Int. j. inf. tecnol.
https://doi.org/10.1007/s41870-022-00993-y

ORIGINAL RESEARCH

An Effective and enhanced RSA based Public Key Encryption Scheme (XRSA)

Raza Imam[1] · Faisal Anwer[1] · Mohammad Nadeem[1]

Received: 3 April 2022 / Accepted: 2 May 2022

Abstract Cryptography is the mechanism of providing significant security services such as confidentiality, authentication, and integrity. These services are essentially more important in the current era, where lots of IoT devices are generating data and being stored in the cloud environment. Public key cryptography is highly effective in achieving confidentiality and authentication services. One of the popular public-key cryptography schemes is the RSA algorithm, which besides other concepts uses two very large prime integers. In the proposed study, the authors enhanced the RSA algorithm in the context of generation of a more complex key pair, i.e., public and private key, so that adversary should never be able to determine the private key using public-key. The proposed scheme uses four random large prime numbers to generate public–private key pairs and applies XOR operation along with the more complex intermediate process in key-generation encryption and decryption phases to achieve higher algorithm complexity, which would require more time to break the proposed cipher and would make it extremely difficult for third-parties to attack, hence boosting security. The method is also compared with other RSA-based algorithms to demonstrate the potency of the proposed algorithm in terms of enhanced performance and security.

✉ Faisal Anwer
faisalanwer.cs@amu.ac.in

Raza Imam
razaimam45@gmail.com

Mohammad Nadeem
mnadeem.cs@amu.ac.in

1 Department of Computer Science, Aligarh Muslim University, AMU Campus, Aligarh 202002, India

Keywords Algorithm · Asymmetric key cryptography · Information security · Public key cryptosystem · Encryption

1 Introduction

Information and communication technology has introduced the concept of a global village, where every individual can access and share information from any part of the world. On one hand, it provides enormous benefits and at the same time poses many serious threats including data theft, online transaction fraud, cheating scam, and others. Information security comprises several properties such as confidentiality, authentication, integrity, and others to tackle these concerns [1]. Cryptography is the most important Mathematical concept that provides these security services.

Cryptography is the study of strategies for secure communication that only allows the information to be visible to the sending and receiving parties. Cryptography includes algorithms including keys that transform information or text into an unreadable form known as cipher-text. The ciphers can only be converted to readable text using the algorithm and keys. The process or function of transforming regular plaintext into cipher-text is referred as encryption, whereas the process of transforming back the cipher-text into plain-text is referred to as decryption process. Cryptography techniques are used to achieve certain key security requirements, and if a cryptosystem is not able to achieve these requirements, it can't be said a reliable system [2, 3]. These requirements are as follows:

- Confidentiality—It ensures that the information can only be accessed by the intended users, no third party other than these users can fetch that information.

- Integrity—It confirms that data received by the receiver is unaltered and without tampering.
- Authentication—It ensures sender's and receiver's identities and further assures that the information origin and destination are known or identified.

Any cryptographic framework is generally represented by three fundamental dimensions, that is, the activities associated with the change of plain content to encoded one, the quantity of keys involvement, and last one is the technique for handling the plain content [1]. On the ground of these dimensions, Cryptography is predominantly distinguished into two categories—Symmetric Key Cryptography, and Asymmetric/Public Key Cryptography, which are described as follows:

Symmetric Key Cryptography—In this scheme, both encryption as well as decryption processes, use only a single key, where the sender and receiver must maintain the security while sharing the algorithm and the key. Some examples include AES, DES, Triple DES, Blowfish, etc.

Asymmetric or Public Key Cryptography—Unlike Symmetric scheme, asymmetric key cryptography uses "two" functionally associated keys, public-key and private-key. In the encryption process, public key is used, whereas private key is used in decrypting the encrypted text. Public key is announced and thus known to everyone, but the Private Key is only known to the recipient of the data. One of the most recognized and efficient public Key encryption algorithms is RSA [4]. Some other examples include ElGamal, and Elliptic Curve Cryptography [5, 6].

RSA (Rivest, Shamir, Adleman) named on its creators, is a public key cryptography encryption scheme and is one of the most efficient public key encryption schemes present in today's world and is widely used for data encryption over the Web, but still considered to have some drawbacks, like relatively higher computational cost in comparison to other popular encryption schemes, vulnerable to chosen ciphertext-attack, timing attack, factoring problem and so on [7].

1.1 Trend

RSA has been extensively applied in several domains since its inception in 1978. The authors have systematically searched research articles related to the application of standard or modified RSA algorithm and its different variants and found around 84 related articles [8]. These articles are then categorized in several domains (or categories) such as cloud security [9–12], image cryptography [13, 14], wireless security [15], and others [16–18]. Lightweight cryptosystems using hybrid RSA variants have also been proposed in recent years, which eventually, showing effective results in smart gadgets and IOT devices [19–22] Fig. 1 depicts the progression of all RSA variants by classifying related publications in different categories on a yearly basis, where the starting year of this classification is taken as 1978 which is its introductory year [8]. Moreover, RSA applications are not only restricted to key exchanges, digital signatures [23], emails, VPNs (Virtual Private Networks), and web browsers, but it is quite useful in distributed networking, privacy-preserving systems [24], and Peer-to-Peer networks as well

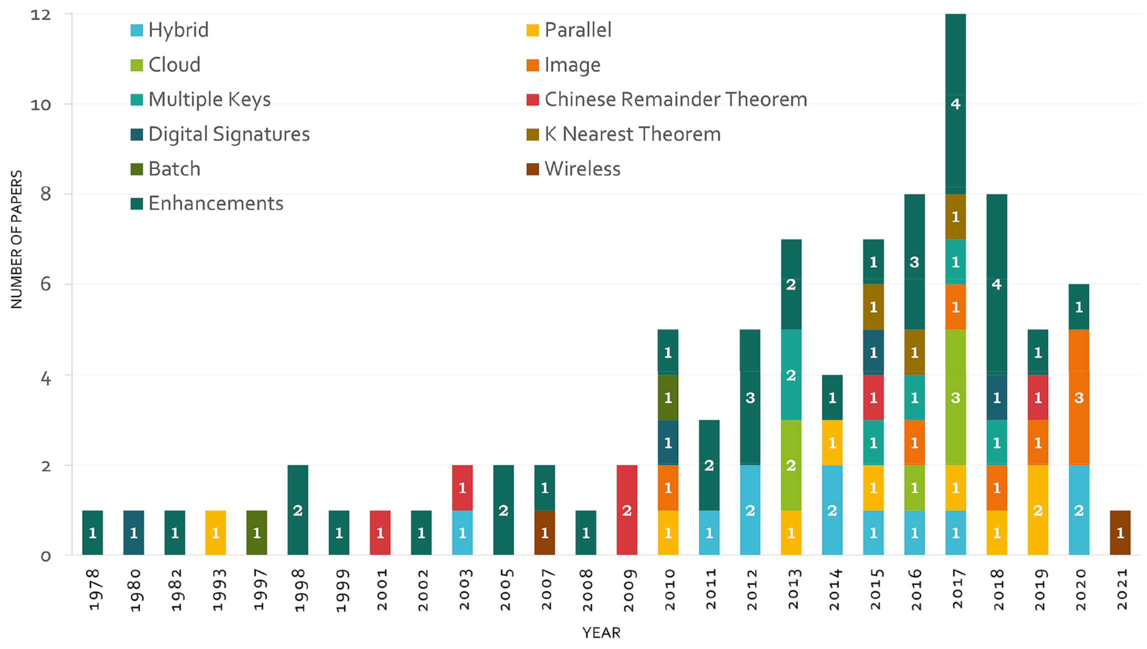

Fig. 1 Yearly trend of all modified variants related to RSA algorithm

[25]. A study of 2016 by Vijaykumar et al. [26] focused on achieving a secure group communication in Peer-to-Peer distributed network via an efficient group key management methodology, where the group key is computed using the Chinese Remainder Theorem (CRT) approach. Their work is able to achieve the objective by proposing a RSA-based method RSAGKM, that reduces computation complexity without increasing the storage complexity in P2P communications, which is also the most prominent characteristic of this study.

1.2 Contribution

In the history of RSA cryptosystem, many individuals worked on RSA system to make it more secure. This paper focuses on more secure and efficient enhancement of RSA. Our approach also involves the assessment and discussion of several modified RSA enhancements to know about the performance and security in comparison to the standard RSA and our proposed RSA. In our proposed XRSA study, the authors aim to enhance the standard RSA algorithm by incorporating mathematical modifications to achieve higher complexity than in standard RSA. XRSA algorithm uses 4 random primes instead of 2, and, the computation of public key exponent E is not directly calculated, rather additional variables are calculated, like E1, E2 and subsequently E'. Our proposal achieves better performance in the context of 4 large random primes when compared to other similar RSA variants in terms of key generation, encryption, and decryption phases. Moreover, in terms of Brute force attack, which is presented in Sect. 5, XRSA aims to achieve much higher attacking time to break the cipher than other variants to prove the algorithm secure against such attacks.

1.3 Organization

The remainder of this paper is organized in the following manner: Sect. 2 focuses on the Standard RSA and related enhancements in the RSA cryptosystems highlighting the recent adapted approaches and their shortcomings. The authors then suggest the proposed method along with a relevant example and a flowchart in Sect. 3. Section 4 highlights the mathematical proof of proposed method while Sect. 5 focuses on the implementation, analysis and discussion of the method in terms of performance and security. Finally, Sect. 6 concludes the paper along with future direction of work.

2 State of the art

RSA algorithm is still one of the secure public key cryptography schemes as of present, but it offers several disadvantages, for instance, very slow key generation, prone to the factorization attacks and so on. Due to recent advancements in computing technologies such as quantum computing and the presence of advance factorization techniques, an attacker can find the primes factors of the public key component [27, 28]. In this aspect, researchers and scientists are continuously working on to improve its security and efficiency of public key cryptographic techniques. This segment showcases the working system of the standard RSA, and the corresponding recent research works that have been done in to escalate and improve the RSA public key cryptography.

2.1 RSA scheme

The standard RSA algorithm which was proposed in 1977 by three scientists named as Ron Rivest, Adi Shamir and Leo Adleman [29]. The basis of RSA is that the factorization of large prime numbers difficult and enigmatic. The algorithm initially follows the generation of one public key and one private key, thus finally the public key with two components e and n is generated, and the private key with d and n components is generated.

The algorithm of the standard RSA consists of three phases—Key generation, Encryption and Decryption:

Standard RSA Algorithm

```
Phase 1: Key Generation
1:  Initialize two random large primes as p, q
2:  Evaluate modulus n as,
    $n = p*q$
3:  Find Euler totient function as,
    $\varphi(n) = (p-1)*(q-1)$
4:  Now, compute public key exponent e, such that,
    $1 < e < \varphi(n)$, and GCD of $(e, \varphi(n)) = 1$
5:  And compute private key exponent d, such that,
    $e*d = 1*\text{mod } \varphi(n)$
6:  Generated Public key is (n, e), and Private key is (n, d)
Phase 2: Encryption
$c = m^e \text{ mod } n$,
that uses the public key components as (n, e),
and Plaint text message as m
Phase 3: Decryption
$m = c^d \text{ mod } n$,
that uses the private key components as (n, d),
where c is the encoded cipher string and m is the original string.
```

In terms of modern cryptography development, one of the most adaptive solutions in respect to the enhanced RSA version came to be the involvement of multiple prime generations rather than using only two large primes [30], and thus trying for more coherent results of the more secure version of RSA. Lately researchers have also proposed many other upgrades to the current RSA system and most among those enhanced adaptive algorithms consist of methods like varying key sizes, the inclusion of multiple modulus in the generation of the public–private key pairs, secret key components, exponential powers, increment in the complexity, number theory, Chinese remainder theorem and others, which have been discussed in the next section.

2.2 Related works

In this section, we focus on most prominent works on RSA algorithms, which uses more than two random generated prime numbers to create public and private keys. Some methods are implemented and compared with the proposed method, while others are assessed theoretically as we critically analyzed all the methods and compared them in tabular form as shown in Table 1.

2.2.1 An enhanced and secured RSA key generation scheme (ESRKGS)

Thangavel et al. [31] in 2015 suggested an enhanced and secure system, keeping the length of all initially generated 4 primes of same length as of standard RSA. The key generation depends upon all four large primes but the public and private key modulus 'n' in the generated keys is the multiple of two large prime integers, which eventually offers more trouble to the attacker in context of security. Therefore, any one with simply the knowledge of public key exponent n cannot break the cipher. The proposed solution also involves a variable E1 to increase complexity and is used in evaluating e and d exponents. The corresponding analysis shows that the key generation time is more than standard RSA because of higher complexity, also, the encryption-decryption time is more than that of the standard RSA. In 2016, Erkam Lüy [32] et al. attacked and analyzed this algorithm, and concluded that it has the same degree of security as of standard RSA.

The algorithm of the ESRKGS is as follows:

Table 1 Algorithm analysis-and-evaluation of the recent related enhanced models

Algorithm Model	Total Modulus	Intermediate Variables as of standard RSA	Encryption Scheme	Key Generation	Decryption Scheme	Shortcomings / Comments
Thangavel et al. [31]	3, i.e., n, m, N	e1,e2, $E1 = e1^{e2}$	$C = M^E \mathrm{mod}\ n$	$E*D = 1*\mathrm{mod}\ [\varphi(N)*E1]$	$M = C^D \mathrm{mod}\ n$	Same security level as Standard RSA [32]
Islam et al. [33]	1, i.e., N	Not Any	$C = (M^E \mathrm{mod}\ N)^F \mathrm{mod}\ N$	$E*D = 1*\mathrm{mod}\varphi(N)$	$M = (M^G \mathrm{mod}\ N)^D \mathrm{mod}\ N$	Higher Total Execution time
Amalarethinam and Leena [35]	2, i.e., N1, N2	Not Any	$C = M^E \mathrm{mod} N1$	$E*D = 1*\mathrm{mod}\varphi(N1)$	$M = C^D \mathrm{mod} N2$	Additional time required for magic rectangle [9, 36]
Jaju and Chowhan [37]	1, i.e., N	X, derived from N	$C = M^E \mathrm{mod} X$	$E*D = 1*\mathrm{mod}\varphi\ (N)$	$M = C^D \mathrm{mod} X$	Higher Encryption-Decryption time
Raghunandhan et al. [38]	1, i.e., N	$X(\leftrightarrow N)$,$F(\leftrightarrow E)$, $G(\leftrightarrow D)$	$C = M^F \mathrm{mod} X$	$F*G = 1*\mathrm{mod}\varphi(X)$	$M = C^G \mathrm{mod} X$	Higher Encryption-Decryption time
Minni et al. [39]	1, i.e., N	X $((\leftrightarrow N)$,	$C = M^E \mathrm{mod} X$	$E*D = 1*\mathrm{mod}\varphi(X)$	$M = C^D \mathrm{mod} X$	Higher time Complexity [31]
Mezher [40]	1, i.e., N	$i = 1,2,3\ldots m$	$C = M^{Ei} \mathrm{mod} N$	$E*D = 1*\mathrm{mod}\varphi(N)$	$C = M^{Ei} \mathrm{mod} N$	Time Costly
Mathur et al. [41]	1, i.e., N, $L \rightarrow N$	P, Q, O, N, s.t., $Q = P*E$	$C = M^E \mathrm{mod} N$	$E*D = 1*\mathrm{mod}\varphi\ (N)$	$M = C^D \mathrm{mod} N$	Slower, i.e., higher encryption-decryption time [42]
Panda and Chattopadhyay [43]	1, i..e., N	P1, w (key components)	$C = M^E \mathrm{mod}\ w$	$E*D = 1*\mathrm{mod}\ [\varphi(N)*P1]$	$M = C^D \mathrm{mod}\ w$	Security is not efficient [44]
Abdeldaym et al. [45]	2, i.e., N, Z	g, t key components	$C1 = M^E \mathrm{mod} N$ $C = C1^g \mathrm{mod}\ Z$	$E*D = 1*\mathrm{mod}\varphi(N)$	$M = C^t \mathrm{mod}\ Z$	Lower computational speed
Jahan et al. [46]	1, i.e., N	x, y, s.t. $y = x*E$	$C = M^{y/x} \mathrm{mod} N$	$E*D = 1*\mathrm{mod}\varphi\ (N)$	$M = C^D \mathrm{mod}\ Z$	Higher computational cost
Karakra et al. [47]	2	R	$R = M^2\ \mathrm{mod}\ N$	$E = D - 1\ \mathrm{mod}\ \varphi(N)$	$M = C^D \mathrm{mod}\ N$	Higher Security level

ESRKGS Algorithm

```
Phase 1: Key Generation
1: Generate 4 large random primes as a, b, c and d where $a \neq b \neq c \neq d$
2: Compute modulus n = a*b, m = c*d, N = n*m
3: Calculate the Euler-totient functions as:
    • $\varphi(n) = (a-1)*(b-1)$
    • $\varphi(m) = (c-1)*(d-1)$
    • $\varphi(N) = \varphi(n)*\varphi(m)$
4: Evaluate random integer e1 which is such that:
    • $1 < e1 < \varphi(n)$,
    • GCD of $(e1, \varphi(n)) = 1$
5: Similarly, evaluate e2 which is also a random integer such that:
    • $1 < e2 < \varphi(m)$,
    • GCD of $(e2, \varphi(m)) = 1$
6: Now compute $E1 = e1^{e2} \bmod N$
7: Finally compute public key exponent E such that that:
    • $1 < E < \varphi(N)*E1$,
    • GCD of $(E, \varphi(N)*E1) = 1$
8: Compute private key exponent D such that, $E * D = 1 * \bmod \varphi(N)*E1$
9: Finally, the generated key pairs are:
    • Public-Key is (E, n) and,
    • Private-Key is (D, n)
Phase 2: Encryption
Considering M < n, $C = M^E \bmod n$,
Where (E, n) are public key components, and Plaint text message is M and C is the Ciphertext
Phase 3: Decryption
$M = C^D \bmod n$, where private key components are (D, n).
```

2.2.2 *Modified and secure RSA (MRSA)*

Another enhanced and secure approach of RSA is proposed by Islam et al. [33] named as MRSA, which is also implemented on the multiple large prime numbers of the same size as of standard RSA. The resulting public key as well as private key consists of three key components instead of two, whereas in encryption and decryption, the message is encrypted and decrypted using nested power modulus. This is done in order to enhance its adaptable security. The encryption and decryption time is quite higher in comparison to RSA which eventually makes MRSA computationally costly. In respect to the total time execution, MRSA is seen as having a higher time complexity.

The algorithm of the MRSA is as follows:

MRSA Algorithm

```
Phase 1: Key Generation
1: Generate 4 primes as a, b, c, d
2: Compute modulus N = a*b*c*d
3: Calculate the Euler-totient function $\varphi(N) = (a-1)*(b-1)*(c-1)*(d-1)$
4: Compute random integer E such that:
    • $1 < E < \varphi(N)$,
    • GCD of $(E, \varphi(N)) = 1$\
5: And also compute random F such that:
    • $1 < F < \varphi(N)$,
    • GCD of $(F, \varphi(N)) = 1$
6: Now find private key exponent D corresponding to E as,
    • $E * D = 1 * \bmod \varphi(N)$
7: Similarly, another private key exponent G corresponding to F as,
    • $F * G = 1 * \bmod \varphi(N)$
8: Finally, the keys generated are as:
    • Public-key: (E,F,N) and,
    • Private-key: (D,G,N)
Phase 2: Encryption
Now, encrypting the original plaintext M as:
$C = (M^E \bmod N)^F \bmod N$ that uses the public key components as (E, F, N).
Phase 3: Decryption
Decrypting the ciphertext C multiple times as:
$M = (C^G \bmod N)^D \bmod N$ that uses the public key components as (F, G, N).
```

3 Proposed RSA encryption model based on XOR operations: XRSA

The proposed model consists of the initialization of 4 large random primes, 2 extra primes in comparison to the standard RSA. The use of 4 random primes by the recommended framework enhances the RSA security. The computation of public key exponent E is also not direct, rather it involves the XOR operation and intermediate variables which increases the complexity and hence increases the security. The values E1, E2 are required to determine the value of E' and its corresponding D', thus increasing the time require to attack the cipher, and thereafter using E' and D', we can initiate the generation of public and private key pairs as (E,N) and (D,N) respectively.

Thus, any adversary with just the details of N cannot discover all the primes which is the core for interpreting the value D. The parameter E' also raises the complexity of the algorithm. Consequently, the involvement XOR operations in the algorithm elevates the security. Also, by knowing the value of E (public key), attackers cannot deduce the value of D (private key) since E and D are not directly used in encryption and decryption process. Although using of intermediate private–public keys add some extra time but overall, it adds more security in overall process.

3.1 Algorithm of the proposed scheme

This model consists of 3 stages:

Generation of public–private keys
Encryption
Decryption

The Algorithm of proposed XRSA is followed as:

XRSA Algorithm

```
Phase 1: Key generation:
1:  Generate 4 large random prime numbers as p1, p2, p3, p4
2:  Compute modulus x = p1*p2,   y = p3*p4, N = x1*x2
3:  Calculate the Euler-totient functions
        • $\varphi(x) = (p1-1) * (p2-1)$,
        • $\varphi(y) = (p3-1) * (p4-1)$,
        • and $\varphi(N) = \varphi(x) * \varphi(y)$
4:  Compute components E1 and E2 such that:
        • $1 < E1 < \varphi(x)$
        • $1 < E2 < \varphi(y)$
        • And GCD of $(E1*E2, \varphi(N)) = 1$
5:  Next, find the integer E' as:
        • $E' = (E1 * E2) \bmod N$
6:  Compute D' such that:
        • $E'*D'=1*\bmod (\varphi (N))$
7:  Finally, Public key exponent E is as,
        • E = E' XOR N
8:  Corresponding private key exponent D is as,
        • D = D' XOR N
9:  Finally, the generated key pair is:
        • Public: (E, N) and Private: (D, N)

Phase 2: Encryption:
Using Public Key, the sender can encrypt the message as below:
E''= E XOR N
        $C = M^{E''} \bmod N$
Phase 3: Decryption:
Recipient can decrypt the cipher texts by private key (D, N) as:
D'' = D XOR N
        $M = C^{D''} \bmod N$, where C is encrypted-text and M is
        original-text.
```

3.2 Flowchart of XRSA

The flowchart provided as a pictorial representation of the proposed method is shown in Fig. 2. As depicted in the illustration, the method starts with initializing 4 random primes and ends after getting the actual plaintext from ciphertext.

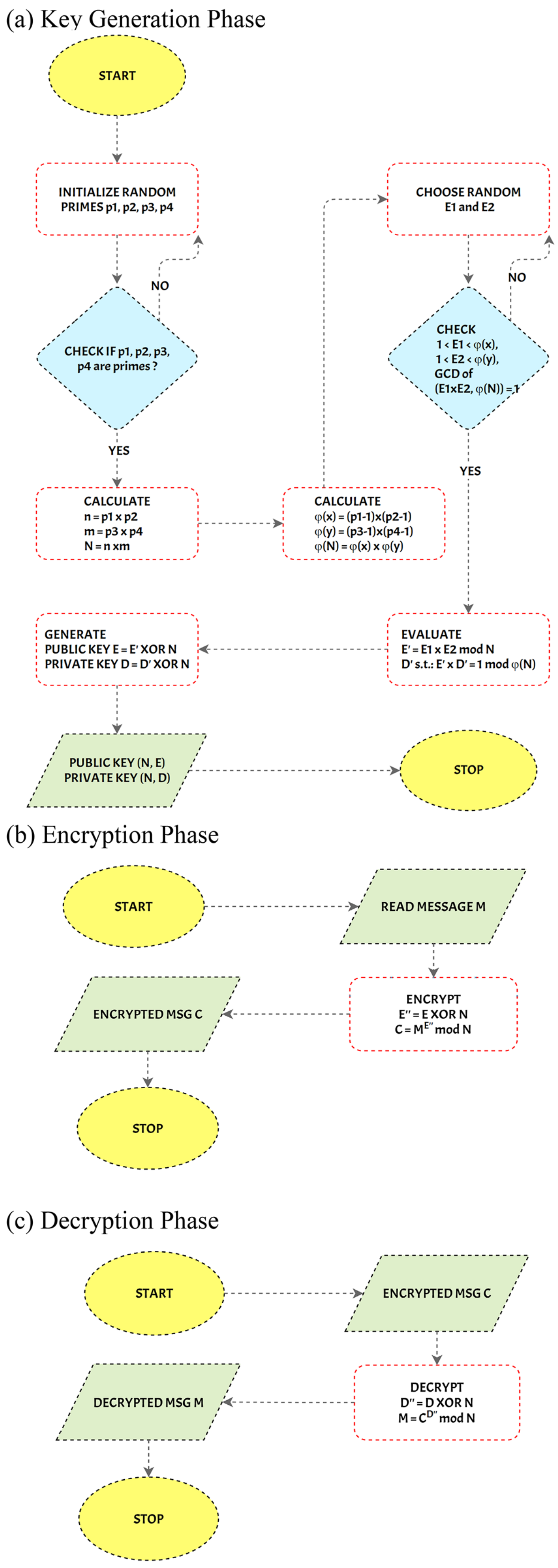
(a) Key Generation Phase
START
INITIALIZE RANDOM PRIMES p1, p2, p3, p4
CHECK IF p1, p2, p3, p4 are primes ?
NO
YES
CALCULATE n = p1 x p2 m = p3 x p4 N = n xm
CALCULATE φ(x) = (p1-1)x(p2-1) φ(y) = (p3-1)x(p4-1) φ(N) = φ(x) x φ(y)
CHOOSE RANDOM E1 and E2
CHECK 1 < E1 < φ(x), 1 < E2 < φ(y), GCD of (E1xE2, φ(N)) = 1
NO
YES
EVALUATE E' = E1 x E2 mod N D' s.t.: E' x D' = 1 mod φ(N)
GENERATE PUBLIC KEY E = E' XOR N PRIVATE KEY D = D' XOR N
PUBLIC KEY (N, E) PRIVATE KEY (N, D)
STOP
(b) Encryption Phase
START
READ MESSAGE M
ENCRYPT E'' = E XOR N C = M^E'' mod N
ENCRYPTED MSG C
STOP
(c) Decryption Phase
START
ENCRYPTED MSG C
DECRYPT D'' = D XOR N M = C^D'' mod N
DECRYPTED MSG M
STOP

Fig. 2 Flowchart of the proposed XRSA model in the top-down order **a** Key generation, **b** Encryption, **c** Decryption

All three activities: (a). Key Generation, (b). Encryption, and (c). Decryption processes are illustrated in three separate flowcharts for each of the phases.

3.3 Example on the basis of the proposed XRSA algorithm

Example of XRSA Algorithm

Phase 1: Key generation:

1: Choosing 4 prime numbers as p1 = 61, p2 = 137, p3 = 97, p4 = 113

2: Compute modulus **x** = 61*137 = 8357; **y** = 97*113 = 10961; **N** = x*y = 91601077

3: Finding the Euler-totient functions as
- φ(x) = (61-1) * (137-1) = 8160
- φ(y) = (97-1) * (113-1) = 10752
- and φ(N) = φ(x) * φ(y) = 87736320

4: Now, computing E1 and E2 such that:
- 1 < E1 < φ(x)
- 1 < E2 < φ(y)
- And GCD of (E1*E2, φ(N)) = 1
- Thus, we have E1 = 3667, and E2 = 3931

5: Compute E' as:
- E' = (E1 * E2) mod N
- Hence, we have E' = 14414977

6: Next, evaluate D' such that:
- E'*D'=1*mod (φ (N))
- Hence, D' = 80399233

7: Public key exponent E is as,
- E = E' XOR N
- Therefore, E = 95308852

8: Private key exponent D is as,
- D = D' XOR N
- Therefore, D = 29324084

9: Finally, the generated public-private pair of keys are:
- Public-key is (E, N) and Private-key is (D, N)
- Or **Public**: (95308852, 91601077), and **Private**: (29324084, 91601077)

Phase 2: Encryption:

Using Public Key, encrypting the message M = 59 as:

E''= E XOR N = 14414977

Since C = $M^{E''}$ mod N

Therefore, C = $59^{14414977}$ mod 91601077 = 16994362

Phase 3: Decryption:

Recipient can decrypt the cipher texts by private key (D, N) as:

D'' = D XOR N = 80399233

And since, M = $C^{D''}$ * mod N

Hence, M = $16994362^{80399233}$ mod 91601077 = 59,

where C is encrypted-text and M is original-text.

4 Mathematical Proof of The Proposed XRSA Method

Mathematically, the proposed algorithm XRSA is proven as follows:

The encrypted text C is calculated using:

$C = M^{E''} \mathrm{mod}.N$ (1)

Plaintext message M is calculated using,

$M = C^{D''} \mathrm{mod}\ N$ (2)

Now, the objective is to fetch back the original message M from $C^{D''} \mathrm{mod}\ N$

So, using equation (1) and (2), we can say,

$C^{D''} \mathrm{mod}.N = M^{E''D''} \mathrm{mod}.N$

$= M^{(E\ XOR\ N)(D\ XOR\ N)} \mathrm{mod}.N$

[Since E''= E XOR N and D''= D XOR N]

$= M^{(E'\ XOR\ N\ XOR\ N)(D'\ XOR\ N\ XOR\ N)} \mathrm{mod}\ N$

[Since E= E` XOR N and D = D` XOR N]

$= M^{(E'\ XOR\ 0)(D'\ XOR\ 0)} \mathrm{mod}\ N$

[Since N XOR N = 0]

$= M^{E'\ D'} \mathrm{mod}\ N$

[Since any entity XOR with 0 is always that entity]

$= M^{1*\mathrm{mod}\ \varphi(N)} \mathrm{mod}.N$

[Since D'*E' = 1*mod φ(N)]

$= M^{(1+K.\ \varphi(N))} \mathrm{mod}.N$

$= M^1 M^{K.\varphi(N)} \mathrm{mod}\ N$

$= M^1 \mathrm{mod}.N$

= M, which is the original plaintext.

[Hence Proved]

5 Implementation results and analysis

The proposed XRSA algorithm is implemented through the Jupyter Notebook using the SageMath, which is a free open-source software system that offers a numerous tools and APIs on symmetric and asymmetric key cryptography. SageMath contributed functions for Euclidean algorithm, random large prime generation of different bits length, mod-inverse, power-mod, and other concepts. The hardware used in the implementation and analysis is quad CPU with clock rate of 2.1—3.7 GHz AMD Ryzen 5 3500U alongside 8 GB of RAM. The next two sub-sections focus on the performance and security evaluations of XRSA in respect to other RSA variants.

5.1 Performance analysis

In respect to parameters referencing to performance analysis like Key generation time, Encryption time and Decryption time, our proposed RSA model, i.e., XRSA, is compared with the standard RSA by Rivest et al. [29] alongside two other enhanced version of RSA, namely ESRKGS [31] and MRSA [33].

5.1.1 *Key Generation*

The tests are performed and analyzed by generating initial random primes for each of the four cases using a prime generation algorithm based on Miller-Rabin Method to keep the results away from any inclination or partiality. Each test is performed for the varying bit sizes of initial primes, which ranges as, 100, 128, 256, 512, 1024, 2048 and, 4096. Table 2 showcases the performance and comparison analysis of XRSA with RSA, ESRKGS, and MRSA RSA. Figure 3 shows key generation time comparison with different key sizes and it can be concluded that the key generation time is highest in the case of MRSA while it is lowest and best in standard RSA case. Our proposed XRSA lies at the second-best key generation time among all implemented algorithms. The increase in time is bearable since the security is enhanced in the proposed method. It is obvious that inclusion of 4 initial random primes in the algorithm will increase time complexity than that of 2 random primes, but rather our proposed XRSA model is efficiently taking lesser key generation time in comparison to ESRKGS and MRSA, which also involve 4 random primes in their algorithm. On arranging all 4 algorithms corresponding to their key generation time, it can be concluded that overall RSA has the lowest, XRSA has the second lowest, ESRKGS lies at third lowest, and MRSA has the highest key generation time.

5.1.2 *Encryption and decryption*

Considering the parameters of encryption and decryption time, which is represented in Figs. 4 and 5, it is deduced that encryption and decryption execution time of the standard RSA is the lowest and hence best among all in encryption-decryption time, but since XRSA algorithm involves extra steps in the encryption and decryption stages that eventually increases the overall complexity of the proposed algorithm. The attacker cannot guess the private key or plain text using public key and product of primes, since one more XOR step is added in final encryption and decryption. Further, in view of 4 primes, XRSA shows better result in correspondence to ESRKGS up to the typical 1024-bit size, for both encryption as well as decryption time. In Contrast to MRSA, XRSA has better efficiency in terms of encryption, decryption, and key generation time for every bit. For higher bit sizes like 2048 and 4096 bits, XRSA encryption and decryption time is almost equal to ESRKGS, but for commonly recommended prime size of 1024 bits, the proposed XRSA is more efficient and better among all discussed algorithms. Also considering the box plots, i.e., Figs. 6, 7 and 8, which are implemented by taking 10 random samples for the 4 discussed algorithms on 1024-bit. Considering every parameter, i.e., encryption, decryption and key generation time one can interpret the algorithm's median performance. Considering the outliers, median and quartiles data, the performance of XRSA is almost equal or just next to RSA and superior to ESRKGS and MRSA in terms of key generation, noting that XRSA, ESRKGS and MRSA feature the involvement of 4 primes. And in terms of encryption-decryption time, box plots shows that the median time is best for RSA, then that for XRSA is slightly lower than ESRKGS, and MRSA has the highest encryption and decryption time. Thus, it can be deduced that in terms of overall performance assessment that, XRSA algorithm is best after standard RSA among all of the 4 discussed algorithms. While ESRKGS and MRSA are the least efficient in performance among the discussed variants.

The increased key time in XRSA as compared to standard RSA is justified by the fact that the time to break the XRSA system is higher because of the complexity introduced and hence brings more security to the overall process.

5.2 Security analysis

In order to showcase the reliability of any proposed RSA variant, the security parameters should always be cross-checked. Mok and Chuah [34] mentioned two simulations to attack the standard RSA algorithm using brute force techniques, first is that when the public key components (n, e) and ciphertext C is known, the objective is to estimate the exact value of d such that $e*d = 1*\mathrm{mod}\ \varphi(n)$. As soon as the correct private key component, i.e., d is acquired, implies that the attack is successful. The later simulation involved the direct factorization of the modulus n to acquire the engaged prime integers. As in our undertaken algorithms, i.e., standard RSA, XRSA, ESRKGS and MRSA, most of them generate 4 random primes in their algorithms instead of two, so that's why we've utilized the first mentioned method in our implementation of brute force attacks, i.e., cracking the value of private key component d in each of the cases by finding the value of euler-totient function.

Table 3 represents the Brute force attack time comparison that ranges from primes of length of 4 bits up to 8-bit primes. Since attacking on above 8-bit primes, it's taking ages to execute the results, so we kept our analysis only up to 8 bits. Figure 9 also shows an illustrative manner of all of the 4 considered RSA algorithms including the standard RSA via a line graph. As from the corresponding table and graph, it can clearly be seen that, over all the prime lengths,

Table 2 Analysis and comparison of XRSA model with RSA, ESRKGS and MRSA models

Model	Length of primes (in bits)	Key generation Time (in ms)	Encryption Time (in ms)	Decryption Time (in ms)	Total Execution time (in ms)
RSA	100	32.12	0.41	0.34	32.88
	128	36.39	0.53	0.47	37.39
	256	57.86	1.38	1.22	60.47
	512	109.36	2.50	2.40	114.26
	1024	1171.63	21.83	20.77	1214.23
	2048	6535.75	74.65	53.41	6663.81
	4096	48,639.66	319.20	304.30	49,263.16
XRSA	100	45.71	0.82	0.77	47.30
	128	59.51	1.85	1.31	62.67
	256	116.92	2.52	2.50	121.94
	512	236.14	13.16	11.95	261.25
	1024	1153.50	49.60	45.08	1248.17
	2048	8382.77	232.48	276.76	8892.01
	4096	88,691.33	1252.62	1301.02	91,244.96
ESRKGS	100	71.47	1.37	1.34	74.18
	128	95.41	1.82	1.75	98.97
	256	184.01	5.75	4.71	194.47
	512	285.78	14.76	15.17	315.72
	1024	2212.26	50.15	49.06	2311.47
	2048	17,754.69	235.01	252.97	18,242.67
	4096	152,723.83	1333.50	1349.11	155,406.44
MRSA	100	106.37	2.09	2.03	110.48
	128	114.83	2.51	2.52	119.86
	256	206.15	6.91	6.77	219.83
	512	487.25	23.83	23.54	534.62
	1024	2003.02	117.69	112.12	2232.83
	2048	14,373.83	668.95	626.23	15,669.01
	4096	161,718.40	3507.21	3635.91	168,861.51

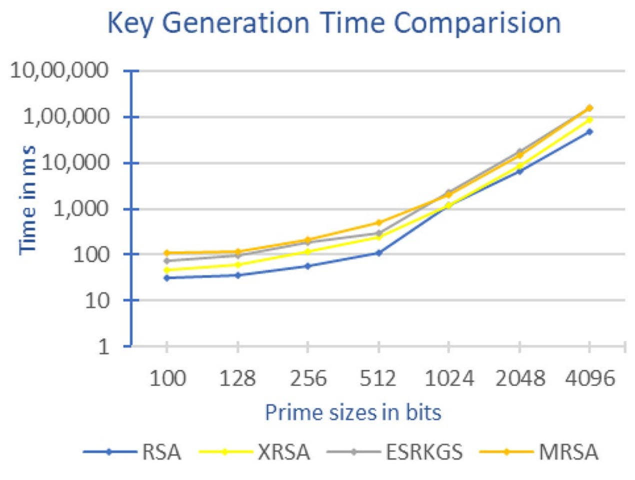

Fig. 3 Key generation time comparison with different key sizes

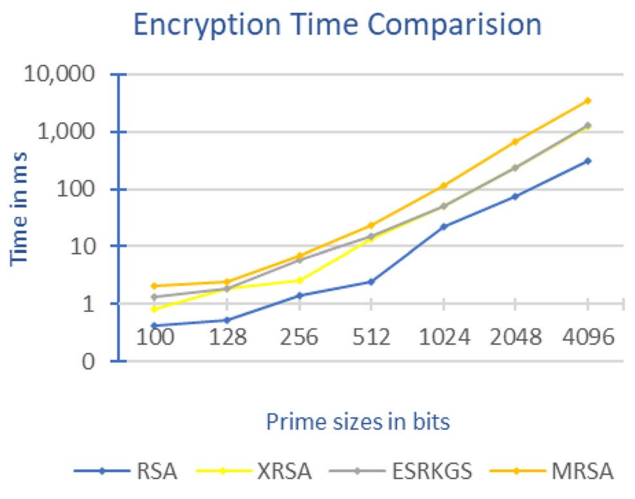

Fig. 4 Encryption time comparison with different key sizes

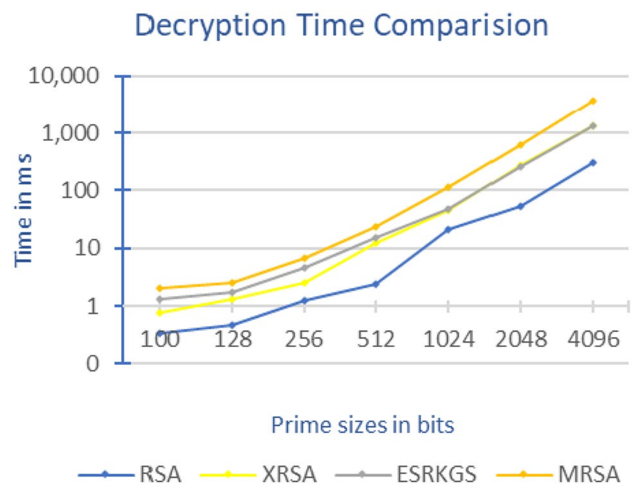

Fig. 5 Decryption time comparison with different key sizes

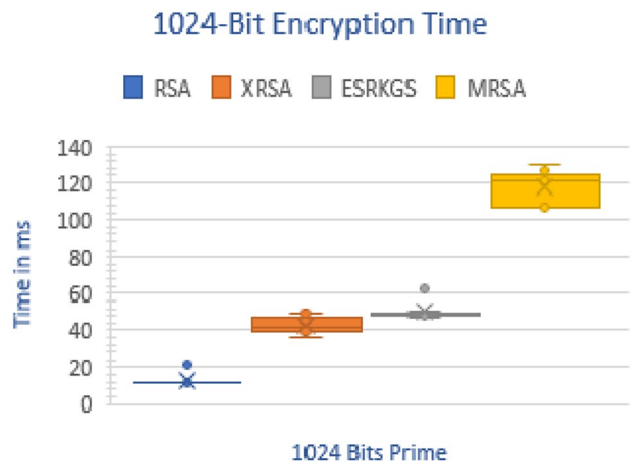

Fig. 6 Encryption time comparison on 10 samples

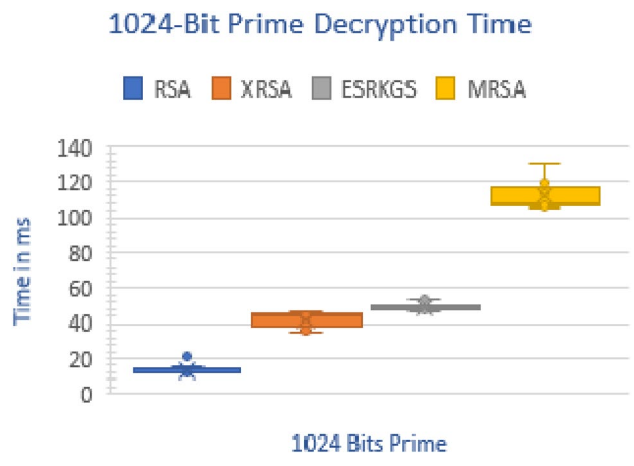

Fig. 7 Decryption time comparison on 10 samples

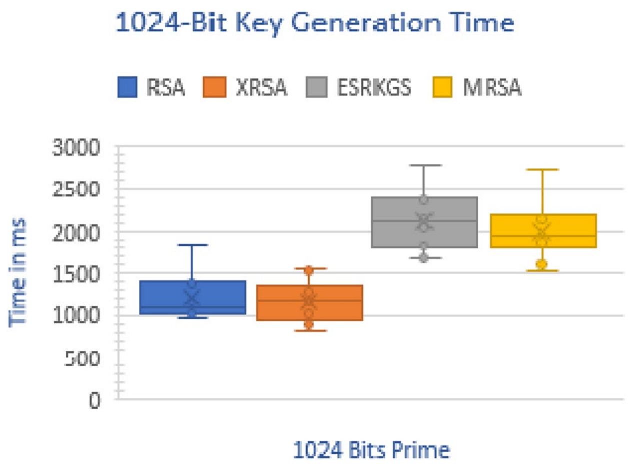

Fig. 8 Key generation comparison on 10 samples of 1024 bits

RSA is taking the least amount of time in brute-forcing its private key component, while ESRKGS is also showing the same results as of standard RSA, almost over all of the prime lengths. This simply implies that standard RSA and ESRKGS displays the same complexity levels, and hence the same security in their respective algorithms [32], and hence can be concluded as the least secure algorithms among all of the discussed variants. Between MRSA and the proposed RSA variant XRSA, the later one is showing a significantly higher breaking time in comparison to MRSA, ESRKGS and standard RSA over all prime bits. Hence, it can be concluded that the proposed RSA variant, XRSA, has the highest security among all as it presents the highest brute-force attacking time among all. Having the second highest and slightly less attacking time than of XRSA, MRSA can be stated as the second secure among all. However, ESRKGS and standard RSA can be considered the least secure among all 4 RSA variants, since with the minute difference in their attacking time, both ESRKGS and standard RSA can be stated as least secure.

6 Conclusion and future scope

This paper dealt with the enhancement of the standard RSA algorithm in terms of efficiency and reliability by analyzing their performance and security, respectively. The enhanced algorithm is named as XRSA where XOR operations with other modular arithmetic operations are used to enhance the security of RSA algorithm. The inclusion of the intermediate variables and corresponding XOR operations enhances the security, as this results in a more complex algorithm, which takes a higher time to break the cipher in comparison to the standard RSA. The XRSA initially uses 4 primes instead of

Table 3 Brute force attack time comparison in millisecond

Prime length (in bits)	RSA	XRSA	ESRKGS	MRSA
4	1.74	19.53	1.17	20.04
5	2.48	304.92	2.06	284.36
6	4.30	4539.59	4.05	3587.92
7	12.23	63,509.64	11.49	48,074.28
8	47.55	2,759,566.10	47.14	829,912.91

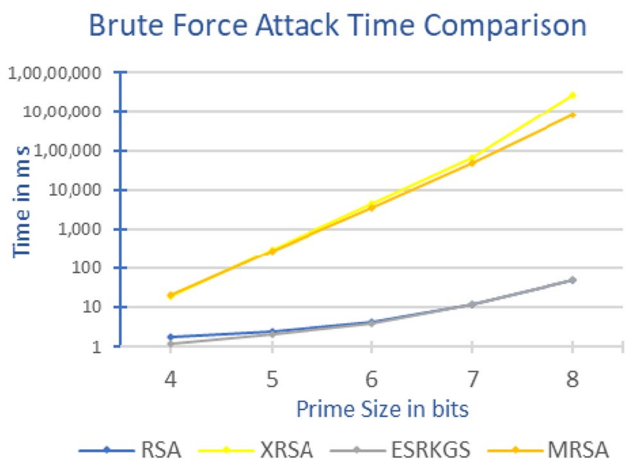

Fig. 9 Brute force attack time comparison on prime lengths ranging from 4 to 8 bits

2 to boost its time to attack the cipher and therefore higher security. The key generation time of the proposed RSA based algorithm is more efficient even using 4 primes as compared to other multiple-prime algorithms. Considering the encryption and decryption parameters, XRSA shows better security as well as better performance. The encryption key is not directly used for encrypting the message, instead a modified key is used in the proposed algorithm. Although this takes some extra computation time as compared to others but we see its advantageous as it makes guessing private key more tough. Altogether, following the performance and security analysis, the proposed RSA variant, XRSA, showed superior results with increased security.

As a future work, we are planning to extend this algorithm to run in parallel machine so that several related operations can be performed simultaneously on multi-core processors. This would be computationally less costly as compared to RSA public key cryptography scheme and will have wide acceptability in IoT devices.

Acknowledgements We would like to thank Prof. Khurram Mustafa for his valuable and critical review.

References

1. Stallings W (2006) Cryptography and network security, 4/E
2. Zhu H et al (2019) A secure and efficient data integrity verification scheme for cloud-IoT based on short signature. IEEE Access 7:90036–90044. https://doi.org/10.1109/ACCESS.2019.2924486
3. Jegadeesan S, Obaidat MS, Vijayakumar P, Azees M (2021) SEAT: secure and energy efficient anonymous authentication with trajectory privacy-preserving scheme for marine traffic management. IEEE Trans Green Commun Netw. https://doi.org/10.1109/TGCN.2021.312661810.1109/TGCN.2021.3126618
4. Mullai A, Mani K (2020) Enhancing the security in RSA and elliptic curve cryptography based on addition chain using simplified Swarm Optimization and Particle Swarm Optimization for mobile devices. Int J Inf Technol 13(2):551–564. https://doi.org/10.1007/S41870-019-00413-8
5. Gupta A, Walia NK (2014) Cryptography algorithms: a review. Int J Eng Dev Res 2(2): 1667–1672. http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=FEF3E8340DC536679E3C83BF43F1616C?doi=10.1.1.674.7141&rep=rep1&type=pdf.
6. Anas M, Imam R, Anwer F (2022) Elliptic curve cryptography in cloud security: a survey. Int Conf Cloud Comput Data Sci Eng. https://doi.org/10.1109/CONFLUENCE52989.2022.9734138
7. Mumtaz M, Ping L (2019) Forty years of attacks on the RSA cryptosystem: a brief survey. J Discret Math Sci Cryptogr 22(1):9–29. https://doi.org/10.1080/09720529.2018.1564201
8. Imam R, Areeb QM, Alturki A, Anwer F (2021) Systematic and critical review of RSA based public key cryptographic schemes: past and present status. IEEE Access 9:155949–155976. https://doi.org/10.1109/ACCESS.2021.3129224
9. El Makkaoui K, Ezzati A, Beni-hssane A (2017) Cloud-RSA: an enhanced homomorphic encryption scheme. pp. 471–480. doi: https://doi.org/10.1007/978-3-319-46568-5
10. El Makkaoui K, Beni-Hssaneb A, Ezzatia A, El-Ansarib A (2017) Fast cloud-RSA scheme for promoting data confidentiality in the cloud computing. Proc Comput Sci 113:33–40. https://doi.org/10.1016/j.procs.2017.08.282
11. Moghaddam FF, Alrashdan MT, Karimi O (2013) A hybrid encryption algorithm based on RSA small-e and efficient-RSA for cloud computing environments. J Adv Comput Netw 1(3):238–241. https://doi.org/10.7763/jacn.2013.v1.47
12. Bansal VP, Singh S (2015) A hybrid data encryption technique using RSA and Blowfish for cloud computing on FPGAs. In: 2015 2nd Int. Conf. Recent Adv. Eng. Comput. Sci. RAECS 2015. doi: https://doi.org/10.1109/RAECS.2015.7453367.
13. Alsabti KDM, Hashim HR (2016) A new approach for image encryption in the modified RSA cryptosystem using MATLAB. Glob J Pure Appl Math 12(4):3631–3640
14. Jagadiswary D, Saraswady D (2017) Estimation of modified RSA cryptosystem with hyper image encryption algorithm. Indian J Sci Technol 10(7):1–5. https://doi.org/10.17485/ijst/2017/v10i7/111000
15. Frunza M, Scripcariu L (2007) Improved RSA encryption algorithm for increased security of wireless networks. In: 2007 International symposium on signals, circuits and systems, pp 1–4
16. Chakraborty M, Jana B, Mandal T, Kule M (2018) An performance analysis of RSA scheme using artificial neural network. 2018 9th Int Conf Comput Commun Netw Technol ICCCNT 2018:1–5. https://doi.org/10.1109/ICCCNT.2018.8494032
17. Reddy LS (2020) "RM- RSA algorithm. J Discret Math Sci Cryptogr. https://doi.org/10.1080/09720529.2020.1734292
18. Al-Barazanchi I, Shawkat SA, Hameed MH, Al-Badri KSL (2019) "Modified RSA-based algorithm: a double secure approach. Telkomnika Telecommun Comput Electron Control

17(6):2818–2825. https://doi.org/10.12928/TELKOMNIKA.v17i6.13201

19. Wahab OFA, Khalaf AAM, Hussein AI, Hamed HFA (2021) Hiding data using efficient combination of RSA cryptography, and compression steganography techniques. IEEE Access 9:31805–31815. https://doi.org/10.1109/ACCESS.2021.3060317
20. Bhattacharjya A, Zhong X, Li X (2019) A lightweight and efficient secure hybrid rsa (shrsa) messaging scheme with four-layered authentication stack. IEEE Access 7(5):30487–30506. https://doi.org/10.1109/ACCESS.2019.2900300
21. Mustafa I et al (2020) A lightweight post-quantum lattice-based RSA for secure communications. IEEE Access 8:99273–99285. https://doi.org/10.1109/ACCESS.2020.2995801
22. Rawat AS, Deshmukh M (2021) Computation and communication efficient secure group key exchange protocol for low configuration system. Int J Inf Technol 13(3):839–843. https://doi.org/10.1007/S41870-021-00638-6
23. Ochoa-Jimenez E, Rivera-Zamarripa L, Cruz-Cortes N, Rodriguez-Henriquez F (2020) Implementation of RSA signatures on GPU and CPU architectures. IEEE Access 8:9928–9941. https://doi.org/10.1109/ACCESS.2019.2963826
24. Jegadeesan S, Obaidat MS, Vijayakumar P, Azees M, Karuppiah M (2021) Efficient privacy-preserving anonymous authentication scheme for human predictive online education system. Clust Comput 2021:1–15. https://doi.org/10.1007/S10586-021-03390-5
25. Vijayakumar P, Azees M, Kannan A, Deborah LJ (2016) Dual authentication and key management techniques for secure data transmission in vehicular Ad hoc networks. IEEE Trans Intell Transp Syst 17(4):1015–1028. https://doi.org/10.1109/TITS.2015.2492981
26. Vijayakumar P, Naresh R, Jegatha Deborah L, Hafizul Islam SK (2016) An efficient group key agreement protocol for secure P2P communication. Secur Commun Netw 9(17):3952–3965. https://doi.org/10.1002/SEC.1578
27. Chen L et al (2016) Report on post-quantum cryptography, US Department of Commerce, National Institute of Standards and Technology. doi: https://doi.org/10.6028/NIST.IR.8105.
28. Roetteler M, Svore KM (2018) Quantum computing: Codebreaking and beyond. IEEE Secur Priv 16(5):22–36. https://doi.org/10.1109/MSP.2018.3761710
29. Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21:120–126
30. Ivy PU KM, Mandiwa P (2012) A modified RSA cryptosystem based on 'n' prime numbers. Int J Eng Comput Sci 1(2):62–66 (**ISSN2319–7242**). http://www.rsa.com/rsalabs/node.asp?i.
31. Thangavel M, Varalakshmi P, Murrali M, Nithya K (2015) An enhanced and secured RSA key generation scheme (ESRKGS). J Inf Secur Appl 20:3–10. https://doi.org/10.1016/j.jisa.2014.10.004
32. Lüy E, Karatas ZY, Ergin H (2016) Comment on 'an enhanced and secured RSA key generation scheme (ESRKGS).' J Inf Secur Appl 30:1–2. https://doi.org/10.1016/j.jisa.2016.03.006
33. Islam MA, Islam MA, Islam N, Shabnam B (2018) A modified and secured RSA public key cryptosystem based on 'n' prime numbers. J Comput Commun 06(03):78–90. https://doi.org/10.4236/jcc.2018.63006
34. Mok CJ, Chuah CW (2019) An intelligence brute force attack on RSA cryptosystem. Commun Comput Appl Math 1(1). https://fazpublishing.com/ccam/index.php/ccam/article/view/6. Accessed 10 Sept 2021
35. Amalarethinam IG, Leena HM (2017) Enhanced RSA algorithm with varying key sizes for data security in cloud. In: Proceedings—2nd world congress on computing and communication technologies, WCCCT 2017, pp. 172–175, doi: https://doi.org/10.1109/WCCCT.2016.50
36. Patel SR, Shah K (2014) Security enhancement and speed monitoring of RSA algorithm. www.ijedr.org.
37. Jaju SA, Chowhan SS (2015) A modified RSA algorithm to enhance security for digital signature. In: International Conference and Workshop on Computing and Communication (IEMCON), pp. 1–5
38. RKR, Shetty S, Aithal G, Rakshith (2018) Enhanced RSA algorithm using fake modulus and fake public key exponent Raghunandhan
39. Minni R, Sultania K, Mishra S, DRVPM (2013) An algorithm to enhance security in RSA
40. Mezher AE (2018) Enhanced RSA cryptosystem based on multiplicity of public and private keys. Int J Electr Comput Eng 8(5):3949–3953. https://doi.org/10.11591/ijece.v8i5.pp3949-3953
41. Mathur S, Gupta D, Goar V, Kuri M (2017) Analysis and design of enhanced RSA algorithm to improve the security
42. Akhter S, Chowdhury MB (2019) Bangla and english text cryptography based on modified blowfish and Lempel-Ziv-Welch algorithm to minimize execution time. In: 1st International Conference Robotic, Electrical and Signal Processing Techniques ICREST 2019, pp. 96–101. doi: https://doi.org/10.1109/ICREST.2019.8644450
43. Panda PK, Chattopadhyay S (2017) A hybrid security algorithm for RSA cryptosystem
44. Agrawal S, Patel M, Sinhal A (2021) An enhance security of the color image using asymmetric RSA algorithm, pp 279–286. doi: https://doi.org/10.1007/978-981-15-5077-5_25
45. Abdeldaym RS, Elkader HMA, Hussein R (2019) Modified RSA algorithm using two public key and chinese remainder theorem. Int J Electron Inf Eng 10(1):51–64. https://doi.org/10.6636/IJEIE.201903
46. Jahan I, Asif M, Jude Rozario L (2015) Improved RSA cryptosystem based on the study of number theory and public key cryptosystems. Am J Eng Res (1): 143–149. www.ajer.org
47. Karakra A, Alsadeh A (2016) A-RSA: Augmented RSA