**Param Patil**

**2019130049**

**TE Comps**

**Batch C**

**Experiment 2: Diffie Hellman Algorithm**

**Github Link:** [**https://github.com/pparam-2610/CSS\_LAB\_5thSem**](https://github.com/pparam-2610/CSS_LAB_5thSem)

**Aim:**

**To implement Diffie Hellman Key Exchange Algorithm in Python.**

**Code:**

P = int(input('Enter Prime Number P: '))

G = int(input('Enter Prime Number G: '))

a = int(input('Enter Private KeyA for Alice: '))

x = int(pow(G,a,P))

b = int(input('Enter Private KeyB for Bob: '))

y = int(pow(G,b,P))

ka = int(pow(y,a,P))

kb = int(pow(x,b,P))

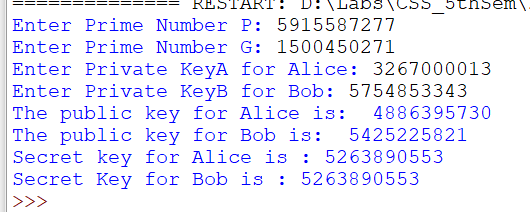
print('The public key for Alice is: ',x)

print('The public key for Bob is: ',y)

print('Secret key for Alice is : %d'%(ka))

print('Secret Key for Bob is : %d'%(kb))

1. **Diffie Hellman**



**Conclusion:**

**In the second experiment, I implemented Diffie Hellman algorithm,**

**Diffie Hellman Algorithm:** which is a method to share a common secret key in public communication. It allows two parties to generate a key which they can use to secure their subsequent communications. The reason why Diffie Hellman is difficult to crack is because suppose A is having a,g^a, g^b and B is having b, g^a, g^b, but for a third person he could only find out g^a and g^b . For A he/she can calculate g^a from a, but for a third person it is naturally difficult to find out a from g^a. In modern times, Diffie Hellman along with authentication is considered the safest mode of communication between two parties, because if there is only Diffie Hellman, then it is not possible for B to verify he/she is communicating with A. There could be a man in the middle attack, but once there is authentication, after that since all the communication will be carried out by the common secret key, one can be sure about the transfer. The Diffie Hellman is majorly used in SSL, SSH and there are 4 major DH groups; 1,2,5,14 DH group 1 consists of a 768 bit key, group 2 consists of 1024 bit key, group 5 is 1536 bit key length and group 14 is 2048 bit key length. Group 14 is the strongest and most secure of the ones just mentioned, but there are other key lengths as well. The higher the Diffie-Hellman algorithm used, the more secure it will be, but it will be more CPU intensive, whereas the lower the algorithm, it will be the opposite, less secure and less CPU time required to compute the algorithm.