Experiment No. 4

Title: RSA Cipher

Batch: A3 **Roll No.: 16010421073** **Experiment No.:4**

Aim: To implement RSA cipher

Resources needed: Windows/Linux

Theory: Pre Lab/ Prior Concepts:

RSA is a public key algorithm named after its inventers Rivest, Shamir and Adleman. The characteristics of public key cryptography (which is also called as Asymmetric Cryptography) are as below:

- It has two keys. One key is called as private key and the other one is called as public key. Everyone who uses this cryptography has to have two keys each.
- Keys used for encryption and decryption should be different. If one is used for encryption, then other must be used for decryption. Any key can be used for encryption and then remaining key can be used for decryption.
- Public Key Cryptography is based on solid foundation of mathematics.
- It has large computational overheads. Hence ciphertext generated is of much larger size than the plaintext. Hence it is normally used for encrypting small size of data blocks. For example, passwords, symmetric keys, etc. It is not preferred to encrypt large files.
- RSA algorithm gets its security from the fact that it is extremely difficult to factorize large prime number.
- Security of the algorithm depends on the size of the key. Greater the size of the key, larger is the security. The key length is variable. The most commonly used key length is 512 bits.

Procedure / Approach /Algorithm / Activity Diagram:

A. Key generation Algorithm:

1. Choose large prime numbers p and q.
2. Calculate product n= pq. The value of n can be revealed publicly, but n is large enough that even supercomputers cannot factor it in a reasonable amount of time (years or even centuries).
3. Calculate phi = (p-1)(q-1)
4. Select e < phi such that it is relatively prime to phi. The public key is (e, n).
5. Determine d such that ed = 1 mod phi. The private key is (d, n). d, p ,q and phi are kept secret, only (e,n) is made public.

B. Encryption:

6. Suppose M is the message.
7. Encrypt this message using public key of the receiver as follows
   C = M^e mod n.

C. Decryption:

8. Decrypt the C generated in step 7 using private key of the receiver as follows M = C^d mod n.

Results: (Program printout as per the format)
Code:

```
def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

def mod_inverse(e, phi):
    for d in range(2, phi):
        if (e * d) % phi == 1:
            return d
    return None

def generate_keys():
    p = int(input("Enter first prime number 'p': "))
    q = int(input("Enter second prime number 'q': "))
    n = p * q
    phi = (p - 1) * (q - 1)
    e = int(input(f"Enter a value of e (1 < e < {phi}) that is coprime
with{phi}: "))
    d = mod_inverse(e, phi)
    return ((e, n), (d, n))

def encrypt(public_key, message):
    e, n = public_key
    encrypted_message = [pow(ord(char), e, n) for char in message]
    return encrypted_message

def decrypt(private_key, encrypted_message):
    d, n = private_key
    decrypted_message = ''.join([chr(pow(char, d, n)) for char in
encrypted_message])
    return decrypted_message

def main():
    print("RSA Algorithm Program")
    while True:
        print("\nMenu:")
        print("1. Generate RSA Key Pair")
        print("2. Encrypt Message")
```

```
        print("3. Decrypt Message")
        print("4. Exit")
        choice = input("Enter your choice: ")

        if choice == "1":
            public_key, private_key = generate_keys()
            print("Public Key (e, n):", public_key)
            print("Private Key (d, n):", private_key)

        elif choice == "2":
            public_key = tuple(map(int, input("Enter recipient public key (e,
n): ").split()))
            message = input("Enter the message to encrypt: ")
            encrypted_message = encrypt(public_key, message)
            print("Encrypted Message:", encrypted_message)

        elif choice == "3":
            private_key = tuple(map(int, input("Enter your private key (d, n):
").split()))
            encrypted_message = list(map(int, input("Enter the encrypted
message (space-separated integers): ").split()))
            decrypted_message = decrypt(private_key, encrypted_message)
            print("Decrypted Message:", decrypted_message)

        elif choice == "4":
            print("Exiting the program.")
            break

        else:
            print("Invalid choice. Please select a valid option.")


if __name__ == "__main__":
    main()
```

Output:

```
RSA Algorithm Program

Menu:
1. Generate RSA Key Pair
2. Encrypt Message
3. Decrypt Message
4. Exit
Enter your choice: 1
Enter first prime number 'p': 31
Enter second prime number 'q': 17
Enter a value of e (1 < e < 480) that is coprime with480: 113
Public Key (e, n): (113, 527)
Private Key (d, n): (17, 527)

Menu:
1. Generate RSA Key Pair
2. Encrypt Message
3. Decrypt Message
4. Exit
Enter your choice: 2
Enter recipient public key (e, n): 113 527
Enter the message to encrypt: Hello Mumbai
Encrypted Message: [293, 16, 244, 244, 162, 32, 213, 83, 500, 149, 250, 394]

Menu:
1. Generate RSA Key Pair
2. Encrypt Message
3. Decrypt Message
4. Exit
Enter your choice: 3
Enter your private key (d, n): 17 527
Enter the encrypted message (space-separated integers): 293  16  244 244  162  32  213  83 500 149 250 394
Decrypted Message: Hello Mumbai

Menu:
1. Generate RSA Key Pair
2. Encrypt Message
3. Decrypt Message
4. Exit
Enter your choice: 4
Exiting the program.
```

Questions:

1. **In RSA cryptosystem each plaintext character is presented by the number between 00(A) and 25(Z). The number 26 represents the blank character. Bob wants to send Alice the message "Hello World". So the plaintext is as below, 07 04 11 11 14 26 22 14 17 11 03 . Suppose p=11, q=3. Generate receiver's key pair and show encryption and decryption of the message using RSA cipher.**

Ans:

```python
def generate_key_pair(p, q):
    n = p * q
    phi_n = (p - 1) * (q - 1)
    e = 65537
    d = pow(e, -1, phi_n)
    return (e, n), (d, n)

def encrypt(message, public_key):
    e, n = public_key
    ciphertext = [pow(c, e, n) for c in message]
    return ciphertext

def decrypt(ciphertext, private_key):
    d, n = private_key
    plaintext = [pow(c, d, n) for c in ciphertext]
    return plaintext
```

```
def main():
  p = 11
  q = 3
  public_key, private_key = generate_key_pair(p, q)

  message = [7, 4, 11, 11, 14, 26, 22, 14, 17, 11, 3]
  ciphertext = encrypt(message, public_key)
  plaintext = decrypt(ciphertext, private_key)

  print("Plaintext:", message)
  print("Ciphertext:", ciphertext)
  print("Decrypted text:", plaintext)

if __name__ == "__main__":
  main()
```

```
Plaintext: [7, 4, 11, 11, 14, 26, 22, 14, 17, 11, 3]
Ciphertext: [28, 16, 11, 11, 20, 5, 22, 20, 8, 11, 9]
Decrypted text: [7, 4, 11, 11, 14, 26, 22, 14, 17, 11, 3]
```

2. List the attacks on RSA.

Ans:

1. Plain text Attack

Plain text attacks are classified into three categories

- **Short message attack:** In this type of attack, the assumption is that the attacker knows some blocks of the plain text message. If an attacker knows some block of plain text, then he could try to encrypt the blocks of plain text using the information and try to convert it into cipher text. To prevent a short message attack, we can use the padding bits for encryption.
- **Cycling attack:** In cycling attack, the reverse process is done. An attacker assumes that the ciphertext is formed using some permutations operations. If the attacker assumption becomes true, then he can try the reverse process to obtain the plain text from the cipher text.
- **Unconcealed message attack:** In some rare cases, it is found that some encrypted cipher text is the same as the plain text i.e original text. This means that the plain text is not hidden. Such type of attack is called an unconcealed message attack

2. Chosen cipher Attack

In this type of attack, the attacker can find out the plain text from cipher text using the extended euclidean algorithm.

3. Factorization Attack

In factorization Attack, the attacker impersonates the key owners, and with the help of the stolen cryptographic data, they decrypt sensitive data, bypass the security of the system. This attack occurs on An RSA cryptographic library which is used to generate RSA Key. By doing this, Attackers can have the private keys of n number of security tokens, smartcards, Motherboard Chipsets by having a target's public key.

.

Outcomes: CO2: Illustrate different cryptographic algorithms for security.

Conclusion: (Conclusion to be based on the objectives and outcomes achieved)

Thus we successfully implemented RSA Algorithm

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References: Books/ Journals/ Websites:

1. Behrouz A. Forouzan, “Cryptography and Network Security”, Tata McGraw Hill
2. William Stalling, “Cryptography and Network Security”, Prentice Hall