Pimpri Chinchwad College of Engineering
Computer Laboratory - III

Assignment No - 3

1 Aim

Encrypt message to be transmitted using RSA algorithm. Decrypt the encrypted message using RSA and show that it is equivalent to original message. Develop program in C++/Python/Java.

2 Objective

- To understand the working of RSA.
- To encrypt data using password for preventing unauthorized access.

3 Software Requirements

- Linux Operating System
- Java,C++

4 Mathematical Model

S = s, e, x, y, Fme, DD, NDD

S = Initial State

E = End State
X = Input Value
X={x1,x2, x3}
where, x1=p
x2=q
x3=plaintext message
Y =Output
Y={y1,y2}
y1={"Public Key"}
y2={"Private Key"}
Fme = Main function
Fme={f1}
f1="RSA Algorithm to perform data encryption".
Pu=RSA(x3)
Pr=RSA(x3)
where, Pu = Pubic Key.
Pr = Private Key.
DD = Deterministic data {plaintext}
NDD = Non Deterministic Data {public key, private key}

5 Theory

RSA is one of the first practical public-key cryptosystems and is widely used for secure data transmission. In such a cryptosystem, the encryption key is public and differs from the decryption key which is kept secret. In RSA, this asymmetry is based on the practical difficulty of factoring the product of two large prime numbers, the factoring problem. RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who first publicly described the algorithm in 1977. Clifford Cocks, an English mathematician working for the UK intelligence agency GCHQ, had developed an equivalent system in 1973, but it was not declassified until 1997.

A user of RSA creates and then publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime numbers can feasibly decode the message. Breaking RSA encryption is known as the RSA problem; whether it is as

hard as the factoring problem remains an open question.

RSA is a relatively slow algorithm, and because of this it is less commonly used to directly encrypt user data. More often, RSA passes encrypted shared keys for symmetric key cryptography which in turn can perform bulk encryption-decryption operations at much higher speed.

Many protocols like SSH, OpenPGP, S/MIME, and SSL/TLS rely on RSA for encryption and digital signature functions. It is also used in software programs – browsers are an obvious example, which need to establish a secure connection over an insecure network like the Internet or validate a digital signature. RSA signature verification is one of the most commonly performed operations in IT.

RSA derives its security from the difficulty of factoring large integers that are the product of two large prime numbers. Multiplying these two numbers is easy, but determining the original prime numbers from the total – factoring – is considered infeasible due to the time it would take even using todays super computers.

The public and the private key-generation algorithm is the most complex part of RSA cryptography. Two large prime numbers, p and q, are generated using the Rabin-Miller primality test algorithm. A modulus n is calculated by multiplying p and q. This number is used by both the public and private keys and provides the link between them. Its length, usually expressed in bits, is called the key length. The public key consists of the modulus n, and a public exponent, e, which is normally set at 65537, as it's a prime number that is not too large. The e figure doesn't have to be a secretly selected prime number as the public key is shared with everyone. The private key consists of the modulus n and the private exponent d, which is calculated using the Extended Euclidean algorithm to find the multiplicative inverse with respect to the totient of n.

6 Algorithm

6.1 RSA algorithm

RSA algorithm consists of 2 tasks:

1. **Public Key and Private Key generation:**

   - Choose two distinct prime numbers p and q. For security purposes, the integers p and q should be chosen at random, and should be of similar bit-length of 1024 bits or higher.
   - Compute n = p*q. n is used as the modulus for both public and private keys. Its length, usually expressed in bits, is the key length.
   - Compute m = (p-1)*(q-1). m is actually the Euler's totient function value of n.
   - Choose an integer e such that 1 ¡ e ¡ n and greatest common divisor gcd(e, m) = 1; i.e., e and m are coprime numbers.
   - Compute d such that d*e mod m = 1. d is also called the modular multiplicative inverse of e with modulo m.
   - Package the public key as n,e.
   - Package the private key as n,d.

2. **Message encryption and decryption:**

   To encrypt a message, the sender can follow these steps:

   - Divide the encrypted message back into blocks of the same block size used in the encryption process.
   - Decrypt the block with the private key as C^d mod n.
   - Put decrypted block together to get the original message.

6.2 Square And Multiply Algorithm

```
Function exp-by-squaring-iterative(x, n)
if n < 0 then
  x := 1 / x;
  n := -n;
if n = 0 then return 1
y := 1;
while n > 1 do
  if n is even then
    x := x * x;
    n := n / 2;
  else
```

```
    y := x * y;
    x := x * x;
    n := (n  1) / 2;
return x * y
```

6.3 Extended Euclidean Algorithm

To find multiplicative inverse:

```
  ExtEuclid (a,b) {
// returns a triple (d,s,t) such that d = gcd(a,b) and
// d == a*s + b*t

if (b == 0) return (a,1,0) ;

(d1, s1, t1) = ExtEuclid(b,a%b) ;
d = d1 ;
s = t1 ;
t = s1 - (a div b) * t1 ;   // note: div = integer division
return (d,s,t) ;
 }
```

7 Example

1. Choose p = 3 and q = 11
2. Compute n = p * q = 3 * 11 = 33
3. Compute $\phi(n) = (p-1) * (q-1) = 2 * 10 = 20$
4. Choose e such that $1 < e < \phi(n)$ and e and n coprime.
5. Let e = 7
6. Compute a value for d such that (d * e) % (n) = 1. One
7. solution is d = 3 [(3 * 7) % 20 = 1]
8. Public key is (e, n) =¿ (7, 33)
9. Private key is (d, n) =¿ (3, 33)

10. The encryption of m = 2 is c = 27 % 33 = 29

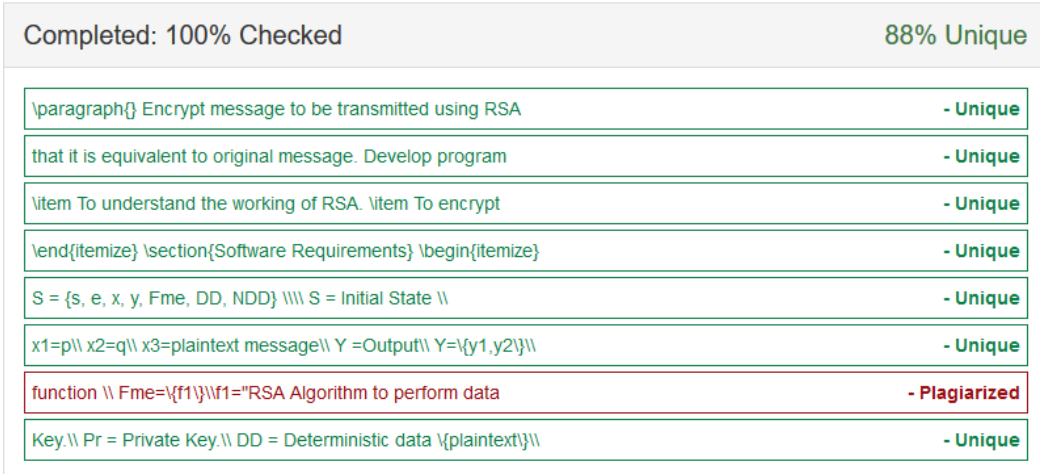
11. The decryption of c = 29 is m = 293 % 33 = 2

8 Conclusion

Thus we have studied and implemented RSA algorithm to encrypt and decrypt message.

Roll No.	Name of Student	Date of Performance	Date of Submission
302	Abhinav Bakshi	21/01/2016	04/02/2016

9 Plagarism Report



10 Output

```
-----------------------------------------------
RSA output


Enter Message :
123
P : 179
Q : 13
```

```
2327
Cipher : 301
Plaintext : 123
-----------------------------------------------
```