Experiment No. 6

Title: Diffie-Hellman Key Exchange Protocol

Batch: B2 **Roll No.: 16010420117** **Experiment No.: 6**

Title: To implement Diffie-Hellman key exchange protocol.

Resources needed: Windows/Linux OS

Theory:

To implement Diffie-Hellman, the two end users Alice and Bob, while communicating over a channel they know to be private, mutually agree on positive whole numbers p and q, such that p is a prime number and q is a generator of p. The generator q is a number that, when raised to positive whole-number powers less than p, never produces the same result for any two such whole numbers. The value of p may be large but the value of q is usually small. Once Alice and Bob have agreed on p and q in private, they choose positive whole-number personal keys a and b, both less than the prime-number modulus p. Neither user divulges their personal key to anyone; ideally they memorize these numbers and do not write them down or store them anywhere. Next, Alice and Bob compute public keys a* and b* based on their personal keys according to the formulas

$$a^* = q^a \bmod p$$

and

$$b^* = q^b \bmod p$$

The two users can share their public keys a* and b* over a communications medium assumed to be insecure, such as the Internet or a corporate wide area network (WAN). From these public keys, a number x can be generated by either user on the basis of their own personal keys. Alice computes x using the formula

$$x = (b^*)^a \bmod p$$

Bob computes x using the formula

$$x = (a^*)^b \bmod p$$

The value of x turns out to be the same according to either of the above two formulas. However, the personal keys a and b, which are critical in the calculation of x, have not been transmitted over a public medium. Because it is a large and apparently random number, a potential hacker has almost no chance of correctly guessing x, even with the help of a powerful computer to conduct millions of trials. The two users can therefore, in theory, communicate privately over a public medium with an encryption method of their choice using the decryption key x.

Algorithm : Make use of client-server chatting application**.**

A. Client/Sender

1. Choose a large prime number *p*
2. Calculate generator *g* of *p*
3. Share *p* and *g* with the Server/Receiver
4. Select any natural number (client secrete) *a*
5. *Calculate* $R_A = g^a \ mod \ p$ and send it to the Server/Receiver
6. Upon receiving R_B from the Server/Receiver, calculate shared key $K_{AB} = (R_B)^a \ mod \ p$

B. Server/Receiver:

1. Select any natural number (server secrete) *b*
2. Upon receiving *p* and *g*, calculate $R_B = b^a \ mod \ p$ and send it to the Client/Sender
3. Upon receiving R_A from the Client/Sender, calculate shared key $K_{AB} = (R_A)^b \ mod \ p$

NOTE/OBSERVATION : Manually verify that the K_{AB} at both the ends is same.

Code:

```
import random

P = int(input("Enter the Value of P "))

G = int(input("Enter the Value of G "))


print('The Value of P is :%d'%(P))
print('The Value of G is :%d'%(G))

# Alice will choose the private key a
a = random.randint(1, 100)
print('The Private Key a for Alice is :%d'%(a))

# gets the generated key
x = int(pow(G,a,P))

# Bob will choose the private key b
b = random.randint(1, 100)
print('The Private Key b for Bob is :%d'%(b))

# gets the generated key
y = int(pow(G,b,P))


# Secret key for Alice
ka = int(pow(y,a,P))

# Secret key for Bob
kb = int(pow(x,b,P))

print('Secret key for the Alice is : %d'%(ka))
print('Secret Key for the Bob is : %d'%(kb))
```

Output:

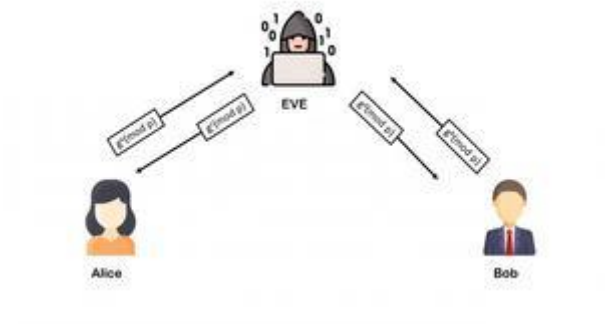
```
PS C:\Users\l enovo\Desktop\noins> python .\diffieHellmanKeyExchange.py
Enter the Value of P 23
Enter the Value of G 10
The Value of P is :23
The Value of G is :10
The Private Key a for Alice is :24
The Private Key b for Bob is :56
Secret key for the Alice is : 8
Secret Key for the Bob is : 8
PS C:\Users\l enovo\Desktop\noins>
```

Questions:

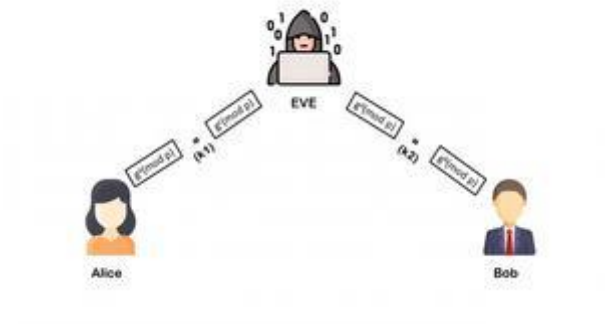
1. Explain any one attack on Diffie-Hellman key exchange protocol.
   Ans: Man in the Middle attack can manipulate the communications between Alice and Bob, and break the security key exchange.

Step by Step explanation of this process:
Step 1: Selected public numbers p and g, p is a prime number, called the "modulus" and g is called the base.
Step 2: Selecting private numbers.
let Alice pick a private random number a and let Bob pick a private random number b, Malory picks 2 random numbers c and d.

Step 3: Intercepting public values,
Malory intercepts Alice's public value (g^a(mod p)), block it from reaching Bob, and instead sends Bob her own public value (g^c(modp)) and Malory intercepts Bob's public value (g^b(mod p)), block it from reaching Alice, and instead sends Alice her own public value (g^d (modp))



Step 4: Computing secret key
Alice will compute a key $S_1=g^{da}(\text{mod } p)$, and Bob will compute a different key, $S_2=g^{cb}(\text{mod } p)$



Step 5: If Alice uses S_1 as a key to encrypt a later message to Bob, Malory can decrypt it, re-encrypt it using S_2, and send it to Bob. Bob and Alice won't notice any problem and may assume their communication is encrypted, but in reality, Malory can decrypt, read, modify, and then re-encrypt all their conversations.

Outcomes:

CO2: Illustrate different cryptographic algorithms for security

---

Conclusion:
We learned diffie hellman key exchange and implemented it successfully.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

- Mark Stamp, “Information security Principles and Practice” Wiley