Egyptian Informatics Journal 23 (2022) 291–301

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

An improved random bit-stuffing technique with a modified RSA algorithm for resisting attacks in information security (RBMRSA)

Falowo O. Mojisola [a], Sanjay Misra [b,*], C. Falayi Febisola [a], Olusola Abayomi-Alli [c], Gokhan Sengul [d]

[a] Department of Computer Science, Federal University of Agriculture Abeokuta, Ogun State, Nigeria
[b] Department of Computer Science and Communication, Østfold University College, Halden, Norway
[c] Department of Software Engineering, Kaunas University of Technology Kaunas, Lithuania
[d] Departmnt of Computer Engineering, Atilim University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 20 November 2021
Revised 1 February 2022
Accepted 11 February 2022
Available online 4 March 2022

Keywords:
Computational complexity
Exhaustive search attack
Randomization
Bit insertion
Avalanche effect

ABSTRACT

The recent innovations in network application and the internet have made data and network security the major role in data communication system development. Cryptography is one of the outstanding and powerful tools for ensuring data and network security. In cryptography, randomization of encrypted data increases the security level as well as the Computational Complexity of cryptographic algorithms involved. This research study provides encryption algorithms that bring confidentiality and integrity based on two algorithms. The encryption algorithms include a well-known RSA algorithm (1024 key length) with an enhanced bit insertion algorithm to enhance the security of RSA against different attacks. The security classical RSA has depreciated irrespective of the size of the key length due to the development in computing technology and hacking system. Due to these lapses, we have tried to improve on the contribution of the paper by enhancing the security of RSA against different attacks and also increasing diffusion degree without increasing the key length. The security analysis of the study was compared with classical RSA of 1024 key length using mathematical evaluation proofs, the experimental results generated were compared with classical RSA of 1024 key length using avalanche effect in (%) and computational complexity as performance evaluation metrics. The results show that RBMRSA is better than classical RSA in terms of security but at the cost of execution time.

1. Introduction

Enhancing data and network security is becoming one of the most urgent tasks in enterprise, company, and private communications where the exchange of very vital and sensitive information should be adequately protected among the users through the internet and networks. A cryptographic algorithm is one of the powerful tools for data security implementation, which comprises various security techniques and makes up the science of encryption and decryption [1]. Cryptography is defined as storing data or messages and converting such data or messages into a chaotic state that only the right user can access and process [2]. Encryption Algorithm is the security of data to confidentiality privacy access control, among others. The original form of data is the clear text or Plaintext, which is converted into codes that are not understood [3]. Decryption means conversion of message in the chaotic state back to readable forms at the other end. The RSA is one of the examples of asymmetric key cryptography, invented and named after the three scientists.

RSA Algorithm is a Public Key algorithm called RSA cryptosystem [4]. RSA is an asymmetric cryptography algorithm because the public key is shared or stored in a public database, in which the user can convert the plain message into ciphertext with the use of a public key (e), and decryption is done through the use of private key 'd' which is not shared with anyone but reserved secretly. Security of this cryptosystem is built on time and energy required to factor big numbers [5]. RSA cryptography as a block cipher, the plaintexts, and encrypted text are integer in the interval

* Corresponding author.
E-mail addresses: sanjay.misra@hiof.no (S. Misra), olusola.abayomi-alli@ktu.edu (O. Abayomi-Alli), gokhan.sengul@atilim.edu.tr (G. Sengul).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.eij.2022.02.001
1110-8665/

0 to N-1 for a specific value of n; every message is mapped to an integer. The message is encrypted inside the block, and each block has a binary value that should be not equal or greater than the N. RSA algorithm is done in three steps: key generation, encryption, and decryption [6].

RSA cryptographic algorithms have security setbacks, which prone RSA algorithms to different attacks, including timing attacks, common modulus attacks, known-plaintext attacks, and exhaustive search attacks [7]. The length of time required to process this cryptosystem is much higher, a smaller key length reduced the processing time but to the detriment of the security of this cryptosystem, but with the higher key length, the longer time is required to process the RSA cryptosystem.

Nowadays, despite the increase in key length, classical RSA is still prone to different attacks as a result of advancements in computing technology and hacking methods, instead of increasing the key length of classical RSA or increasing the primes involve to generate keys, we tried to enhance the classical RSA so as to compensate the classical RSA without increasing the key length (1024). Recently, some cryptography algorithms that were developed gave their best with time but at the detriment of security. This research study introduced a random Bit-insertion Algorithm with modified RSA (RBMRSA) to increase the security strength of the RSA cryptosystem without increasing the key length by converting the ciphertext more complex structure so that the attacker would not decrypt easily even if the attackers knew the secret key, thereby making it more reliable and efficient. The bit insertion is proposed in this research study to effectively overcome the weakness of the classical RSA cryptosystem and achieve high diffusion degree and efficient security against different RSA attacks. The Fermat little theorem is used to test the Primality of the 3 prime numbers needed to generate both encryption and decryption keys because strong prime numbers are needed in RSA security.

The main aim of this study is to design enhanced Bit insertion techniques in modified RSA with specific objectives.

- Identify the strength and weaknesses of the classical RSA cryptosystem.
- Develop enhanced bit insertion in the RSA algorithm (using three prime numbers p, q, r), which can be used to encrypt and decrypt the message in order to prevent common RSA attacks.
- Implement and evaluate the performance of the proposed system.
- Compare the proposed system with classical RSA using Mathematical evaluation proves and other security metrics.

Thorough descriptions of the proposed algorithm and its process were presented in this research work; the authors observed the performance of the new cryptography algorithm (RBMRSA) and compared it with classical RSA. Experimental results showed that RBMRSA outperformed classical RSA with regards to the security levels.

The rest of the paper is structured as follows. In section 2, several authors' past related works on RSA cryptosystems were presented in detail in this section. In section 3, the proposed approach is presented, section 4 states the implementation requirements, and the security analysis and evaluation proof are shown in section 5. Conclusive remarks and future works in section 6.

2. Related work

This section discussed the various results obtained from other previous methods to provide more perspective in the cryptographic field. Different researchers carried out extensive studies in this regard: Babu and Vijayalakshmi [8] designed and implemented an RSA cryptosystem with Fernet cipher encryption. Multi-layer encryption is used to construct a sophisticated and complex approach to encryption. The study enables message delivery through an unsecured network, increasing encryption and decryption time.

Authors in [9] propose an enhanced symmetric key cryptosystem with the use of a public key which makes it difficult to break the original message. The symmetric key system used 512bits as the key size. The proposed method is efficient to encrypt a large quantity of information over the existing ones, which is only effective for successfully transmitting the small quantities of information. Another study from [10] introduces scrambled alphanumeric randomization and RSA algorithm to ensure enhanced encryption in the electronic medical record. The method proposed a secure means of data encryption and decryption by applying the concept of scrambled alphanumeric randomization to provide a clear understanding of the operational mechanism of the RSA algorithm during the decryption and encryption. The unique alphanumerical techniques employ a unique numbering or letter sequence to every letter. The limitation of this approach is that more encryption and decryption time is required due to less processing speed.

Authors in [11] implemented RSA Cryptosystem in Text file Security. The objective of the research work was to develop a new RSA Cryptosystem for the security of text files and then implement it. The limitation of the research is that it does not provide or increase the security of RSA against common RSA attacks.

A study by [12] proposed an efficient cryptographic scheme for text message protection against brute force attacks. The research is carried out to reduce the key's ciphertext size and complexity. The algorithm also requires little effort in encrypting the messages. The techniques might not be effective in resisting some attacks. [13] Developed novel RSA techniques for encryption using one-time encryption keys and unpredictable biosignals for wireless communication devices. The encryption key is reduced in this design with true prime number random number generator (TPRNG) that produces the Primes, which is difficult to know or guessed through Biological-Signal, also replaceable RSA encryption key. The algorithm is implemented through Verilog. The technique is only suitable and applicable in a real-time environment.

A similar study in [14] proposed a method in which the public key is derived as a result of optimization techniques. The optimization performance is evaluated using the peak Signal to Noise Ratio. The method resists statistical and differential attacks alone. Authors in [15] implemented novel encryption to provide efficient security and generate the output in the waveform to not modify and attack the original information. The algorithm resists the timing attacks. Authors in [16] developed RSA cryptography Algorithm using three keys, but the study still requires some security to solve the problem of data upload from the server.

Authors in [17] proposed a framework that provides dual security for RSA cryptosystem but the dual security system is not applicable in all systems because the algorithms for the formation of the key have increased the complexity of the proposed system. [18] also developed an enhanced scheme of RSA cryptosystem. The algorithm utilizes four large prime numbers. These multiple primes make the computational and space complexity of the novel system higher than classical RSA. The public component of n is derived by multiplying two large numbers. The value of Encryption and Decryption is generated by multiplying four large prime numbers. The newly built algorithm resists Brute Forte Attack. And the new approach is much more efficient when compared with Classical RSA. A parallel technique using a new parallel Data Structure known as a concurrent index list of character blocks was proposed by authors [19]. The aim is to fasten the rate of execution of the RSA Cryptosystem, and the system does not take care of the security aspect of the RSA.

Author [20] proposed an Optimized Homomorphic CRT-RSA Algorithm for secure and efficient communication with the use of multiple keys for efficient communication; the security performance was evaluated against classical RSA. The experimental results obtained proved that the proposed algorithm enhances the security and also decreases the involvement of intruders during the communication, but the flaw of this approach consumes more resources than classical RSA. The findings of [21] focus on enhancing the data security in the cloud environment by proposing Quasai modified levy flight distribution for RSA cryptosystem to solve the problem of data security. The secured key generation and the data security are done by RSA cryptosystem thus, data is secured against unauthorized access. The proposed approach [22] introduces an efficient RSA cryptosystem by applying Cuckoo Search Algorithm (CSA) to secure and overcome data integrity problems. CSA is applied to avoid brute forte attacks and to optimize the encryption of the key. The proposed system is faster than classical RSA and provides a better throughput while increasing the private key length. The work of [23] applied double encryption using the AES and RSA; the file is encrypted twice using both RSA and AES. The corresponding keys are generated during the execution of the algorithm; the techniques increase the security level to some extent when compared to classical RSA. A secured Message transfer using RSA algorithm and Improved play fair Cipher in cloud computing is proposed by [24]; the objective of the study is to provide security to the data sent and also secure the key. The proposed system consists of the 3 stages, which include encryption of text using the play fair cipher, the second stage performs XOR operation on the encrypted text, and the final stage is done with RSA algorithm. The proposed algorithm increases the security level of RSA than classical RSA but requires more computational resources. Author [25] introduces estimation and ensuring the security of encrypted data by hybridizing RSA and AES algorithms with third-party confirmation. The system secured authentication and strictly addressed security and privacy by protecting the data from unauthorized access.

Due to different security lapses of RSA such as a brute-forte attack, ciphertext attack, common modulus attacks, timing attacks, and some others, different researchers gave their best in introducing various RSA modification algorithms, and some combined the RSA with other encryption algorithms in order to address aforementioned lapses. But some of the proposed algorithms are still vulnerable to some attacks in one way or the other, while some that are very efficient might require a relatively high computation time, high memory utilization, and consume more computational resources.

2.1. Motivation of study

Based on a critical review of existing works, we identified some drawbacks of RSA, which make RSA prone to some common attacks such as exhaustive search, timing attacks, common modulus attacks, and some gaps in the existing studies. Therefore, the research is at this moment tailored towards using the proposed RBMRSA to improve the security of RSA without using more than three primes and not increasing the key length but at the same time achieving effective security against different attacks.

Thus, this study presents an improved random Bit-insertion Approach with Modified RSA in resisting Attacks in Information Security.

3. Proposed method (RBMRSA)

This section describes the methodology used to secure and protect information which is in two phases. The basic idea behind the first phase consists of RBMRSA encryption operation, which involves a multi-prime and second phase of our proposed method (RBMRSA) as a Bit insertion algorithm. The first phase is the encryption phase which consists of key generation procedure *(e, d)* and encryption procedure. The key generation procedure requires three prime numbers *(p, q, r)* instead of two primes involved in classical RSA to derive both public and private keys used in encrypting and decrypting data, respectively. The second phase of our proposed method (RBMRSA) obtained the output of the first phase, which is encrypted text (Ciphertext), and then converted it into binary format to generate the binary ciphertext. The Binary encrypted message then undergoes a bit stuffing process, the output results of the process are sent to where the message is required. Fig. 1 shows the different stages of RBMRSA.

3.1. Encryption procedure

This is the first step of any encryption process, which comprises a series of procedures involved in generating public keys and Secret Key by the receiver Y. In this framework, modified RSA, the intended receiver Y performs mathematical steps below to compute public and Secret Key. These are some of the definitions involved in encryption operations.

- Definition 1: ***(****Encryption domain):* In any cryptography system, users generate the keys to be used for encrypting and decrypting; thus, let 'T' be the encryption domain with tuples.

$$T = \{p, q, r, e, d, N, \varphi(N)\}$$

Where *(p, q, and r)* are prime numbers [26] a $\mathbb{Z}^+$ prime q ≥ 2 is a prime i*ff* the positive divisor are 1 and *q on:*p

e: the public key used in converting plain messages by the sender.

d: secret key used in decrypting the information and is only opened and revealed to the receiver.

N: the modulus size is the product of p, q, and r.

$\varphi(N)$: the Euler phi function of a given integer N>=2 means the count of integer in the range of 1 to 1-N that is relatively prime to N.

- Definition 2: *(Division Algorithm for Integers)* [26]

Let $f, g \in \mathbb{Z}, \ni g > 1, \exists$ unique $q, r \in \mathbb{Z} \ni f = gq + r, 0 \leq r \leq g$ *if* $r = 0$, *theng*|*f*

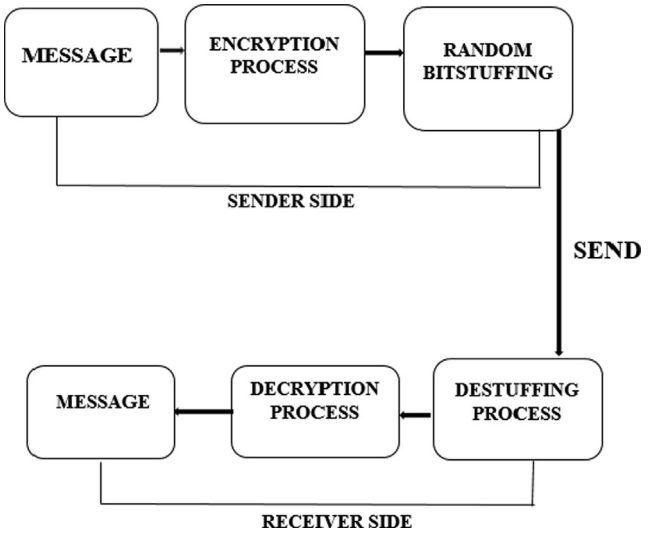

Fig. 1. Flow of RBMRSA Algorithm.

- Definition 3: *(Greatest Common Divisor (G.C.D))* [26]

  Let $f, g \in \mathbb{Z}$. An $\mathbb{Z}^+$ *'d'* is the *G.C.D* of (f, *g) iff d|fandd|g*

- Definition 4: *(Coprime)* [27]

The two integer *f and g* are said to be Coprime if great common divisor $g.c.d(f, g) = 1$

- Definition 5**:** *(Modulusz (N)* [28]

  The integer modulo n represented by Z_n is a set of equivalence classes of integers {0, 1, 2. . .., n-1}. Operations in Z_n are done in modulo of 'n'

- Definition 6*: (Euler Totient function (φ(N))* [29]

  Euler Totient is defined as the count of integer within the range of 1 to N-1 that is relatively prime to N.

$\varphi(N) = \varphi(pqr) = \varphi(p) \times \varphi(q) \times \varphi(r)$

$\varphi(p) = p - 1$ Iff p is a prime likewise $\varphi(q) = q - 1$ and $\varphi(r) = r - 1$. Then $\varphi(N) = (p - 1)(q - 1)(r - 1)$

- Definition 7**:** *(RSA modulus)* [30]

  Let p, q and r be a large prime number $p \neq q \neq r$ *then* RSA modulus $N = pqr$

Theorem 1 (*(Fermat little theorem)*). [31]*: Let p be a prime number, and 'r' is any* $\mathbb{Z} \ni p \nmid r$*, then r and p should be co-prime that is* $g.c.d(r, p) = 1$ *then* $r^{p-1} \equiv 1$ *mod p.*

Also Fermat little theorem can be written as $r^{p-1} - 1 \equiv 0$ *mod p*

Theorem 2 (*(Euler theorem):*). *Let (f, n) be two* $\mathbb{Z}^+ \ni g.c.d(f, g) = 1$*thenf* $^{\varphi(n)} \equiv 1$ *mod n .φ(N) Is the Euler phi function.*

(i) ***Key Generation phase:*** The first important stage in encryption operation is the key generation phase. In the key generation phase, large and unique prime numbers *p, q, and r* that are certain of primarily were obtained through the use of the Fermat little theorem. The key generation phase involved mathematical steps in deriving a public key 'e' and Secret Key 'd' through three unique prime p, q and $rp \neq q \neq r$. The encryption key 'e' generated is used to encrypt the message by the sender X while the Secret Key 'd' for decryption is kept secret by the recipient Y. Fig. 2 shows the procedure of generating bot*h encryption key 'e' and decryption Key 'd' by the user.*

(ii) Decryption Key (d) is reserved confidentially and private, but the encryption key (e) is then distributed to the database for secure communication between the sender and recipient.

- *Encryption Procedure:* The phase uses three parameters as input values, i.e. *(e, M, N), where M is the plain message, and N* is equal to the product of *p, q, and r*. The encryption operation of messages is performed by the sender by the following procedure.

(a) The Sender (X) retrieves the receiver public key 'e' from the database.

(b) Compute

$$E(M) = Ciphertext(CT) = M^e mod N \quad (1)$$

3.2. Bit insertion

This is the second phase of RBMRSA, in which the Encrypted message (*EM))* obtained in *Equation (1)* was converted into binary format.

$$Ciphertext(C) \rightarrow BinaryE(M) \rightarrow BCT \quad (2)$$

The output of *Equation (2)* then undergoes a technique called bit stuffing. Bit stuffing is defined as the addition of noise as extra bits into the information sequence. Inserted bits must not be taken as the overhead bits [32]. Random Bit Stuffing (RB) is recommended in this framework as a reasonable means of enhancing the security of RSA by increasing the randomness of encrypted data Binary encrypted Message and Computational Complexity against ciphertext and exhaustive search attacks. The output result of *Equation (2)* which is in binary representation, then scans for these conditional patterns 101,01 and 1 to be a bit stuffed randomly with the binary strings represented by 'R' of random length before its dissemination through an unsecured network system.

$$BinaryE(M) + RB \rightarrow RandomBitstuffB(EM)) \quad (3)$$

Introduction of random bit component *'RB'* makes the RSA encryption semantically secured, *'RB'* is generated using binary random number generator; therefore, the random *'RB'* makes it harder for the attacker to compute the original message from the

Input: *Prime Number p, q, and r*

Output: *N, Secret key (d) and Public key (e)*

Process:

1: ***Begin***

2: ***Generate*** *p, q, r randomly* $\ni p \neq q \neq r$

3: Test for primality

4: ***If*** *(p != q) and (p != r) and (q != r)* ***do***

$N = p \times q \times r$

Else

go to step 2

5: compute φ (N) = (p-1) (q-1) (r-1)

6: pick 'e' such that 1 < e < φ (N)

7: ***If*** *g.c.d (e, φ (N)) = 1. then*

Compute 'd' such that $d \equiv e^{-1} mod (\varphi(N))$

Return N, d, e

Else
go to step 5

8: ***End If***

9 : ***End If***

10 : ***End***

Fig. 2. Algorithm RBMRSA Key generation procedure.

ciphertext, that is encrypting the same messages twice generate different ciphertext *Random (B(EM))*. Fig. 3 below represents the flow diagram of the bit stuffing process of an encrypted message *to* generate Random Bit-stuffed ciphertext Random (B(EM)). Random (B(EM)) is the final output to be sent through the internet or communication network.

3.3. Sending the message

After the *B(EM)* has been Bit-stuffed to form bit stuffed ciphertext *Random (B(EM))* is now ready to be sent through the internet to the recipient end.

3.4. Receiving phase

This architecture enables the recipient Y to receive Randomized bit stuffed ciphertext *(RB(B(EM))* from the sender X, in which the recipient firstly de-stuffed by removing the additional bits appended in a reversed ways encryption was performed and then converted back to non-binary format and finally decrypted using its secret key generated from RSA algorithm with three prime numbers. The hybridization of the algorithms strictly secures the message against different attacks as a result of an additional level of complexity.

The output of Fig. 4, which is Binary Ciphertext *(BCT)*, then undergoes the decryption Process as follows.

$$BE(M) \rightarrow CT \rightarrow (CT)dmodN \rightarrow PlainMessage \quad (4)$$

3.5. Implementation requirements

*T*he software requirements for the RBMRSA implementations are as follows:

(i) Windows 8 operating system with 5.0 index rating.
(ii) Java programming language was employed because of its flexibility, adaptability and Security. Java Development Kit (JDK 7), Net Beans 8.0.1
(iii) Microsoft Office Excel 2010 was used for the plotting of graphs.
(iv) Crammer's online software for generating linear solution to polynomial regression equation.

The minimum hardware requirements RBMRSA have the following hardware configurations:
Speed: Intel Dual core processor of 2.67 GHz
Memory: 4 GB of RAM
Hard disk: 500 GB Capacity

4. Security and performance evaluation analysis

This section will examine the security and evaluate the performance of the proposed RBMRSA scheme.

The result generated using the RBMRSA algorithm in this research work will be evaluated against similar schemes. However,

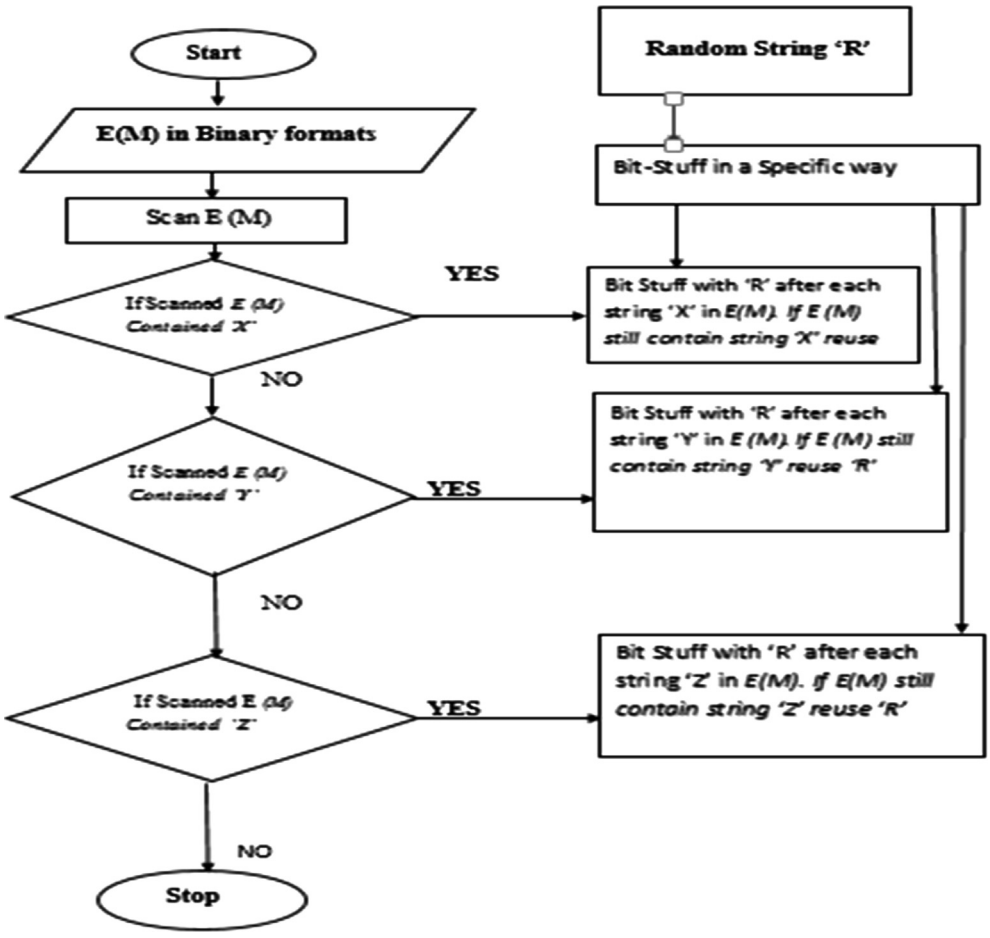

Fig. 3. Random Bit stuffed ciphertext RB(B(EM)) flow diagram.

Input: Random Bit stuff (BCT)

Output: E(M)

BCT= Bit-Stuffed Cipher text in binary

E(M) = Encrypted Message

Process

1: ***Begin***

2 : X←101, Y←01 and Z←1

3: Search BCT

4: ***For*** *i←1 to length of BCT*

5: ***If*** *Pattern X is proper subset of BCT* ***then***

6: delete random 'R' after each pattern X contained in BCT

7: ***Else***

8: ***If*** *pattern Y is proper subset of BCT* ***then***

9: delete random 'R' after each pattern Y in BCT

10 ***Else***

11: ***If*** *pattern Z is proper subset of BCT* ***then***

12: delete random 'R' after each pattern Z in BCT

13: ***End If***

14: ***End If***

15 ***End If***

16: ***End for***

17: Return E(M)

18: ***End***

Fig. 4. Algorithm of RBMRSA De-Stuffing.

explanations of those analysis metrics used in the evaluation before analyzing and evaluating this research work are essential.

4.1. Security analysis

RBMRSA resists attacks described below through the evidence and proof. The classical RSA is not secured against Common Modulus Attack *(C.M.A.)*.

(i) ***Common Modulus Attack (C.M.A.):*** This can retrieve any plaintext encrypted with two different public keys e_1 and e_2 of the same Modulus N Let $N = pq$ and (e_1, N), (e_2, N) are two public keys *then g.c.d* $(e_1, e_2) = 1$

Suppose Message 'M' is encrypted with both keys e_1 and e_2 *(EM)* $1 = M\ e1 \bmod N$ *and (EM) 2* = M e2 mod *N, then* Message *'M'* can be generated.

Proof. I. *f* e_1 *and* e_2*, are known, then obtain two* $\mathbb{Z}^+ f, gf.e_1 + g.e_2 \equiv 1$. *Therefore Calculate*

$((EM)_1)^f.(E(M)_2)^g \equiv M_1^{\{f.e} M_2^{\{g.e} \equiv M_1^{(fe} {}_2^{ge}$ *Recall that* $(f.e_1 + g.e_2) \equiv 1$, then $M_1^{(fe}{}_{2)}^{ge} \equiv M^1 \equiv M$

It indicates that if the attacker obtains e_1 and e_2 and the equivalent ciphertexts $(EM)_1$ *and* $(EM)_2$ the attacker would able to capture all the plaintext 'M' encrypted two times to various users.

Theorem 3. *In the RBMRSA security scheme, Common modulus attack (C.M.A.) cannot retrieve any the Plaintext encrypted with the two different public keys* e_1 *and* e_2 *of the same modulus N In RBMRSA, the attacker cannot get the message 'M' because* BCT_1 *is not the real* $(EM)_1$ *likewise* BCT_2.

$BCT_1 = (EM)_1 + R$ and $BCT_2 = (EM)_2 + R$.

Proof. Quite obtaining e_1 and e_2, find two $\mathbb{Z}^+ f, gf.e_1 + g.e_2 \equiv 1$
Calculate:

$$\left((B.E.)_1\right)^f (BE)_{2)\ g} \equiv (E)_1 + R^f\left((E)_1 + R\right)^g$$
$$\equiv (\ mod\ + R)^f (\ mod\ + R)^g \quad (5)$$

Using binomial theorem to solve Equation (5), also recall that $(f\ e\ 1 + g.e_2) \equiv 1$.

The attacker finds *(B.M.),* which is not the actual message and not BCT_1 and BCT_2. Therefore RBMRSA is secured against *C.M.A.*

(ii) Timing attack (T.A.)

Reference [33] observed that it is likely to derive decryption key'd' without factoring modulus *N* through probability knowledge. In RBMRSA, attackers still need to compute a binary random component S for bit stuffing to decrypt the *(E.M.).* Knowing the decryption key'd' would not permit the attacker to decrypt the plaintext successfully without R. Because 'R' is appended to the encrypted message randomly, *BCT* depends on the value of 'R', encrypting the same message do not generate the same *(B.E.M.).*

$$LetBCT_1 = (EM)_1 + R_1 \quad (6)$$

$$BCT_2 = (EM)_1 + R_2 \quad (7)$$

Since binary strings 'R' is generated randomly, which implies that $R_1 \neq R_2$, then eqn. (6) produced the output that is different from equation *(7)* output.

(iii) Known plain text attack (K.P.A.)

The conventional RSA is not secure against Known Plain Text attack, while RBMRSA resists the attack. K.P.A. has various plaintexts with its cipher texts version. The attacker exploits Known Plaintext to obtain a newly captured plaintext that is encrypted using a similar key.

Proof: Let plain text $P_1, P_2, P_3 \ldots\ldots P_n$ and the corresponding *(E.M.) as* $(EM)_1, (EM)_2, (EM)_3 \ldots\ldots\ldots (E.M.)_n$. *Build a set W=* $(P_1, (EM)_1)$ $(P_2, (EM)_2)$ $(P_3, (EM_3)) \ldots\ldots (P_n\ (E.M.)_n)$ Where Pn ∈ *P* and $(E.M.)_n \in$ *(E.M.).* In RSA, encrypting similar Plaintext severally using similar key generates similar cipher texts, built on prebuilt set *W*, the attacker can use earlier taken Plaintext to newly generate the plain text P_{n+1} , provided the equivalent $(E.M.)_{n+1}$ is contained in the set.

In RBMRSA, random bit stuffed into the encrypted message makes the final encrypted message *BCT* of the same message different from each other.

Proof: Given plain text $P_1, P_2, P_3 \ldots\ldots P_n$ and the corresponding ciphertext *BCT* as $(E.M.)_1 + R_1, (E.M.)_2 + R_2, (E.M.)_3 + R_3 \ldots\ldots\ldots (E.M.)_n + R_n$.

Where $R_1 \neq R_2 \neq R_3 \ldots\ldots\ldots \neq R_n$ since each *(E.M.)* is a bit stuffed with the different random binary string *'R'*, the attacker cannot build a set of plain text with the corresponding BCT.

4.2. Security and performance evaluation metrics

(i) ***Execution time:*** The execution time of any cryptographic algorithm determines the performance of any particular encryption and decryption technique. The execution time defines how fast or slows the algorithm

The encryption and decryption results of the proposed Algorithm RBMRSA and Classical RSA are shown in Table 1 and Table 2, respectively. From the results in Tables 1 and 2, it is clearly seen that the computational complexities of both encryption and decryption algorithm of RBMRSA are higher than that of classical RSA, which indicates the algorithm will be much complex and require much time for the attackers to break easily than that of the classical RSA.

Table 1
Encryption and Decryption time (Millisecond) of RBMRSA with different sizes (byte).

Message Size	Encryption time	Decryption time
20	4.25	40.60
25	3.33	43.00
37	2.75	26.58
45	3.86	20.05
50	3.60	18.00
55	3.70	30.03
60	3.49	17.01

Table 2
RSA Encryption and Decryption time (Millisecond).

Message Size	Encryption time	Decryption time
20	3.0012	4.5123
25	3.1157	8.9618
37	3.3247	10.6470
45	3.3969	11.9962
50	3.4630	12.9991
55	3.4797	13.4335
60	3.5100	14.9783

The graphical representations of encryption and decryption time of proposed Algorithm RBMRSA and Classical RSA when plotted against the message sizes in bytes are shown in Fig. 5 and Fig. 6, respectively. From the results in Figs. 5 and 6, it is seen that the attacker can never or less predict the decryption time of the proposed Algorithm RBMRSA than that of classical RSA This is due to randomized bit insertion, which served as double layer encryption technique that prevents the system from timing attacks.

The graphical comparison of encryption time of proposed Algorithm RBMRSA and Classical RSA when plotted against the message sizes in bytes are shown in Fig. 7. Fig. 8 also the graphical comparison of decryption time of proposed Algorithm RBMRSA and Classical RSA when plotted against the message sizes in bytes.

(ii) ***Avalanche effect*:** The amount of variation that occurs in ciphertext by slight change or variation in Plaintext is the avalanche effect [34]. A good cipher or encryption algorithm must generate completely different output for a minimal change in the input. The degree of changes in the ciphertext is proportional to the level of security of an algorithm, i.e., the higher the avalanche effect of an algorithm, the higher the security level of such an algorithm. The aim behind flipping only a single bit in avalanche effect (%) is to test the sensitivity of the proposed algorithm by examining if the very slightest change in the plaintexts would completely produce a radical or huge change in the ciphertext, so it will be difficult for an attacker to perform statistical analysis on the ciphertext.

$$\%Avalancheeffect(plaintext) = \frac{Countofchangedbitsinciphertext}{Totalbitsinciphertext} \times 100 \quad (8)$$

The outcome of flipping one bit in 1 K.B. plaintext of RBMRSA generated a 47.52% change in the ciphertext and that of RSA was estimated to be 0.2%. In which the value in % represents the avalanche effect. Fig. 9 indicates a graphical chart of the avalanche effect measured in percentage.

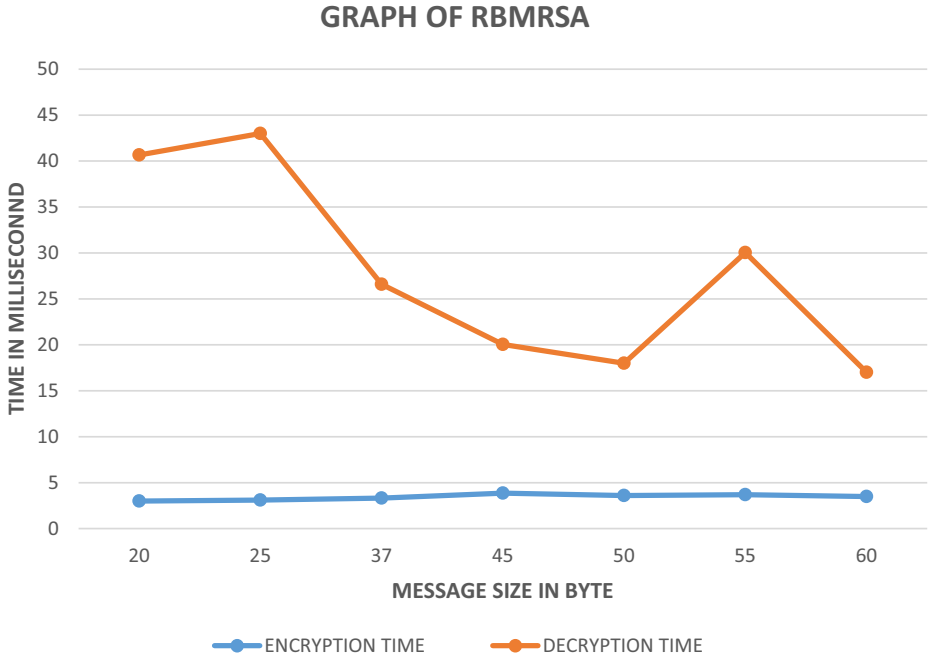

Fig. 5. Graph showing the Encryption and decryption time of RBMRSA.

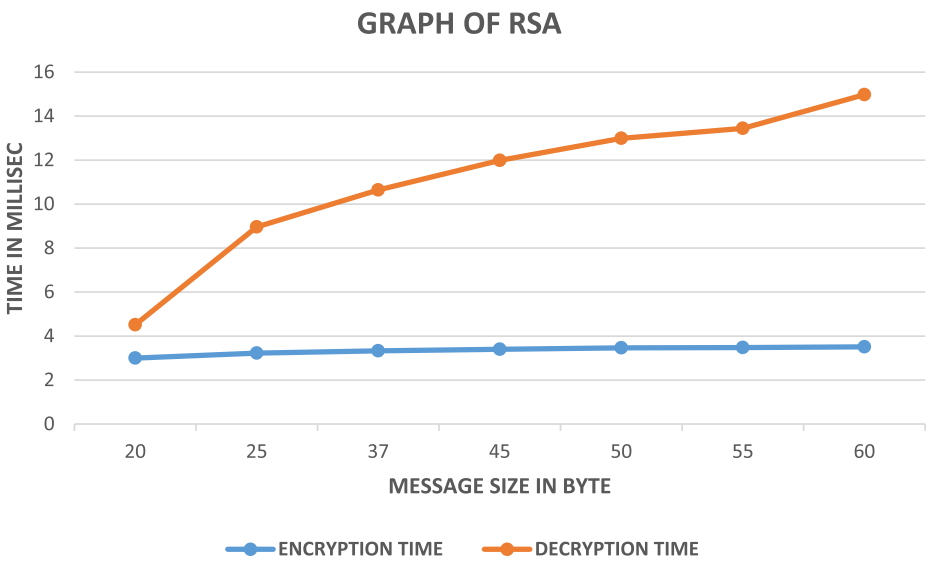

Fig. 6. Graph showing the Encryption and decryption time of RSA.

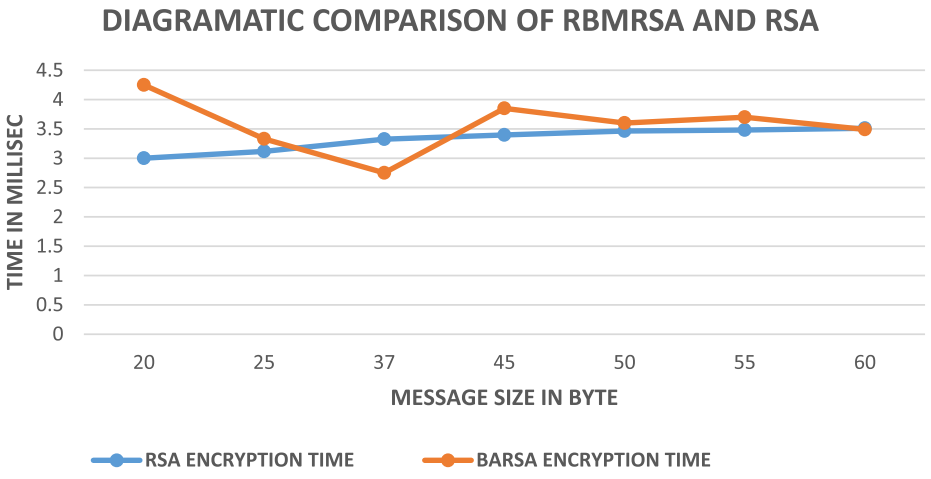

Fig. 7. Graph showing the comparison in Encryption time of RBMRSA and RSA.

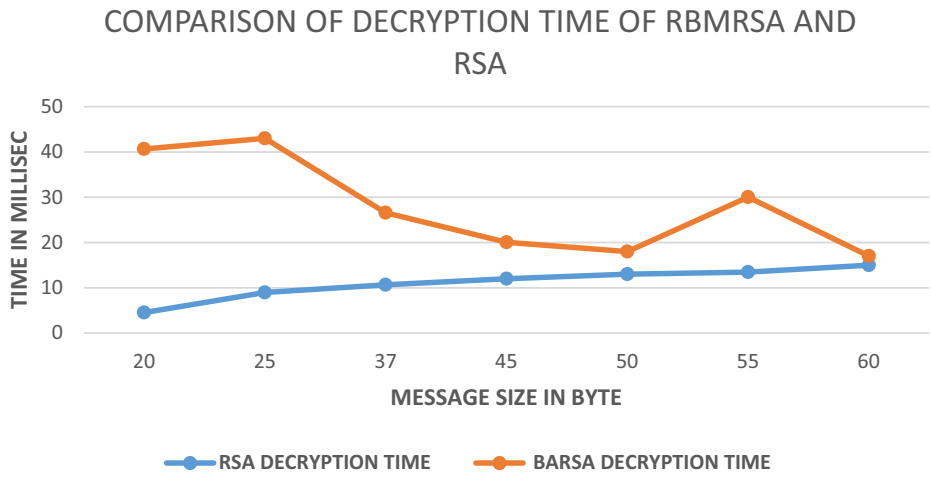

Fig. 8. Graph showing the comparison in decryption time of RBMRSA and RSA.

(iii) ***Computational Complexity:*** Major parameter that determines the effectiveness of a cryptographic system is the time taken by the computer to generate a solution to a problem through a specific algorithm with the specified input. With the use of Big O notation to conduct analysis, we can observe the degree of complexity of the algorithm.

The Equation connecting X and Y can be obtained in Table 1 with the use of polynomial curve equation [35]

$Y = F(X) = ax^6 + bx^5 + cx^4 + dx^3 + ex^2 + fx + g$ to find the represented equation of complexity of encryption algorithm of RBMRSA.

$$y = -2.97E - 8x^6 + 6.1E - 6x^5 - 4.9E - 4x^4 + 1.8E - 2x^3 - 2.8\ E - 1x^2 + 0.7x + 21 \quad (9)$$

Equation (9) denotes the mathematical model of RBMRSA Encryption, where (x) is the message size measured in bytes and (y) represents the time involved to generate ciphertext, which is measured in a millisecond.

$$Z = -7.4E - 7x^6 + 1.4E - 4x^5 - 1.0E - 2x^4 + 3.4E - 1.x^3 - 4.7x^2 + 10x + 265 \quad (10)$$

Equation (10) denotes the mathematical model of RBMRSA Decryption. (x) Signifies the message size measured in byte and (z) represents the decryption time (millisecond). The RBMRSA was evaluated by computing its complexity with Big-O analysis through sets of message inputs. The Equations show that the best fitting curve can be achieved for Big O encryption within the degree range of 3 and degree 3 for decryption. Hence, RBMRSA and RSA were executed under the same condition.

$$y = 2.2E - 9x^6 - 4.4E - 7x^5 + 3.4E - 5x^4 - 1.2E - 3x^3 + 1.9E - 2x^2 - 3.7\ E - 2x + 1.8 \quad (11)$$

Equation (11) denotes the mathematical model for encrypting RSA, where (x) is the message size measured in byte and (y) represents the time involved to generate ciphertext, which is measured in Millisecond. Of RSA

$$Z = -1.6E - 11x^6 + 1.8E - 6x^5 - 3.9E - 4x^4 + 3.2E - 2x^3 - 1.318x^2 + 26.3x - 198 \quad (12)$$

The equation (12) is the mathematical model of RSA Decryption,

Equation (11) shows that the order of polynomial for RSA encryption is O(n^2), and Equation (12) shows that the order is decryption is O(n^3).

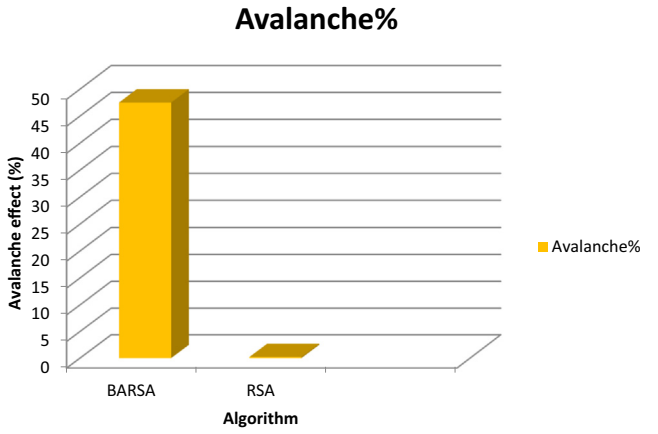

Fig. 9. Screenshots showing the avalanche effect (%) of RBMRSA and RSA.

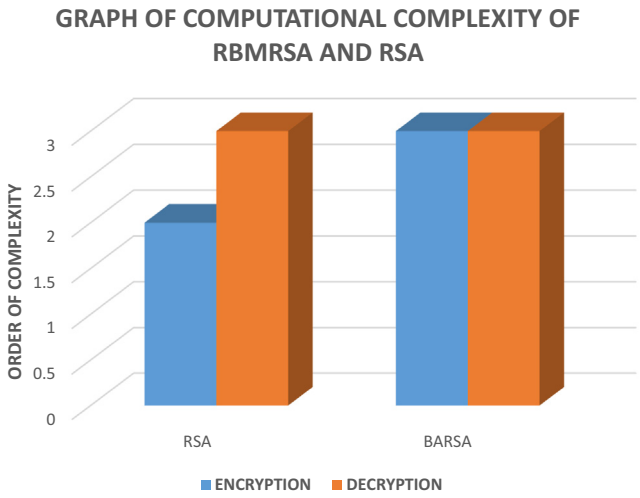

Fig. 10. Comparison graph of the complexity of RBMRSA and RSA.

Analyzing both RBMRSA parameters has clearly shown that the algorithm's speed depends on the execution time, and these relationships have been converted into equations.

Fig. 10 *represents the graphical comparison of the order of complexity of both encryption and decryption of proposed Algorithm RBMRSA and Classical RSA. The results indicated that the order of complexity RBMRSA is larger than that of RSA, which infers better security at the execution time cost.*

(iv) ***Attack steps analysis:*** This is the steps involved to execute the best-known attack; the steps analysis aids in computing the time required for a successful attack using a specific processor without running this attack on the algorithm.

Attack Steps = 2^{l} where l is the key length

RBMRSA is the attack steps for l = (1024) bits, and N is the additional bits due to bit-stuffing. The attack steps = $2^{(1024)}$ + N Steps = 1.7978×10^{308} + N Steps where N > 0

RSA, the attack steps for p = 1024bits, the attack steps is 2^{1024} Steps = 1.7978×10^{308} Steps

Table 3 *shows the different results generated during the experimental comparison of the proposed Algorithm RBMRSA with classical RSA using different security evaluation metrics.*

4.3. Discussion of results

The RBMRSA and Classical RSA were analyzed by measuring the computational complexity of RBMRSA and Classical RSA through the Big-O test with the series of test inputs (bytes); the results for RBMRSA encryption and decryption shows that the values of X appear highest in the range of degree 3 for Big O encryption O (n^3) and highest in the range of degree 3 for Big O Decryption O (n^3). The Big O notation for Classical RSA encryption is O (n^2), and the Big O notation classical RSA decryption is O (n^3) which indicates the complexity.

The high order of complexity of RBMRSA than RSA is as a result of high encryption and decryption time of RBMRSA which is due to the fact that RBMRSA depend not only on the size of the messages but majorly on the quantity STRING *X,Y,Z* patterns the length of binary Ciphertext *(BCT)* contained if length *BCT* contains more *X, Y,Z* pattern, the *BCT* attracts more extra Random bits which directly increases the computational time of RBMRSA. Therefore necessitate a possible variation in Encryption and Decryption time of the same text messages encrypted twice. In classical RSA, the

Table 3
Comparisons of RBMRSA, RSA using Different Metrics.

Encryption Algorithm	*Computational Complexity Encryption*	*Computational Complexity Decryption*	Avalanche effect%	Attack Steps
RBMRSA	$O(n^3)$	$O(n^3)$	45.82%	$2^{(1024)}$ + N
Classical RSA	$O(n^2)$	$O(n^3)$	0.20%	2^{1024}

Table 4
Differences in RBMRSA and Classical RSA.

S/N	Classical RSA	HE-CRT-RSA	RBMRSA
1	Once the attacker knows the secret key 'd', decryption made easy	Multiple secret keys makes it difficult for attackers, it will require the attackers to know the at least two secret key to hack	The proposed RBMRSA Algorithm provides more complex structures as a result of extra bits, even the attacker with the secret key 'd' cannot decrypt easily.
2	It is vulnerable to Common modulus attacks, timing attacks, known-plaintext attacks, as shown in section 5.1	It is less vulnerable to Common modulus attacks, timing attacks, known-plaintext attacks	RBMRSA resists Common modulus attacks, timing attacks, known-plaintext attacks, as shown in section 5.1
3	The time required to Brute forte lesser than RBMRSA requires two prime numbers.	It is less vulnerable to brute forte attacks	The time required to Brute forte is greater than Classical RSA. RBMRSA requires three prime numbers.
4	The computational time is lesser than the proposed RBMRSA.	It requires higher computational time when compared to classical RSA and RBMRSA	The computational time is high compared with Classical RSA.
5	Classical RSA algorithm requires lesser CPU usage when compared with RBMRSA.	HE-CRT-RSA algorithm requires high CPU usage when compared with RBMRSA	The algorithm computation could be CPU intensive than that of Classical RSA.

time complexity solely depends on the message sizes, which means encrypting the same message twice will generate the same ciphertext. Hence, the security of RBMRSA is a result of high computational difficulty and complexity. Section 4.1 of this paper has also compared the classical RSA and proposed RBMRSA using mathematical proofs for security evaluation.

HE-CRT-RSA [20] utilises multiple public and private keys for both encryption and decryption, respectively, thus consuming more computational resources compared with Classical RSA and RBMRSA. Table 4 shows the comparison between Classical RSA, HE-CRT-RSA, and RBMRSA based on their strengths and weaknesses.

5. Conclusion and future works

In this work, RBMRSA, a new security approach to messages, was developed. This approach was implemented using the Java programming language. The complexity measure of the algorithm helps in the performance evaluation of RBMRSA for complexity; the results showed that the computational complexity of the RBMRSA technique is higher than the RSA algorithm due to the execution time incurred, which corroborates the fact that there is always a tradeoff between security and execution time. The measures of sensitivity of the algorithm to a slight change in Plaintext (avalanche effect) were conducted. The result proves that the amount of bit varied in the ciphertext to a total number of bits in ciphertext in (%) of RBMRSA is 46%, RSA is 0.20%. Hence, the greater the avalanche, the higher the security level, which infers that the RBMRSA algorithm is sensitive to a bit of change in Plaintext by producing significant changes in the ciphertext.

The proposed RBMRSA method is appropriate for environments that require a high level of security but at the expense of execution time. The RBMRSA of 1024bits is compared with 1024 bits of RSA; mathematical security analysis also shows that RBMRSA resists some attacks, which the RSA algorithm is prone to. The overall results showed that RBMRSA is a strong tool for securing the transmission of private and confidential messages over a vulnerable network.

The security provided by RBMRSA can be extended further to encrypt image files with additional metrics, including attack analysis, correlation analysis, and histogram analysis. Future work should be carried out to accommodate a scheme to speed up the implemented algorithm in order further to reduce the execution time of the RBMRSA System.

References

[1] Seo JH. Efficient of Data Signature from RSA without random Oracles. Inform Sci New York 2020;512:471–80.
[2] Thiyagarajan R, Meenakshi Priya B. Enhanced of EAACK using P2P ACK and RSA Public Key Cryptography. Measurements 2019;136:116–21.
[3] Thangavel M, Varalakshmi P, Murrali M, Nithya K. An Enhanced and Secure RSA Key Generation Scheme. J Inform Security Application 2015;20(4):3–10.
[4] William S, Stalling W. Cryptography and network security principle and practice. 4th ed. Pearson Education India; 2015. p. 591–661.
[5] RamaChandraRao GAV, Lakshmi PV, Ravi Shankar N. RSA Public Key Cryptosystem using Modular Multiplication. Int J Comput Applications 2013;80(5):38–42.
[6] Meng X, Zheng X. Cryptanalysis of RSA with a small parameters revisited. Math Comput Inf Process Lett 2015;115(11):858–62.
[7] Sharma K, Agrawal A, Pandey D, Khan RA, Dinkar SK. RSA based encryption approach for preserving confidentiality of big data. J King Saud Univ - Comput Inform Sci 2019. doi: https://doi.org/10.1016/j.jksuci.2019.10.006.
[8] D.S. Babu, Y. Vijayalakshmi, Enhancement of E-commerce security through asymmetric key Algorithm, Computer Communication Elsevier, vol.153, pp. 125-134, 2020.
[9] Pandey K, Rangari KV, Shina K. An Enhanced Symmetric Key Cryptosystem Algorithm to Improve Data Security. Int J Comput Application 2013;74 (20):29–33.
[10] Osamor VC, Edosomwan I. Employing Scrambled Alpha-numeric Randomization and RSA Algorithm to Ensure Enhanced Encryption in Electronic Medical Records. Informat Med Unlock 2021;25:1–10.
[11] Tanmado Sihotang H, Efendi S. Designed and Implementation of RSA Cryptography Algorithm. J Phys Conf Ser 2020;1641(2020):1–8.
[12] Joshi A, Wazid M, Goudar RH. An Efficient Cryptographic Scheme for Text Message Protection against Brute force and Cryptanalytic Attacks. Procedia Comput Sci 2015;48:360–6.
[13] Yu H, Kim Y. New RSA Encryption Mechanism Using One Time Encryption Keys and Unpredictable Bio Signal for Wireless Communication Devices. Electron MDPI 2020;9(246):1–10.
[14] Shankar K. An Optimal RSA Encryption Algorithm for Secret Images. Int J Pure Appl Math 2018;118(2):2491–500.
[15] Bangera KN, Reddy NVS, Paddambail Y, Shivaprasad G. In: Multilayer Security using RSA Cryptography and dual audio Stenography. Information and Communication Technology (RTEICT); 2017. p. 492–5.
[16] Stergio C, Kim KE, Gupta BG. Secure an Integration of IoT and Cloud Computing. Future Gener Comput Syst 2018;78(6):964–75.
[17] Abdulshaheed HR, Binti SA, Sadiq II. Proposed Smart Solution Based on Cloud Computing and Wireless Sensing. Int J Pure Appl Math 2018, 2018,:119 (18):427–49.

[18] Thangavel M, Varalakshmi P, Murrali M, Nithy K. An Enhanced and Secure RSA Key Generation Scheme. J Inform Security Application 2015, 2015,:20(4):3–10.
[19] Rawat A, Sehgal K, Tiwari A, Sharma A, Joshi A. A Novel Accelerated Implementation of RSA Using Parallel Processing. J Discr Math Sci Cryptogr 2019;22(2):309–22.
[20] R. Abid, C. Iwendi, A. Javed, M. Rizwan and Z. Jaid "An Optimized Homomorphic CRT-RSA Algorithm for Secure and Efficient Communication" Personal and Ubiquitous Computing Springer, 2021. http:/doi.org/10.1007/500779-021607-3,
[21] Kaliyamoorthy P, Ramalingam AC. QMLFD Based RSA cryptosystem for Enhancing Data security in the public cloud system. Wireless Personal Communication, Springer 2022;122:752–82.
[22] R. Shree, C. Chelvan and M .Rajesh "An Efficient RSA Cryptosystem by Applying Cuckoo Search Optimization Techniques," Concurrency and Computation: Practice and experience. Wiley Online Library, vol. 31, no. 12. http:/doi.org/10.1002/cpe.4845, 2019.
[23] K. Jaspin, S. Selva, S. Sahana and G. Thamnas "Efficient and Secured File Transfer in Cloud through Double Encryption using AES and RSA Algorithm." International of Emerging Smart Computing and Informatics. IEEE, pp. 791-796, 2021. doi:/10.1109/ESCI50559.2021.9397005
[24] P. Hemanth, N. Raj and N. Yadva "A Secured Message Transfer Using RSA Algorithm an Improved Playfair Cipher in Cloud Computing. "International Conference of Convergence in Technology I2CT" IEEE, pp.931-936, 2017. doi:/10.1109/I2CT.2017.8226265.
[25] S. Almanaun, M. Mahmood and M. Amin "Ensuring the Security of Encrypted Information with Hybrid of AES and RSA Algorithm with the Third Party Confirmation," 5th International Conference of Intelligent Computing and Control System, IEEE, pp.337-343, 2021. doi:/10.1109/ICICCS51141.2021.9432174.
[26] Kak A. Computer and Network Security: Public Key Cryptography and RSA Algorithm. Lecture Notes: Prudue University; 2017. p. 1–65.
[27] M. Calderbank "RSA Cryptosystem: History, Algorithm, Prime" Department of Mathematics, University of Chicago. http//www.math.uchicago.edu. August 2007
[28] A. Nitaj. "The Mathematical Cryptography of RSA Cryptosystem". University of Caen, France Laboratory of Mathematics Nicolas Oresme, pp.1-31. http://www.math.unicaen.fr/ nitaj. 2005
[29] Ford K. The Number of Solution of $\phi(x) = m$. Ann Math 1999;150(1):283–311.
[30] Akchiche O, Khadir O. Factorizing RSA Modulus When a Message is Closed to a Multiple of the Prime. Int J Comput Math Taylor and Francis 2018;3(3):196–203.
[31] Robinson J. Fermat little theorem: Elaboration on History of Fermat Theorem and Implication of Euler's Generalization by Means of the Totient Theorem. In: Mathematics department. University of Arizona; 2011. p. 1–32.
[32] Aviran S, Paul H, Jack KW. An Improvement to Bit Stuffing Algorithm. IEEE Trans Inform Theory 2005;51(8):2885–91.
[33] Kocher P. Timing Attacks on Implementations of Deffie-Hellman, RSA, D.S.S. and other Systems in Advances in Cryptology. CRYPTO Springer 1996;1996:104–13.
[34] A. Krishna. "Performance Evaluation of New Encryption Algorithm with Emphasis on Probabilistic Encryption and Time Stamp in Network Securing," M.Sc. Thesis, Department of Computer Science and Engineering, Acharya University of Engineering and Technology. pp. 1-153. August, 2009.
[35] P. Danziger. "Big O Notation: Comparing Algorithm," Department of Computer Science Ryerson University, pp. 1-5, 2010. http/www.scs.ryerson.cal/mth607/handouts/bigO.pdf.