

T.Y. B.Tech EE (2020-21)

Trimester: VII

Subject: Communication Protocols

Name: Shreerang Mhatre

Class: TY

Roll No.: 52

Batch: A3

Experiment No.: 07

Name of the Experiment: Study of RSA algorithm.

Performed on: 27/11/2023

Submitted on: 04/12/2023

Aim: To study the RSA encryption algorithm.

Prerequisite:

Basic knowledge of data communication.

Objectives:

- To understand how data encryption and decryption works.
- To learn logic behind RSA algorithm.

Theory:

What is Data Encryption?

Encryption is a security method in which information is encoded in such a way that only authorized user can read it. It uses an encryption algorithm to generate ciphertext that can only be read if decrypted.

The Need for Encryption

Beyond the obvious benefit of protecting private information from being stolen or compromised, encryption also provides a means of proving that information is authentic and comes from the point of origin it claims to come from. It can be used to verify the origin of a message and confirm that it hasn't been altered during transmission.

RSA Algorithm (Rivest-Shamir-Adleman)

RSA is one of the most successful, asymmetric encryption systems today. RSA is widely used to secure sensitive data, particularly when it is being sent over an insecure network such as the internet. The idea of RSA is based on the fact that it is difficult to factorize a large integer.

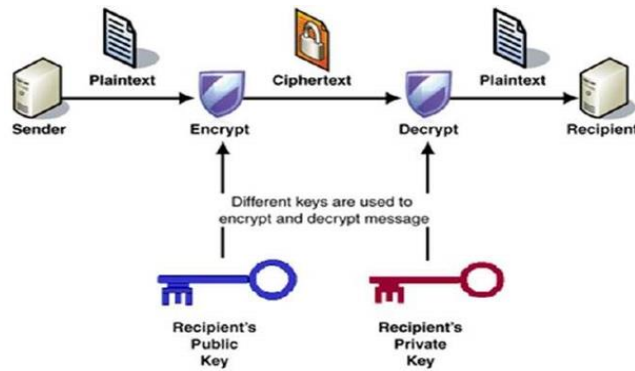
Under RSA encryption, messages are encrypted with a code called a public key, which can be shared openly. Due to some distinct mathematical properties of the RSA algorithm, once a message has been encrypted with the public key, it can only be decrypted by another key, known as the private key. Each RSA user has a key pair consisting of their public and private keys. As the name suggests, the private key must be kept secret.

The public key consists of two numbers where one number is multiplication of two large prime numbers and the private key is also derived from the same two prime numbers. Under RSA encryption, once data or a message has been turned into a cipher text with a public key, it can only be decrypted by the private key from the same key pair. Private keys are composed of d and n . We already know n , and the following equation is used to find d :

$d = 1/e \bmod \lambda(n)$. And public keys are made up of a prime number e , as well as n . The number e can be anything between 1 and the value for $\lambda(n)$.

RSA encryption can be used by previously unknown parties to securely send data between themselves as shown below:

- First, they each need to set up their own key pairs and share the public key with one another. The two entities need to keep their private keys secret in order for their communications to remain secure.
- Once the sender has the public key of their recipient, they can use it to encrypt the data that they want to keep secure. Once it has been encrypted with a public key, it can only be decrypted by the private key from the same key pair. Even the same public key can't be used to decrypt the data. This is due to the properties of trapdoor functions.
- When the recipient receives the encrypted message, they use their private key to access the data. If the recipient wants to return communications in a secure way, they can then encrypt their message with the public key of the party they are communicating with. Again, once it has been encrypted with the public key, the only way that the information can be accessed is through the matching private key.



Key Generation Algorithm:

1. Generate two large random primes, p and q , of approximately equal size such that their product $n=pq$ is of the required bit length, e.g. 1024 bits.
2. Compute $n=pq$ and $\phi=(p-1)(q-1)$.
3. Choose a random integer e , $1 < e < \phi$, such that $\gcd(e, \phi)=1$.
4. Compute the secret exponent d , $1 < d < \phi$, such that $ed \equiv 1 \pmod{\phi}$.
5. The public key is (n, e) and the private key (d, p, q) . Keep all the values d , p , q and ϕ secret.

Encryption:

1. Obtains the recipient's public key (n, e) .
2. Represents the plaintext message as an integer m with $0 < m < n$.
3. Computes the ciphertext $C = M^e \pmod{N}$
4. Sends the ciphertext C to the recipient.

Decryption:

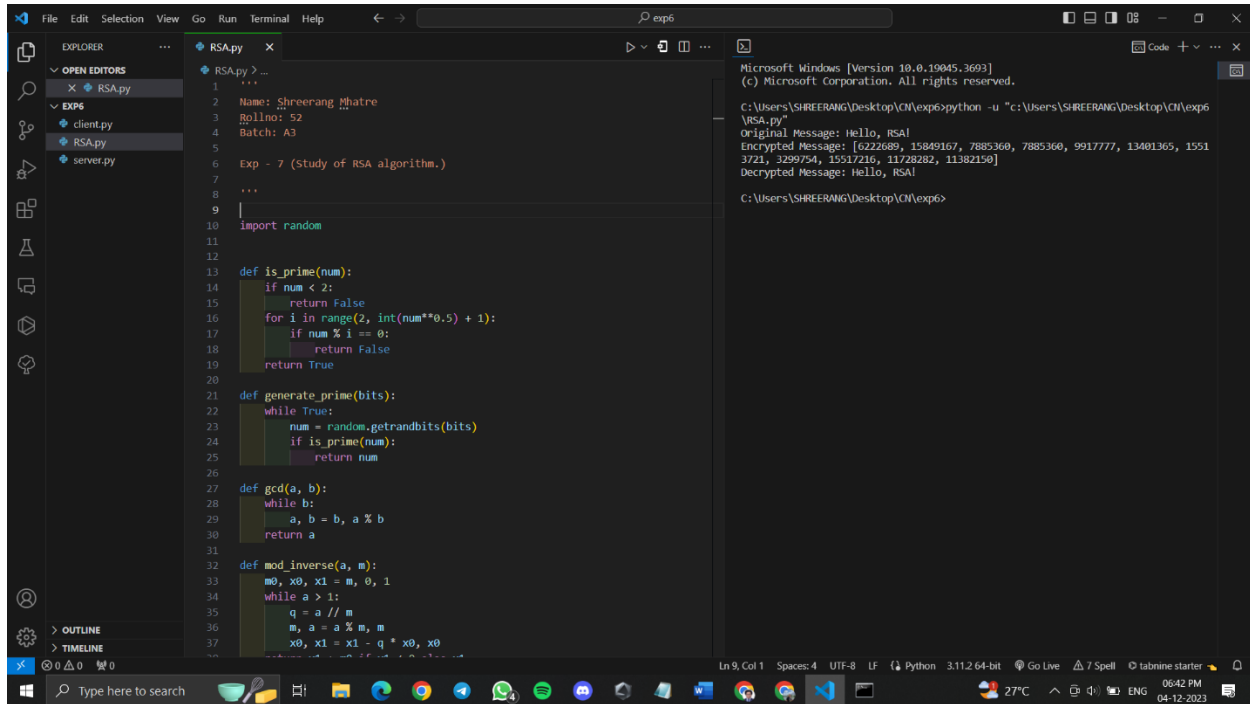
1. Uses his private key (n, d) to compute $M = C^d \pmod{N}$
2. Extracts the plaintext from the message representative M .

Conclusion:

Post Lab Questions:

1. Describe the symmetric and asymmetric key cryptography?
2. Enlist and briefly describe the important features of five most popular algorithms for cryptography.
3. Compare RSA algorithm with any other cryptography algorithms

Execution of RSA Algorithm:



The screenshot shows the Visual Studio Code editor with the file explorer on the left. The 'EXPLORER' pane shows a project named 'exp6' with files 'client.py', 'RSA.py', and 'server.py'. The 'RSA.py' file is open in the editor, showing the following code:

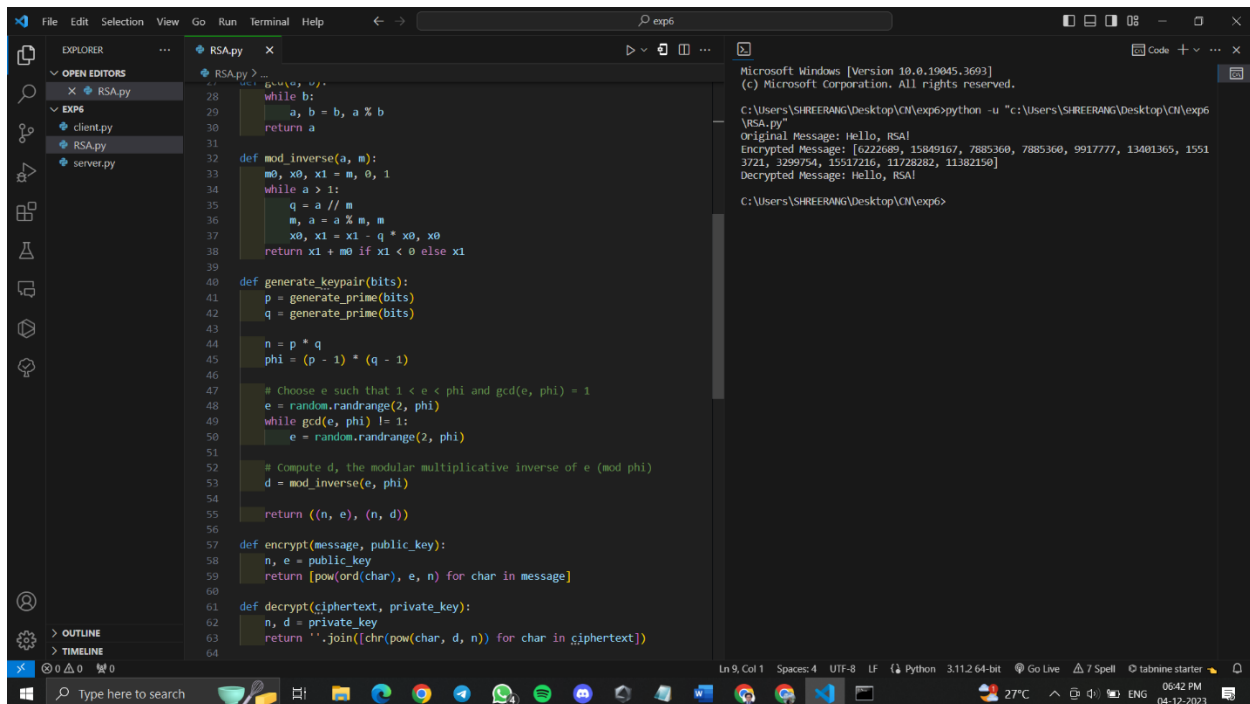
```
1  ...
2  Name: Shreerang Mhatre
3  Rollno: 52
4  Batch: A3
5
6  Exp - 7 (Study of RSA algorithm.)
7  ...
8
9
10 import random
11
12
13 def is_prime(num):
14     if num < 2:
15         return False
16     for i in range(2, int(num**0.5) + 1):
17         if num % i == 0:
18             return False
19     return True
20
21 def generate_prime(bits):
22     while True:
23         num = random.getrandbits(bits)
24         if is_prime(num):
25             return num
26
27 def gcd(a, b):
28     while b:
29         a, b = b, a % b
30     return a
31
32 def mod_inverse(a, m):
33     m0, x0, x1 = m, 0, 1
34     while a > 1:
35         q = a // m
36         m, a = a % m, m
37         x0, x1 = x1 - q * x0, x0
```

The terminal window on the right shows the execution of the script:

```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\SHREERANG\Desktop\exp6>python -u "C:\Users\SHREERANG\Desktop\exp6\RSA.py"
Original Message: Hello, RSA!
Encrypted Message: [6222689, 15849167, 7885360, 7885360, 9917777, 13481365, 15513721, 3299754, 15517216, 11728282, 11382150]
Decrypted Message: Hello, RSA!

C:\Users\SHREERANG\Desktop\exp6>
```



The screenshot shows the Visual Studio Code editor with the file explorer on the left. The 'EXPLORER' pane shows a project named 'exp6' with files 'client.py', 'RSA.py', and 'server.py'. The 'RSA.py' file is open in the editor, showing the following code:

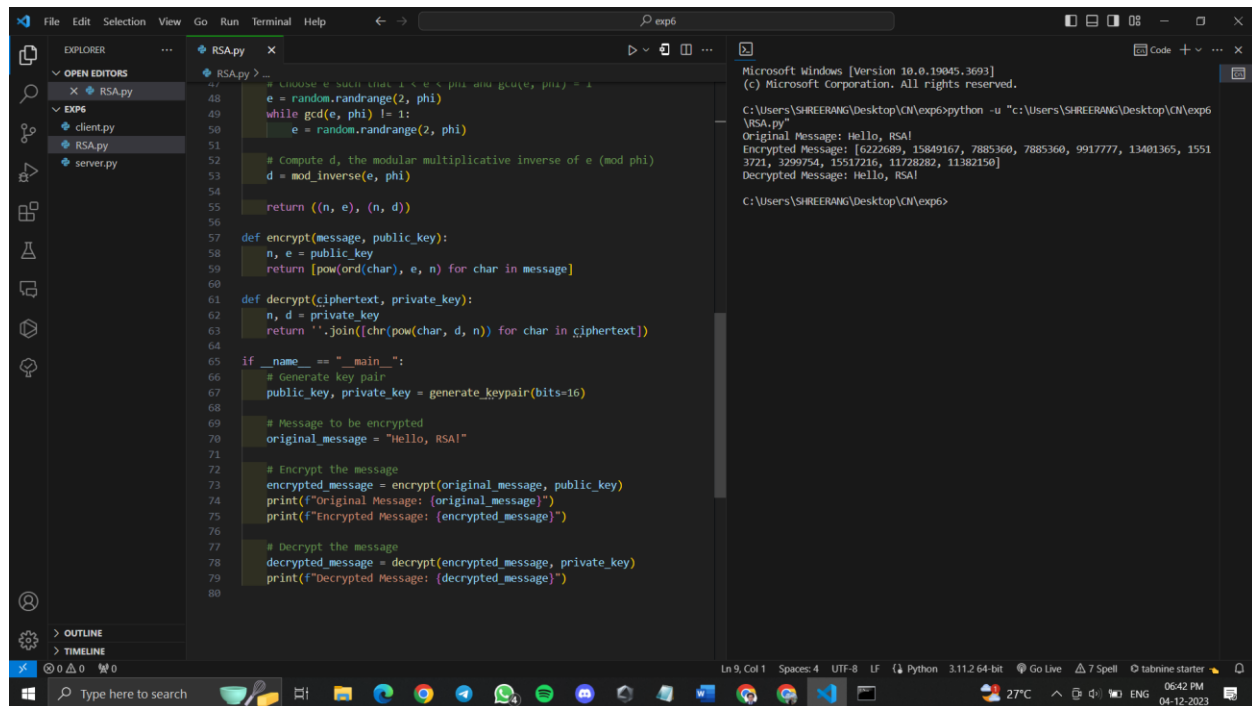
```
28     while b:
29         a, b = b, a % b
30     return a
31
32 def mod_inverse(a, m):
33     m0, x0, x1 = m, 0, 1
34     while a > 1:
35         q = a // m
36         m, a = a % m, m
37         x0, x1 = x1 - q * x0, x0
38     return x1 + m0 if x1 < 0 else x1
39
40 def generate_keypair(bits):
41     p = generate_prime(bits)
42     q = generate_prime(bits)
43
44     n = p * q
45     phi = (p - 1) * (q - 1)
46
47     # Choose e such that 1 < e < phi and gcd(e, phi) = 1
48     e = random.randrange(2, phi)
49     while gcd(e, phi) != 1:
50         e = random.randrange(2, phi)
51
52     # Compute d, the modular multiplicative inverse of e (mod phi)
53     d = mod_inverse(e, phi)
54
55     return ((n, e), (n, d))
56
57 def encrypt(message, public_key):
58     n, e = public_key
59     return [pow(ord(char), e, n) for char in message]
60
61 def decrypt(ciphertext, private_key):
62     n, d = private_key
63     return ''.join([chr(pow(char, d, n)) for char in ciphertext])
64
```

The terminal window on the right shows the execution of the script:

```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\SHREERANG\Desktop\exp6>python -u "C:\Users\SHREERANG\Desktop\exp6\RSA.py"
Original Message: Hello, RSA!
Encrypted Message: [6222689, 15849167, 7885360, 7885360, 9917777, 13481365, 15513721, 3299754, 15517216, 11728282, 11382150]
Decrypted Message: Hello, RSA!

C:\Users\SHREERANG\Desktop\exp6>
```



Exp 7

* Past lab Questions:

Q1) Describe the symmetric and asymmetric key cryptography?

→ ① Symmetric key cryptography -

It is a technique that uses a single key for both encryption and decryption of data. This makes it faster than asymmetric key cryptography, but it requires secure key distribution. In symmetric key cryptography the same key is used to encrypt and decrypt the data making it less secure than asymmetric key cryptography.

② Asymmetric key cryptography -

It uses two different keys to encrypt and decrypt data. These keys are known as the public key and the private key. The public key is used to encrypt the data, while the private key is used to decrypt the data.

Asymmetric key cryptography is more secure than symmetric key cryptography but it is slower.

Q2) Enlist and briefly describe the important features of five most popular algorithms for cryptography

→ ① Advanced Encryption Standard (AES) -
AES is a symmetric key encryption algorithm that is widely used for securing sensitive data.

② Triple Data Encryption Standard (3DES) -
3DES is a symmetric key encryption algorithm that is based on the original DES algorithm.

③ Rivest-Shamir-Adleman (RSA) -
RSA is an asymmetric key encryption algorithm that is widely used for securing data in transit.

④ Blowfish -
Blowfish is a symmetric key encryption algorithm that is used for securing data in transit.

⑤ Elliptic Curve Cryptography (ECC) -
It is used for securing data in transit

Q3) Compare RSA algorithm with any other cryptography algorithm.

→

① RSA vs AES -

RSA is an asymmetric key encryption algorithm, while AES is symmetric key encryption algorithm. RSA is slower than AES, but it is more secure. AES is used for encryption large amounts of data, while RSA is used for encrypting small amounts of data.

② RSA vs DSA -

RSA & DSA are both asymmetric key encryption algorithms. RSA is faster than DSA, but it requires larger key sizes to be secure. DSA is slower than RSA but requires smaller key sizes to be secure.

③ RSA vs Blowfish -

RSA & Blowfish are both symmetric key encryption algorithms. RSA is used for encrypting small amounts of data, while Blowfish is used for encrypting large amounts of data.

④ RSA vs ECC -

RSA and ECC are both asymmetric key encryption algorithms. ECC is faster and more secure than RSA, but it is not as widely used. RSA is slower than ECC, but is more widely used.

⑤ RSA vs 3DES -

RSA is used for encrypting small amounts of data, while 3DES is used for encrypting large amounts of data. RSA is slower than 3DES, but it is more secure. 3DES is faster but it is less secure.