

A multi-moduli approach towards enhancing the security of multi-prime RSA

Jay Mehta *

Hitarth Rana †

Department of Mathematics

Sardar Patel University

Vallabh Vidyanagar 388120

Gujarat

India

Abstract

The popular RSA cryptosystem, based on the problem of integer factorization can widely be seen in practice even today. Since its inception, several enhancements and generalizations of RSA are proposed including its several multi-prime variants where the RSA modulus is a product of more than two primes. In this paper, we propose a generalized multi-moduli RSA cryptosystem (GMMRSA) which is a multi-prime RSA but with more than one modulus. We show that this multi-moduli approach yields a more secure cryptosystem than a multi-prime RSA with a single modulus without compromising its computational and time efficiencies.

Subject Classification: 11T71, 94A60.

Keywords: RSA cryptosystem, Multi-prime RSA, Multi-moduli RSA, Key-size, Security.

1. Introduction

The purpose of cryptography is to provide a secure way of communicating over an insecure communication channel. In cryptography, a message called *plaintext* is converted into *ciphertext* (which is gibberish and in an unrecognizable form) using an encryption algorithm and ciphertext is to be converted back into plaintext using a decryption algorithm with some key information. Based on the usage of key information, cryptography can mainly be divided into two different categories: (i) symmetric key (or private key) cryptography and (ii) asymmetric

* ORCID-ID: 0000-0003-1739-9639

† ORCID-ID: 0009-0004-4360-8187

* E-mail: jay_mehta@spuvvn.edu (Corresponding Author)

† E-mail: hitarth.rana@spuvvn.edu

key (or public key) cryptography. Symmetric key cryptosystems usually uses only one secret key that is shared with all the parties involved in the communication, and that one key is used by all the parties while encrypting the message to be sent as well as decrypting the received message. In some cases, the decryption key is different yet easily computable from the encryption key. On the contrary, in asymmetric key cryptography, the encryption key is made public and its corresponding private key is utilized for decryption.

Public (asymmetric) key cryptosystems are widely used nowadays, since they let two distant parties communicate securely without any prior key agreement. The RSA public key cryptosystem (PKC), named after Rivest, Shamir and Adleman [1] is one such example of a PKC which is still in practice. It depends on the difficulty of factorizing a product of a pair of large primes. In RSA, two primes p and q are chosen secretly, and their product $N = pq$ is published. Then a public key e is an integer chosen such that e is co-prime with $\phi(N)$. The private key d is calculated such that $ed \equiv 1 \pmod{\phi(N)}$. Note that the key generation process involves knowledge of $\phi(N)$, i.e., product of $(p - 1)$ and $(q - 1)$ which in turn requires knowledge of the prime factors p and q of RSA modulus N . The encryption uses public key e while the decryption requires private key d . Decryption process utilizes the fact that d is inverse of e modulo $\phi(N)$ combined with Euler's theorem, which states that "If N is a positive integer and a and N are relatively prime, then $a^{\phi(N)} \equiv 1 \pmod{N}$ where $\phi(N)$ is the Euler's totient function." It can be proven that inverting $e \pmod{N}$ is equivalent to factoring N and vice-versa [2, §1.1, fact 1]. Hence, the underlying hard problem for the RSA cryptosystem is integer factorization problem.

One of the obvious approaches in an attempt to break an RSA encryption is to solve the underlying integer factorization problem. There are few algorithms which can efficiently factorize an RSA modulus under certain conditions. One such popular algorithm is Pollard's $p - 1$ factorization algorithm [3]. This algorithm is quite efficient given the condition that one of $p - 1$ and $q - 1$ possesses smaller (odd) prime factors. Another such algorithm is due to Lenstra known as the Elliptic curve factorization algorithm [4] (ECF) which uses Elliptic curves in calculation, and it is also efficient under the same condition as Pollard's algorithm. There are also various other attacks on RSA which are mentioned in [2, 5]. These attacks include low private or public exponent attacks like Franklin-Reiter related message attack and Hastad's broadcast attack, and other attacks like Coppersmith's short pad attack, Timing attack, Partial key exposure attack, Bleichenbacher's attack, etc. It is to be noted that as explicitly concluded in [2] and [5], all these attacks are avoidable by taking care of different aspects and hence RSA has proven itself to be an efficient and secure cryptographic protocol to be used in practice in the digital world.

After seeing the attacks which aim at factorization of RSA modulus, a natural solution to increase the security of RSA is to use more than two primes in RSA in such a way that it adds to its security without increasing computational cost by a significant extent. This idea, multi-prime RSA, seems to be first

appeared in an article by Richard Schroepel under the name Captain Nemo [6] in 1996. He suggested that using three primes instead of two in an RSA modulus will result in faster key generation (using the Chinese remainder theorem) and reduced size of primes will be compensated by having to find three prime factors instead of two. There is also a US patent by Collins et al. [7] in 1998 on multi-prime RSA. Since then, various researchers have suggested using multiple primes in their proposed schemes, for example, see [8], [9], [10], [11], [12]. In all these schemes, the RSA modulus is a product of more than two primes. However, depending on the size of the modulus, using a higher number of prime factors in an RSA modulus can lower the security of the cryptosystem. We shall discuss this in detail in Section 4.1. In an attempt to overcome this security concern while using more than two primes in a multi-prime RSA cryptosystem, we propose using more than one modulus in which every modulus is a product of, preferably but not limited to, two primes. We call it a generalized multi-moduli RSA cryptosystem (GMMRSA), where we use n -moduli ($n \geq 2$). A particular case of GMMRSA for $n = 2$ is given by Manu and Goel [13] in 2017. In this paper, we not only describe a generalized multi-moduli RSA scheme but also provide its detailed security and efficiency analysis. We shall show that such a scheme is more secure than the two-prime RSA cryptosystem as well as any multi-prime RSA cryptosystem while it is computationally as efficient as any multi-prime RSA cryptosystem.

Before we propose our generalized multi-moduli RSA cryptosystem, we recall the famous RSA PKC and also describe a few multi-prime RSA cryptosystems which use a single RSA modulus with more than two prime factors in the succeeding section.

2. The RSA cryptosystem and its multi-prime variants

2.1 The RSA PKC

There are three aspects of the RSA cryptosystem, i.e. key generation, encryption, and decryption. In RSA, firstly two primes, say p_1 and p_2 are chosen, and an integer N is taken to be their product, i.e., $N = p_1 p_2$.

1. Key generation:

- First, the Euler totient of N is calculated as

$$\phi(N) = \phi(p_1)\phi(p_2) = (p_1 - 1)(p_2 - 1).$$

- Then an integer e is selected to be relatively prime with $\phi(N)$.
- Since e is co-prime with $\phi(N)$, it is invertible modulo $\phi(N)$. Thus, there is an integer d satisfying $de \equiv 1 \pmod{\phi(N)}$.

Now, (N, e) is published by the receiver of the message, while the integer d and the primes p_1 and p_2 are kept private.

2. Encryption:

The public keys N and e are used by the sender to encrypt their message M , called the plaintext. The encryption is carried out as follows.

$$C \equiv M^e \pmod{N}.$$

Here, C is called the ciphertext that is communicated to the receiver by the sender.

3. **Decryption:**

The receiver decrypts the ciphertext C back into the original message M with the help of d which is their private key. The decryption process is as given below.

$$\begin{aligned} C^d \pmod{N} &\equiv M^{ed} \pmod{N} \\ &\equiv M \pmod{N} \quad (\because ed \equiv 1 \pmod{\phi(N)}). \end{aligned}$$

Here, e is said to be the *encryption exponent*, d is said to be the *decryption exponent*, and $N = p_1 p_2$ is said to be the *RSA modulus*.

2.2 Multi-prime RSA cryptosystem

As described in the introduction, many versions of multi-prime variants of the RSA were introduced to enhance the security of RSA cryptosystem by making the modulus a product of more than two primes, unlike standard RSA having the modulus with exactly two prime factors. For instance, one such scheme was patented by Collins et al. [7] and also described by Nemo [6] using a 3-prime modulus. It was assumed at the time that the RSA modulus comprising of three or more prime factors is not much easier to factorize than a modulus consisting of two prime factors [7, §4]. They choose n distinct primes $p_1, p_2, \dots, p_n$ and the RSA modulus is their product $N = p_1 p_2 \cdots p_n$. The key-generation, encryption, and decryption as described in their multi-prime RSA is given below.

1. **Key generation:**

- First, the Euler totient $\phi(N)$ of N is computed as a product of $(p_1 - 1), (p_2 - 1), \dots, (p_n - 1)$.
- The receiver chooses a public key e such that $\gcd(e, \phi(N)) = 1$.
- The receiver calculates private key d satisfying $de \equiv 1 \pmod{\phi(N)}$.

Now, the public key is the pair (N, e) while d is the private key.

2. **Encryption:**

Similarly as RSA, the public key (N, e) is used for the encryption of the message (plaintext) M as follows:

$$C \equiv M^e \pmod{N}.$$

3. **Decryption:**

Ciphertext C can be decrypted to obtain the original plaintext M with the use of the private key d as follows:

$$\begin{aligned} C^d \pmod{N} &\equiv M^{ed} \pmod{N} \\ &\equiv M \pmod{N} \quad (\because ed \equiv 1 \pmod{\phi(N)}). \end{aligned}$$

In practice, the Chinese Remainder theorem can be used expedite calculations for key generation and decryption.

As mentioned earlier, many variants of the multi-prime RSA have been proposed in the last three decades after the introduction of the idea. For instances, a generalized $2n$ -prime RSA was proposed by Tariq Shah et al. [12] which uses n encryption and decryption exponents modulo $\phi(N)$, where N is taken to be the product of all the $2n$ primes $p_1, p_2, \dots, p_{2n}$; Ghazali et al. [8] proposed a hybrid multi-prime RSA cryptosystem in which two parties agree on two secret numbers a and b and encryption is carried out as $C \equiv (aM + b)^e \pmod{N}$, where C is the ciphertext while the decryption is carried out using Chinese remainder theorem with appropriate operations that includes use of shared secret keys a and b to get the plaintext message M back.

Due to the size limitations in practical implementation, an RSA modulus consisting of large number of primes becomes vulnerable to factorization attacks as described in Section 4. To keep a multi-prime RSA cryptosystem secure and resilient against various attacks imposed on it, we propose distributing these primes into multiple moduli instead of taking a single modulus as the product of these primes. Our proposed generalized multi-moduli RSA cryptosystem (GMMRSA) consisting of even number of primes is described in the section below in which we use n -moduli, ($n \geq 2$), each of which is a product of a pair of primes.

3. The proposed generalized multi-moduli RSA cryptosystem

In this section, we describe our generalized multi-moduli RSA cryptosystem (GMMRSA), the computational efficiency and the security aspects of which are discussed in the following section by comparing it with multi-prime RSA cryptosystems. A particular case of our proposed cryptosystem using dual modulus is given by Manu and Goel in [13] without discussing much on its security and efficiency aspects. Our proposed GMMRSA scheme consists of n moduli $N_1, N_2, \dots, N_n$ each of which is preferably, but not limited to, a product of two primes. In what follows, we describe our GMMRSA system with each modulus as a product of two primes for brevity.

1. Key generation:

- First, Euler totient for n moduli $N_1, N_2, \dots, N_n$ are calculated as

$$\phi(N_1) = (p_1 - 1)(q_1 - 1)$$

$$\vdots$$

$$\phi(N_n) = (p_n - 1)(q_n - 1).$$

- Now, the receiver chooses n different public keys

$$e_1 \pmod{\phi(N_1)} \text{ such that } \gcd(e_1, \phi(N_1)) = 1$$

$$\vdots$$

$$e_n \pmod{\phi(N_n)} \text{ such that } \gcd(e_n, \phi(N_n)) = 1.$$

- The receiver calculates private keys $d_1, d_2, \dots, d_n$ as

$$d_1 e_1 \equiv 1 \pmod{\phi(N_1)}$$

$$\vdots$$

$$d_n e_n \equiv 1 \pmod{\phi(N_n)}.$$

Now, n pairs (N_i, e_i) for $i = 1, 2, \dots, n$ are published, and private keys $d_1, d_2, \dots, d_n$ are kept secret.

2. **Encryption:**

The above n public key pairs are used for the encryption of plaintext M as follows:

$$\begin{aligned} C_1 &\equiv (M)^{e_1} \pmod{N_1} \\ &\vdots \\ C &= C_n \equiv (C_{n-1})^{e_n} \pmod{N_n}. \end{aligned}$$

In general, for all $1 \leq i \leq n$, we write

$$C_i \equiv (C_{i-1})^{e_i} \pmod{N_i},$$

where C_0 is the plaintext M while C_n is the ciphertext C .

3. **Decryption:**

The ciphertext C can be decrypted to obtain the original message M using private keys $d_1, d_2, \dots, d_n$. The decryption process is as follows.

$$\begin{aligned} C_{n-1} &\equiv (C)^{d_n} \pmod{N_n} \\ &\vdots \\ C_1 &\equiv (C_2)^{d_2} \pmod{N_2} \\ M &\equiv (C_1)^{d_1} \pmod{N_1}. \end{aligned}$$

In general, for all $1 \leq i \leq n$, we write

$$C_{i-1} \equiv (C_i)^{d_i} \pmod{N_i},$$

where $C_0 = M$ and $C_n = C$.

4. Security and efficiency of the proposed generalized multi-moduli RSA cryptosystem

Now, we discuss about the security and the computational or the time efficiency of our proposed GMMRSA.

4.1 Security comparison between various multi-prime RSA and the proposed GMMRSA

Here we exhibit two security attributes of our proposed multi-moduli RSA cryptosystem. Firstly, we show that the surge in the security of the proposed scheme using n -moduli becomes n -fold as compared to the RSA cryptosystem with a single modulus having two prime factors. Secondly, we discuss that our proposed multi-moduli RSA approach is more preferable and secure compared to any multi-prime RSA cryptosystem with a single modulus.

Consider the GMMRSA with two moduli N_1 and N_2 , which are the products $p_1 q_1$ and $p_2 q_2$ respectively, where all the four primes are distinct. The public key pairs are (N_1, e_1) and (N_2, e_2) , in which e_1 and e_2 are the encryption exponents coprime to $\phi(N_1)$ and $\phi(N_2)$ respectively. The private keys d_1 and d_2 are decryption exponents which inverses of e_1 and e_2 modulo $\phi(N_1)$ and $\phi(N_2)$ respectively. Given a plaintext message M , encryption is given by

$$\begin{aligned} C_1 &\equiv M^{e_1} \pmod{N_1}, \\ C &= C_2 \equiv C_1^{e_2} \pmod{N_2}, \end{aligned}$$

where C is the ciphertext. The decryption process is carried out as follows.

$$\begin{aligned} C_1 &\equiv C_2^{d_2} \pmod{N_2}, \\ M &= C_0 \equiv C_1^{d_1} \pmod{N_1}. \end{aligned}$$

Now, suppose an adversary has factored $N_2 = p_2 q_2$ and hence has determined the value d_2 from $\phi(N_2)$ by solving the congruence $e_2 d_2 \equiv 1 \pmod{\phi(N_2)}$. Then they can determine C_1 as follows.

$$C^{d_2} = C_2^{d_2} \equiv (C_1^{e_2})^{d_2} \equiv C_1 \pmod{N_2}.$$

However, recovering M from $C_1 = M^{e_1}$ is equivalent to breaking the RSA cryptosystem with public key (N_1, e_1) and private key d_1 . This would require the adversary to factor N_1 .

On the other hand, if an adversary has determined d_1 by factoring $N_1 = p_1 q_1$, then they can recover the message M from $C_1 = M^{e_1}$ as follows.

$$C_1^{d_1} \equiv (M^{e_1})^{d_1} \equiv M \pmod{N_1}.$$

However, observe that the adversary does not know the value of C_1 yet. Recovering C_1 from the intercepted ciphertext $C = C_2$ amounts to solving the congruence

$$C_1 \equiv C_2^{d_2} \pmod{N_2}$$

which is again equivalent to breaking the RSA cryptosystem with public key (N_2, e_2) and private key d_2 , where C_1 acts as a plaintext message. This would require an adversary to factor N_2 . Using the ciphertext $C_2 = C$, an adversary may also attempt to compute

$$C^{d_1} \equiv (C_1^{e_2})^{d_1} \equiv (M^{e_1 e_2})^{d_1} \equiv M^{e_2} \pmod{N_1}.$$

Again, determining the message M from the above amounts to factoring N_2 . Thus, in any case, even if one of the moduli is factored, i.e., if one of the primes is compromised, then the two-moduli GMMRSA remains as secure as the RSA cryptosystem. In other words, breaking the GMMRSA with two moduli N_1 and N_2 would essentially require factoring both of them.

In general, for a GMMRSA with n -moduli, if the modulus $N_i = p_i q_i$, ($i = 1, 2, \dots, n-1$), is factored, then an adversary can compute d_i and hence can recover C_{i-1} from C_i as follows.

$$C_{i-1} \equiv C_i^{d_i} \equiv (C_{i-1}^{e_i})^{d_i} \pmod{N_i}.$$

However, since $i < n$, determining C_i needs an additional knowledge of C_{i+1} besides factoring $N_{i+1} = p_{i+1} q_{i+1}$. In case the modulus $N_n = p_n q_n$ is factored, then an adversary can determine C_{n-1} but again determining C_{n-2} would essentially require factoring N_{n-2} .

Evidently, breaking the proposed n -moduli ($n \geq 2$) GMMRSA amounts to breaking n different RSA cryptosystems with single modulus having two prime factors. Consequently, its security becomes n -fold as that of the usual two-prime RSA.

One of the primary goals of all multi-prime RSA cryptosystems and its variants is to increase the security of RSA by increasing number of prime factors of the RSA modulus. Although, this would result in security loss in practice except in some cases. In day-to-day communication or transmission of information, the size of the RSA modulus has to be kept optimal so that it cannot be easily factorized by best known methods but at the same time, the operations like key generation, encryption, and decryption can be done in feasible time. Due to the current limitations of the technology and computing power, the optimal size of RSA modulus that is being used in practice is 2048-bit.

One of the best methods known to factorize an RSA modulus is the Lenstra's ECF method which has an expected runtime given as

$$E[N, p] = (\log_2 N)^2 e^{\frac{1}{(2\log(p)\log(\log(p)))^2}}$$

which is as mentioned in [4], where p is a prime factor of the RSA modulus N . Hence, the expected runtime for factorization of N depends on the size of both N and p . When using the Lenstra's method to factor a r -prime k -bit RSA modulus N , since size of the prime factors will be less than that of k -bits 2-prime RSA, the time taken will decrease significantly. It is also evident that as the number of primes in the RSA modulus N rises, it gets easier to factorize N using Lenstra's method, thereby compromising the multi-prime RSA's security. Let us understand this by means of an example.

Example 1: (Lenstra's method for factorization). Let us fix the size of the integer which is to be factorized to be 55-bit. Let $N = 300594712466212949$ be a 55-bit integer having four prime factors $p_1 = 32749, p_2 = 21121, p_3 = 26423$ and $p_4 = 16447$. Let $N' = 18030821594987827$ be another 55-bit integer to be factorized, where $18030821594987827 = 537164267 \times 33566681$.

Now, we shall use Lenstra's method to factorize both N and N' .

Let us first factorize $N = 300594712466212949$ into product of primes.

- The coefficient A of X in elliptic curve $E: Y^2 = X^3 + AX + B$ to be 432535 and let $P = (435, 4245)$ be a point on E . Using Lenstra's algorithm, we get $43!P = \mathcal{O}$ and 300594712466212949 is factorized as $32749 \times 9178744769801$, where 32749 is a prime factor of $N = 300594712466212949$. Time taken to find one factor of 300594712466212949 is $t_1 = 0.016135692596435547$ second. Note that since 9178744769801 is a composite number, we need to further factorize it.
- Now, let $N = 9178744769801$, coefficient of X in elliptic curve to be 2 and point $P = (535, 456)$. Using Lenstra's algorithm, we get $83!P = \mathcal{O}$ and 9178744769801 is factorized as 21121×434579081 , where 21121 is a prime factor of $N = 9178744769801$. Time taken to find one factor of 9178744769801 is $t_2 = 0.028803348541259766$ second. Note that since 434579081 is a composite number, we need to further factorize it.

- Let $N = 434579081$, coefficient of X in elliptic curve to be 535 and point $P = (324, 8)$. Using Lenstra's algorithm, we get $59!P = \mathcal{O}$ and 434579081 is factorized into prime factors as 26423×16447 .

Time taken to factorize 434579081 is $t_3 = 0.011239290237426758$ second.

Hence, total time taken to factorize $N = 300594712466212949$ is

$$\begin{aligned} t &= t_1 + t_2 + t_3 \\ &= 0.016135692596435547 + 0.028803348541259766 \\ &\quad + 0.011239290237426758 \\ &= 0.05617833137512207 \text{ seconds.} \end{aligned}$$

Now, we factorize $N' = 18030821594987827$ into product of primes.

The coefficient of X in elliptic curve to be 646 and point $P = (42368, 685)$. Using Lenstra's algorithm, we get $66809!P = \mathcal{O}$ and N' is factorized into prime factors as $537164267 \times 33566681$. The time taken to factorize 18030821594987827 is 84.53681373596191 seconds.

Thus, as demonstrated in the above example, it is evident that as the number of prime factors of the RSA modulus of a given size increases, the size of its prime factors reduces, thereby making it comparatively easy to factorize and hence break the RSA. To get optimum security using r -prime RSA, it is advisable that the number of prime factors in its modulus must not increase than a "safe value" such that it cannot be easily factorized either by the NFS or the Lenstra's elliptic curve method. Hinek [14] gives a table, given below, which shows the maximum safe number of prime factors for the respective RSA modulus size.

Table 1
Approximate values of safe primes for given sizes of modulus as in [14].

Size of the Modulus	1024 bits	2048 bits	4096 bits	8192 bits
Highest possible number of safe primes	3	3	4	5

As mentioned earlier, in practice, a 2048-bit modulus is widely used for the RSA cryptosystem. From Table 1, it can be seen that the highest number of primes that is safe to be used with 2048-bit modulus is 3. It can be noted that the security of an r -prime RSA is the same for all the values of r with respect to both the NFS and the Lenstra's method. However, if the number of prime factors is beyond the safe value r , then the RSA becomes less secure and, in fact, its security is inversely proportional to r .

While using even number of primes in a multi-prime RSA scheme, the smallest number of primes that can be used is 4. However, in a 2048-bit RSA modulus, using 4 primes renders an insecure cryptosystem as it is more than the safe value. On the other hand, in our proposed generalized multi-moduli RSA scheme, 4 primes generate 2 moduli, each of which is a product of a pair of primes and can be of same size as an RSA modulus, i.e. 2048-bit. This allows us to use larger prime numbers (of the same size as used in 2-prime RSA) than those being used in any multi-prime RSA.

Note that even if one of the moduli used in our GMMRSA with n moduli is factored, the message M is still secure as it is protected by the other $n - 1$ moduli.

Another form of attack mentioned in [2, §2.1] is that using the same modulus N for many parties, it is possible to factorize N . But in our proposed GMMRSA scheme with moduli $N_1, N_2, \dots, N_n$; even if one modulus, say N_1 , is used for more than one implementation, then the other modulus, if utilized only once, ensures that our proposed GMMRSA scheme remains secure against the attack cited above. Thus, such an attack fails on our proposed cryptosystem.

4.2 Comparison of the time-efficiency of the GMMRSA with various multi-prime RSA

For any multi-prime RSA cryptosystem, the immediate concern beside its security is its time-efficiency. It should be made sure that the usage of multiple primes in RSA does not make it computationally infeasible. That is, while aiming to improve the security of the scheme, the increment in the number of computations in the scheme should not rise to a larger extent so as to belittle the security advantage. In this section, we compare the computational efficiency of GMMRSA with some multi-prime RSA cryptosystems discussed above. We first compute and compare the time taken in various multi-prime RSA schemes with the GMMRSA for 4, 6, 8, and 10 primes. Thereafter, we actually give an account of the computational cost, i.e., the number of computations in the various schemes and establish that our proposed GMMRSA does not incur any significant additional computational cost as compared to the other multi-prime RSA cryptosystems taken into consideration.

Table 2
Time consumption for different calculations using different algorithms for four prime RSA

Name of the Scheme	Chosen primes and computed modulus	Size of modulus (in bits)	Time taken (in microseconds)		
			Key generation	Encryption	Decryption
Collins [7]	$p_1 = 11, p_2 = 17,$ $p_3 = 19, p_4 = 23$ $\Rightarrow N = 81719.$	17	19.6	8.6	7.6
Tariq Shah [12]			29.0	15.7	13.9
Ghazali [8]			15.7	7.2	7.0
Proposed Generalized multi-moduli RSA	$p_1 = 251, q_1 = 277$ $\Rightarrow N_1 = 69527,$ $p_2 = 311, q_2 = 337$ $\Rightarrow N_2 = 104807.$		23.4	14.7	12.4

We give below, by means of an example, a comparison of time taken for the key generation as well as the time taken for the encryption and the decryption for three multi-prime RSA, namely, due to Collins et al. [7], the generalized $2n$ -prime RSA due to Tariq Shah et al. [12] and the modified variant due to Ghazali et al. [8] with our GMMRSA using four, six, eight, and ten primes with

different size of modulus using a Python program. This comparison is given in Tables 2, 3, 4 and 5. We also illustrate this comparison as a comparative bar chart in Figures 1, 2, 3 and 4. Despite using multiple primes as factors of n different moduli instead of one, it can be observed that time taken for calculations are just nearly n -fold of that of other multi-prime RSA using a single modulus.

Here, all the times are in microseconds and rounded off to 1 digit after decimal place.

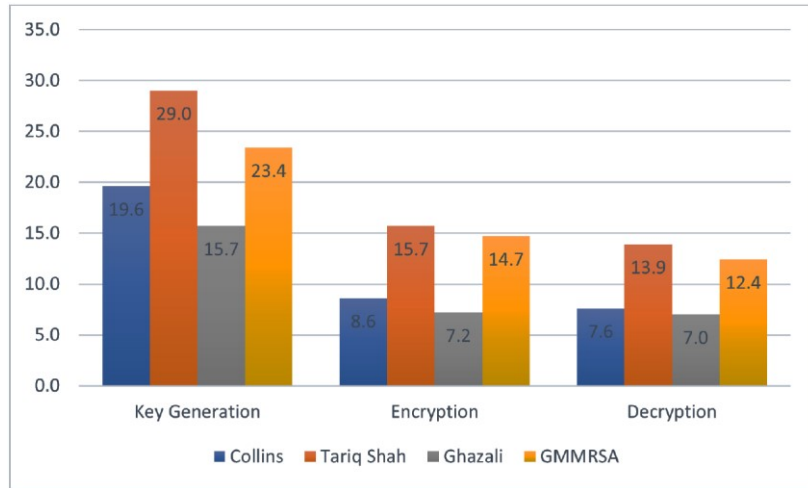


Figure 1

Time consumption for different calculations using different algorithms for four prime RSA

Table 3

Time consumption for different calculations using different algorithms for six prime RSA

Name of the scheme	Chosen primes and computed modulus	Size of modulus (in bits)	Time taken (in microseconds)		
			Key Generation	Encryption	Decryption
Collins [7]	$p_1 = 3, p_2 = 5, p_3 = 7, p_4 = 11, p_5 = 13, p_6 = 17 \Rightarrow N = 255255.$	18	16.2	8.3	6.3
Tariq Shah [12]			33.4	16.8	17.8
Ghazali [8]			14.2	8.0	6.5
Proposed Generalized multi-moduli RSA	$p_1 = 313, q_1 = 457 \Rightarrow N_1 = 143041$ $p_2 = 401, q_2 = 499 \Rightarrow N_2 = 200099$ $p_3 = 373, q_3 = 509 \Rightarrow N_3 = 189857$		31.3	16.4	17.1

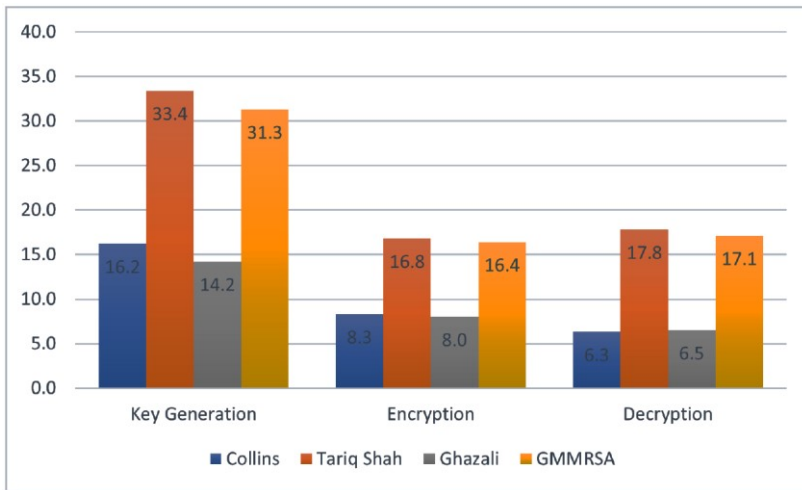


Figure 2
Time consumption for different calculations using different algorithms for six prime RSA

Table 4
Time consumption for different calculations using different algorithms for eight prime RSA

Name of the Scheme	Chosen primes and computed modulus	Size of modulus (in bits)	Time taken (in microseconds)		
			Key generation	Encryption	Decryption
Collins [7]	$p_1 = 3, p_2 = 5, p_3 = 7,$ $p_4 = 11, p_5 = 13, p_6 = 17,$ $p_7 = 19, p_8 = 23$ $\Rightarrow N = 111546435.$	27	18.8	8.1	9.8
Tariq Shah [12]			35.7	18.1	31.5
Ghazali [8]			15.9	8.4	9.7
Proposed Generalized multi-moduli RSA			36.5	17.8	31.3

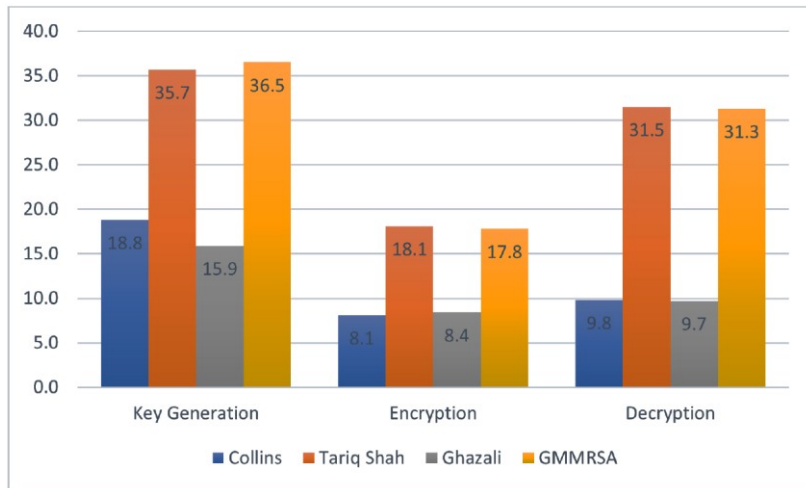


Figure 3
Time consumption for different calculations using different algorithms for eight prime RSA

Table 5
Time consumption for different calculations using different algorithms for ten prime RSA

Name of the Scheme	Chosen primes and computed modulus	Size of modulus (in bits)	Time taken (in microseconds)		
			Key generation	Encryption	Decryption
Collins [7]	$p_1 = 17, p_2 = 11,$ $p_3 = 23, p_4 = 7,$ $p_5 = 19, p_6 = 3,$ $p_7 = 5, p_8 = 13,$ $p_9 = 29, p_{10} = 31$ $\Rightarrow N = 100280245065.$	37	19.5	9.0	17.2
Tariq Shah [12]			46.6	20.3	87.5
Ghazali [8]			33.6	7.2	19.3
Proposed Generalized multi-moduli RSA			54.3	23.5	89.5

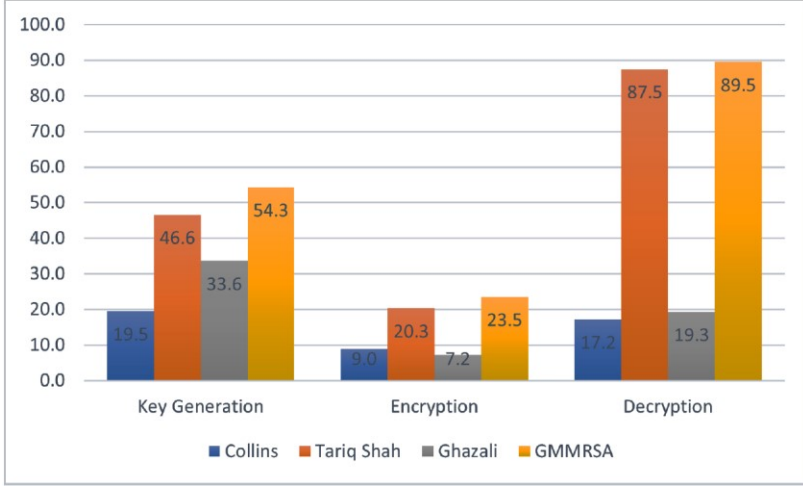


Figure 4
Time consumption for different calculations using different algorithms for ten prime RSA

From the above comparison of some existing multi-prime RSA schemes with our proposed GMMRSA scheme, it is apparent that the key generation time, the encryption time and the decryption time is almost same and depends only on the number of moduli n which is half of the number of primes for a $2n$ -prime RSA. Hence, the proposed GMMRSA scheme incurs no extra computational cost as compared to the multi-prime RSA cryptosystems under consideration.

The first step is key generation which requires computing $\phi(N)$ and decryption exponent. In the systems using a single modulus $N = p_1 p_2 \cdots p_{2n}$, computing $\phi(N)$ takes $2n - 1$ multiplications, whereas in our proposed GMMRSA computing $\phi(N_i)$, $i = 1, 2, \dots, n$, takes one multiplication each and hence a total of n multiplications. Further, the computing $d \equiv e^{-1}(\text{mod } \phi(N))$ takes approximately a multiple of $\log_2(\phi(N))$ steps for the cryptosystems given in [7] and [8] instead of a multiple of $2n \log_2(\phi(N))$ to compute $d_i \equiv e_i^{-1}(\text{mod } \phi(N))$, $(i = 1, 2, \dots, n)$, for [12] and a multiple of $\sum_{i=1}^n 2n \log_2(\phi(N_i))$ in our proposed GMMRSA. Thus, in all the multi-prime RSA cryptosystems it takes $\mathcal{O}(\log_2 \phi(N))$ steps including the proposed GMMRSA cryptosystem, where $N = \max_{1 \leq i \leq n} N_i$. Coming to next step which is encryption, computing the ciphertext C which is $M^e(\text{mod } N)$ incurs a maximum of $2 \log_2(e)$ multiplications using the square-and-multiply algorithm for the cryptosystems [7] and [8] whereas it takes at most $2n \log_2(e)$ multiplications to compute

$$C \equiv ((M^{e_1})^{e_2} \dots)^{e_n} \pmod{N}$$

and at most $2 \log_2(e_1) + 2 \log_2(e_2) + \dots + 2 \log_2(e_n)$ multiplications, i.e., at most $2n \log_2(e)$ multiplications for GMMRSA, where $e = \max_{1 \leq i \leq n} e_i$. Likewise,

the maximum number of multiplications required in the decryption process can be evaluated in all the cryptosystems under consideration. The following table summarizes the computational costs of the cryptosystems under consideration.

Table 6
Comparison of the number of computations among the various multi-prime RSA cryptosystems discussed above

Operation		Multi-prime RSA ($2n$ -primes) proposed by Collins et al. [7] and by Ghazali et al. [8]	Multi-prime RSA by Tariq Shah et al. [12] and our proposed GMMRSA ($2n$ -primes $\approx n$ -moduli)
Key generation	Calculation of $\phi(N)$	$2n - 1$ multiplications	Tariq Shah: $2n - 1$ multiplications GMMRSA: n multiplications
	Inverting e modulo $\phi(N)$	$\mathcal{O}(\log_2 \phi(N))$ steps	Tariq Shah: $\mathcal{O}(\log_2 \phi(N))$ GMMRSA: $\mathcal{O}(\log_2 \phi(N))$, where $N = \max_{1 \leq i \leq n} N_i$
Encryption		at most $2\log_2(e)$ multiplications	at most $2n\log_2(e)$ multiplications, where $e = \max_{1 \leq i \leq n} e_i$
Decryption		at most $2\log_2(d)$ multiplications	at most $2n\log_2(d)$ multiplications, where $d = \max_{1 \leq i \leq n} d_i$

5. Conclusion

In multi-prime RSA schemes proposed by other researchers, only one RSA modulus is used, which is the product of all the primes used in the scheme. This may lead to some security concerns as explained in this article. However, proposed Generalized Multi-Moduli RSA (GMMRSA) uses multiple moduli that are product of two (or more) primes which enhances the security of the system to multiple folds while also being cost-efficient. Thus, the GMMRSA proposed in this paper seems more preferable over a single modulus multi-prime RSA cryptosystem.

Acknowledgement

The second author acknowledges the financial support of CSIR-HRDG in the form of Junior Research Fellowship. The authors are grateful to Prof. Shigeru Kanemitsu for his valuable comments and suggestions which improved a draft of this paper. The authors thank the anonymous referees for their valuable suggestions and remarks.

References

- [1] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126 (1978).

- [2] D. Boneh, "Twenty years of attacks on the RSA cryptosystem," *Notices Amer. Math. Soc.*, vol. 46, no. 2, pp. 203–213 (1999).
- [3] J. M. Pollard, "Theorems on factorization and primality testing," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 76(3), pp. 521–528, Cambridge University Press (1974).
- [4] H. W. Lenstra Jr, "Factoring integers with elliptic curves," *Annals of mathematics*, pp. 649–673 (1987).
- [5] M. Mumtaz and L. Ping, "Forty years of attacks on the RSA cryptosystem: A brief survey," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 22, no. 1, pp. 9–29 (2019).
- [6] C. Nemo, "RSA Moduli Should Have 3 Prime Factors," *Scientific Bulgarian Magazine*, 1996. <http://fgrieu.free.fr/CaptainNemo.pdf>.
- [7] T. Collins, D. Hopkins, S. Langford, and M. Sabin, "Public key cryptographic apparatus and method," (1998). United States Patent 5,848,159, <https://patents.google.com/patent/US5848159A/en>.
- [8] M. G. Kamardan, N. Aminudin, N. Che-Him, S. Sufahani, K. Khalid, and R. Roslan, "Modified Multi Prime RSA Cryptosystem," in *Journal of Physics: Conference Series*, vol. 995, no. 1, pp. 012030, IOP Publishing (2018).
- [9] T. Pathirage, H. Wijewardana, L. Lakshan, H. Hydher, and L. Yasakethu, "Multiprime RSA verilog implementation using 4-primes," in *2021 10th International Conference on Information and Automation for Sustainability (ICIAfS)*, pp. 60–65, IEEE (2021).
- [10] N. A. A. Sani and H. Kamarulhaili, "RSA cryptography and multi prime RSA cryptography," in *AIP Conference Proceedings*, vol. 1870, no. 1, AIP Publishing (2017).
- [11] S. Tanwar and A. Kumar, "An efficient and secure identity based multiple signatures scheme based on RSA," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 22, no. 6, pp. 953–971 (2019).
- [12] T. Shah, M. Zohaib, Q. Xin, B. Almutairi, and M. Sajjad, "Generalization of RSA cryptosystem based on $2n$ primes," *AIMS Mathematics*, vol. 8, no. 8, pp. 18833–18845 (2023).
- [13] Manu and A. Goel, "Encryption algorithm using dual modulus," in *2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT)*, pp. 1–4 (2017).
- [14] M. J. Hinek, "On the security of multi-prime RSA," *Journal of Mathematical Cryptology*, vol. 2, no. 2, pp. 117–147 (2008).

Received September, 2024