

e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science

Volume:02/Issue:05/May-2020

www.irjmets.com

# RSA ALGORITHM: A THEORETICAL STUDY AND IMPLEMENTATION K. Nanda Kishore<sup>\*1</sup>, Sujan Chhetri<sup>\*2</sup>

<sup>\*1,2</sup> Student, Department of Information Science and Engineering, School of Engineering and Technology, Jain University, Bangalore, Karnataka, India

## ABSTRACT

There are many aspects in which security can be provided, even many applications too like secure payments, private communications. One such indispensable aspect is Cryptography. Though it's probably the oldest art, still the techniques used in Cryptography are a crucial means in achieving security. It not only clinches in making information restricted but also delivers various protection and security features like system security, digital signatures. So, the methods, encryption, and decryption of cryptography play a vital role in achieving the security mentioned above. The quality of the security provided will be entirely dependent on the quality of the encryption and decryption algorithms which in turn can be said to be based on the structure of mathematics and the confidentiality of key. The key can be said as the essence of encryption, as by knowing the key a person can encrypt or decrypt the information. Hence, choosing the key is the most vital process. This paper provides an overview and implementation of RSA, a public-key cryptosystem.

**KEYWORDS :** cryptography, public-key cryptography, RSA algorithm.

## I. INTRODUCTION

The main problem that cryptography tries to solve is preventing unauthorized access to information [1]. In general, cryptography works by encrypting the plaintext into a cipher text which will be turned back into plaintext by decrypting it. Thus, the methods encryption and decryption are the crucial schemes in achieving better security. Mainly the usage of an efficient encryption algorithm helps to attain better safety to the end-user and greater affiliation for the attacker.

The RSA algorithm is a public-key cryptosystem that is widely used in digital signatures despite being published in 1977. It was first presented by R. L. Rivest, A. Shamir, and L. Adleman in the paper "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems" in 1978. The essence of the RSA algorithm lies in prime factorization which is known as the unsolved problem in computer science [2] as that's what keeps RSA more secure and difficult to break. This paper implements the RSA algorithm in Java by generating very huge prime numbers.

## II. THE RSA CRYPTOSYSTEM

RSA (Rivest–Shamir–Adleman) is among the first public-key cryptosystems which are widely used for secure data transmission [3]. In this Cryptosystem, the encryption key is public and is different from the decryption key which is kept as private or secret also known as a private key. This Asymmetric Encryption system uses two distinct but related keys. The Public Key, is used for encryption and the other, the Private Key, is for decryption. The Private Key is to be kept private so that only the authenticated receiver can decrypt the message. This asymmetry is based on the practical difficulty of factoring the product of two large prime numbers. A user generates and then publish a public key on the basis of two large prime numbers, along with an additional value with the key. The prime numbers that are generated must be kept as a secret. Anyone who has the public key can encrypt the message, but only the message can only be decoded by the one with the prime numbers. The security becomes much strong as it gets harder for the users to decrypt the encrypted message without the private key, as the complexity of the factorization of large integers increases. The use of two keys makes this encryption system a very complex technique. Hence, it proves to be very beneficial in terms of data security. Till date there are not any methods to defeat the system if a larger key is used for encryption.



## e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science www.irjmets.com

- Public Key Cryptography : Public Key cryptography is a method of encrypting data with two different **(A)** keys i.e. private and public [4]. The public key is available for anyone to use. The private key is kept private or as a secret for each individual. The sender can encrypt a message using the receiver's public key, but that encrypted message can only be decrypted with the receiver's private key. This ensures that only the receiver will be able to read the message. Let us assume, User X wants to send a private message, 'm', to user Y. User X gets User Y's public key from mail. User X encrypts message 'm' using Y's public key. This produces a cipher text message 'c' which is sent over a mail. Upon receiving, User Y decrypts message 'c' using their private key. This results in the original message 'm'.
- **(B) RSA** Cryptography : RSA Cryptography is a cryptographic system that uses the RSA algorithm which is an asymmetric algorithm in which the two set of keys are used i.e. Public Key and Private Key. The Public Key is given to everyone and the Private key is kept private or secret [5].

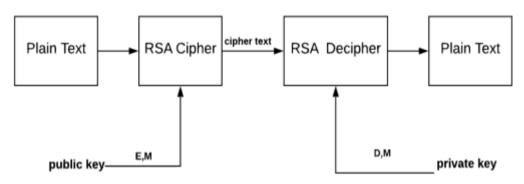


Fig-1: Working of the RSA Algorithm

#### **(C)** The Process of RSA Algorithm :

Volume:02/Issue:05/May-2020

- The RSA algorithm involves the following steps to obtain the public and private keys: Randomly generate two primes P and Q of huge length.
- Compute the value of n which is the product of P and Q and the Euler function if n i.e, n which is on the product or (P-1) and (Q-1).
- Pick an integer e which should be relatively prime with n and the gcd(e, n) should be equal to 1.
- Now the public key will be (e,n).
- Calculate the secret exponent d such that  $e^*d \mod \phi n = 1$
- The private key obtained is (d,n).
- The process of Encryption i.e, encrypting the plaintext(m) to ciphertext(c) is done by:  $c = m^e \mod n$ .
- The process of Decryption involves taking the ciphertext(c) obtained and generating the original  $plaintext(m): m = c^d \mod n$

### (D) Security and Applications :

The security of the RSA algorithm depends on the integer factorization method. Thus, by taking two prime numbers of large lengths, it takes a very long time to perform factorization thereby making it more secure. In general, prime factorization is considered to be an unsolvable problem in computer science as it takes a very long time even for the supercomputers; hence it's vital to generate P and Q values as primes and of very huge length. But ultimately decomposing the n value is the main way to attack the algorithm.

The HTTPS (Hypertext Transfer Protocol Secure) uses a secure communication protocol called TLS (Transport Layer Security) which was previously the SSL (Secure Sockets Layer) [6]. This SSL helps to encrypt the data transfer between a client and a server and by using the RSA encryption it achieves this. In general, the data transmission over the SSL can be classified into two steps:

- SSL handshake
- The actual transfer of data



## e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science www.irjmets.com

The SSL handshake is where the communication begins over the SSL. It's where the RSA algorithm is used to generate public key and establish a secure connection. For example, basically a server generates two prime numbers and multiplies them and generates a public key(e). Now any client that wants to communicate i.e., transfer the data to the server takes that public key(e), encrypts the data, and sends it. Now the server as it knows the private key it decrypts the data using it. After this initial handshake the SSL uses the AES algorithm i.e a symmetric encryption algorithm for the actual transfer of data. In general, the SSL uses 128 or 256 digits for the keys, and in order to break these keys, a very vast amount of computing power is required [7]. Thus, RSA is said to be very hard to break and again it depends mainly on the length of the prime numbers generated.

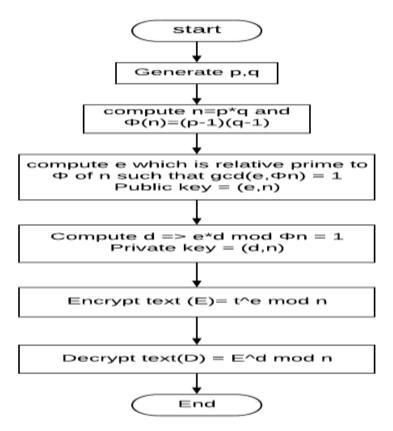


Fig-2: Flowchart Describing the Process of the RSA Algorithm

#### III. **IMPLEMENTATION**

### A. Code

/\* You can find the code in my GitHub

Volume:02/Issue:05/May-2020

Repo

itsknk/CryptographySuite \*/ public class

RSAalgo {

BigInteger p;

BigInteger q;

BigInteger n;

BigInteger z;

BigInteger e;



International Research Journal of Modernization in Engineering Technology and Science Volume:02/Issue:05/May-2020 www.irjmets.com BigInteger d; int bitlength = 1024

; Random r;

public RSAalgo

```
(){ r = new Random();
```

//generates a big prime number within the specified length

p = BigInteger . probablePrime(bitlength, r);

$$\label{eq:generalized_states} \begin{split} q &= BigInteger \ . \ probablePrime(bitlength, r); \\ System \ . \ out \ . \ println( \ "The Value of p: " + p); \\ System \ . \ out \ . \ println( \ "The Value of q: " + q); \end{split}$$

//Calculating N i.e, N=p\*q n = p . multiply(q);

//Calculating  $\phi$ N i.e,  $\phi$ N=(p-1)(q-1)

z =p . subtract(BigInteger . ONE ) . multiply(q . subtract(BigInteger . ONE )); System . out . println( "The Value of N is: " + n);

System . out . println( "The Value of  $\phi N$  is: " + z);

//generates e which will be a relative prime to  $\phi N$ e = BigInteger . probablePrime(bitlength / 2 , r);

//to check if  $1 \le \phi$ , such that  $gcd(e,\phi)=1$ 

while (z .gcd(e) .compareTo(BigInteger .ONE )>0&&e .compareTo(z) < 0 ) {

e . add(BigInteger . ONE );

}

//calculating d i.e, d should be =  $e^* dmod(\phi N)=1 d = e \cdot modInverse(z);$ 

//Public Key = (e,N)

System . out . println( "Public Key generated: ("

+ e + "," + n + ")" );

//Private Key = (d,N)

System . out . println( "Private Key generated: ("

+ d + "," + n + ")" );

}

public RSAalgo (BigInteger e , BigInteger d , BigInteger n )
{ this . e = e;

this d = d; this n = n;

}

public static void main (String [] args ) throws IOException
{ RSAalgo rsa = new RSAalgo();

DataInputStream inp =new DataInputStream(System . in); String question ;

System.out.println( "Enter the plain text:" );



e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science

Volume:02/Issue:05/May-2020 www.irjmets.com question = inp.readLine(); System.out.println( "Encrypting String: " + question); //converting the string to Bytes System.out.println( "String in Bytes: " + bytesToString(question . getBytes())); // encrypt byte [] encrypted = rsa.encrypt(question . GetBytes()); System.out.println( "Encrypted String in Bytes: " + bytesToString(encrypted)); // decrypt byte [] decrypted = rsa.decrypt(encrypted); System.out.println( "Decrypted String in Bytes: " + bytesToString(decrypted)); System.out.println( "Decrypted String: " + new String(decrypted)); } public static String bytesToString ( byte [] encrypted ) { String something = ""; for ( byte b : encrypted) { something += Byte . toString(b); } return something; } //Encryption public byte [] encrypt ( byte [] msg ){ return ( new BigInteger(msg) . modPow(e,n) . toByteArray()); //to encrypt, msge\*mod(N) } //Decryption public byte [] decrypt ( byte [] msg ) { return ( new BigInteger(msg) . modPow(d,n) . toByteArray()); //to decrypt, msg^d\*mod(N)

}}

### **B.** Output

sansept01g0011:/mt/09291196576141/j0turf5\_j2v4 H2A10 The volum of p: 13144894715648460007000113355157626340804856723787528935283560144305769284412129621738169066139170704564388790804001535515762634080485676 9822559322387370382002951252095386599583613362487945816878979285567237843286628553336010126587634836940954928343933808997988569909276815834784904660 95997118097085212342023 The volum of q: p: 1420745043145787263456671638244216414865644439563853279900124591466658458743355013555142002841208535571357428220339567240114844253570628 492866958524105872095166434090629864017487178673489332361374480198298295966712243485700458743345504179013305949483927209509589931928231868969838438655 3024590508240441127792983

### Fig.3 : Huge Prime Numbers are Generated for p and q

The Value of N (s) 1906322159555000000014770750409315020493404945424453576533551099740550004450639624154959049044745753576525005104072571092 23396274540102274025172055277158046006001497105562260001607269715542534053155124422960763639030110404743154757544431964771359764423640791353828282750011400142204 2009050510903104746500567455915682150501505225000100090133402074537045554750000547506736574000664506490672557214441579777777547003762001660404413223444577556014607255724441579777777547003762000004044132234445077357644457755505000040041328304880074442548254103042330937031751 0274147660159305686457267125880906603477075640933502040930843654824453573080759411605104012709720600040441322344880074402548254103042330937031751 0274147660159305686864572671258809066034770756409335020409308476548243355730807594150510401270972060004044132334488007440254825410304233093703370 0274147660159305686864572671258809066034770756409335020409308476548244355755931352309974055000645063902415487553570829809144170403430927189 1533392745402582772555799528715656000054800695674095503066580726577554273546951045755423349206763639302116467735976144254825400975558282715857829200006450639020116467735976144254825400975040933002040913082769240730927189 15333927454028277825579595287156600005480089667849563300665806726677152542334959164373352540290280206425103020126469773597515535828273580470377 2220381601900283624296056276095161168566291597197074146494115376014475549044715576039820215401646773597614425649773575537723 2209381601900283624292602667466773597515423340971877177377232607150576159542470064231031000134002352449365420056574667735957515737737 22093816019002836242956026576691647725959469351027102717237235601585761585427100642310310023124306205665746677175355403946096274490 10976114411252440266726242150016557221590161557212927364667736715425471553760348475647981447564798138271835364098674490 150717441641067262504710533378007586180



e-ISSN: 2582-5208

International Research Journal of Modernization in Engineering Technology and Science Volume:02/Issue:05/May-2020 www.irjmets.com

Fig.4 : n and on are Calculated



Fig.5 : The Public(e,n) and Private(d,n) Keys Obtained



Fig.6 : Encryption and Decryption of the text "one hundred years of solitude"

### C. Explanation

The implementation of RSA is somewhat complex as the whole concept of making RSA secure lies on the factorization of two large integers [8], the primes generated should be very huge in length. Thus, instead of using integer variables, we use Big Integer for p and q to generate random huge prime numbers. As mentioned above in the process we find the values of n,  $\phi$ n first. Then we choose an integer e which will be relatively prime to  $\phi$ n and gcd(e, $\phi$ n) =1. The public

key(e,n) and private key(d,n) will be generated as per the process mentioned above. In Java, we convert the data entered i.e., the plaintext into bytes and we use these bytes to encrypt the plain text, similarly, we use the encrypted text and decrypt it and convert the decrypted bytes into the string which will return the entered plain text.

## **IV. CONCLUSION**

This paper explored the core concepts of cryptography i.e., encryption and decryption, then public-key cryptography mainly the RSA cryptosystem. The encryption and decryption algorithms play a vital role in achieving security and the quality of those algorithms shows the quality of the security provided. In general, in a public-key cryptosystem, the encryption key will be public and will be different from the secret key which will be used for decryption. The RSA is one such public-key cryptosystem or the asymmetric cryptosystem which is widely used. The core idea of attaining security in RSA is by factorization of integers and hence in RSA, we use two very large prime numbers as their factorization is very difficult and considered as an unsolvable problem in computer science. This paper provided the implementation of the RSA algorithm in Java and this code can be utilized to implement the RSA in the required applications. The data generated to or from the server over the HTTPS is encrypted and the SSL certificates of the websites i.e. where HTTPS is applied depends on the RSA. But the future scope involves the rise in the usage of other cryptosystems like Diffie-Hellmann and Elliptic curve instead of RSA, as the recent research has exhibited that the RSA keys of length 2048 bits can be effectively demoted by padding oracle attack and other [9].

## V. REFERENCES

- [1] W. Diffie and M. E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory, Nov. 1976,22: 644-654.
- [2] R.L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM, Feb. 1978, 21(2): 120-126.
- [3] RSA (cryptosystem)-https://en.wikipedia.org/wiki/RSA\_(cryptosystem).



### International Research Journal of Modernization in Engineering Technology and Science Volume:02/Issue:05/May-2020 www.irjmets.com

- [4] Public Key Cryptography-https://www.britannica.com/topic/public-key-cryptography RSA [5] cryptography-https://simple.wikipedia.org/wiki/RSA algorithm.
- [5] HTTPS-https://en.wikipedia.org/wiki/HTTPS.
- [6] How does HTTPS provide security? - https://stackoverflow.com/questions/3968095/how-does-HYPERLINK "https://stackoverflow.com/questions/3968095/how-does-https-provide-security" HYPERLINK "https://stackoverflow.com/questions/3968095/how-does-https-provide-security" HYPERLINK "https://stackoverflow.com/questions/3968095/how-does-https-provide-security" httpsprovide-security.
- Why are "large prime numbers" used in RSA/encryption? https://stackoverflow.com/questions/ [7] HYPERLINK "https://stackoverflow.com/questions/11832022/why-are-large-prime-numbers-used-inrsa-encryption" HYPERLINK "https://stackoverflow.com/questions/11832022/why-are-large-primenumbers-used-in-rsa-encryption" HYPERLINK "https://stackoverflow.com/questions/11832022/whyare-large-prime-numbers-used-in-rsa-encryption" 11832022/why-are-large-prime-numbers-used-in-rsaencryption.
- [8] E. Ronen, et al., "The 9 Lives of Bleichenbacher's CAT: New Cache ATtacks on TLS Implementations," in 2019 2019 IEEE Symposium on Security and Privacy (SP), San Fransisco, CA, US, 2019 pp. 966-983.doi: 10.1109/SP.2019.00062