**Cryptography**

**BITS F463**

**Assignment**

**RSA Algorithm**

**Meriam Joseph (2018A7PS0291U)**

**Parth Ganeriwala (2018A7PS131U)**

**ABSTRACT**

Rapid growth of e-communications has necessitated the need for securing transmitted data. Cryptographic algorithms help with this by performing tasks like authentication, data encryption and digital signatures on transmitted data. This can be achieved by various methods, including, using algorithms that require a public and private key such as asymmetric algorithms to encrypt and decrypt data. RSA algorithm is the most widely popular of this class of cryptographic algorithms. However, RSA algorithm is, is relatively slow in comparison to symmetric algorithms due to its calculations with large numbers, which takes place in the decryption step of the algorithm. Hence, in this paper, we implement RSA Algorithm and compare how CRT optimization can make the decryption step run faster in comparison to the standard RSA Decryption.

**INTRODUCTION**

The growth of the Internet, and consequently, e-commerce has brought forth the importance of privacy and security in e-communications. Large volumes of data that is transmitted and stored everyday requires to be protected and secured. Encryption of data, hence, is a standard method for achieving this. Here, an individual, who wants to send a private message encrypts or enciphers a message before transmitting it, and only the intended recipient has the knowledge of how to decrypt or decipher it. [1]

There are various methods to encrypt data. Asymmetric algorithms are one of the classes of cryptographic algorithms that require two different mathematically linked keys, - a private key and a public key. The public key is one that is shared publicly while the private is kept a secret and not shared with anyone. Hence, why the asymmetric method of encryption is also called “public key cryptography”. These cryptographic keys, in addition to algorithms are what makes encryption possible. By allowing a user to receive information encrypted by the public using the public key decrypting it by using the secret private key. An advantage of asymmetric encryptions is that it provides security against man in the middle (MiTM) attacks. Hence, these encryptions are especially useful in web /email servers that connect to thousands of people. It also proves authentication by making sure that the data is only seen and decrypted by the recipient that is supposed to receive it.

The RSA Algorithm in an asymmetric algorithm which was invented in 1978. It was named after its creators Ron Rivest, Adi Shamir and Leonard Adleman. It is one of the most widely popular asymmetric encryption algorithms. The idea of RSA is formulated from the fact that it is difficult to factorize a large integer. It relies on prime factorization that involves two huge prime numbers and multiplication of these two numbers to create another giant number. The aim here, is to determine the original prime numbers from this giant multiplied number.

RSA provides the advantage of scalability, and hence comes in various lengths like 4096-bit, 2048-bit, 1024-bit and 768-bit. Even if the lower key lengths are successfully brute forced, one can ensure encryption of higher key lengths because the difficulty of brute forcing the key increases with an expanding key length [2]. Since RSA is based on a simple mathematical approach, its implementation in public key infrastructure (PKI) is also to the point. The adaptability with PKI and its security makes its usage extensive in applications including SSL/TLS certificates, crypto currencies, and email encryption.

However, RSA is a relatively slow algorithm and hence is not commonly used to directly encrypt user data or large data. More often, RSA is used to transmit shared keys for [symmetric key](https://en.wikipedia.org/wiki/Symmetric-key_algorithm) cryptography, which are then used for bulk encryption-decryption [3]. The speed constraint on the algorithm is traced back to the decryption step where the decryption value is very large and hence, requires a lot of work to compute.

The basic RSA is vulnerable to set of indirect attacks like chosen plaintext, timing, known plaintext, common modulus, and frequency of blocks (FOB) attacks. Breaking the RSA [encryption](https://en.wikipedia.org/wiki/Encryption) is called the [RSA problem](https://en.wikipedia.org/wiki/RSA_problem). An open question that is found here is if the RSA problem is as difficult as the factoring problem. However, so far there has been no published methods on how to defeat the system if a large enough key is used [4].

Pseudocode for RSA Algorithm

Input: p1 and p2

int n=p1\*p2

int phi=(p1-1) \* (p2-1)

int e= findCoprime(phi)

d= (1 mod(phi))/e

public\_key = (e,n)

private\_key =(d,n)

C=(Pe)%n

P=(Cd)%n

* Where, p1 and p2 are two large prime numbers.
* phi is the value of the totient function ϕ(n)=(p1−1) x (p2−1).
* e is the coprime of phi and 1 < e < ϕ(n)
* d is calculated using the extended Euclidian algorithm
* C is the ciphertext and P is the plaintext.

Since the process for both the encryption and decryption steps of RSA becomes slow whenever it encounters a large bit length, many techniques were proposed to speed up these two, especially the decryption step, where the private key exponents are always larger than public key exponents.

The Chinese Remainder Theorem is one of the many methods which can be used to decrease the computation time as required by the RSA Algorithm. This theorem divides the private key into small sub exponents and hence allows for modular exponentiation to be solved quickly. Finally, the results of each sub exponent are merged and used to find the result. The CRT model was discovered by Sun-Tsu, a Chinese mathematician. This method can help fin d x from x≡a1 mod m1, x≡a2 mod m2, • • •, x≡ak mod mk by using the following equation:

Diagram

Description automatically generated with medium confidence 

**Pseudocode for applying CRT with RSA**

qInv = q-1 mod p

m1= c (dP) d

m2= c (dQ) d

h = ( qInv \* ( m1-m2 ) ) % p

m= m2 + h \* q

where, q and p are distinct prime numbers and m1 , m2 are the modular exponentiations and h is the multiplicative inverse.

**LITERATURE SURVEY**

This section presents previous studies relating to the various types of cryptographic techniques which are one of the principals means to protect information security used, modified to fit the best-case scenario. Previous studies relating to the usage of DES, AES and RSA Algorithms has been thoroughly researched upon which has helped in the formulation of our problem statement relating to the RSA Algorithm.

Initially we investigate the comparison between the asymmetric and symmetric algorithms and from the study by Aamer Nadeem and Dr. M Younus Javed [5], where performance evaluation was done by the implementation of symmetric algorithms such as DES, 3DES, AES and Blowfish was achieved with Java. They concluded that as the block size increases, so does the speed time as larger block sizes need less execution time to encrypt the data. They also concluded that Blowfish was the fastest symmetric algorithm and 3DES was the slowest. According to a survey conducted in [6], we find that the efficiency of the different algorithms was affected by the difference parameter.

In the current day, where there is an increasing demand on clod application, it is necessary to provide efficiency along with robust and high-security algorithms that are suitable with the large scale of data in the cloud. From a comparative study [7] on the various encryption techniques with respect to symmetric key and asymmetric key algorithms, through intense analysis it was concluded that symmetric key algorithms were efficient in terms of speed and power consumption whereas asymmetric key algorithms in terms of tunability. In asymmetric key encryption RSA algorithm was found to be the most efficient with regard to speed and security. In another paper [8] the analysis based on experiments on symmetric key encryption algorithm based ESEBM and asymmetric key encryption algorithm such as RSA was done, and we find that the RSA based decryption method is very slow because of repeated encryptions by the sender and needs more time than other algorithms.

In a review paper on AES, DES, RSA Encryption Standards [9] we see that in a hyper cryptosystem the symmetric cryptographic algorithms such as AES, DES will provide actual bulk data encryptions whereas the asymmetric cryptographic algorithms such RSA will facilitate the secure key exchange of the symmetric algorithms along with the integrity check of the entire process. With respect to the usage of RSA Algorithm in File Encryption and Decryption System [10] we see that for small files it is easier as the text can be more convenient to communicate and manage.

In an implementation of the RSA's algorithm [11], we see that the security depends on the difficulty of integer factorization, whether it is equivalent to integer factorization has not been fully proved in theory because there is no evidence providing that to cracking RSA and would require for making large numbers factorization.

Regarding the time consumption, in a study done [12] it was found that using the experimental text files, AES algorithm consumed the least encryption and RSA consumed the longest encryption time. While addressing the concerns of asymmetric encryption algorithms taking too much time, we see that with the absence of any cardinal implementation errors, RSA-based bulk encryption algorithm provides sufficient security to ransomware developers and in the analysis [13], we see that the time concerns have now become manageable with modern computing power in end user systems. Another study [14] derived well-defined formulas that gave approximations of the mutual information I(K;Z) between the running time Z and the decryption key K that is the basis in RSA decryption algorithms. The study allowed us to get an accurate security comparison of different algorithms and helped understanding the essential risk of passive timing attacks.

A study developed a more secure and efficient algorithmic process for RSA decryption by using the Chinese remainder theorem (CRT) [15] to design the algorithmic workflow and achieved an outstanding efficiency of 99.17% where the client was able to detect any misbehavior of the cloud server. In another paper [16] an optimized parallel implementation of RSA signatures was presenting using some of the more efficient and effective arithmetic algorithms for both CPU and GPU high-end architectures. In another ground-breaking technique [17], the RSA algorithm was used with an optimization technique to encrypt a secret image. In the encryption process, the Particle Swarm optimization (PSO) algorithm was used to obtain the optimal public key in the RSA algorithm. Work was also done in the field of cloud computing where RSA algorithm was used to encrypt data to enhance the data security in the cloud [18].

A proposal for the concept of digital signature with RSA algorithm, to encrypting the data while it is being transferred over the network was presented. There has been research done in improving the RSA Algorithm [19] to make it more efficient and it was done by generalizing the RSA encryption scheme to be implemented in the general linear group on the ring of integer mod n. It was claimed to be efficient, scalable and a flexible workflow.

Research has also been done in the field of enhancing the factorization of the RSA Public Key Encryption [20] by finding the prime factor of RSA Modulo N, which would allow us to generate the private key and decrypt the secret message.

Hybrid encryption algorithms have also been developed with more emphasis on RSA and Diffie-Hellman [21] which brought us to a conclusion that this could be a great use for secure communications. We know that the security of the RSA cryptosystem is based on the two mathematical problems which are: the problem of factoring large numbers knowing mathematical attack and the problem of trying all possible private keys known as brute force attack. A modified RSA Encryption Algorithm (MREA) [22] was proposed which would improve the security and this can be confirmed as it is based on the factoring problem. In another paper [23], a hypothesis was presented which concluded that if the encryption procedure can be broken in a certain number of operations, then the integer R which is used as a modulus can be factored in only a few more operations making the system unbreakable as opposed to the current version.

In [24], CRT algorithm is used to shorten the bit size of the decryption value of RSA, and hence speed up the decryption time. The paper concluded that CRT could speed up decryption process by up to three times. In another article [25], a modifies RSA Algorithm was proposed to have two public key and a Chinese Remainder Theorem (CRT). The modified RSA-CRT algorithm performed two times faster than the standard RSA Algorithm. [26] demonstrates how an embedded device can use key generation and RSA-CRT methods to implement RSA efficiently without storing the private key itself. In [27], a public key cryptography has been enhances using RSA, RSA-CRT and N-Prime RSA with multiple keys. This RSA Algorithm makes use of four keys quicker and more efficiently, hence achieving high computational speed and decreasing the mathematical complexity of the steps.

**PROBLEM FORMULATION AND POSSIBLE SOLUTIONS**

RSA algorithm runs considerably slowly due to its calculation with large numbers. This is usually in cases where large data needs to be encrypted by the same computer and, in the decryption step where d is used in the exponent of the calculation. Some of the possible solutions to this re as follows: -

1. This can be sped up by remembering both p1 and p2, but it would still be slow in comparison to symmetric encryption algorithms.
2. Another solution to speeding up the algorithm is to use small encryption components, i.e., by assigning a small value to e.
3. CRT or Chinese Reminder Theorem can also be used to speed up the decryption process and hence speeding up RSA algorithm.

We have decided to implement the Chinese Remainder Theorem (CRT) to speed up the decryption process of the RSA Algorithm and then compare the time taken to run the modified version with the standard version of RSA. This is because, CRT is a technique to reduce modular calculations with large moduli to similar calculations for each factors of the modulus. This means, that if we know the factorization of a modulus N, then we can split the message M into two- p and q. Then, we compute the modulo of each separately and then recombine them.

**IMPLEMENTATION AND RESULTS**

We have implemented the problem statement with a short Python script of the RSA cryptosystem which uses numbers on 1024 bits and decryption done both the classical way and with CRT. This is how we will be able to find out the difference in the time taken and thus the optimization proposed by our method. We see that the CRT method of decryption is about four times faster even though there are more steps in the procedure the modular exponentiation to be carried out uses much shorter exponents and so it is less expensive in the end. To have a visualization of our proposed method, we have Figure 2 to give us the results. We try to encrypt the plaintext “pythonisthebest" and then decrypt the cipher text using normal RSA decryption and CRT decryption and compute the time taken for each. We can see the following Figure 1 which contains the CRT function for our implementation.

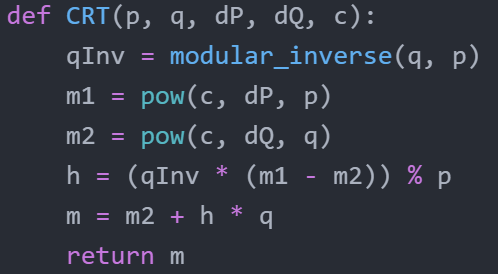


Figure : CRT Algorithm explanation

We start with the calculation of the multiplicative inverses of the prime numbers p and q. We then compute the CRT representation of the message (m1, m2) using arithmetic modular version of p and q and the shorted, pre-computed CRT exponents given as dP and dQ. We then use Garner’s formula given by

*a = x1 + x2p1 + x3p1p2 + …+ xkp1 … pk−1*

which is called the mixed radix representation of a. Garner's algorithm computes the coefficients x1, …, xk. We use the Garner’s formula with the pre-computed h which has qInv to recover m.

From the results we can see, that the CRT algorithms has decryption speeds of upto 66% faster than the standard RSA Algorithms thus proving our proposed solution to be more efficient in decryption RSA cipher text.

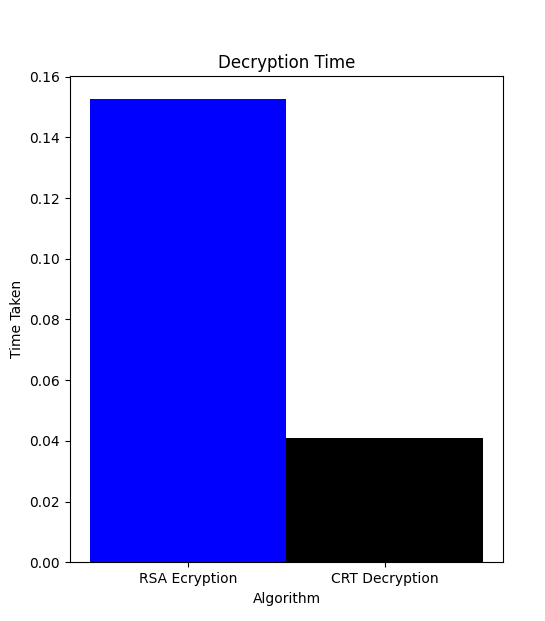


Figure : Results of our Implementation

**CONCLUSION**

The widely popular asymmetric algorithm, RSA Algorithm is commonly used to encrypt and decrypt small amounts of data and used in digital signatures. Since RSA is a slow algorithm, it is mostly used to transmit shared keys for symmetric key cryptography. Since it is the decryption process of RSA that constraints its efficiency, RSA can be made relatively faster by using CRT optimization in the RSA Decryption process. This is done by using smaller e values and remembering both the initial prime numbers generated. This method has been implemented and tested against the basic RSA decryption to compare performance of the basic and variant algorithms. As per the results obtained, optimization using the Chinese Remainder Theorem (CRT) for decryption speeds up the standard RSA Algorithm by 66.258%

# References

|  |  |
| --- | --- |
| [1] | V. T. University, "Cryptography: RSA Algorithm," 2019. [Online]. Available: https://courses.cs.vt.edu/~cs5204/fall00/protection/rsa.html. [Accessed 23 03 2021]. |
| [2] | Y. L. H. P. L. H. Qing Liu, "Two efficient variants of the RSA cryptosystem," IEEE, Qinhuangdao, 2010. |
| [3] | S. H. Emilia Rutkowski, "Cryptanalysis of RSA: Integer Prime Factorization Using Genetic Algorithms," IEEE, Glasgow, 2020. |
| [4] | A. A. Abdallah Karakra, "A-RSA: Augmented RSA," IEEE, London, 2016. |
| [5] | M. Y. J. Aamer Nadeem, "A Performance Comparison of Data Encryption Algorithms," IEEE, Karachi, 2005. |
| [6] | S. A. E. Q. W. M. Y. K. Muneer Bani Yassein, "Comprehensive Study of Symmetric Key and Asymmetric Key Encryption Algorithms," IEEE, Antalya, 2017. |
| [7] | ,. S. S. Nivedita Bisht, "A Comparative Study of Some Symmetric and Asymmetric Key Cryptography Algorithms," IJIRSET, Pithoragarh, 2015. |
| [8] | K. M. R. A. H. R. S. T. Sahan Ahmad, "A Comparison between Symmetric and Asymmetric Key Encryption Algorithm based Decryption Mixnets," IEEE, Fukui, 2015. |
| [9] | B. K. Aljaafari Hamza, "A Review Paper on DES, AES, RSA Encryption Standards," IEEE, Muscat, 2020. |
| [10] | G. L. Suli Wang, "File encryption and decryption system based on RSA algorithm," IEEE, Jingdezhen, 2011. |
| [11] | X. T. Xin Zhou, "Research and Implementation of RSA Algorithm for Encryption and Decryption," IEEE, Harbin, 2011. |
| [12] | A. S. Dr. Prerna Mahajan, "A Study of Encryption Algorithms AES, DES and RSA for Security," Global Journals Inc. (USA), Chennai, 2013. |
| [13] | C. C. D. T. D. A. I. M. J. S. R. E. Pranshu Bajpai, "The Suitability of RSA for Bulk Data Encryption," IEEE, New York, 2020. |
| [14] | Y. K. Tomonori Hirata, "Information leakage through passive timing attacks on RSA decryption system," IEICE, Nagoya City, 2020. |
| [15] | J. Y. C. T. L. T. J. L. L. G. H. W. Hanlin Zhang, "Efficient and Secure Outsourcing Scheme for RSA Decryption in Internet of Things," IEEE, Qingdao, 2020. |
| [16] | L. R.-Z. N. C.-C. F. R.-H. EDUARDO OCHOA-JIMÉNEZ, "Implementation of RSA Signatures on GPU and CPU Architectures," IEEE, Mexico, 2020. |
| [17] | S. K, "An Optimal RSA Encryption Algorithm for Secret Images," International Journal of Pure and Applied Mathematics, Tamil Nadu, 2018. |
| [18] | K. L. M. M. Uma Somani, "Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing," IEEE, 2010. |
| [19] | M. A. A.-F. M. A.-F. H. S. J. Sattar J Aboud, "An Efficient RSA Public Key Encryption Scheme," IEEE, Chennai, 2008. |
| [20] | A. G. P. G. S. B. B R Ambedkar, "An Efficient Method to Factorize the RSA Public Key Encryption," IEEE, Bareilly, 2011. |
| [21] | J. S. Shilpi Gupta, "A Hybrid Encryption Algorithm based on RSA and Diffie-Hellman," IEEE, India, 2012. |
| [22] | A. K. G. P. S. Ravi Shankar Dhakar, "Modified RSA Encryption Algorithm (MREA)," IEEE, Kota, 2012. |
| [23] | H. C. Williams, "A Modification of the RSA Public-Key Encryption Procedure," IEEE, 1980. |
| [24] | A. A. R. &. A. I. Hidayat, "Implementation of RSA and RSA-CRT Algorithms for Comparison of Encryption and Decryption Time in Android-based Instant Message Applications," *Journal of Advances in Information Systems and Technology,* vol. 2, pp. 1-10, 2020. |
| [25] | H. M. H. A. R. M. H. Rasha S.A, "Modified RSA Algorithm Using Two Public Key and Chinese Remainder Theorem," *I.J of Electronics and INformation Engineering,* vol. 10, no. 1, pp. 51-64, 2019. |
| [26] | M. T. a. Q. X. Mike Hamburg, "Improvements to RSA key generation and CRT," Rambus, Inc, 2020. |
| [27] | K. P. a. P. Sriramya, "Enhancing Public Key Cryptography using RSA, RSA-CRT and N-Prime RSA with Multiple Keys," in *IEEE, Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, 2021. |