NAME:-PRANITA BODALA

CLASS:- TYIT B

ROLL NO:-- 508

D.O.S:- 30-09-23

GRADE:-

SIGN:-

ASSIGNMENT 1

ENCRYPTION AND DECRYPTION USING RSA ALGORITHM

RSA algorithm is an asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key. As the name describes that the Public Key is given to everyone and the Private key is kept private.

An example of asymmetric cryptography:

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

2.The server encrypts the data using the client’s public key and sends the encrypted data.

3.The client receives this data and decrypts it.

Since this is asymmetric, nobody else except the browser can decrypt the data even if a third party has the public key of the browser.

The idea of RSA is based on the fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is a multiplication of two large prime numbers. And private key is also derived from the same two prime numbers. So if somebody can factorize the large number, the private key is compromised. Therefore encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024-bit keys could be broken in the near future. But till now it seems to be an infeasible task.

**Let us learn the mechanism behind the RSA algorithm : >> Generating Public Key:**

Select two prime no's. Suppose P = 53 and Q = 59.

Now First part of the Public key : n = P\*Q = 3127.

We also need a small exponent say e :

But e Must be

An integer.

Not be a factor of Φ(n).

1 < e < Φ(n) [Φ(n) is discussed below],

Let us now consider it to be equal to 3.

Our Public Key is made of n and e

**>> Generating Private Key:**

We need to calculate Φ(n) :

Such that Φ(n) = (P-1)(Q-1)

so, Φ(n) = 3016

Now calculate Private Key, d :

d = (k\*Φ(n) + 1) / e for some integer k

For k = 2, value of d is 2011.

Now we are ready with our – Public Key ( n = 3127 and e = 3) and Private Key(d = 2011) Now we will encrypt **“HI”**:

Convert letters to numbers : H = 8 and I = 9

Thus Encrypted Data c = (89e)mod n

Thus our Encrypted Data comes out to be 1394

Now we will decrypt 1394 :

Decrypted Data = (cd)mod n

Thus our Encrypted Data comes out to be 89

8 = H and I = 9 i.e. "HI".

CODE:--

import random

import math

# A set will be the collection of prime numbers,

# where we can select random primes p and q

prime = set()

public\_key = None

private\_key = None

n = None

# We will run the function only once to fill the set of

# prime numbers

def primefiller():

# Method used to fill the primes set is Sieve of

# Eratosthenes (a method to collect prime numbers)

seive = [True] \* 250

seive[0] = False

seive[1] = False

for i in range(2, 250):

for j in range(i \* 2, 250, i):

seive[j] = False

# Filling the prime numbers

for i in range(len(seive)):

if seive[i]:

prime.add(i)

# Picking a random prime number and erasing that prime

# number from list because p!=q

def pickrandomprime():

global prime

k = random.randint(0, len(prime) - 1)

it = iter(prime)

for \_ in range(k):

next(it)

ret = next(it)

prime.remove(ret)

return ret

def setkeys():

global public\_key, private\_key, n

prime1 = pickrandomprime() # First prime number

prime2 = pickrandomprime() # Second prime number

n = prime1 \* prime2

fi = (prime1 - 1) \* (prime2 - 1)

e = 2

while True:

if math.gcd(e, fi) == 1:

break

e += 1

# d = (k\*Φ(n) + 1) / e for some integer k

public\_key = e

d = 2

while True:

if (d \* e) % fi == 1:

break

d += 1

private\_key = d

# To encrypt the given number

def encrypt(message):

global public\_key, n

e = public\_key

encrypted\_text = 1

while e > 0:

encrypted\_text \*= message

encrypted\_text %= n

e -= 1

return encrypted\_text

# To decrypt the given number

def decrypt(encrypted\_text):

global private\_key, n

d = private\_key

decrypted = 1

while d > 0:

decrypted \*= encrypted\_text

decrypted %= n

d -= 1

return decrypted

# First converting each character to its ASCII value and

# then encoding it then decoding the number to get the

# ASCII and converting it to character

def encoder(message):

encoded = []

# Calling the encrypting function in encoding function

for letter in message:

encoded.append(encrypt(ord(letter)))

return encoded

def decoder(encoded):

s = ''

# Calling the decrypting function decoding function

for num in encoded:

s += chr(decrypt(num))

return s

if \_\_name\_\_ == '\_\_main\_\_':

primefiller()

setkeys()

message = "Test Message"

# Uncomment below for manual input

# message = input("Enter the message\n")

# Calling the encoding function

coded = encoder(message)

print("Initial message:")

print(message)

print("\n\nThe encoded message(encrypted by public key)\n")

print(''.join(str(p) for p in coded))

print("\n\nThe decoded message(decrypted by public key)\n")

print(''.join(str(p) for p in decoder(coded)))

OUTPUT:--

