PRACTICAL-5

AIM: RSA algorithm is a public key encryption technology and is considered the most secure way of encryption to secure sensitive data, particularly when it is being sent over an insecure network such as the internet. The public and private key generation algorithm is the most complex part of the RSA algorithm. The strength of RSA is the difficulty of factoring large integers that are the product of two large prime numbers, which is considered infeasible due to the time it would take using even today's highly configured computers. Implement the RSA algorithm.

THEORY:

Public Key encryption algorithm is also called the Asymmetric algorithm. Asymmetric algorithms are those algorithms in which sender and receiver use different keys for encryption and decryption. Each sender is assigned a pair of keys:

- **Public key**
- **Private key**

The **Public key** is used for encryption, and the **Private Key** is used for decryption. Decryption cannot be done using a public key. The two keys are linked, but the private key cannot be derived from the public key. The public key is well known, but the private key is secret and it is known only to the user who owns the key. It means that everybody can send a message to the user using user's public key. But only the user can decrypt the message using his private key.

RSA Algorithm are as follows:

1. We will take two large prime numbers, x, and y. The prime numbers need to be large so that they will be difficult for someone to figure out.
2. Calculate n = x*y
3. Calculate the totient function; $\phi(n)=(x-1)(y-1)$.
4. We also need a small exponent to say e. e must be an integer, but it should not be a factor of n. $1 < e < \phi(n)$.
5. The public key consists of n and e.
6. Now calculate Private Key, d, $d = (k*\Phi(n) + 1) / e$ for some integer k.
7. A plain message m is encrypted using public keys e and n. To convert the plaintext into ciphertext, we use the following formula to get ciphertext C. C = m^e mod n
8. A ciphertext c is decrypted using private key d and n. We use the following formula to calculate plain text m from the ciphertext c. m = c^d mod n.

CODE:

```
#include<iostream>
#include<math.h>
#include<string.h>
#include<stdlib.h>

using namespace std;

long int p, q, n, t, flag, e[100], d[100], temp[100], j, m[100], en[100], i;
char msg[100];
int prime(long int);
void ce();
long int cd(long int);
void encrypt();
void decrypt();
int prime(long int pr)
{
   int i;
   j = sqrt(pr);
   for (i = 2; i <= j; i++)
   {
      if (pr % i == 0)
         return 0;
   }
   return 1;
}
int main()
{
   cout << "\nENTER FIRST PRIME NUMBER\n";
   cin >> p;
   flag = prime(p);
   if (flag == 0)
   {
      cout << "\nWRONG INPUT\n";
      exit(1);
   }
   cout << "\nENTER ANOTHER PRIME NUMBER\n";
   cin >> q;
   flag = prime(q);
   if (flag == 0 || p == q)
   {
      cout << "\nWRONG INPUT\n";
      exit(1);
   }
   cout << "\nENTER MESSAGE\n";
   fflush(stdin);
   cin >> msg;
```

```
    for (i = 0; msg[i] != '\0'; i++)
        m[i] = msg[i];
    n = p * q;
    t = (p - 1) * (q - 1);
    ce();
    cout << "\nPOSSIBLE VALUES OF e AND d ARE\n";
    for (i = 0; i < j - 1; i++)
        cout << e[i] << "\t" << d[i] << "\n";
    encrypt();
    decrypt();
    return 0;
}
void ce()
{
    int k;
    k = 0;
    for (i = 2; i < t; i++)
    {
        if (t % i == 0)
            continue;
        flag = prime(i);
        if (flag == 1 && i != p && i != q)
        {
            e[k] = i;
            flag = cd(e[k]);
            if (flag > 0)
            {
                d[k] = flag;
                k++;
            }
            if (k == 99)
                break;
        }
    }
}
long int cd(long int x)
{
    long int k = 1;
    while (1)
    {
        k = k + t;
        if (k % x == 0)
            return (k / x);
    }
}
void encrypt()
{
    long int pt, ct, key = e[0], k, len;
```

```
  i = 0;
    len = strlen(msg);
    while (i != len)
    {
        pt = m[i];
        pt = pt - 96;
        k = 1;
        for (j = 0; j < key; j++)
        {
            k = k * pt;
            k = k % n;
        }
        temp[i] = k;
        ct = k + 96;
        en[i] = ct;
        i++;
    }
    en[i] = -1;
    cout << "\nTHE ENCRYPTED MESSAGE IS\n";
    for (i = 0; en[i] != -1; i++)
        printf("%c", en[i]);
}
void decrypt()
{
    long int pt, ct, key = d[0], k;
    i = 0;
    while (en[i] != -1)
    {
        ct = temp[i];
        k = 1;
        for (j = 0; j < key; j++)
        {
            k = k * ct;
            k = k % n;
        }
        pt = k + 96;
        m[i] = pt;
        i++;
    }
    m[i] = -1;
    cout << "\nTHE DECRYPTED MESSAGE IS\n";
    for (i = 0; m[i] != -1; i++)
        printf("%c", m[i]);
}
```

.

OUTPUT:

```
Output                                                    Clear

/tmp/krrcfYAmXR.o
ENTER FIRST PRIME NUMBER
11
ENTER ANOTHER PRIME NUMBER
13
ENTER MESSAGE
Dhaval20IT162
POSSIBLE VALUES OF e AND d ARE
7   103
17  113
19  19
23  47
29  29
31  31
37  13
41  41
43  67

THE ENCRYPTED MESSAGE IS
!aval  ITA
THE DECRYPTED MESSAGE IS
Dhaval20IT162
```

```
Output                                                    Clear

/tmp/krrcfYAmXR.o
ENTER FIRST PRIME NUMBER
11
ENTER ANOTHER PRIME NUMBER
6
WRONG INPUT
```

LATEST APPLICATION:

- **Secure online communications:** RSA is often used to encrypt and decrypt data transmitted over the internet, such as email and instant messaging.

- **Digital signatures:** RSA can be used to create digital signatures, which can be used to ensure the authenticity of electronic documents.

- **Secure file transfer:** RSA can be used to encrypt files before they are transferred over a network, ensuring that only authorized individuals can access the information.

- **Virtual private networks (VPNs):** RSA is commonly used in VPNs to secure communications between a remote user and a private network.

LEARNING OUTCOME:

Here we learned about the RSA algorithm which is an asymmetric algorithm that uses public and private keys for encryption and decryption. The receiver's public key is used to encrypt the data and the receiver's private key to decrypt the data.

REFERENCES:

1. https://www.javatpoint.com/rsa-encryption-algorithm
2. https://chat.openai.com/chat