Aditya Motwani
2019130040
TE Comps

Experiment 2: Key Exchange Algorithms

Aim: To implement the Diffie Hellman Key Exchange Algorithm.

Github Link:

https://github.com/adityamotwani/CSS-Lab-2019130040/blob/main/exp2.py

Code:

```
from math import sqrt

def isPrime(n):
    if (n <= 1):
        return False
    if (n <= 3):
        return True
    if (n % 2 == 0 or n % 3 == 0):
        return False
    i = 5
    while(i * i <= n):
        if (n % i == 0 or n % (i + 2) == 0) :
            return False
        i = i + 6
    return True

def power(x, y, p):
    res = 1
    x = x % p
    while (y > 0):
        if (y & 1):
            res = (res * x) % p
        y = y >> 1
        x = (x * x) % p
    return res

def findPrimefactors(s, n) :
    while (n % 2 == 0) :
```

```
        s.add(2)
        n = n // 2
    for i in range(3, int(sqrt(n)), 2):
        while (n % i == 0) :
            s.add(i)
            n = n // i
    if (n > 2) :
        s.add(n)

def findPrimitive( n) :
    s = set()
    if (isPrime(n) == False):
        return -1
    phi = n - 1
    findPrimefactors(s, phi)
    for r in range(2, phi + 1):
        flag = False
        for it in s:
            if (power(r, phi // it, n) == 1):
                flag = True
                break
        if (flag == False):
            return r
    return -1

print("Enter a prime number: ")
p = int(input())

while(isPrime(p) == False):
    print("The given number is not prime, please try again")
    p = int(input())


print("Enter the second prime number: ")
a = int(input())

while(isPrime(a) == False):
    print("The given number is not prime, please try again")
    a = int(input())
```

```
print("Enter private key for 1st user: ")
x1 = int(input())
y1 = power(a, x1, p)
print("The public key for 1st user is", y1)

print("Enter private key for 2nd user: ")
x2 = int(input())
y2 = power(a, x2, p)
print("The public key for 1st user is", y2)

k1 = power(y2, x1, p)
k2 = power(y1, x2, p)

print("The common secret key for 1st user is", k1)
print("The common secret key for 2nd user is", k2)
```

Output:

```
Enter a prime number:
4093082899
Enter the second prime number:
2860486313
Enter private key for 1st user:
5915587277
The public key for 1st user is 1312879516
Enter private key for 2nd user:
3367900313
The public key for 1st user is 1963984909
The common secret key for 1st user is 20893543
The common secret key for 2nd user is 20893543
PS C:\Users\adity\Desktop\ADM\College\Coding\Cryptography SS>
```

Conclusion:

In this experiment I learned about different types of key exchange algorithms and implemented the Diffie Hellman algorithm. My findings about the same are as follows:

1) The Diffie Hellman algorithm uses primitive roots of any prime numbers that are given by the user.
2) Exponentiation is a very costly operation and to use numbers of order greater than 10^6 in the power, will not work in time constraints.
3) Since the exponentiation and modulo operation are used together in this algorithm, I wrote a custom power calculation function that accepts 3 numbers (base, power, modulo number) and calculates the power using the binary exponentiation algorithm in order to minimize the time complexity.

I think the Diffie Hellman algorithm is the simplest key exchange algorithm that uses simple prime factorization concepts to make sure that information is transferred without revealing it to any outsider.