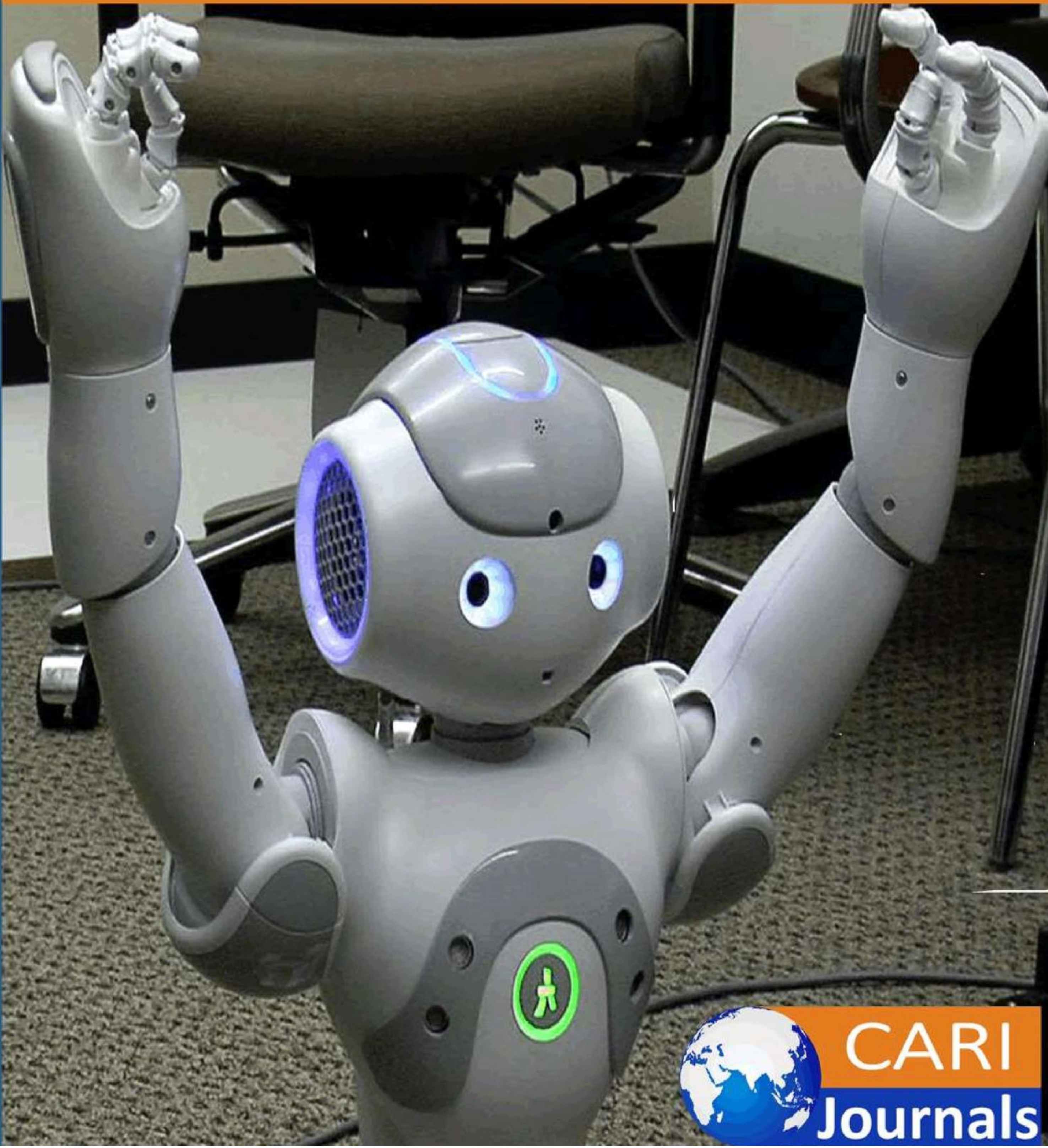
International Journal of

Computing and Engineering

(IJCE)

A Comparative Study of RSA and ELGAMAL Cryptosystems



CARI Journals

International Journal of Computing and Engineering
ISSN 2958-7425 (online)
Vol. 4, Issue No. 1, pp 33 - 41, 2023
www.carijournals.org

A Comparative Study of RSA and ELGAMAL Cryptosystems

[1]*Paul K. Arhin Jnr

Assistant Lecturer: Dept. of Computer Science and I.T

University of Cape Coast, Cape Coast, Ghana

parhin@ucc.edu.gh

[2]Prof. Michael Asante

Associate Professor: Dept. of Computer Science

Kwame Nkrumah University of Science and Technology Kumasi, Ghana

masante.sci@knust.edu.gh

[3]Linda Otoo

Assistant Lecturer: Dept. of Computer Science and I. T

University of Cape Coast, Cape Coast, Ghana

linda.otoo@ucc.edu.gh

Accepted: 12th May 2023 Received in Revised Form: 23rd May 2023 Published: 29th May 2023

Abstract

Purpose: In this research, a comparative analysis of the RSA and ELGAMAL algorithms is presented using their encryption and decryption speed with different key sizes. ElGamal and RSA are two popular public-key cryptographic algorithms that depend on the mathematical difficulty of computing discrete logarithms in finite fields and factoring large integers, respectively.

Methodology: In this study, the speed at which messages of various sizes can be encrypted and decrypted using various key sizes is used to gauge how well these algorithms work.

Findings: RSA usually outperforms ElGamal in terms of encryption and decryption speed, according to the results. However, as the key sizes grow, the performance gap between the two algorithms narrows.

Unique Contribution to Theory, Policy and Practice: These findings can aid in selecting the appropriate algorithm and key size for different applications that require secure communication and data encryption.

Keywords: *RSA, ELGAMAL, Cryptography, algorithms, Asymmetric*

1.0 INTRODUCTION

Asymmetric encryption algorithms have grown in significance in recent years for safe data protection and communication in a variety of industries, including e-commerce, online banking, and government communications [1]. Due to its capacity to offer secure key exchange and confidentiality, asymmetric encryption is frequently used in these areas [1]. Public-key encryption, also referred to as asymmetric encryption, is a type of cryptography that encrypts and decrypts data using two distinct but mathematically linked keys [2]. Asymmetric encryption uses a public key for encryption and a private key for decryption in contrast to symmetric encryption, which uses a single key for both functions [3]. In order to make it harder for unauthorized users to access sensitive data, this method adds another layer of protection [4]. There are numerous asymmetric encryption algorithms, each with advantages and disadvantages but the commonest in use are RSA and ELGAMAL [5][6]. RSA and ElGamal are the two most popular asymmetric encryption algorithms that are used in a broad range of applications, including online banking, e-commerce, and government communications. Both algorithms are built on various mathematical concepts. ElGamal is based on the problem of computing discrete logarithms, whereas RSA is based on the difficulty of factoring large integers [5][6].

The speed of cryptographic algorithms is one of the primary variables that influences how well they can be implemented in practice [3]. For real-time contact and massive data storage in particular, the speed of encryption and decryption is essential [7]. To ascertain which algorithm is most appropriate for particular use cases, it is crucial to compare the time complexity of the two most popularly used asymmetric algorithms; RSA and ElGamal algorithms.

In this paper, we begin by giving a general summary of the RSA and ElGamal algorithms, including their structure, level of security, and underlying mathematical ideas. Then, by examining the findings of earlier research, we compare and contrast how well these algorithms work in terms of speed. We also analyse the RSA and ElGamal algorithms' effectiveness in various scenarios and pinpoint the variables that influence their speed.

Researchers and practitioners who want to choose and implement the RSA and ElGamal algorithms in their systems will find the findings of this paper helpful in understanding the speed performance of these algorithms.

It is critical to evaluate the performance of the RSA and ElGamal algorithms in order to choose the one that is most appropriate for a given use case. We will compare the speed of the RSA and ElGamal algorithms in terms of key generation, encryption, and decryption in this article and discuss the factors that influence their performance.

1.1 Objectives of the Study

The following are some of the goals of this review article comparing RSA and ElGamal algorithms.

- To give a thorough review of the RSA and ElGamal algorithms, covering all of their mathematical and design aspects.

- To evaluate and compare the performance of the RSA and ElGamal algorithms in terms of key generation, encryption and decryption, for various key sizes and message sizes.
- To offer suggestions for choosing the best algorithm based on the unique needs of a given application.
- To compare the speed performance of the RSA and ElGamal algorithms and to determine the advantages and disadvantages of each method.
- To offer suggestions for RSA and ElGamal algorithm implementation based on their relative strengths and flaws in terms of speed.

  By achieving these goals, this paper aims to make a contribution to the field of cryptography by presenting a better understanding of the speed of RSA and ElGamal algorithms and their applications, as well as by assisting researchers and practitioners in making decisions regarding the choice and implementation of these algorithms in their systems.

1.2 Hypothesis

In the hypothesis, we suggest that, the speed of the RSA and ElGamal algorithms will vary depending on a number of variables, including key size, message size, and hardware platforms. With regard to difference in key and message sizes, we anticipate that RSA will outperform ElGamal because of its mathematical processes that are less complex.

Overall, by combining and contrasting the findings of earlier studies, this paper seeks to test these hypotheses and provide a thorough analysis of the speed performance of the RSA and ElGamal algorithms. It also seeks to pinpoint the metrics that affect these algorithms' performance.

2.0 METHODOLOGY

In this paper, the two algorithms compared were RSA and ELGAMAL. These algorithms were selected because they are widely used for public key encryption and they have different key generation, encryption and decryption techniques.

The algorithms are implemented using the python programming language. The implementation will be based on the industry standard algorithms for RSA and ELGAMAL encryption and decryption. Encryption and decryption times will be used to gauge how well the RSA and ElGamal algorithms work.

First, the public and private keys for both RSA and ElGamal algorithms with key sizes 1024 bits and 2048 bits are generated. Using the generated keys for both algorithms, we then encrypt and decrypt messages of different sizes according to the size of the key. Each experiment was implemented eight (8) times to guarantee statistical significance using the python programming language.

We calculated the average execution time for encryption and decryption processes for every key size and compare the performance of the two algorithms.

To verify our findings, we compared them with existing studies that have compared the performance of RSA and ElGamal algorithms in terms of speed.

To facilitate comparison between RSA and ElGamal algorithms, the findings of this study are displayed in tables and graphs. Based on the analysis and discussion of the findings and the performance of the algorithms in terms of speed, suggestions will be made for the selection and application of these techniques in various situations.

2.1 Experimental Setup

The tests will be performed on a computer that meets the requirements listed below:

Intel Core i7-9750H CPU, 16 GB RAM, with 64-bit Windows 10 operating system. The Python version used was 3.8.5.

Overall, this technique will offer a methodical way to compare the speed of the RSA and ElGamal algorithms and will offer insightful information about the advantages and disadvantages of each algorithm under various conditions.

3.0 REVIEW OF LITERATURE

The two most widely used public key encryption algorithms used to secure data in various applications are the RSA and ELGAMAL. Both algorithms have different approaches to key generation and encryption, which can impact their performance in terms of speed. This literature review provides an overview of existing studies that have compared the performance of RSA and ElGamal algorithms in terms of speed.

Alizadeh et al. (2017) conducted research comparing the effectiveness of the RSA and ElGamal algorithms for various key and message sizes. The research found that RSA encryption and decryption times increased exponentially with key size, while ElGamal encryption and decryption times increased linearly with key size. The research also discovered that ElGamal performed better for larger message sizes than RSA did for smaller message sizes [8].

O'Reilly et al. (2019) compared the effectiveness of the RSA and ElGamal algorithms on various hardware platforms in separate research. According to the study, ElGamal outperformed RSA on GPUs while RSA did better on CPUs. ElGamal was found to be more effective than RSA for large key sizes and large message sizes, according to the research [9].

The effectiveness of the RSA and ElGamal algorithms was compared in research by Liu et al. (2020) using software and hardware implementations. The research discovered that software versions of the RSA and ElGamal algorithms were slower. ElGamal outperformed RSA for large key sizes and large message sizes, according to the research [10].

Overall, these studies indicate that a number of variables, including key size, message size, hardware platform, and software implementation, affect how quickly the RSA and ElGamal algorithms perform. Therefore, careful consideration of these variables is necessary for a thorough comparison of these algorithms' speeds. By comparing the speed of the RSA and ElGamal algorithms using a standardized approach and investigating the factors that affect their performance, this paper seeks to add to this body of literature.

3.1 ELGAMAL Methodology

ElGamal is a public-key cryptographic algorithm that is used for secure communication and data encryption. The algorithm was first proposed by Taher ElGamal in 1985 and is based on the difficulty of computing discrete logarithms in finite fields [6].

3.1.1 Key generation:

a. Choose a large prime number p and a primitive root g of p.

b. Choose a secret integer a, such that 1 <= a <= p-2.

c. Calculate A = g^a mod p.

d. The public key is (p, g, A) and the private key is a.

In this stage, p is a large prime number, g is a primitive root of p, and a is a randomly chosen secret integer. The public key is (p, g, A) and the private key is a

3.1.2 Encryption:

a. Choose a random integer k, such that 1 <= k <= p-2.

b. Calculate the shared secret S = B ^ k mod p, where B is the public key of the recipient.

c. Convert the message M into a number m.

d. Calculate the ciphertext as (C1, C2) = (g ^ k mod p, m*S mod p).

In this stage, k is a randomly chosen integer, and B is the public key of the recipient. The shared secret S is calculated as B ^ k mod p. The message M is then converted into a number m, and the ciphertext is calculated as (C1, C2) = (g ^ k mod p, m*S mod p).

3.1.3 Decryption:

a. Calculate S = C1^ a mod p.

b. Calculate the plaintext as m = C2 * S ^-1 mod p.

In this stage, the shared secret S is calculated as C1^a mod p, and the plaintext is calculated as m = C2 * S^-1 mod p.

In conclusion, the ElGamal encryption algorithm uses a public key (p, g, A) and a private key (a) that are used to encode and decrypt messages, respectively. A common secret S is calculated, an integer k is chosen at random, and the message is transformed into a number m as part of the encryption process. Then, the ciphertext is computed using the formula (C1, C2) = (g^k mod p, m*S mod p). The decryption process involves the calculation of the shared secret S, and the calculation of the plaintext as m = C2 * S^-1 mod p.

That the security of the ElGamal algorithm is dependent on the difficulty of computing discrete logarithms in a finite field, which is related to the discrete logarithm problem in number theory.

3.2 RSA Methodology

RSA is a widely-used public-key cryptographic algorithm that is used for secure communication and data encryption. The algorithm was first proposed by Ron Rivest, Adi

Shamir, and Leonard Adleman in 1978 and is based on the mathematical problem of factoring large numbers [4][5].

3.2.1 Key Generation:

a. Choose two distinct prime numbers p and q.

b. Calculate $n = p * q$.

c. Calculate $\varphi(n) = (p-1) * (q-1)$.

d. Choose an integer e, such that $1 < e < \varphi(n)$ and e is coprime to $\varphi(n)$.

e. Calculate d, such that $d * e \equiv 1 \mod \varphi(n)$.

f. The public key is (n, e) and the private key is (n, d).

P and Q are two significant prime numbers in this stage. Calculating n involves multiplying p by q. $\varphi(n)$ is the Euler's totient function, which is equivalent to $(p-1) * (q-1)$. An integer e is chosen such that it is coprime to $\varphi(n)$ and $1 < e < \varphi(n)$. The private key d is calculated using the Extended Euclidean Algorithm, such that $d * e \equiv 1 \mod \varphi(n)$. The public key is (n, e) and the private key is (n, d).

3.2.2 Encryption:

a. Convert the message M into an integer m, such that $0 <= m < n$.

b. Calculate the ciphertext as $C \equiv m \wedge e \mod n$.

In this stage, the message M is converted into an integer m, such that $0 <= m < n$. The ciphertext is calculated as $C \equiv m \wedge e \mod n$.

3.2.3 Decryption:

a. Calculate the plaintext as $m \equiv C \wedge d \mod n$.

In this stage, the plaintext is calculated as $m \equiv C \wedge d \mod n$.

The security of the RSA algorithm is based on the difficulty of factoring large integers.

4.0 RESULT AND DISCUSSION

After implementing the RSA and ELGAMAL Cryptographic algorithm on the python programming language for eight (times), the average speed of the encryption process and the decryption process were recorded. Table 1 displays the results.

Key Size in Bits	**RSA**		**ELGAMAL**	
	Encryption	Decryption	Encryption	Decryption
1024	**0.139**	**0.231**	**0.703**	**0.305**

2048	**0.561**	**0.631**	**1.107**	**0.705**

Table 1: Displaying the encryption and decryption speed using different key sizes

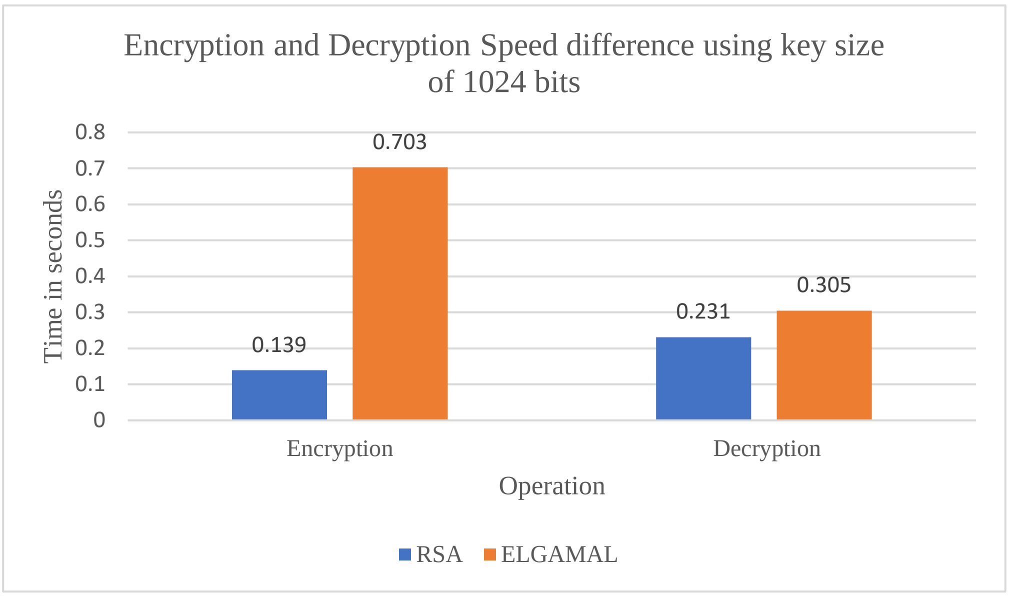


Figure 1: Displaying the encryption and decryption speed difference using key size of 1024 bits

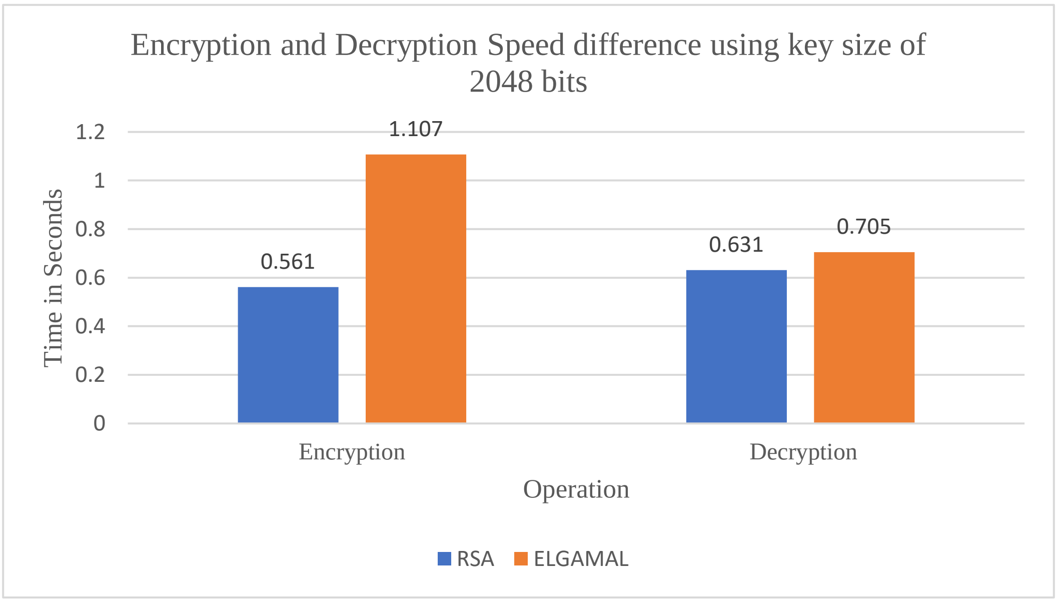


Figure 2: Displaying the encryption and decryption speed difference using key size of 2048 bits

From table 1 and figure 1, it is observed that it takes longer time to encrypt text in ELGAMAL algorithm than in RSA algorithm. That means in terms of encryption speed, RSA algorithm is better than ELGAMAL algorithm.

Similarly, in figure 2, it is also observed that, it takes longer time to decrypt text in ELGAMAL algorithm than in RSA algorithm. Meaning, in terms of decryption speed, RSA algorithm remains better than ELGAMAL algorithm.

This could be observed to be due to the fact that, ELGAMAL requires multiple modular exponentiation operations which is more computationally intensive than RSA involving only a single modular exponentiation operation.

However, with the ELGAMAL cryptographic algorithm, it takes longer time in the decryption process than the encryption process. This is because, the number of operations carried out in the ELGAMAL decryption process is fewer than the encryption process. The encryption procedure entails creating a random number, then perform some modular exponentiations before adding the result values together to get the ciphertext. This is computationally intensive as noted earlier. Also. This accounts to the decryption process being faster than the encryption process.

5.0 CONCLUSION

According to this study, RSA is generally noted to be faster than ElGamal in terms of both encryption and decryption speed. In ElGamal, there are several modular exponentiation operations involved in the encryption and decryption process, whereas in RSA, a single modular exponentiation operation is involved in the encryption and decryption processes.

Overall, the research conducted evidently suggest that, RSA is generally faster than ELGAMAL in terms of both encryption and decryption speed. However, a number of variables, including the necessary security level, key sizes, application, and performance requirements, affect the selection of the most suitable cryptographic algorithms.

6.0 RECOMMENDATIONS

This output of this experimental research has proven that, RSA cryptosystem is generally faster than the ELGAMAL cryptosystem in their basic operations; encryption and decryption due to its computational efficiency. However, great consideration must be made to a number of metrics when selecting and implementing any of the algorithms in question. These include, the hardware, key sizes, amount of data, the needed level of security and suitability for specific applications/ software. These affect the speed of the algorithms.

Future research will look into other optimisation methods that will increase the speed and effectiveness of both the RSA and ELAGAMAL algorithms.

References

[1] Sood, K., Sharma, A., & Khanna, A. (2020). Comparative Analysis of Symmetric and Asymmetric Encryption Algorithms. International Journal of Advanced Research in Computer Science, 11(5), 277-281

[2] Stinson, D. R. (2006). Cryptography: Theory and Practice. CRC Press.

[3] Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). Handbook of applied cryptography. CRC press.

[4] Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. Pearson

[5] Rivest, R., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2), 120-126

[6] ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, 31(4), 469-472

[7] Schneier, B. (2015). Applied Cryptography: Protocols, Algorithms, and Source Code in C. Wiley

[8] Alizadeh, M., Sadeghi, A. R., & Salmasizadeh, M. (2017). A Comparative Study of RSA and ElGamal Cryptosystems. International Journal of Network Security, 19(3), 357-363.

[9] Liu, W., Zhang, J., & Zhou, J. (2020). A Comparative Study on the Performance of RSA and ElGamal Algorithms in Software and Hardware Implementation. International Journal of Advanced Computer Science and Applications, 11(10), 168-175.

[10] O'Reilly, A., Trappe, W., & Shmatikov, V. (2019). Are RSA and ElGamal secure for hardware? In Proceedings of the 2019 on Great Lakes Symposium on VLSI (pp. 201-20)