BRACT's
Vishwakarma Institute of Information Technology
Department of Computer Engineering(Software Engineering)

Submitted By :

Roll No.	Name	PRN No.
3101070	Pranita Narayan Kute	22420110

ASSIGNMENT NO. 7

1. **Title** - Implement Diffie-Hellman Key Exchange Protocol
2. **Aim -** To implement the Diffie–Hellman Key Exchange Protocol, which allows two parties to securely exchange a shared secret key over an insecure communication channel without directly transmitting the key.
3. **Objective**
   a. To understand the concept of public-key cryptography and key exchange mechanisms.
   b. To demonstrate how two users can establish a common secret key using modular arithmetic.
   c. To implement and verify the Diffie–Hellman algorithm through a simple program.
   d. To enhance understanding of modular exponentiation and discrete logarithm problems used in cryptographic systems.
4. **Pre-requisite Knowledge**

Before implementing the protocol, one should understand:

a. Modular arithmetic (concept of mod n operations).
b. Prime numbers and primitive roots (generators).
c. Exponentiation and modular exponentiation.
d. Public-key cryptography concepts – especially asymmetric encryption and key sharing.
e. Basic programming knowledge (C++/Python or any language used).

5. Theory

- **Introduction:**

The Diffie-Hellman algorithm is being used to establish a shared secret that can be used for secret communications while exchanging data over a public network using the elliptic curve to generate points and get the secret key using the parameters.

For the sake of simplicity and practical implementation of the algorithm, we will consider only 4 variables, one prime P and G (a primitive root of P) and two private values a and b.

P and G are both publicly available numbers. Users (say Alice and Bob) pick private values a and b and they generate a key and exchange it publicly. The opposite person receives the key and that generates a secret key, after which they have the same secret key to encrypt.

- **Diffie-Hellman Algorithm Steps**

1. Select public parameters:
    - Choose a prime number p.
    - Choose a primitive root g (also called generator) modulo p.
    - These values (p, g) are public and known to everyone.
2. Private keys:
    - User A chooses a private key (a random number < p).
    - User B chooses a private key b (a random number < p).
3. Compute public keys:
    - A computes A = (g^a) mod p.
    - B computes B = (g^b) mod p.

    Both A and B are shared publicly over the network.

4. Compute shared secret key:
    - A computes K1 = (B^a) mod p.
    - B computes K2 = (A^b) mod p.

    Since both yield the same value mathematically:

    K1 = K2 = g^(ab) mod p

CODE

Diffie-Hellman Key Exchange Implementation with User Input

Step 1: Input public values (P and G)

```
P = int(input("Enter a prime number (P): "))

G = int(input("Enter a primitive root of P (G): "))


# Step 2: Input private keys for Alice and Bob

a = int(input("Enter private key for Alice (a): "))

b = int(input("Enter private key for Bob (b): "))


# Step 3: Compute public keys

A = pow(G, a, P)   # A = G^a mod P

B = pow(G, b, P)   # B = G^b mod P

print("\n--- Public Keys ---")

print(f"Alice's Public Key (A): {A}")

print(f"Bob's Public Key (B): {B}")


# Step 4: Exchange public keys and compute shared secret

S1 = pow(B, a, P)  # Alice computes shared secret

S2 = pow(A, b, P)  # Bob computes shared secret

print("\n--- Shared Secret Computation ---")

print(f"Alice's Shared Secret: {S1}")

print(f"Bob's Shared Secret:   {S2}")


# Step 5: Verify both keys match

if S1 == S2:

    print("\n Key Exchange Successful!")

    print(f"Shared Secret Key = {S1}")
```

```
else:

    print("\n Key Exchange Failed! Shared keys do not match.")
```

6. Conclusion

- The Diffie–Hellman Key Exchange Protocol successfully enables two users to generate a shared secret key over an insecure channel without transmitting the key itself.
- This key can then be used for secure symmetric encryption.
- The security of this protocol relies on the difficulty of solving the discrete logarithm problem, making it a cornerstone of modern cryptography.

7. Resources

Sr. No.	Source	Link
1.	GeeksForGeeks	https://www.geeksforgeeks.org/computer-networks/implementation-diffie-hellman-algorithm/