**Lab Objectives**

1. Understand the working principles of RSA and Diffie-Hellman algorithms.
2. Perform manual computations for RSA and Diffie-Hellman key exchange.
3. Implement RSA encryption and decryption in Python.
4. Simulate the Diffie-Hellman key exchange process.
5. Analyze the security aspects of these algorithms.

**Activity 1: Understanding RSA Step-by-Step**

**Objective:** Manually perform RSA key generation, encryption, and decryption with small prime numbers. Use examples from the lecture slides in last week to understand the process.

**Task 1: Key Generation (Manual Calculation)**

1. Choose two small prime numbers, **p** and **q**.

p = 3 q = 11

1. Compute **n = p × q**

n=3×11=33

1. Compute **ϕ(n) = (p-1) × (q-1)**.

φ(33)=(3−1)×(11−1)=2×10=20

1. Choose a small public exponent **e**.

e=3

1. Compute the private key **d = e⁻¹ mod ϕ(n)** (find the modular inverse).

D×3≡1 mod 20

3×7=21≡1 mod 20

1. Write down the public and private keys.

**Public Key (e, n) = (3, 33)**

**Private Key (d, n) = (7, 33)**

**Task 2: Encrypt a Message (Manual Calculation)**

1. Convert a small message (e.g., **M = 5**) into numerical form.
2. Compute **C = M^e mod n**.

C=5\*\*3 mod 33

C=125 mod 33

C=125−(33×3)=125−99=26

**Task 3: Decrypt the Message (Manual Calculation)**

1. Compute **M = C^d mod n** to retrieve the original message.

M=26\*\*7 mod 33

26\*\*2=676≡14 mod 33

26\*\*4=(26\*\*2)\*\*2=14\*\*2=196≡31mod33

26\*\*7=26×26\*\*2×26\*\*4≡26×14×31mod33

26×14=364≡1mod33

1×31=31mod33

**Activity 2: Understanding Diffie-Hellman Key Exchange**

**Objective:** Simulate the Diffie-Hellman key exchange by working in pairs and manually computing the values.

In this task, you will work in groups of two. **Do not sit near your partner; instead, communicate using Teams messages (or another suitable online channel).**

1. **Pairing and Communication:** Select a partner of your choice. **Do not sit near your partner.** Coordinate through Teams messages for communication.
2. **Selecting Public Values:** Decide on two public numbers **p** (prime number) and **g** via Teams messages. Ideally, g should be a primitive root modulo p for better security but for convenience, you can use any number.

P = 23

g = 5

1. **Choosing Private Values:** Each student privately selects a secret number (**a** for Student 1, **b** for Student 2). Keep this secret and do not share it even with your partner.
2. **Computing Public Keys:** Each student computes:
   * Student 1: **A = g^a mod p**
   * Student 2: **B = g^b mod p**
3. **Exchanging Public Keys:** Share the computed values **A** and **B** with your partner via Teams messages. Note: Do not share private keys at any step.

A = 8

B = 19

1. **Computing the Shared Secret:** Each student computes the shared secret using their private key and the received public key:
   * Student 1: **s = B^a mod p**
   * Student 2: **s = A^b mod p**
2. **Verification:** Both students should arrive at the same shared secret **s**. If the values differ, check your calculations, and repeat the procedure.

S = 2

**LAB LOGBOOK REQUIREMENT:** Document in your lab logbook.

1. Your partner's name,

Maher Salimboev

1. Values: p, g, s, and your private key (Do not write your partner’s private key)

p = 23

g = 5

privatekey= 15

publickey = g^b mod p = 5^15 mod 23

* Computing step by step:
  + 5^1 mod 23 = 5
  + 5^2 mod 23 = 25 mod 23 = 2
  + 5^4 mod 23 = 4
  + 5^8 mod 23 = 16
  + 5^15 = 5^8 × 5^4 × 5^2 × 5^1 = 16 × 4 × 2 × 5
  + Working through: 16 × 4 = 64 mod 23 = 18
  + 18 × 2 = 36 mod 23 = 13
  + 13 × 5 = 65 mod 23 = 19
* Public Key B = 19

s = A^b mod p = 8^15 mod 23

* Computing step by step:
  + 8^1 mod 23 = 8
  + 8^2 mod 23 = 64 mod 23 = 18
  + 8^4 mod 23 = 18^2 mod 23 = 324 mod 23 = 2
  + 8^8 mod 23 = 2^2 mod 23 = 4
  + 8^15 = 8^8 × 8^4 × 8^2 × 8^1 = 4 × 2 × 18 × 8
  + Working through: 4 × 2 = 8
  + 8 × 18 = 144 mod 23 = 6
  + 6 × 8 = 48 mod 23 = 2
* Shared Secret s = 2

**Optional Challenge (work in groups)**

* Work with your partner to simulate a **hybrid encryption** system where:
  + Diffie-Hellman is used to exchange a symmetric encryption key.
  + AES is used for encrypting and decrypting messages (you can use Python library for performing encryption and decryption).
* This mimics **real-world encryption systems**.
* Share any secret message using the D with your partner using encryption.
* Your partner will decrypt it using same key (which you shared using Diffie-Hellman)