Bindu

Answer to the question 1

Threat to RSA and ECC :-

1) Shor's Algorithm purpose

A quantum algorithm that can efficiently factor large integers and solve discrete logarithm problems.

→ RSA vulnerability :-

- RSA security is based on the difficulty of factoring large integers.
- Shor's algorithm can factor this large numbers efficiently on powerful quantum computer → breaking RSA encryption.

→ ECC vulnerability :-

- ECC security relies on the Elliptic Curve Discrete Logarithm Problem (ECDLP)

Bindu

- Short's algorithm can also solve ECDLP quickly → breaking Ecc based schemes like ECDSA and ECDH.

Potential consequences for Digital Infrastructure

~~Infructure~~

1) Breakage of public key encryption.
2) Attacken could forge digital signatures, breaking authentication.
3) Threat to stored encryption data.
4) Impact on breaking, E-commerce and government systems.
5) Loss of confidentiality.

Bindu

Answer to the Question 2

* Role of Quantum key Distribution (QKD) in future Cryptographic System.

① Generates encryption keys using quantum mechanics.

② Offers unbreakable security against any computational power.

③ Detects eavesdropping instantly. (अनधिकार)

④ Protect data from quantum computer attacks.

⑤ Works well with one-time pad for perfect secrecy.

⑥ Future proof solution for sensitive communication.

* Difference between QKD and classical public key encryption :-

QKD	Classical public key Encryption
① Law of quantum mechanics	① Mathematical problem (Factoring, discrete log)
② Resistent to quantum attack.	② Vulnerable to Shor's algorithm and other quantum attacks
③ key exchanged via quantum channels.	③ key exchanged via classical communication over public channels
④ Detection of eavesdropping possible	④ Detection of eavesdropping is not possible.
⑤ no computational assumption.	⑤ Relies on computational hardness assumptions.
⑥ Require special quantum handware and fiber/ optical links.	⑥ Works with existing internet infrastructure.
⑦ Currently slower and distance limited.	⑦ Fast and long distance capable.

Bindu

Answer to the question 3

* Different between Lattice-Based cryptography and traditional number theoretic approaches (e.g RSA)

Lattice based cryptography	Traditional number theoretic approach.
① Based on handness of lattice problems (e.g Short vector problem, Learning with errors)	① Based on factoring large integers RSA or Solving discrete Logarithms (ECC)
② Considered secure against both classical and quantum attacks.	② Vulnerable to quantum algorithms like Short's algorithm.
③ Works in high dimentional vecton spaces.	③ works in modular arithmetic over integers or finite fields.
④ Large key compaire to RSA/ECC.	④ Smaller key for the same classical security.
⑤ Often faster key generation and encryption. Some schemes slower in decryption.	⑤ Mature and well Optimized for performance.

⑥ Actively being standardized by NIST for post-quantum cryptography.	⑥ Already standardized and widely deployed.
⑦ Expected to replace RSA/ ECC in a post-quantum era.	⑦ Likely to be phased out once quantum computers are practical

Answer to

Bindu

Answer to the question 25

Seive of Eratosthenes Algorithm

The Sieve of Erathosthenes is an efficient method for finding all prime numbers up to a given limit n.

Steps:-

1. Create a list of number from 2 to n.
2. Start with the first prime $p = 2$
3. Eliminate all multiples of p greater than or equal to p^2
4. Move to the next number that is still marked as prime.
5. Repeat untill $p^2 > n$
6. Remaining unmarked numbers are prime.

Example:- Find all primes less than 50.

1) Start: numbers 2 to 49

② $p=2 \rightarrow$ Remove multiples of 2: 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, ... 48

③ $p=3 \rightarrow$ Remove multiples of 3: 9, 12, 15, ... 48.

④ $p=5 \rightarrow$ Remove multiples of 5: 25, 30, 35, 40, 45.

⑤ $p=7 \rightarrow$ Remove multiples of 7: 49

⑥ Remaining numbers are: prime < 50

→ 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47.

Time complexity Comparison:-

method	Time complexity	
Sieve of Eratosthenes	$O(n \log \log n)$	very efficient for generating many primes at once.
Trial Division	$O(n\sqrt{n})$ for all numbers.	Slower, checks each number individually for primality.

Bindu

Answer to the question 6

Korselt's Criterion (necessary and sufficient condition :-

A composite integer n is a carmichael number if all three hold :-

1) n is composite number.

2) n is squire-free (no prime square divides n)

3) For every prime p dividing n (p-1)|(n-1).

(use this criterion to test numbers; it is equivalent to the carmichael definition $a^{n-1} \equiv 1 \pmod{n}$ for all a, coprime to n.

Test 1 :- n = 561

1) Factorization : $561 = 3 \times 11 \times 17$ composite.

2) Square-free ? yes (primes are distinct)

3) $n-1 = 560$. check, $(p-1)/(n-1)$ for each prime divison.

- For $p=3$; $p-1=2$, $560/2 = 28 \rightarrow$ divides.
- For $p=11$; $p-1=10$, $560/10 = 56 \rightarrow$ divides.
- For $p=17$; $p-1=16$ $560/16 = 35 \rightarrow$ divides.

All conditions satisfied. 561 is a carmichael number.

~~Test-2~~.

Test-2; $n=1105$

1) Factorization : $1105 = 5 \times 13 \times 17 \rightarrow$ composit.
2) Squre free.
3) $n-1 = 1104$. Check divisibility.

- P=5; P-1=4; 1104/4 = 276 → divides.
- P=13; P-1=12. 1104/12 = 92 → divides.
- P=17; P-1=16 1104/16 = 69 → divides.

All conditions Satisfied → 1105 is Carmichael number.

Test-3, n=1729

1) Factorization; 1729 = 7 × 13 × 19 → composite.
2) square free.
3) n-1 = 1729-1 = 1728 check divisibility.
   - P=7, P-1=6, 1728/6 = 288 → divides.
   - P=13, P-1=12; 1728/12 = 144 → divides.
   - P=19; P-1 = 18 1728/18 = 96 → divides.

All conditions Satisfied → 1729 is a ~~cha~~ carmichael number.

Bindu

Answer to the question 7

1) Is $(2_{11}, +, \times)$ a ring?

→ Yes, in fact it is a commutative ring with unity, and moreover a field because 11 is prime.

→ Justify (brief):-

- Addition:- 2_{11} is closed under +, associative has identity 0, every element a has additive inverse 11-a and addition is commutative ⇒ $(2_{11}, +)$ is an abelian group.
- Multiplication:- × is closed, associative and commutative, multiplicative identity $1 \in 2_{11}$ exists.

Distributivity:- multiplication distributes

over addition.

- Field property :- Because 11 is prime, every non zero element has a multiplicative inverse mod 11. Hence a field ⇒ a in particular ring.

So; Z_{11} satisfy all ring axioms (indeed Stronger; field).

2) Is $(Z_{37}, +)$ an Abelian group?

→ Yes;

Reason

- Closure under + (mod 37)
- Associativity holds (integer addition).
- Identity 0 exists.
- Every $a \in Z_{37}$ has additive inverse $37 - a$
- Commutative : $a + b = b + a$.

Hence $(Z_{37}, +)$ is an abelian (cyclic) group of order 37.

3) Is $(Z_{35}, \times)$ an Abelian group?

→ No.

→ counterarguments.

- Although multiplication mod 35 is closed, associative and commutative and has identity 1, not every element has a multiplicative inverse.
- Example, $7 \in Z_{35}$. If 7 had an inverse x then, $7x \equiv 1 \pmod{35}$. But $\gcd(7, 35) = 7 \neq 1$. So, no solution exist. Also 0 has no invers.
- Therefore the inverse axiom fails and $(Z_{35}, \times)$ is not a group.

Bindu

(The Subset of unit $2_{35}^{\times} = \{ a : \gcd(a, 35) = 1 \}$ would be a group, but the whole 2_{35} under $\times$ is not.

Answer to the question 8

we will find $-52 \bmod 31$.

1) First divide -52 by 31

- $-52 \div 31$ gives quotient -2 ~~(since = 2~~
- Since, $-2 \times 31 = -62$

2) now find the remainder :-

$$-52 - (-62)$$
$$= -52 + 62$$
$$= 10$$

Answer is 10.

Answer the question 9

We will use Extended Euclidean Algorithm to find the inverse of $-7 \bmod 26$.

We need x such that

$$7x \equiv 1 \pmod{26}$$

Step – 1 : Apply Euclidean Algorithm :–

$$26 = 3 \times 7 + 5$$

$$7 = 1 \times 5 + 2$$

$$5 = 2 \times 2 + 1$$

$$2 = 2 \times 2 + 0$$

Since GCD is 1, inverse exists.

Step – 2 : Back Substitute :–

From, ~~$5 = 7 - 1 \times 5$~~ $5 = 26 - (3 \times 7)$

From, $2 = 7 - 1 \times 5$

From $1 = 5 - 2 \times 2$

Substitute,

~~$2 = 7 - 1 \times (7 - 1 \times 5$~~

$2 = 7 - 1(26 - (3 \times 7))$

$= 7 - 26 + 3 \times 7$

$\therefore 2 = 4 \times 7 - 26$

Now, $1 = 5 - 2 \times 2$

$1 = (26 - 3 \times 7) - 2(4 \times 7 - 26)$

$1 = 26 - 3 \times 7 - 8 \times 7 + 2 \times 26$

$1 = 3 \times 26 - 11 \times 7$

Step-3:- Identify inverse:-

$1 \equiv -11 \times 7 \bmod 26$

$\therefore -11 \times 7 \equiv 1 \bmod 26$

Bindu

So, inverse of 7 mod 26 is,

$-11 + 26 = 15$

So, $15 \times 7 \equiv 105 \equiv 1 \pmod{26}$.

This means the multiplicative inverse of 7 mod 26 is 15.

Answer to the question 10

Step-1:- multiply the numbers,

$(-8) \times 5 = -40$

Step-2: Reduce modulo 17

$-40 \bmod 17$

Since modulo results are usually taken as non-negative remainder, we add multiples of 17 until the result become positive,

Bindu

$-40+17 = -23$ (Still negative)

$-23+17 = -6$ (still negative)

$-6+17 = 11$ positive.

Final Answer $= (-8 \times 5) \bmod 17 = 11$.

Answer to the question 11

Bezout's theorem:-

Statement:-

If a and b are integer and $\gcd(a, b) = d$

Then the existing integer x and y such that

$$ax + by = d$$

if $d = 1$ then,

$$ax + by = 1$$

in this case, x is multiplicative inverse of a modulo b.

Proof :-

1) From GCD definition :-

The set of all integen of the form $ax+by$ contains $gcd(a,b)$

2) Using Euclidean Algorithm :-

we can find $gcd(a,b)$ and write it as a combination of a and b.

3) If gcd = 1,

The equation become $ax+by=1$.

take modulo b, $ax \equiv 1 \pmod{b}$ meaning x is the inverse of a mod b.

Find the multiplicative inverse of 97 mod 385.

we want x such that:

$$97x \equiv 1 \pmod{385}$$

on, $97x + 385y = 1$

Step-1: Apply Extended Euclidean Algorithm.

1. $385 = 3 \times 97 + 94$
2. $97 = 1 \times 94 + 3$
3. $94 = 31 \times 3 + 1$
4. $3 = 3 \times 1 + 0$

Step-2: Back Substitution:-

$$94 = 385 - 3 \times 97$$

$$3 = 97 - 1 \times 94$$

$$1 = 94 - 31 \times 3$$

Substitution,

$$1 = \overset{94}{(385 - 3 \times 97)} - 31(97 - 1 \times \overset{94}{94})$$

$$= 385 - 3 \times 97 - 97 \times 31 + 94 \times 31$$

$1 = 94 \times 32 - 97 \times 31$

$= 32(385 - 3 \times 97) - 97 \times 31$

$= 385 \times 32 - 96 \times 97 - 97 \times 31$

$\therefore 1 = 385 \times 32 - 127 \times 97$

Take modulo;

$-127 \times 97 \equiv 1 \bmod 385$

Since we need a positive inverse,

$-127 \pmod{385} \Rightarrow 385 - 127 = 258$

So,

Inverse of 97 mod 385 is 258.

Ans;

Answer to the Question 12.

1) Bezout's identity :-

Statement :-

For any integers a, b not both zero, there exist integers x, y such that,

$$ax + by = \gcd(a, b)$$

Proof :-

- Run the Euclidean algorithm to compute $d = \gcd(a, b)$. This gives a sequence of divisions with remainders :-

$$a = q_1 b + r_1, \quad b = q_2 r_1 + r_2$$

$$r_1 = q_3 r_2 + r_3$$

until the remaind is zero, the last nonzero remainder is d.

- work backwards, each remainder is an integer linear combination of a and b.

when you express the last nonzero remainder in terms of the previous remainders and substitute back repeatedly, you obtain integers x, y with $ax+by=d$.

- Thus Bezout coefficients exist and the Euclidean algorithm provides them explicitly.

2) Find x such that $43x \equiv 1 \pmod{240}$

we need integers x, y with

$$43x + 240y = 1$$

Applying the extended Euclidean algorithm,

Step-A → Euclidean algorithm (compute gcd)

$$240 = 5 \times 43 + 25$$

$$43 = 1 \times 25 + 18$$

$$25 = 1 \times 18 + 7$$

$$18 = 2 \times 7 + 4$$

$$7 = 1 \times 4 + 3$$

$$4 = 1 \times 3 + 1$$

$$3 = 3 \times 1 + 0$$

Step-B :- Back Substitute

$1 = 4 - 1 \cdot 3$

$3 = 7 - 1 \cdot 4$

$1 = 4 - 1 \cdot (7 - 1 \cdot 4)$

$= 4 - 7 + 1 \cdot 4$

$1 = 2 \cdot 4 - 1 \cdot 7$

$4 = 18 - 2 \cdot 7$

$\Rightarrow 1 = 2 \cdot (18 - 2 \cdot 7) - 1 \cdot 7$

$= 2 \cdot 18 - 4 \cdot 7 - 1 \cdot 7$

$= 2 \cdot 18 - 5 \cdot 7$

$7 = 25 - 1 \cdot 18$

$\Rightarrow 1 = 2 \cdot 18 - 5 (25 - 1 \cdot 18)$

$= 7 \cdot 18 - 5 \cdot 25$

• $18 = 43 - 1 \cdot 25$

$\Rightarrow 1 = 7(43 - 1.25) - 5.25$

$= 7 \cdot 43 - 12 \cdot 25$

• $25 = 240 - 5 \cdot 43$

$\Rightarrow 1 = 7 \cdot 43 - 12 \cdot (240 - 5 \cdot 43)$

$= 7 \cdot 43 + 60 \cdot 43 - 12 \cdot 240$

$1 = 67 \cdot 43 - 12 \cdot 240$

→ Reduce modulo 240:

$\therefore 43 \cdot 67 \equiv 1 \mod 240$

$\therefore x = 67$

$\therefore x \equiv 67 \pmod{240}$

Ans

Bindu

Arswen to the Question 13

prove Fermat's Little Theorem and explain how it used for test primality. Is 561 a prime number based on this test?

→ If p is prime and a is not divisible by p then $a^{p-1} \equiv 1 \bmod p$.

Proof:-

→ Consider the numbers a, 2a, 3a --- (p-1)a

→ These numbers are distinct modulo p (since p is prime and gcd(a,p) = 1

→ Their product is congruent to (p-1)! mod p

a, 2a, 3a, ... (p-1)a ≡ (p-1)! mod p.

$a^{p-1}(p-1)! \equiv (p-1)! \bmod p$

→ Since (p-1)! is not divisible by p, we can cancel it $a^{p-1} \equiv 1 \bmod p$.

Fermat's primality Test:-

→ For a candidate n, pick a random a where

$1 < a < n$.

→ If $a^{n-1} \not\equiv 1 \bmod n$, n is composite.

→ If $a^{n-1} \equiv 1 \bmod n$, n is likely prime

Is 561 prime?

$561 = 3 \times 17 \times 11$

So, 561 is composite.

Now check for fermets behaviour:

$561 - 1 = 560$. For base a co-primes to 561, one can show that $a^{560} \equiv 1 \pmod{561}$

Indeed 561 is a carmichael number because it squrefree and for each prime divisor p of 561, $p-1$ divides 560

$3-1 = 2 \mid 560$, $11-1 = 10 \mid 560$, $17-1 = 16 \mid 560$

By the carmicheal property every a

with gcd $(a, 561) = 1$ will satisfy.

$a^{560} \equiv 1 \mod (561)$. So, for such bases Fermat's test declares probably prime.

Evaluate $5^{123} \mod 175$:-

Step1 : Factorize 175, $175 = 5^2 \times 7$

Step2 : compute $5^{123} \mod 25$

Since $25 = 5^2$ for $k \geq 2$, $k^2 \equiv 0 \mod 25$

thus, $5^{123} \equiv 0 \pmod{25}$

Step-3 : compute $5^{123} \mod 7$

By Fermats Little Theorem $5^6 \equiv 1 \mod 7$

write $123 = 6 \times 20 + 3$

$5^{123} \equiv (5^6)^{20} \times 5^3 = 1^{20} \times 125 \equiv 6 \mod 7$

Step-4 :- combine Result via CRT

we need x such that.

$x \equiv 0 \mod 25, \quad x \equiv 6 \mod 7$

Let, $x = 25k$ then

$25k \equiv 6 \mod 7$

$4k \equiv 6 \mod 7$

$k \equiv 5 \mod 7$

Thus $k = 7m + 5$ and

$x = 25(7m+5)$

$= 175m + 125$

The smallest non-negative solution is 125

Bindu

Answer to the question 14

Chinese Remainder Theorem (CRT)

Statement:- If $m_1, m_2 \cdots m_k$ are pairwise co-prime positive integers and we have the system

$$x \equiv a_1 \pmod{m_1}$$
$$x \equiv a_2 \pmod{m_2}$$
$$\vdots$$
$$x \equiv a_k \pmod{m_k}$$

Then there exists a unique solution modulo

$$M = m_1 m_2 m_3 \cdots m_k.$$

Proof:-

1) Since m_i are coprime each $M_i = \frac{M}{m_i}$ is coprime with m_i

2) By Bezout's identity, there exists y_i such that $m_i y_i \equiv 1 \pmod{m_i}$

3) The solution is,

$$x \equiv \sum_{i=1}^{k} a_i m_i y_i \pmod{M}$$

④ This x satisfies all congruences and is unique modulo M.

Math

Given system,

$$x \equiv 2 \pmod 3$$
$$x \equiv 3 \pmod 5$$
$$x \equiv 2 \pmod 7$$

we know,

$$x = (a_1 M_1 M_1^{-1} + a_2 M_2 M_2^{-1} + a_3 M_3 M_3^{-1}) \bmod m$$

Here $a_1 = 2,\ a_2 = 3,\ a_3 = 2$

$$m_1 = 3,\ m_2 = 5,\ m_3 = 7$$

$$M = m_1 m_2 m_3 = 3 \times 5 \times 7 = 105$$

Now,

$$M_1 = \frac{105}{3} = 35, \quad M_2 = \frac{105}{5} = 21$$

$$M_3 = \frac{105}{7} = 15$$

$M_1 \times M_1^{-1} \equiv 1 \pmod{m_1}$

or, $35 \times M_1^{-1} \equiv 1 \bmod 3$

or, $35 \times (2) \equiv 1 \bmod 3$

$\therefore M_1^{-1} = 2$

again

$m_2 \times m_2^{-1} \equiv 1 \bmod m_2$

$21 \times 1 \equiv 1 \bmod 5$

So,

$m_2^{-1} = 1$

And

$m_3 \times m_3^{-1} \equiv 1 \bmod m_3$

or, $15 \times m_3^{-1} \equiv 1 \bmod 7$

or, $15 \times 1 \equiv 1 \bmod 7$

$\therefore m_3^{-1} = 1$

So,

$$X = a_1 M_1 M_1^{-1} + a_2 M_2 M_2^{-1} + a_3 M_3 M_3^{-1}$$

$$= 2 \times 35 \times 2 + 3 \times 21 \times 1 + 2 \times 15 \times 1$$

$$= 140 + 63 + 30$$

$$X = 233 \mod 105$$

or, $233 \mod 105$

$233 = 105 \times 2 = 23$

$\therefore X = 23 \pmod{105}$

Ans:

Bindu

Answer to the Question 15

CIA Triad in information Security.

→ The CIA Triad is fundamental model for designing and evaluating the Security of information System. It consists of 3 core principles.

1) Confidentiality.

- Ensures that information is accessible ~~also~~ only to authorized users.
- Prevent unauthorized disclosure of sensitive data.
- Achieved through methods like encryption, access controls and authentication.

2) Integrity:-

- ~~Coun~~ Guarantees that info is accurate consistent and unaltered except by

authorized entries.

- Protects against data corruption.
- Achieved through hashing, checksums, digital signature and version control.

3) Availability :-

- Ensures that information and resource are accessible when needed.
- Protects against service disruptions caused by failures or Attacks.
- Achieved through redundency, backups, load balancing and disaster recovery Plans.

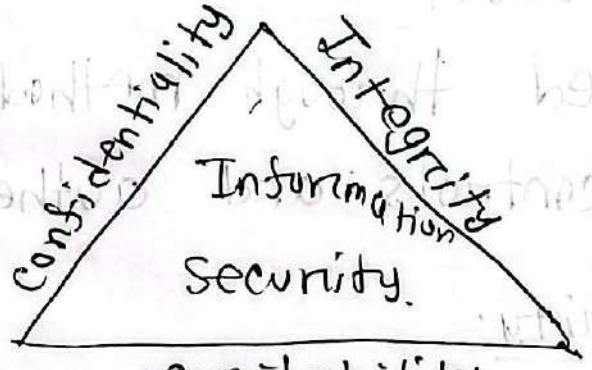


Fig- CIA Triad.

Bindu.

How the world togather;-

- Confidentiality :- keep secrete safe.
- Intigrity keeps truth intact.
- Availability keeps services reliable.

Answer to the Question 16

Difference between Steganography and Cryptography

Steganography	Cryptography
① Hides the existence of the message.	① Protect the content of the message.
② Message is invisible within another medium.	② Ciphertext is visible but unreadable without key.
③ Concealment techniques	③ Mathematical algorithm and keys.
④ Hard to detect if done well.	④ Easy to detect but hard to read Encrypted data.
⑤ Hiding a text file inside an image.	⑤ Encrypting a message with AES or RSA.

⑥ Require a cover medium.	⑥ works without an cover medium.
⑦ If detected, Hiden data can be extracted easily.	⑦ If key is comprom message can be read

Common Techniques for Hiding Data in digital media.

1) LSB insertion :- Replace least Significant bits of image/audio with secret bits.

2) Frequency domain embedding :- Hide data in DCT/DWT coefficients.

3) Text Steganography :- use spacing, font changes, or invisible characters.

4) Audio/Video steganography :- Embed data in sound or video frames.

5) Metadata Hiding :- Storie data in unused file metadata fields.

Bindu

Answer to the question 17

Difference between phishing, malware and Denial-of-Service (DoS) Attack.

Feature	Phising	Malware	Denial of Service (DoS)
Method	1) Tricks users to share info using fraudulent email, websit on msg.	1) Malicious software to damage, steal or control data.	1) Overloads target system to make it unavailable.
Impact	2) Loss the personal information, identity theft.	2) Data theft, connuption or complete system compromise.	3) Service disruption, downtime, loss of availability.
Target	3) Individuals orgs	3) System, networks	3) Servers, networks.
Prevention	4) User awarness training, email filtering, URL veriphication.	4) Antivirus, patch management and secure downloads.	4) Firewalls, rate limiting, Dos mitigation tools.
Delivery	5) Fack mails and sites.	5) Damage or steal data.	5) massive traffic flood.

Bindu

Answer to the question 18

How GDPR help mitigate cyber attacks and protect user privacy.

1) Data protection principles → Enforces rules for lawful, fair and transparent process of personal data.

2) User Consent :— Requires explicit consent before collecting or processing user data.

3) Data minimization :- Limit collection to only necessary data, reducing exposure risk,

4) Breach notification :- mandates reporting of data breaches within 72 hours to authorities and users.

5) Security measure :- Requires organizations to implement technical and organizational safeguards.

6) Right to access and Erasure:- Give user control over their personal data.

7) Heavy penalties:- Impose large fine for non compliance, motivating stronger security

(Bindu)

Answer to the Question 19

DES (Data Encryption Standard) Basic working using 64 bit plaintext and 56 bit key.

1) Input block and key :-

- Plaintext : 64 bits (P= 0123456789ABCDEF in hex)
- Key :- 64 bits ~~at~~ input ; but only 56 bits are used (8 bits for parity)

2) Initial permutation (IP) :-

- The 64-bit plaintext undergoes initial permutation using a fixed table.
- The bits are rearranged according to the IP table.
- Output : Two 32 bit halves L_0 and R_0.

3) 16 Round of Feistel Structure :-

Each round has :-

- Expansion (E)
- key mixing (X-OR)
- Substitution (S-box)
- permutation (P)

Steps per round

a) Expansion :-

- R(n-1) (32 bits) → Expanded to 48 bits using expansion table which introduce redundency.

b) key mixing :-

- The 48 bits expanded ER(n-1) is XORed with 48 bit subkey K(n).
- Subkey are generated from the 56-bit main key, one for each round.

c) Substitution (S-box)

- The X-OR result is divided int 8 blocks of the 6-bits.

- Each bits goes through an S-box, outputting 4 bits each.
- Final result : 32 bits.

d) Permutation :-

- The 32 bit output from S-boxes is permuted again using a fixed table.

e) Feistel Swap :-

- $L(n) = R(n-1)$
- $R(n) = L(n-1) \cdot XOR\ f(R(n-1), k(n))$

4) Repeat for 16 Rounds :-

- Each round uses a different 48-bit subkey generated from the original key using PC-1, Shifts, and PC-2 tables.

5) Final permutation (FP) :-

- After 16 rounds, R_{16} and L_{16} are swapped and then the final permutation (inverse of PI)

is applied.

- The result is 64-bit ciphertext.

Answer to the Question 20

DES round function Calculation;-

Given that,

$R_0 = 0xF0F0F0F0$

$K_1 = 0x0F0F0F0F$

$L_0 = 0xAAAAAAAA$

Step 1 :-

$f(R_0, K_1) = R_0 \text{ XOR } K_1$

Let's convert both to binary first,

$R_0 = 0xF0F0F0F0$

$= 11110000\ 11110000\ 11110000\ 11110000$

$K_1 = 0x0F0F0F0F$

$= 00001111\ 00001111\ 00001111\ 00001111$

Now XOR bit by bit

$$\begin{array}{l} 1111\ 0000\ (R_0) \\ \oplus\ 0000\ 1111\ (K_1) \\ \hline 1111\ 1111 \rightarrow FF \end{array}$$

Repeate for all 4 bytes,

$$f(R_0, K_1) = 0xFFFFFFFF$$

Step-2 :-

$$R_1 = L_0 \text{ XOR } f(R_0, K_1)$$

$$L_0 = 0xAAAAAAAA$$

$$= 10101010\ 10101010\ 10101010\ 10101010$$

$$f(R_0, K_1) = 0xFFFFFFFF$$

$$= 11111111\ 11111111\ 11111111\ 11111111$$

Now XOR

$$\begin{array}{l} 10101010 \\ \oplus\ 11111111 \\ \hline 01010101 \rightarrow 55 \end{array}$$

Repeat for all 4 bytes

Bindu

$R_1 = 0x55555555$

Step-3 :- $L_1 = R_0 = 0xF0F0F0F0$

Final Answer

- $f(R_0, K_1) = 0xFFFFFFFF$
- $L_1 = R_0 = 0xF0F0F0F0$

  $R_1 = 0x55555555$

Answer to the question 21

input word :- [0x23, 0xA7, 0xUC, 0x19]

S-box provided

Row\ Col →	3	4	7	9	A	C
1	6D	*	*	C6	*	*
2	D4	*	*	*	*	*
4	*	A1	*	*	*	2E
A	*	*	63	*	D2	*

Bindu

Now lookup

1) 0x23 → Row = 2, Col = 3 → D4

2) 0x A7 → Row = A, Col = 7 → (~~not given, Not Available~~)
= 63

3) 0x 4C → Row = 4, Col = C → 2F

4) 0x19 → Row = 1, Col = 9, → (~~not give, not Available~~)
= C6

Final Answer output word = [D4, N/A, 2E, N/A]
= [D4, 63, 2E, C6]

Answer to the Question 22

AddRoundKey

Given,

Input word :-

0x1A, 0x2B, 0x3C, 0x4D

Round key word :-

0x55, 0x66, 0x77, 0x88

8421

1st Byte:-

Input byte = 0x1A = 00011010

Round key = 0x 55 = 01010101

X-OR result = 01001111

= 0x 4F

2nd Byte:-

Input byte = 0x 2B = 00101011

Round key = 0x 66 = ~~0110~~ 01100110

XOR result = 01001101

= 0x 4D

3rd Byte:-

Input byte = 0x 3C = 00111100

Round key = 0x 77 = 01110111

X-OR result = 01001011

= 0x 4B

Bindu

4th byte:-

Input byte = 0x4D = 01001101

Round key = 0x88 = 10001000

XOR result = 11000101

= 0xC5

After AddRoundKey output = [0x4F, 0x4D, 0x4B, 0xC5]

Answer to the question 23

AES MixColumns:-

matrix ;
$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

Input column =
$$\begin{bmatrix} 0x01 \\ 0x02 \\ 0x03 \\ 0x04 \end{bmatrix}$$

multiply Two matrix.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} 0x01 \\ 0x02 \\ 0x03 \\ 0x04 \end{bmatrix}$$

$$= \begin{bmatrix} (02\times01)\oplus(03\times02)\oplus(01\times03)\oplus(01\times04) \\ (01\times01)\oplus(02\times02)\oplus(03\times03)\oplus(01\times04) \\ (01\times01)\oplus(01\times02)\oplus(02\times03)\oplus(03\times04) \\ (03\times01)\oplus(01\times02)\oplus(01\times03)\oplus(02\times04) \end{bmatrix}$$

Byte-1

$02\times01 = 0x02$, $03\times02 = 0x06$, $01\times03 = 0x03$

$01\times04 = 0x04$

$\Rightarrow 0x02 \oplus 0x06 \oplus 0x03 \oplus 0x04$

Here,

$0x02(0010) \oplus 0x06(0110) = 0100$

$= 0x04$

$0x03(0011) \oplus 0x04(0100) = 0111 = 0x07$

Now,

$0x04(00000100) \oplus 0x07(00000111) = 0000\ 0011$

$= 0x03$

Final $= 0x03$

Byte -2

~~$01 \times 01 = 0x01$~~, $02 \times 02 = 0x04$, ~~$03 \times 03 = 0x09$~~

$01 \times 04 = 0x04$, $03 \times 03 = (0x02 + 0x01) * 0x03$

$= (0x02 \times 0x03) \oplus (0x01 \times 0x03)$

$= 0x06 \oplus 0x03$

$= 0110 \oplus 0011$

$= 0101 = 0x05$

$= 0x01 \oplus 0x04 \oplus 0x05 + 0x04$

$= 0x0001 \oplus 0x0100 \oplus 0x0101 + 0x0100$

$= 0x0100$

$= 0x04$

(21)

জোড় = 0
বিজোড় = 1

8421
0100

Byte-3

$(0x01 \times 0x01) \oplus (0x01 \times 0x02) \oplus (0x02 \times 0x03) + (0x03 \times 0x04)$

$= 0x01 \oplus 0x02 \oplus 0x06 + (0x02 + 0x01) \times 0x04$

$= 0x01 \oplus 0x02 \oplus 0x06 + (0x02 \cdot 0x04) \oplus (0x01 \times 0x04)$

$= 0x01 \oplus 0x02 \oplus 0x06 + 0x08 \oplus 0x04$

$= 0x00001 \oplus 0x00010 \oplus 0x00110 \oplus 0x01000$
$+ 0x00100$

$= 0x1001 = 0x09$

Byte-4

$(0x03 \times 0x01) \oplus (0x01 \times 0x02) \oplus (0x01 \times 0x03) + (0x02 \times 0x04)$

$= 0x03 \oplus 0x02 \oplus 0x03 + 0x08$

$= 0x0011 \oplus 0x0010 \oplus 0x0011 + 0x1000$

$= 0x1010 = 0xA$

Output = 0x03, 0x04, 0x09, 0x0A

Bindu

Answer to the question 24

what is AES-OFB:-

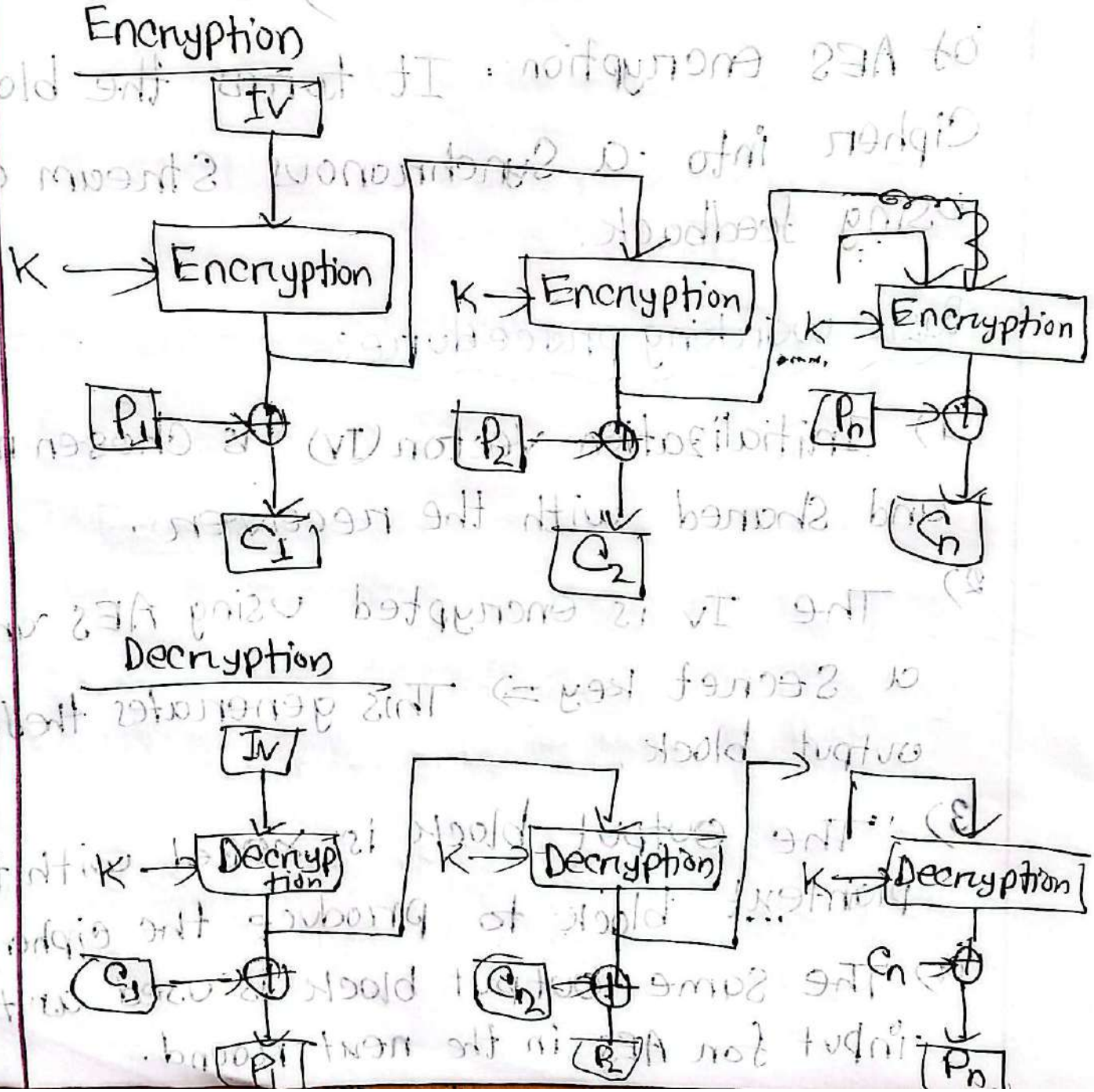
OFB (Output Feedback Mode) is a Stream Cipher of AES encryption. It turns the block Cipher into a Synchronous Stream cipher using feedback.

Basic working procedure:-

1) Initialization Vector (IV) is chosen randomly and shared with the receiver.
2) The Iv is encrypted using AES and a secret key ⇒ This generates the first output block.
3) The output block is XORed with the plaintext block to produce the ciphertext.
4) The same output block is used as the input for AES in the next round.

5) This repeats for every block.

6) Decryption uses the same output stream to XOR with the ciphertext ⇒ gives original plaintext.



Bindu

How Synchronization is ensured:-

1) Both sender and receiver use same IV and key.

2) They generate the same output ~~Stream~~ Stream using AES.

3) So, Ciphertext and plaintext remain in Sync — even over Streaming.

Answer to the Question 25

AES mode that cause error propagation:-

1) CBC (Ciphen Block Chaining) and CFB (Ciphen Feedback) mode both cause error propagation.

2) A Single bit error in the ciphertext will effect multiple blocks in decryption.

In CBC mode:-

- Each ciphertext block is used to decrypt the next plaintext block.
- So, if one ciphertext block is corrupted
  - → The current plaintext block becomes garbled.
  - → The next block is also affected due to XOR with corrupted block.
- **Effect:-** One error → effect two blocks

Example:-

Let's we have 3 ciphertext blocks C_1, C_2, C_3

if C_2 is corrupted ⇒

P_2 = wrong (because its derived from Decrypt$(C_2) \oplus C_1$)

P_{2i} = also wrong (Decrypt$(C_3) \oplus C_2$)

In CFB Mode:-

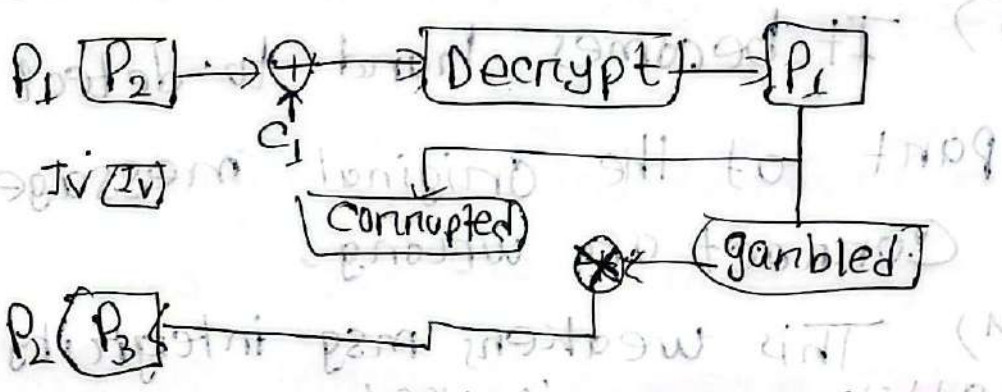
- Each ciphertext block is fed back into the encryption process.
- If a ciphertext block is corrupted
  - → The corresponding plaintext block become wrong.
  - → And the next one is also affected.

Example:-

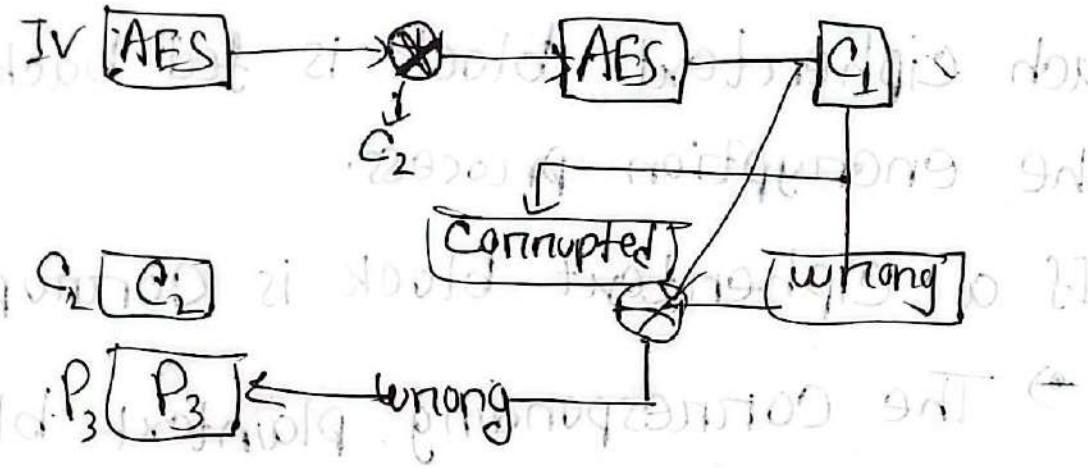
If C_2 is corrupted

- P_2 = wrong.
- P_3 = also affected.

CBC mode



CFB mode



Impact on integrity due to Error propagation:-

1) Single-bit error in ciphertext causes multiple-bit errors in decrypted output (CBC/CFB).

2) Even a small corruption in transmission can distort entire plaintext blocks.

3) It becomes hard to detect which part of the original message was correct or wrong.

4) This weakens msg integrity if no additional MAC is used.

Bindu

Answer to the question 26

I will recommend CTR (counter mode) for ~~for~~ encryption large files with parallel processing.

Justification:-

1) Support parallel processing:-

each block is encrypted independently using a counter value, allowing simultaneous encryption/decryption.

2) No block dependency:-

CTR does not rely on previous ciphertext like CBC so it is fasten and more efficient.

3) Maintains data confidentiality:-

CTR hides repeating patterns in ~~ptai~~ plaintext, unlike ECB which leaks structure

Bindu

4) Ideal for large file:-

Works best for big files where speed and efficiency matter.

5) Error does not propagate:-

A ~~sign~~ single-bit error affects only the corresponding block, making recover ~~easily~~ easier.

6) Flexible counter-based design:-

Counters can be precomputed and reused securely with a unique nonce, improving performance.

Answer to the question 27

we have to calculate the ciphertext and also have to decrypt it using the private key.

given,

$m = 1$,

$e = 5$,

$n = 14$

$d = 11$

Encryption:-

$C = m^e \bmod n$

$C = 1^5 \bmod 14$

$= 1 \bmod 14$

$= 1$

Ciphertext is 1.

Decryption :-

$m = C^d \bmod n$

$= 1^{11} \bmod 14$

$= 1 \bmod 14$

$= 1$

Observation:- Since the original message was 1, exponentiation didn't change it's value. This

Bindu

is a special case in RSA.

Answer to the question 28

we have to generate the digital signal

Given,

$H(m) = 5$

$d = 3$

$n = 33$

Digital Signal generation formula,

$$S = H(m)^d \bmod n$$

$$= 5^3 \bmod 33$$

$$= 125 \bmod 33$$

$$\therefore S = 26$$

$33 \overline{)125} \; 3$ (99), remainder 26

digital Signature $S = 26$

Bindu

Answer to the Question 29

Given,

P = 17 (primary modulo)

g = 3 (Base)

- Aleya's private key : a = 4
- Badol's private key : b = 5

Step-1

Aleya's public key :-

Formula

$$A = g^a \bmod p$$
$$= 3^4 \bmod 17$$
$$= 81 \bmod 17$$
$$= 13$$

17 | 81 | 4
68
13

Step-2

Badol's public key,

Formula :-

Bindu

$B = g^b \bmod p$

$= 3^5 \bmod 17$

$= 243 \bmod 17$

$= 243 - (17 \times 14)$

$= 5$

Badol's public key = 5

Final Answer

Aleya's public key 13

Badol's public key 5.

Answer to the Question 30

Given,

$H(x) = ($ Sum of ASCII values of characters in $x) \bmod 100$

1) message "AB"

- ASCII ('A') = 65
- ASCII ('B') = 66

Sum = 65 + 66 = 131

Hash,

H('AB') = 131 mod 100

= 31

2) message "BA"

ASCII ('B') = 66

ASCII ('A') = 65

Sum = 66 + 65 = 131

Hash :-

H("BA") = 131 mode 100

= 31

Comparison

H("AB") = 31, H("BA") = 31

Same hash value → This means "AB" and "BA" produce the same hash.

Binda

Implication on Collision resistance:-

- This is a collision, — two different inputs produce the same output hash
- It shows that weak hash functions are not collision-resistant.
- It can easily lead to security vulnerabilities.

Answer to the question 31

Given data,

- message m = 15
- Secret key k = 7
- modulus = 17

Step-1: MAC calculation:-

Formula:-

- MAC = (m+k) mod 17

MAC = (15 + 7) mod 17

= 22 mod 17

= 5

∴ MAC = 5

Step-2 : Attacker's attempt:-

Suppose attacken changes the message from 15 → 10

New MAC Formula

$MAC_{new} = (10 + k) \bmod 17$

But k is unknown ; so the attacken can't Comput the connect MAC.

If attacken guess

- He wants MAC to match else mac 5

$10 + k \equiv 5 \pmod{17}$

or, $k = -5 \bmod 17$

i $k = 12 \pmod{17}$

This guessed key is wrong because real key = 7

Receiver will checks:-

$(10+7) \bmod 17 = 17 \bmod 17$

$\equiv 0$

This does not mateh 5 → Attack fails.

Assumption

1) Without knowing the secret key attacken cannot generate connect MAC.
2) Wrong MAC will be detected by the receiver.
3) Large modulus makes brute-force impractical.
4) MAC ensures message integrity and authentication.

Bindu

Answer to the question 32

Steps involved in the ~~th~~ TLS Handshake process:-

1) Client Hello:-

- Cliend send Client Hello, selecting TLS version, Ciphen suite and sends it's random numben.

2) Serven Hello:-

- Serven replies with Serven Hello, ~~msg~~
- Selecting TLS version, Ciphen suite. and sends its random numben.

3) Serven Cintificate:-

- Serven sends it's digital & cintificate. to prove identity.

4) Serven key exchange:-

- For some Ciphen Suites, the Serven sends additional key exchange information.

5) Server Hello Done:-

- The server signals that its part of the handshake is complete.

6) Client key exchange:-

- The client generates a pre master secret and encripts using servers public key,
- Sends it to the server.

7) Session key generation:-

- Both side use:-
  - client random.
  - Server random.
  - pre-master secret
    → to derive the session key using a key derivation function.

8) Finish messages:-

- Both encrypt and send a 'Finished' message using the session key to

encryption is working.

* Secure comminication begins.

How symmetric keys are established Securely:-

1) The pre-master secret is exchanged via asymmetric encryption.

2) Even if someone capture the handshake without the private key they can't derive the session key.

3) Once derived, the session key is used for fast symmetric encryption of all further messages.

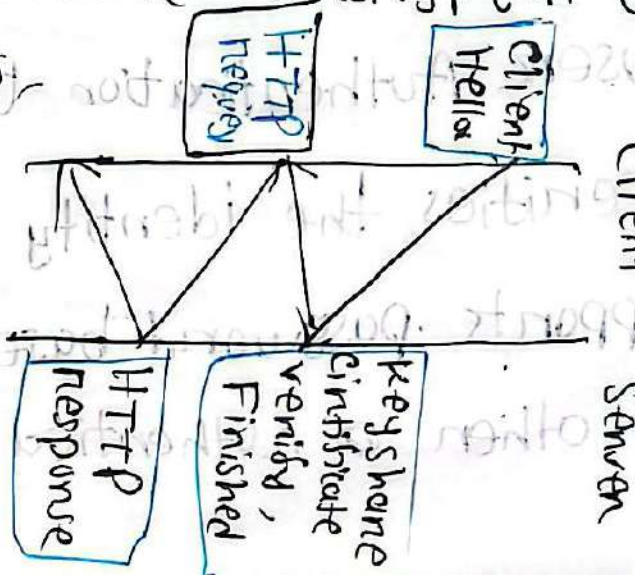


Fig:- TLS

Answer to the question 33.

SSH Layered Architecture (protocol stack)

→ SSH has 3 layers, each with a specific role:-

1) Transport Layer:-

- Establishes a secure and encrypted connection.
- Handles server authentication and confidentiality.
- negotiates encryption algorithms and compression.

2) User Authentication layer:-

- Verifies the identity of the client(user)
- Supports password-based, public key or other authentication method.

3) Connection layer:-

- Manages multiple logical channels over the Single SSH connection.
- Support Services like remote Shell, file transfer (SCP/SFTP) and port forwarding.

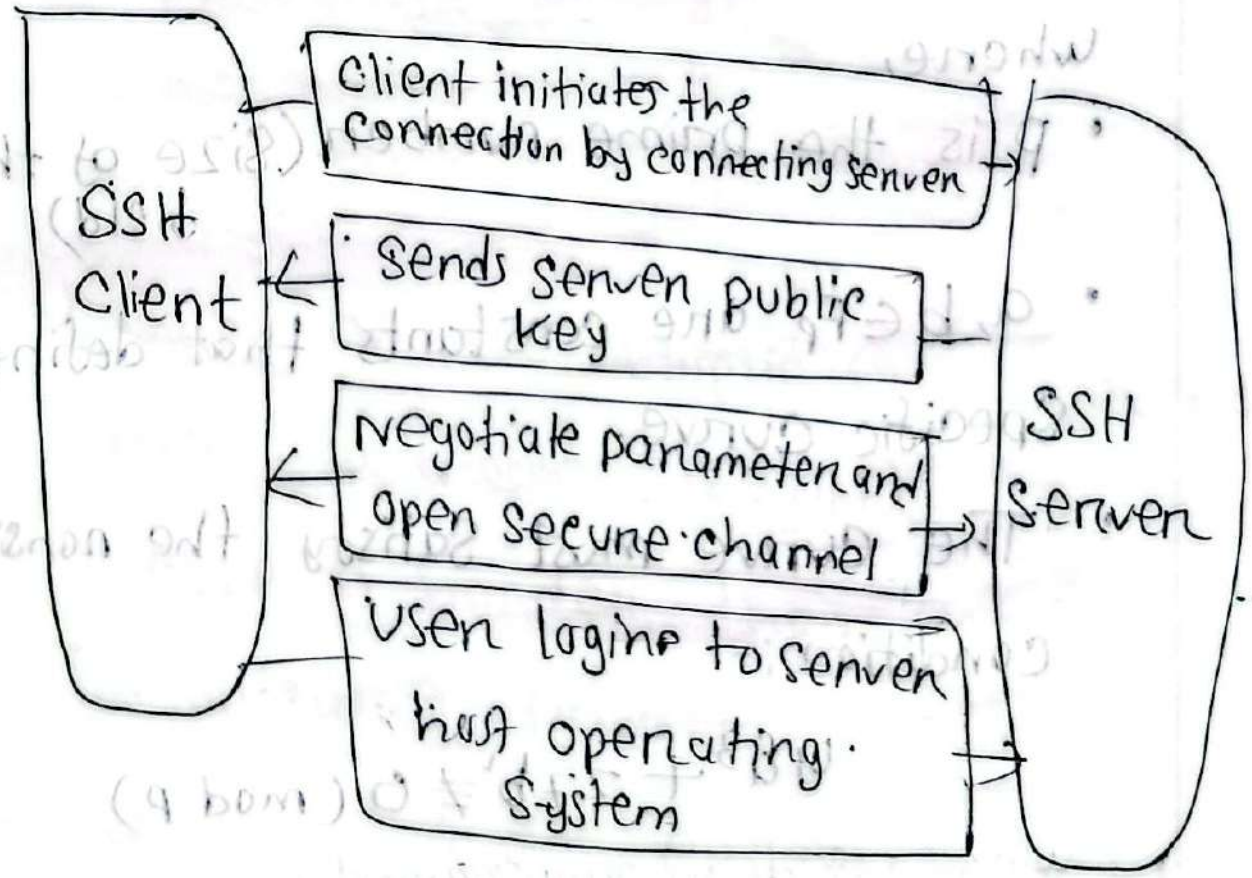


Fig: SSH protocol.

Bindu

Answer to the question 34

Steps involved in the ~~th~~ TLS Handshake process:-

1) Client Hello:-

- Cliend send Client Hello, Selecting TLS version, Ciphen suite and sends it's random numben.

2) Serven Hello:-

- Serven replies with Serven Hello, ~~msg~~
- Selecting TLS version, Ciphen suite. and sends its random numben.

3) Serven cintificate:-

- Serven sends it's digital & cintificate. to prove identity.

4) Serven key exchange:-

- For some Ciphen Suites, the Serven sends additional key exchange information.

5) Server Hello Done:-

- The Server signals that its part of the handshake is complete.

6) Client key exchange:-

- The client generates a pre master secret and encripts using server's public key.
- Sends it to the server.

7) Session key generation:-

- Both Side use:
  - client random.
  - Server random.
  - pre-master Secret
    → to derive the session key using a key derivation function.

8) Finish messages:-

- Both encrypt and send a 'Finished' message using the Session key to

Bindu

Answer to the question 35

General form of Elliptic curve over a finite field:

over a prime finite field F_p, an elliptic curve is defined as:

$$y^2 \bmod p = (x^3 + ax + b) \bmod p$$

where,

- p is the prime number (size of the field)
- $a, b \in F_p$ are constants that define the specific curve.

The curve must satisfy the nonsingularity condition:

$$4a^3 + 27b^2 \neq 0 \pmod p$$

Why used in cryptography:

① High security with small key sizes.

Bindu

2) Efficient in computation and bandwidth.

3) Based on ~~electri-~~ Elliptic Curve - Discrete Logarithm Problem (ECDLP) very hard to solve.

4) Widly used in secure protocols like TLS, SSH, Bitcoins etc.

5) Bandwidth Savings.

Answer to the question 3c

1) Based on the Elliptic Curve Discrete Logarithm Problem (ECDLP).

2) ECDLP is harder than RSA's integer factorization problem.

3) Security grows exponentially with Ecc key size.

4) Requires much smaller key ~~with~~

Bindu

for same security level.

5) Example:- ECC 256 bit ≈ RSA 3072 bit security.

6) Smaller key means faster encryption/decryption.

7) Saves bandwidth and storage space.

Answer to the question 37

We check if P=(3,6) satisfy the curve.

Curve equation,

$$y^2 \equiv x^3 + 2x + 3 \mod 97$$

Step-1

L.H.S = y^2

$= 6^2 = 36$

LHS = 36 (mod 97)

Bindu

Step-2

RHS := $x^3 + 2x + 3$

$= 3^3 + 2 \cdot 3 + 3$

$= 27 + 6 + 3$

$= 36$

RHS = 36 (mod 97)

L.HS = R.HS

So, point P = (3, 6) lies on the elliptic curve

Answer to the question 38

We use ElGamal encryption formula:-

Public key,

P = 23, g = 5, h = 8, m = 10, k = 6

Step1 :- Compute C_1

$C_1 = g^k \bmod p$

$= 5^6 \bmod p$

First,

$5^2 = 25 \equiv 2 \pmod{23}$

or, $(5^2)^3 = 2^3 \pmod{23}$

$\therefore 5^6 = 8 \mod 23$

$\therefore C_1 = 8$

Step-2 : Compute C_2 :-

$C_2 = m \times H^k \mod p$

$= 10 \times 8^6 \mod 23$

first

$h^k = 8^6 \mod 23$

~~$8^2 = 64 \mod$~~

$8^2 = 64 \equiv 18 \mod 23$

$8^4 = (18^2) = 324 \equiv 2 \mod 23$

$8^6 = 8^4 \cdot 8^2 = 2 \times 18 = 36 \equiv 13 \pmod{23}$

So,

$C_2 = 10 \times 13 = 130 \bmod 23$

$= 15$

$C_2 = 15$,

Ciphertext $= (C_1, C_2) = (8, 15)$.

Answer to the question 39

Importance of Lightweight cryptography in IoT.

1) Low power consumption :- IoT devices often run on batteries, so cryptographic algorithm must be energy efficient.

2) Low memory usage :- Many IoT devices have limited RAM / Rom, so algorithms must be small in code size.

3) Low Computational Cost:- Weak processors in IoT devices require algorithms with low complexity.

4) Fast execution:- Enables real-time communications and quick encryption/decryption.

5) Adequate security:- Even with reduced resources, must provide confidentiality, integrity and ~~authentication~~ authentication.

Example:

- PRESENT Cipher:- A lightweight block cipher designed for resource-constrained devices.
- Block size:- 64 bits, key size: 80 or 128 bits.

- Used in RFID tags, wireless sensors and embedded systems.

Answer to the question 40

3 common IoT specific attacks and their mitigation :-

1) Firmware Hijacking :-

- Explanation :- Attacker replaces or modifies device firmware to gain control or insert malicious code.
- Mitigation :-
  a) Tamper-Evident Seals and Secure Enclosure - prevent unnoticed access.
  b) Protect Sensitive data on the device.
  c) Sensor-based Alerts.

2) Physical Tempering:-

- Explaination:- Direct physical access to IoT devices to alert componets, extract keys on bypas security.

mitigation:-

1) Secure Boot.
2) Firmware integrity check.
3) Regular update.

3) IoT Botnets:-

- Explaination:- Large Scale infection of IoT devices to perform DDos attacks.
- mitigation:-
  a) change default credentials.
  b) Firmware & software updates.
  c) network segmentation & Firewalls.

Answer to the Question – 4

Complete Python-based code

Here a Python PRNG (Pseudo-Random number Generator) that uses current System time and a customer seed value to generate random numbers. I will also include sample output so

when.

Phython program

```
import time
def custom_prng(seed, count):
    current_time = int(time.time_ns())
    combined_seed = seed ^ current_time
    a = 1664525
    c = 1013904223
    m = 2**32
    random_numbers = []
    x = combine_seed
    for _ in range(count):
        x = (a*x + c) % m
        random_numbers.append(x)
    return random_numbers
seed_value = 12345
count = 5
numbers = custom_prng(seed_value, count)
```

```
print("custom_prng.(seed_value, count)
print("custom PRNG output:")
for i, num in enumerate(numbers, 1):
print("Random number {i}: {num}")
```

Sample output:

```
Custumen PRNG output:
Random number 1: 1885951992
Random number 2: 2910389135
Random number 3: 2475739278
Random number 4: 3399447855
Random number 5: 1326364459
```

Explaination:-

1) Seed initialization → combines custom seed and system time (nanoseconds)

Using XOR for randomness.

ii) LCG Formula → uses $(a*x+c)\%m$ to generate pseudo-random numbers.

iii) Quantum Safety: This PRNG is not Cryptography secure but is suitable for simulation / random tasks.

iv) Dynamic output → Even with the same seed different times give different result