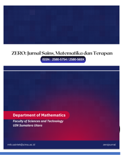
Zero : Jurnal Sains, Matematika, dan Terapan
E-ISSN : 2580-5754; P-ISSN : 2580-569X
Volume 9, Number 1, 2025
DOI: 10.30829/zero.v9i1.24486
Page: 289-296



e^{th} Root Attack on Dual Modulus RSA

[1] **Bety Hayat Susanti**
Dept. of Cryptographic Engineering, Politeknik Siber dan Sandi Negara, Bogor, Indonesia

[2] **Tsamara Khadijah Silim**
Dept. of Cryptographic Engineering, Politeknik Siber dan Sandi Negara, Bogor, Indonesia

[3] **Nadia Paramita Retno Adiati**
Dept. of Cryptographic Engineering, Politeknik Siber dan Sandi Negara, Bogor, Indonesia

[3] **Mareta Wahyu Ardyani**
Dept. of Cryptographic Engineering, Politeknik Siber dan Sandi Negara, Bogor, Indonesia

Article Info

Article history:

Accepted, 20 July 2025

Keywords:

Dual Modulus RSA
e^{th} Root Attack
Public Key Cryptanalysis

ABSTRACT

The Rivest–Shamir–Adleman (RSA) algorithm relies on the presumed difficulty of integer factorization, making it vulnerable to certain attacks, particularly in the quantum era. One proposed variant, dual modulus RSA, is claimed to enhance resilience against specific cryptanalytic techniques. This study evaluates its security by applying an e^th-root attack using an advanced fraction method. The results demonstrate that the plaintext can be recovered without the private key, confirming that dual modulus RSA, like standard RSA, remains susceptible under particular conditions. Although dual modulus RSA incurs higher computational cost, the attack remains effective. These findings suggest that structural changes alone do not guarantee improved security and emphasize the need for rigorous cryptanalysis of RSA variants against established mathematical attacks.

Corresponding Author:

Bety Hayat Susanti,
Department of Cryptographic Engineering,
Politeknik Siber dan Sandi Negara, Bogor, Indonesia
Email: bety.hayat@poltekssn.ac.id

1. INTRODUCTION

The general idea of information security is to ensure the confidentiality of said information from unauthorized parties. Further, the security goals stretch to integrity and availability as well [1]. These goals can be achieved using cryptography, which ensures secure communication through the internet. Based on the keys used, cryptography is broadly categorized into two categories: symmetric cryptography and asymmetric cryptography. Symmetric cryptography uses one shared key between interacting parties, the sender and recipient, whereas in asymmetric cryptography, interacting parties need to use both public and private keys.

The RSA algorithm is one of the public-key algorithms that utilizes two different keys [2]. It has several modifications (called variants) to increase security. Some examples are RSA with multiple primes, the rebalanced RSA, and the RSA-CRT [3]. This paper examines an attack on the dual modulus RSA variant. This variant modifies the key generation, encryption, and decryption processes. The dual modulus RSA has three different types, namely Dual-RSA Small-e, Dual-RSA Small-d, and Dual Generalized RSA Rebalanced [4]. Several studies have examined the dual modulus RSA and the attacks carried out against it [5], [6], [7], [8], [9], [10].

The dual modulus RSA variant offers theoretically superior security compared to standard RSA due to its use of two separate moduli and the application of double encryption. This design expands the effective key space and increases the computational difficulty for an attacker. Since the ciphertext is encrypted twice with distinct public keys and moduli, any attempt to recover the original message must either break both moduli or find a way to invert both encryptions simultaneously. This complicates factorization-based attacks, as an adversary would need to solve two distinct hard problems instead of one. Additionally, the redundancy introduced by the two encryptions mitigates vulnerabilities arising from low-exponent attacks or partial message exposure, making the dual modulus RSA more resilient to a range of cryptanalytic techniques.

The security of RSA is tested against several types of attacks, including those based on integer factorization, the use of quantum computers, attacks on the RSA function, and attacks on the RSA implementation [11], [12], [13], [14]. These attack techniques, originally designed for public key algorithms in general, have proven effective against RSA. Attacks can also be carried out directly by extracting keys and messages from the algorithm's trapdoors or even without trapdoors (for example, see [15], [16], [17], [18], [19]).

The e^{th} root attack is one of the methods used to penetrate the RSA algorithm by exploiting vulnerabilities related to small public exponents. This attack allows recovery of the original message without factoring the public modulus n, targeting instead the mathematical properties of exponentiation [12]. Several instances of the e^{th} root attacks have been demonstrated by researchers [20], [21]. This type of attack can also be used to assess the security of the dual modulus RSA algorithm. In this variant, the message is encrypted twice with different public keys, and the attack aims to recover the original message M using known parameters and ciphertexts. The presence of two encryption layers, each with potentially distinct exponents and moduli, increases the challenge.

In this study, the modified RSA with dual modulus and double encryption will be evaluated under the e^{th} root attack model. This attack was selected because it has not yet been applied to this variant in existing literature. Therefore, the process and modeling of the e^{th} root attack will be adapted to the dual modulus RSA structure. Moreover, the study includes a comparative analysis of the time and effort required to perform the attack on both standard RSA and dual modulus RSA. This comparison is essential to assess whether the added complexity of double encryption truly enhances resistance against such cryptanalytic attacks.

1.1 Standard RSA Algorithm

The RSA algorithm was founded by R. Rivest, A. Shamir, and L. Adleman. It is an asymmetric key algorithm that utilizes two keys, namely public and private keys [22], [23], [24]. In RSA, both the plaintext and ciphertext are integers between 0 and $n-1$. The recommended value of n is 1024 bits, or 309 decimal digits. This means that n is less than 2^{1024}. The algorithm can be used to provide secrecy and further developed into a digital signature scheme. The security of the RSA algorithm is based on the intractability of the integer factorization problem.

The RSA algorithm consists of key generation, encryption, and decryption. The steps in the key generation, encryption, and decryption processes are detailed in Algorithms 1, 2, and 3, respectively.

Algorithm 1. Key Generation for RSA Public-Key Encryption [22]

Input: Two distinct large primes, p and q
Output: Public key (n, e), private key d
Summary: Each entity creates an RSA public key and a corresponding private key. Each entity A should do the following:

1. Generate two large random (and distinct) primes p and q, each is roughly the same size.
2. Compute $n = pq$ and $\varphi(n) = (p-1)(q-1)$.
3. Select a random integer e, $1 < e < \varphi(n)$, such that $\gcd(e, \varphi(n)) = 1$.
4. Use the extended Euclidean algorithm to compute the unique integer d, $1 < d < \varphi(n)$, such that $ed \equiv 1 \pmod{n}$.
5. A's public key is (n, e); A's private key is d.

Algorithm 2. Encryption in RSA [22]

Input: Message m; A's public key (n, e)
Output: Ciphertext c
Summary: B encrypts a message m for A and does the following:

1. Obtain A's authentic public key (n, e).
2. Represent the message as an integer m in the interval $[0, n-1]$.
3. Compute $c = m^e \bmod n$.
4. Send the ciphertext to A.

Algorithm 3. Decryption in RSA [22]

Input: Ciphertext c; private key d
Output: Message m
Summary: To recover the plaintext m from c, A should do the following:

1. Use the private key d to recover $m = c^d \bmod n$.

1.2 Modified RSA with Dual Modulus RSA

The modified RSA algorithm used in this paper is the dual modulus and double encryption methods, where there are changes in the key generation, encryption, and decryption processes from the original RSA algorithm. The dual modulus algorithm used is based on the DMRJT (Dual Modulus RSA based on Jordan Totient Function) created by Balram *et al.* in 2015 [25]. The key generation, encryption, and decryption processes of the modified RSA with a dual modulus algorithm are specified in Algorithms 4, 5, and 6, respectively.

Algorithm 4. Key Generation in Modified RSA with a Dual Modulus

Input: Four distinct large primes, p, q, r and s

Output: Public keys (n_1, n_2, e_1, e_2), private key $(d_1, d_2, p, q, r, s, \varphi_1(n), \varphi_2(n))$

Summary: Each entity creates public keys and corresponding private keys. Each entity A should do the following:

1. Generate 4 primes p, q, r, s with roughly the same size.
2. Compute the values of $n_1 = p \times q$ and $n_2 = r \times s$.
3. Compute the values of $\varphi_1(n) = (p-1)(q-1)$ and $\varphi_2(n) = (r-1)(s-1)$.
4. Choose two random integers e_1 and e_2 with $\gcd(e_1, \varphi_1(n)) = 1$ and $\gcd(e_2, \varphi_2(n)) = 1$.
5. Compute the value of private keys d_1 and d_2 such that $e_1 d_1 = 1 \bmod(\varphi_1(n))$ and $e_2 d_2 = 1 \bmod(\varphi_2(n))$.
6. A's public keys are (n_1, n_2, e_1, e_2); A's private keys are $(d_1, d_2, p, q, r, s, \varphi_1(n), \varphi_2(n))$.

Algorithm 5. Encryption in Modified RSA with a Dual Modulus

Input: Message m; A's public key (n_1, n_2, e_1, e_2)

Output: Ciphertext c

Summary: B encrypts a message m for A and does the following:

1. Obtain A's authentic public key (n_1, n_2, e_1, e_2).
2. Represent the message as an integer m such that $1 < m < n_1 - 1, n_2 - 1$.
3. Compute $c = ((m^{e_1} \bmod n_1)^{e_2} \bmod n_2)$.
4. Send the ciphertext to A.

Algorithm 6. Decryption in Modified RSA with a Dual Modulus

Input: Ciphertext c; private key (d_1, d_2, p, q, r, s)

Output: Message m

Summary: To recover the plaintext m from c, A should do the following:

1. Use the private keys to recover $m = ((c^{d_2} \bmod n_2)^{d_1} \bmod n_1)$.

The dual modulus RSA modified algorithm was built to improve the security of RSA using the concept of double encryption. However, the dual modulus algorithm requires a longer execution time compared to the original RSA because it uses two pairs of public and private keys. An example of the usage of the algorithm with relatively small parameters is as follows.

Key generation

1. Choose 4 primes $p = 7, q = 11, r = 23$, and $s = 31$.
2. Compute the values of $n_1 = p \times q = 77$ and $n_2 = r \times s = 713$.
3. Compute the values of $\varphi_1(n) = (p-1)(q-1) = 60$ and $\varphi_2(n) = (r-1)(s-1) = 660$.
4. Choose $e_1 = 53$ and $e_2 = 37$.
5. Compute the value of the private keys d_1 and d_2 such that $e_1 d_1 = 1 \bmod(\varphi_1(n)) = 17$ and $e_2 d_2 = 1 \bmod(\varphi_2(n)) = 553$.
6. The public keys are $(77, 713, 53, 37)$ and the private keys are $(17, 553, 7, 11, 23, 31, 60, 660)$.

Encryption

1. The public keys of $A = (77, 713, 53, 37)$.
2. Choose a message $m = 36$.
3. Compute the ciphertext $c = ((m^{e_1} \bmod n_1)^{e_2} \bmod n_2) = 568$.

Decryption

1. The private keys are $(17, 553, 7, 11, 23, 31, 60, 660)$.
2. Calculate the message $m = ((c^{d_2} \bmod n_2)^{d_1} \bmod n_1) = 36$

Based on the example above, one ciphertext is produced from one encryption process per message.

1.3 e^{th} Root Attack

e^{th} root attack is an indirect algorithmic attack that focuses on retrieving messages without the knowledge of the key [12]. The attack is generally formulated using Equation (1) and the formula below:

$$m \equiv \sqrt[e]{c}\ (mod\ n) \tag{1}$$

Equation (1) is used to perform an attack on the e^{th} root of the ciphertext modulo n. Therefore, to recover the message m, the Root Finding Problem (RFP) is used by utilizing the value of the ciphertext c. The RSA algorithm can be attacked by exploiting information on the values of $c, n,$ and $e,$ by targeting a message m until the original message is found. Theoretically, it is possible to try every possible element from the multiplicative group $\mathbb{Z}_n^*$ where n is an integer, but this will be infeasible once the value of n is too large. The attack also exploits the value of $\varphi(n)$ to compute the e^{th} root from c modulo n, so that if the value of $\phi(n)$ is known, it will be easier to recover the message m in polynomial time, as stated in Theorem 1.

Theorem 1 [12]

$$\varphi(n) \overset{\mathcal{P}}{\Rightarrow} RFP(C) \tag{2}$$

Based on Theorem 1, it can be proven that RFP is possible if the value of $\varphi(n)$ is known. The congruence $ed \equiv 1 \ (mod\ \varphi(n))$ will reveal the values of d dan k using the following equation:

$$ed - k\varphi(n) = 1, k \in \mathbb{Z} \tag{3}$$

Equation (3) can be solved by performing the extended Euclid algorithm, which operates in polynomial time. In addition, the linear Diophantine equation consists of d and k can be solved using the finite continued fraction method. Once the value of d is known, m can be recovered using Equation (2), which meets the value to finish the e root in $\sqrt[e]{c}\ (mod\ n)$, with its explanation as follows:

$$\begin{aligned} m^e &\equiv (c^d)^e \ (mod\ n) \\ &\equiv c^{de} \ (mod\ n) \\ &\equiv c^{1+k\varphi(n)} \ (mod\ n) \\ &\equiv c \cdot c^{\varphi(n)} \ (mod\ n) \\ m^e &\equiv c \ (mod\ n) \end{aligned} \tag{4}$$

The following explanation is the mathematical derivation to solve the RFP:

Finite Continued Fraction

An expansion of continued fractions from $\xi \in \mathbb{R}$ with ${}^{p_n}/_{q_n}$ can be written with the following equation [25]:

$$\xi = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\dots + \cfrac{1}{a_n}}}} = [a_0, a_1, a_2, \dots, a_n] \tag{5}$$

The equation can be written as $\xi = [a_0, a_1, a_2, \dots, a_n]$. If ξ is a rational number, then an expansion of continued fractions is denoted as ${}^{p_n}/_{q_n} = [a_0, a_1, a_2, \dots, a_n]$ will be acquired. Theorem 2 is used to calculate the desired continued fraction as follows.

Theorem 2 [12]

If $\xi > 0$ with $\gcd(p_n, q_n) = 1$ and

$$\left|\xi - \frac{p_n}{q_n}\right| < \frac{1}{2{q_n}^2} \tag{6}$$

then ${}^{p_n}/_{q_n}$ is a convergent of the continued fraction expansion of ξ.

Euclid's Algorithm

This algorithm is performed by computing the greatest common divisor of a and b [12]. It is used to compute the linear congruence:

$$ax \equiv 1 \ (mod\ n) \text{ or } ax \equiv b \ (mod\ n) \tag{7}$$

which then completes the linear Diophantine equation of the form $ax + by = c$.

2. RESEARCH METHOD

In this study, we utilized a combination of literature review and experimental methods. The literature review was conducted to collect relevant information regarding the e^{th} root attack and the dual modulus RSA algorithm. Meanwhile, the experimental method was applied to evaluate the susceptibility of these algorithms to the e^{th} root attack. The experiment was conducted in three key stages: setting up the experimental environment, executing the e^{th} root attack, and analyzing the results. The experimental environment was established using the Python programming language.

An e^{th} root attack is performed by targeting the value of the private key d to recover the message m, given the values of the ciphertext c and the public keys, n and e. For dual modulus RSA with double encryption, the attack can be modeled as follows:

$$c \equiv (m^{e_1}(\text{mod } n_1))^{e_2}(\text{mod } n_2) \tag{8}$$

$$m \equiv \left(\sqrt[e_1]{\sqrt[e_2]{c}(mod\ n_2)}\right)(mod\ n_1) \tag{9}$$

Under the assumption of known $\varphi(n)$, the private key values d_1, d_2, and corresponding integers k_1, k_2 satisfy:

$$e_1\, d_1 - \varphi(n_1)k_1 = 1 \tag{10}$$

$$e_2\, d_2 - \varphi(n_2)k_2 = 1 \tag{11}$$

This approach generalizes to an i-modulus RSA, where i represents the number of modulus-public key pairs. The recovered message m can be expressed recursively for multiple moduli:

$$m \equiv \left(\sqrt[e_1]{\left(\sqrt[e_{\dots}]{\sqrt[e_i]{c}(\text{mod } n_i)}\right)(\text{mod } n_{j_{\dots}})}\right)(\text{mod } n_1) \tag{12}$$

To recover the original m, we used the continued fraction method as described in Algorithm 7.

Algorithm 7. Continued Fraction Method to Recover d and k

```
Input: Public key $e$, $\varphi(n)$
Output: Values of $d$ and $k$ such that $ed - \varphi(n)k = 1$
  1. Compute the continued fraction expansion of $e/\varphi(n)$.
  2. Extract the convergent of the continued fraction, which provides approximate values for $d/k$.
  3. Identify $d$ and $k$ from the convergents using:
     $d = (-1)^{n-1} q_{n-1}$
     $k = (-1)^{n-1} q_{n-2}$
  4. Verify $ed - \varphi(n)k = 1$ to ensure correctness.
```

3. RESULT AND ANALYSIS

In the e^{th} root attack, the difficulty lies in the selection of the correct value of $\varphi(n)$ to solve the RFP. In other words, the difficulty of the attack is to find the value of $\varphi(n)$ on the modulus being used. The value $\varphi(n)$ can be found by searching all integers that are relatively prime to the given e.

The following example illustrates the application of the e^{th} root attack on dual modulus RSA, a cryptographic vulnerability that occurs when the same plaintext is encrypted with different public keys that share a common exponent. The example walks through each step of the attack, demonstrating how the mathematical properties of dual modulus RSA are exploited to recover the original plaintext efficiently.

For the dual modulus RSA with the given parameters: $c = 568$, $e_1 = 53$, $e_2 = 71$, $n_1 = 851$, and $n_2 = 1457$, the recovered message m satisfies:

$$m \equiv \left(\sqrt[53]{\sqrt[71]{568}\ (\text{mod } 1457)}\right)(\text{mod } 851) \tag{13}$$

The values of $\varphi_1(851) = 792$, $\varphi_2(1457) = 1380$ were assumed. Next, there were two trials on one ciphertext with dual modulus. The first step was to recover the value of m' by finding the values of d and k in the following equation:

$$71d - 1380k = 1 \tag{14}$$

By using ${}^{e}/_{\varphi(n)}$, the continued fraction was used as follows:

$$\frac{71}{1380} = 0 + \cfrac{1}{19+\cfrac{1}{2+\cfrac{1}{3+\cfrac{1}{2+\frac{1}{4}}}}} = [0.19, 2, 3,2,4]$$

Next, the continued fraction resulted in a convergent:

$$\left[0, \frac{1}{19}, \frac{2}{39}, \frac{7}{136}, \frac{16}{311}, \frac{71}{1380}\right]$$

Finding the values of d and k by considering the convergent value yields:

$$d = (-1)^{n-1} q_{n-1} = (-1)^4 311 = 311 \tag{15}$$

$$k = (-1)^{n-1} q_{n-1} = (-1)^4 16 = 16 \tag{16}$$

Proving with the substitution of result values d and k in Equations (15) and (16), so that it was proven to meet Equation (3), gives us:

$$71(311) - 1380(16) = 1$$

Then, we computed the message m':

$$m' \equiv (m''^{71})^{311} \bmod 145 \equiv 568^{311} \bmod 1457$$
$$m' \equiv 143 \bmod 851 \tag{17}$$

Based on the result using the value of the first public key, we then performed the same method using the second public key. To attack the dual modulus RSA with double encryption, we executed the method as many times as the number of moduli being used to recover the real message, which in this case was two. We then used the values of e_1 and $\varphi_1(n)$ to recover m. The computation was carried out as follows.

$$53d - 792k = 1 \tag{18}$$

The value of $^{e}/_{\varphi(n)}$ was found by using the continued fraction equation as follows:

$$\frac{53}{792} = 0 + \frac{1}{14 + \frac{1}{1 + \frac{1}{16 + \frac{1}{1 + \frac{1}{2}}}}} = [0.14, 1, 16, 1, 2]$$

From the result of the continued fraction, the following convergent was obtained.

$$\left[0, \frac{1}{14}, \frac{1}{15}, \frac{17}{254}, \frac{18}{269}, \frac{53}{792}\right]$$

Next, the values of d and k are found by using the convergent results.

$$d = (-1)^{n-1} q_{n-1} = (-1)^4 269 = 269 \tag{19}$$

$$k = (-1)^{n-1} q_{n-1} = (-1)^4 18 = 18 \tag{20}$$

We then substituted the values of d and k in Equations (19) and (20) to validate our results by using Equation (5), as follows:

$$53(269) - 792(18) = 1$$

We then computed the message m:

$$m \equiv (m^{53})^{269} \bmod 851 \equiv 143^{269} \bmod 851$$
$$m \equiv 89 \bmod 851 \tag{21}$$

To assess performance, the attack was executed across several modulus sizes by varying the number of digits in *n*. The execution time for both standard RSA and dual modulus RSA was measured and compared. The results, illustrated in Figure 1, indicate that although dual modulus RSA tends to take slightly more time on average, the overall difference is minimal. Thus, the modulus size has a limited impact on attack execution time.

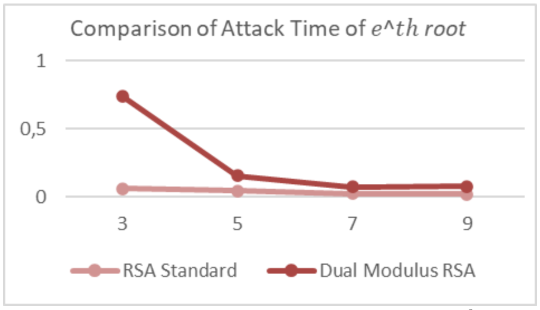


Figure 1. The time needed to perform the e^{th} root attack

The message being used for the encryption and description processes is 90, and all the values of $n, c,$ and e along with the assumed value of $\varphi(n)$ successfully recovered the original message m. Figure 1 displays that despite the difference in size of the modulus, the execution time was similar for both the original RSA and the dual modulus

RSA. On average, the time needed to perform the e^{th} root attack on the dual modulus RSA is longer, but it can be inferred that the size of the modulus does not significantly affect the execution time.

The e^{th} root attack on the dual modulus RSA would be successful if executed on the algorithm with the value of the public key n_2 greater than n_1. If the value of n_1 is greater than n_2, then several public keys cannot be decrypted. Therefore, to make this attack successful, the value of the public key used should be $n_2 > n_1$, since there will be ambiguity when a decryption process is carried out on the ciphertext, only several values can be decrypted. This requirement also applies to the public exponent, i.e., $e_2 > e_1$.

In the e^{th} root attack, the number of prime factors used in the construction of the modulus n can influence the attack feasibility, particularly when the resulting modulus is small. This type of attack does not aim to recover the private key, but instead directly computes the plaintext message m from the ciphertext c when $m^e < n$, allowing the attacker to extract the e^{th} root of c without modular reduction.

In the case of dual modulus RSA, where encryption may involve multiple moduli (e.g., n_1, n_2), the risk can be greater if the same plaintext is encrypted using the same small exponent e across different moduli. This creates the conditions for a CRT-based e^{th} root attack (e.g., Håstad's attack), where the attacker reconstructs me over the integers and then takes the e^{th} root to recover m without factoring any modulus.

Therefore, while the e^{th}root attack does not rely on the factorization of the modulus; it is affected by the size and number of moduli used, particularly when small exponents and repeated messages are involved. The modified RSA scheme must ensure that encryption is properly padded and that the modulus size is large enough to avoid conditions that enable this class of attack.

4. CONCLUSION

The e^{th} root attack, which falls under the category of Root Finding Problem attacks, typically exploits knowledge of $\varphi(n)$ and applies linear Diophantine approximation techniques to recover the original message from the ciphertext. Our experiments show that even when the dual modulus RSA variant is used—where the message is encrypted twice with different moduli but the same public exponent—an attacker can still retrieve the original message using this attack. Although our analysis found that dual modulus RSA requires more computational time to break compared to standard RSA, this increase in effort does not fundamentally eliminate the vulnerability.

These findings have important implications for the practical security of RSA-based systems. The fact that the e^{th} root attack remains effective, even with dual modulus encryption, underscores the risks of encrypting the same message with the same exponent across multiple moduli. While dual modulus RSA may slow down an attacker, it does not provide robust protection against root-finding attacks, especially when small public exponents are used or when proper padding is not implemented.

To address these weaknesses, it is essential to adopt stronger countermeasures. Using randomized padding schemes, such as Optimal Asymmetric Encryption Padding (OAEP), can ensure that identical messages generate different ciphertexts, thereby disrupting the conditions needed for the e^{th} root attack. Additionally, selecting larger public exponents and enforcing strict key management policies can further reduce the risk of such attacks.

Looking ahead, future research should focus on developing advanced message randomization and padding techniques, as well as exploring the integration of dual modulus RSA with other cryptographic primitives to enhance security. Formal security analyses of these variants under various attack scenarios will also be valuable in understanding their practical strengths and limitations.

In summary, while dual modulus RSA presents some improvement over standard RSA in terms of resistance to the e^{th} root attack, it is not a comprehensive solution. A combination of robust cryptographic practices and continued research is necessary to ensure the security of RSA implementations in real-world applications.

5. REFERENCES

[1] [M. A. Islam, M. A. Islam, N. Islam, and B. Shabnam, "A modified and secured RSA public key cryptosystem based on 'n' prime numbers," *J. Comput. Commun.*, vol. 6, no. 3, pp. 78–90, 2018. https://doi.org/10.4236/jcc.2018.63006.

[2] A. Nitaj, M. R. Bin, K. Ariffin, N. Nur, H. Adenan, and N. A. Abu, "Classical attacks on a variant of the RSA cryptosystem," *Cryptology ePrint Archive*, Paper 2021/1160, 2021. https://doi.org/10.1007/978-3-030-88238-9_8.

[3] M. Cherkaoui-Semmouni, A. Nitaj, W. Susilo, and J. Tonien, "Cryptanalysis of RSA variants with primes sharing most significant bits," in *Proc. ISC'21, Lecture Notes in Computer Science*, vol. 12912, 2021, pp. 42–53. https://doi.org/10.1007/978-3-030-91356-4_3.

[4] H. M. Sun, M. E. Wu, W. C. Ting, and M. J. Hinek, "Dual RSA and its security analysis," *IEEE Trans. Inf. Theory*, vol. 53, no. 8, pp. 2922–2933, Aug. 2007. https://doi.org/10.1109/TIT.2007.901248.

[5] B. Hutagaol, M. A. Budiman, and H. Mawengkang, "Hybrid cryptosystems MRC-4 algorithm and RSA dual modulus to secure data," in *Proc. 2023 Int. Conf. Comput. Sci. Inf. Technol. (ICOSNIKOM)*, Medan, Indonesia, 2023, pp. 1–6.

[6] A. Nitaj and M. Boudabra, "Improved cryptanalysis of the multi-power RSA cryptosystem variant," in *Progress in Cryptology – AFRICACRYPT 2023*, vol. 14024, Cham, Switzerland: Springer, 2023, pp. 153–170.

[7] L. Peng, L. Hu, Y. Lu, and M. Zhao, "Cryptanalysis of dual RSA," *Des. Codes Cryptogr.*, vol. 83, pp. 1–21, 2017.

[8] K. R. Raghunandan, R. R. Dsouza, N. Rakshith, S. Shetty, and G. Aithal, "Analysis of an enhanced dual RSA algorithm using Pell's equation to hide public key exponent and a fake modulus to avoid factorization attack," in *Adv. Artif. Intell. Data Eng. (AIDE)*, vol. 1343, Singapore: Springer, 2021, pp. 731–746.

[9] B. Swami, R. Singh, and S. Choudhary, "Dual modulus RSA based on Jordan-totient function," *Procedia Technol.*, vol. 24, pp. 1581–1586, 2016. https://doi.org/10.1016/j.protcy.2016.05.143.

[10] G. A. Zimbele and S. A. Demilew, "Hidden real modulus RSA cryptosystem," *Int. J. Comput.*, vol. 22, no. 2, pp. 238–247, 2023. https://doi.org/10.47839/ijc.22.2.3094.

[11] Y.-S. Sien, C. W.-C. Chiou, and W.-C. Sun, "A factorization attack algorithm on RSA cryptosystem using fast searching algorithm," *J. Appl. Math. Comput.*, vol. 6, no. 4, pp. 390–404, 2022. https://doi.org/10.26855/jamc.2022.12.001.

[12] Y. Y. Song, *Cryptanalysis Attacks on RSA*. Berlin, Germany: Springer, 2008.

[13] M. Stamp and R. M. Low, *Applied Cryptanalysis: Breaking Ciphers in the Real World.* Hoboken, NJ, USA: Wiley, 2007.

[14] M. Hinek, *Cryptanalysis of RSA and Its Variants.* Boca Raton, FL, USA: CRC Press, 2009.

[15] I. H. Masri and B. H. Susanti, "Cryptanalysis on polynomial congruence-based public key with Chinese Remainder Theorem," in *Proc. 2023 IEEE Int. Conf. Cryptogr., Informatics, Cybersecurity (ICoCICs)*, 2023, pp. 159–164.

[16] I. K. Y. Sucipta, B. H. Susanti, and S. S. Carita, "Cryptanalysis of the RSA cryptosystem based on n prime numbers," in *2024 7th Int. Conf. Inf. Commun. Technol. (ICOIACT)*, Ishikawa, Japan, 2024, pp. 270–275. https://doi.org/10.1109/ICOIACT64819.2024.10912893.

[17] D. A. Ferdianto, B. H. Susanti, and S. Rosdiana, "Common modulus attack on the elliptic curve-based RSA algorithm variant," in *2024 7th Int. Conf. Inf. Commun. Technol. (ICOIACT)*, Ishikawa, Japan, 2024, pp. 101–106. https://doi.org/10.1109/ICOIACT64819.2024.10913331.

[18] P. Q. Nguyen, "Public-key cryptanalysis," in *Recent Trends in Cryptography, Contemp. Math.*, vol. 477, pp. 67–119, 2009.

[19] D. Zhang, H. Wang, S. Li, and B. Wang, "Progress in the prime factorization of large numbers," *J. Supercomput.*, vol. 80, pp. 11382–11400, 2024. https://doi.org/10.1007/s11227-023-05876-y.

[20] A. C. Ramachandran, P. Ramachandran, and L. Velusamy, "Computing the e^{th} root of a number using a variant of the RSA algorithm (for even e's)," U.S. Patent US8595349B2, Nov. 26, 2013.

[21] Y. Wang, H. Zhang, and H. Wang, "Quantum algorithm for attacking RSA based on the e^{th} root," *J. Sichuan Univ. (Eng. Sci. Ed.)*, vol. 50, pp. 72–77, 2018.

[22] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, 1st ed. Boca Raton, FL, USA: CRC Press, 1996.

[23] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 8th ed. Upper Saddle River, NJ, USA: Pearson Education, 2023.

[24] D. R. Stinson and M. Paterson, *Cryptography: Theory and Practice*, 4th ed. Boca Raton, FL, USA: Chapman & Hall/CRC, 2017. https://doi.org/10.1201/9781315282497.

[25] A. Nitaj, Y. Kadri, M. T. Ahmed, and A. El Maazouz, "Security issues of novel RSA variant," *IEEE Access*, vol. 10, pp. 13020–13032, 2022.