Security Lab
Lab Assignment No. 7

Aim: Implementation and analysis of RSA cryptosystem.

RSA algorithm is a public key encryption technique and is considered as the most secure way of encryption. It was invented by Rivest, Shamir and Adleman in 1978 and hence named RSA algorithm.

Algorithm :

The RSA algorithm holds the following features:

1. The RSA algorithm is a popular exponentiation in a finite field over integers including prime numbers.
2. The integers used by this method are sufficiently large making it difficult to solve.
3. There are two sets of keys in this algorithm: private key and public key.

You will have to go through the following steps to work on RSA algorithm:

STEP 1: Generate the RSA modulus.
The initial procedure begins with selection of two prime numbers namely p and q, and then calculating their product N, as shown: ***N = p * q***
Here, let N be the specified large number.

STEP 2: Derived Number (e)
Consider number e as a derived number which should be greater than **1** and less than **(p-1)** and **(q-1)**. The primary condition will be that there should be no common factor of **(p-1)** and **(q-1)** except **1**

STEP 3: Public key
The specified pair of numbers n and e forms the RSA public key and it is made public.

STEP 4: Private Key
Private Key d is calculated from the numbers p, q and e. The mathematical relationship between the numbers is as follows: ***ed = 1 mod (p-1) (q-1)***
The above formula is the basic formula for Extended Euclidean Algorithm, which takes p and q as the input parameters.

Encryption Formula: Consider a sender who sends the plain text message to someone whose public key is (n,e). To encrypt the plain text message in the given scenario, use the following syntax: ***C = Pe mod n***

Decryption Formula: The decryption process is very straightforward and includes analytics for calculation in a systematic approach. Considering receiver C has the private key d, the result modulus will be calculated as:
Plaintext = Cd mod n

Code:

```
from math import gcd
import random

primes = [23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, \
 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, \
 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, \
 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, \
 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, \
 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997]


def multiplicative_inverse(e, phi):
    for i in range(1, phi):
        if (e * i) % phi == 1:
            return i
    return None

def generateKeys():
    [p, q] = random.sample(primes, 2)
    n = p * q
    phi_n = (p - 1) * (q - 1)

    e = 2
    while e < phi_n:
        if gcd(e, phi_n) != 1:
            e += 1
```

```
        else:
            break

    d = multiplicative_inverse(e, phi_n)
    return [n, e, d]

encrypt = lambda plainText, n, e : (plainText ** e) % n

decrypt = lambda cipherText, d, n : (cipherText ** d) % n

msg = input("\nPlease enter your message: ")

cipherArray, encryptedText, decryptedText, decryptedCipherText = [], [], [], []

n, privateKey, publicKey = generateKeys()
print(f"\nPrivate Key(e) = {privateKey}")
print(f"Public Key(d) = {publicKey}")
print(f"n = {n}\n")

for i in msg:
    cipherArray.append(encrypt(ord(i), n, privateKey))
for i in cipherArray:
    encryptedText.append(str(i))
print(f"The encrypted text is: {''.join(encryptedText)}")

for i in cipherArray:
    decryptedCipherText.append(decrypt(i, publicKey, n))
for i in decryptedCipherText:
    decryptedText.append(chr(i))

print(f"The decrypted text is: {''.join(decryptedText)}\n")
```

Output:

```
PS D:\III Year Engineering\CNS Lab Experiments> & "C:/Users/Ninad Rao/AppData/Local/Programs/Python/Python39/python.exe" "d:/III Year Engineering/CNS Lab Experiments/rsaA
lgorithm.py"

Please enter your message: My name is Ninad

Private Key(e) = 5
Public Key(d) = 9485
n = 12137

The encrypted text is: 25542037764494699686846667764763119687764534763149469910138
The decrypted text is: My name is Ninad
```

Conclusion: Hence we have successfully analyzed and written a program to implement the RSA cryptosystem.