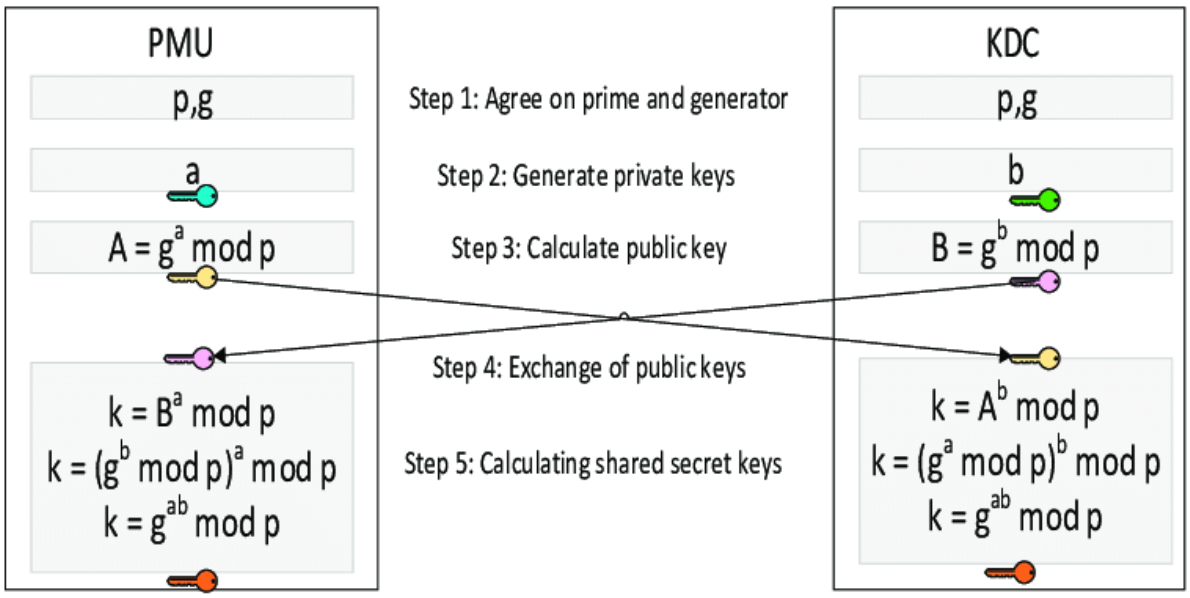
Experiment 2

Diffie Hellman Key Exchange

Name: Kashish Jain
UID: 2019130022
Class: TE Comps

Objective:
Implement Diffie Hellman key exchange algorithm in Python.

Theory:
Diffie –Hellman Key exchange algorithm
The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. Although Diffie–Hellman key agreement itself is an anonymous (no authenticated) key-agreement protocol, it provides the basis for a variety of authenticated protocols, and is used to provide perfect forward secrecy in Transport Layer Security's ephemeral modes (referred to as EDH or DHE depending on the cipher suite).



Code:

```
from math import sqrt

def is_prime(n):
    if n > 1:
        for i in range(2, int(sqrt(n)) + 1):
            if (n % i == 0):
                prime_flag = 1
                break
        if (prime_flag == 0):
            return True
        else:
            return False
    return False


def diffie_hellman():
    p = int(input('Enter the prime number: '))
    if not is_prime(p):
        p = int(input("Number not prime\nEnter again: "))

    g = int(input('Enter the generator(Primitive root of P): '))


    x = int(input('\nEnter the Secret x: '))
    y = int(input('Enter the Secret y: '))
    X = int(pow(g,x,p))
    Y = int(pow(g,y,p))
    ka = int(pow(Y,x,p))
    kb = int(pow(X,y,p))

    print('\nSecret key K1 :',ka)
    print('Secret Key K2 :',kb)

if __name__  == '__main__':
    diffie_hellman()
```

Output:

```
Enter the prime number: 12
Number not prime
Enter again: 13
Enter the generator(Primitive root of P): 5
Number should be primitive root of P
Enter again: 6

Enter the Secret x: 5
Enter the Secret y: 4

Secret key K1 : 3
Secret Key K2 : 3
```

Conclusion:

- I learned the procedure for encrypting messages using various cryptographical algorithms.
- I observed that through this algorithm the users can communicate with each other through an insecure channel without compromising any important data. It makes it possible to generate a symmetric key between 2 parties without sharing it over a compromised channel. It would be nearly impossible to crack the key for this algorithm if the value of the generator is large enough.

Github Links

Repository

https://github.com/kashishvjain/CSS-Lab