**Cryptography, Network and Security**

Assignment 8

Implement the Diffie-Hellman Key Exchange algorithm for a given problem

Code:

#include <iostream>

#include <cmath>

using namespace std;

*// Function to perform modular exponentiation (base^exp) % mod*

*// It returns (base^exp) % mod efficiently*

long long modPow(long long base, long long exp, long long mod)

{

    long long result = 1;

    base = base % mod;

    while (exp > 0)

    {

        if (exp % 2 == 1)

        {

            result = (result \* base) % mod;

        }

        exp = exp >> 1;

        base = (base \* base) % mod;

    }

    return result;

}

int main()

{

*// Publicly known prime number p and base g*

    long long p = 23; *// Prime number (Example)*

    long long g = 5; *// Generator (Example)*

    cout << "Publicly known values (p = " << p << ", g = " << g << ")" << endl;

*// Alice chooses her private key (secret value a)*

    long long a;

    cout << "Alice, enter your private key (a): ";

    cin >> a;

*// Bob chooses his private key (secret value b)*

    long long b;

    cout << "Bob, enter your private key (b): ";

    cin >> b;

*// Alice computes her public key A = (g^a) % p*

    long long A = modPow(g, a, p);

    cout << "Alice's public key (A) is: " << A << endl;

*// Bob computes his public key B = (g^b) % p*

    long long B = modPow(g, b, p);

    cout << "Bob's public key (B) is: " << B << endl;

*// Alice and Bob exchange their public keys (A and B)*

*// Alice computes the shared secret S = (B^a) % p*

    long long shared\_secret\_Alice = modPow(B, a, p);

    cout << "Alice's computed shared secret: " << shared\_secret\_Alice << endl;

*// Bob computes the shared secret S = (A^b) % p*

    long long shared\_secret\_Bob = modPow(A, b, p);

    cout << "Bob's computed shared secret: " << shared\_secret\_Bob << endl;

*// Both shared secrets should be the same*

    if (shared\_secret\_Alice == shared\_secret\_Bob)

    {

        cout << "Key exchange successful! Shared secret: " << shared\_secret\_Alice << endl;

    }

    else

    {

        cout << "Key exchange failed. The shared secrets do not match." << endl;

    }

    return 0;

}