CAUCHY – Jurnal Matematika Murni dan Aplikasi
Volume 10 (2) (2025), Pages 805-820
p-ISSN: 2086-0382; e-ISSN: 2477-3344

Security Analysis of Modified ESRKGS-RSA Using Lenstra's Elliptic Curve Method

Bety Hayat Susanti*, Aditya Sukhoi Lean Sumule, and Mareta Wahyu Ardyani

Department of Cryptographic Engineering, Politeknik Siber dan Sandi Negara, Indonesia

Abstract

The Enhanced and Secure RSA Key Generation Scheme (ESRKGS), introduced in 2014, aimed to improve RSA security by employing a modulus constructed from four prime factors. However, subsequent studies in 2016 revealed that this structure did not provide additional security over standard RSA. In response, a modified version of ESRKGS was proposed in 2021, incorporating dual encoding techniques using a masking parameter γ and double encryption. This study evaluates the security of the modified ESRKGS by simulating an attack scenario in which the adversary is assumed to know of $\phi(N)$, enabling recovery of encrypted messages. Additionally, we implement Lenstra's Elliptic Curve Method (ECM) to assess the factorization resistance of the four-prime modulus when $\phi(N)$ is not known. Experimental results indicate that ECM can efficiently factor the modulus into its four constituent primes under practical time constraints. These findings demonstrate that, despite recent modifications, the ESRKGS variant remains vulnerable to factorization-based attacks. This highlights the necessity for more rigorous cryptographic design principles in multiprime RSA systems and calls into question the long-term viability of ESRKGS-based schemes in high-security applications.

Keywords: ESRKGS; Lenstra algorithm; RSA; RSA factorization attack $\phi(N)$.

Copyright © 2025 by Authors, Published by CAUCHY Group. This is an open access article under the CC BY-SA License (https://creativecommons.org/licenses/by-sa/4.0)

1 Introduction

Asymmetric cryptography, also referred to as *public key cryptography*, is a cryptographic system that utilizes two distinct keys: a public key for encryption and a private key for decryption [1]. Unlike symmetric cryptography, which relies on a single key for both encryption and decryption, public key cryptography differentiates these processes. In this system, the private key d is kept confidential and is used exclusively for decryption, while the public key e is openly shared and used for encryption.

For instance, when an entity Y wishes to send a message m to entity Z, Y uses Z's public key and the encryption function E to generate the ciphertext c, such that:

$$c = E_e(m)$$

Entity Z then applies its private key to decrypt the ciphertext and retrieve the original plaintext:

$$m = D_d(c)$$

*Corresponding author. E-mail: bety.hayat@poltekssn.ac.id

Submitted: February 20, 2025 Reviewed: June 16, 2025 Accepted: August 03, 2025
DOI: https://doi.org/10.18860/cauchy.v10i2.32189

The security of this system fundamentally relies on the computational difficulty of deriving the private key d from the public exponent e, a problem rooted in well-established mathematical constructions [1]. While e is openly distributed, its authenticity must be verified to ensure that it corresponds to the correct private key d and is legitimately associated with the intended recipient Z. This verification step is crucial in preventing man-in-the-middle or impersonation attacks. One of the primary advantages of asymmetric cryptography over symmetric cryptography lies in its ability to eliminate the need for secure key exchange, thereby reducing logistical and security challenges.

Among various public-key algorithms, the RSA scheme remains one of the most widely deployed. Proposed in 1977 by Rivest, Shamir, and Adleman [2]–[4], RSA has become a cornerstone of modern cryptographic systems. The algorithm operates in three fundamental stages: *key generation*, *encryption*, and *decryption*. Its security is predicated on the difficulty of factoring a large composite modulus $N = p \cdot q$ into its prime factors p and q. The computational intractability of this factorization problem underpins the secrecy of the private key.

In practice, increasing the size of N directly enhances security by making factorization exponentially more difficult. However, this improvement comes at the cost of increased computational effort and resource consumption for both encryption and decryption. As a result, numerous variants and optimizations of RSA have been proposed, aiming to strike a balance between cryptographic strength and computational efficiency. These include techniques such as the use of smaller public exponents, optimizations based on the Chinese Remainder Theorem (CRT), and multiprime RSA, each addressing different performance-security trade-offs.

RSA security is challenged by various types of attacks, such as those targeting integer factorization, exploiting quantum computing capabilities, manipulating the RSA function itself, or exploiting vulnerabilities in its implementation [2], [3], [4], [5]. Although initially developed for general public key cryptosystems, these attack methods have shown considerable effectiveness against RSA. In some cases, attacks may also succeed by directly retrieving keys or plaintext, either through weaknesses in the trapdoor mechanism or independently of it, as demonstrated in [6], [7], [8], [9], [10], [11].

In 2015, Thangavel *et al.* proposed a modified RSA scheme designed to increase complexity by incorporating four prime numbers instead of the two used in the conventional RSA algorithm [12]. This modification, termed the Enhanced and Secured RSA Key Generation Scheme (ESRKGS), was claimed to provide significantly higher security than traditional RSA. The ESRKGS scheme generates the modulus N as the product of four prime numbers for key generation. However, encryption and decryption operations are performed using a modulus derived from only two of these prime numbers. Thangavel *et al.* argued that, given N (the public key), an attacker could naively factorize it to obtain two prime factors (p and q), which are insufficient to deduce the private key operating under the four-prime modulus.

However, in 2016, Lüy *et al.* published research that challenged the security claims of the ESRKGS scheme [13]. Their work demonstrated that the use of four prime numbers did not provide any substantial improvement in security over conventional RSA. In fact, Lüy *et al.* proved that the ESRKGS scheme's security level was equivalent to that of standard RSA, refuting the original claims made by Thangavel *et al.*

In 2021, Azzahra introduced a new modification of the ESRKGS scheme [14]. This updated scheme altered the encryption and decryption parameters, which were both performed under a modulus N. Additionally, a new parameter, γ, was introduced as part of the public key, resulting in encryption and decryption processes being executed twice. Azzahra claimed that this modification could mitigate previously identified vulnerabilities in the ESRKGS scheme. Subsequent studies have explored the security and efficiency of the ESRKGS algorithm and its modifications [15], [16], [17].

Building upon this framework, the present study assesses the security of the ESRKGS modification proposed by Azzahra. The analysis demonstrates an attack on the scheme through a

mathematical approach that leverages its structural properties, under the assumption that the attacker has knowledge of $\phi(N)$. For cases where $\phi(N)$ is unknown, an alternative strategy is introduced, employing the Lenstra elliptic curve algorithm to factorize N. Together, these approaches provide a unified evaluation of the modified ESRKGS, revealing potential vulnerabilities that should be addressed before any practical deployment.

The structure of this paper is as follows. Section 1 presents the background of the problem and outlines the theoretical foundations underlying this study, including a review of relevant literature. Section 2 describes the experimental procedures employed. Section 3 discusses the results in detail, emphasizing the most significant findings and their implications for the security of the analyzed schemes. Finally, Section 4 concludes the paper by summarizing the key contributions and outlining potential directions for future research.

RSA Algorithm

Until the late 1970s, only symmetric cryptographic systems were known. However, during the decade from 1970 to 1980, asymmetric cryptographic systems were introduced. One of the most prominent schemes in asymmetric cryptography is the RSA algorithm, developed in 1977 at the Massachusetts Institute of Technology (MIT) by Ronald Rivest, Adi Shamir, and Leonard Adleman [18]. The algorithm was patented in 1983, but the patent expired around 2000, making it freely available for use today.

Like other cryptographic algorithms, RSA is used for both encryption and decryption. However, RSA is less efficient when encrypting long messages, so it is more commonly employed for encrypting short messages [1]. For example, if Bob wants to send a secure message to Alice, Alice will provide Bob with her public key while keeping her private key confidential.

Algorithm 1 RSA Key Generation

Require: Primes p and q

1: Alice generates two large random (and distinct) primes p and q, each roughly of the same size.
2: Alice computes $N = p \cdot q$ and $\phi(N) = (p-1)(q-1)$.
3: Alice selects a random integer e, $1 < e < \phi(N)$, such that $\gcd(e, \phi(N)) = 1$.
4: Alice computes the unique integer d, $1 < d < \phi(N)$, such that $ed \equiv 1 \mod \phi(N)$.
5: **return** Alice's public key, (N, e); and Alice's private key, d.

Algorithm 2 RSA Encryption

Require: Message M and Alice's public key (N, e)

1: Bob obtains Alice's authentic public key (N, e).
2: Bob computes $C \equiv M^e \mod N$.
3: **return** Ciphertext C, which is then sent to Alice.

Algorithm 3 RSA Decryption

Require: Ciphertext C and Alice's private key, d

1: Alice uses her private key d to recover M from C by computing $M \equiv C^d \mod N$.
2: **return** Message M.

ESRKGS Algorithm

The security of the RSA algorithm is rooted in the computational difficulty of factoring large numbers into their prime factors. However, advancements in technology and mathematical

techniques have led to the development of methods that can solve such factorization problems more efficiently. The vulnerability of RSA arises from its reliance on N, a product of two prime numbers. If N is successfully factored, an attacker can easily determine the private keys and gain access to the encrypted information [19].

To address this weakness, Thangavel *et al.* proposed the Enhanced and Secured RSA Key Generation Scheme (ESRKGS) in 2014, aiming to bolster the security of the traditional RSA algorithm [12]. In this scheme, as with RSA, Alice provides Bob with her public key while keeping her private key secure.

Algorithm 4 ESRKGS Key Generation

Require: Primes p, q, r, and s

1: Alice generates four large random (and distinct) primes p, q, r and s, each roughly of the same size.
2: Alice computes $n = p \cdot q$ and $m = r \cdot s$.
3: Alice computes $\phi(n) = (p-1)(q-1)$ and $\phi(m) = (r-1)(s-1)$, then subsequently computes $\phi(N) = \phi(n) \cdot \phi(m)$.
4: Alice chooses e_1, $1 < e_1 < \phi(n)$, with $\gcd(e_1, \phi(n)) = 1$; and e_2, $1 < e_2 < \phi(m)$, with $\gcd(e_2, \phi(m)) = 1$.
5: Alice computes $E_1 \equiv e_1^{e_2} \mod N$.
6: Alice chooses a random number E, $1 < E < \phi(N) \cdot E_1$ such that $\gcd(E, \phi(N) \cdot E_1) = 1$.
7: Alice then computes D such that $D \equiv E^{-1} \mod (\phi(N) \cdot E_1)$.
8: **return** Alice's public key, (N, E); and Alice's private key, D.

Algorithm 5 ESRKGS Encryption

Require: Plaintext $M < N$ and Alice's public key (E, N)

1: Bob obtains Alice's authentic public key (E, N) to encrypt message M.
2: Bob computes $C \equiv M^E \mod N$.
3: **return** Ciphertext C, which is then sent to Alice.

Algorithm 6 ESRKGS Decryption

Require: Ciphertext C and Alice's private key, D

1: Alice receives ciphertext C and performs decryption using her private key D.
2: Alice computes $M \equiv C^D \mod N$.
3: **return** Message M.

Attack on ESRKGS

In 2016, Lüy *et al.* published an attack on the ESRKGS algorithm [13]. Their analysis demonstrated that an attacker could recover the original message M if they could factorize $n = p \cdot q$. Once n is factored into its prime components p and q, the attacker can compute $\phi(n) = (p-1)(q-1)$. Using the public key E, they can exploit the relationship $\gcd(E, \phi(N) \cdot E_1) = 1$.

Here, N is the product of n and another integer m, and $\phi(N)$ is the product of $\phi(n)$ and $\phi(m)$. It follows that $\gcd(E, \phi(N) \cdot E_1) = 1$, meaning E has an inverse modulo $\phi(N)$. Let $D_1 \equiv E^{-1} \mod \phi(N)$. For any message $M < N$, the attacker can compute $C^{D_1} \equiv M^{ED_1} \equiv M \mod (N)$, thereby decrypting the message.

Given the following parameters on ESRKGS:

$$\begin{aligned} p &= 79, q = 101, r = 109, s = 89 \\ n &= 7979, m = 9701, N = 77404279 \\ \phi(n) &= 7800, \phi(m) = 9504, \phi(N) = 74131200 \\ e_1 &= 2761, e_2 = 587 \\ E_1 &= 74034755, E = 442692186722853 \\ D &= 4707099085177517 \\ \text{Public key} &= (442692186722853, 7979) \\ \text{Private key} &= (4707099085177517, 7979) \\ M &= 59, C = 2883 \end{aligned}$$

If an attacker successfully factors $n = 7979$ into its prime components $p = 79$ and $q = 101$, they can easily compute $\phi(n) = (p-1)(q-1) = 7800$. With access to the public key $E = 442692186722853$, the attacker can determine the private key $D_1 \equiv E^{-1} \mod \phi(N) = 3317$. Using this private key, the attacker can decrypt the ciphertext $C = 2883$ to recover the original message:

$$M \equiv C^{D_1} \mod \phi(n) \equiv 2883^{3317} \mod 7979 = 59 \tag{1}$$

Thus, the attacker successfully retrieves the plaintext message.

A Modification on ESRKGS

Azzahra proposed a new cryptographic scheme as a modification of the ESRKGS scheme [14]. The enhancements include changes to the encryption and decryption parameters, both of which are now computed using a modulus N. Additionally, the author introduced a new parameter, γ, as part of the public key. The encryption and decryption processes in this modified scheme are performed twice, with γ playing a key role. The parameter γ is chosen such that $\gcd(\gamma, N) = 1$, and its inverse, γ^{-1}, becomes a component of the private key. Azzahra claims that the addition of the parameter γ in the encryption function increases the difficulty for an attacker to determine the message M, since the attacker must also be able to find γ^{-1}, which is required in the decryption process [14].

Furthermore, the key generation process in Azzahra's proposed scheme incorporates a notable enhancement through the generation of two distinct key pairs: (E, D) and (g, t). Here, D and t are the modular inverses of E and g, respectively, with respect to $\phi(N)$. Both pairs are actively employed during the encryption and decryption phases, with (E, D) functioning as the primary RSA-like exponent pair, while (g, t) introduces an additional transformation stage.

The inclusion of this secondary key pair (g, t) is designed to add an extra cryptographic layer, thereby increasing the complexity of potential attack vectors. This dual-key strategy directly addresses the vulnerabilities observed in the original ESRKGS scheme, particularly its deterministic structure and susceptibility to specific cryptanalytic techniques. By diversifying the encryption process and decoupling key dependencies, the modified scheme aims to improve resilience against both classical and adaptive attacks. In essence, this modification seeks to preserve the intended efficiency of ESRKGS while significantly enhancing its robustness in practical deployment scenarios.

The following section outlines the key generation, encryption, and decryption procedures in Azzahra's proposed scheme. Each step is presented to emphasize the essential computational operations and parameter relationships that define the scheme's structure and functionality.

Algorithm 7 Modified ESRKGS Key Generation

Require: Primes p, q, r, and s

1: Choose random primes p, q, r and s.
2: Compute $n = p \cdot r$, $m = q \cdot s$, and $N = n \cdot m = p \cdot r \cdot q \cdot s$.
3: Compute $\phi(n) = (p-1)(r-1)$, $\phi(m) = (q-1)(s-1)$, and $\phi(N) = \phi(n) \cdot \phi(m)$.
4: Choose a random integer e_1, $\frac{\phi(n)}{2} < e_1 < \phi(n)$, such that $\gcd(e_1, \phi(n)) = 1$.
5: Choose a random integer e_2, $\frac{\phi(n)}{2} < e_2 < \phi(n)$, such that $\gcd(e_2, \phi(n)) = 1$.
6: Compute $E_1 \equiv e_1^{e_2} \mod N$.
7: Compute $bb = \phi(N) \times E_1$.
8: Choose a random integer E, $\frac{bb}{2} < E < bb$, such that $\gcd(E, bb) = 1$.
9: Choose a random integer g, $\frac{bb}{2} < g < bb$, such that $\gcd(g, bb) = 1$.
10: Compute $D \equiv E^{-1} \mod bb$.
11: Compute $t \equiv g^{-1} \mod bb$.
12: Compute $n_1 = \frac{N}{n}$ and $m_1 = \frac{N}{m}$.
13: Compute $n_1^{(} = -1$ and m_1^{-1} such that $n_1 \cdot n_1^{-1} \equiv 1 \mod n$ and $m_1 \cdot m_1^{-1} \equiv 1 \mod m$.
14: Choose two primes j_1 and j_2 then compute j_1^{-1} dan j_2^{-1} such that $j_1 \cdot j_1^{-1} \equiv 1 \mod n$ and $j_2 \cdot j_2^{-1} \equiv 1 \mod m$.
15: Compute the encryption parameter γ by using the Chinese Remainder Theorem such that $\gamma = ((j_1 \cdot n_1 \cdot n_1^{-1}) + (j_2 \cdot m_1 \cdot m_1^{-1})) \mod N$.
16: Verify if $\gcd(\gamma, N) = 1$. If $\gcd(\gamma, N) \neq 1$, then choose another j_1 and j_2.
17: Compute γ^{-1} such that $\gamma \cdot \gamma^{-1} \equiv 1 \mod N$.
18: **return** Public key (E, γ, g, N); and private key (D, γ^{-1}, t).

Algorithm 8 Modified ESRKGS Encryption

Require: Plaintext M with $\gcd(M, N) = 1$ and public key (E, γ, g, N)

1: Compute $C_1 \equiv (\gamma \cdot (M^E \mod N)) \mod N$.
2: Compute $C_2 \equiv C_1^g \mod N$.
3: **return** Ciphertext C_2.

Algorithm 9 Modified ESRKGS Decryption

Require: Ciphertext C_2 and private key (D, γ^{-1}, t)

1: Compute $CT_1 \equiv C_2^t \mod N$.
2: Compute $M \equiv ((CT_1 \cdot \gamma^{-1}) \mod N)^D \mod N$.
3: **return** Message M.

Table 1 presents a comparative overview of three RSA-based cryptographic schemes: Classic RSA, ESRKGS, and Modified ESRKGS. It illustrates how each variant alters the original RSA structure, the rationale behind these modifications, and their impact on overall security.

Table 1: Comparison of RSA Variant

Scheme	Number of Primes	Key Features	Known Weakness	Security Assumption
Classic RSA	2	$\phi(N)$, large e/d	ECM, QS, sub-exp factorization	Integer factorization
ESRKGS	4	Split decryption via CRT	No extra hardness from extra primes	Integer factorization
Modified ESRKGS	4	γ, double encryption	No added security from ESRKGS	Integer factorization

Classic RSA employs a modulus with two prime factors and relies on the hardness of integer factorization. However, it remains vulnerable to sub-exponential factorization attacks such as the Elliptic Curve Method (ECM) and the Quadratic Sieve (QS). ESRKGS attempts to improve decryption efficiency by using a four-prime modulus and applying the Chinese Remainder Theorem (CRT), yet this structural change does not enhance its cryptographic strength. The Modified ESRKGS further introduces a new parameter, γ, along with a double encryption mechanism. Despite these additions, it similarly fails to provide significant resistance against known attacks. Ultimately, although ESRKGS and its modified version introduce structural variations, they do not offer a tangible security advantage over Classic RSA, as all three schemes rely on the same factorization-based security foundation.

2 Methods

The methods outlined in this section form the basis for the experimental procedures applied when $\phi(N)$ is unknown, relying on the Lenstra elliptic curve algorithm to factorize N. By presenting the parameter selection and algorithmic steps, this section provides the essential background for understanding the cryptographic constructs and attack models examined in the subsequent sections.

Lenstra Algorithm

The Lenstra algorithm, introduced by mathematician Hendrik Lenstra in 1987, is a factorization method designed to decompose large integers into smaller prime factors [20]. For instance, given a large integer N, the algorithm aims to find prime numbers p and q such that $N = p \cdot q$. This algorithm leverages principles from number theory and algebra, with a particular focus on elliptic curve theory, to accelerate the factorization process.

An elliptic curve is a cubic equation in two variables, resembling those used to calculate the lengths of curves. The general form of an elliptic curve equation is:

$$y^2 = x^3 + ax + b \tag{2}$$

For the equation above to represent a nonsingular elliptic curve, it must satisfy the condition $4a^2 + 27b^2 \neq 0$. The unique properties of nonsingular elliptic curves enable the definition of an addition operation for points on the curve [8]. However, this addition operation differs fundamentally from the standard addition of integers. Instead, it involves combining two points on the curve to produce a third point. The process of point addition on an elliptic curve is classified into three distinct cases:

1. In the first case, two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ have two distinct x coordinates ($x_1 \neq x_2$). The formulas for the slope (λ) and the resulting coordinates (x_3, y_3) of the third point are given as follows:

$$\begin{aligned} \lambda &\equiv (y_2 - y_1) \cdot (x_2 - x_1)^{-1} \pmod{N} \\ x_3 &\equiv \lambda^2 - x_1 - x_2 \pmod{N} \\ y_3 &\equiv \lambda(x_1 - x_3) - y_1 \pmod{N} \end{aligned}$$

   If $(x_2 - x_1)$ does not have an inverse modulo N, then $\gcd((x_2 - x_1), N) \neq 1$. In such cases, N can be factored.

2. In the second case, two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ are identical ($x_1 = x_2$ and $y_1 = y_2$). The formulas for the slope (λ) and the resulting coordinates (x_3, y_3) of the third point are are calculated as follows:

$$\begin{aligned} \lambda &\equiv (3y_1 + a)(2y_1)^{-1} \pmod{N} \\ x_3 &\equiv \lambda^2 - x_1 - x_2 \pmod{N} \\ y_3 &\equiv \lambda(x_1 - x_3) - y_1 \pmod{N} \end{aligned}$$

If $2y_1$ does not have an inverse modulo N, then $\gcd(2y_1, N) \neq 1$, and N can be factored.

3. In the third case, two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ are additive inverses of each other. The line connecting these points does not intersect the curve at a third point. Instead, it is said to intersect at the point at infinity, denoted as $\mathcal{O}$. This point $\mathcal{O}$, also known as the zero point, serves as the identity element of the elliptic curve group.

3 Results and Discussion

To critically evaluate Azzahra's proposed modification of the ESRKGS algorithm, it is important to first understand both the intended security enhancements and their cryptographic implications. The modification introduces additional random parameters within specified ranges and applies the encryption procedure twice, presumably to strengthen the scheme. However, such adjustments do not necessarily yield significant improvements in complexity or resistance to attacks, and the private key construction remains susceptible to derivation from public information.

This section presents a structured analysis of potential vulnerabilities in the modified ESRKGS algorithm. We begin by outlining two main attack vectors that exploit the algorithm's mathematical properties and public key structure. The first focuses on deriving the decryption key under certain assumptions, while the second leverages the formulation of the public key to recover the claimed private key. Following this conceptual analysis, we proceed to a detailed mathematical derivation demonstrating how an adversary can successfully retrieve the original message using publicly available parameters and standard number-theoretic techniques.

1. *Deriving the Decryption Key*
   Assume an attacker can determine the value of $\phi(n)$. The public key E satisfies $\gcd(E, \phi(N) \cdot E_1) = 1$, where $E_1 = e_1^{e_2} \mod N$. Since $\phi(N) \cdot E_1$ is the product of $\phi(N)$ and E_1, it follows $\gcd(E, \phi(N)) = 1$. This implies that E has an inverse modulo $\phi(N)$, denoted as D_1. As a result, for any message $M < N$, $C^{D_1} \equiv M^{ED_1} \equiv M \mod N$.
2. *Exploiting the Public Key Structure*
   The public key γ is defined as $\gamma = ((j_1 \cdot n_1 \cdot n_1^{-1}) + ((j_2 \cdot m_2 \cdot m_2^{-1})) \mod N$, where parameters j_1, j_2, n_1, m_1 are generated randomly within certain ranges. However, the claimed private key γ^{-1} is computed modulo N. This means that, given sufficient knowledge of the public key, an attacker can determine γ^{-1} and compromise the security of the system.

Mathematical Analysis

To substantiate the conceptual attack description, we now present a detailed mathematical analysis of the modified ESRKGS algorithm. This analysis formalizes the attack steps using number-theoretic properties and modular arithmetic, showing explicitly how the decryption key can be derived and how the original plaintext can be recovered from the ciphertext. By systematically expressing the encryption and decryption processes in algebraic form, the relationships between the public key parameters, the ciphertext components, and the private key elements become evident, enabling the complete reconstruction of the message without legitimate authorization.

We begin by defining the key parameters, the ciphertext structure, and the known value available to the attacker:

$$\begin{aligned} \text{Public key} &= (E, \gamma, g, N), \\ \text{Ciphertext} &= C_2 = C_1^g \mod N, \\ \text{Known value} &= \phi(N). \end{aligned}$$

From the ciphertext definition, we can express C_2 directly in terms of C_1 and subsequently

expand C_1 in terms of M, E, and γ:

$$C_2 \equiv C_1^g \mod N \tag{3}$$
$$C_2 \equiv ((\gamma \cdot (M^E \mod N)) \mod N)^g \mod N. \tag{4}$$

Rearranging the expression allows us to separate the contribution of γ and M in the exponentiated form:

$$C_2 \equiv (\gamma^g \cdot (M^{Eg} \mod N)) \mod N \tag{5}$$
$$(\gamma^{-1})^g \equiv \gamma^{-g} \mod N. \tag{6}$$

By multiplying Equations (4) and (6), we eliminate the γ factor, leaving only the term involving M:

$$C_2 \cdot (\gamma^{-1})^g \equiv \left(\gamma^g \cdot (M^{Eg} \bmod N)\right) \bmod N \cdot \gamma^{-g} \bmod N \tag{7}$$
$$C_2 \cdot (\gamma^{-1})^g \equiv M^{Eg} \bmod N. \tag{8}$$

Next, we define the modular inverses of E and g with respect to $\phi(N)$, denoted by D_1 and t_1 respectively:

$$D_1 \equiv E^{-1} \bmod \phi(N) \tag{9}$$
$$t_1 \equiv g^{-1} \bmod \phi(N). \tag{10}$$

Multiplying these inverses yields the inverse of the product Eg modulo $\phi(N)$:

$$D_1 \cdot t_1 \equiv \left(E^{-1} \bmod \phi(N)\right) \cdot \left(g^{-1} \bmod \phi(N)\right) \tag{11}$$
$$D_1 \cdot t_1 \equiv (Eg)^{-1} \bmod \phi(N). \tag{12}$$

Finally, raising Equation (8) to the power of Equation (12) recovers the original message M:

$$\left(C_2 \cdot (\gamma^{-1})^g\right)^{D_1 \cdot t_1} \equiv \left(M^{Eg} \bmod N\right)^{(Eg)^{-1}} \tag{13}$$
$$(C_2 \cdot (\gamma_{\text{inv}})^g)^{D_1 \cdot t_1} \equiv M \bmod N. \tag{14}$$

Implementation of Modified ESRKGS

This section presents the implementation of the Modified ESRKGS algorithm, covering the key generation, encryption, and decryption processes. The modifications are designed to enhance security without compromising computational efficiency. We detail the underlying mathematical principles, parameter selection, and procedural steps to demonstrate how the scheme operates in practice. Furthermore, we examine the security implications of these modifications. All parameters used in the implementation are generated independently by the authors.

Key Generation

In this phase, Alice generates the core cryptographic parameters and computes the corresponding public and private keys, which form the foundation of the encryption and decryption processes.

1. Alice chooses four primes: $p = 79,\ q = 101,\ r = 109,\ s = 89$.
2. Alice computes:

$$n = p \cdot q = 7979, \quad m = r \cdot s = 9701, \quad N = n \cdot m = 77404279.$$

3. Alice computes:

$$\phi(n) = (p-1)(q-1) = 7800, \quad \phi(m) = (r-1)(s-1) = 9504,$$

then

$$\phi(N) = \phi(n) \cdot \phi(m) = 74131200.$$

4. Alice chooses e_1 where $\frac{\phi(n)}{2} < e_1 < \phi(n)$ and $\gcd(e_1, \phi(n)) = 1$, with $e_1 = 2761$.
5. Alice chooses e_2 where $\frac{\phi(m)}{2} < e_2 < \phi(m)$ and $\gcd(e_2, \phi(m)) = 1$, with $e_2 = 587$.
6. Alice computes:

$$E_1 = e_1^{e_2} \bmod N = 74034755.$$

7. Alice computes:

$$bb = \phi(N) \times E_1 = 5488285229856000.$$

8. Alice chooses E where $\frac{bb}{2} < E < bb$ and $\gcd(E, bb) = 1$, with $E = 2744142614928019$.
9. Alice chooses g where $\frac{bb}{2} < g < bb$ and $\gcd(g, bb) = 1$, with $g = 3744142614928009$.
10. Alice computes:

$$D \equiv E^{-1} \pmod{bb} = 5343856671175579,$$

11. Alice computes:

$$t \equiv g^{-1} \pmod{bb} = 1088432811336889.$$

12. Alice computes:

$$n_1 = \frac{N}{n} = 9701, \quad m_1 = \frac{N}{m} = 7979.$$

13. Alice computes:

$$n_1^{-1} \pmod{n} = 3313, \quad m_1^{-1} \pmod{m} = 5673.$$

14. Alice chooses two primes $j_1 = 31$ and $j_2 = 37$, then computes:

$$j_1^{-1} \pmod{n} = 4633, \quad j_2^{-1} \pmod{m} = 5506.$$

15. Alice computes:

$$\gamma = \left((j_1 \cdot n_1 \cdot n_1^{-1}) + (j_2 \cdot m_1 \cdot m_1^{-1})\right) \bmod N = 39376396,$$

16. Alice computes:

$$\gamma^{-1} \pmod{N} = 40051234.$$

17. The public key is:

$$(2744142614928019,\ 3744142614928009,\ 39376396,\ 77404279).$$

18. The private key is:

$$(5343856671175579,\ 1088432811336889,\ 40051234,\ 77404279).$$

Encryption

Once the keys have been generated, Bob can encrypt his message using Alice's public key to produce a secure ciphertext.

1. Bob uses Alice's authentic public key (E, g, γ, N) to encrypt a message $M < n$, where $M = 59$.

2. Bob computes:

$$C_1 = \Big(\gamma \cdot (M^E \bmod N)\Big) \bmod N = 50594777$$
$$C_2 = C_1^g \bmod N = 71485117.$$

Decryption

After receiving the ciphertext, Alice uses her private key to reverse the encryption process and recover the original plaintext message.

1. Alice receives the ciphertext C and decrypts the message using her private key (D, t, γ^{-1}) with $C_2 = 71485117$.
2. Alice computes:

$$CT_1 = C_2^t \bmod N = 50594777$$
$$M = \Big((CT_1 \cdot \gamma^{-1}) \bmod N\Big)^D \bmod N = 59.$$

Attack Simulation

Using a small-scale mathematical example, the attack is simulated with the following parameters:

1. $p = 79, \quad q = 101, \quad r = 109, \quad s = 89$
2. $n = 7979, \quad m = 9701, \quad N = 77404279$
3. $\phi(n) = 7800, \quad \phi(m) = 9504, \quad \phi(N) = 74131200$
4. $e_1 = 2761, \quad e_2 = 587, \quad E_1 = 74034755$
5. $bb = 5488285229856000$
6. $E = 2744142614928019, \quad g = 3744142614928009$
7. $D = 5343856671175579, \quad t = 1088432811336889$
8. $n_1 = 9701, \quad m_1 = 7979$
9. $n_1^{-1} \bmod n = 3313, \quad m_1^{-1} \bmod m = 5673$
10. $j_1 = 31, \quad j_2 = 37$
11. $j_1^{-1} \bmod n = 4633, \quad j_2^{-1} \bmod m = 5506$
12. $\gamma = 39376396, \quad \gamma^{-1} \bmod N = 40051234$
13. Public key: (2744142614928019, 3744142614928009, 39376396, 77404279)
14. Private key: (5343856671175579, 1088432811336889, 40051234, 77404279)
15. Message: $M = 59$
16. Ciphertext: $C_2 = 71485117$

If an attacker obtains the value of $\phi(N)$, they can exploit the algorithm to recover the message M using the ciphertext C_2 and the public key (E, g, γ, N). The attack proceeds as follows:

1. Given public key $(E, g, \gamma, N) = (2744142614928019, 3744142614928009, 39376396, 77404279)$ and the ciphertext $C_2 = 71485117$.
2. $\phi(N) = 74131200$.
3. Compute $D_1 = E^{-1} \bmod \phi(N) = 72180379$,
4. Compute $t_1 = g^{-1} \bmod \phi(N) = 58844089$
5. Compute $D_1 \cdot t_1 = (E \cdot g)^{-1} \bmod \phi(N) = 28457731$.
6. Compute $\gamma^{-1} \bmod N = 40051234$,
7. Compute $(\gamma^{-1})^g \bmod N = \gamma^{-g} \bmod N = 57947710$.
8. Recover the message:

$$M = \Big(C_2 \cdot (\gamma^{-1})^g \bmod N\Big)^{D_1 \cdot t_1} \bmod N = (71485117 \cdot 57947710)^{28457731} \bmod 77404279 = 59.$$

This demonstrates that an attacker can successfully decrypt the ciphertext, indicating that the scheme is vulnerable to this attack.

Analysis of E_1 and Security Implications

The value E_1 is derived from random values e_1 and e_2, where $\gcd(e_1, \phi(n)) = \gcd(e_2, \phi(n)) = 1$. However, this does not enhance security because encryption operates modulo N. Once the attacker determines $\phi(N)$, they can compute the inverses of E and g. Furthermore, the claimed security of the private key element γ^{-1} is unfounded, as it can be directly derived from the public parameter γ using modular inversion. As a result, the second layer of encryption introduced in this modification does not enhance the cryptographic strength of the algorithm.

The addition of γ as a public parameter and the use of double encryption do not alter the underlying algebraic structure of the ciphertext or obscure the relationships among the key components. Since the decryption process remains algebraically reversible once $\phi(N)$ is known, the scheme inherits the same structural vulnerabilities as its predecessor. Moreover, the extra encryption step introduces greater computational overhead without contributing meaningful entropy or resistance to known attacks.

It is important to note that the attack based on $\phi(N)$ assumes that the totient value is known or can be inferred, which is not typically the case in standard RSA settings. However, under certain implementation flaws or side-channel leaks, such assumptions may be valid. The Lenstra-based factorization attack, on the other hand, does not rely on knowledge of $\phi(N)$ and is more broadly applicable to poorly sized keys.

Alternative Attack: Factoring N

If the attacker does not have access to $\phi(N)$, the most practical approach to break the system is to factor N directly. Various integer factorization algorithms exist; in this case, the Lenstra Elliptic Curve Factorization Method (ECM) is applied to $N = 77404279$, which is a composite number. Using the elliptic curve equation:

$$y^2 = x^3 + ax + b \pmod{q}$$

where $a = 1$, $b = 1$, and $q = N = 77404279$, we define $P = (0, 1)$ and perform scalar multiplication until reaching the point at infinity. This choice is arbitrary but valid, as ECM works with randomly chosen curves and points until a non-trivial factor of N is revealed.

The steps are as follows:

1. We perform scalar multiplications of P on the chosen elliptic curve, following the standard group law:

$$\begin{aligned} 2P &= (19351070,\ 67728743), \\ 3P &= (72,\ 611), \\ 20P &= (29168093,\ 46167503), \\ 21P &= \mathcal{O}. \end{aligned}$$

2. At the 21^{st} multiplication, reaching the point at infinity implies

$$\gcd(x_2 - x_1, N) \neq 1 \quad \text{or} \quad \gcd(2 \cdot y_1, N) \neq 1.$$

3. Since $21P = 20P + P$, where $20P = (29168093,\ 46167503)$ and $P = (0, 1)$.
4. Determine the prime factors as follows:
   - If $x_2 \neq x_1$, then compute $\gcd(x_2 - x_1, N)$.
   - If $x_2 = x_1$ and $y_2 = y_1$, then compute $\gcd(2 \cdot y_1, N)$.
5. The given point satisfies the first condition, $x_2 \neq x_1$, such that

$$\gcd(x_2 - x_1, N) = \gcd(29168093,\ 77404279) = 101.$$

6. Update:
$$N = \frac{77404279}{101} = 766379.$$
7. Repeat the process until four prime factors are obtained:
$$p = 101, \quad q = 79, \quad r = 89, \quad s = 109.$$

The diagram in Figure 1 illustrates the core structure of the Modified ESRKGS encryption scheme, with a specific focus on its vulnerability when the Euler totient function $\phi(N)$ is known to an adversary. Although the scheme introduces additional cryptographic layers, such as masking the plaintext using a parameter γ and applying a secondary exponentiation with base g, its security ultimately relies on the secrecy of $\phi(N)$. The diagram demonstrates how an attacker, equipped with knowledge of $\phi(N)$, can reverse the encryption process and successfully recover the original message, exposing a critical weakness in the scheme's design.

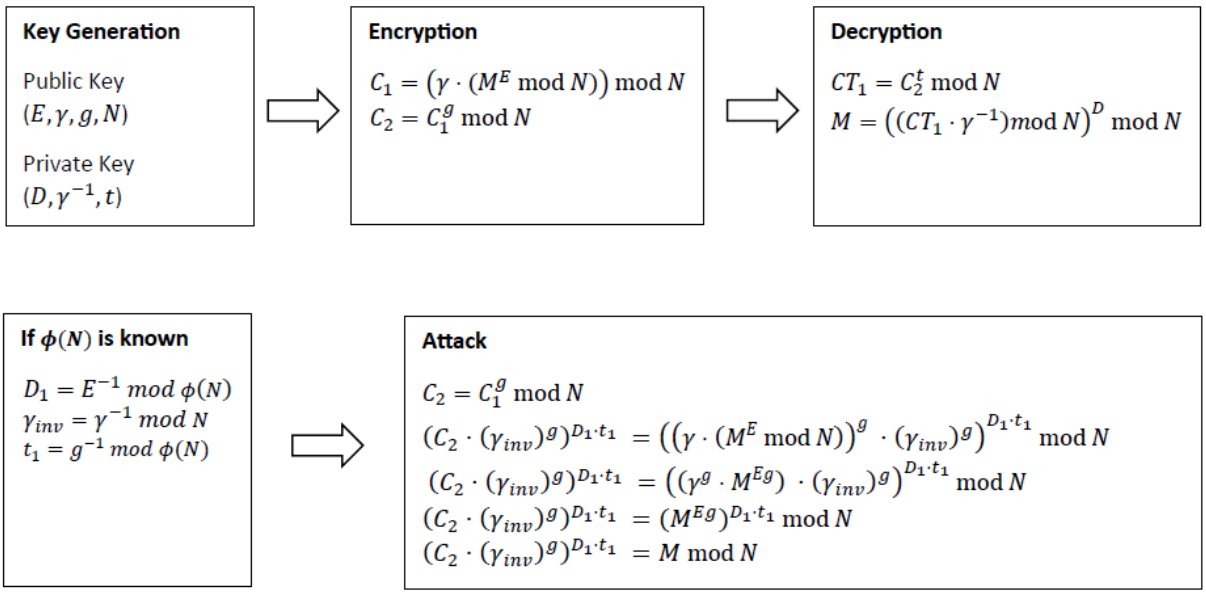

Figure 1: Attack process flow

In the key generation phase, two key pairs are constructed. The public key consists of (E, γ, g, N), while the private key comprises (D, γ^{-1}, t), where D and t are the modular inverses of E and g, respectively, with respect to $\phi(N)$.

Encryption is performed in two steps: first, the message M is masked and exponentiated as
$$C_1 = \left(\gamma \cdot (M^E \bmod N)\right) \bmod N, \tag{15}$$
and then further transformed into
$$C_2 = C_1^g \bmod N. \tag{16}$$

Decryption reverses this process by applying the exponent t, removing the γ masking, and performing the standard RSA decryption.

However, the lower part of the diagram demonstrates that if an attacker gains knowledge of $\phi(N)$, for example, through side-channel analysis, leakage, or factorization of N, they can reconstruct the private parameters (D_1, γ^{-1}, t_1). These are functional equivalents of the original private key components, computed as follows:
$$D_1 = E^{-1} \bmod \phi(N), \quad \gamma_{\text{inv}} = \gamma^{-1} \bmod N, \quad t_1 = g^{-1} \bmod \phi(N). \tag{17}$$

Using knowledge of $\phi(N)$, an attacker can systematically reverse the encryption process through modular arithmetic. By computing C_2 and undoing the layered exponentiation and

masking steps, the original plaintext M can be fully recovered. As illustrated in the accompanying diagram, each transformation applied during encryption is algebraically invertible when $\phi(N)$ is known.

This exposes a critical weakness in the scheme: despite its structural modifications, such as double encryption and the use of the parameter γ, the scheme introduces no additional computational hardness beyond that of classic RSA. If the modulus N can be factored, or if $\phi(N)$ is otherwise disclosed, the entire system becomes vulnerable to complete decryption.

In a broader cryptographic context, the Modified ESRKGS remains rooted in the same security assumption as RSA: the difficulty of factoring large integers. This makes it inherently insecure in the post-quantum era, where algorithms such as Shor's [21] can efficiently break such assumptions. Consequently, the long-term viability of ESRKGS, regardless of internal enhancements, is fundamentally limited.

Moreover, the scheme's practical relevance appears minimal. There is no evidence of its adoption in real-world cryptographic systems, either in its original or modified form. Its contribution is primarily academic, serving as a case study in the design and analysis of multiprime RSA variants, rather than as a practical or post-quantum-secure alternative to existing public-key schemes.

4 Conclusion

This study investigated the ESRKGS modification proposed by Azzahra to evaluate its effect on computational complexity and cryptographic security. The modification extends the factorization problem from $n = p \cdot q$ to $N = p \cdot q \cdot r \cdot s$, thereby increasing the mathematical complexity of the underlying task.

The analysis reveals that, although the factorization structure becomes more complex, this change does not result in a substantial improvement in security. In particular, when $\phi(N)$ is known, an adversary can efficiently recover the original message M, effectively nullifying the intended cryptographic enhancement. The scheme does not introduce any fundamentally new hardness assumptions and remains susceptible to classical factorization-based attacks.

Given these findings, the modified ESRKGS cannot be considered a viable alternative to RSA, especially in post-quantum scenarios. Future research should focus on developing cryptographic schemes that incorporate novel hardness assumptions and offer provable resistance against both classical and quantum attacks.

CRediT Authorship Contribution Statement

Bety Hayat Susanti: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Validation, Writing – original draft, Writing – review & editing, Funding acquisition. **Aditya Sukhoi Lean Sumule**: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Mareta Wahyu Ardyani**: Data curation, Validation, Writing – review & editing.

Declaration of Generative AI and AI-assisted technologies

During the preparation of this manuscript, the authors used ChatGPT version 4 and Grammarly to assist in language refinement, grammar checking, formatting adjustments, and improving clarity. The authors reviewed and verified all content generated or suggested by these tools to ensure accuracy, validity, and alignment with the intended meaning. The final responsibility for the content of this manuscript rests entirely with the authors.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical Statement

This cryptanalysis is conducted strictly for academic purposes to evaluate the security of experimental cryptographic algorithms. No real-world systems or confidential data were compromised during this research.

Funding and Acknowledgments

This research was funded by the Politeknik Siber dan Sandi Negara, Bogor, West Java, Indonesia. We extend our sincere appreciation to the reviewers for their valuable feedback and suggestions.

References

[1] R. S. Dhakar, A. K. Gupta, and P. Sharma, "Modified RSA encryption algorithm (MREA)," in *2012 Second International Conference on Advanced Computing & Communication Technologies (ACCT)*, Rohtak, India, 2012. DOI: 10.1109/ACCT.2012.74.

[2] S. Y. Yan, *Cryptanalytic Attacks on RSA*. Springer, 2008. DOI: 10.1007/978-0-387-48742-7.

[3] M. Stamp and R. lOW, *Applied Cryptanalysis: Breaking Ciphers in the Real World*. Jon Wiley and Sons, 2007. DOI: 10.1002/9780470148778.

[4] M. J. Hinek, *Cryptanalysis of RSA and Its Variants*. Chapman & Hall, 2009. DOI: 10.1201/9781420075199.

[5] Y.-S. Sun, C. W. Chiou, and W.-C. Sun, "A factorization attack algorithm on RSA cryptosystem using fast searching algorithm," *Journal of Applied Mathematics and Computation*, vol. 6, no. 4, pp. 390–404, 2022. DOI: 10.26855/jamc.2022.12.001.

[6] I. H. Masri and B. H. Susanti, "Cryptanalysis on polynomial congruence-based public key with Chinese Remainder Theorem," in *2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs)*, Bogor, Indonesia, 2023, pp. 159–164. DOI: 10.1109/ICoCICs58778.2023.10276994.

[7] I. K. Y. Sucipta, B. H. Susanti, and S. S. Carita, "Cryptanalysis of the RSA cryptosystem based on n prime numbers," in *2024 7th International Conference on Information and Communications Technology (ICOIACT)*, Ishikawa, Japan, 2024. DOI: 10.1109/ICOIACT64819.2024.10912893.

[8] D. A. Ferdianto, B. H. Susanti, and S. Rosdiana, "Common modulus attack on the elliptic curve-based RSA algorithm variant," in *2024 7th International Conference on Information and Communications Technology (ICOIACT)*, Ishikawa, Japan, 2024. DOI: 10.1109/ICOIACT64819.2024.10913331.

[9] P. Q. Nguyen, "Public-key cryptanalysis," in *Recent Trends in Cryptography*, I. Luengo, Ed., vol. 477, AMS–RSME, 2009. Available online.

[10] D. Zhang, H. Wang, S. Li, and B. Wang, "Progress in the prime factorization of large numbers," *The Journal of Supercomputing*, vol. 80, no. 8, pp. 11 382–11 400, 2024. DOI: 10.1007/s11227-023-05876-y.

[11] B. H. Susanti, T. K. Silim, N. P. R. Adiati, and M. W. Ardyani, "e^{th} Root Attack on Dual Modulus RSA," *ZERO: Jurnal Sains, Matematika dan Terapan*, vol. 9, no. 1, pp. 289–296, 2025. DOI: 10.30829/zero.v9i1.24486.

[12] M. Thangavel, P. Varalakshmi, M. Murrali, and K. Nithya, "An Enhanced and Secured RSA Key Generation Scheme (ESRKGS)," *Journal of Information Security and Applications*, vol. 20, pp. 3–10, 2015. DOI: 10.1016/j.jisa.2014.10.004.

[13] E. Lüy, Z. Y. Karatas, and H. Ergin, "Comment on "An Enhanced and Secured RSA Key Generation Scheme (ESRKGS)"," *Journal of Information Security and Applications*, vol. 30, pp. 1–2, 2016. DOI: 10.1016/j.jisa.2016.03.006.

[14] T. F. Azzahra, "Modifikasi Algoritme ESRKGS Thangavel *et al.* Menggunakan Metode Multi Kunci dan Teorema Sisa Cina," Undergraduate thesis, Department of Cryptography, National Cyber and Crypto Polytechnic, 2021. Available online.

[15] G. P. Aditya, A. Aminuddin, and S. Arifianto, "Improvisasi Algoritma RSA Menggunakan Generate Key ESRKGS pada Instant Messaging Berbasis Socket TCP," *Jurnal Repositor*, vol. 2, no. 11, 2020. DOI: 10.22219/repositor.v2i11.30959.

[16] A. Aminuddin, G. P. Aditya, and S. Arifianto, "RSA algorithm using key generator ESRKGS to encrypt chat messages with TCP/IP protocol," *Jurnal Teknologi dan Sistem Komputer*, vol. 8, no. 2, pp. 113–120, 2020. DOI: 10.14710/jtsiskom.8.2.2020.113-120.

[17] E. M. Horra, A. M. Beyene, and S. Y. Techan, "Enhanced Avalanche Effect Analysis Algorithm Considering both Single and Double Key Pair RSA Algorithms," *Research Square*, 2024. DOI: 10.21203/rs.3.rs-4113962/v1.

[18] D. Wulansari, M. A. Muslim, and E. Sugiharti, "Implementation of rsa algorithm with chinese remainder theorem for modulus N 1024 bit and 4096 bit," *International Journal of Computer Science and Security (IJCSS)*, vol. 10, no. 5, pp. 186–194, 2016. Available online.

[19] Y. F. Alias, M. A. M. Isa, and H. Hashim, "Timing Attack: An Analysis of Preliminary Data," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, no. 1–4, pp. 29–32, 2017. Available online.

[20] H. W. L. Jr., "Factoring Integers with Elliptic Curves," *The Annals of Mathematics*, vol. 126, no. 3, pp. 649–673, 1987. Available online.

[21] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, Santa Fe, NM, USA, 1994. DOI: 10.1109/SFCS.1994.365700.