Assignment no ICS 4

Roll no 41239

Title: Implementation of RSA Algorithm

Problem Definition: Implementation of RSA Algorithm

Software Requirements:

Python 3.7,Colab

Hardware Requirement:

8GB RAM, 500 GB HDD,Keyboard,Mouse

Learning Objectives:
Learn RSA Algorithm

Outcomes:
After completion of this assignment students are able to understand the How to encrypt and decrypt messages.

Theory :

1.**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 the keys can be given to anyone. The other key must be kept private. The algorithm is based on the fact that finding the factors of a large composite number is difficult: when the integers are prime numbers, the problem is called prime factorization. It is also a key pair (public and private key) generator.

2.RSA makes the public and prívate keys by multiplying two large prime numbers p and q

a.It's easy to find & multiply large prime No. ($n=pq$)

b.It is very difficult to factor the number n to find p and q

c.Finding the private key from the public key would require a factoring operation

d.The real challenge is the selection & generation of keys.

3.RSA is complex and slow, but secure 100 times slower than DES on s/w & 1000 times on h/w

4.The Rivest-Shamir-Adleman (RSA) algorithm is one of the most popular and secures public- key encryption methods. The algorithm capitalizes on the fact that there is no efficient way to factor very large (100-200 digit) numbers.

Using an encryption key (*e*,*n*), the algorithm is as follows:

1. Represent the message as an integer between 0 and (*n*-1). Large messages can be broken up into a number of blocks. Each block would then be represented by an integer in the same range.
2. Encrypt the message by raising it to the *e*th power modulo *n*. The result is a ciphertext message C.
3. To decrypt ciphertext message C, raise it to another power *d* modulo *n*

The encryption key (*e*,*n*) is made public. The decryption key (*d*,*n*) is kept private by the user.

How to Determine Appropriate Values for *e, d,* and *n*

1. Choose two very large (100+ digit) prime numbers. Denote these numbers as *p* and *q*.
2. Set *n* equal to *p* * *q*.
3. Choose any large integer, *d*, such that GCD(*d*, ((*p*-1) * (*q*-1))) = 1
4. Find *e* such that *e* * *d* = 1 (**mod** ((*p*-1) * (*q*-1)))

Rivest, Shamir, and Adleman provide efficient algorithms for each required operation[4].

How Does RSA Works?

RSA is an **asymmetric** system, which means that a key pair will be generated (we will see how soon) , a **public** key and a **private** key , obviously you keep your private key secure and pass around the public one.

The algorithm was published in the 70’s by Ron **R**ivest, Adi **S**hamir, and Leonard **A**dleman, hence RSA, and it sort of implement’s a trapdoor function such as Diffie’s one.

RSA is rather slow so it's hardly used to encrypt data, more frequently it is used to encrypt and pass around **symmetric** keys which can actually deal with encryption at a **faster** speed.

RSA Security:

- It uses prime number theory which makes it difficult to find out the key by reverse engineering.
- Mathematical Research suggests that it would take more than 70 years to find P & Q if N is a 100 digit number.

Algorithm

The RSA algorithm holds the following features −

- RSA algorithm is a popular exponentiation in a finite field over integers including prime numbers.
- The integers used by this method are sufficiently large making it difficult to solve.
- There are two sets of keys in this algorithm: private key and public key.

You will have to go through the following steps to work on RSA algorithm −

Step 1: **Generate the RSA modulus**

The initial procedure begins with selection of two prime numbers namely p and q, and then calculating their product N, as shown −

N=p*q

Here, let N be the specified large number.

Step 2: Derived Number (e)

Consider number e as a derived number which should be greater than 1 and less than (p-1) and (q-1). The primary condition will be that there should be no common factor of (p-1) and (q-1) except 1

Step 3: Public key

The specified pair of numbers **n** and **e** forms the RSA public key and it is made public.

Step 4: Private Key

Private Key **d** is calculated from the numbers p, q and e. The mathematical relationship between the numbers is as follows −

ed = 1 mod (p-1) (q-1)

The above formula is the basic formula for Extended Euclidean Algorithm, which takes p and q as the input parameters.

Encryption Formula

Consider a sender who sends the plain text message to someone whose public key is **(n,e).** To encrypt the plain text message in the given scenario, use the following syntax −

C = Pe mod n

Decryption Formula-

Plaintext = Cd mod n

Example

1. P=7, Q=17
2. 119=7*17
3. (7-1)*(17-1)= 6*16 =96 factor 2 & 3, so E=5
4. (D*5) mod (7-1)*(17-1)=1, so D=77
5. CT=10^5 mod 119 =100000 mod 119 =40
6. Send 40
7. PT=40^{77} mod 119 = 10

Conclusion

Thus we learn that to how to Encrypt and Decrypt the message by using RSA Algorithm.