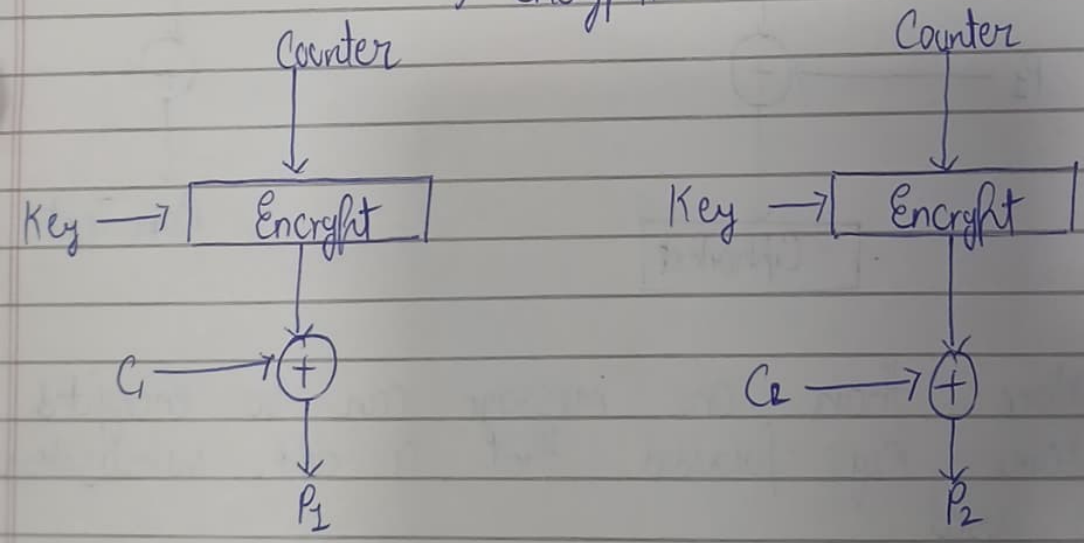
4) Counter Mode

→ Block ciphers in counter mode use sequence numbers as the input to the algorithm

Plaintext is very easy to manipulate, any change in ciphertext directly affects the plaintext



a) Encryption



(b) Decryption

Synchronization error is unrecoverable
A ciphertext error affects only the corresponding bit or plaintext.

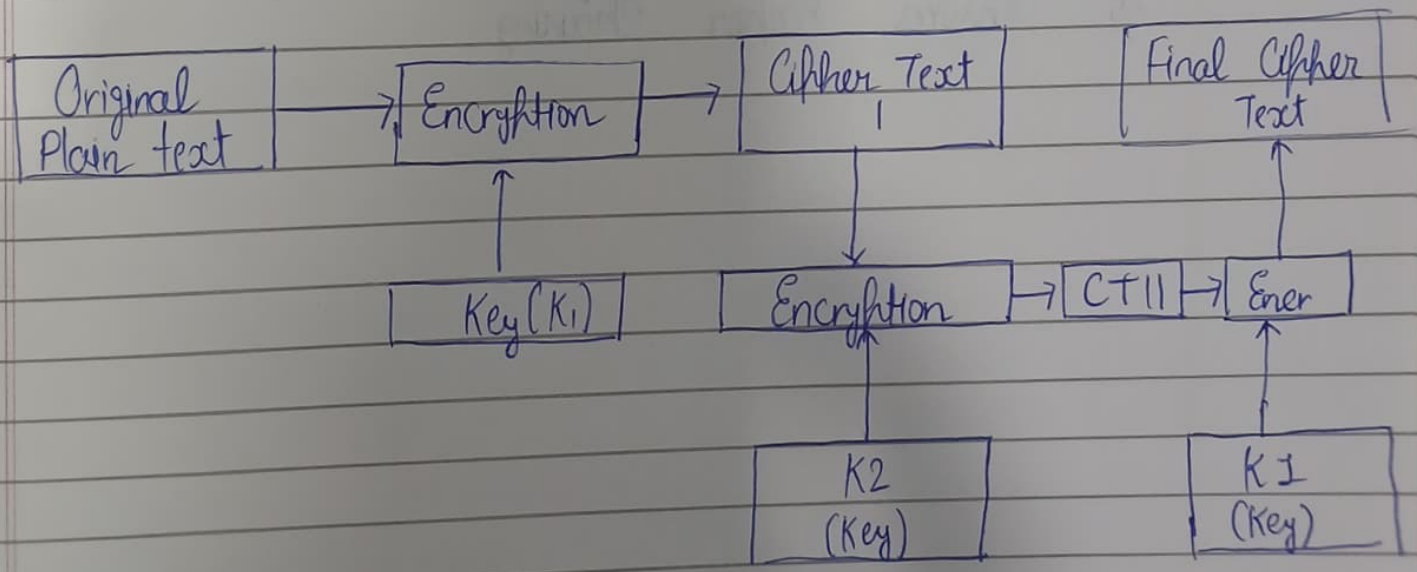
Q. Explain the triple DES scheme with two keys with suitable diagram

Ans Triple DES performs the same operation as double DES

→ Triple DES using two keys K_1 & K_2 while encrypting plain text

→ First it performs encryption on plaintext P, which is encrypted using K_1 obtains first cipher text C_1

→ Again this cipher text is encrypted using key K_2 which obtain the second cipher text C_2.



→ Which is again encrypted using K_1 & generate final cipher text C_3

Its Decryption will be reverse

Q What is an Initialization Vector (V)? What is its significance.

Ans An initialization Vector (IV) is a random or unique value ad used along with a secret key to initialize encryption in certain modes of block ciphers (like CBC, CFB, OFB etc)

- Significance of (IV)
  1) Prevents Repetition
  2) Provides randomness
  3) Enhances Security
  4) Ensures Proper Chaining

🔹 Significance / Purpose of IV:

1. **Prevents Repetition (Ensures Uniqueness):**
   Without an IV, encrypting the same plaintext with the same key would always produce the same ciphertext — easy for attackers to detect patterns.
2. **Provides Randomness:**
   The IV introduces randomness into the first block of encryption, making the ciphertext look completely different even for identical inputs.
3. **Enhances Security:**
   It protects against **known-plaintext** and **replay attacks** by making ciphertexts unpredictable.
4. **Ensures Proper Chaining:**
   In modes like **CBC (Cipher Block Chaining)**, the IV acts as a "previous ciphertext block" for the first encryption round.

🔹 Important Points:

- IV **does not need to be secret**, but it **must be unique and unpredictable** for every encryption operation.
- For **decryption**, the **same IV** used during encryption must be provided.

✅ In short:

> An **Initialization Vector (IV)** is a random value used to start the encryption process, ensuring that identical plaintexts encrypted with the same key produce **different ciphertexts**, thereby improving **security and randomness**.

Would you like me to explain **how IV is used in CBC mode** with a simple diagram or step example?

What is the limitation of Electronic Codebook Mode (ECB)? How it is overcome by
Cipher Block Chaining (CBC) mode.

Here's a clear and exam-ready explanation 👇

🔷 Limitation of Electronic Codebook (ECB) Mode:

In **ECB mode**, each plaintext block is **encrypted independently** using the **same key**.

$$C_i = E_K(P_i)$$

Because of this:

1. **Identical plaintext blocks → produce identical ciphertext blocks.**
   → Patterns in data remain visible in the ciphertext.
2. **No diffusion between blocks.**
   → Each block is encrypted separately — changes in one block **don't affect others**.
3. **Not suitable for long or structured data.**
   → Especially insecure for images, documents, or repeating data (patterns become visible).

📉 **Result:** ECB is **not secure** for practical use except for very short or random data.

🔷 How CBC (Cipher Block Chaining) Overcomes This Limitation:

In **CBC mode**, each plaintext block is **XORed** with the **previous ciphertext block before encryption**.

$$C_i = E_K(P_i \oplus C_{i-1})$$

- For the **first block**, a random **Initialization Vector (IV)** is used instead of C_0:

🔷 How CBC (Cipher Block Chaining) Overcomes This Limitation:

In **CBC mode**, each plaintext block is **XORed** with the **previous ciphertext block before encryption**.

$$C_i = E_K(P_i \oplus C_{i-1})$$

- For the **first block**, a random **Initialization Vector (IV)** is used instead of C_0:

$$C_1 = E_K(P_1 \oplus IV)$$

🔷 Advantages of CBC over ECB:

Aspect	ECB Mode	CBC Mode
Pattern visibility	Same plaintext → same ciphertext	Patterns are hidden
Block dependency	Each block independent	Each block depends on previous
Error propagation	No chaining	One block error affects next only
Use of IV	Not used	Used in first block for randomness
Security	Weak for repetitive data	Much stronger security

✅ In short:

> **ECB mode** leaks patterns because identical plaintext blocks produce identical ciphertexts.
> **CBC mode** fixes this by chaining blocks together — each block's encryption depends on the previous ciphertext (and the IV for the first block), making the ciphertext more secure and random.

Ch-4 Public Key Cryptosystems with Applications

Q-1 List the Requirements of public key cryptography

Ans 1) It is computationally easy for a party B to generate a pair ✡

2) It is computationally easy for the reciever B to decrypt the resulting ciphertext using the private key to recover the orignal message

3) It is ✡ for a sender A to generate the corresponding ciphertext

4) It is ✡ infeasible for an adversary, knowing the public key (PU_b) to determine the private PR_b

5) It is ✡ infeasible for an adversary, knowing the (PU_b) and a ciphertext (c) to recover the original message (m)

Q-2 Difference between Symmetric Key & Asymmetric Key

Ans

Comparison between Public Key and Private Key Algorithms

Sr. No.	Symmetric Key Cryptography	Asymmetric Key Cryptography
1.	Same key is used for encryption and decryption.	One key is used for encryption and the other key for decryption.
2.	Very fast.	Slower.
3.	Key exchange is a big problem.	Key exchange is not a problem.
4.	Also called *secret key encryption*.	Also called *public key encryption*.
5.	The key must be kept secret.	One of the two keys (private key) must be kept secret.
6.	The sender and receiver must share both the algorithm and the key.	The sender and receiver must each have one of the matched pair of keys.
7.	Size of the resulting encrypted text is usually the same as or less than the original clear text size.	Size of the resulting encrypted text is more than the original clear text size.
8.	Cannot be used for digital signatures.	Can be used for digital signatures.

PT: Plaintext
CT: Ciphertext

Page No.
Date

Q-3 Explain the RSA algorithm

Ans → RSA is a block cipher in which the PT & CT are integers between 0 and n-1 for some n
→ Typical size for n is 1024 bits
→ One user uses a public key & another uses private key

- Key Generation

1) Pick two large prime numbers p and q

2) Calculate $n = p \times q$

3) Calculate $\phi(n) = (p-1)(q-1)$

4) Pick e, so that $\gcd(e, \phi(n)) = 1$, $1 < e < \phi(n)$,

5) Calculate d, so that $d.e \bmod \phi(n) = 1$
ie d is multiplicative inverse of e in $\bmod \phi(n)$

6) Get Public key $K_U = \{e, n\}$

7) Get Private key $K_R = \{d, n\}$

- Encryption
For Plaintext block $P < n$, $C = P^e \bmod n$

- Decryption
For Ciphertext block C, $P = C^d \bmod n$

Page No.
Date

Q Compute Public and Private Key of RSA with $p=11$, $q=17$ and $e=7$

Ans
$$n = p \times q$$
$$= 11 \times 17$$
$$= 187$$

$$\phi = (p-1)(q-1)$$
$$= 10 \times 16$$
$$= 160$$
$$\phi = 160$$

$d \bmod \phi = 1$
$ax + by = \gcd(a, b)$

$a = \phi$, $b = e$

$\phi x + ey = \gcd(\phi, e)$
$160x + 7y = \gcd(160, 7)$

No	a	b	d	k
1	1	0	160	-
2	0	1	7	22
3	1	-22	13	
4				
5				

Won't Work here

$$7 \overline{)160}\ 22$$
$$14$$
$$20$$
$$2$$

$d \bmod \phi = 1$
$ed = 1 \bmod \phi(n)$

$$d = \frac{(\phi(n) \times i) + 1}{e}$$

$$d = \frac{(160 \times 1) + 1}{7} = 23$$

$d = 23$

Public Key $K_U = \{e, n\} = \{7, 187\}$

Private Key $K_R = \{d, n\} = \{23, 187\}$

Q. RSA $P = 53$, $q = 59$ & $e = 3$

Ans $n = P \times q$

$= 53 \times 59$

$= 3127$

$\phi(n) = (P-1)(q-1)$

$= 52 \times 58$

$= 3016$

$de \bmod \phi = 1$

$$d = \frac{(\phi(n) \times i) + 1}{e}$$

$$= \frac{3016 \times 1 + 1}{3}$$

$$= \frac{3017}{13}$$

$= 491$ (as $13 \times 491 = 6383$)

$$= \frac{(3016 \times 2) + 1}{3}$$

$$= \frac{6033}{3}$$

$$\boxed{d = 2011}$$

$Ku = \{e, n\} = \{3, 3127\}$

$Kr = \{d, n\} = \{2011, 3127\}$

Q// Explain Diffie Hellman Key Exchange Algorithm

Ans
- Select two numbers
  1) prime number q
  2) α is an integer that is primitive root of q

- Suppose the users A and B wish to exchange the key
  1) User A selects a random integer $X_A < q$ and computes $Y_A = \alpha^{X_A} \bmod q$

  2) User B selects a random integer $X_B < q$ and computes $Y_B = \alpha^{X_B} \bmod q$

  3) Both side keeps the X value private and makes the Y value available publicly to other side

  4) User A computes key as $K = (Y_B)^{X_B} \bmod q$

List application of RSA algorithm

Here's a clear and concise list of the **applications of the RSA algorithm**, perfect for exams 👇

🔷 Applications of RSA Algorithm

1. **Data Encryption**
   - RSA is used to **encrypt sensitive data** so that only the intended receiver (who has the private key) can decrypt it.
   - Example: Securing communication between a client and server.

---

2. **Digital Signatures**
   - RSA is widely used to **create and verify digital signatures**.
   - The sender signs a message using their **private key**, and the receiver verifies it using the **public key**.

---

3. **Key Exchange**
   - RSA helps in **securely exchanging symmetric keys** over insecure channels.
   - Often used to establish a session key in protocols like **SSL/TLS**.

---

4. **Secure Email**
   - Used in applications like **PGP (Pretty Good Privacy)** and **S/MIME** for **encrypting and signing emails**.

---

5. **Digital Certificates**
    - RSA is used in **Certificate Authorities (CAs)** to sign and verify **digital certificates** that authenticate websites and organizations.

---

6. **Authentication**
    - RSA can verify the **identity of users or devices** during secure logins or transactions.

---

7. **VPNs and Secure Network Protocols**
    - Used in **HTTPS, SSH, VPNs, SSL/TLS** for secure communication over networks.

✅ In short:

> RSA is used for **encryption, digital signatures, secure key exchange, authentication, and digital certificates** — forming the backbone of modern secure communication.

Would you like me to make a **short 5-mark version (just key points)** for quick memorization?

Q// Explain Diffie Hellman Key Exchange Algorithm

Ans
- Select two numbers
  1) prime number q
  2) α is an integer that is primitive root of q

- Suppose the users A and B wish to exchange the key
  1) User A selects a random integer $X_A < q$ and computes $Y_A = \alpha^{X_A} \bmod q$

  2) User B selects a random integer $X_B < q$ and computes $Y_B = \alpha^{X_B} \bmod q$

  3) Both side keeps the X value private and makes the Y value available publicly to other side

  4) User A computes key as $K = (Y_B)^{X_B} \bmod q$

5) User B computes the key as $K = (Y_A)^{X_B} \bmod q$

Eg:

$q = 353$

$\alpha = 3$

Let, A & B keys be

$X_A = 97$ $X_B = 233$

- Calculate the public keys

A computes $Y_A = \alpha^{X_A} \bmod q$

$= 3^{97} \bmod 353$

Eg: $q = 13$

$\alpha = 3$

Let $X_A = 5$ $X_B = 4$ [Private keys]

$Y_A = \alpha^{X_A} \bmod q$

$= 3^5 \bmod 13$

$= 9$

$Y_B = \alpha^{X_B} \bmod q$

$= 3^4 \bmod 13$

$= 3$

After exchanging keys, each can compute common secret key

A computes $K = (Y_B)^{X_A} \bmod q$

$= (3)^5 \bmod 13$

$= 9$

B computes K = $(Y_A)^{X_B} \bmod q$
= $(9)^4 \bmod 13$
= 6561 mod 13
= 9

Limitations of Diffie Hellman Key Exchange
1) Does not protect against man-in-the-middle attacks
2) Even can Lack of authentication procedure
3) As it is computationally intensive, it is expensive in terms of resources and CPU performance time
4) Digital Signature cannot be signed by Diffie Hellman
5) Encryption of information cannot be performed with help of this algorithm

Q11 GTU W-23 Q-3(c)
$q = 17$
$\alpha = 5$
$X_A = 4$ $X_B = 6$

1) Calculate the public keys
A computes $Y_A = \alpha^{X_A} \bmod q$
= $(5)^4 \bmod 17$
= 13
B computes $Y_B = \alpha^{X_B} \bmod q$
= $(5)^6 \bmod 17$
= 2

After exchanging keys, each can compute common secret key

A computes $K = (Y_B)^{X_A} \bmod q$
= $(2)^4 \bmod 17$
= 16 mod 17
= 16

B computes $K = (Y_A)^{X_B} \bmod q$

$= (13)^6 \bmod 17$

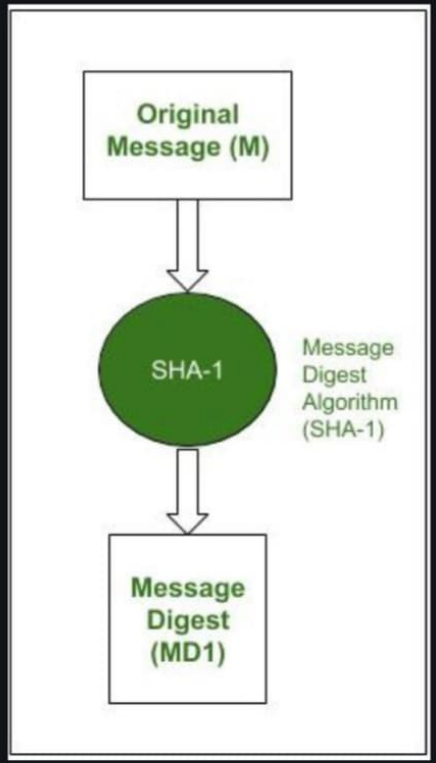
$= 16$

$\boxed{K = 16}$

RSA Signature Scheme

Let us understand how RSA can be used for performing **digital signatures step-by-step**. Assume that there is a sender (A) and a receiver (B). A wants to send a message (M) to B along with the digital signature (DS) calculated over the message.

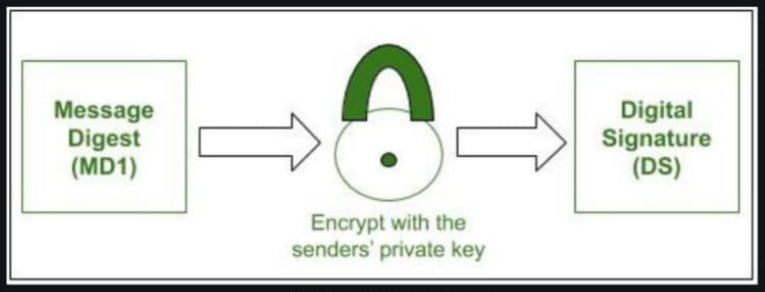
- **Step 1:** Sender A uses the SHA-1 Message Digest Algorithm to calculate the message digest (MD1) over the original message M.



Message digest calculation

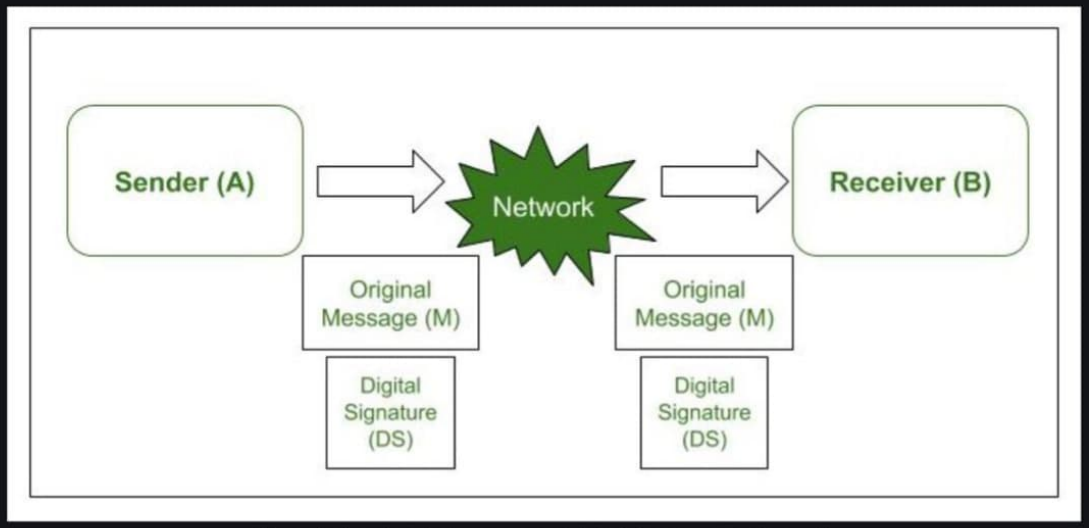
Message digest calculation

- **Step 2 :** A now encrypts the message digest with its private key. The output of this process is called Digital Signature (DS) of A.



Digital signature creation

- **Step 3 :** Now sender A sends the digital signature (DS) along with the original message (M) to B.

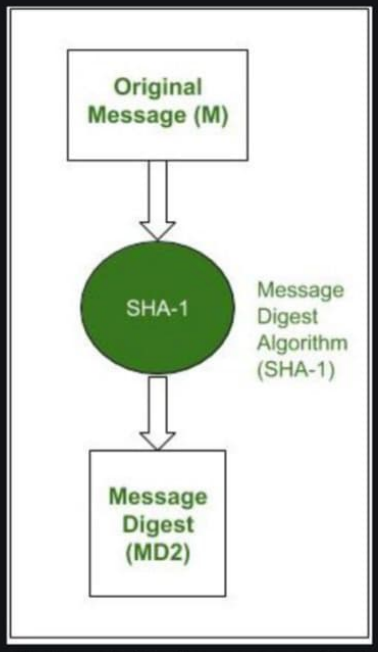


Transmission of original message and digital signature simultaneously

- **Step 4 :** When B receives the Original Message(M) and the Digital Signature(DS) from A, it first uses the same message-digest algorithm as was used by A and calculates its own Message Digest (MD2) for M.

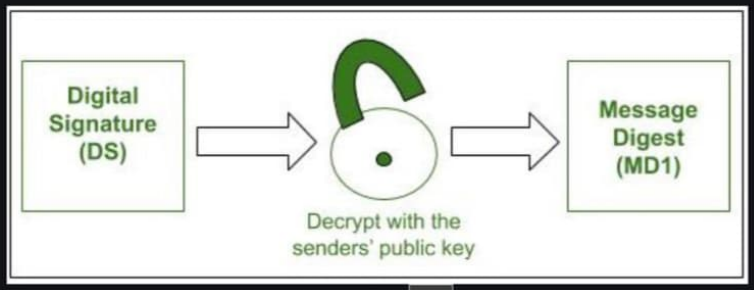
Transmission of original message and digital signature simultaneously

- **Step 4 :** When B receives the Original Message(M) and the Digital Signature(DS) from A, it first uses the same message-digest algorithm as was used by A and calculates its own Message Digest (MD2) for M.

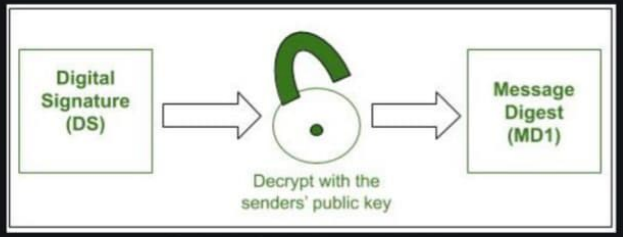


Receiver calculates its own message digest

- **Step 5 :** Now B uses A's public key to decrypt the digital signature because it was encrypted by A's private key. The result of this process is the original Message Digest (MD1) which was calculated by A.

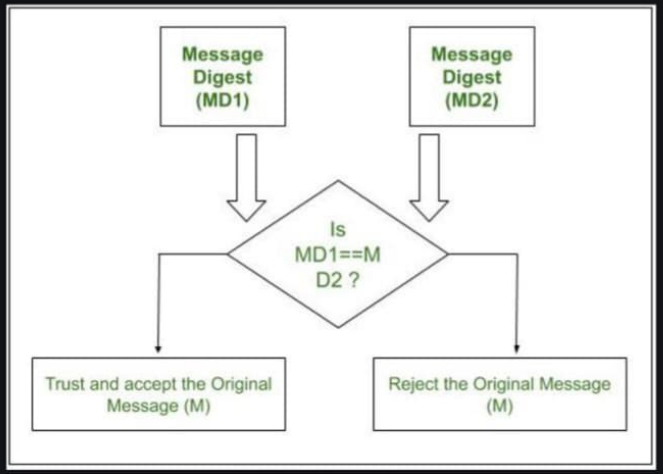


- **Step 5 :** Now B uses A's public key to decrypt the digital signature because it was encrypted by A's private key. The result of this process is the original Message Digest (MD1) which was calculated by A.



Receiver retrieves sender's message digest

- **Step-6 :** If MD1==MD2, the following facts are established as follows.
  - B accepts the original message M as the correct, unaltered message from A.
  - It also ensures that the message came from A and not someone posing as A.



Digital signature verification

The message digest (MD1) was encrypted using A's priva key to produce a digital signature. Therefore, the

Ch-5 Cryptographic Hash Functions

Q-1 Define Hash Function in Cryptographic and list its application

Ans Cryptographic has Functions are mathematical algorithms that transform input data into a fixed length sequence of length characters, refered to as hash values

A hash Function is a computationally efficient Function mapping binary strings to of arbitrary length to binary strings of some fixed length, called hash-values

- Applications of Hash Function
  1) Password Storage
  2) Digital Signatures
  3) Data Integrity Verification
  4) Message Authentication Codes
  5) Block Chain and Cryptocurrencies
  6) Digital Certificates

Q-2 Explain a simple hash Function and its limitation

Ans See Q-1

Limitations
1) Collisions Occur: Different inputs can produce the same hash value

2) Poor Security
→ Simple hash functions are easy to reverse or predict - not suitable for cryptographic use

3) Non-Uniform distribution:
→ Hash values may cluster in certain ranges, reducing efficiency in data structures

4) No resistance properties
→ They do not satisfy pre-image, second pre-image or collision resistance, which are required for cryptographic security

Q-3 Requirements for a Cryptographic Hash Function?

Ans 1) H can be applied to a block of data of any size
2) H produces a fixed length output
3) H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical
4) For any given value h, it is computationally infeasible to find x such that H(x) = h. This is called one-way property
5) For any given block x, it is computationally infeasible to find y ≠ x such that H(y) = H(x) This is called weak collision resistance
6) It is computationally infeasible to find pair (x, y) such that H(x) = H(y). This is called as strong collision resistance

Page No.
Date

Q-4 What is Collision resistance property of Hash function ?

Ans Collision Resistance means that it should be computationally infeasible to find two different inputs that produce the same hash value
Formally,
If $h(x1) = h(x2)$, where $x1 \neq x2$
then such a pair (x_1, x_2) is called a collision
Eg:
$h("cat") = 256BH$
$h("dog") = 256BH$
→ A Collision has occured

Q-5 Explain the following properties of hash function
(i) One Way Property
(ii) Weak Collision Property

Ans (i) One way Property
→ A hash function is said to have a one-way property if, given a hash value $h(x)$, it is computationally infeasible to find the original input x

if $h("Hello") = A4594$
It should be impossible to find the original message "Hello" from A4594

(ii) Weak Collision Property
→ A hash function is weak collision resistant if, for a given input $X1$
It is hard to find another input $X2$ (where $X_1 \neq X_2$) such that
$h(x_1) = h(x_2)$

Eg:
If h("Message 1") = 98AB23
it should be extremely difficult to find another "Message 2" such that h("Message 2") = 98AB23.

Q-6 List Basic Uses of Hash Functions

Ans See Image

Basic Uses of Hash Function (7 Marks Answer):

Hash functions play a vital role in computer security and data integrity.
Below are the **main uses (applications)** of hash functions explained clearly:

---

1. Password Storage

- Instead of storing actual passwords, systems store **hash values** of passwords.
- When a user logs in, the entered password is hashed and compared with the stored hash.
- This ensures security even if the database is compromised.

---

2. Data Integrity Verification

- Hash functions are used to check whether data has been modified during transmission or storage.
- By comparing the **hash of the original data** with the **hash of the received data**, changes can be detected.

---

3. Digital Signatures

- Hash functions generate a **message digest** that is digitally signed instead of signing the whole message.
- This makes the process faster and ensures message authenticity.

---

4. Message Authentication Code (MAC)

- Hash functions combined with a secret key produce a **MAC**, which verifies both **integrity** and **authenticity** of a message.

5. Digital Certificates and SSL/TLS

- Used in digital certificates to verify the **identity of websites and users**.
- Ensures secure communication over the internet.

---

6. Blockchain and Cryptocurrencies

- Hash functions link blocks together securely.
- They ensure **immutability** and prevent alteration of blockchain data.

---

7. File or Data Indexing

- Hash functions help in **fast data retrieval** by mapping large data sets to smaller hash tables (e.g., in databases and file systems).

✅ **In Summary:**

Use	Purpose
Password storage	Protects user credentials
Data integrity	Detects data changes
Digital signatures	Ensures authenticity
MAC	Verifies message authenticity
Digital certificates	Secures online communication
Blockchain	Maintains immutability
File indexing	Enables fast data access

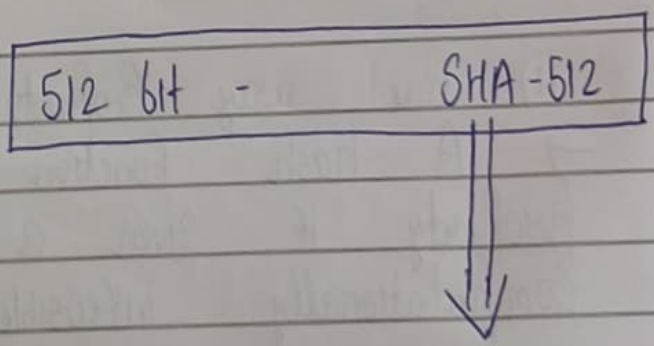
Q-7 Explain SHA-512 [Asked many times 4m, 7m in different ways]

Ans Secure Hash Algorithm (SHA)

If output - 128 bit - SHA-1

256 bit - SHA-256

512 bit - SHA-512



$H(M) = h(512\text{-bit})$

- SHA 512
  → Here Plaintext is processed in terms of block

Plain Text Block Size = 1024 bits
No & Rounds / Steps = 80
Each Round → @ Word = 64 bit
(W) ↓ generated from plain text

PT → Plain Text

Each round → constant K

Buffers → Store Intermediate Results
→ Store O/P (Hash Code)
Each buffer size = 64 bit
Total 8 buffer [8 × 64 = 512]

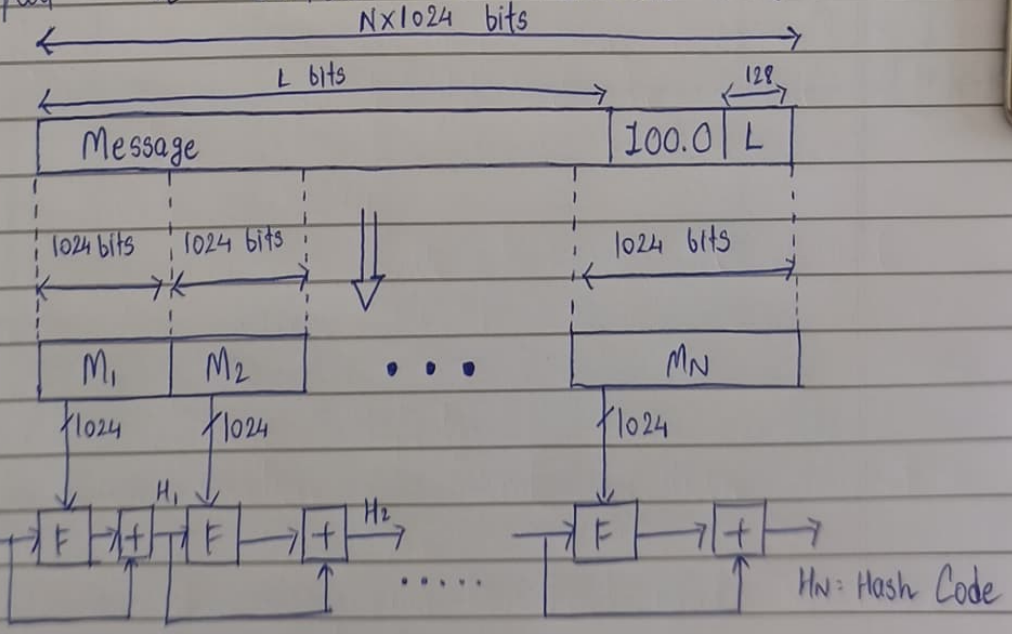
① Pad the bits 100... so that length of PT is 128 < Multiple of 1024 bits

② Append 128 bit representation of original plain text such that length = Multiple

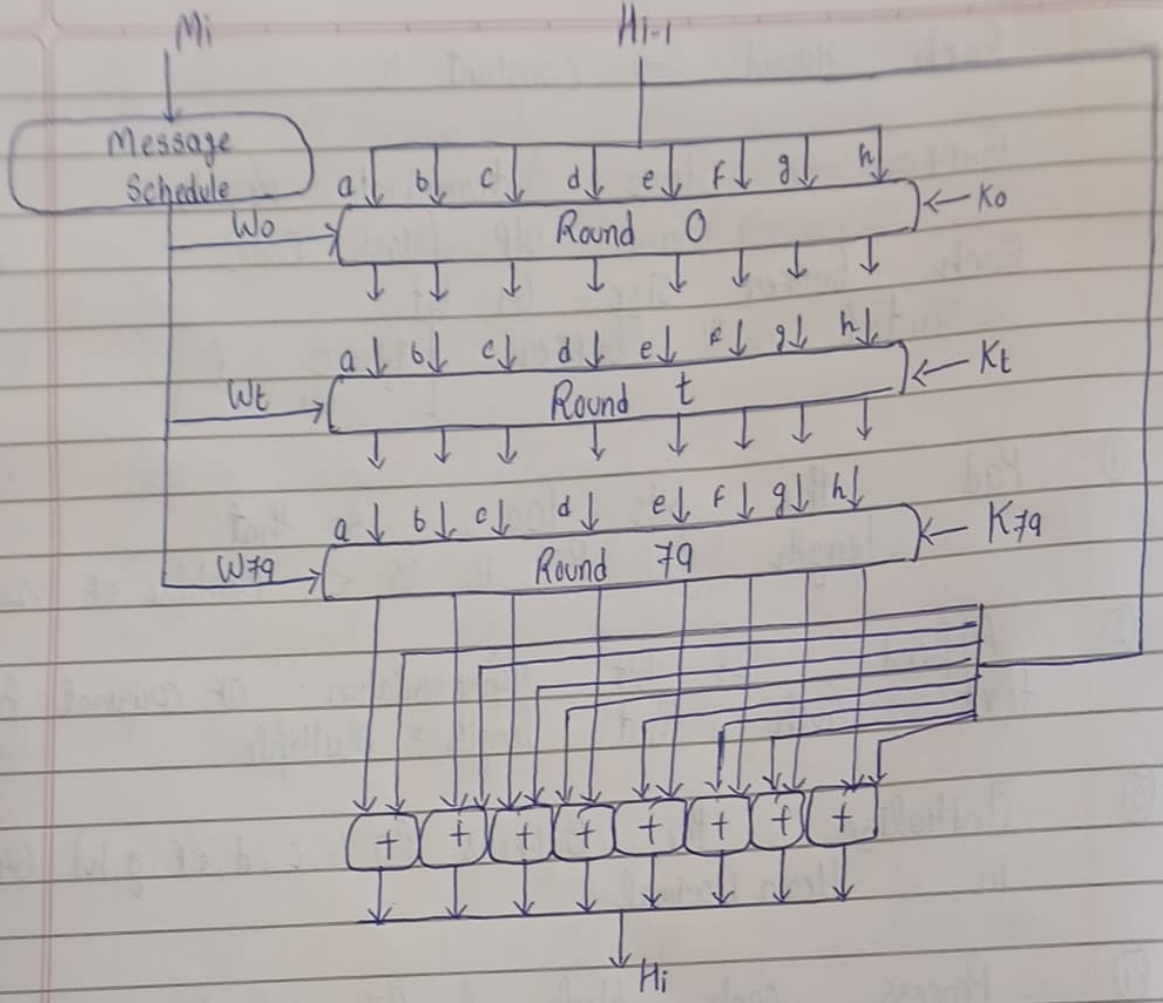
③ Initialize the Buffers (a, b, c, d, e, f, g, h) 64 bit in Hexa Decimal

④ Process each block & PT in 80 rounds/steps

⑤ Output in Buffers is HashCode (512)



Message Digest using SHA 512



SHA 512 Processing of Single 1024-bit block

SHA -512 round Function

Each round is defined by the following set of equations

$$T_1 = h + Ch(e,f,g) + (\Sigma_1^{512} e) + W_t + K_t$$

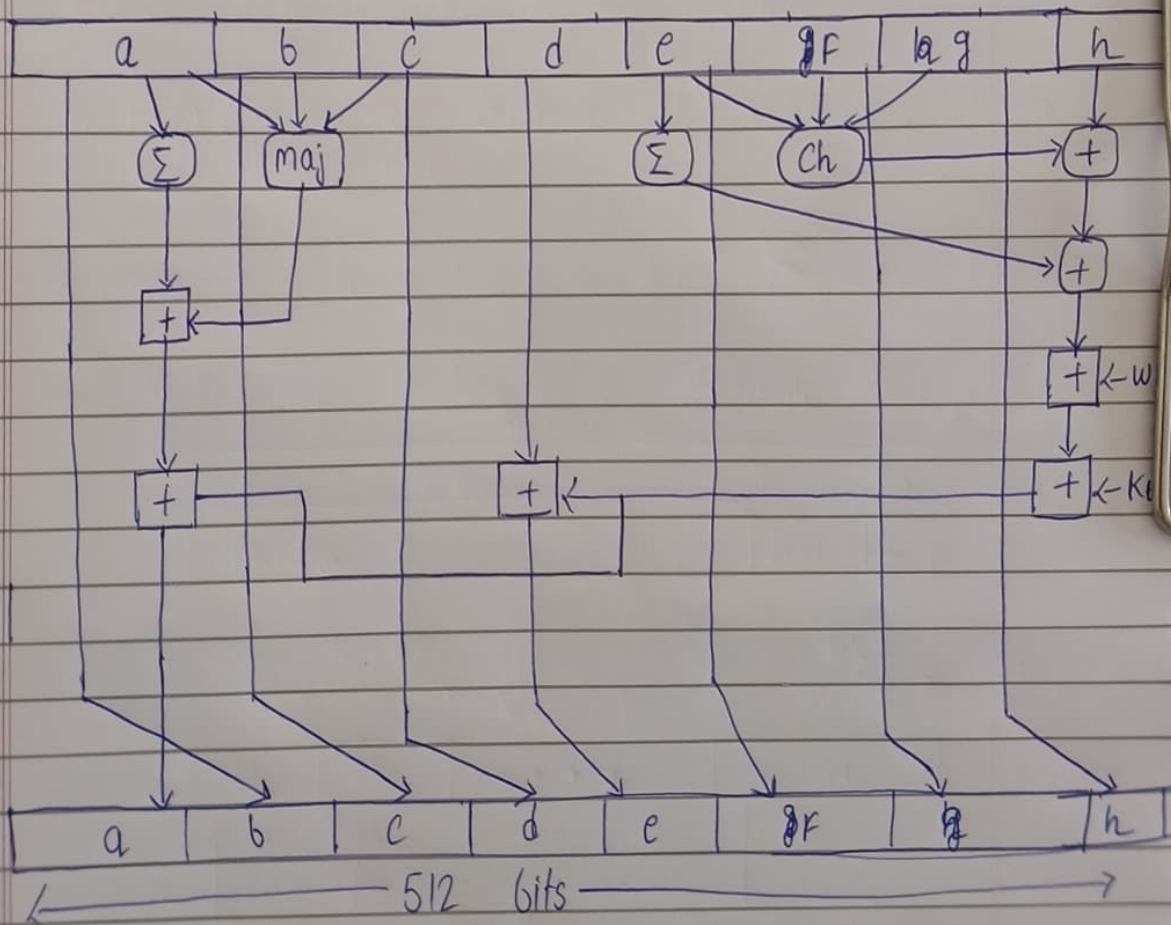
$$T_2 = (\Sigma_0^{512} a) + Maj(a,b,c)$$

$a = T_1 + T_2$ $\quad e = d + T_1$

$b = a$ $\quad f = e$

$c = b$ $\quad g = f$

$d = c$ $\quad h = g$



S What is block size and message digest size in SHA 512? With the help of diagram explain a round of SHA-512 algorithm

SHA-512 Specifications

Block Size and Message Digest Size

Parameter	Size
Block Size	**1024 bits** (128 bytes)
Message Digest Size	**512 bits** (64 bytes)
Word Size	64 bits
Number of Rounds	80 rounds
Initial Hash Values	Eight 64-bit words (a, b, c, d, e, f, g, h)

Message Digest Generation Using SHA-512 (Concise)

Overview

- **Output:** 512-bit (64-byte) hash
- **Block Size:** 1024 bits
- **Word Size:** 64 bits
- **Rounds:** 80 per block

Process Steps

Step 1: Padding

1. Append '1' bit to message
2. Append '0' bits until length ≡ 896 (mod 1024)
3. Append 128-bit message length
4. Result: Message length is multiple of 1024 bits

Example: Message of 1000 bits → Padded to 1024 bits

Step 2: Initialize Hash Values

Eight 64-bit initial hash values (H_0 to H_7) derived from square roots of first 8 primes:

```
H₀ = 6a09e667f3bcc908    H₄ = 510e527fade682d1
H₁ = bb67ae8584caa73b    H₅ = 9b05688c2b3e6c1f
H₂ = 3c6ef372fe94f82b    H₆ = 1f83d9abfb41bd6b
H₃ = a54ff53a5f1d36f1    H₇ = 5be0cd19137e2179
```

Step 3: Process Each 1024-bit Block

A. Prepare Message Schedule (80 words)

```
W₀ to W₁₅: First 16 words from message block (64 bits each)

W₁₆ to W₇₉: W_t = σ₁(W_{t-2}) + W_{t-7} + σ₀(W_{t-15}) + W_{t-16}
```

Where:

- $\sigma_0(x) = \text{ROTR}^1(x) \oplus \text{ROTR}^8(x) \oplus \text{SHR}^7(x)$
- $\sigma_1(x) = \text{ROTR}^{19}(x) \oplus \text{ROTR}^{61}(x) \oplus \text{SHR}^6(x)$

B. Initialize Working Variables

```
a = H₀, b = H₁, c = H₂, d = H₃,
e = H₄, f = H₅, g = H₆, h = H₇
```

C. 80 Rounds of Compression

For each round t (0 to 79):

```
T₁ = h + Σ₁(e) + Ch(e,f,g) + K_t + W_t
T₂ = Σ₀(a) + Maj(a,b,c)

h = g, g = f, f = e, e = d + T₁
d = c, c = b, b = a, a = T₁ + T₂
```

Functions:

- $\Sigma_0(a) = \text{ROTR}^{28}(a) \oplus \text{ROTR}^{34}(a) \oplus \text{ROTR}^{39}(a)$
- $\Sigma_1(e) = \text{ROTR}^{14}(e) \oplus \text{ROTR}^{18}(e) \oplus \text{ROTR}^{41}(e)$
- $\text{Ch}(e,f,g) = (e \wedge f) \oplus (\neg e \wedge g)$
- $\text{Maj}(a,b,c) = (a \wedge b) \oplus (a \wedge c) \oplus (b \wedge c)$
- **K_t** = Round constant (80 predefined values)

D. Update Hash Values

After 80 rounds:

$$H_0 = H_0 + a,\ H_1 = H_1 + b,\ H_2 = H_2 + c,\ H_3 = H_3 + d$$
$$H_4 = H_4 + e,\ H_5 = H_5 + f,\ H_6 = H_6 + g,\ H_7 = H_7 + h$$

Step 4: Generate Final Digest

After processing all blocks, concatenate:

$$\text{Message Digest} = H_0 \,||\, H_1 \,||\, H_2 \,||\, H_3 \,||\, H_4 \,||\, H_5 \,||\, H_6 \,||\, H_7$$

Output: 512-bit hash (128 hex characters)

Example

Input: "abc"

Output:

```
ddaf35a193617abacc417349ae20413112e6fa4e89a97ea20a9eeee64b55d39a
2192992a274fc1a836ba3c23a3feebbd454d4423643ce80e2a9ac94fa54ca49f
```

Key Properties

✓ Deterministic
✓ Fixed 512-bit output
✓ One-way (irreversible)
✓ Collision resistant
✓ Avalanche effect
✓ Security: 2^{256} collision resistance

Page No.
Date

Q. MD5 (Message Digest Algorithm)

Ans MD5 - Message Digest

Plaintext : 512 bits — 64 < (multiple of 512)

① Append Padding bits

② Append 64 bit representation

} PT length = Multiples of 512

PT: 512 - 64
= 448

③ Initiliaze the MD buffers (buffer - 32 bit, 4 buffers)
ABCD

Output of MD5 size: 128 bits

④ Process Message

⑤ Output

Not asked in GTU see detailed version in images

Q. Compare MD5 and SHA

MD5	SHA
1) MD length is 128 bits	1) Length is 160 bits
2) Speed is faster	2) Slower
3) No of iteration is 64	3) is 80
4) Buffer space is 128-bits	4) 160 bits
5) MD5 is vulnerable to Cryptanalytic attacks	5) SHA-1 appears not to be vulnerable
6) MD5 uses a little endian scheme	6) big endian scheme
7) Simple to Implement and do not need any large programs	7) Same
8) No limit on maximum message size	8) Maximum message size is $2^{64}-1$ bits

Why is SHA more secure than MD5?

Why SHA is More Secure than MD5

Both **MD5 (Message Digest 5)** and **SHA (Secure Hash Algorithm)** are cryptographic hash functions, but **SHA** (especially SHA-2 and SHA-3 families) is considered **much more secure** than MD5.

---

1. Hash Length (Output Size)

Algorithm	Digest Size	Security Level
MD5	128 bits	Low
SHA-1	160 bits	Moderate
SHA-256 / SHA-512	256 / 512 bits	Very High

➡️ **Longer hash = stronger security,**
because it's exponentially harder for attackers to find collisions or reverse the hash.

---

2. Collision Resistance

- **MD5:** Collisions can be found easily (two different inputs producing the same hash).
  → Proven to be **broken** in 2004.
- **SHA (SHA-2/SHA-3):** No practical collision attacks known.
  → Much higher **collision resistance**.

3. Pre-image and Second Pre-image Resistance

- **MD5:** Weak — attackers can generate messages with the same MD5 hash as a genuine file.
- **SHA:** Strong — designed to make it computationally infeasible to reverse or find a second input with the same hash.

---

4. Algorithm Design

- **MD5:** Produces only a 128-bit hash using 64 rounds of simpler operations.
- **SHA-2 (e.g., SHA-512):** Uses 80 rounds of complex bitwise operations, modular additions, and rotations — making it **harder to break**.

---

5. Real-World Use

- **MD5:** No longer used for security purposes — only for checksums or non-secure integrity checks.
- **SHA-2 / SHA-3:** Used in **SSL/TLS**, **digital signatures**, **blockchain**, and **password hashing**.

✅ In Summary:

Feature	MD5	SHA (SHA-2/SHA-3)
Hash size	128 bits	256–512 bits
Collision resistance	Weak	Strong
Security	Broken	Secure
Usage	Obsolete	Used in modern cryptography

Ch-6 MAC

Q-1 What is MAC? State the main difference between MAC & Hash Function

Ans → Message authentication is a mechanism or service used to verify the integrity of a message. Message integrity guarantees that the message has not been changed.

→ A MAC algorithm, sometimes called a keyed hash function accepts as input a secret key and an arbitrary-length message to be authenticated, and outputs a MAC.

→ The MAC value protects both a message's data integrity as well as its authenticity, by allowing verifiers to detect any changes to the message content.

	Basis	MAC	Hash Function
1)	Key Usage	Uses a secret key along with message	No key used; works only on the message
2)	Purpose	Ensures both integrity and authenticity	Ensures only integrity (detects changes)
3)	Security Dependance	Security depends on the secret key	Security depends on hash algo. properties
4)	Output	Fixed-length code (eg 128 or 256 bits)	Fixed-length hash value (eg 128, 256, 512 bits)
5)	Example Algorithms	HMAC, CMAC, CBC-MAC	MD5, SHA-1, SHA-256
6)	Used in	Secure communication protocols (TLS, IPsec)	Digital signatures, checksums, data integrity checks

Q-2 What is the difference between MAC and message digest?

Ans

	Basis	MAC	Message Digest
1)	Defination	A short code generated using a secret key and a message to ensure integrity & authenticity	A fixed-size output generated from a message using a hash-function, ensuring integrity also
2)	Key Used	Yes	No
3)	Purpose	Ensures both message integrity and sender authentication	Ensures only message integrity
4)	Security Dependence	The secret key and the MAC algorithm	The hash function's properties (one-way, collision resistance)
5)	Example Algorithms	HMAC, CMAC CBC-MAC	MD5, SHA-1, SHA-256, SHA-512
6)	Used In	Secure Communication	Digital signatures, file verification, password hashing
7)	Vulnerability	Secure as long as the key remains secret	Can be forged or tampered since no secret key is used.

Q-3 Explain the types of message attacks are addressed by message authentication?

Ans

	Types of Attack	Description
1)	Content Modification	Altering the data in message
2)	Sequence Modification	Changing message order
3)	Replay Attack	Re-sending old valid messages
4)	Masquerade Attack	Impersonating legitimate sender
5)	Delay Attack	Delaying valid messages

Q-4 Difference between Authentication v/s Authorization

Ans

Basis	Authentication	Authorization
1) Definition	Process of verifying the identity of a user or system	Process of granting or denying access to resources after authentication
2) Purpose	Confirms who you are	Determines what you can do
3) Performed When	Always performed before authorization	Performed after successful authentication
4) Involves	User Identification and credential Verification	Access control and permission management
5) Output	Confirms the user's identity	Grants specific rights or privileges
6) Security Focus	Validates the identity of the user	Controls access to system resources

Q-5 Explain with the diagrams Basic Uses of Message Authentication Code (MAC).

Ans

1) Explain NIST digital signature algorithm.

7.5 NIST Digital Signature Algorithm

- The National Institute of Standards and Technology (NIST) requests comments on Federal Information Processing Standard. The NIST has announced proposed changes to a standard that specifies how to implement digital signatures, which

can be used to ensure the integrity of electronic documents, such as wills and contracts, as well as the identity of the signer.

- These proposed changes to the Federal Information Processing Standard (FIPS) 186-3, known as the Digital Signature Standard, were posted for public comment on April 10, 2012. First published in 1994 and revised several times since then, the standard provides a means of guaranteeing authenticity in the digital world by means of operations based on complex math that are all but impossible to "forge". Updates to the standard are still necessary as technology changes.
- A Digital Signature is a function provided by Public Key Infrastructure (PKI). The process entails transforming a message or data and some secret information held by the sender into a tag called a signature. It provides proof of the source and verification of the integrity of the data.
- The sender generates a digital signature using his/her private key. The recipient verifies the sender's identity using the sender's public key.
- The purpose of a digital signature is to provide a means for an entity to bind its identity to data, and to detect unauthorized modifications to data.
- There are three algorithms suitable for digital signature generation and verification:
  1. Digital Signature Algorithm (DSA)
  2. Rivest-Shamir-Adleman, a reversible Digital Signature Algorithm (RSA)
  3. Elliptic Curve Digital Signature Algorithm (ECDSA)

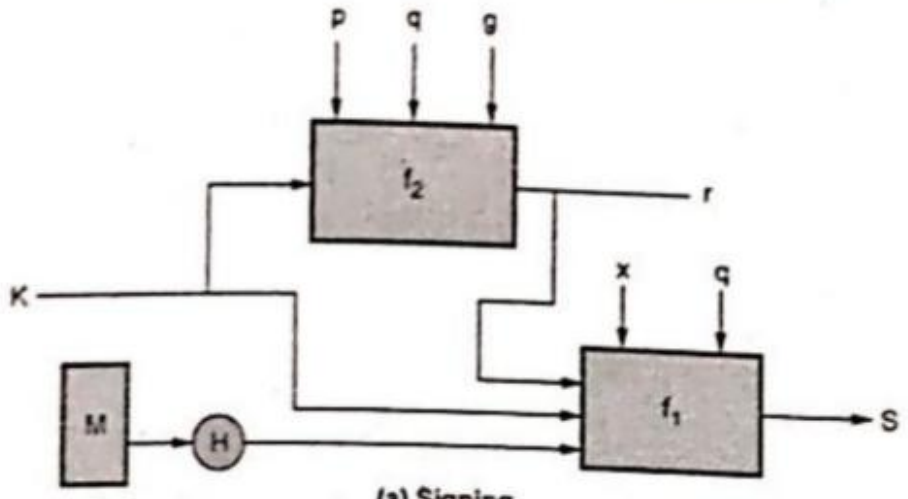
Benefits:

1. Digital signatures eliminate the need for transmitting passwords for authentication, which reduces the threat of their compromise
2. Using a private key to generate digital signatures for authentication prevents an attacker from using the same information to masquerade as another entity and authenticate repeatedly.
3. Digital signatures provide security for electronic mail, Electronic Funds Transfer (EFT), Electronic Data Interchange (EDI), software distribution, data storage, and other applications that require data integrity assurance and data origin authentication.

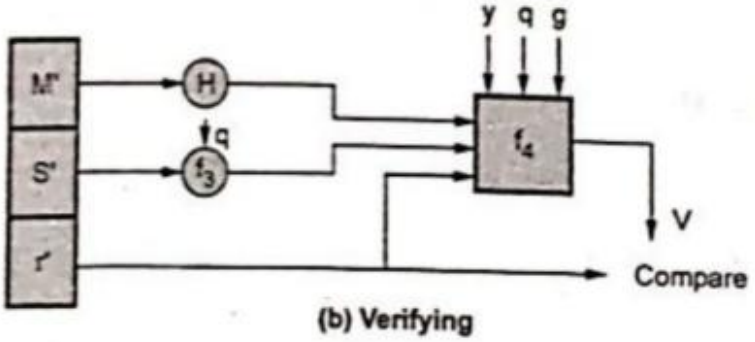
2) Explain with neat diagram Digital signature algorithm.

7.1.4 Digital Signature Algorithm

- There are three parameters that are public and can be common to a group of users. **Prime number q** is chosen and it is **160-bit**. A **prime number p** is selected with a length between **512** and **1024 bits** such that q divides (P – 1).
- g is chosen to be of the form $h^{(P-1)/q}$ mod p where h is an integer between 1 and (P – 1)
- With these number, user selects a private key and generate a public key. The private key x must be a number from 1 to (q – 1) and should be chosen randomly or pseudorandomly.
- The public key is calculated from the private key as $y = g^x$ mod p.
- To create a signature, a user calculates two quantities, **rands**, that are functions of
  i) Public key components (p, q, g)
  ii) User's private key (x)
  iii) Hash code of the message H(M)
  iv) An additional integer (K)
- **At the receiving end**, verification is performed. The receiver generates a quantity V that is a function of the public key components, the sender's public key and the hash code of the incoming message. If this quantity matches the r components of the signature, then the signature is validated.
- Fig. 7.1.3 shows the functions of signing and verifying.



(a) Signing



(b) Verifying

Fig. 7.1.3 Signing and verifying

3) Explain schnorr Algorithm For digital signature.

- Schnorr signature scheme is based on **discrete logarithms**.
- It minimizes the message-dependent computation required to generate a signature.
- Most computation is done **before knowing the message**, making it very fast.
- It uses a prime modulus **p**, and **p−1** has a large prime factor **q**.
- The common public parameters are: **p, q, and α (a generator)**.

⭐ 2. Key Generation (Generation of Private/Public Key Pair)

(Write steps exactly like in screenshot – 2 marks)

1. **Choose primes p and q**
   such that **q is a prime factor of (p−1)**.
2. **Choose an integer α**, such that
   α^q ≡ 1 (mod p)
   The values **α, p, q** form the global public key usable by all users.
3. **Choose a random integer s** with
   0 < s < q
   This is the **user's private key**.
4. **Compute v = α^(−s) mod p**
   (or equivalently α^(q−s) mod p)
   This is the **user's public key**.

So,

✔ **Private key = s**
✔ **Public key = v**

⭐ 3. Signature Generation (2 marks)

A user with private key **s** and public key **v** signs a message **M** as follows:

1. Choose a random integer **r** with
   0 < r < q
2. Compute
   x = α^r mod p
   (This is preprocessing and independent of message.)
3. Concatenate the message M with x and compute:
   e = H(M || x)
   (Hash of message and x)
4. Compute signature component
   y = (r + s·e) mod q

Final signature is the pair **(e, y)**.

⭐ 4. Signature Verification (2 marks)

Any other user can verify the signature (e, y) as follows:

1. Compute
   x' = α^y · v^e mod p
2. Compute hash:
   e' = H(M || x')
3. If
   e' = e,
   then the signature is **valid.**

⭐ 5. Summary (Write this at the end for neat marks)

- Schnorr signature scheme uses **p, q, α** as public parameters.
- **Private key = s, Public key = v = α^(−s) mod p.**
- To sign:
  - Pick r, compute x = α^r mod p
  - e = H(M||x)
  - y = (r + s·e) mod q
- Signature: **(e, y)**
- To verify:
  - x' = α^y · v^e mod p
  - e' = H(M||x')
  - If e' = e → signature valid.

4) Name the four key steps in the creations of the digital certificate

⭐ 1) Four Key Steps in the Creation of a Digital Certificate (GTU Answer)

The creation of a digital certificate involves the following **four key steps:**

Step 1: Generate a Key Pair

The organization or user first generates a **public key and private key.**
The **private key** is kept secure, and the **public key** will be included in the certificate.

Step 2: Prepare Certificate Information

The entity prepares all necessary identification details, such as:

- Name and address (Distinguished Name – DN)
- Organization details
- Email, domain, or other identity data
- The public key that needs to be certified

This information is sent to the Certification Authority (CA).

Step 3: Certification Authority Verifies the Identity

The CA validates and confirms:

- The identity of the requesting organization/person
- That the submitted public key really belongs to them

Once the identity is verified, the CA proceeds to create the certificate.

Step 4: CA Creates and Digitally Signs the Certificate

The CA:

1. Bundles the public key + identity information
2. Adds issue date, expiry date, and serial number
3. **Digitally signs the certificate using its own private key**

This creates the final **Digital Certificate** (X.509 certificate).

5) What are the common causes of revoking a digital certificate.

⭐ Common Causes of Revoking a Digital Certificate (Easy + Detailed – 3 Marks)

A digital certificate may be **revoked** (cancelled before expiry) when it is no longer trustworthy.
The most common causes are:

---

1. Private Key Compromise

If the **private key is stolen, leaked, hacked, or misused**, the certificate becomes unsafe.
Anyone who gets the private key can:

- Impersonate the user
- Create fake signatures
- Decrypt confidential data

So the certificate must be immediately revoked.

---

2. Change in User's Information

If the identity information in the certificate changes, the old certificate becomes invalid.
Examples:

- Name change
- Organization change
- Department or role change
- Domain name change

Because the certificate contains identity details, any change makes it outdated.

---

3. Employee Leaves the Organization

If an employee leaves a company or loses authorized access, their certificate should be revoked to prevent misuse.

4. Certificate Misuse or Unauthorized Use

If the certificate is used for illegal or unintended purposes, the CA revokes it.
Example: using a code-signing certificate to sign malware.

---

5. CA Error or Incorrect Issuance

If the **Certification Authority issues a certificate mistakenly**, with wrong details or to an unauthorized person, it must be revoked.

---

6. Certificate Expired or No Longer Needed

If the certificate has expired early, or the owner no longer needs it, it may be revoked instead of waiting for automatic expiry.

6) Discuss the security of the digital signature.

A **digital signature** provides strong security for electronic documents.
Its security is based on **public-key cryptography**, **hash functions**, and **private key protection**.
The following security features make digital signatures trustworthy:

---

1. Authentication

A digital signature confirms **who signed the message**.
Only the owner of the private key can create the signature, and the receiver verifies it using the sender's public key.

- ✔ Ensures that the signer is real and not an impersonator.
- ✔ Protects against identity fraud.

---

2. Integrity

Digital signatures use cryptographic **hash functions**.
If even one character of the message is modified:

- The hash value changes
- The signature becomes invalid

Thus, the receiver immediately knows that the message has been **tampered with**.

- ✔ Guarantees the message has not been changed during transmission.

---

3. Non-repudiation

Once a person signs a message with their private key:

- They **cannot deny** signing it later
- The digital signature becomes legal proof of their action

This prevents the signer from claiming "I didn't send it."

- ✔ Provides legal and evidential value.

4. Confidentiality (Indirectly)

Although encryption provides confidentiality, digital signatures **support secure communication** by ensuring:

- The sender is genuine
- The content is intact
- The message has not been changed

In many systems, digital signatures are used together with encryption to achieve confidentiality.

---

5. Protection Against Forgery

Modern digital signatures use strong mathematical problems such as:

- RSA
- DSA
- ECDSA
- Schnorr

These algorithms make it **computationally impossible** to forge a signature without the private key.

✔ Prevents attackers from generating fake signatures.

7) what is replay attack? How can we avoid this attack using digital signature.

⭐ 1. What is a Replay Attack? (3 Marks)

A **Replay Attack** is a type of network attack in which an attacker:

- **Intercepts a valid message**
- **Records it**
- **Resends (replays) it later**

to trick the receiver into performing the same action again.

In this attack, the message is **not altered**.
Instead, it is **captured and resent** to gain unauthorized access or repeat a transaction.

⭐ Example of Replay Attack:

Suppose a user sends:

> "Transfer ₹10,000 to Account X"
> (digitally signed)

If an attacker captures this message and sends it again later,
the bank might perform the same transfer **again**, even though the message is old.

So the attacker **replays** the valid message to commit fraud.

⭐ Why Replay Attacks Are Dangerous?

- They can lead to **duplicate payments or transactions**
- Can allow **re-login without password**
- Can **bypass authentication**
- Do not require breaking encryption or digital signatures

The attacker only needs to **capture and resend** a valid message.

⭐ 2. How Digital Signatures Help Prevent Replay Attacks? (4 Marks)

Digital Signatures alone **prove identity and integrity**,
but **do NOT automatically stop replay attacks** unless combined with additional techniques.

Below are the methods used with digital signatures to prevent replay attacks:

⭐ A. Use of Timestamps

When a message is digitally signed, it includes a **timestamp**:

- The receiver checks whether the timestamp is **recent**
- Old messages are rejected

So even if an attacker replays a signed message later,
the timestamp will show it is **expired**, and it will not be accepted.

- ✔ Prevents reuse of old messages
- ✔ Ensures freshness

⭐ B. Use of Nonces (Random Numbers)

A **nonce** is a random number used **only once**.

Process:

1. Receiver sends a random **nonce** to the sender
2. Sender signs the message **along with the nonce**
3. Receiver verifies the signature + nonce

If an attacker replays the message:

- The nonce will not match
- Signature verification will fail

- ✔ Prevents attacker from reusing old messages
- ✔ Ensures every message is unique

⭐ C. Session Tokens / Unique Identifiers

Each signed message contains:

- A unique session ID
- Transaction ID
- Sequence number

If the attacker tries to replay the message:

- The system detects **duplicate IDs**
- Rejects the message

✔ Ensures every transaction is accepted only once

⭐ D. Digital Signature Ensures Integrity

Digital signatures guarantee that:

- Message was not modified
- Sender is genuine
- Data is authenticated

This ensures that **only fresh, signed messages** from the real user are accepted.

IS CHAP 8 NOTES

1. Explain mutual authentication using symmetric key cryptography. (3m)

Mutual authentication is a security process in which **both communicating parties verify each other's identity** before starting communication. In **symmetric key cryptography**, this is done using a **shared secret key** known only to both parties.

Working Principle:

Let two users be **A (client)** and **B (server)** who share a secret key **K**.

1. **A → B : Encrypted Challenge**
   A sends an authentication request with a random number (nonce) encrypted using key K.
2. **B → A : Response + Challenge**
   B decrypts the message using the same key K, proves its identity by responding correctly, and sends its own challenge encrypted with K.
3. **A → B : Final Verification**
   A decrypts B's challenge and sends back the correct response, confirming its identity.

If both responses are correct, **mutual authentication is successfully achieved**.

Example:

- A sends: E(K, NA)
- B replies: E(K, NA + NB)
- A responds: E(K, NB)

Where:

- **K** = shared secret key
- **NA, NB** = random numbers (nonces)

2. Explain one way authentication using symmetric key cryptography. (3m)

Definition:
One-way authentication is a process in which **only one party (server or sender) is authenticated**, while the other party is not verified. In **symmetric key cryptography**, both parties share a **common secret key**, which is used to verify the identity of only one side.

Working Process:

1. The client (A) sends a request to the server (B).
2. The server encrypts a message (or a random number) using the **shared secret key** and sends it to the client.
3. The client decrypts the message using the same secret key.
4. If the decrypted message is correct, the server is considered **authenticated**.

Example:

- Client sends: "Hello"
- Server replies: Encrypted nonce = E(K, R)
- Client decrypts: D(K, E(K, R)) = R
  If correct, the client trusts the server.

Key Points:

- Only the **server is verified**, not the client.
- Uses a **single shared secret key**.
- Simple but **less secure** compared to mutual authentication.

3. Compare link encryption and end to end encryption. (4m)

Parameter	Link Encryption	End-to-End Encryption
Definition	Encrypts data on each communication link separately. Data is decrypted and re-encrypted at every intermediate node.	Encrypts data at the sender and decrypts it only at the final receiver.
Security Level	Less secure because data is exposed at intermediate nodes during decryption.	More secure because data remains encrypted throughout the entire transmission.
Encryption Points	Encryption and decryption occur at **every hop** (router or switch).	Encryption occurs only at **sender**, decryption only at **receiver**.
Key Management	Different keys are used for each link.	Same encryption key is shared between sender and receiver.
Protection Scope	Protects data on individual links only.	Protects data from source to destination completely.
Example	VPN tunnels between network nodes.	HTTPS, WhatsApp end-to-end encryption.

4. What is the role of Key Distribution Centre? Give the several techniques for the distribution of public keys. (4m)

Role of Key Distribution Centre (KDC):

A **Key Distribution Centre (KDC)** is a trusted third party responsible for **managing and securely distributing cryptographic keys** between communicating users in a network.

Main Roles:

1. **Authentication:** Verifies the identity of users before allowing communication.
2. **Key Generation:** Generates secret session keys for secure communication.
3. **Key Distribution:** Safely distributes these keys to legitimate users.
4. **Secure Communication:** Ensures only authorized users can communicate securely.

Techniques for Distribution of Public Keys:

1. **Public Announcement:**
   Users publish their public keys openly (e.g., on websites or directories).
   → Simple but vulnerable to fake key attacks.
2. **Publicly Available Directory:**
   A trusted directory stores public keys with user identities.
   Users retrieve keys from this verified source.
3. **Public Key Authority:**
   A trusted authority provides public keys upon request and verifies their authenticity.
4. **Public Key Certificates:**
   Uses **Digital Certificates** issued by a Certificate Authority (CA) that binds a public key with the user's identity.

5. Give the difference Session Key and Master Key. (4m)

Point	Session Key	Master Key
Definition	A session key is a temporary key used to encrypt data during a single communication session.	A master key is a long-term key used to generate or protect other keys, including session keys.
Purpose	Used for **actual data encryption and decryption** during communication.	Used for **key management and key distribution.**
Validity	Exists only for the duration of one session and then is discarded.	Remains valid for a longer time and is reused securely.
Security Level	Provides security for one-time communication.	Provides overall security for the system by protecting session keys.
Example	HTTPS uses a session key for one secure connection.	A Key Distribution Center (KDC) uses a master key to issue session keys.
Risk Impact	If compromised, only one session is affected.	If compromised, many sessions may become insecure.

6. Draw X.509 certificate format. (4m)

8.2 X.509 Certificates

GTU : Dec.-11, May-12, Winter-15

- X.509 is part of X.500 recommandations for directory service i.e. set of servers which maintains a database of information about users and other attributes.
- X.509 defines authentication services e.g. certificate structure and authentication protocols. Also X.509 also defines alternative authentication protocols base on use

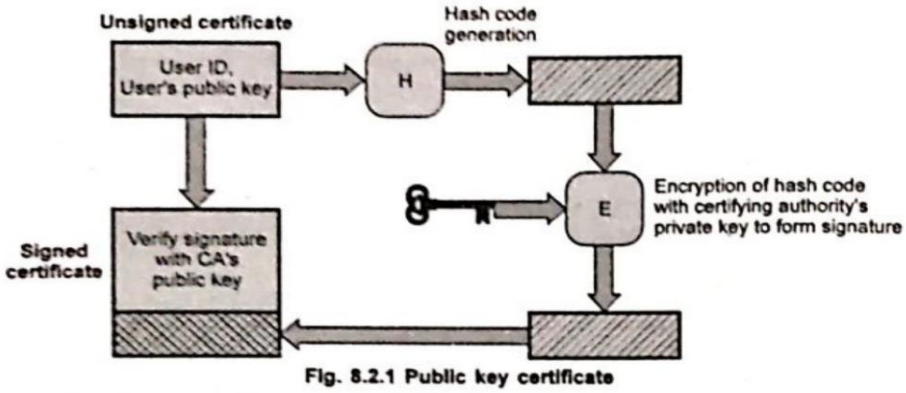
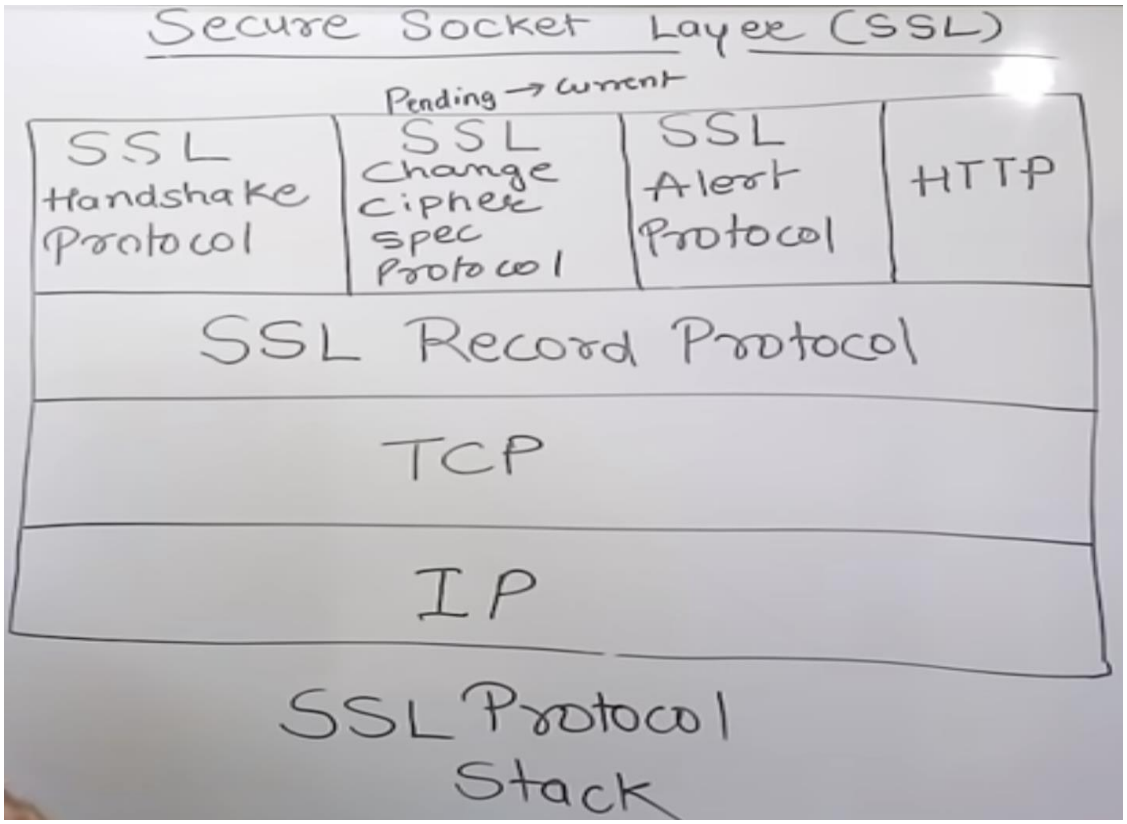


Fig. 8.2.1 Public key certificate

of public-key certificates. The X.509 certificate format is emplied in S/MIME, IP security, SET and SSL/TLS.

- X.509 standard uses RSA algorithm and hash function for digital signature. Fig. 8.2.1 shows generation of public key certificate.

7. Explain SSL Architecture with neat diagram. (7m)



Components of SSL Architecture

1) SSL Record Protocol

- It is the core layer of SSL.
- Provides basic security services to higher protocols.
- Functions:
  - Fragmentation of data
  - Compression
  - Encryption
  - Message Integrity using MAC
- It receives data from upper SSL protocols and sends it securely over TCP.

---

2) SSL Handshake Protocol

- Used to establish a secure connection before data transfer.
- Performs:
  - Server and client authentication
  - Negotiation of encryption algorithm and hashing function
  - Generation and exchange of session keys
- It ensures that both parties agree on security parameters.

3) SSL Change Cipher Spec Protocol

- Used to indicate that the communication will now use the newly agreed encryption settings.
- It changes from **pending state to current state**.
- It is a one-byte message that activates the selected cipher suite.

---

4) SSL Alert Protocol

- Used to convey error messages and warnings.
- Types of alerts:
  - Warning alerts (e.g., close_notify)
  - Fatal alerts (terminates connection)
- Ensures proper termination and error handling.

---

5) HTTP over SSL

- Application layer protocols like HTTP use SSL to provide secure communication.
- When HTTP runs over SSL, it becomes **HTTPS**.

Working of SSL Architecture (Flow)

1. Client initiates a secure connection.
2. SSL Handshake authenticates server and negotiates cryptographic methods.
3. Session keys are generated.
4. Change Cipher Spec activates encryption.
5. Data is transferred securely via SSL Record Protocol.
6. Any issues are reported using Alert Protocol.

Advantages of SSL

- Data confidentiality through encryption
- Authentication of server
- Data integrity
- Protection from eavesdropping and man-in-the-middle attacks

8. What do you mean by key-distribution? Give at least one method for key distribution with proper illustration. (7m)

Meaning of Key Distribution

For **symmetric encryption** to work, both communicating parties must share the **same secret key**, and this key must be protected from unauthorized access.
Key distribution refers to the **secure process of delivering a cryptographic key to two parties who wish to exchange data without allowing others to see the key**.

Thus, the main objective of key distribution is:

- Secure sharing of secret keys
- Preventing key exposure
- Enabling confidential communication

Methods of Key Distribution

For two users A and B, key distribution can be achieved by the following methods:

1. **Physical delivery by User A**
   User A selects a key and physically delivers it to User B.
2. **Physical delivery by Third Party**
   A trusted third party selects the key and delivers it to both A and B.
3. **Using old shared key**
   If A and B previously shared a key, a new key can be sent encrypted using the old key.
4. **Key Distribution Centre (KDC) Method** ✅ (Most secure and widely used)
   Both users have encrypted links with a trusted third party called KDC, which distributes the session key securely.

Key Distribution using KDC (Illustrated Method)

Concept:

- User A shares master key Ka with KDC
- User B shares master key Kb with KDC
- KDC generates a session key Ks

Steps (as per given scenario):

1. A sends request to KDC including IDA, IDB and nonce N1.
2. KDC replies with encrypted message using Ka containing:
   - Session Key Ks
   - Ticket for B encrypted with Kb
3. A forwards the ticket to B.
4. B challenges A using nonce N2.
5. A responds using Ks, proving possession of session key.

After authentication, secure data transfer begins.

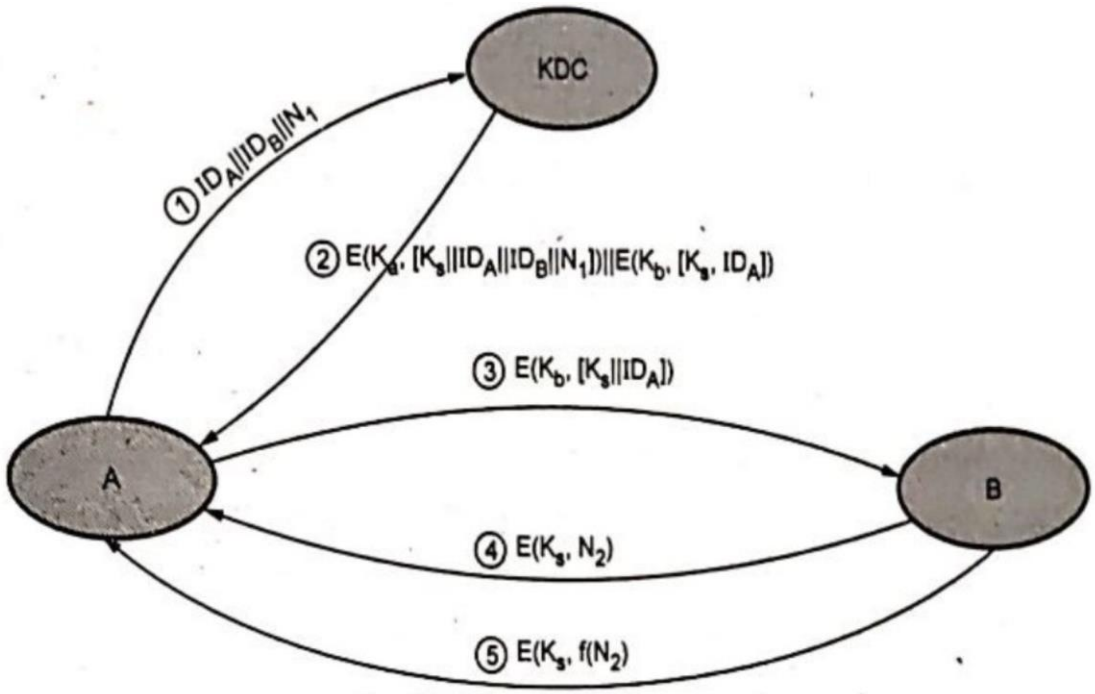


Fig. 8.1.9 Key distribution scenario

Key Hierarchy Concept (From Diagram)

Key distribution is based on a hierarchy of keys:

- **Master Key**
  Long-term key shared with KDC (less frequently used)
- **Session Key**
  Temporary key for each session
- Data is protected using:
  - Session Key → Cryptographic protection
  - Master Key → Non-cryptographic protection

Advantages of KDC Method

- High security
- Centralized key control
- Avoids direct key transmission
- Prevents interception
- Supports authentication

9. Write a short note on Pretty Good Privacy(PGP). (4M)

Pretty Good Privacy (PGP)

Pretty Good Privacy (PGP) is a widely used security program that provides **cryptographic privacy and authentication** for data communication. It is mainly used to secure **emails, files, and digital documents** over insecure networks.

PGP uses a combination of:

- **Symmetric encryption** (for fast data encryption)
- **Public key cryptography** (for key exchange)
- **Hash functions** (for integrity)

Main Functions of PGP

1. **Confidentiality**
   Data is encrypted using a session key, and the session key is encrypted using the receiver's public key.
2. **Authentication**
   Digital signatures verify the sender's identity.
3. **Integrity**
   Hashing ensures the message is not altered during transmission.
4. **Non-repudiation**
   Sender cannot deny sending the message.

Working of PGP (Brief)

1. Message is hashed → Digital signature created.
2. Message is compressed.
3. Session key encrypts the message.
4. Session key is encrypted using receiver's public key.
5. Encrypted message is sent securely.

Advantages of PGP

- Strong security
- Ensures privacy and authenticity
- Efficient hybrid encryption
- Widely used for secure email communication

10. List use of public key cryptography. (3m)

Public Key Cryptography is used to provide secure communication and verification in computer networks. Its main uses are:

1. **Encryption / Confidentiality**
   Public key is used to encrypt data so that only the intended receiver can decrypt it using their private key.
2. **Digital Signatures**
   It is used to create digital signatures for authentication and to verify the sender's identity.
3. **Key Exchange**
   Securely distributes session keys for symmetric encryption.
4. **Authentication**
   Verifies the identity of users in secure systems such as SSL/TLS and secure emails.
5. **Secure Email Communication**
   Used in systems like PGP and S/MIME to protect email content.
6. **Integrity Checking**
   Ensures that data has not been modified during transmission.

11. List requirements of public key exchange algorithm. (4m)

A public key exchange (public key cryptography) algorithm must satisfy the following requirements:

1. **Feasible Key Pair Generation**
   It must be computationally easy to generate a pair of keys: a public key and a private key.
2. **Easy Encryption and Decryption**
   It should be easy for the sender to encrypt data using the public key and for the receiver to decrypt it using the private key.
3. **Infeasible to Derive Private Key**
   It must be computationally impossible to determine the private key from the public key.
4. **Infeasible to Decrypt Without Private Key**
   It should be extremely difficult to recover the original message without knowing the private key.
5. **Both Keys Must Be Inverses**
   The public and private keys should be mathematically related such that what one key encrypts, the other can decrypt.
6. **Security Against Attacks**
   The algorithm should be resistant to brute-force and cryptanalytic attacks.

12. How public key cryptography can be used for digital signature? Explain (4m)

Digital Signature Using Public Key Cryptography

Public key cryptography provides a secure method to create **digital signatures**, which are used to verify the **authenticity, integrity, and non-repudiation** of a message.

A digital signature ensures that:

- The message is sent by the genuine sender (authentication)
- The message has not been altered (integrity)
- The sender cannot deny sending it (non-repudiation)

How Public Key Cryptography Works for Digital Signatures

Step 1: Message Hashing

The sender first generates a **hash (message digest)** of the original message using a hash function (e.g., SHA-256).
This hash uniquely represents the message.

Step 2: Encrypting the Hash with Sender's Private Key

The sender encrypts the hash value using **their private key**.
This encrypted hash is the **digital signature**.

Step 3: Sending Message + Signature

The sender sends:

- Original message
- Digital signature

to the receiver.

Verification Process (Receiver Side)

Step 4: Decrypt Signature Using Sender's Public Key

The receiver decrypts the digital signature using the **sender's public key**, recovering the original hash created by the sender.

Step 5: Hash the Received Message

The receiver generates a new hash of the received message using the same hash function.

Step 6: Compare Both Hashes

- If both hash values match → Message is **authentic and unmodified**.
- If they don't match → Message is **altered or forged**.

CH-9

1. **What is the role of AS and TGS in Kerberos?(3 marks)**

✅ Role of AS and TGS in Kerberos (10 Easy Points)

Authentication Server (AS)

1. **Checks user identity** when they log in using their username and password.
2. If the password is correct, the AS sends a **Ticket Granting Ticket (TGT)**.
3. The TGT proves the user is genuine without sending the password again.
4. AS also creates a **session key** shared between the user and TGS.
5. AS keeps the user's password safe because it is **never transmitted on the network**.

Ticket Granting Server (TGS)

6. The TGS receives the **TGT** from the user whenever they request a service.
7. It verifies the TGT to confirm the user was already authenticated by the AS.
8. The TGS then issues a **Service Ticket** for the specific service (like file server, email server, etc.).
9. This Service Ticket is encrypted with the **service server's secret key**, so only that server can read it.
10. TGS ensures the user can access multiple services **without re-entering the password** (single sign-on).

⭐ Simple Summary

- **AS = First Login + Gives TGT**
- **TGS = Gives Access to Other Services Using TGT**
- Together, they make Kerberos **secure and password-free after the first login**.

2. **Explain authentication mechanism of Kerberos.**

✅ Authentication Mechanism of Kerberos (Easy Explanation)

Kerberos uses a **ticket-based** system to authenticate users securely without sending passwords over the network.
It works in **three main steps**:

---

1. User Login → AS Authentication

- The user enters their **username and password**.
- Password is converted into a key (hashed).
- This is sent to the **Authentication Server (AS)**.
- AS verifies the user and sends back a **Ticket Granting Ticket (TGT)** + a session key.
- The TGT is encrypted with the **TGS's secret key**, so only TGS can read it.

➡️ **Purpose:** User is authenticated once, and password is never sent again.

---

2. Requesting Service → TGS Authorization

- When the user wants to access a service (file server, mail server, etc.),
  they send the **TGT** + service request to the **Ticket Granting Server (TGS)**.
- TGS verifies the TGT and checks if the user is allowed to access that service.
- If yes, TGS issues a **Service Ticket**.

➡️ **Purpose:** User gets permission to use specific services.

---

3. Accessing the Service → Server Authentication

- The user sends the **Service Ticket** to the **Application Server** (e.g., file server).
- The server decrypts the ticket and verifies the session key.
- If everything matches, the server allows the user to access the service.

➡️ **Purpose:** Mutual authentication — both server and user confirm each other.

⭐ Simple Summary (Exam Perfect)

1. **AS authenticates the user** and issues a Ticket Granting Ticket (TGT).
2. **TGS authorizes access** to services and gives Service Tickets.
3. **Service Server provides access** only after verifying the Service Ticket.
4. Password is **never sent across the network**, improving security.
5. User logs in **once** and accesses all services securely (Single Sign-On).

3.Discuss the working of KERBEROS authentication protocol.

⭐ Working of Kerberos Authentication Protocol (7 Marks)

Kerberos is a **ticket-based** network authentication protocol that provides secure login and access to services in a distributed environment. It uses **symmetric key cryptography**, **trusted servers**, and **time-stamped tickets** to ensure secure communication.
The working involves **three main phases**: Authentication, Ticket Granting, and Service Access.

---

1. INITIAL AUTHENTICATION (User ↔ Authentication Server – AS)

1. The user enters their **username and password** on the client machine.
2. The client sends the **username** to the Authentication Server (AS).
3. AS looks up the user in its database.
4. If valid, AS creates a **Ticket Granting Ticket (TGT)** and a **session key**.
5. The TGT is encrypted with the **TGS's secret key**, while the session key is encrypted with the **user's secret key** (derived from their password).
6. The user decrypts the session key using their password.
7. From this point, the user's password is **never sent again**.

Result: User is authenticated and now holds a TGT.

---

2. OBTAINING SERVICE TICKET (Client ↔ Ticket Granting Server – TGS)

1. When the user wants to access a network service (file server, email server, etc.), the client sends:
   - The **TGT**
   - A request for the specific service
   - An authenticator encrypted with the session key
2. TGS verifies the TGT and the authenticator.
3. If valid, TGS creates a **Service Ticket** + a **client–server session key**.
4. The Service Ticket is encrypted with the **Service Server's secret key**.
5. The client receives the Service Ticket and stores it.

Result: User now holds a Service Ticket for the service they want.

3. ACCESSING THE SERVICE (Client ↔ Application Server)

1. The client sends the **Service Ticket** and a fresh authenticator to the application server.
2. The server decrypts the Service Ticket using its secret key.
3. It verifies the authenticator and checks the timestamp.
4. If valid, the server sends a confirmation message (optional mutual authentication).
5. The client and server now communicate securely using the **session key**.

Result: User gains access to the requested service securely.

---

4. KEY FEATURES OF KERBEROS

1. **Single Sign-On:** Login once → access all services.
2. **Mutual Authentication:** Both client and server validate each other.
3. **No Password Transmission:** Password is never sent in plaintext on the network.
4. **Time-Stamps:** Prevent replay attacks.
5. **Ticket System:** Eliminates repeated password verification.
6. **Symmetric Key Cryptography:** Faster and efficient.

⭐ Summary (7 Marks in 5 Lines)

- Kerberos uses **AS**, **TGS**, and **Service Server** to authenticate users.
- AS verifies the user and issues a **TGT**.
- TGS uses the TGT to give **Service Tickets** for specific services.
- Service servers verify the Service Ticket and allow access using session keys.
- Kerberos ensures secure, passwordless, and time-controlled authentication.