PRACTICAL: 5

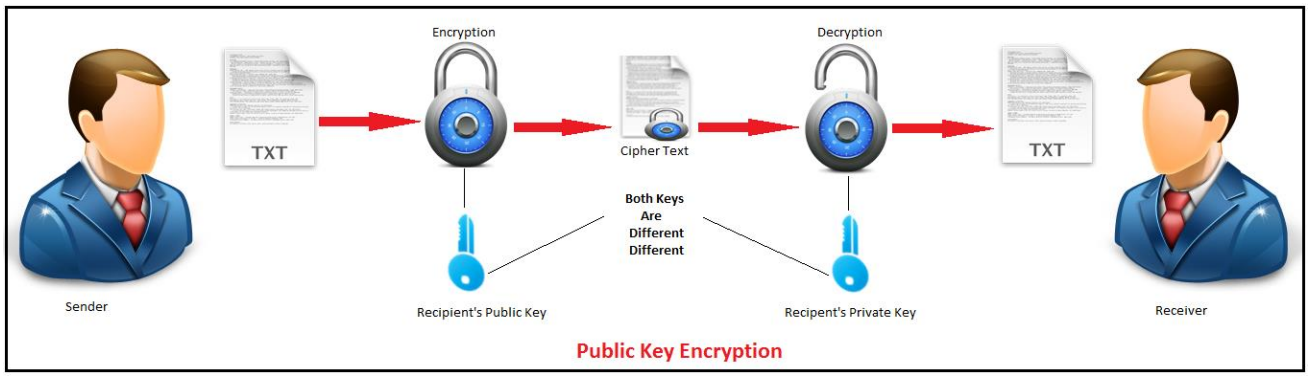
AIM:

RSA algorithm is a public key encryption technique and is considered as the most secure way of encryption to secure sensitive data, particularly when it is being sent over an insecure network such as the internet. The public and private key generation algorithm is the most complex part of the RSA algorithm. The strength of RSA is the difficulty of factoring large integers that are the product of two large prime numbers, which is considered infeasible due to the time it would take using even today's highly configured computers. Implement the RSA algorithm.

THEORY:

RSA ALGORITHM

The RSA algorithm is an asymmetric cryptography algorithm; this means that it uses a public key and a private key (i.e two different, mathematically linked keys). As their names suggest, a public key is shared publicly, while a private key is secret and must not be shared with anyone. It is named after those who invented it in 1978: Ron Rivest, Adi Shamir, and Leonard Adleman.



The sender encrypts the data with the receivers public key and creates cipher text which is send to the receiver. At receivers end the receiver decrypts the data with his own private key.

An example of asymmetric cryptography:

a) A client (for example browser) sends its public key to the server and requests for some data.

b) The server encrypts the data using clients public key and sends the encrypted data.

c) Client receives this data and decrypts it.

How it works

1. Generating the keys

a) Select two large prime numbers, x and y. The prime numbers need to be large so that they will be difficult for someone to figure out.

b) Calculate **n = (x*y)**.

c) Calculate the totient function; **phi(n) = (x-1)(y-1)**

d) Select an integer e should not be a factor of n, such that e is co-prime to **phi(n)** and **1<e<phi(n)** . The pair of numbers (n,e) makes up the **public key**.

-Note : Two integers are co-prime if the only positive integer that divides them is 1.

e) calculate phi(n): Such that **phi(n) = (P-1)(Q-1)**

f) Calculate d such that **e.d = 1 mod phi(n)**. d can be found using the extended euclidean algorithm. The pair (n,d) makes up the **private key**.

g) Now private key **d = (k*phi(n) + 1) / e** for some integer k

2. Encryption

Given a plaintext P, represented as a number, the ciphertext C is calculated as:

C = P^e mod n

eg : "HE" H = 2 and E = 3

C = 23^e mod n

3. Decryption

Using the private key (n,d), the plaintext can be found using:

P = C^d mod n

CODE:

```
import sympy
import math

def gcd(a,b):
    while(1):
        t=a%b
        if t==0:
            return b
        a=b
        b=t

if __name__=="__main__":
    primeNumber1 = sympy.randprime(1000000000,9999999999)
    primeNumber2 = sympy.randprime(1000000000,9999999999)
    n= primeNumber1*primeNumber2
    phi = (primeNumber1-1)*(primeNumber2-1)
    e=2
    while e<phi:
        track = gcd(e,phi)
        if track==1:
            break
        else:
            e+=1

    d1 = 1/e
    d= math.fmod(d1,phi)
    message = int(input("Enter Your Message : "))
    c = pow(message,e)
    m = pow(c,d)
    C = math.fmod(c,n)
    M = math.fmod(m,n)

    print(f" Original Message : {message}")
    print(f" PrimeNumber1 p : {primeNumber1}")
    print(f" PrimeNumber2 q : {primeNumber2}")
    print(f" phi(n): phi({n})= {phi}")
    print(f" e = {e}")
    print(f" d = {e}")
    print(f" Encrypted Message = {C}")
    print(f" Decrypted Message = {M}")
```

OUTPUT:

```
Enter Your Message : 700
 Original Message : 700
 PrimeNumber1 p : 6842416261
 PrimeNumber2 q : 4617376321
 phi(n): phi(31594010821966755781)= 31594010810506963200
 e = 7
 d = 7
 Encrypted Message = 1.9166278356066492e+19
 Decrypted Message = 699.9999999999998
```

```
Enter Your Message : 123456789
 Original Message : 123456789
 PrimeNumber1 p : 4513130767
 PrimeNumber2 q : 2437926419
 phi(n): phi(11002680729271033373)= 11002680722319976188
 e = 5
 d = 5
 Encrypted Message = 4.9506315291805286e+17
 Decrypted Message = 123456789.00000013
```

LATEST APPLICATIONS:

1. encrypt - needs recipient's public key
2. decrypt - needs recipient's private key
3. sign - needs the sender's private key

LEARNING OUTCOME:

In this practical we learned about the RSA Algorithm and it is used for data encryption.

REFERENCES:

1. https://cppsecrets.com/users/1004910997104971161111071171109710849496410310997105108469911109/C00-RSA-Cryptography-Algorithm-Implementation.php
2. https://embeddedsw.net/OpenPuff_Steganography_Home.html
3. GFG : https://www.geeksforgeeks.org/rsa-algorithm-cryptography/