Md. Fahim
ET-21089

Final Assignment
Cryptography and Cyber Law

Question 1: How does Shor's algorithm threaten the security of RSA and Elliptic Curve cryptography (ECC), and what are the potential consequences for current digital infrastructure?

Ans:

Shor's Algorithm: Shor's algorithm, a quantum computing algorithm, posseses a significant threat to the security of RSA and Elliptic curve cryptography (ECC) by efficiently solving the Mathematical problems these algorithms rely on. These systems ensure the confidentiality, authenticity, and integrity of everything from emails to online banking system and e-commerce.

Final Assignment
Cryptography and Cyber Law

Question 1: How does Shor's algorithm threaten the security of RSA and Elliptic curve cryptography (ECC), and what are the potential consequences for current digital infrastructure?

Ans:

Shor's Algorithm: Shor's algorithm, a quantum computing algorithm, posseses a significant threat to the security of RSA and Elliptic curve cryptography (ECC) by efficiently solving the mathematical problems these algorithms rely on. These systems ensure the confidentiality, authenticity, and integrity of everything from emails to online banking system and e-commerce.

Md. Fahim
ET-21039

Shor's algorithm poses a direct and grave threat to widely used in public key cryptography schemes like RSA and ECC, because it enables a quantum computer to efficiently solve the mathematical problems – integer factorization (for RSA) and the elliptic-curve discrete logarithm problem (for ECC) that underlie their security, in polynomial time instead of the exponential time required by classical algorithms.

RSA: Based on factoring large composite numbers, Shor's algorithm uses quantum period finding and quantum fourier transform to break RSS efficiently.

ECC: Relies on the difficulty of discrete logarithms on elliptic curves.

Shor's algorithm can solve this problem too, allowing recovery of private keys from public ones.

Estimates indicate ECC may actually be easier to break than RSA in a quantam scenario: for 128-bit classical security, ECC might require only ~2330 logical qubits versus ~4098 for RSA (though both are currently well beyond existing quantam hardware capabilities).

Potential consequences for digital infrastructure

i) Breakdown of internet security:

→ Protocols like TLS/SSL (used in HTTPS websites) will become insecure.

→ Attackers can intercept and decrypt web traffic, exposing sensitive data (password, credit cards etc).

Md. Fahim
ET-21039

ii) Data vulnerability:

→ Harvest now, decrypt later.
→ Retrospective Decryption.
→ Compromised confidentiality.

iii) Compromise of financial system:

→ Online banking and crypto currencies depend on RSA/ ECC.
→ Quantum computers could forge signatures, steal assests and manipulate financial transactions.

iv) Threat to Goverenment and Military data

→ Encrypted data being collected today could be decrypted in the future by quantum computers.
→ Long-term confidentiality of stored data is at severe risk.

v) Loss of trust in Digital Identity & Signatures:

→ Quantum attacks can forge digital certificates and software signatures.

vi) Erosion of privacy:

→ Personal and sensitive data exchanged through insecure networks could be expressed, leading to identity theft and privacy breaches.

vii) National security risks:

→ Compromised government communications and vital services could have serious national security implications.

viii) Urgency for cryptographic Migration:

→ If not addressed, there may be a sudden collapse in digital security often referred to as 'Q-day'.

→ Migrating to post quantum encryptography (PQC) is essential to avoid widespread disruption.

Md. Fahim
IT-21089

Shor's algorithm is not just a theoritical threat — it is ticking time bomb for all the cryptographic systems based on RSA and ECC.

Though large scale quantum computers do not exist, their development is advancing rapidly. Once available, they could instantly break the foundations of current digital security.

Question 2: Discuss the role of quantum key distribution (QKD) in future cryptographic systems. How does it differ from classical public key encryption?

Answer:

Quantum key distribution (QKD) is a secure communication method that utilized principles of quantum Mechanics to generate and distributes cryptographic keys. It allows two parties to create a shared secret key that canbe used to encrypt and decrypt messages, with security guaranted by the law of physics.

Role of QKD in future cryptographic systems

1. Quantum-safe key exchange:

→ Secure key sharing quantum physics.
→ Resistant to quantum attacks.

M. Farhan
FT-21029

2. Unconditional security:

→ QKD's security is based on laws of quantum physics, such as:
- Heisenberg's uncertainty principle.
- No-cloning theorem

→ Any attempt to eavesdrop on the quantum channel disturbs the system, alerting the legitimate parties.

3. Eavesdroping detection:

→ Detects any interception via quantum disturbance.
→ Compromised keys are discarded.

4. Post-Quantum Ready:

→ Used with symmetric encryption.
→ Builds quantum-secure communication.

5. Quantum internet backbone:

→ Core tech for future secure quantum networks
→ Already tested via satellites & fiber optics.

6. Physics - based trust:
→ Security from the laws of nature, not from hard math.
→ Not breakable by any computing power.

7. High security use:
→ Best for military, banking government.
→ Adapts quickly to new threats.

8. Crypto - Agility:
→ Frequent secure key updates.
→ Long-term sensitive data protection.

How QKD differs from classical public key encryption:

Feature	Classical Public key encryption (RSA, ECC)	Quantum key distribution (QKD)
Principle	Based on hard mathematical problems.	Based on quantum physics
Quantum Resistance	Vulnerable to quantum attacks.	Resistant to quantum attacks by design.
security type	Computational security.	Information Theoretic Security

Md. Fahim
ID- 21039

Eavesdropping Detection	No built-in mechanism.	Any interception disturbs the quantu state and is detecte
Purpose	key exchange, digital signature.	Secure key exchan only.
Communication Medium	Classical channel (e.g. internet, radio)	Quantum channels (e.g. photons over fiber optics
Maturity & adoption	Fully developed and widely used across digital systems.	Emerging technolog limited to high- security applicatio
Scalability & cost	Scalable, low cost implementation	High cost, limited range
Deployment	Widely deployed worldwide	Experimental, critical systems only

Question 3: What are the main differences between lattice based cryptography and traditional number theoretic approaches like RSA, particularly in the context of quantum resistance?

Answer:

Lattice-based cryptography and traditional number-theoretic cryptography (such as RSA and ECC) differ significantly in their mathematical foundations, security assumptions and resistance to quantum attacks.

Lattice based cryptography offers a significant advantage over traditional number theoretic approaches like RSA in terms of quantum resistance.

While traditional methods rely on problems that quantum computers can easily solve, lattice-based cryptography is built on problems that are believed to be difficult.

Md. Fahim
ET-21039

Here's more detailed breakdown of the differences:

1. Security Foundation:
   - Traditional (RSA/ECC):
     → Relies on the difficulty of factoring large numbers or solving the discrete logarithm problem.
   - Lattice-based Cryptography:
     → Relies on the difficulty of solving problems on high dimensional lattice such as SVP and LWE.
2. Mathematical Foundation:
   - Traditional (RSA/ECC):
     → Based on hard number-theoretic problems like integer factorization (RSA) and discrete logarithm problem.
   - Lattice-Based Cryptography:
     → Based on problems in high dimentional geometry, such as the shortest vector problem (SVP) and LWE.

3. Quantum Resistance:

- Traditional (RSA/ECC)
  → Vulnerable to quantum algorithms like Shor's algorithm, which can solve their underlying problems in polynomial time.
- Lattice-Based Cryptography:
  → Considered quantum-resistant as no efficient quantum algorithms are known to solve lattice problems.

4. Key size and performance:

- Traditional (RSA/ECC):
  → Use relatively smaller keys, but need very large keys for post-quantum security.
- Lattice-Based Cryptography:
  → Use larger keys and ciphertexts, but many schemes are still efficient and scalable.

5. Advanced Applications:

- Traditional (RSA / ECC)
  → Mostly limited to encryption, signatures and key exchange.
- Lattice based cryptography
  → Supports advanced techniques like homomorphic encryption, identity based encryption and post quantum signatures

Traditional number-theoretic cryptosystems like RSA are not secure in the quantum era, as they can be broken by quantum algorithms.

Lattice-based cryptography on the other hand, is based on quantum hard problems and is becoming the foundation of future cryptographic systems.

Hence, lattice based schemes are considered a key solution for post quantum security.

Ans. to the 2. no 04

Here is a Python PRNG (Pseudo-Random Number Generator) that uses current system time and a custom seed value to generate random numbers. I'll also include sample output

Python 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 = combined_seed
    for _ in range(count):
        x = (a * x + c) % M
        random_numbers.append(x)
    return random_numbers
```

Md. Fahim
IT-21039

```
seed_value = 12345
count = 5
numbers = custom_prng(seed_value, count)
print("Custom PRNG Output:")
for i, num in enumerate(numbers, 1):
    print("Random Number {i}: {num}")
```

Sample Output:

Custom PRNG Output:
Random Number 1: 1885951992
Random Number 2: 2910388135
Random Number 3: 2475239278
Random Number 4: 339994285
Random Number 5: 1321364488

Explanation:

(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 cryptographically secure but is suitable for simulation/random tasks.

(iv) Dynamic Output → Even with the same seed different times give different results.

Ans. to the q. no-05

Sieve of Eratosthenes Algorithm:

The sieve of Eratosthenes is an ancient and efficient method for finding all prime numbers up to a given limit n. It works by progressively eliminating the multiples of each prime number, starting from the smallest prime (2).

Algorithm steps:

1. Create a list of integers from 2 to n.
2. Start with the first number in the list ($p \leq 2$), which is prime.
3. Eliminate all multiple of p.
4. Find the next number in the list that is not marked. This is the next prime.
5. Repeat steps 3-4 until all multiples up to $\sqrt{n}$ have been processed.
6. The remaining unmarked numbers are all primes.

Find all Primes less than 50:

Let's use the Sieve of Eratosthenes to find all prime numbers less than 50.

1. Create a list of numbers from 2 to 49:

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49.

2. Start with the first prime, 2. Then eliminates all multiple of '2':

2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49

3. Eliminate all multiple of 3:

2, 3, 5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37, 41, 43, 47, 49

4. Eliminate all multiple of 5:

2, 3, 5, 7, 11, 13, 17, 19, 23, ~~25~~ 29, 31, 37, 41, 43, 47, 49

5. Eliminate all multiple of 7:

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

6. The next number is 11, and $11^2 = 121$, which is greater than 50, the algorithm stops here.

Md. Fahim
ITE-21039

The remain unmarked numbers are the prime numbers less than 50:
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47.

Time complexity comparison:

→ Sieve of Eratosthenes: $O(n \log \log n)$
- very efficient for large ranges.

→ Trial Division: $O(n\sqrt{n})$ - much slower for large n.

Thus, the sieve is significantly faster than trial division when generating all primes up to a large number.

Ans. to the 2. no. of 6

Definition: A Carmichael number is a composite integer n such that

$$a^{n-1} \equiv 1 \pmod{n}$$

for every integer a with $\gcd(a, n) = 1$.

A positive integer n is a Carmichael number if all three conditions hold:

1. n is composite.
2. n is square free.
3. For every prime p dividing n, $(p-1)$ divides $(n-1)$; i.e. $p-1 \mid n-1$

Verification for the given numbers:

We check each n against Korselt's criterion:

1. $n = 561$

→ Factorization: $561 = 3 \times 11 \times 17$ → composite and square free.

→ Compute $n-1 = 560$

For $p = 3$: $A-1 = 2.2 \mid 560$ ($560/2 = 280$).
For $p = 11$: $A-1 = 10.10 \mid 560$ ($560/10 = 56$)
For $p = 17$: $p-1 = 16.16 \mid 560$ ($560/16 = 35$)

All conditions satisfied → 561 is a Carmichael number.

(2.) $n = 1105$

→ Factorization: $1105 = 5 \times 13 \times 17$

→ composite and square free.

→ Compute $n-1 = 1104$

For $p \leq 5$; $p-1 \leq 4.4 \mid 1104$ ✓ $(1104/4 = 276)$

For $p \leq 13$; $p-1 \leq 12.12 \mid 1104$ $(1104/12 = 92)$

For $p \leq 17$; $p-1 \leq 16.16 \mid 1104$ $(1104/16 = 69)$

All condition satisfied → 1105 is a Carmichael number.

(3). $n = 1729$

Factorization: $1729 = 7 \times 13 \times 19$

composite and square free.

Compute $n-1 = 1728$

For $p \leq 7$: $p-1 \leq 6 \cdot 6 \mid 1728$ $\neq (1728/6 = 288)$

For $p \leq 13$: $p-1 \leq 12 \cdot 12 \mid 1728$ $(1728/12 = 144)$

For $p \leq 19$: $p-1 \leq 18 \cdot 18 \mid 1728$ $(1728/18 = 96)$

All conditions are satisfied → 1729 is a Carmichael number.

Ans. to the q. no - 07

Yes, In fact $\mathbb{Z}_{11}$ is a commutative ring with unity and moreover a field.

Justification:

→ $(\mathbb{Z}_{11}, +)$ is an abelian group: closures, associatively, identity 0, inverse $(-a \equiv 11-a)$, and commutativity hold (addition mod 11)

→ Multiplication mod 11 is closed and associative and distributes over addition.

→ There is a multiplicative identity $1 \in \mathbb{Z}_{11}$.

Md. Fahim
ID - 21030

→ Because 11 is prime, every non zero element $a \in \mathbb{Z}_{11}$ has a multiplicative inverse mod 11, thus all ring axioms hold and every non zero element is invertible → $\mathbb{Z}_{11}$ is a field.

Yes, $(\mathbb{Z}_{37}, +)$ is an abelian group.

Justification:

→ Set $\mathbb{Z}_{37} = \{0, 1, \ldots, 36\}$ with addition modulo 37.

→ Closure, associativity and commutativity follow from integer addition.

→ Identity is 0; for each a, additive inverse is $37 - a$.

→ Thus $(\mathbb{Z}_{37}, +)$ satisfies all group axioms and is abelian.

No, $(\mathbb{Z}_{35}, \times)$ is not a group.

Justification:

→ Although multiplication mod 35 is associative and has identity 1, not every element has a multiplicative inverse in $\mathbb{Z}_{35}$.

→ Example: $5 \in \mathbb{Z}_{35}$, $\gcd(5, 35) = 5 > 1$.
There is no x with $5x \equiv 1 \pmod{35}$ because any product $5x$ is divisible by 5 and can not be congruent to 1. Hence 5 has no inverse.

→ Therefore the inverse axiom fails and $(\mathbb{Z}_{35}, \times)$ is not a group; so it is not an abelian group.

Ans. to the Q. no. 08

We want r with $0 \leq r < 31$ and $-52 \equiv r \pmod{31}$.

Compute:

$$-52 + 2 \cdot 31 \leq -52 + 62 \leq 10$$

So, $-52 \equiv 10 \pmod{31}$

Alternate:

$$-52 \equiv -52 + 31 \equiv -21 + 31 = 10$$

So, the remainder is 10.

Md. Fahim
ID-21099

Ans: to the q.no-09

We solve $7x \equiv 1 \pmod{26}$ or find integers x, y with $7x + 26y = 1$.

Use the Euclidean algorithm:

$$26 = 3 \cdot 7 + 5,$$
$$7 = 1 \cdot 5 + 2,$$
$$5 = 2 \cdot 2 + 1,$$
$$2 = 2 \cdot 1 + 0.$$

Back-substitute to express 1 as a linear combination:

$$5 - 2(7 - 1 \cdot 5) = 3 \cdot 5 - 2 \cdot 7$$
$$= 3(26 - 3 \cdot 7) - 2 \cdot 7 = 3 \cdot 26 - 11 \cdot 7$$

Thus $-11 \cdot 7 + 3 \cdot 26 = 1$. Therefore $x \equiv -11 \pmod{26}$

$$x \equiv -11 \equiv 15 \pmod{26}.$$

The multiplicative inverse of 7 modulo 26 is 15.

Ans. to the q. no. 10

Step 1 Multiply the numbers:

$-8 \times 5 \equiv -40$

Step 2 Reduce modulo 17:

We find the equivalent positive remainder

$-40 + 3 \times 17 \equiv -40 + 51 \equiv 11$

$-40 \equiv 11 \pmod{17}$

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

Explain of simplification method:

Negative numbers in modular arithmetic can be converted to their positive equivalent before multiplying.

$-8 \equiv (9 \bmod 17)$

$9 \times 5 \equiv 45$

Reduce 45 Modulo 17:

$45 - 2 \times 17 \equiv 45 - 34 \equiv 11$

This ~~Matter~~ Match the previous result.

∴ Final answer - 11

Ans: to the Q. no - 1.1

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

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

In particular, if $\gcd(a, b) = 1$ there are integers x, y with $ax + by = 1$; then x is the multiplicative inverse of a modulo b.

Proof:

Let, $d = \gcd(a, b)$

→ By the Euclidean algorithm, d divides both a and b.

→ The set of all integers of the form $ax + by$ is closed under addition and substraction.

→ Let M be the smallest positive number in this set. Then M divides both a and b.

→ Hence, $M \leq d$, and there exist integers x, y such that:

$$ax + by = d$$

Finding the multiplicative inverse of 97 modulo 385:

We want x such that:

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

This is equivalent to:

$$97x + 385y = 1$$

We use the extended Euclidean Algorithm.

1. Euclidean Algorithm:

$$385 = 97 \cdot 3 + 94$$
$$97 = 94 \cdot 1 + 3$$
$$94 = 3 \cdot 31 + 1$$
$$3 = 1 \cdot 3 + 0$$

So, $\gcd(97, 385) = 1$

2. Back substitution:

From $94 = 3 \cdot 31 + 1$,

$$1 = 94 - 3 \cdot 31$$
$$= 94 - (97 - 94)\,31$$
$$= 94 \cdot 32 - 97 \cdot 31$$
$$= (385 - 97 \cdot 3)\,32 - 97 \cdot 31$$

$\Rightarrow 1 = 385.32 - 97.127$

3. Conclusion:

We have $1 = (-127).97 + 32.385$

Therefore $x \equiv -127 \pmod{385}$

Since we want the positive inverse:-

$-127 \equiv 385 - 127 \equiv 258 \pmod{385}$

The Multiplicative inverse of 97 mod 385 is 258.

Ans. to the Q. no 12

Bezout's Identity statement:

For any integers a and b, there exist integers x and y such that:

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

Proof:

1. Let $d = \gcd(a, b)$ by difinition, d divides, both a and b, so $d \mid a = da'$ and $b = db'$, where a', b' are integers with $\gcd(a', b') = 1$

2. Since a' and b' are coprime, there exists integers x_0, y_0 such that:

$$a'x_0 + b'y_0 = 1$$

3. Multiplying through d gives:

$$a(dx_0) + b(dy_0) = d$$

Thus, $x = x_0$ and $y = y_0$ are integer solutions to:

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

Hence proved.

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

This asks for the multiplicative inverse of 43 modulo 240. We solve the Diophantine equation:

$$43x - 240y = 1$$

by extended Euclidean Algorithm.

Computing gcd chain:

$$240 = 5 \cdot 43 + 25$$
$$43 = 1 \cdot 25 + 18$$
$$25 = 1 \cdot 18 + 7$$
$$18 = 2 \cdot 7 + 4$$
$$7 = 1 \cdot 4 + 3$$
$$4 = 1 \cdot 3 + 1$$
$$3 = 3 \cdot 1 + 0$$

So, gcd (43, 240) = 1 and the inverse exist:
Now back-substitute to express 1, as a linear combination of 43 and 240.
Back substitution:

$$1 = 4 - 1 \cdot 3$$
$$= 4 - 1(7 \cdot 1 - 4) = 2 \cdot 4 - 1 \cdot 7$$
$$= 2(18 - 2 \cdot 7) - 1 \cdot 7$$
$$= 2 \cdot 18 - 5 \cdot 7$$
$$= 2 \cdot 18 - 5(25 - 1 \cdot 18)$$
$$= 7 \cdot 18 - 5 \cdot 25$$
$$= 7(43 - 1 \cdot 25) - 5 \cdot 25$$
$$= 7 \cdot 43 - 12 \cdot 25$$
$$= 7 \cdot 43 - 12(240 - 5 \cdot 43)$$
$$= 67 \cdot 43 - 12 \cdot 240$$

Thus $43 \cdot 67 - 240 \cdot 12 = 1$,
So, $x = 67$ is a solution.
$43^{-1} \equiv 67 \pmod{240}$ or $43 \cdot 67 \equiv 1 \pmod{240}$.

Ans: to the q. no 13

Fermat's Little Theorem (FLT) statement:
If p is a prime number and a is any integer with gcd (a, p) = 1, then

$$a^{p-1} \equiv 1 \pmod{p}$$

Proof:

Consider the nonzero residues modulo p: 1, 2, ..., p-1. Multiply each by a (with gcd (a, p) = 1).

The set a·1, a·2, ..., a·(p-1) is a permutation of 1, 2, ... p-1 modulo p.

Taking the product of all elements in both sets and reducing mod p gives -

$$a^{p-1} \cdot (1 \cdot 2 \cdots (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \pmod{p}$$

Cancelling the nonzero product $1 \cdot 2 \cdots (p-1)$ (which is invertible mod p) yields

$$a^{p-1} \equiv 1 \pmod{p}$$

Md. Fahim
ID: 2102...

Primality test using FLT:

Choose a with $1 < a < p$ and $\gcd(a, p) = 1$.
If $a^{p-1} \not\equiv 1 \pmod{p} \Rightarrow p$ is composite.
If it is 1, p may be prime - but there are composites (Carmichael numbers) that also pass.

Is 561 prime?

Take $a = 2$, $\gcd(2, 561) = 1$

$561 = 3 \cdot 11 \cdot 17$

By Chinese Remainder theorem and FLT:

$2^{560} \equiv 1 \pmod{3}, 1 \pmod{11}, 1 \pmod{17}$

So, $2^{560} \equiv 1 \pmod{561} \rightarrow 561$ passes FLT but is not prime. (It's a carmichael numbe

Evaluate $5^{193} \bmod 175$

1. Factor the module: $175 = 25 \cdot 7$. Work modulo 25 and 7 and combine by the CR

2. Mod 25: $5^2 = 25 \Rightarrow 5^n \equiv 0 \pmod{25}$ for all $n \geq 2$ since $123 \geq 2$.

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

3. Mod 7: $\gcd(5, 7) = 1$ so by Fermat $5^6 \equiv 1 \pmod 7$. Reduce the exponent:

$$123 \equiv 123 - 6 \cdot 20 = 3 \pmod 6$$

$$5^{123} \equiv 5^3 = 125 \equiv 6 \pmod 7.$$

4. Combine by CRT: Find x with

$$x \equiv 0 \pmod{25}, \quad x \equiv 6 \pmod 7.$$

Check Multiples of 25: $25 \cdot 5 = 125 \equiv 6 \pmod 7$

Thus the solution modulo 175 is

$$5^{123} \equiv 125 \pmod{175}$$

Fermat's Little Theorem can not be applied directly modulo 175, because 5 and 175 are not coprime that's why we used CRT.

Ans. to the q. no - 14

Statement: The chinese Remainder Theorem states that a system of linear congruences with pairwise coprime modulo has a unique solution modulo the product of the modulo.

If $m_1, m_2 \ldots m_k$ are pairwise coprime positive integers and $a_1 \ldots a_k$ are any integers, then the system.

$$x \equiv a_i \pmod{m_i} \quad (i = 1, \ldots k)$$

has a unique solution modulo $M = m_1 m_2 \ldots m_k$

Proof (constructive):

Let $M = \prod_{i=1}^{k} m_i$ and $M_i = M/m_i$. since $\gcd(M_i, m_i) = 1$, there exists an inverse y_i such that $M_i y_i \equiv 1 \pmod{m_i}$ Then,

$$x = \sum_{i=1}^{k} a_i M_i y_i$$

Satisfies $x \equiv a_j \pmod{m_j}$ for each j (because for $i \neq j$, M_i is divisible by m_j)

Uniqueness modulo M follows because any two solutions differ by a multiple of all m_i, hence by a multiple of M.

Solve the system:

Start with $x \equiv 2 \pmod 3$ put $x = 2 + 3k$

Plug into mod 5:

$2 + 3k \equiv 3 \pmod 5 \Rightarrow 3k \equiv 1 \pmod 5$

Inverse of 3 mod 5 is 2 (since $3 \cdot 2 = 6 \equiv 1$) so

$k \equiv 2 \pmod 5 \Rightarrow k = 2 + 5t$

Thus $x \equiv 2 + 3k = 2 + 3(2 + 5t) = 8 + 15t$

Now impose mod 7:

$8 + 15t \equiv 2 \pmod 7 \Rightarrow 15t \equiv 6 \equiv 1 \pmod 7$

Since $15 \equiv 1 \pmod 7$, this gives $t \equiv 1 \pmod 7$.

So, $t = 1 + 7s$. Then

$x = 8 + 15(1 + 7s) = 8 + 15 + 105s = 23 + 105s$

Therefore the solution modulo $3 \cdot 5 \cdot 7 = 105$

$x \equiv 23 \pmod{105}$

Ans. to the q. no 15

The CIA triad stands for confidentiality, Integrity and Availability.
It's a fundamental for information security.

→ **Confidentiality:**
- Ensures data is disclosed only to authorized parties.
- Mechanisms: Encryption, access controls, authentication, secure channels.
- Contribution: Prevents data leaks, protects privacy and secrets.

→ **Integrity:**
- Ensures data is accurate and has not been tampered with.
- Mechanisms: Cryptographic hashes, digital signatures, MACs, checksums, version control.
- Contribution: detects and prevents unauthorized modification, supports trust in data and transactions.

→ Availability:

- Ensures authorized user can access data and services when needed.
- Mechanisms: redundancy, backups, DDoS protection, fault tolerance, Monitoring.
- Contribution: Keeps systems usable and resilient, prevents service disruption which could harm business operations.

All three are required together: strong confidentiality with no availability is useless, availability without integrity/confidentiality is risky. Good security design balances CIA based on system requirements.

Ans. to the q. no - 15

Difference: Steganography vs Cryptography

- Cryptography transforms plaintext into an unreadable form (ciphertext) so that the content is hidden (confidential), but the existence of the message is usually obvious. Goal: Unreadability without the key.
- Steganography hides the existence of the message ~~for~~ by embedding it into another 'innocuous' object (image, audio, video, text).
- Goal: Secrecy of existence (covert communication)

They can be used together. First encrypt the secret, then hide the ciphertext with steganography.

Common steganography techniques (digital Media)

1. LSB (Least Significant Bit) substitution (image/audio): replace least significant bits of pixels/samples with message bits. Simple and high capacity but vulnerable to processing/noise.
2. Masking & filtering (images): Hide data in perceptually significant areas (watermarking style), robust against some image processing
3. Transform-domain methods: embaded data in frequency coefficients. More robust to compression and common edits.
4. Spread-spectrum steganography: Spread message bits across many samples, low detectability and robust.

5. Protocol / Metadata Hiding: Store message in file Metadata / unused header fields, or network protocols fields.

6. Text steganography: Using whitespace, synonyms, or formating changes.

Ans. to the q. no-17

Phishing:

- Method: Social engineering - attackers send deceptive emails / messages or create fake websites to trick users into revealing credentials, personal data, or clicking malicious link.
- Goal / Impact: Credential theft, identity theft, initial foothold in networks.
- Defences: User education, email filtering Multi-factor authentication (MFA) URL / Websites scanners, anti phishing policies.

Malware:

- Method: Software (viruses, worms, trojans, ransomware, spyware) delivered via email attachments, drive-by downloads, infected devices, or malicious installers.

- Goal/Impact: Data theft, destruction, system compromise, encryption for ransom establishing persistent backdoors.
- Defenes: Endpoint protection/AV, applications, White listing, patch Management, least privilege, network segmentation, backups.

Denial of service (DOS/DDOS):

- Method Overwhelm a service or network with traffic or exploit resource limits.
- Goal/Impact Availability loss - legitimate users cannot access service, can cause business disruption and revenue loss.

Defence: Rate limiting, traffic filtering, DDOS Mitigation services (scrubbing), content delivery networks, redundancy.

Comparison:

- Phishing targets people
- Malware targets systems/software to steal, modify or destroy data.

Ans. to the q.no-18

Role of GDPR:

The general Data Protection Regulation (GDPR) is a European Union (EU) law designed to strengthen the protection of personal data.

1. Data Minimization: GDPR requires organizations to collect only necessary personal data, reducing the amount of sensitive information that can be stolen in a cyber attack.

2. Security Measures: It mandates strong technical and organizational measures to safeguard data against unauthorized access, altercation, or loss.

3. User Rights: Individuals have rights such as access to their data, correction, detection and data probability.

GDPR significantly reduces the risk and impact of cyber attacks, while ensuring the privacy and trust of users.

Ans. to the q. no - 19

Basic Working of the DES Algorithm:

The data Encryption Standard (DES) is a symmetric key block cipher that encrypts data in fixed size blocks.
It operated on;
- Plaintext block: 64 bits (8 bytes)
- Key: 56 bits
- output: 64 bit cyphertext.

1. Initial Permutation (IP)

- The 64-bit plaintext is rearranged according to a fixed table.

Ex: If the plaintext is p[p1, p2 ... p64], the IP might move p58 to position 1, p50 to position 2 etc.

2. Rounds (16 iterations)

The output of IP is divided into two halves:
- Left half (L0): first 32 bits.
- Right half (R0): last 32 bits.

a) Expansion (E-box)

- The 32-bit right half(R0) is expanded to 48 bits by duplicating certain bits, using the E expansion table.

b) Key Mixing

- A 48 bit subkey is generated from the Main 56-bit key using a key schedule
- The expanded R is XORed with the subkey: R-expanded ⊕ subkey.

c) Substitution (S-boxes)

- The 48 bit result is divided into 8 blocks of 6 bits.
- Each block is passed through a s-box to produce a 4-bit output.
- Total output = 32 bits.

d) Swap and combine

Formula for each round:

$L_n = R_{n-1}$

$R_n = L_{n-1} \oplus f(R_{n-1}, K_n)$

Md. Fahim
IT-21059

Ans. to the Q.920-20

Given:

$R_0 = 0_x F0F0F0F0$

Round key $K_1 = 0_x 0F0F0F0F$

$L_0 = 0_x AAAAAAAA$

Step 1: Compute $f(R_0, K_1)$ assuming XOR operation only,

$f(R_0, K_1) = R_0 \oplus K_1 = 0_x F0F0F0F0 \oplus 0_x 0F0F0F0F$

Peform XOR byte-wise:

- F0 ⊕ 0F = Ff
- F0 ⊕ 0F = Ff
- F0 ⊕ 0f = ff
- F0 ⊕ 0F = Ff

So, $f(R_0, K_1) = 0_x FFFFFFFF$

Step 2: Compute $L_1 = R_0 = 0_x F0F0F0F0$

Step 3: Compute $R_1 = L_0 \oplus f(R_0, K_1)$

$R_1 = 0_x AAAAAAAA \oplus 0_x FFFFFFFF$

XOR byte-wise:

- AA ⊕ FF = 55
- AA ⊕ FF = 55
- AA ⊕ FF = 55
- AA ⊕ FF = 55

So, $R_1 = 0x55555555$

$\therefore L_1 = 0xF0F0F0F0$, $R_1 = 0x55555555$

Ans. to the Q. no-21

Given input word:

$[0x23, 0xA7, 0x4C, 0x19]$

Partial AES S-box table:

Row \ col	3	4	7	9	A	C
1	6D	.	.	C6	.	.
2	D4	.	.	.	.	.
4	.	A1	.	.	2E	.
A	.	.	63	D2	.	.

For each byte, the high nibble (4 bits) is the row, and the low nibble is the column. Let's find the corresponding S-box value for each ~~byte~~ byte!

- For 0x23: row = 2, col = 3 → From table, row = 2 col 3 = D4
- For 0xA7: row = A, col = 7 → From table row A, col 7 = 63
- For 0x4C: row = 4, col = C → From table, row 4 col C = (not given in table → dot), so we leave it as is or unknown.
- For 0x19: row = 1, col = 9 → From table, row 1 col 9 = 6b

Resulting output word:
[0x04, 0x63, unknown, 0x6b]
since 0x4C is not present, we indicate unknown or no mapping given.

Ans. to the q.no- 22

The AddRoundKey step in AES encryption involve performing a bitwise XOR operation between the input word and the round key word.

Given:

input word : [0x14, 0x2B, 0x3C, 0x4D]

Round key word : [0x55, 0xBB, 0x77, 0x88]

Compute the output word (XOR each byte).

Solution (byte-wise XOR):

$0x1A \oplus 0x55 = 0x4F$ (00011010 ⊕ 01010101 = 01001111)

$0x2B \oplus 0xBB = 0x4D$ (00101011 ⊕ 10111011 = 01001101)

$0x3C \oplus 0x77 = 0x4B$ (00111100 ⊕ 01110111 = 01001011)

$0x4D \oplus 0x88 = 0xC5$ (01001101 ⊕ 10001000 = 11000101)

Resulting output word: [0x4F, 0x4D, 0x4B, 0xC5]

SEM-2 2022

Brief explanation: The AddRoundKey step performs a bitwise XOR between each byte of the state and the corresponding byte of the round key, producing the output word shown above.

Ans. to the Q.No-23

The MixColumns operation in AES works by multiplying each column of the state matrix by a fixed matrix over the Galois field GF(2^8).

Given that,

Input column (bytes):

$$a = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 0x01 \\ 0x01 \\ 0x03 \\ 0x04 \end{bmatrix}$$

MixColumns Matrix

$$M = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

Multiplication rules used (AES)

- Multiply by 01: value itself.
- Multiply by 02: (called xtime): Left shift by 1 bit; if original MSB = 1 then XOR with $0x1B$ after the shift. Multiply by 03: $03.x = (02.x) \oplus (01.x)$

Input bytes in binary:

- $0x01 = 00000001_2$
- $0x02 = 00000010_2$
- $0x03 = 00000011_2$
- $0x04 = 00000100_2$

ROW 1 calculation

$$b_0 = (02.a_0) \oplus (03.a_1) \oplus (01.a_2) \oplus (01.a_3)$$

compute terms:

→ $02.a_0 = 02.0x01 = xtime(0x01) = 0x02$

→ $03.a_1 = (02.0x02) \oplus 0x02 = xtime(0x02)$ XOR

$0x02 = 0x04$ XOR $0x02 = 0x06$

→ $01.a_2 = 0x03$

→ $01.a_3 = 0x04$

Now, XOR them:

$b_0 = 0x02 \oplus 0x04 \oplus 0x04 = (0x02 \oplus 0x06) = 0x04$

$\oplus 0x04 \oplus 0x03 \oplus 0x04 = 0x04 \oplus 0x03$

$\oplus 0x04 = 0x03$

So, $b_0 = 0x03$

Row 2 calculation

$b_1 = (01 \cdot a_0) \oplus (02 \cdot a_1) \oplus (03 \cdot a_2) + (01 \cdot a_3)$

Terms

$01 \cdot a_0 = 0x01$

$02 \cdot a_1 =$ xtime$(0x02) = 0x04$

$a_3 \cdot a_2 =$ xtime$(0x03)$ XOR $0x03 =$ (xtime $(0x03)$ XOR $0x03 = 0x06$ XOR $0x03 = 0x05$

$01 \cdot a_3 = 0x04$

XOR $b_1 = 0x01 \oplus 0x04 \oplus 0x05 \oplus 0x04 = ((0x01 \oplus 0x04) \oplus 0x05) \oplus 0x04 = 0x01$

$\oplus 0x04 = 0x04$

So, $b_1 = 0x04$

Md. Fahim
IT-21039

ROW 3

$b_2 = (01.a_0) \oplus (01.a_1) \oplus (02.a_2) \oplus (03.a_3)$

XOR $b_2 = 0x01 \oplus 0x02 \oplus 0x06 \oplus 0x0C$

$= 0x09$

ROW 4

$b_3 = (03.a_0) \oplus (01.a_1) \oplus (01.a_2) \oplus (02.a_3)$

XOR $b_3 = 0x03 \oplus 0x02 \oplus 0x03 \oplus 0x08$

$= 0x0A$

Final result (output column)

$$b = \begin{bmatrix} 0x03 \\ 0x04 \\ 0x09 \\ 0x0A \end{bmatrix}$$

Binary terms

$0x03 = 00000011_2$

$0x04 = 00000100_2$

$0x09 = 00001001_2$

$0x0A = 00001010_2$

Ans to the 2.no-24

AES-OFB is a mode of operation for the advanced Encryption Standard (AES) that turns a block cipher into a stream cipher.

How it works

1. Initialization The process starts with a unique Initialization Vector (IV) which is typically a random value.

2. Key stream Generation The output of the AES encryption of the IV becomes the first part of the keystream.

3. Encryption: Each block of the keystream is XORed with a corresponding block of plaintext to produce the ciphertext.

4. Decryption The decryption process is identical to the encryption process. The same IV is used to generate the exact same key stream.

Ans. to the q. no - 25

AES Modes Causing Error Propagation:

Some AES modes (CBC and CFB) cause an error in a ciphertext block to affect multiple plaintext blocks during decryption.

1. CBC (Cipher Block Chaining) Mode:

- Decryption formula: $P_i = AES^{-1}(C_i) \oplus C_{i-1}$
- Effect effects:
  - If C_i has a 1-bit error → P_i becomes completely random.
  - The same bit position in P_{i+1} will also be flipped.

2. CFB (Cipher Feedback) Mode

- Decryption formula: $P_i = C_i \oplus AES(C_{i-1})$
- Error effect:
  - if C_i has a 1-bit error → the same bit in P_i is wrong.
  - P_{i+1} becomes completely random.
  - After that decryption returns to normal.

Ans. to the q.no-26

Recommended Modes: AES- CTR (Counter Mode).

justification

- Parallel processing:
  - In CTR, each block is encrypted by XORing the plaintext with a keystream generated.
  - The counter values for all blocks are known in advance, so encryption and decryption of different blocks can happen independently and parallel.
- Performance
  - No chaining between blocks faster than CBC for large files.
- Security
  - Unlike ECB, CTR does not produce identical ciphertext for identical ciphertext.
  - As secure as CBC when IV/nonce is unique.

Md. Fahim
IT-21039

Ans. to the q. no - 27

Given, $M = 1$, $e = 5$, $n = 14$, $d = 11$.

Encryption: $C \equiv M^e$ and $n = 1^5 \bmod 14 = 1$.

Decryption: $M \equiv C^d \bmod n = 1^{11} \bmod 14 = 1$

Why this works: RSA relies on $M^{ed} \equiv M \pmod n$ when $ed \equiv 1 \pmod{\phi(n)}$. Here, $\phi(14) = 6$ and $e.d = 5.11 = 55 \equiv 1 \pmod 6$, so decryption recovers M.

∴ Ciphertext $C = 1$ and Decryption Message $M = 1$

Ans. to the q. no - 28

Given: Hash $H(M) = 5$, $d = 3$, $n = 33$.

Signature generation (Sign with private key d):

We need to generate a digital signature for a message hash.

- The message hash is $H(M) = 5$
- The RSA private key is $(d, n) = (3, 33)$
- The digital signature S, is computed using the formula:

$$S = H(M)^d \pmod n$$

$$S = 5^3 \pmod{33} = 125 \pmod{33}$$

- $125 = 3 \times 33 + 26$, so $125 \pmod{33} = 26$
- The digital signature is 26.

Ans. to the Q. no-23

We'll compute the public keys for Aleya and Badol based on the Diffie-Hellman protocol.

- Public values: Prime Modulus $p = 17$, base $g = 3$
- Aleya's Private key: $a = 9$
- Badol's Private key: $b = 5$
- Aleya's public key (A):
  - $A = g^a \pmod p = 3^9 \pmod{17} = 81 \pmod{17}$
  - $81 = 4 \times 17 + 13$. So $81 \pmod{17} = 13$
  - Aleya's public key is 13.
- Badol's public key (B):
  - $B = g^b \pmod p = 3^5 \pmod{17} = 243 \pmod{17}$.
  - $243 = 14 \times 17 + 5$, So $243 \pmod{17} = 5$
  - Badol's public key is 5.

Ans. to the q. no - 30

Definition:

$H(x) = (\sum \text{ASCII (chars in } x)) \bmod 100$

ASCII values: A = 65, B = 66

Compute hashes:

$H(\text{"AB"}) = (65 + 66) \bmod 100 = 131 \bmod 100$
$= 31;$

$H(\text{"BA"}) = (66 + 65) \bmod 100 = 131 \bmod 100 = 31$

Result:

Both "AB" and "BA" produces the same hash → collision.

Collision Resistance

- The two different messages, "AB" and "BA", produce the same hash value. This is called collision.
- This hash function is not collision-resistant because it's based on a simple modulus sum

Ans. to the 2.no-31

Given, Message $M = 15$, $K = 7$

Compute MAC:

$$MAC = (15 + 7) \bmod 17 = 22 \bmod 17 = 5$$

Attackers scenario: Suppose attackers changes Message to $M' = 10$, but does not know K.

Read MAC for M' is:

$$MAC' = (10 + 7) \bmod 17 = 17 \bmod 17 = 0$$

Why forging is easy:

Because MAC is linear: $MAC = M + K \pmod{17}$ if attackers knows one valid pair (M, MAC), and wants to create $M' = M + 4$, they can compute $MAC' = MAC + 4 \pmod{17}$ without knowing K.

Example, here: $4 = -5 = 12 \pmod{17}$;

$MAC(25 + 12 = 17 = 0$, which matches the true MAC.

Md. Fahim
IT-21039

Ans. to the 2. no- 32

TLS Handshake steps & symmetric key establishment:

1. Client Hello → Client sends supported TLS version, cipher suites, random numbers
2. ServerHello → Server selects TLS version & cipher, suite, sends its random number.
3. Server certificate → Contains servers public key, signed by CA for Authentication.
4. Key exchange →
   - RSA Mode
   - ECDHE / DHE mode
5. Pre-Master secret → Master secret.
6. Session key → From Master secret.

Non symmetric keys are established secure: Asymmetric encryption (RSA or Diffie-Hellman) ensures that only the intended parties can compute the shared secret from which symmetric keys are derived.

Ans. to the q. no - 33

SSH refers to "secure shell" is a protocol that provides a secure channel over an unsecured network.

(1) Transport layer protocol:

This is the lowest layer, responsible for managing the secure connection.

- Handles encryption
- integrity protection
- compression
- server authentication.

(2) User Authentication Protocol

This layer runs on top of the transport layer and handles client authentication

- Varifies the clients identity using password
- public keys or other methods

(3) Connection Protocol:

Multiplexes the encrypted channel in to multiple logical channels. This is the highest layer.

Ans. to the 2. no - 34

TLS handshake process step:

1. ClientHello → Purpose connection parameters.
2. ServerHello → Purpose connection parameters
3. Certificate & key exchange.
4. Generate Master Secret
5. Generate Session Keys.
6. Finished Messages
7. Secure Communication

Ans. to the 2. no - 35

The general form of an elliptic curve equation over a finite field is:

$$y^2 = x^3 + ax + b \pmod{p}$$

Here, p is a large prime number that defines the finite field, and a and b are constants. $4a^3 + 27b^2 \neq 0 \pmod{p}$

Use in cryptography; Provides a group structure for elliptic curve cryptography(E enabling secure key change, signature.

Ans. to the Q. No-36

Elliptic Curve Cryptography (ECC) is a public key cryptography technique that provides the same cryptographic strength as RSA but with much smaller key sizes.

1. Mathematical Foundation:
   - RSA is based on the integer factorization problem (IFP).
   - ECC is based on the Elliptic Curve Discrete Logarithm Problem (ECDLP).

2. Key size comparison:
   - ECC 256-bit ≈ RSA 3072-bit security level.
   - ECC 384-bit ≈ RSA 7680-bit security level

3. Conclusion:
   ECC provides strong security with reduced key sizes due to the computational hardness of the ECDLP compared to integer factorization.

Ans. to the q. no-32

Given that,

Curve: $y^2 \equiv x^3 + 2x + 3 \pmod{97}$

point: $P \equiv (3, 6)$

$LHS \equiv y^2 \equiv 6^2 \equiv 36 \pmod{97}$

$RHS \equiv x^3 + 2x + 3 \equiv 27 + 6 + 3 \equiv 36 \pmod{97}$

since, $LHS \equiv RHS$, point lies on the curve.

Ans. to the q. no-33

To compute the ElGamal ciphertext, we use the following public values:

$p = 23$, $g = 5$, $A = 8$, and the message $M = 10$, the random number is $k = 6$

The ciphertext consists of two parts (C_1, C_2).

- C_1 computation

$C_1 \equiv g^k \pmod{p}$

$C_1 \equiv 5^6 \pmod{23}$

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

$5^4 = (5^2)^2 \equiv 4 \pmod{23}$

$5^6 \equiv 5^4 \times 5^2 = 8 \pmod{23}$

So, $C_1 = 8$.

C_2 computation:

$C_2 \equiv M \times k^k \pmod{p}$

First, we calculate k^k:

$k^k \equiv 8^6 \pmod{23}$

$8^2 \equiv 64 \equiv 2 \times 23 + 18 \equiv 18 \pmod{23}$

$8^4 \equiv (8^2)^2 \equiv 18^2 \equiv 324 \equiv 14 \times 23 \times 2 \equiv 2 \pmod{23}$

$8^6 \equiv 8^4 \times 8^2 \equiv 2 \times 18 \equiv 36 \equiv 1 \times 23 + 13 \equiv 13 \pmod{23}$

Now, we compute C_2:

$C_2 \equiv 10 \times 13 \pmod{23} \equiv 130 \pmod{23}$

$130 \equiv 5 \times 23 + 15 \equiv 15 \pmod{23}$

So, $C_2 \equiv 15$

The ElGamal ciphertext is $(8, 15)$

$C_1, C_2 = (8, 15)$

Ans. to the q. no - 39

IoT devices usually have limited resources - small memory, low CPU, and constrained battery life.

Lightweight cryptography is designed to:

- Reduce computational overhead
- Lower memory usage
- Extend battery life
- Maintain security

Example Algorithm: PRESENT

- A block cipher designed for extremely constrained environments.
- Uses a 64 bit block size and supports 80 or 128 bit keys.
- Very low hardware footprint (can fit into under 1000 gate equivalents).

Ans. to the q. no - 40

Attacks

Firmware Hijacking:

Mitigation strategies

- Use secure boot
- Sign firmware updates digitally.
- Deliver updates over encrypted channels.

Physical Tampering

- Tamper-evident seals / enclosures
- Secure debug ports (disable JTAG / SWD)
- Store keys in secure hardware modules.

Botnets

- Change default passwords.
- Use firewall & intrusion detection
- Keep firmware updated to patch vulnerabilities.