04.09.2020

Ex No: 4

IMPLEMENTATION OF RSA

AIM:

To write a C program to implement the RSA encryption algorithm.

DESCRIPTION:

RSA is an algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of them can be given to everyone. A basic principle behind RSA is the observation that it is practical to find three very large positive integers e, d and n such that with modular exponentiation for all integer m:

$$(m^e)^d = m \pmod{n}$$

The public key is represented by the integers n and e; and, the private key, by the integer d. m represents the message. RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. The intention is that messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key.

ALGORITHM:

STEP-1: Select two co-prime numbers as p and q.

STEP-2: Compute n as the product of p and q.

STEP-3: Compute (p-1)*(q-1) and store it in z.

STEP-4: Select a random prime number e that is less than that of z.

STEP-5: Compute the private key, d as e * $mod^{-1}(z)$.

STEP-6: The cipher text is computed as $message^{e}$* mod n.

STEP-7: Decryption is done as $cipher^{d}$mod n.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
     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();
```

```
void main()
{
      clrscr();
      printf("\nENTER FIRST PRIME NUMBER\n");
      scanf("%d",&p);
      flag=prime(p);
      if(flag==0)
       {
           printf("\nWRONG INPUT\n");
           getch();
       }
       printf("\nENTER ANOTHER PRIME NUMBER\n");
       scanf("%d",&q);
       flag=prime(q);
       if(flag==0||p==q)
       {
        printf("\nWRONG INPUT\n");
        getch();
       }
       printf("\nENTER MESSAGE\n");
       fflush(stdin);
       scanf("%s",msg);
       for(i=0;msg[i]!=NULL;i++)
       m[i]=msg[i];
       n=p*q;
       t=(p-1)*(q-1);
       ce();
       printf("\nPOSSIBLE VALUES OF e AND d ARE\n");
       for(i=0;i<j-1;i++)
       printf("\n%ld\t%ld",e[i],d[i]);
       encrypt();
```

```
        decrypt();
        getch();
}
int prime(long int pr)
{
int i;
j=sqrt(pr);
for(i=2;i<=j;i++)
{
if(pr%i==0)
return 0;
}
return 1;
}
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;
```

```
printf("\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;


printf("\nTHE DECRYPTED MESSAGE
IS\n"); for(i=0;m[i]!=-1;i++)
printf("%c",m[i]);


}
```

```
void generateSymmetricKey() {

try {

     Random r = new Random();

     int num = r.nextInt(10000);

     String knum = String.valueOf(num);

     byte[] knumb = knum.getBytes();

     skey=getRawKey(knumb);

     skeyString = new String(skey);

     System.out.println("DES Symmetric key = "+skeyString);

}

catch(Exception e)

{

     System.out.println(e);

}

}


private static byte[] getRawKey(byte[] seed) throws Exception

{

     KeyGenerator kgen = KeyGenerator.getInstance("DES");

     SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");

     sr.setSeed(seed);

     kgen.init(56, sr);


     SecretKey skey =
     kgen.generateKey(); raw =
     skey.getEncoded(); return raw;

}


private static byte[] encrypt(byte[] raw, byte[] clear)
throws Exception {


          SecretKeySpec skeySpec = new
          SecretKeySpec(raw, "DES");
```

```
            Cipher cipher =
            Cipher.getInstance("DES");
            cipher.init(Cipher.ENCRYPT_MODE,
            skeySpec); byte[] encrypted =
            cipher.doFinal(clear); return
            encrypted;

    }

    private static byte[] decrypt(byte[] raw, byte[] encrypted)

    throws Exception

    {


            SecretKeySpec skeySpec = new
            SecretKeySpec(raw, "DES");


            Cipher cipher =
            Cipher.getInstance("DES");
            cipher.init(Cipher.DECRYPT_MODE,
            skeySpec); byte[] decrypted =
            cipher.doFinal(encrypted); return
            decrypted;

    }


    public static void main(String
         args[]) { DES des = new DES();

    }

    }
```

OUTPUT:

```
> clang++-7 -pthread -std=c++17 -o main main.cpp
> ./main
Message data = 12
p = 3
q = 7
n = pq = 21
totient = 12
e = 5
d = 5
Encrypted data = 3
Original Message sent = 12>
```

RESULT:

Thus the C program to implement RSA encryption technique had been implemented successfully