A Comprehensive Evaluation of the Rivest-Shamir-Adleman (RSA) Algorithm Performance on Operating Systems using Different Key Bit Sizes

Kwame Assa-Agyei
Department of Computer Science
Nottingham Trent University
Nottingham, United Kingdom

Funminiyi Olajide
Department of Computer Science
Nottingham Trent University
Nottingham, United Kingdom

ABSTRACT

In today's digital world, practically everyone uses the Internet for various purposes. Most data sent over the Internet contains personal or private information that people desire to keep hidden. There are numerous encryption techniques available for concealing data. However, none of the previous research has thoroughly examined different bit sizes of RSA algorithms on Windows and Linux. According to previous studies, there is a range of factors, such as operating systems, compilers, and environmental conditions that affect how well cryptographic algorithms function. This study investigates the various key bit sizes used in the RSA technique (512, 1024, 2048, and 4096). The time it took to generate the following: private keys, public keys, signature blocks, and verification processes utilized in the RSA method was the basis for this experiment. Two virtual machines, one running Windows and the other running Linux, were used for the experiment. The experiment was conducted on three HP laptops, each equipped with a 3.38GHz Intel Core i5 processor, 12GB of RAM, and a 1TB SSD. The experiments were repeated three times on each laptop, and the average times were recorded for both virtual machines. It was demonstrated in this investigation that the Linux operating system outperforms the Windows operating system in terms of overall performance. According to the results, both RSA private and public key generation were faster on Linux than on Windows. Furthermore, the test for digital signature and verification throughput indicated that total signatures and verifications per second were higher on Linux than on Windows for all RSA key bit sizes. Finally, increasing the RSA key bit reduced throughput for both digital signatures and verification in both operating systems.

Keywords

Cryptography, Asymmetric Cryptosystems, RSA, Throughput, Digital Signature, Verification, Performance, Key Bits, Public key, Private Key

1. INTRODUCTION

The quick and continuous evolution of network technology is also revolutionizing our environment and different parts of our everyday lives, such as business, legal, and social lives. This surge in network technology development, however, has a drawback. The more links made to diverse global computer networks on a daily basis, the more vulnerable the connected systems are to unwanted access [1]. This is because common methods such as network scanning, spoofing, and so on have become more sophisticated, making information sharing unsafe. Furthermore, the recent emergence of internet-based transaction applications such as internet banking, online shopping, and bill payment, among others, which involve the sharing of very sensitive information between two or more parties, necessitates the use of very secure end-to-end connections that ensure the information's confidentiality, integrity, authenticity, and so on [2]. As a result, security is a critical component in network technology development that must be addressed in order to safeguard information from destruction and unwanted infiltration. This security issue has prompted the creation of numerous technologies such as passwords, biometrics, patterns, encryption, and so on to aid in the elimination of information/network security difficulties, particularly the protection of information from unwanted access [3]. However, of all these approaches, cryptography is known to be the safest and most reliable in keeping sensitive information confidential from unwanted users [4]. Cryptography is the study and use of techniques used to safeguard network communication from intruders such as hackers and attackers. Through different modifications, it renders the information incomprehensible to a third party or an intruder. Cryptography uses four objectives and goals to do this [5]:

1. Confidentiality: guard the user's identity and data privacy against being accessed by others.
2. Integrity: the preservation of data against being altered by others.
3. Authentication: Ensuring that the data originated from a specific party.
4. Non-repudiation: The inability of a single party to deny transmitting a message.

The presence or absence, as well as the type and number of keys, determine the type of cryptosystem, as well as the encryption and decryption phases involved in that cryptosystem [6]. Key-based cryptosystems are classed or grouped into two types based on the number and kind of key(s) utilized. The first is symmetric key cryptosystems such as Blowfish, Twofish, 3DES, DES, AES, RC6, and others, which are also known as private, secret, or conventional key cryptosystems due to the use of a single key for both encryption and decryption. Despite its ease of implementation, their main risk is the use of only one key for encryption and decryption. The second type is asymmetric key cryptosystems such as RSA, Elgamal, and others, which are commonly referred to as public-key cryptosystems because the sender and receiver employ two separate keys in this cryptosystem. The public key is used for encryption (converting plaintext/message to cipher text), while the private key is used for decryption (converting cipher text back to plaintext or message). The encryption key is made available to everyone who wishes to send a message to the receiver (with whom the key for decryption is stored in secret) [7]. A comparable secret key cannot be easily derived from a specific public key [8]. Many theoretical analyses have been conducted to evaluate algorithms and their behaviours, but it is necessary to evaluate them practically, especially because many factors, such as operating systems, compilers, environment specifications, and many others, can affect the performance of a cryptographic algorithm's outcome [9]. The study's main purpose is to compare the performance of various key bit sizes on Windows and Linux. This research will focus on the following specific objectives:

1. To compute the time required to generate public and private keys
2. To determine the speed required to perform digital signature and verifications per second
3. To raise awareness of the importance of understanding and selecting the optimal operating system for the RSA cryptographic techniques.

This paper is organized as follows. Section 2 presents the principle of tradition RSA algorithm and literature review of previous works are presented in Section 3. Section 4 presents the experimental setting and tools used to evaluate our methodology, whereas Section 5 presents the implementation and section 6 discusses the results and comparison of the study. Finally, concluding remarks are presented in Section 7

2. THE PRINCIPLE OF TRADITIONAL RSA ALGORITHM

Ron Rivest, Adi Shamir, and Leonard Adleman devised RSA in 1978. It is a well-known public key encoding method for key exchange, digital signatures, and database block encryption. The RSA algorithm has a different size key and various sizes of encoding blocks. It is an asymmetric encoding system that relies on numeral synthesis. It generates both the public and private keys using two basic numbers [10]. It is one of the most well-known public key cryptosystems for key exchange, digital signatures, and data block encryption. It produces two keys: a public key for encryption and a private key for message decryption. The public and private keys are generated using two prime numbers. These two distinct keys are utilized for encryption and decryption. When the message is sent, the sender encrypts it with the recipient's public key, and the receiver decrypts it with his or her own private key. The user can utilize the private key of certification authorities (CA) key pairs to sign a document with a secure digital signature. On the recipient side, the public key of these key pairs can be used to validate the signature and the integrity of the document. The certificate is signed just once, but it must be checked multiple times. Because signature verification is the most common approach, the RSA algorithm is ideally suited for this task. When sending an email, the message must be signed and encrypted. Each recipient must then validate the signature using the right decryption key. Using RSA for encryption and digital signatures provides a solid foundation for safe emails [11]. RSA operations can be divided into three categories: key creation, encryption, and decryption [12].

a. Key Generation
Every user creates a unique public/private key combination by:
At random, two large primes are chosen: - p, q

- Calculating the system modulus
- N= p.q
  NB: ø (N) = (p-1) (q-1)
- Choose an encode key e at random:1<e<ø (N), gcd (e, ø (N)) =1
- Solve the following equation to find the decode key d: e.d=1 mod ø (N) and 0≤d≤N
- Publish their public encode key: KU= {e, N}
- Keep secret decode key: KR= {d, p, q}

b. **Usage of Keys: To encrypt a message M, the Sender:**
- Obtains public key of recipient KU={e,N}
- Computes: C=Me mod N, where 0≤M<N

c. **To decrypt the cipher text C, the Receiver:**
- Uses their private key KR={d,p,q}
- Computes: M=Cd mod N

This protects client information over the Internet and prevents other users from accessing the original data because it has been encoded [13].

3. RELATED WORK

RSA is regarded as one of the first systems for generating digital signatures. The scheme's security is predicated on the difficulty of factorizing a large number N, which is the product of two large primes (p and q) [14].

There are some benefits and drawbacks of adopting RSA techniques over traditional symmetric encryption schemes[15]:

Advantages

1. The fundamental benefit of RSA is greater security, as private keys are never transferred or divulged to anyone. In contrast, there is always the possibility that an adversary will discover the secret key while it is being transmitted in a secret-key system.

2. Another significant advantage of public-key systems is that they may be used to generate digital signatures. Authentication via secret-key systems necessitates the disclosure of some secrets as well as the trust of a third party.

3. Digitally signed messages can be proven valid by a third party, such as a judge, making them legally enforceable.

Disadvantage

1. One downside of employing public-key cryptography for encryption is its poor processing speed.

Digital signatures can be secured using public key cryptography. Examples include the RSA algorithm and the Digital Signature Algorithm (DSA). The RSA method is utilized in a variety of schemes and is a fundamental technique to implementing Digital Signature Schemes. The use of digital signatures with the RSA technique will improve cloud computing data security [16]. Mohamad et al. [17] provide a study of the RSA scheme of asymmetric cryptography approaches in the year 2021. It seeks to present the areas of RSA scheme use, such as public networks, wireless sensor networks, image encryption, cloud computing, proxy signature, Internet of Things, and embedded device, based on the achievements of researchers over the previous decade. Aside from that, the article investigated the trends and performance parameters of the RSA scheme, such as security, speed, efficiency, computational complexity, and space, based on the number of experiments completed. In 2015, Saxena and Kapoor [18] published a survey of various parallel implementations of the RSA algorithm that includes a wide range of hardware and software implementations. Parallel programming is a new area of research that tries to increase performance and efficiency by executing instructions more rapidly and effectively on multi-core machines. The authors' goal was to educate forthcoming researchers on the different parallel RSA implementation strategies that have already been created. They explored a number of concurrent RSA implementation strategies proposed by numerous experts worldwide in order to achieve high performance and throughput in the realm of RSA and public key cryptography. This survey is limited on one feature of RSA schemes: parallel implementation. Every Secure Socket Layer connection starts with a handshake, in which the two sides convey their capabilities to the other party, complete authentication, and agree on their session keys. The session keys are then used to encrypt the remainder of the communication, which may span several sessions. They are then erased. The purpose of the key exchange phase is to allow the two parties to safely negotiate the keys, preventing anyone else from learning these keys. There are other key exchange mechanisms available, but the most generally used one at the present is based on RSA and uses the server's private key to safeguard the session keys [19]. This paper's author presented a concrete RSA signature structure that can enable variable-sized file blocks and public auditing. The researchers presented a formal security model for IDCDIC and demonstrated the security of their architecture under the RSA assumption with large public exponents in the random oracle model. They demonstrated the presence of their proposal by creating a protocol prototype. The results of the implementation showed that the suggested ID-CDIC protocol is feasible and flexible in real life [20]. According to authors in [21], researchers did a comparison study of RSA and ECC, RSA has around ten times more computing overheads than ECC. ECC reduces the length of system parameters and key pairs. ECC offers significant bandwidth savings over RSA because RSA requires a substantially bigger key size for the same degree of protection. ECC generates keys more quickly than RSA. ECC is faster at encryption but slower at decryption than RSA. As a result, they concluded that ECC will be more efficient in terms of security for iOS devices. It was concluded that, when compared to the RSA technique, the elliptic curve discrete logarithm problem made ECC the most efficient with a smaller key size. Public-key encryption can be used to solve difficulties associated with traditional encryption. However, because it adds a lot of overhead, it has not gained as much traction as traditional encryption. As a result, it is critical to identify solutions to reduce overheads while not sacrificing other aspects of security so that the desirability of public-key can be exploited. When the RSA and ECC ciphers were examined, the ECC cipher was shown to have significantly lower overheads than the RSA cipher. Because of its ability to provide the same level of security as RSA while employing shorter keys, the ECC has been proven to have various advantages. However, its downside, which may even overshadow its beauty, is its lack of maturity, as mathematicians consider that not enough research on ECDLP has been done. Furthermore, the researchers concluded that, while both methods are valid, RSA is superior to ECC for the time being since its security can be trusted more [22]. The authors of this paper designed an encoding approach, which first determines whether the private and public-key values generated during the encoding procedure contain a prime number, then combines with Pascal's triangle theorem, the RSA algorithm model, and an inductive technique to build a new cryptosystem that meets homomorphic computation of some operations on cipher texts. The authors also concluded that RSA is a partially homomorphic cryptosystem due to its multiplication characteristics. However, a completely homomorphic encryption should meet not only the multiplication criterion but also the addiction characteristic. The addition algorithm was designed to achieve the fully homomorphic encryption characteristic [23]. The authors in [23] carried out an experiment on the Research and Implementation of RSA Algorithms for Encryption and Decryption in 2011. It was determined that the encryption and decryption system can protect information against tampering, forgery, and counterfeiting by ensuring the information's confidentiality, integrity, and certainty. It explored how to apply RSA information security issues to one's everyday life, as well as the application of RSA and the fundamental concepts of data encryption and decryption. It presented a new software based on RSA cryptography and its vast application to improve the RSA algorithm. The authors in [24] carried out an experiment on the Research and Implementation of RSA Algorithms for Encryption and Decryption in 2011. It was determined that the encryption and decryption system can protect information against tampering, forgery, and counterfeiting by ensuring the information's confidentiality, integrity, and certainty. It explored how to apply RSA information security issues to one's everyday life, as well as the application of RSA and the fundamental concepts of data encryption and decryption. It presented a new software based on RSA cryptography and its vast application to improve the RSA algorithm. The authors performed their review without using any proper research methodology and considered very few articles for their review. The [25] paper study analyzed RSA with different key sizes and word length variables in terms of encryption and decryption process memory size and execution time. The experiment results demonstrated that RSA execution time is slower and requires more memory than ECC. The fundamental issue with the DES algorithm is key agreement and key distribution, but RSA encryption and decryption take longer. The simulation results showed that RSA is slower in performance than DES and that the RSA algorithm throughput is not better than the DES algorithm. As a result, the paper establishes the concept and specifications for elliptic curve lightweight cryptography. This type of study is missing in the literature for RSA public key cryptography, a worthy competitor of ECC.

According to the analysis of the aforementioned surveys on RSA schemes and their connected domains, a systematic and detailed study on RSA public key cryptography is missing. All research concentrate on a fairly narrow region and the experiments are mostly carried out on a specific operating system and key bit size. This research will be very useful for researchers and practitioners in the field of cryptography, particularly in the field of RSA public key cryptography, in order to understand the specific key bits to be implemented when executing public and private keys, signature, and verification techniques for generating purpose systems.

4. EXPERIMENTAL DESIGN

To obtain reliable values for comparing cryptographic algorithms on Windows and Linux, they must be executed on workstations with comparable setups. As a result, the Oracle VM Virtual Box is utilized to generate two virtual machines running the two operating systems required for the experiment on three different machines (A, B and C). The three machines A, B and C have identical specifications, featuring a 3.38GHz Intel Core i5 processor, 12GB of RAM, and a 1TB SSD. Two virtual computers were deployed on each machine for this investigation, one for each operating system, with the following configurations:

Table 1. Virtual machines set-up configuration on the t machines

Virtual Machine	Configuration Details
Machine 1	Windows 10 Pro-64-bits, 1.8GHz (1 Core), 2048 MB memory, 50 HDD size
Machine 2	Fedora 34, 1.8GHz (1 Core), 2048 MB memory size, 50 HDD size

The algorithms are implemented using the Python 3.10. Different key bit sizes were employed in the experiment: 512, 1024, 2048, and 4096. Several metrics were collected during the experiments:

a) The generation time for both private and public keys.
b) The time required to generate and validate digital signature

A series of similar experiments were carried out, and average values were determined to ensure that the metrics collected were relevant. Tables II to V gives the results of a comparison of the execution time of key creation, signature block, and verification for various key bit sizes. The dataset is made up of two operating systems, namely Windows and Linux. These operating systems were chosen because market share statistics suggest that they are the most extensively used. Furthermore, the majority of users rely on this software for their daily transactions (i.e. e-banking, e-commerce, and social network)[1].

5. IMPLEMENTATION OF RSA ALGORITHMS

5.1 RSA private and public key generation Throughput

Both RSA private and public key generation were executed by generating keys of various sizes of 512, 1024 2048, and 4096 bits. The experiment was carried out ten (10) times on each virtual machine and the mean time in megabits per second was recorded. Table I and Table II provide detailed performance analysis of the throughput on RSA private and public key generation using Windows and Linux. The data generated during the experiment are imported into R (version 3.12) and the necessary graphs *(Fig 1 and Fig 2)* were plotted.

Table 2. RSA Private Key Generation Throughput on Windows and Linux

OS	Key Bits	Mean Throughput in MB/s
Windows	512	8.932
Linux	512	10.729
Windows	1024	6.788
Linux	1024	7.333
Windows	2048	3.042
Linux	2048	3.27
Windows	4096	0.558
Linux	4096	0.621

[1] Net Marketshare, 2022

Table 3. RSA public key generation throughput on Windows and Linux

OS	Key Bits	Mean Throughput in MB/s
Windows	512	84.11
Linux	512	116.656
Windows	1024	89.613
Linux	1024	102.955
Windows	2048	64.306
Linux	2048	70.643
Windows	4096	32.277
Linux	4096	39.895

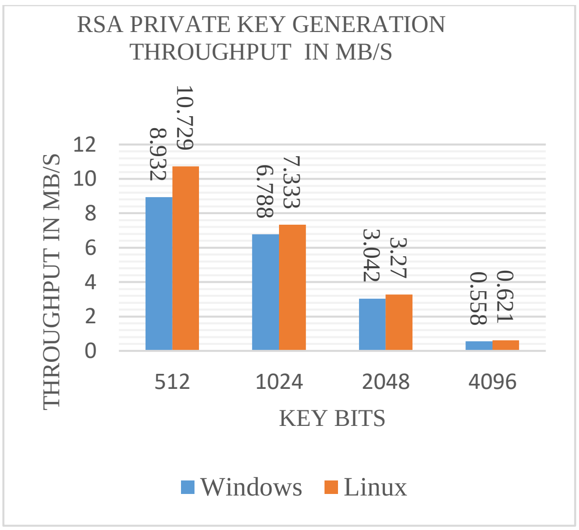

Fig. 1: Average RSA private key generation throughput on Windows and Linux

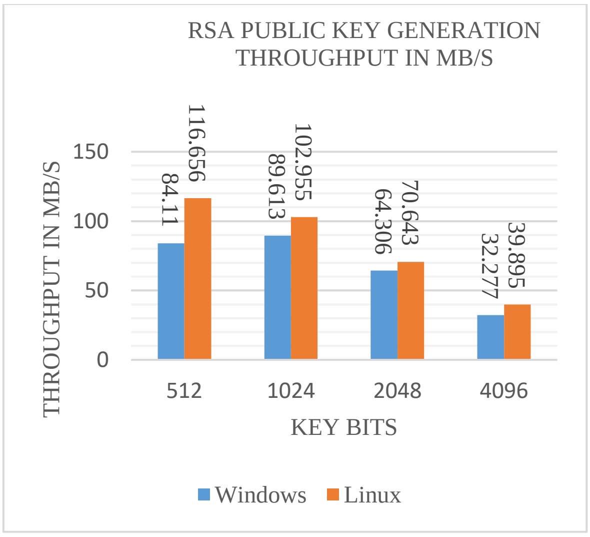

Fig. 2: Average RSA public key generation throughput on Windows and Linux

5.2 RSA Signature and verification throughput

RSA performs digital signature by applying the private key to generate a signature that can be verified by using the public key. The experiments measured and recorded the speed used in creating the signatures and verifying them using the various key bit sizes of 512, 1024, 2048, and 4096 bits. Both the signature and verification throughput are recorded in Tables III and IV on Windows and Linux respectively. *Fig 3* and *Fig 4* show the graphs of the throughput of signatures and verification performed by RSA of key bit sizes of 512, 1024, 2048, and 4096 bits.

Table 4. RSA signature throughput on Windows and Linux

OS	**Key Bits**	**Sign/s**
Windows	512	17653.6
Linux	512	20955.5
Windows	1024	6628.5
Linux	1024	7248.8
Windows	2048	1485.5
Linux	2048	1597
Windows	4096	135.7
Linux	4096	151.7

Table 5. RSA signature verification throughput on Windows and Linux

OS	**Key Bits**	**Verify/s**
Windows	512	163732.5
Linux	512	227844.4
Windows	1024	87471.6
Linux	1024	103677.5
Windows	2048	31413.9
Linux	2048	34494.3
Windows	4096	7881.4
Linux	4096	9740.2

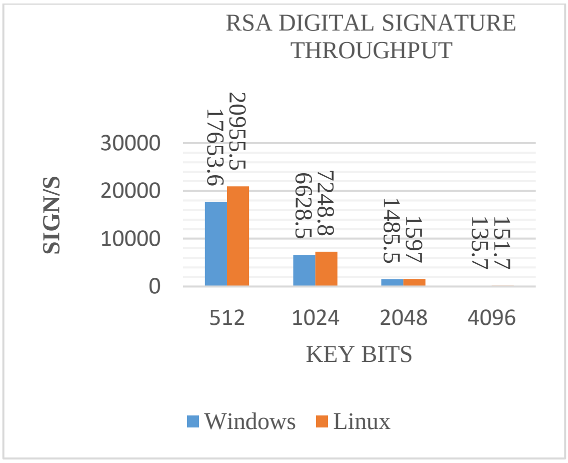

Fig. 3: RSA sign throughput on Windows and Linux

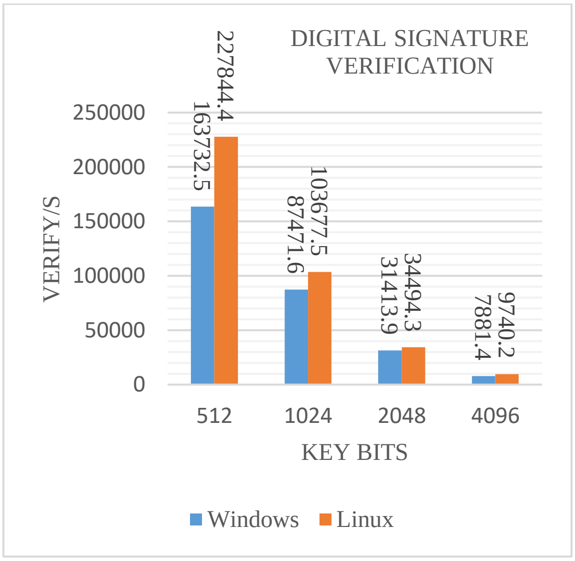

Fig. 4: RSA signature verification throughput on Windows and Linux

6. DISCUSSION OF RESULTS

The RSA algorithm is tested with four key bit sizes: 512, 1024, 2048, and 4096. The throughput required to produce the private keys for the four (4) different key bit sizes was investigated. In Fig 1, it was discovered that Linux used 10.729MB/s to generate private keys, whereas Windows used 8.932MB/s for the key bit of 512. The speed utilized to generate 1024 key bit sizes for both Windows and Linux, on the other hand, decreased to 6.788MB/s and 7.333MB/s, respectively. Using the 2048 key bit, Linux achieved a faster speed of 3.27 than

Windows, which achieved a speed of 3.042MB/s. It can be concluded that using 4096 key bits on both Windows and Linux took a lower time to generate the keys after the 2048 key bit. *Fig 2* also depicts a similar pattern to *fig 1*. Using 512 key bit in generating public keys on Linux shows a speed of 116.656 MB/s while 4096 also shows the lowest times in producing the public keys on Windows and Linux. It can be concluded that both RSA private and public key generations were faster on Linux than on Windows for all key bit sizes tested. It was also observed that as the key bits increases in size, the speed decreases. Fig 3 shows the time taken to generate signature for RSA algorithms with different input key sizes and it can be seen that using 512 key bits on Linux can generate up to 20955.5 signatures per second. The experimental values in Table IV and V show that as the speed decreases as the key bit size increases. Figure 4 also depicts the time required for RSA signature verification with various input key sizes. It can be shown that 4096 took the least amount of time to validate signatures across all key length sizes.

7. CONCLUSION

This paper presents a comparison set on one of the most well-known asymmetric cryptography algorithms. In this work, it was proven that Linux operating system has better overall performance than Windows operating system. From the obtained results, both RSA private and public key creation were faster on Linux than on Windows operating systems. It was also discovered that when the size of the key bit increases, the throughput decreases. For all RSA key bit sizes, the test for digital signature and verification throughput found that total signings and verification per second were greater on Linux than on Windows. Finally, when the RSA key bits increased, the throughput for both signings and verification decreased in both operating systems. The study also identified future works to consider, such as the measurement of other performance metrics such as memory consumption, and energy consumption.

8. ACKNOWLEDGMENTS

Our gratitude to all the people and groups who helped to complete this study piece. Our gratitude is also expressed to the research team and our colleagues, who helped us out by sharing their skills, information, and encouragement. They played a critical role in carrying out the studies and collecting the results.

9. REFERENCES

[1] R. S. Cordova, R. L. R. Maata, A. S. Halibas, and R. Al-Azawi, "Comparative analysis on the performance of selected security algorithms in cloud computing," *2017 Int. Conf. Electr. Comput. Technol. Appl. ICECTA 2017*, vol. 2018-Janua, pp. 1–4, 2017, doi: 10.1109/ICECTA.2017.8252030.

[2] M. E. Haque, S. Zobaed, M. U. Islam, and F. M. Areef, "Performance Analysis of Cryptographic Algorithms for Selecting Better Utilization on Resource Constraint Devices," *2018 21st Int. Conf. Comput. Inf. Technol. ICCIT 2018*, pp. 21–23, 2019, doi: 10.1109/ICCITECHN.2018.8631957.

[3] S. Omer, A. Faroog, M. Koko, A. Babiker, and N. Mustafa, "Comparison of Various Encryption Algorithms and Techniques for improving secured data Communication," *IOSR J. Comput. Eng. Ver. III*, vol. 17, no. 1, pp. 2278–661, 2015, doi: 10.9790/0661-17136269.

[4] A. V. Mota, A. Sami, K. C. Shanmugam, Bharanidharan Yeo, and K. Krishnan, "Comparative Analysis of Different Techniques of Encryption for Secured Data Transmission," *IEEE Int. Conf. Power, Control. Signals Instrum. Eng.*, vol. 54, no. 4, pp. 847–860, 2017.

[5] S. Al Busafi and B. Kumar, "Review and analysis of cryptography techniques," *Proc. 2020 9th Int. Conf. Syst. Model. Adv. Res. Trends, SMART 2020*, pp. 323–327, 2020, doi: 10.1109/SMART50582.2020.9336792.

[6] I. Jahan, M. Asif, and L. Jude Rozario, "Improved RSA cryptosystem based on the study of number theory and public key cryptosystems," *Am. J. Eng. Res.*, no. 1, pp. 143–149, 2015, [Online]. Available: www.ajer.org.

[7] M. Panda, "Performance analysis of encryption algorithms for security," in *International Conference on Signal Processing, Communication, Power and Embedded System, SCOPES 2016 - Proceedings*, 2017, pp. 278–284, doi: 10.1109/SCOPES.2016.7955835.

[8] H. Kim and S. Lee, "Design and implementation of a private and public key crypto processor for next-generation it security applications," *Malaysian J. Comput. Sci.*, vol. 19, no. 1, pp. 29–45, 2006.

[9] H. Dibas and K. E. Sabri, "A comprehensive performance empirical study of the symmetric algorithms:AES, 3DES, Blowfish and Twofish," *2021 Int. Conf. Inf. Technol. ICIT 2021 - Proc.*, pp. 344–349, 2021, doi: 10.1109/ICIT52682.2021.9491644.

[10] O. G. Abood and S. K. Guirguis, "A Survey on Cryptography Algorithms," *Int. J. Sci. Res. Publ.*, vol. 8, no. 7, 2018, doi: 10.29322/ijsrp.8.7.2018.p7978.

[11] N. Thein, H. A. Nugroho, T. B. Adji, and I. W. Mustika, "Comparative Performance Study on Ordinary and Chaos Image Encryption Schemes," *Proc. - 2017 Int. Conf. Adv. Comput. Appl. ACOMP 2017*, pp. 122–126, 2018, doi: 10.1109/ACOMP.2017.25.

[12] M. Panda and A. Nag, "Plain Text Encryption Using AES, DES and SALSA20 by Java Based Bouncy Castle API on Windows and Linux," *Proc. - 2015 2nd IEEE Int. Conf. Adv. Comput. Commun. Eng. ICACCE 2015*, pp. 541–548, 2015, doi: 10.1109/ICACCE.2015.130.

[13] M. Gobi and R. Sridevi, "An Approach for Secure Data Storage in Cloud Environment," *Int. J. Comput. Commun. Eng.*, vol. 2, no. 2, pp. 206–209, 2013, doi: 10.7763/ijcce.2013.v2.171.

[14] S. B. Sadkhan and R. S. B. Sadkhan, "Analysis of Different Types of Digital Signature," *8th IEC 2022 - Int. Eng. Conf. Towar. Eng. Innov. Sustain.*, pp. 241–246, 2022, doi: 10.1109/IEC54822.2022.9807502.

[15] Mitali, V. Kumar, and A. Sharma, "A Survey on Various Cryptography Techniques," *Int. J. Emerg. Trends Technol. Comput. Sci.*, pp. 191–199, 2014, doi: 10.2307/j.ctt46nrzt.12.

[16] N. Ferguson, B. Schneier, and T. Kohno, "Chapter 9. Generating Randomness," *Cryptogr. Eng. Des. Princ. Pract. Appl.*, pp. 137–161, 2010.

[17] M. S. A. Mohamad, R. Din, and J. I. Ahmad, "Research trends review on RSA scheme of asymmetric cryptography techniques," *Bull. Electr. Eng. Informatics*, vol. 10, no. 1, pp. 487–492, 2021, doi: 10.11591/eei.v10i1.2493.

[18] S. Saxena and B. Kapoor, "State of the Art Parallel

Approaches For Rsa Public Key Based Cryptosystem," *Int. J. Comput. Sci. Appl.*, vol. 5, no. 1, pp. 81–88, 2015, doi: 10.5121/ijcsa.2015.5108.

[19] G. Mogoş and G. Radu, "Hybrid secure socket layer protocol," *Int. Conf. Sci. Pap. AFASES*, vol. 2, no. 2, pp. 91–96, 2014.

[20] Y. Yu *et al.*, "Cloud data integrity checking with an identity-based auditing mechanism from RSA," *Futur. Gener. Comput. Syst.*, vol. 62, pp. 85–91, 2016, doi: 10.1016/j.future.2016.02.003.

[21] M. Alam, I. Jahan, L. J. Rozario, and I. Jerin, "A Comparative Study of RSA and ECC and Implementation of ECC on Embedded Systems," *Int. J. Innov. Res. Adv. Eng.*, vol. 3, no. 03, pp. 86–93, 2016, [Online]. Available: http://www.ijirae.com/volumes/Vol3/iss3/15.MRAE10096.pdf.

[22] B. K. Alese, E. D. Philemon, and S. O. Falaki, "Comparative analysis of public-key encryption schemes," *Int. J. Eng. Technol.*, vol. 2, no. 9, pp. 1152–1568, 2012, [Online]. Available: http://iet-journals.org/archive/2012/sep_vol_2_no_9/1298141336454596.pdf.

[23] P. Sha and Z. Zhu, "The modification of RSA algorithm to adapt fully homomorphic encryption algorithm in cloud computing," *Proc. 2016 4th IEEE Int. Conf. Cloud Comput. Intell. Syst. CCIS 2016*, vol. 2, no. 1, pp. 388–392, 2016, doi: 10.1109/CCIS.2016.7790289.

[24] X. Zhou and X. Tang, "Research and implementation of RSA algorithm for encryption and decryption," *Proc. 6th Int. Forum Strateg. Technol. IFOST 2011*, vol. 2, pp. 1118–1121, 2011, doi: 10.1109/IFOST.2011.6021216.

[25] P. R. Vijayalakshmi and K. B. Raja, "Performance Analysis of RSA and ECC in Identity- Based Authenticated New Multiparty Key Agreement Protocol," *Int. Conf. Comput. Commun. Appl.*, 2012.