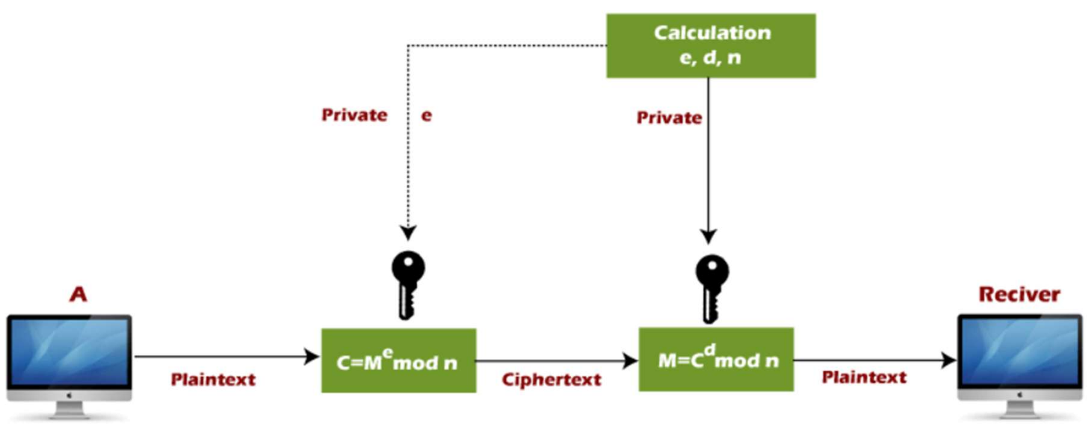
Final Year B. Tech, Sem VII 2022-23
PRN – 2020BTECS00211
Name – Aashita Narendra Gupta
Cryptography And Network Security Lab
Batch: B4
Practical No – 11

Title: Implementation of RSA Algorithm.

Theory:

RSA algorithm is a public key encryption technique and is considered as the most secure way of encryption. It was invented by Rivest, Shamir and Adleman in year 1978 and hence name RSA algorithm.



RSA

RSA algorithm uses the following procedure to generate public and private keys:

- Select two large prime numbers, p and **q**.
- Multiply these numbers to find **n = p x q,** where **n** is called the modulus for encryption and decryption.
- Choose a number **e** less than **n**, such that n is relatively prime to **(p - 1) x (q -1).** It means that **e** and **(p - 1) x (q - 1)** have no common factor except 1. Choose "e" such that 1<e < φ (n), e is prime to φ (n),
  gcd (e,d(n)) =1
- If **n = p x q,** then the public key is <e, n>. A plaintext message **m** is encrypted using public key <e, n>. To find ciphertext from the plain text following formula is used to get ciphertext C.
  $C = m^e \bmod n$
  Here**, m** must be less than **n**. A larger message (>n) is treated as a concatenation of messages, each of which is encrypted separately.
- To determine the private key, we use the following formula to calculate the d such that:
  $D_e \bmod \{(p - 1) \times (q - 1)\} = 1$
  Or
  $D_e \bmod \varphi (n) = 1$
- The private key is <d, n>. A ciphertext message **c** is decrypted using private key <d, n>. To calculate plain text **m** from the ciphertext c following formula is used to get plain text m.
  $m = c^d \bmod n$

Example:

RSA Example

1. Select primes: p=17 & q=11
2. Compute $n = pq$ =17×11=187
3. Compute ø(n)=(p–1)(q-1)=16×10=160
4. Select e : gcd(e,160)=1; choose e=7
5. Determine d: de=1 mod 160 and d < 160
   Value is d=23 since 23×7=161= 10×160+1
6. Publish public key KU={7,187}
7. Keep secret private key KR={23,17,11}

CCLAB

연세대학교 YONSEI UNIVERSITY

Code Snapshots:

```cpp
#include <bits/stdc++.h>
using namespace std;

// void file()
// {
// #ifndef ONLINE_JUDGE
//     freopen("input.txt", "r", stdin);
//     freopen("output.txt", "w", stdout);
// #endif
// }

// Function for extended Euclidean Algorithm
int ansS, ansT;
int findGcdExtended(int r1, int r2, int s1, int s2, int t1, int t2)
{
    // Base Case
    if (r2 == 0)
    {
        ansS = s1;
        ansT = t1;
        return r1;
    }

    int q = r1 / r2;
    int r = r1 % r2;

    int s = s1 - q * s2;
    int t = t1 - q * t2;

    cout << q << " " << r1 << " " << r2 << " " << r << " " << s1 << " " << s2
<< " " << s << " " << t1 << " " << t2 << " " << t << endl;

    return findGcdExtended(r2, r, s2, s, t2, t);
}


int modInverse(int A, int M)
{
    int x, y;
    int g = findGcdExtended(A, M, 1, 0, 0, 1);
    if (g != 1) {
        cout << "Inverse doesn't exist";
        return 0;
    }
    else {

        // m is added to handle negative x
```

```
        int res = (ansS % M + M) % M;
        cout << "inverse is" << res << endl;
        return res;
    }
}

long long powM(long long a, long long b, long long n)
{
    if (b == 1)
        return a % n;
    long long x = powM(a, b / 2, n);
    x = (x * x) % n;
    if (b % 2)
        x = (x * a) % n;
    return x;
}

int findGCD(int num1, int num2)
{
    if (num1 == 0)
        return num2;
    return findGCD(num2 % num1, num1);
}

// Code to demonstrate RSA algorithm
int main()
{
    //file();

    // Two random prime numbers
    long long p, q, e, msg;
    //17 31 7 2

    cout << "Please enter 2 prime number and e and Message to Encript" <<
endl;
    cin >> p >> q >> e >> msg;

    cout << "2 random prime numbers selected are " << p << " " << q << endl;

    // First part of public key:
    long long n = p * q;
    cout << "Product of two prime number n is " << n << endl;

    // Finding other part of public key.
    // e stands for encrypt

    cout << "Taken e is " << e << endl;
```

```
    long long phi = (p - 1) * (q - 1);
    cout << "phi is " << phi << endl;

    while (e < phi) {
        // e must be co-prime to phi and
        // smaller than phi.
        if (findGCD(e, phi) == 1)
            break;
        else
            e++;
    }

    cout << "Final e value is " << e << endl;


    // Private key (d stands for decrypt)

    long long d = modInverse(e, phi);
    cout << "d is " << d << endl;

    cout << "\nso now our public key is " << "<" << e << "," << n << ">" <<
endl;
    cout << "\nso now our private key is " << "<" << d << "," << n << ">" <<
endl << endl;



    // Message to be encrypted

    cout << "Message date is " << msg << endl;

    // Encryption c = (msg ^ e) % n
    long long c = powM(msg, e, n);
    cout << "Encripted Message is " << c << endl;

    // Decryption m = (c ^ d) % n
    long long m = powM(c, d, n);
    cout << "original Message is " << m << endl;

    return 0;
}
```

Output Snapshots:

```
PROBLEMS    OUTPUT    TERMINAL    GITLENS    DEBUG CONSOLE

PS C:\Users\Ashitra\OneDrive\Desktop\7th sem\Practicals\CNS\Programs> cd "c:\Users\Ashitra\OneDrive\De
sktop\7th sem\Practicals\CNS\Programs\" ; if ($?) { g++ RSAAlgo.cpp -o RSAAlgo } ; if ($?) { .\RSAAlgo
 }
Please enter 2 prime number and e and Message to Encript
5 7 3 28
2 random prime numbers selected are 5 7
Product of two prime number n is 35
Taken e is 3
phi is 24
Final e value is 5
0 5 24 5 1 0 1 0 1 0
4 24 5 4 0 1 -4 1 0 1
1 5 4 1 1 -4 5 0 1 -1
4 4 1 0 -4 5 -24 1 -1 5
inverse is5
d is 5

so now our public key is <5,35>

so now our private key is <5,35>

Message date is 28
Encripted Message is 28
original Message is 28
PS C:\Users\Ashitra\OneDrive\Desktop\7th sem\Practicals\CNS\Programs> 
```

Conclusion:

1. It is concluded that, while establishing RSA key pairs, usage keys and general-purpose keys are integrated.

2. In usage RSA keys, two key pairs are used for encryption and signatures.