**Implementation of RSA Encryption Algorithm**

|  |  |
| --- | --- |
| **Supervisor**  **Dr. Narendran Rajagopalan,**  **Associate Professor,**  **NITPY.** | **Assignment by,**  **Gedela. Sailaja (CS22B1013).** |

**Introduction**

RSA (Rivest–Shamir–Adleman) is a public-key encryption algorithm used for secure data transmission. It is an asymmetric encryption algorithm, meaning it uses a pair of keys: a public key for encryption and a private key for decryption. In this assignment, we implement RSA key generation, encryption, and decryption using Python.

**Implementation:**

import random

def gcd(a, b):

while b != 0:

a, b = b, a % b

return a

def modinv(e, phi):

def extended\_gcd(a, b):

if b == 0:

return a, 1, 0

g, x, y = extended\_gcd(b, a % b)

return g, y, x - (a // b) \* y

g, x, \_ = extended\_gcd(e, phi)

if g != 1:

raise Exception('Modular inverse does not exist')

return x % phi

def generate\_keys():

p = 61

q = 53

n = p \* q

phi = (p - 1) \* (q - 1)

e = random.randrange(2, phi)

while gcd(e, phi) != 1:

e = random.randrange(2, phi)

d = modinv(e, phi)

return (e, n), (d, n)

def encrypt(public\_key, plaintext):

e, n = public\_key

cipher = [pow(ord(char), e, n) for char in plaintext]

return cipher

def decrypt(private\_key, ciphertext):

d, n = private\_key

plain = [chr(pow(char, d, n)) for char in ciphertext]

return ''.join(plain)

if \_\_name\_\_ == "\_\_main\_\_":

public\_key, private\_key = generate\_keys()

print("Public Key:", public\_key)

print("Private Key:", private\_key)

message = "HELLO"

print("\nOriginal Message:", message)

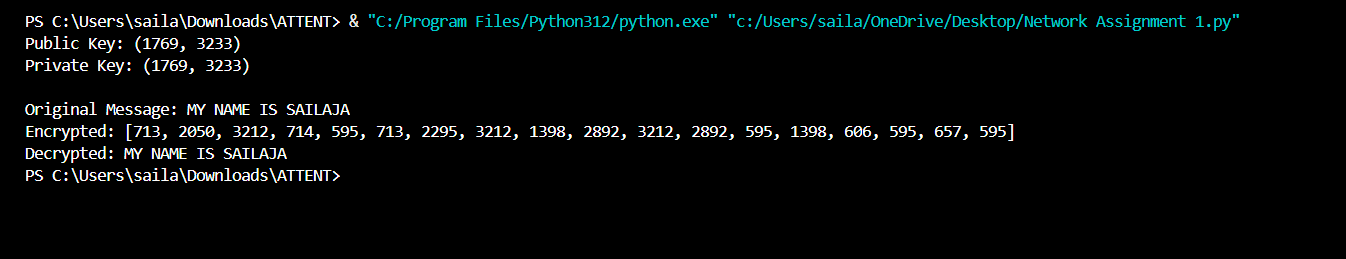
encrypted\_msg = encrypt(public\_key, message)

print("Encrypted:", encrypted\_msg)

decrypted\_msg = decrypt(private\_key, encrypted\_msg)

print("Decrypted:", decrypted\_msg)

**Output:**



**Explanation:**

* Key Generation: Two prime numbers p and q are selected, and RSA keys are generated using n = p \* q and Euler's totient function φ(n).
* Encryption: Each character in the plaintext is converted to its ASCII value and encrypted using the public key.
* Decryption: Encrypted values are decrypted using the private key to retrieve the original message.

**Conclusion:**

This assignment demonstrates the implementation of the RSA encryption algorithm in Python. The algorithm securely encrypts and decrypts messages using asymmetric key pairs. The output shows successful encryption and decryption, verifying the correctness of the implementation.