**Experiment no.3**

**Learning Objective:** Analyze and implement Diffie-Hellman Key Exchange Algorithm

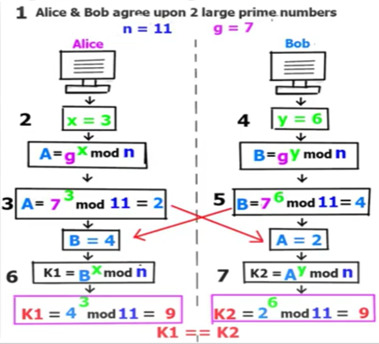
**Tools:** C/C++/Java/Python

**Theory: DIFFIE–HELLMAN KEY EXCHANGE:**

Diffie–Hellman key exchange (D–H) is a specific method of exchanging keys. It is one of the earliest practical examples of key exchange implemented within the field of [cryptography](http://en.wikipedia.org/wiki/Cryptography). The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret [key](http://en.wikipedia.org/wiki/Key_(cryptography)) over an insecure [communications](http://en.wikipedia.org/wiki/Communication) channel. This key can then be used to encrypt subsequent communications using a [symmetric key](http://en.wikipedia.org/wiki/Symmetric_key) [cipher](http://en.wikipedia.org/wiki/Cipher).

The Diffie–Hellman key agreement was invented in 1976 during a collaboration between Whitfield Diffie and Martin Hellman and was the first practical method for establishing a [shared secret](http://en.wikipedia.org/wiki/Shared_secret) over an unprotected communication channel.

Diffie–Hellman establishes a shared secret that can be used for secret communications by exchanging data over a public network.

****

**STEP 1: GLOBAL PUBLIC ELEMENTS**:  
Firstly, Alice and Bob agree on two large prime numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.

**STEP 2: ASYMMETRIC KEY GENERATION BY USER 'A':**Alice chooses another large random number X, and calculates, the public key, A, such that:

A= gX mod n

**STEP 3:** Alice sends the number A to Bob.

**STEP 4: KEY GENERATION BY USER 'B'**:  
Bob independently chooses another large random number Y, and calculates, the public key, B, such that:

B= gY mod n

**STEP 5:** Bob sends the number B to Alice.

**STEP 6: SYMMETRIC KEY (K) GENERATION BY USER 'A':**A now computes the secret key, K1 as follows:

K1= BX mod n

**STEP 7: SYMMETRIC KEY (K) GENERATION BY USER 'B'**:  
B now computes the secret key, K2 as follows:

K2= AY mod n

**NOTE:**

It should be difficult for Alice to solve for Bob's private key or for Bob to solve for Alice's private key. If it is not difficult for Alice to solve for Bob's private key (or vice versa), Eve may simply substitute her own private / public key pair, plug Bob's public key into her private key, produce a fake shared secret key, and solve for Bob's private key (and use that to solve for the shared secret key. Eve may attempt to choose a public / private key pair that will make it easy for her to solve for Bob's private key).

**Code:**

p=int(input("The value of p: "))

q=int(input("The value of q: "))

a=int(input('The Private Key a for Alice is: '))

b=int(input('The Private Key b for Bob is: '))

print("Generated key for Alice: ",pow(q,a)%p)

R=pow(q,a)%p

print("Generated key for Bob: ",pow(q,b)%p)

S=pow(q,b)%p

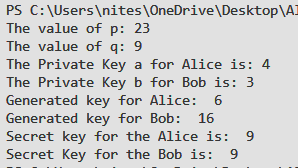
print("Secret key for the Alice is: ",pow(S,a)%p)

Rk=pow(S,q)%p

print("Secret Key for the Bob is: ",pow(R,b)%p)

Sk=pow(R,b)%p

**Output:**

****

**Result and Discussion:**

* Thus we have seen working of Diffie-Hellman Key algorithm.
* We have also seen how it can generate key.
* It isn’t used for encryption just generates key

**Learning Outcomes:** The student will be able to

LO1: Understand the Diffie-Hellman Key Exchange Algorithm

LO2: Analyze and implement the Diffie-Hellman Key Exchange Algorithm

**Course Outcomes:** Upon completion of the course students will be able to analyze and implement Diffie-Hellman Key Exchange Algorithm for generation of shared symmetric key

**Conclusion:** Successfully studied Diffie-Hellman Key algorithm and implemented it.

**For Faculty Use**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Correction Parameters** | **Formative Assessment [40%]** | **Timely completion of Practical**  **[ 40%]** | **Attendance / Learning Attitude [20%]** |  |
| **Marks Obtained** |  |  |  |