full one-shot videos on :**JK Coding Pathshala YouTube channel**

JK Coding Pathshala

https://youtube.com/@jayeshkande9215?feature=shared

Unit V	CRYPTOGRAPHIC ALGORITHM	(06 hrs)
Mathematical preliminaries: Groups, Rings, Fields, Prime numbers, Symmetric key algorithms: Data Encryption Standards, Advanced Encryption Standard, **Public Key Encryption and Hash function:** RSA Digital signatures, Digital Certificates and Public Key Infrastructure: Private Key Management, Diffie Hellman key exchange, The PKIX Model		

Q5) a) Define symmetric key encryption? Explain DES in detail. **[9]**

b) Elaborate Digital signatures, Digital Certificates. **[9]**

OR

P.T.O.

Q6) a) Write a note on RSA algorithm. **[9]**

b) What is private key? Explain Diffie-Hellman key exchange algorithm. **[9]**

Q5) a) Explain difference between symmetric and asymmetric key cryptography. **[9]**

b) Explain Data Encryption Standards with diagram. **[9]**

OR

P.T.O.

Q6) a) Explain following terms: **[9]**

i) Groups

ii) Rings

iii) Prime Numbers

b) Explain diffic Hellman key exchange in detail. **[9]**

Q5) a) Define and list various computer network security mechanisms Also write short notes on the following terms **[9]**

i) Encryption

ii) Decryption

b) Define and explain digital signature. What are the applications of digital signature. **[9]**

OR

Q6) a) Define the following mathematical preliminaries and also explain the following in brief **[9]**

i) Prime numbers

ii) Group

b) What is Cryptography? also explain the following terms **[9]**

i) Public key encryption

ii) Hash function

Q5) a) Explain Data Encryption Standard Algorithm in detail with suitable diagram. **[9]**

b) Explain Diffie-Hellman key exchange algorithm. **[9]**

OR

Q6) a) Explain Private Key Management. **[9]**

b) Explain following terms. **[9]**

i) PKIX Model

ii) Digital Signature

iii) Digital Certificate

Q5) a) Write a short notes on [9]

i) Encryption

ii) Decryption

b) Explain RSA in details [9]

OR

P.T.O.

Q6) a) What is digitel signature? What are application of digital signature [9]

b) Explain following terms. [9]

i) Cryptograply

ii) Symmetric key Cryptograply

iii) Asymmetric key Cryptograply

Mathematical preliminaries: Groups, Rings, Fields, Prime numbers

Group

A group **G**, denoted by **{G, ∘}**, is a set under some operations (∘) if it satisfies the **CAIN properties**.

- **C** – Closure
- **A** – Associative
- **I** – Identity
- **N** – iNverse

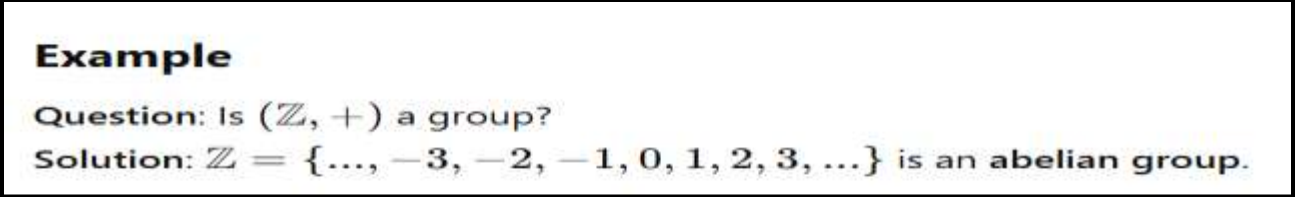
Abelian Group

A group is said to be **Abelian** if it is already a group and **commutative property** is also satisfied, i.e.,

$(a \circ b) = (b \circ a)$ for all **a, b** in **G**.

📌 **Note**:
•The first 4 properties (Closure, Associative, Identity, Inverse) define a **Group**.
•If **Commutative** is also satisfied, then it is an **Abelian Group**.

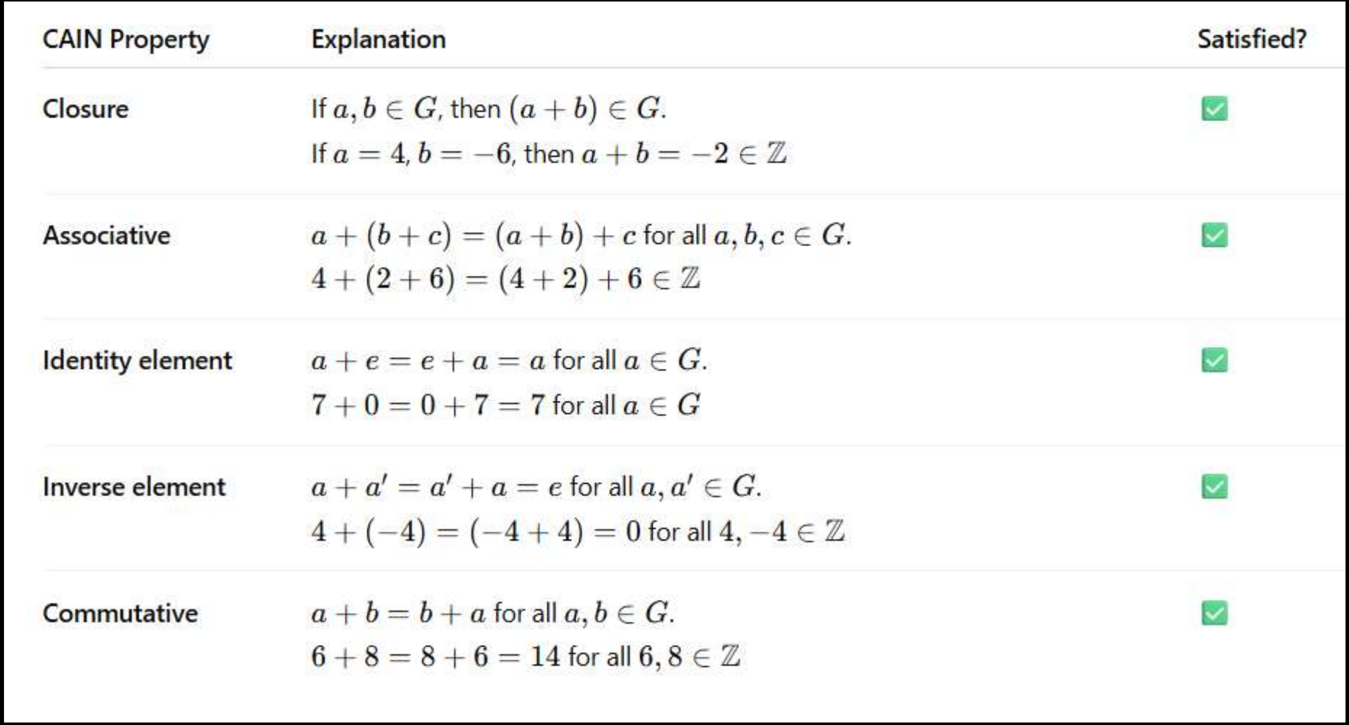
Property	Explanation
Closure	If $a, b \in G$, then $(a \circ b) \in G$
Associative	$a \circ (b \circ c) = (a \circ b) \circ c$ for all $a, b, c \in G$
Identity element	$(a \circ e) = (e \circ a) = a$ for all $a, e \in G$
Inverse element	$(a \circ a') = (a' \circ a) = e$ for all $a, a' \in G$
Commutative	$(a \circ b) = (b \circ a)$ for all $a, b \in G$



Example

Question: Is $(\mathbb{Z}, +)$ a group?

Solution: $\mathbb{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$ is an **abelian group**.



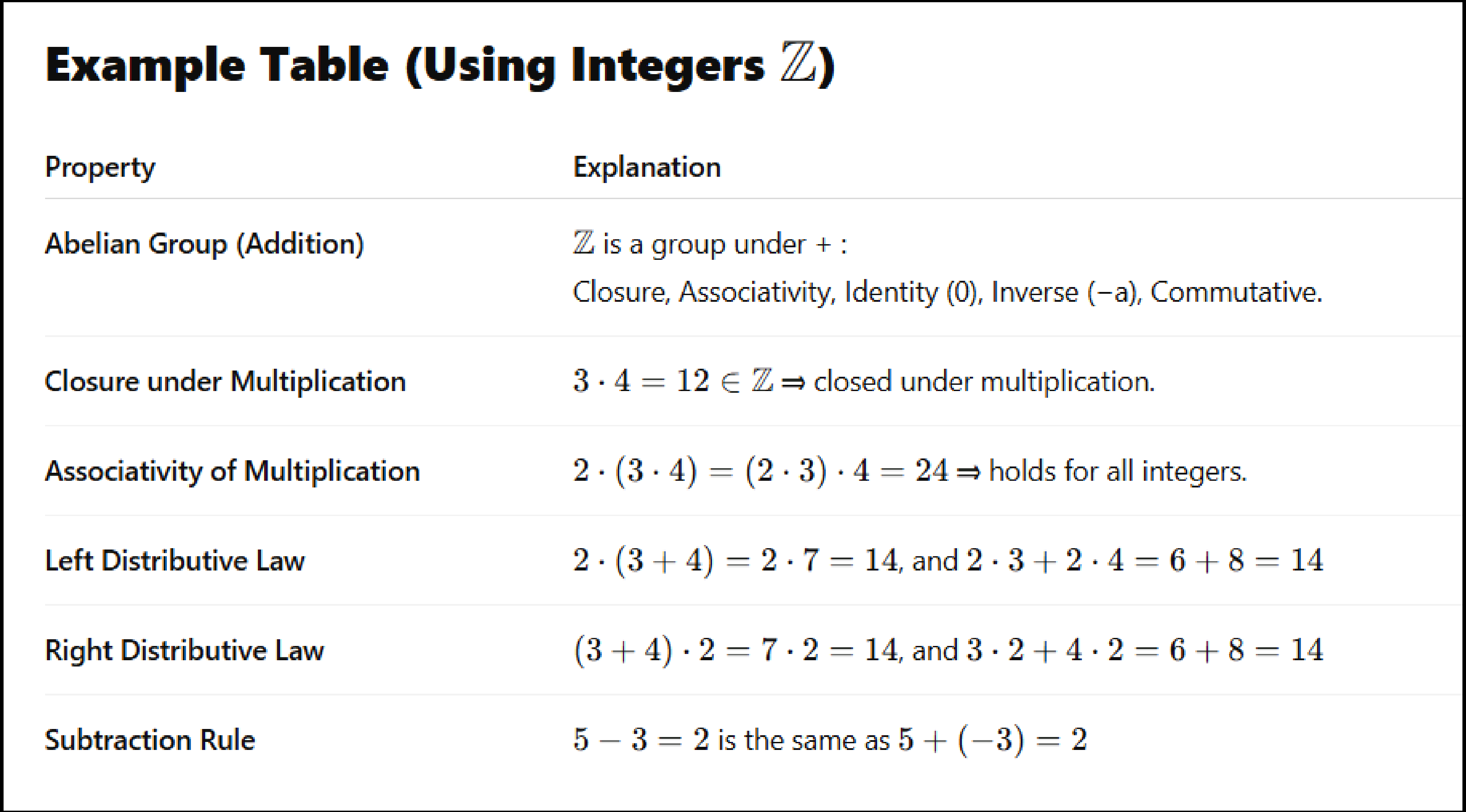
CAIN Property	Explanation	Satisfied?
Closure	If $a, b \in G$, then $(a+b) \in G$. If $a = 4, b = -6$, then $a + b = -2 \in \mathbb{Z}$	✓
Associative	$a + (b + c) = (a + b) + c$ for all $a, b, c \in G$. $4 + (2 + 6) = (4 + 2) + 6 \in \mathbb{Z}$	✓
Identity element	$a + e = e + a = a$ for all $a \in G$. $7 + 0 = 0 + 7 = 7$ for all $a \in G$	✓
Inverse element	$a + a' = a' + a = e$ for all $a, a' \in G$. $4 + (-4) = (-4 + 4) = 0$ for all $4, -4 \in \mathbb{Z}$	✓
Commutative	$a + b = b + a$ for all $a, b \in G$. $6 + 8 = 8 + 6 = 14$ for all $6, 8 \in \mathbb{Z}$	✓

Rings

A ring R denoted by $\{R, +, *\}$, is a set of elements with two binary operations, called **addition** and **multiplication**, such that for all $a, b, c \in R$, the following axioms are obeyed:

- ✅ **Group (A1–A4), Abelian Group (A5)**
- ✅ **Closure under multiplication (M1):**
  If $a, b \in R$, then $ab \in R$
- ✅ **Associativity of multiplication (M2):**
  $a(bc) = (ab)c$ for all $a, b, c \in R$
- ✅ **Distributive laws (M3):**
  - Left: $a(b + c) = ab + ac$ for all $a, b, c \in R$
  - Right: $(a + b)c = ac + bc$ for all $a, b, c \in R$

> **Note:**
> Subtraction: $a - b = a + (-b)$



Example Table (Using Integers $\mathbb{Z}$)

Property	Explanation
Abelian Group (Addition)	$\mathbb{Z}$ is a group under $+$: Closure, Associativity, Identity (0), Inverse (−a), Commutative.
Closure under Multiplication	$3 \cdot 4 = 12 \in \mathbb{Z}$ ⇒ closed under multiplication.
Associativity of Multiplication	$2 \cdot (3 \cdot 4) = (2 \cdot 3) \cdot 4 = 24$ ⇒ holds for all integers.
Left Distributive Law	$2 \cdot (3 + 4) = 2 \cdot 7 = 14$, and $2 \cdot 3 + 2 \cdot 4 = 6 + 8 = 14$
Right Distributive Law	$(3 + 4) \cdot 2 = 7 \cdot 2 = 14$, and $3 \cdot 2 + 4 \cdot 2 = 6 + 8 = 14$
Subtraction Rule	$5 - 3 = 2$ is the same as $5 + (-3) = 2$

Fields

A **field** F, sometimes denoted by $\{F, +, \cdot\}$, is a set of elements with two binary operations — **addition** and **multiplication** — such that for all $a, b, c \in F$, the following axioms are satisfied:

✅ (A1–M6):

F is an **integral domain**, meaning it satisfies:

- Group properties for addition (Abelian group)
- Closure, associativity, and distributivity for multiplication

✅ (M7) Multiplicative Inverse:

For every $a \in F$, $a \neq 0$, there exists $a^{-1} \in F$ such that:
$a \cdot a^{-1} = a^{-1} \cdot a = 1$

> **Note**:
> $\frac{a}{b} = a \cdot b^{-1}$

Familiar Examples of Fields:

- Rational numbers $\mathbb{Q}$
- Real numbers $\mathbb{R}$
- Complex numbers $\mathbb{C}$

Example Table: Field using Rational Numbers $\mathbb{Q}$

Property	Example
Additive Abelian Group	$\frac{2}{3} + (-\frac{2}{3}) = 0$: Addition is associative, has identity (0), and inverses
Closure under Multiplication	$\frac{1}{2} \cdot \frac{3}{4} = \frac{3}{8} \in \mathbb{Q}$
Associativity (Multiplication)	$\frac{1}{2} \cdot (\frac{2}{3} \cdot \frac{3}{4}) = (\frac{1}{2} \cdot \frac{2}{3}) \cdot \frac{3}{4}$
Distributivity	$\frac{1}{2}(\frac{1}{3} + \frac{1}{6}) = \frac{1}{2} \cdot \frac{1}{3} + \frac{1}{2} \cdot \frac{1}{6} = \frac{1}{6} + \frac{1}{12} = \frac{1}{4}$
Multiplicative Inverse	For $\frac{2}{3} \in \mathbb{Q}$, its inverse is $\frac{3}{2}$, and $\frac{2}{3} \cdot \frac{3}{2} = 1$
Division as Inverse	$\frac{a}{b} = a \cdot b^{-1}$, e.g., $\frac{2}{5} = 2 \cdot \frac{1}{5}$

Prime Numbers – Definition and Properties

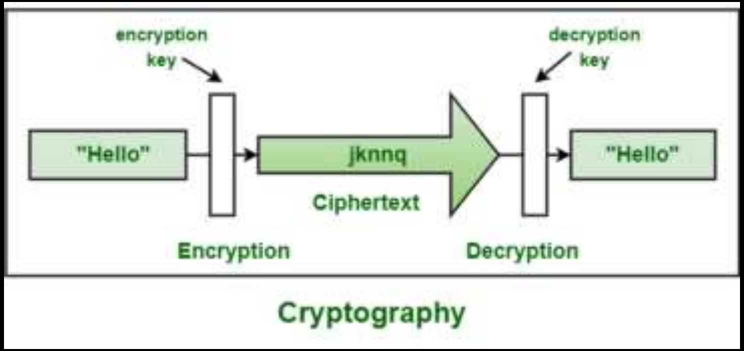
◆ **Definition:**
A **prime number** is a natural number greater than **1** that has **exactly two distinct positive divisors**:
1 and itself.
◆ **Examples:**
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, ...

Property	Description
Divisors	A prime number has only two positive divisors: 1 and itself
First Prime Number	2 (also the **only even** prime number)
Odd Nature	All primes except 2 are odd
Fundamental Theorem	Every natural number greater than 1 is either a prime or can be **factored uniquely** as a product of prime numbers
Infinitude	There are **infinitely many** prime numbers (proved by Euclid)
No Pattern	Primes don’t follow a simple formula or fixed gap

Number	**Prime?**	**Why?**
2	✓ Yes	Divisors: 1, 2
4	✗ No	Divisors: 1, 2, 4 (More than 2)
5	✓ Yes	Divisors: 1, 5
9	✗ No	Divisors: 1, 3, 9
17	✓ Yes	Divisors: 1, 17

🔐 Cryptography kya hota hai?

Cryptography ek technique hai jisme hum apne **data ya message ko secret banate hain** taaki sirf jis bande ko message bhejna hai wahi usse padh sake.
Jaise hum **plain message (readable)** ko **code (ciphertext)** mein badal dete hain — isse kehte hain encryption.
Jab us code ko wapas original message mein badla jata hai — usse kehte hain decryption.



🔐 **Encryption kya hota hai?**
Encryption ka matlab hai:
☞ Normal message (plaintext) ko **secret code** (ciphertext) mein badalna.
◆ **Example:**
- Message: Hello Bhai
- Encrypted message: @8Kjd#32!

Ye code sirf wahi padh sakta hai jiske paas key hai.

🔓 **Decryption kya hota hai?**
Decryption ka matlab hai:
☞ Secret code (ciphertext) ko wapas normal message (plaintext) mein badalna.
◆ **Example:**
Aapka encrypted WhatsApp message aapki mummy ke phone pe decrypt hoke normal form mein dikhayi deta hai:
- Ciphertext: @8Kjd#32!
- Decrypted message: Hello Bhai

Symmetric Key Cryptography (Ek hi key se)

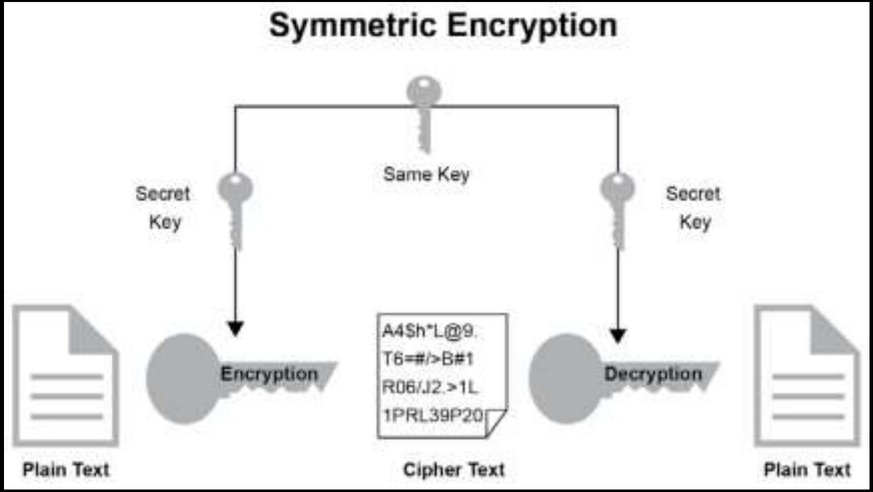
Isme **encrypt aur decrypt dono kaam ek hi key se hote hain**.

📔 Real-Life Example:

Jaise aap ek taala aur ek chaabi use karte ho.
Jo chaabi se taala band kiya, usi se khulega bhi.

◆ Use hota hai:

•ZIP file password
•Wi-Fi encryption (WPA2)
•AES algorithm



✓ Symmetric Key Cryptography – Standard Algorithms

Symmetric cryptography mein **same key** use hoti hai encryption aur decryption ke liye. Isme kai **standard algorithms** use kiye jaate hain jo fast aur efficient hote hain.

Symmetric key algorithms: Data Encryption Standards,
Advanced Encryption Standard,

🔐 DES (Data Encryption Standard) Kya Hai?

📌 Definition:

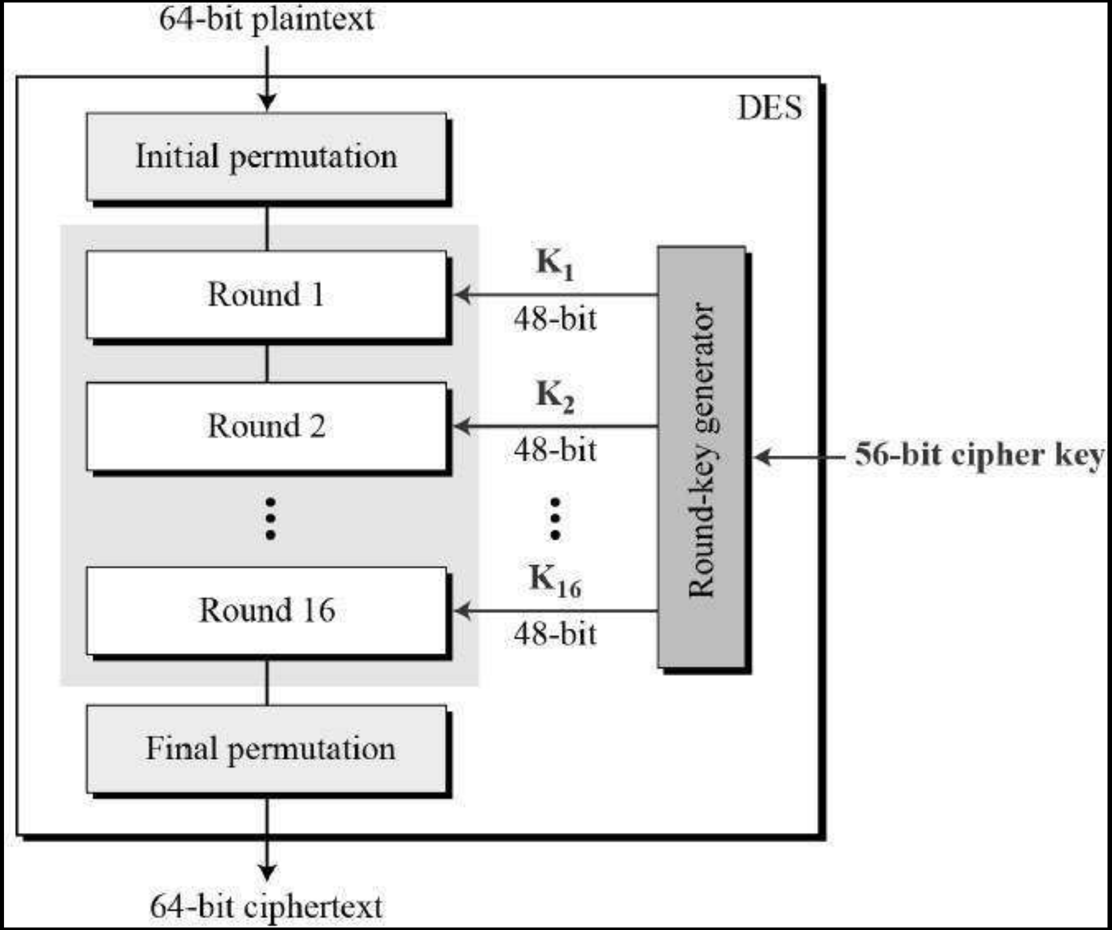
DES (Data Encryption Standard) ek **symmetric-key block cipher** hai jo fixed-size block (64-bit) ko encrypt karta hai ek secret key (56-bit) ke saath.

- **Symmetric** = Same key for encryption & decryption
- **Block Cipher** = Data fixed-size blocks mein divide hota hai (DES mein: 64-bit blocks)
- **Key Size** = 56-bit (technically 64-bit, but 8 bits parity ke liye hote hain)

🔧 DES Kaam Kaise Karta Hai?

➤ High-Level Steps:

1. 64-bit plaintext input
2. Initial permutation
3. 16 rounds of processing (Feistel rounds)
4. Final permutation
5. 64-bit ciphertext output


64-bit plaintext
DES
Initial permutation
Round 1
K_1
48-bit
Round 2
K_2
48-bit
Round 16
K_{16}
48-bit
Round-key generator
56-bit cipher key
Final permutation
64-bit ciphertext

🔐 DES Symmetric Encryption – Diagram Explanation (Step-by-step):

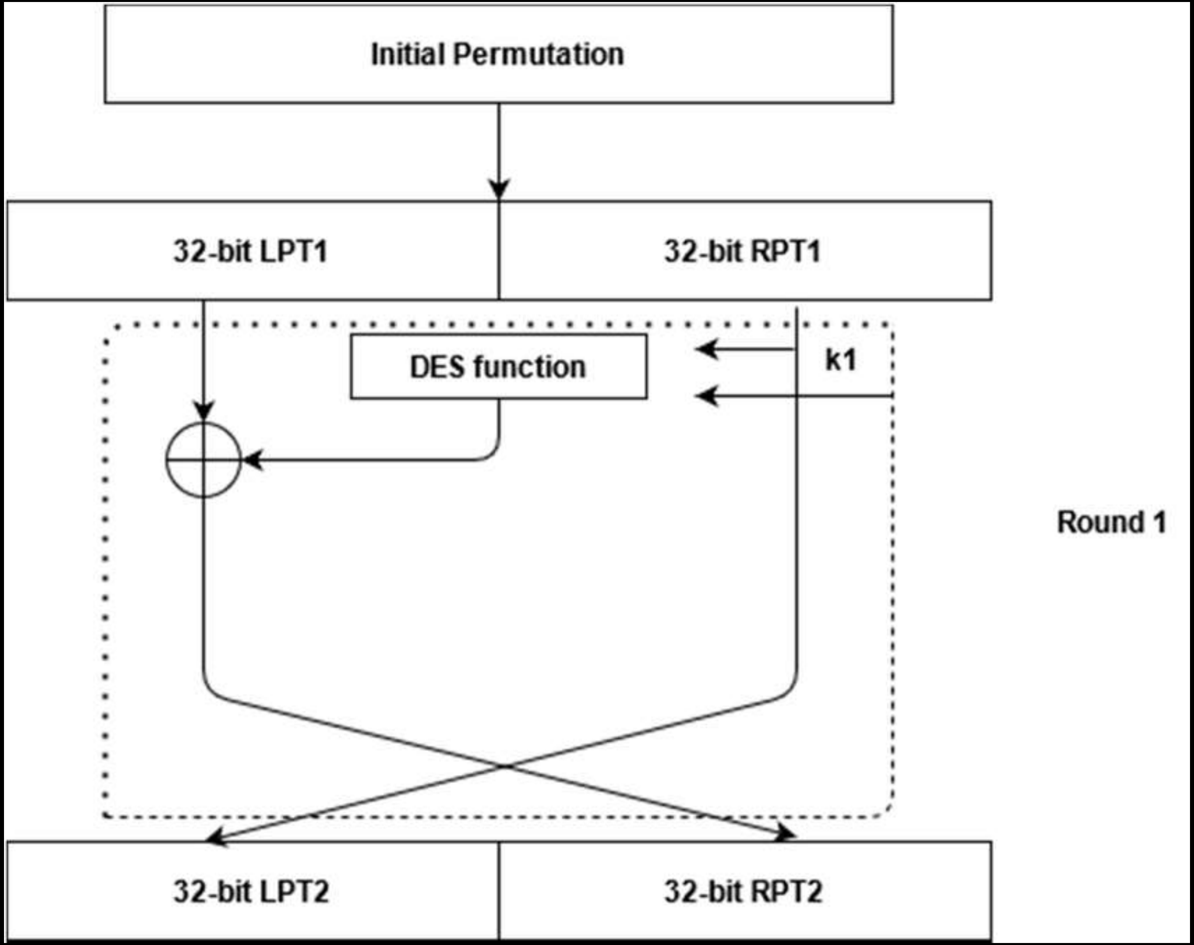
🎯 **Input:**

- **Plaintext (64-bit)** = 10101010 10101010 10101010 10101010 01010101 01010101 01010101 01010101
- **Key (56-bit usable)** = 133457799BBCDFF1 (hex value used in real examples)

◆ Step 1: Initial Permutation (IP)

- Fixed permutation table ke according 64-bit plaintext ke bits shuffle hote hain.
- Output: Permuted 64-bit value

📌 **Purpose**: Confusion badhane ke liye — bit positions change hoti hain.



Initial Permutation
32-bit LPT1
32-bit RPT1
DES function
k1
Round 1
32-bit LPT2
32-bit RPT2

🔐 DES Round 1 Process (Step-wise Explanation in Hinglish)

Step 1: Initial Permutation (IP)

- Plaintext (64-bit) input hota hai.
- Is input par **Initial Permutation** apply hoti hai.
- Result: 64-bit block divide hota hai:
    - Left Part (LPT1) – 32-bit
    - Right Part (RPT1) – 32-bit

Step 2: DES Function ke Input Preparation

- RPT1 ko **DES Function** ke input ke roop mein use kiya jaata hai.
- Saath hi, first round key **K1** bhi function ko di jaati hai.

Step 3: DES Function Execution

•**DES Function (F)**:

- 32-bit RPT1 ko 48-bit mein expand karta hai (Expansion permutation).
- Phir usme 48-bit round key **K1** ka XOR hota hai.
- S-boxes use karke output ko wapas 32-bit mein compress karta hai.
- Phir final permutation apply hoti hai.

•Output: 32-bit data milta hai from DES function.

Step 4: XOR Operation

•DES function ka output aur **LPT1** ke beech XOR operation hota hai:

LPT2 = LPT1 XOR F(RPT1, K1)

Step 5: Swapping

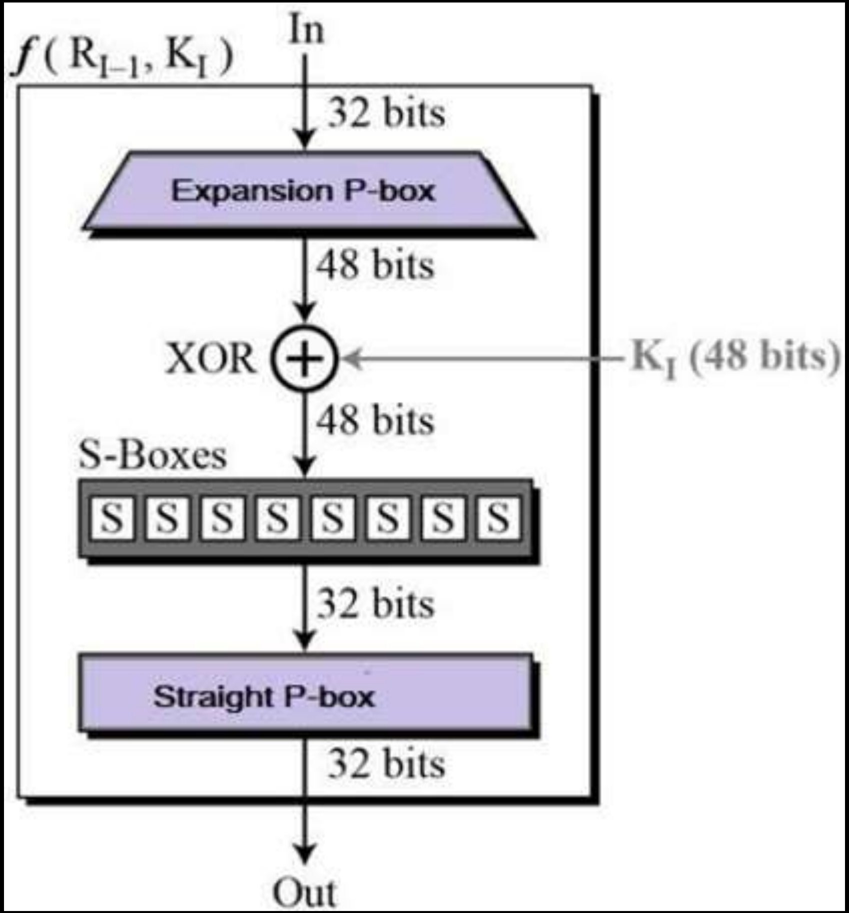
- **Swapping** hota hai:
  - RPT2 = LPT1
  - LPT2 = XOR result (step 4 ka output)

Step 6: Output of Round 1

- Ab humare paas hai:
  - LPT2 (32-bit)
  - RPT2 (32-bit)
- Yeh dono next round ke input ke roop mein jaayenge (Round 2).

Summary of Data Flow

Initial Permutation
↓
LPT1, RPT1
↓
DES Function(F(RPT1, K1)) → XOR with LPT1
↓
Swap LPT1 & RPT1 → LPT2 = result, RPT2 = LPT1
↓
Next Round


$f(R_{I-1}, K_I)$
In
32 bits
Expansion P-box
48 bits
XOR
K_I (48 bits)
48 bits
S-Boxes
S S S S S S S S
32 bits
Straight P-box
32 bits
Out

🔍 DES Round Function f(R, K) – Step-by-Step Explanation

✅ **Input:**

- R(i-1) = 32-bit Right part of previous round.
- K(i) = 48-bit Round Key for the current round.

Step 1: Expansion P-box (Expansion Permutation)

- 32-bit input ko 48-bit mein expand kiya jaata hai.
- Iska purpose hai kuch bits ko repeat karna taaki ye key ke 48-bit se match karein.
- Is stage ke baad:

Output = 48 bits

Step 2: XOR Operation

- Ab ye 48-bit expanded R ke saath 48-bit round key K(i) ka **bitwise XOR** kiya jaata hai.
- Ye key mixing hoti hai.

$$\text{Result} = \text{Expanded } R(i-1) \oplus K(i)$$

Step 3: S-Boxes (Substitution Boxes)

- 48-bit XOR output ko 8 blocks mein divide kiya jaata hai (6 bits each).
- Har block ko ek specific **S-Box** mein daala jaata hai:
  - Har S-Box: 6-bit input leta hai aur 4-bit output deta hai.
- 8 S-boxes × 4-bit output = **32-bit total output**

Step 4: Straight P-box (Permutation)

•32-bit S-box output ko rearrange (permute) kiya jaata hai using a fixed table (Straight P-box).

•Ye security enhance karta hai by diffusing the bits.

Step 5: Output

•Final 32-bit output nikalta hai.

•Ye output aage LPT ke saath XOR hota hai (in the main DES round process).

Input: 32-bit R(i-1)

↓

Expansion P-box → 48-bit

↓

XOR with 48-bit Key (K(i))

↓

S-Boxes → 32-bit output

↓

Straight P-box → Final 32-bit output

🔍 S-Box Working Explained (Step-by-Step in Hinglish)

📌 **Input:**

•Input: 6-bit binary number → **110010**

Step 1: Determine Row Number

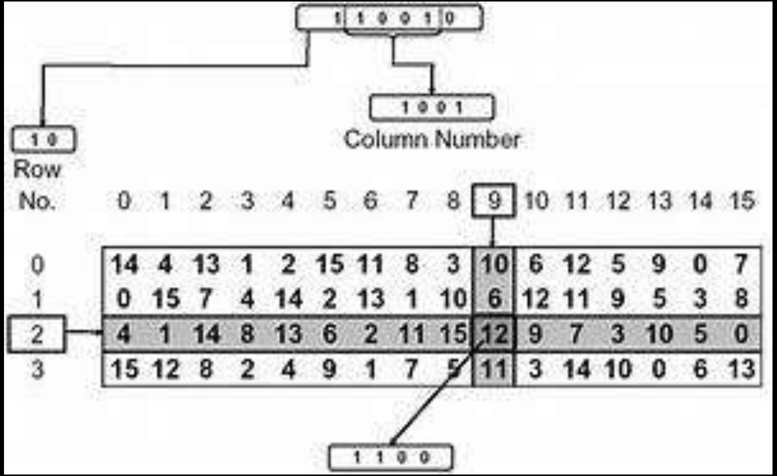
•**First** aur **Last** bit milakar **Row Number** banta hai.

- •First bit = 1
- •Last bit = 0
- •Row = 10 (binary) = **2 (decimal)**

Step 2: Determine Column Number

•**Middle 4 bits** → Column number banate hain.

- •Middle bits = 1001 = **9 (decimal)**



Row No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

📌 **Step 3: Find Value from S-Box Table**

•Use Row = 2, Column = 9

•Table mein row 2, column 9 ka value = **12**

📌 **Step 4: Convert to 4-bit Binary**

•12 (decimal) = 1100 (4-bit binary)

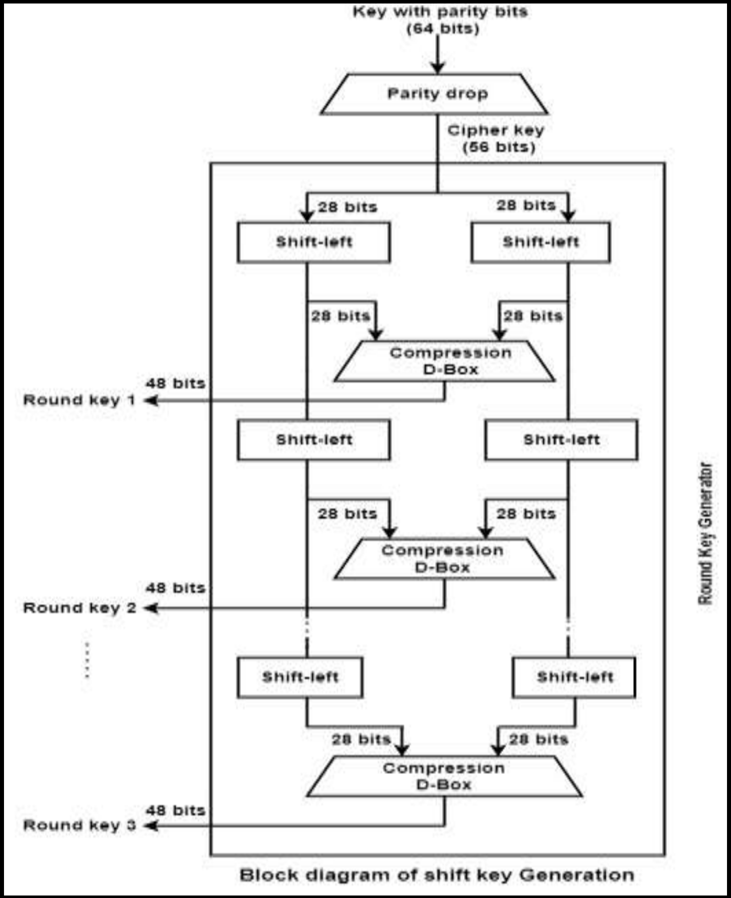
S-Box input:	110010 (6-bit)
Row:	2
Column:	9
S-Box value:	12 (decimal)
Output (binary):	1100 (4-bit)

💡 **Use of S-Box in DES:**

•Har 6-bit block ke liye ek specific S-box hoti hai (total 8 S-boxes).

•Har S-box: 6-bit input → 4-bit output deta hai.

•Total 8 × 4-bit = 32-bit output.



Block diagram of shift key Generation

🔐 Block Diagram of Shift Key Generation — Step-by-Step

✓ Step 1: 64-bit Key with Parity Bits

- Input key: **64-bit** hoti hai.
- Isme se har 8th bit parity ke liye hoti hai (total 8 parity bits).

✓ Step 2: Parity Drop (PC-1)

- Parity bits hata di jaati hain.
- Bacha hua key = **56-bit cipher key**

✓ Step 3: Divide into Two Halves

- 56-bit key ko do hisson mein divide kiya jaata hai:
  - **Left half (C)** = 28 bits
  - **Right half (D)** = 28 bits

✓ Step 4: Left Shift

- Har round mein C aur D ko **left circular shift** (1 ya 2 bit) kiya jaata hai.
- Shift amount har round ke according change hota hai (standard DES table se).

✓ Step 5: Compression Permutation (PC-2)

- Shifted C aur D ko **milakar (total 56 bits)**, ek **Compression D-Box** ke through pass kiya jaata hai.
- Isme se **48 bits** nikaale jaate hain = **Round Key**

🔄 Repeat for All 16 Rounds

- Har round ke liye:
  - C aur D ko shift karo
  - Combine karo
  - Compression D-box lagao
  - 48-bit round key generate karo

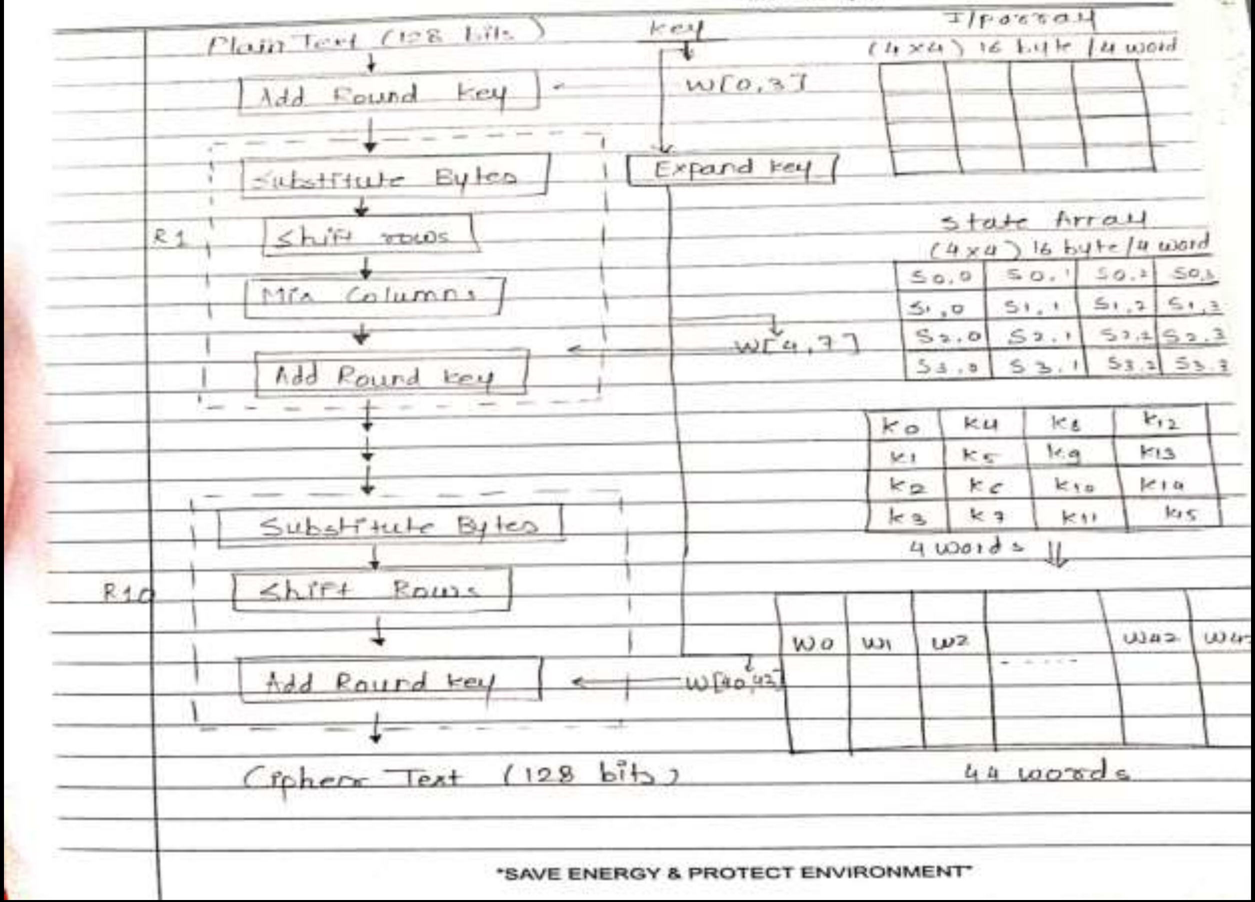
📌 Output:

- **16 Round Keys**, each 48-bit, for 16 DES rounds.

Step	Operation	Size
1	Input key	64 bits
2	Parity Drop (PC-1)	56 bits
3	Split into C & D	28 + 28 = 56
4	Left shift (round-wise)	28 + 28
5	Compression D-box (PC-2)	48-bit key
	🔄 Repeat for 16 rounds	Total 16 keys

⬘ **Advantages**	▨ **Disadvantages**
✓ Simple and easy to implement	✕ Short key length (56-bit) → easily crackable
✓ Fast in hardware implementation	✕ Vulnerable to brute-force attacks
✓ Standardized and well-studied algorithm	✕ Not suitable for modern encryption needs
✓ Widely used in early cryptographic systems	✕ Not secure for sensitive or long-term data
✓ Useful for learning classical encryption	✕ Linear & differential cryptanalysis possible
✓ Basis for modern algorithms like 3DES	✕ Not efficient for software-based encryption

AES(**Advanced Encryption Standard.**)



S0,0	S0,1	S0,2	S0,3
S1,0	S1,1	S1,2	S1,3
S2,0	S2,1	S2,2	S2,3
S3,0	S3,1	S3,2	S3,3

k0	k4	k8	k12
k1	k5	k9	k13
k2	k6	k10	k14
k3	k7	k11	k15

w0	w1	w2	.....	w42	w43

🔐 AES Algorithm (128-bit) – Diagram Explanation in Hinglish

✅ Step 0: Input

- **PlainText (128 bits)**: Yaani 16 bytes ka data (4x4 matrix form mein).
- **Key (128 bits)**: Yaani 16 bytes ka secret key (4x4 matrix).

⬛ Input Array (4x4)

- Ek 4x4 matrix jismein 16 bytes ka PlainText fill kiya jata hai.
- Har cell ek byte ko represent karta hai.

⬛ State Array (4x4)

- Ye woh matrix hai jismein encryption process hoti hai.
- State matrix format:

```
S00 S01 S02 S03
S10 S11 S12 S13
S20 S21 S22 S23
S30 S31 S32 S33
```

🔧 Key Expansion

- 128-bit key se total **44 words** generate kiye jaate hain (har word = 4 bytes).
- Ye words har round mein use hote hain for encryption.
- Jaise: W[0] to W[43].

🔁 Initial Round (Before Round 1)

☞ Add Round Key:

- Pehle 4 words (W[0] to W[3]) ko PlainText matrix ke saath XOR kiya jata hai.
- Ye pehla encryption step hota hai

🔁 Rounds 1 to 9 (R1 to R9)

Har round mein 4 steps hote hain:

1. **Substitute Bytes**
    1. Har byte ko S-Box se replace kiya jata hai.
    2. Isse confusion create hoti hai.
2. **Shift Rows**
    1. Matrix ki har row ko left circular shift karte hain:
        1. Row 0 → No shift
        2. Row 1 → 1-byte left shift
        3. Row 2 → 2-byte left shift
        4. Row 3 → 3-byte left shift
3. **Mix Columns**
    1. Har column ka ek fixed matrix ke saath multiplication hota hai.
    2. Isse diffusion badhti hai (spread of data).
4. **Add Round Key**
    1. Round-specific 4 words se XOR karte hain.
    2. Jaise: Round 1 mein W[4] to W[7], Round 2 mein W[8] to W[11], etc.

🔁 Final Round (Round 10 / R10)

Is round mein **Mix Columns nahi hota**. Sirf 3 steps:

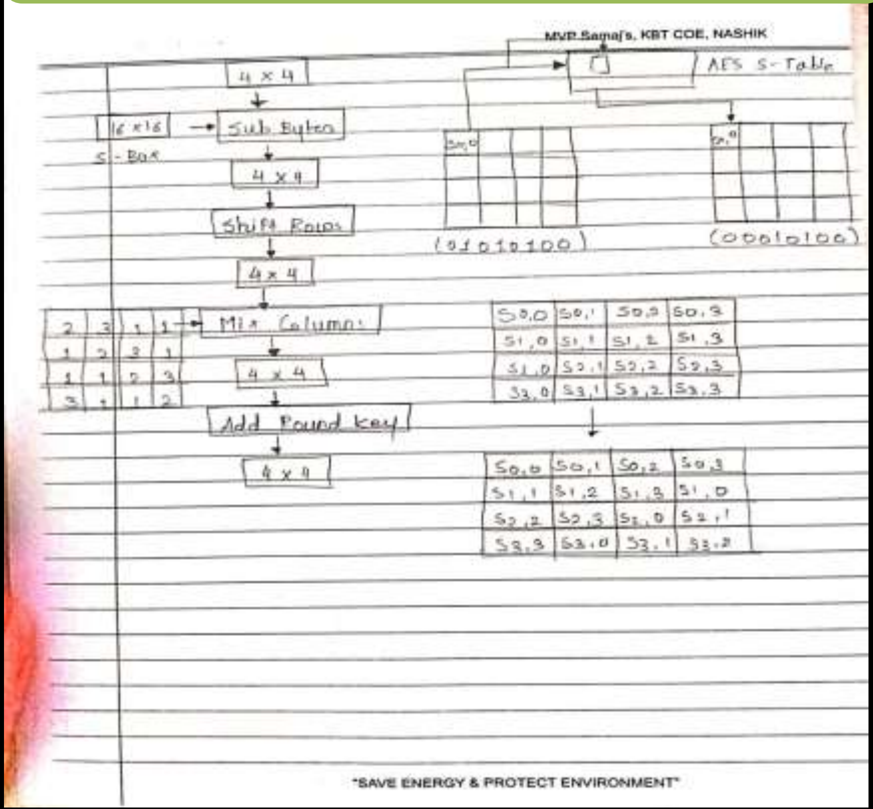
1. Substitute Bytes
2. Shift Rows
3. Add Round Key (W[40] to W[43])

□ Final Output: Cipher Text (128 bits)

Ye encrypted output hai – jo safe hota hai, unauthorized log samajh nahi sakte.

Step	Description
Step 0	Add Round Key (Initial)
Rounds 1-9	SubBytes → ShiftRows → MixColumns → AddRoundKey
Round 10	SubBytes → ShiftRows → AddRoundKey

🔁 AES Round Steps (Left to Right Flow)



1. Input Block (4x4 matrix)

•AES mein data ko 4x4 matrix (State matrix) ke form mein treat kiya jaata hai.

•Har cell 1 byte (8-bit) ka hota hai.

•Is matrix par operations perform kiye jaate hain.

2. SubBytes (Substitution using S-Box)

•Har byte ko AES S-Box (16x16 table) se replace kiya jaata hai.

•Har byte (e.g., S0,0) ko binary mein convert karke S-Box mein dekha jaata hai, jisse ek naya byte milta hai.

•Yeh non-linear substitution confusion create karta hai (security ke liye important).

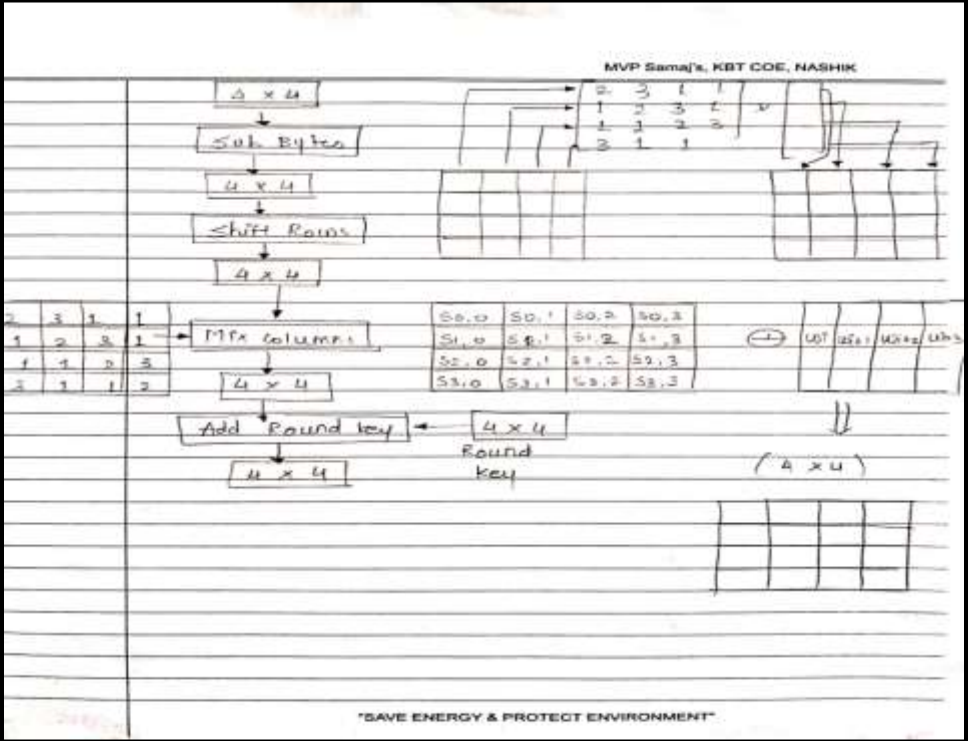
📌 Diagram mein dikhaya gaya hai kaise S0,0 ko S-Box se substitute kiya jaata hai.

3. ShiftRows

•Matrix ke rows ko left shift kiya jaata hai:

- Row 0: No shift
- Row 1: 1 byte shift left
- Row 2: 2 bytes shift left
- Row 3: 3 bytes shift left

 Diagram mein right side mein clearly dikhaya gaya hai ki shifting ke baad elements kaise rearrange hote hain.



4. MixColumns

- Har column ko ek fixed matrix ke saath multiply kiya jaata hai (Galois Field GF(2^8) mein).
- Is matrix ko diagram ke left side mein diya gaya hai:
- Is step se diffusion enhance hoti hai (matlab ek byte ka change poore block ko affect karega).

5. AddRoundKey

- Har byte ko round key ke corresponding byte ke saath XOR kiya jaata hai.
- Round key bhi 4x4 matrix hoti hai, jo key expansion se milti hai.

🔄 **Is Process ko har round ke liye repeat kiya jaata hai** (last round mein MixColumns nahi hota).

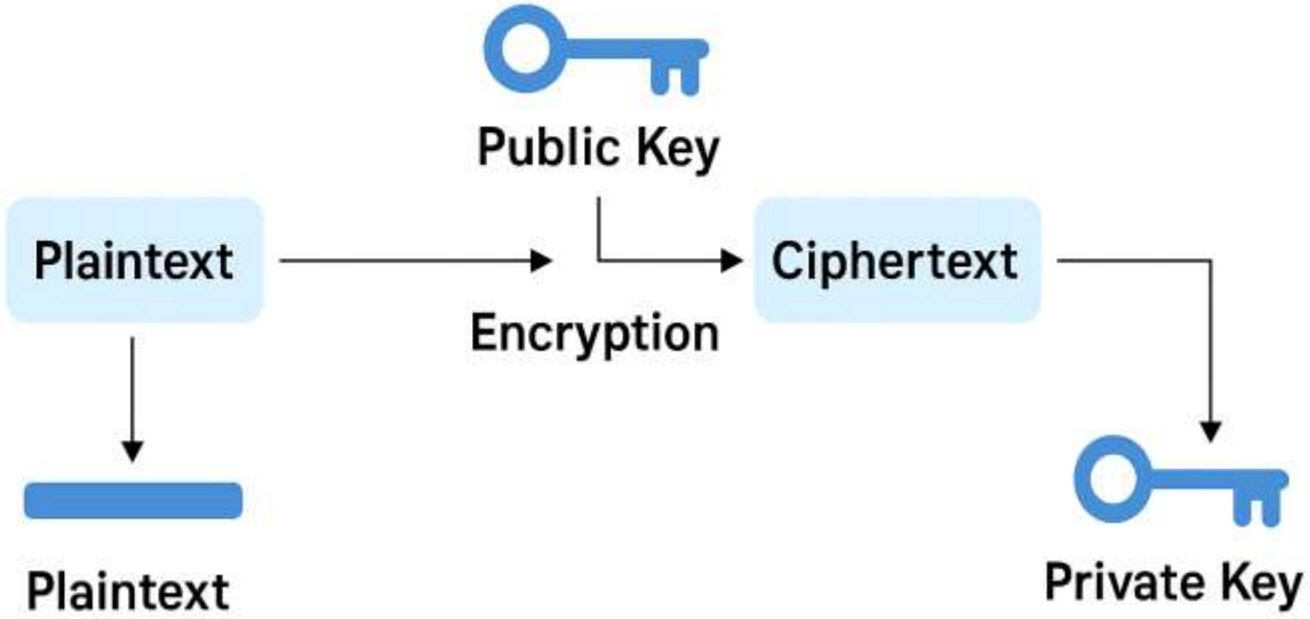
🔢 Step No.	🔍 Step Name	📃 Description (Hinglish Explanation)
0️⃣	Input Preparation	PlainText (128-bit) and Secret Key (128-bit) ko 4x4 matrix mein divide karte hain. Har cell = 1 byte.
1️⃣	Key Expansion	Initial 128-bit key se total **44 words** generate hote hain (4-byte each). W[0] to W[43].
2️⃣	Add Round Key (Initial Round)	State matrix XOR with key words W[0] to W[3]. Ye pre-round step hota hai.
🔁 R1–R9	Main Rounds (9 total)	Har round mein ye 4 operations perform hote hain:
	🔹 SubBytes	Har byte ko AES S-Box se replace karte hain → confusion badhta hai.
	🔹 ShiftRows	4x4 matrix ki rows ko left circular shift karte hain (Row 0 = 0 shift, Row 1 = 1 shift...).
	🔹 MixColumns	Har column ko ek fixed matrix ke saath multiply karte hain → diffusion create hoti hai.
	🔹 Add Round Key	Round-specific key (e.g., W[4]–W[7] for Round 1) se XOR karte hain.
🔟	Final Round (R10)	Last round mein sirf 3 steps hote hain:
	🔹 SubBytes	S-Box substitution (same as above).
	🔹 ShiftRows	Same row shift as earlier rounds.
	🔹 Add Round Key	Final key W[40] to W[43] se XOR karte hain.
✅	Cipher Text Output	Final 4x4 matrix ko linear form mein convert karke encrypted text milega (128-bit encrypted output).

🔐 Asymmetric Cryptography

📄 Definition (Kya hota hai?)
Asymmetric Cryptography ek aisa encryption technique hai jismein **do alag-alag keys** use hoti hain:
•**Public Key (sabko pata hoti hai)**
•**Private Key (sirf user ke paas hoti hai, secret)**
Ye dono keys **mathematically linked** hoti hain — ek se encrypt karo, to doosre se hi decrypt ho sakta hai.

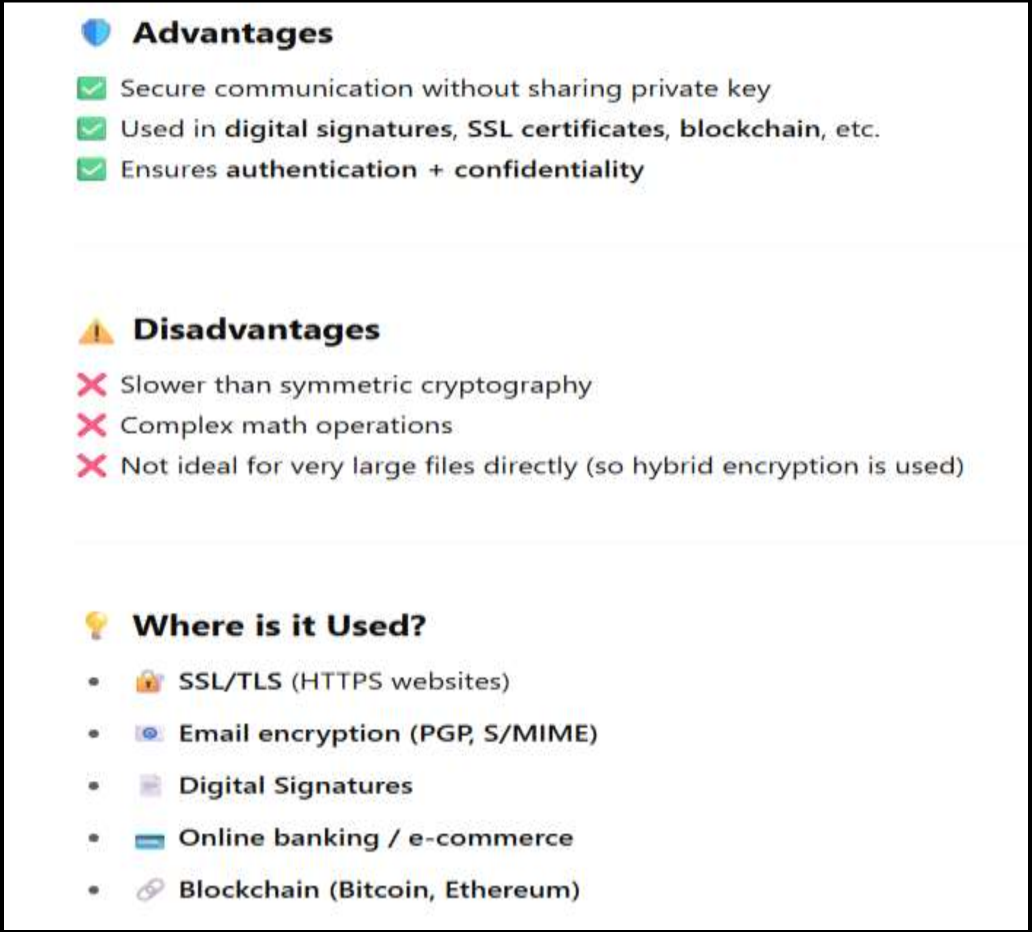
 Simple Analogy (Example ke saath samjho)
Soch lo ek **Tijori (locker)** hai jiska **taala sab khol sakte hain (public key)**, lekin **chabi sirf owner ke paas hai (private key)**.
•Tum kisi ko secret message bhejna chahte ho.
•Tum uske **public key** se message ko **encrypt** karte ho.
•Sirf uske paas jo **private key** hai, usi se wo **decrypt** kar sakta hai.
 Safe hai kyunki:
•Message bhejne wale ko private key nahi chahiye.
•Agar koi public key se encrypt kare, to sirf private key se hi uska solution milega.

ASYMMETRIC CRYPTOGRAPHY



🔄 How it Works (Kaise kaam karta hai?)

🔑 Step	🔍 Action
1️⃣	Receiver apni **public key** sabko de deta hai.
2️⃣	Sender us public key se message encrypt karta hai.
3️⃣	Receiver apni **private key** se message decrypt karta hai.



🛡️ Advantages

✅ Secure communication without sharing private key
✅ Used in **digital signatures**, **SSL certificates**, **blockchain**, etc.
✅ Ensures **authentication + confidentiality**

⚠️ Disadvantages

❌ Slower than symmetric cryptography
❌ Complex math operations
❌ Not ideal for very large files directly (so hybrid encryption is used)

💡 Where is it Used?

- 🔐 **SSL/TLS** (HTTPS websites)
- 📧 **Email encryption (PGP, S/MIME)**
- 📄 **Digital Signatures**
- 💳 **Online banking / e-commerce**
- 🔗 **Blockchain (Bitcoin, Ethereum)**

Point	Symmetric Cryptography	Asymmetric Cryptography
1. Keys Used	Uses **one single key** for both encryption and decryption.	Uses **two keys** – Public key (for encryption) & Private key (for decryption).
2. Speed	Much **faster** in terms of performance and encryption speed.	Relatively **slower** due to complex mathematical operations.
3. Security	Less secure if key is shared improperly.	More secure – even if public key is known, private key remains secret.
4. Key Sharing	**Key must be shared secretly** with the receiver.	No need to share private key; only public key is shared.
5. Complexity	**Simple algorithm** design and implementation.	**Complex algorithm** (uses number theory, RSA, ECC etc.).
6. Suitable For	**Large data encryption** due to speed (e.g., file encryption).	**Secure key exchange**, digital signatures, SSL certificates.
7. Example Algorithms	AES, DES, Blowfish, RC4.	RSA, ECC, DSA, ElGamal.
8. Key Length	Typically **shorter** (e.g., 128-bit, 256-bit).	Typically **longer** (e.g., 1024-bit, 2048-bit or more).
9. Real-life Use	Used in **Wi-Fi encryption (WPA2), file encryption**.	Used in **email encryption, HTTPS, digital signatures**.

🔐 Public Key Encryption (Asymmetric Encryption)

💡 **Basic Idea:**

Ismein **2 alag-alag keys** hoti hain:

•**Public Key**: Sabke saath share ki ja sakti hai.

•**Private Key**: Sirf owner ke paas hoti hai, secret hoti hai.

🛍 **Kaise kaam karta hai?**

1.Encryption (Data ko lock karna):

1. Agar kisi ko message bhejna hai securely, toh **uske public key se encrypt karo**.
2. Sirf **uski private key hi us message ko decrypt** kar sakti hai.

2.🔑 *Jaise chabi se taala lagate ho, but taala sirf ek special chabi se khulta hai!*

1.Decryption (Data ko unlock karna):

1. Jab receiver ko message milta hai, wo **apni private key se message ko decrypt karta hai**.

📨 **Example:**

•Tu chaahta hai ki koi tujhe secret message bheje:

- Tu apni **public key** sabko deta hai.
- Jo bhi tujhe message bhejna chahe, wo uss key se encrypt karega.
- Tu **apni private key** se hi us message ko padh sakta hai.

✅ **Use Cases:**

•SSL/TLS (secure websites)

•Digital signatures

•Secure messaging apps (like WhatsApp, Signal)

🔁 Hash Function

💡 Basic Idea:

Hash function ek aisa **mathematical function** hota hai jo:

•Kisi bhi size ke input ko leke,

•Fixed size ka output (hash) generate karta hai.

•Output hamesha **unique** hota hai (mostly).

□ **Example:**

```
Input: "Hello"
Hash: "5d41402abc4b2a76b9719d911017c592"
```

⚙️ Properties:

1. **Fixed Output Size**: Chaahe input chhota ho ya bada, output hamesha same length ka hoga.
2. **Deterministic**: Same input dene par hamesha same output milega.
3. **One-way**: Hash se original data wapas nahi mil sakta.
4. **Collision Resistant**: Do alag inputs ka same hash aana mushkil hai.

🔒 Use Cases:

- Passwords ko securely store karna
- Data integrity check (file corruption detection)
- Blockchain
- Digital signatures ke saath use

RSA Algorithm

RSA algorithm ek **asymmetric cryptographic algorithm** hai jo data ko securely encrypt aur decrypt karne ke liye use hota hai. Iska full form hai **Rivest–Shamir–Adleman**, jo iske inventors ke naam par rakha gaya hai. RSA do alag-alag keys ka use karta hai — **Public Key** aur **Private Key**.

🔐 RSA Algorithm Basics:

🔑 1. Asymmetric Encryption

•**Public Key**: Ye key sabko di ja sakti hai. Iska use encryption ke liye hota hai.
•**Private Key**: Ye secret hoti hai. Sirf receiver ke paas hoti hai. Iska use decryption ke liye hota hai.

🔐 RSA Algorithm Basics:

🔑 1. Asymmetric Encryption

- **Public Key**: Ye key sabko di ja sakti hai. Iska use encryption ke liye hota hai.
- **Private Key**: Ye secret hoti hai. Sirf receiver ke paas hoti hai. Iska use decryption ke liye hota hai.

⚙️ Step-by-Step RSA Algorithm Process:

Step 1: Do Prime Numbers Choose Karo

- Maan lo do large prime numbers:
  - p = 61
  - q = 53

Step 2: **n Calculate Karo**

$$n = p \times q$$

$$n = 61 \times 53 = 3233$$

Ye n public key aur private key dono me use hota hai.

Step 3: **Euler's Totient Function (φ(n)) Calculate Karo**

$$\varphi(n) = (p - 1) \times (q - 1)$$

$$\varphi(n) = (61 - 1) \times (53 - 1) = 60 \times 52 = 3120$$

📌 Step 4: **Public Key Exponent 'e' Choose Karo**

Aisa number e chuno jo:
- $1 < e < \varphi(n)$
- e and φ(n) co-prime ho (gcd = 1)

$e = 17$ (example value) ☞ Public Key = (e, n) → (17, 3233)

□ Step 5: Private Key 'd' Find Karo

•d aisa number hona chahiye ki:

$$(d \times e) \bmod \varphi(n) = 1$$

Yani,

$$(d \times 17) \bmod 3120 = 1$$

Iska solution d = 2753

☞ Private Key = (d, n) → (2753, 3233)

RSA Encryption:

Example:
Maan lo, message = 65 (numeric form)
Encryption Formula:

cipher = (message ^ e) mod n
cipher = (65 ^ 17) mod 3233 = 2790

RSA Decryption:
Decryption Formula:

message = (cipher ^ d) mod n
message = (2790 ^ 2753) mod 3233 = 65

Step	Description
1	Do prime numbers choose karo (p, q)
2	$n = p \times q$
3	$\phi(n) = (p-1)(q-1)$
4	Public key e choose karo ($1 < e < \phi(n)$)
5	Private key d calculate karo ($d \times e \bmod \phi(n) = 1$)
6	Public key = (e, n), Private key = (d, n)
7	Encrypt: (M^e) mod n, Decrypt: (C^d) mod n

🔐 Real World Usage:

- **Secure websites (HTTPS)**
- **Digital Signatures**
- **Banking and Payment Systems**

📝 Digital Signature Kya Hota Hai?

Digital Signature ek **electronic signature** hai jo kisi document ya message ki **authenticity (asliyat)** aur **integrity (sachchai)** verify karta hai. Ye ensure karta hai ki:

1. **Message kisne bheja** (Authentication)
2. **Message beech me badla toh nahi** (Integrity)
3. **Sender later deny na kar sake** (Non-repudiation)

🔐 Digital Signature Kaam Kaise Karta Hai? (RSA ke sath)

Chaliye step-by-step samajhte hain:

✍️ Step 1: Message Hash Karo

•Sender jo message bhejna chahta hai, uska **hash** generate karta hai (using SHA-256, MD5, etc.)

Message: "Hello"
Hash: SHA256("Hello") → e.g., A1B2C3...

Hashing ka fayda:
•Fixed size ka output
•Message me ek character bhi change hua toh hash completely alag

🔐 Step 2: Hash Ko Private Key Se Encrypt Karo

•Sender apni **Private Key** se hash ko encrypt karta hai.
•Ye **Digital Signature** hota hai.

```
Digital Signature = Encrypt(Hash, Sender's Private Key)
```

📤 Step 3: Message + Signature Send Karo

Sender dono bhejta hai:
•Original Message
•Digital Signature

📥 Step 4: Receiver Signature Verify Karta Hai

Receiver kya karega:

1. **Same message ka hash nikalta hai**
2. Sender ki **Public Key** se Signature decrypt karta hai

```
Decrypted Signature = Decrypt(Signature, Sender's Public Key)
```

3. Dono hash compare karta hai:

```
If Hash(Message) == Decrypted Signature Hash:
    ✔️ Message valid
Else:
    ❌ Tampered / Invalid
```

Digital Signature Ke Fayde:

Feature	Meaning
🔐 Authentication	Verify karta hai ki message authorized person ne bheja
□ Integrity	Check karta hai message me koi badlav nahi hua
⊘ Non-repudiation	Sender baad me deny nahi kar sakta ki message usne nahi bheja

Real Life Use-Cases:

- ✓ Aadhaar-based e-sign
- ✓ Income tax return signing
- ✓ Software Signing (e.g., Windows installers)
- ✓ PDF/Document verification
- ✓ Blockchain transactions

📜 1. What is a Digital Certificate?

Digital Certificate ek digital document hota hai jo kisi entity (user, website, company) ki public key ko verify karta hai.

Ye confirm karta hai ki:
✅ Ye public key **real** hai
✅ Ye key **kis identity** se linked hai (jaise www.example.com)
✅ Is certificate ko **trustable authority** ne issue kiya hai

🔐 Example:

When you visit a secure site (https://...), browser check karta hai uska **digital certificate**, jise **SSL/TLS certificate** bhi kehte hain.

🔑 2. What is inside a Digital Certificate?

Ek digital certificate ke andar ye cheeze hoti hain:

Field	Description
👤 Subject	Jis entity ko certificate mila hai (e.g., domain name)
🔑 Public Key	Us entity ka public key
▯ Serial Number	Unique ID of certificate
🕓 Validity	Start and expiry date
🏢 Issuer	Certificate banane wali authority
✍▯ Digital Signature	Issuer ne apni private key se sign kiya hota hai

□ 3. What is Public Key Infrastructure (PKI)?

PKI ek complete system hai jo public-key encryption and digital certificates ko manage karta hai. Ye ensure karta hai ki:

✅ Trust maintained rahe

✅ Keys properly issued, stored, and revoked ho sake

👥 PKI ke Key Components:

Component	Role
CA (Certificate Authority)	Certificate banata hai aur sign karta hai
RA (Registration Authority)	User ke identity ko verify karta hai, CA ke behalf pe
Public/Private Keys	Encryption/decryption ke liye use hote hain
CRL (Certificate Revocation List)	Expired ya canceled certificates ki list
Digital Certificates	Public key + identity proof

PKI Flow – Step-by-Step

1. **User requests a certificate**
2. **RA** user identity verify karta hai
3. **CA** user ke liye certificate banata hai (including their public key)
4. CA **certificate ko digitally sign** karta hai
5. **User gets the certificate** (usually a .crt or .pem file)
6. Jab dusra koi user ya browser verify karta hai:
    - **CA ka public key** use karke certificate verify karta hai

🔐 Real World Example (Website SSL):

1. Website owner GoDaddy ya DigiCert se SSL certificate leta hai
2. CA us website ke public key ko digitally sign karta hai
3. Jab user site open karta hai:
    1. Browser verify karta hai ki:
        1. Certificate trusted CA se aaya hai?
        2. Expired toh nahi?
        3. Domain match karta hai?
4. Agar sab valid ho:

✅ Green padlock shown, HTTPS starts

Term	Meaning
🔑 Public Key	Openly shared key
🔐 Private Key	Secret key
📜 Digital Certificate	Binds public key to verified identity
⍰ PKI	System to manage digital certificates and keys
🏢 CA	Authority that issues and signs certificates
⍰ CRL	List of invalid certificates

🔐 Private Key Kya Hai?

•**Private Key** ek secret key hoti hai jo **asymmetric cryptography** mein use hoti hai.
•Ye key sirf usi person ke paas hoti hai jiska wo key pair belong karta hai.
•Iska use **data decrypt karne**, **digital signature create karne**, ya **authentication** ke liye hota hai.
•Ye bilkul secret rakhi jati hai — agar private key leak ho jaye to security compromise ho sakti hai.

👫 Public Key vs Private Key

🔢 No.	Feature	Public Key	Private Key
1️⃣	**Type**	Asymmetric Encryption Key	Asymmetric Encryption Key
2️⃣	**Visibility**	Public (can be shared with anyone)	Private (kept secret by the owner)
3️⃣	**Purpose**	Used to encrypt data or verify digital signature	Used to decrypt data or create digital signature
4️⃣	**Security Impact if Exposed**	Low (but must verify source authenticity)	High (data can be compromised if exposed)
5️⃣	**Storage**	Stored in public directories or sent with messages	Stored securely, usually in encrypted format
6️⃣	**Mathematical Relationship**	Mathematically linked to private key	Mathematically linked to public key
7️⃣	**Usage in Digital Signature**	Used for verifying signature	Used for creating signature
8️⃣	**Speed of Operation**	Generally slower in encryption (for large data)	Faster in decryption (when decrypting small chunks)

Private Key Management Kya Hai?

- Private Key Management matlab apni **private key ko safely handle karna**.
- Agar private key safe nahi rahegi, to encryption ya signature ka fayda nahi rahega.

Private Key Management ke Important Points:

1. **Secure Generation:** Strong method se key generate karo.
2. **Safe Storage:** Private key ko encrypted form mein store karo (e.g., password protected files, hardware tokens).
3. **Access Control:** Sirf trusted logon ko access do.
4. **Backup:** Key ka encrypted backup rakho.
5. **Rotation:** Time-time par key change karte raho.
6. **Revocation:** Agar key compromise ho jaye, to use turant invalidate karo.

🔧 Example of Private Key Management

Scenario: Ek company apne employees ke liye digital signatures use karti hai.

1.Key Generation:

1. Employee ka system ek strong algorithm (RSA) se private key generate karta hai.
2. Private key ek encrypted file me save hoti hai, sirf employee ke login credentials se access ho sake.

2.Storage:

1. Private key hardware token (USB device) me bhi store ho sakti hai, jo physical possession me ho.
2. Ya fir encrypted software vault (e.g., password-protected key store) me.

3.Access Control:

1. Employee ke alawa koi doosra us private key ko access nahi kar sakta.
2. Company IT policy me hai ki kisi doosre ko employee ki private key nahi deni.

4.Backup:

1. Encrypted backup company ke secure server par rakha gaya hai.
2. Agar employee ka device kharab ho jaye, backup se restore kiya ja sake.

5.Rotation and Revocation:

1. Har 1 saal baad employee ki private key renew karni hoti hai.
2. Agar employee resign kar jata hai, uski private key turant revoke kar di jati hai, taki wo misuse na kar sake.

Step	Example in Company Context
Generate Key	RSA algorithm se strong private key banaya
Store Securely	Encrypted file + hardware token me store
Access Control	Employee ke alawa koi access nahi
Backup	Encrypted backup secure server par rakha
Rotate & Revoke	Har saal key renew, resign hone par turant revoke

🔐 Diffie-Hellman Key Exchange Algorithm –

Diffie-Hellman ek algorithm hai jiska use do log (maan lo Alice aur Bob) ek common secret key banane ke liye karte hain — bina pehle se ek dusre ko jaane hue. Ye key later encryption ke liye use hoti hai.

🔢 Algorithm ke Steps

✅ Step 1: Ek Prime Number Lo

- Ek prime number x choose karo.
- Yahan diya gaya hai:

- $x=13$

✅ Step 2: Primitive Root Choose Karo (β)

- Ek number β choose karo jise:
  - $\beta < x$ ho
  - Aur β x ka primitive root ho.

📌 **Note: Primitive root ka matlab?**

> Agar $\beta^1, \beta^2, \beta^3, ..., \beta^{x-1} \mod x$ ek **non-repeating sequence** banaye jo sab values cover kare — toh usse primitive root kehte hain.

📌 Example diya hai image mein:

$$x = 7,\ \beta = 3$$

3^1 mod 7 = 3
3^2 mod 7 = 2
3^3 mod 7 = 6
3^4 mod 7 = 4
3^5 mod 7 = 5
3^6 mod 7 = 1

→ Sab values aa gayi (1 to 6) → To 3 is primitive root of 7 ✅

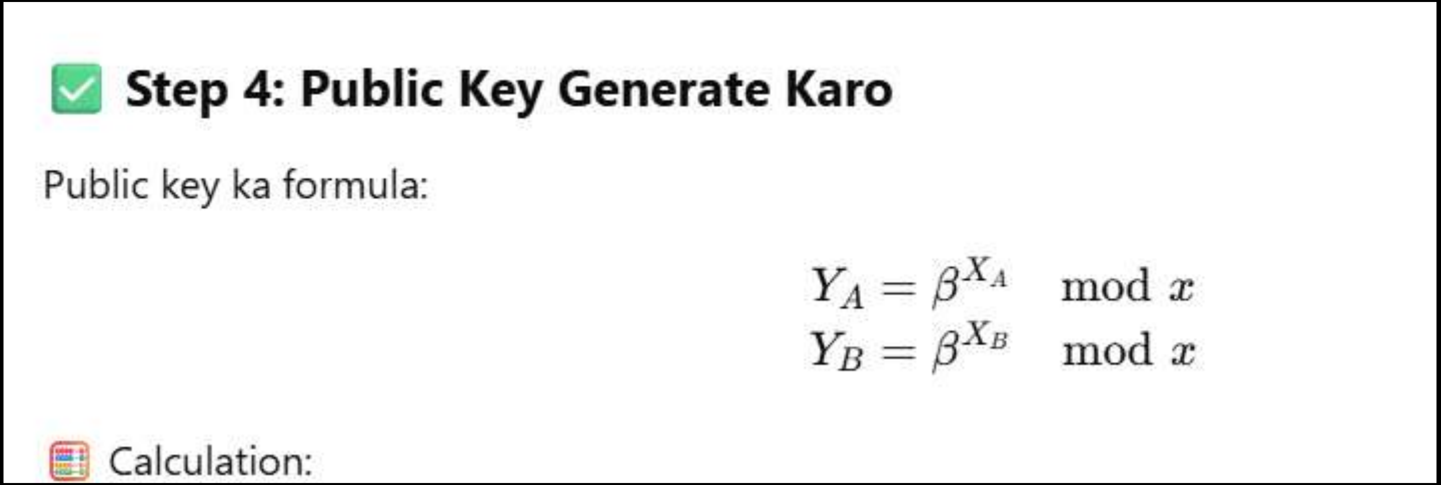
✓ Step 3: Private Key Choose Karo (Secretly)

- Dono log (maan lo Alice = A, Bob = B) apni private keys choose karte hain:

$$X_A = 5,\ X_B = 4$$

Condition:

$X_A < x, X_B < x$



✅ Step 4: Public Key Generate Karo

Public key ka formula:

$$Y_A = \beta^{X_A} \mod x$$
$$Y_B = \beta^{X_B} \mod x$$

🧮 Calculation:

Y_A = 3^5 mod 13 = 243 mod 13 = 9
Y_B = 3^4 mod 13 = 81 mod 13 = 3

✅ Step 5: Shared Secret Key Banao

Ab dono ek dusre ki public key se **same secret key** nikaalte hain:

$$Z_A = Y_B^{X_A} \mod x = 3^5 \mod 13 = 243 \mod 13 = 9$$
$$Z_B = Y_A^{X_B} \mod x = 9^4 \mod 13 = 6561 \mod 13 = 9$$

🎉 Final Result:

$$Z_A = Z_B = 9$$

Yeh hi hai **shared secret key** jo dono ke paas aa gayi — bina kisi ne actual private key share kiye!

Step	Description
1	Prime number xx lo
2	Primitive root β\beta lo
3	Private key XA,XBX_A, X_B lo
4	Public key YA,YBY_A, Y_B banao
5	Secret key ZA=ZBZ_A = Z_B banao

🔐 The PKIX Model (Public Key Infrastructure using X.509)

PKIX (Public Key Infrastructure X.509) ek **standard framework** hai jo **digital certificates** aur **public-key encryption** ke through **secure communication** provide karta hai — mainly **Internet** aur other untrusted networks par.

□ PKIX Kya Hai?

PKIX stands for:
Public Key Infrastructure (X.509)
Ye ek framework hai jo define karta hai ki kaise **public keys**, **digital certificates**, aur **certificate authorities (CAs)** ka use karke **trust** establish kiya jaye.

🔢 No.	▯ Component	📝 Explanation	📌 Example
1️⃣	**End Entities**	Users, devices, or software jo PKIX services ka use karte hain.	Web browsers, email clients
2️⃣	**Certificate Authority (CA)**	Trusted third party jo **digital certificates** issue karta hai after verifying identities.	DigiCert, Let's Encrypt
3️⃣	**Registration Authority (RA)**	CA ke behalf par identity verification karta hai before certificate issue hota hai.	A company’s internal IT department
4️⃣	**Certificate Repository**	Public database jahan **certificates** aur **revocation lists (CRL)** store hote hain.	LDAP server, HTTPS URLs

📄 X.509 Certificate Format

PKIX model **X.509** certificate format ka use karta hai:

🔹 Contents of an X.509 certificate:

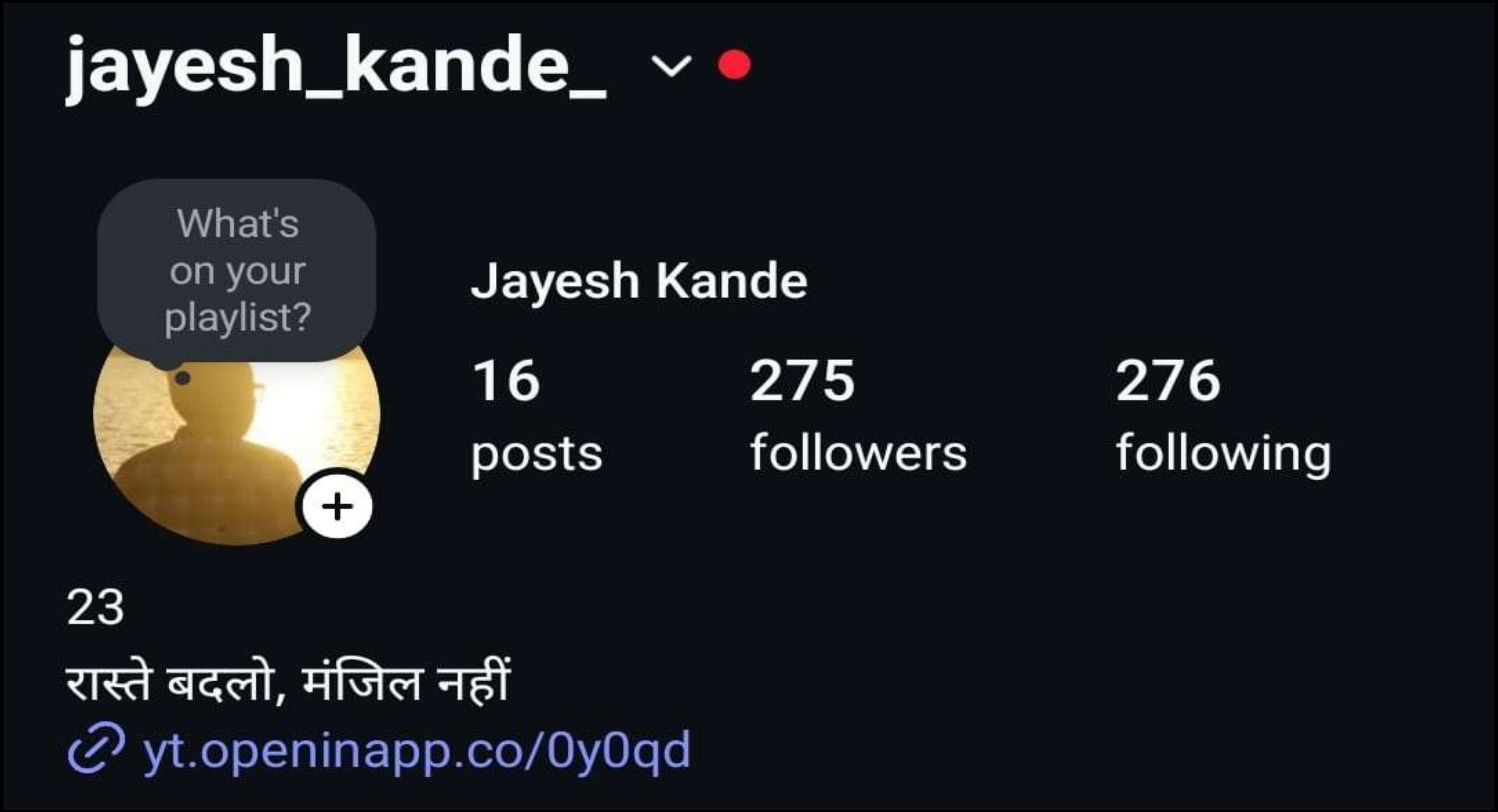
- Certificate holder ka naam
- Public key
- Certificate issuer (CA)
- Expiration date
- Digital signature (CA ki taraf se)

🔄 PKIX Workflow Summary

1. **User requests** certificate from RA.
2. **RA verifies** user identity and forwards request to CA.
3. **CA generates** and digitally signs certificate.
4. Certificate **stored in repository** and given to user.
5. Any third party can now **verify** that public key belongs to the correct identity.

📌 PKIX Use Cases

•HTTPS (SSL/TLS certificates for websites)
•Secure Email (S/MIME)
•Code Signing
•VPN authentication



jayesh_kande_

What's on your playlist?

Jayesh Kande

16 posts

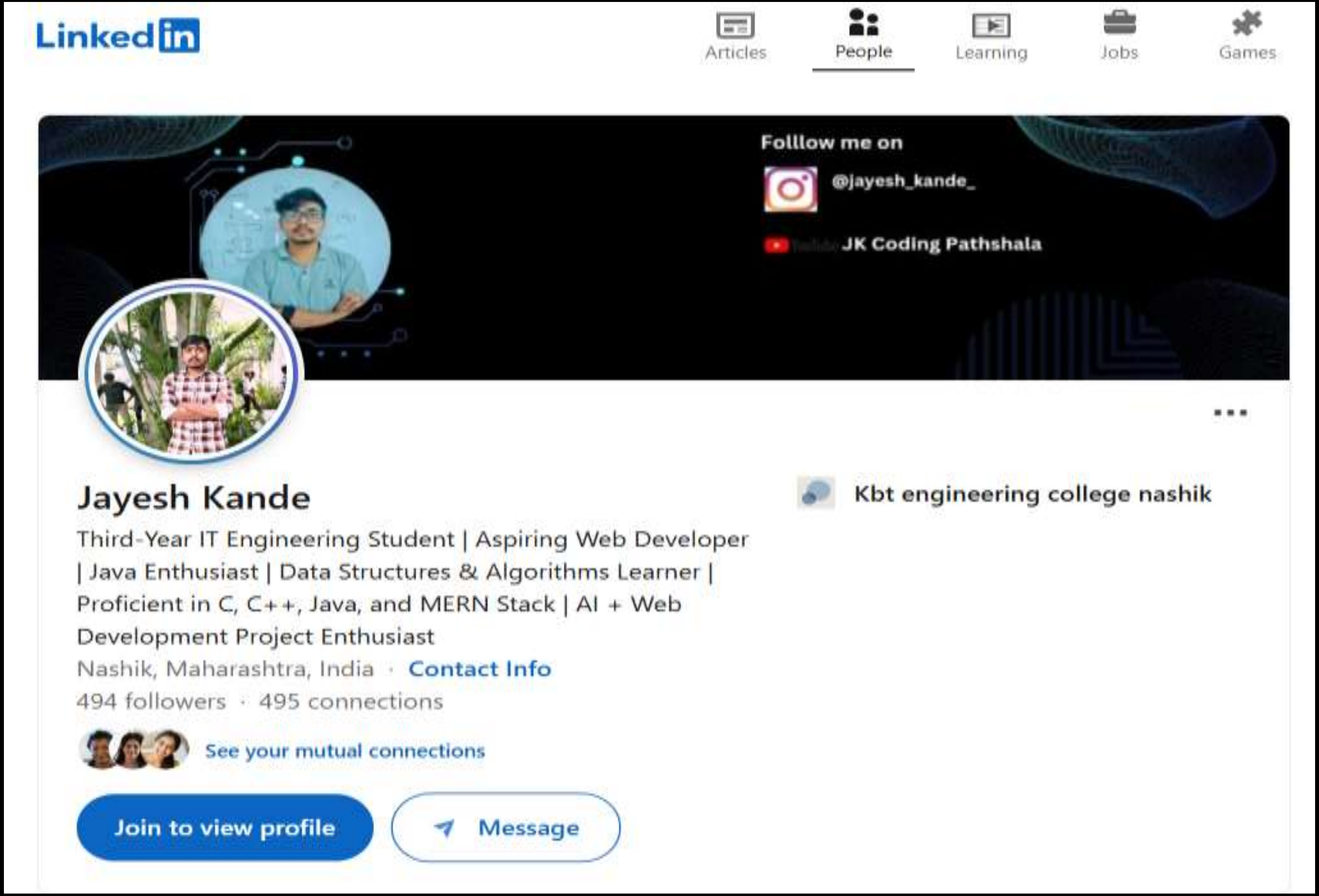
275 followers

276 following

23

रास्ते बदलो, मंजिल नहीं

yt.openinapp.co/0y0qd



Jayesh Kande

Third-Year IT Engineering Student | Aspiring Web Developer | Java Enthusiast | Data Structures & Algorithms Learner | Proficient in C, C++, Java, and MERN Stack | AI + Web Development Project Enthusiast

Nashik, Maharashtra, India · Contact Info

494 followers · 495 connections

See your mutual connections

Kbt engineering college nashik

✨ Thank You for Watching! ✨

➔📱 Follow us on Instagram:**@jayesh_kande_**

🔗 Connect with us on LinkedIn:[**Jayesh Kande**]