MODIFICATION OF THE CAESAR CIPHER USING PRIME NUMBERS TO ENHANCE SECURITY IN CRYPTOGRAPHY

RIZKA PUTRI RAHAYU[1], ACHMAD FAUZI*[2], CHRISTNATALIS HS[3]

[1] STMIK Kaputama, Jl. Venteran No.4A-9A Binjai, 20714, Sumatera Utara, Indonesia
E-mail: fauzyrivai88@gmail.com*

ABSTRACT

The Caesar Cipher is a classical cryptographic algorithm that applies a static character shift, resulting in a limited key space and vulnerability to brute-force and frequency analysis attacks; this limitation highlights a research gap in developing a simple yet more secure and efficient modification of the algorithm. This study aims to enhance the security of the Caesar Cipher by introducing a prime number based dynamic key mechanism, in which each plaintext character is encrypted using the *i*-th prime number according to its position, thereby increasing ciphertext variability and reducing repetitive patterns. The proposed method is evaluated using texts of varying lengths to examine encryption and decryption consistency, key space expansion, character diffusion, and computational efficiency. The security analysis focuses on resistance to brute-force and statistical frequency analysis attacks, while performance evaluation considers processing time and theoretical algorithmic complexity. The experimental results demonstrate that employing prime numbers as dynamic keys significantly enlarges the key space and improves resistance to classical cryptanalysis without introducing substantial computational overhead. Therefore, the proposed approach provides a more effective and secure solution compared to the conventional Caesar Cipher

Keywords: Caesar Cipher; Classical Cryptography; Dynamic Key; Prime Numbers; Security Analysis

1. INTRODUCTION

The rapid advancement of information and communication technology has increased the need for data security systems capable of protecting information from unauthorized access. In this context, cryptography plays a crucial role in ensuring the confidentiality, integrity, and authenticity of data across various digital applications, such as information systems, wireless communications, and the Internet of Things [1]. Symmetric cryptography is still widely used due to its simplicity and efficiency, particularly in systems with limited computational resources. One of the most well-known classical symmetric cryptographic algorithms is the Caesar Cipher. This algorithm operates by shifting each character in the plaintext by a certain fixed value within the alphabet. The Caesar Cipher offers advantages in terms of ease of implementation and understanding of fundamental cryptographic concepts[2]. However, the use of a static shift key results in a very limited key space, making the algorithm vulnerable to brute force attacks and frequency analysis [2],[3]. With the growing demand for data security, various studies have been conducted to modify the Caesar Cipher algorithm in order to enhance its security level. Several approaches have been developed, including the use of dynamic keys, integration with other cryptographic algorithms, and the application of specific mathematical concepts to expand the key space and increase the complexity of the resulting ciphertext[4],[5]. This modification aims to reduce the main weaknesses of the Caesar Cipher without eliminating the simplicity of the algorithm. One potential approach to enhancing the security of the Caesar Cipher is the utilization of prime numbers. Prime numbers possess unique mathematical properties, are non-repetitive, and difficult to predict, making them widely used in various modern cryptographic schemes to expand the key space and improve resistance against cryptanalytic attacks[6],[7]. By utilizing prime numbers as the basis for generating dynamic keys, it is expected that the variation of the ciphertext can increase significantly.

Based on this background, this study proposes a modification of the Caesar Cipher by using a sequence of prime numbers as dynamic and deterministic shift values. Each plaintext character is encrypted using

a different prime number in sequence. Unlike previous approaches that employ random keys or complex algorithms, the proposed method maintains the simplicity of the original Caesar Cipher, making it suitable for lightweight cryptographic applications and educational purposes [5], [8]. The main contributions of this study include the development of a prime number–based Caesar Cipher algorithm while maintaining computational simplicity, the implementation of encryption and decryption methods using dynamic shifts derived from prime numbers, and the evaluation of the algorithm's performance and security, which demonstrates improved resistance to brute force attacks and frequency analysis compared to the traditional Caesar Cipher.

2. METHODOLOGY

The research methodology was designed to evaluate the effectiveness of the prime number based modification of the Caesar Cipher in enhancing cryptographic security. An experimental approach was employed to compare the proposed method with the conventional Caesar Cipher in terms of security and performance aspects..

2.1 Research Methodology Flow

The research was conducted through several stages, which include:

a. A literature review on the Caesar Cipher and the use of prime numbers in cryptography
b. The design of a prime number–based algorithm
c. The implementation of the encryption and decryption processes
d. Testing using text data with varying lengths
e. Security analysis and performance evaluation of the algorithm.
f. Drawing conclusions.

These stages are visualized in the form of a research methodology flowchart, the workflow illustration can be seen in Figure 1.

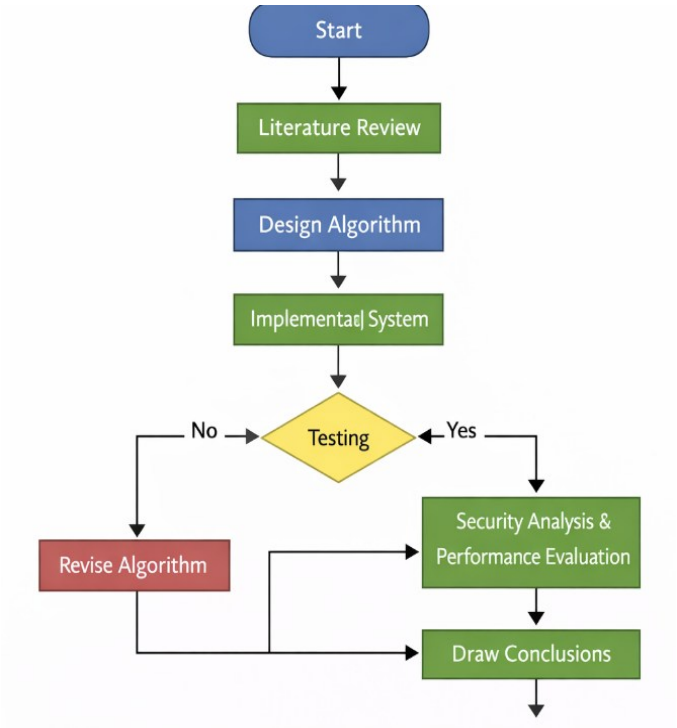

Figure 1. Research Stages Flowchart

2.2 Research Data

The research data consist of alphabetic characters (A–Z) with varying text lengths, categorized as:

a. Short text
b. Medium-length text
c. Long text

Alphabetic data are commonly employed in classical cryptography research to assess the consistency of encryption and decryption mechanisms, as well as to analyze the structural patterns of the generated ciphertext [2], [3].

2.3 Caesar Cipher Algorithm

The conventional Caesar Cipher employs a single fixed shift value (k). The encryption and decryption processes are formulated as follows:

$$C_i = (P_i + k) \, mod \, 26 \quad (1)$$

$$P_i = (C_i - k) \, mod \, 26 \quad (2)$$

This method has a primary limitation in the form of a limited key space and ciphertext patterns that are easily analyzed using brute-force attacks and frequency analysis [2], [4].

2.4 Prime Number Based Caesar Cipher

In the proposed method, the shift values are generated using a sequence of prime numbers as a dynamic key. Each plaintext character is encrypted using the i-th prime number, following a dynamic key approach in classical cryptography aimed at increasing ciphertext variation[5], [6], [9], [10]. The encryption process is formulated as follows:

$$C_i = (P_i + p_i) \, mod \, 26 \quad (3)$$

The decryption procedure is defined as follows:

$$P_i = (C_i - p_i) \, mod \, 26 \quad (4)$$

where P_i denotes the $i - th$ prime number. The conceptual working mechanism of the modified Caesar cipher algorithm can be observed in Figure 2.

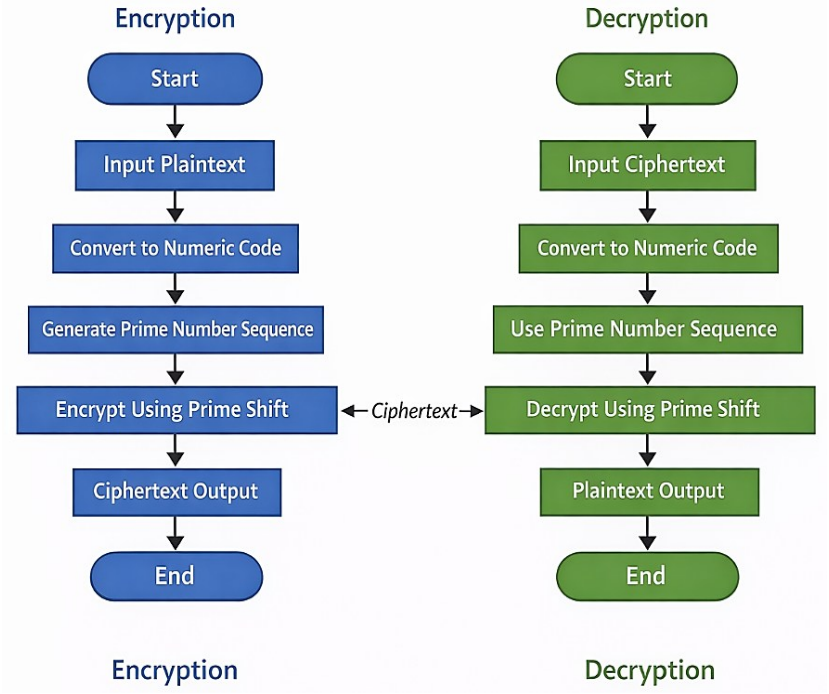

Figure 2. Encryption and Decryption Process

2.5 Encryption Process

The stages of the encryption process are as follows:

a. The plaintext is entered into the system
b. Each letter is converted into its numerical representation (A = 0 to Z = 25).
c. The system generates a sequence of prime numbers corresponding to the length of the plaintext.
d. The numerical values derived from the plaintext are shifted using the prime numbers.
e. The result of the modulo 26 operation is converted back into characters to produce the ciphertext.

Illustration of the encryption process using a prime number–based Caesar cipher [11], [12], [13].

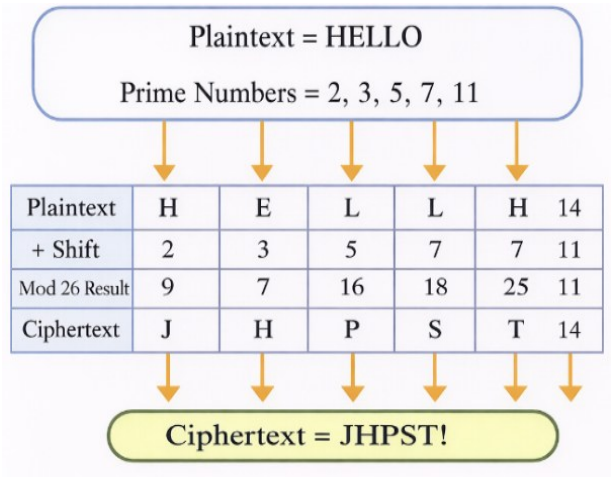

Figure 3. Encryption Process

2.6 Decryption Process

The decryption process consists of the following steps:

a. The encrypted text (ciphertext) is converted into its numerical representation
b. The key used is the identical sequence of prime numbers.
c. The numerical values of the ciphertext are subtracted by the corresponding prime numbers.
d. The result of the modulo 26 operation is converted back into the original plaintext

Illustration of the decryption process using a prime number–based Caesar cipher [14].

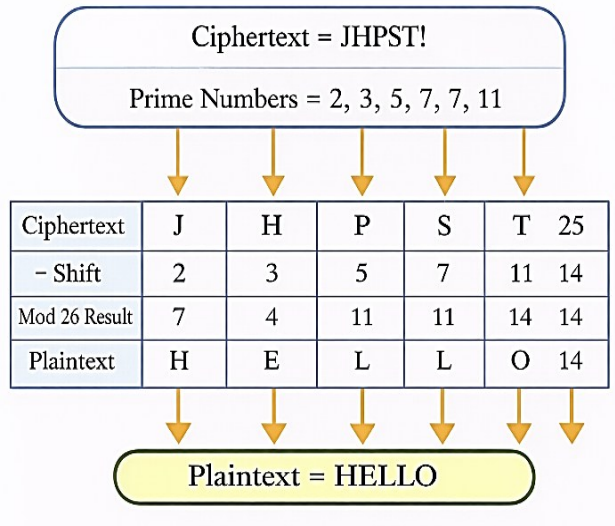

Figure 4. Decryption Process

2.7 Security Analysis

The security analysis was conducted by considering the following aspects:

a. Key space consideration, to evaluate the increase in the probability of key combinations.
b. Ciphertext variation, to observe the extent of character diffusion.
c. Resistance to attacks, particularly brute-force attacks and frequency analysis[2].

2.8 Performance Evaluation

The performance evaluation includes the following aspect:

a. Encryption and decryption execution time
b. Accuracy of the decryption results.
c. Theoretical computational complexity of the algorithm.

The obtained results were utilized to compare the performance of the conventional Caesar Cipher with the proposed prime number based approach[15], [16].

3. RESULTS AND DISCUSSION

This section describes the design and implementation process of the proposed prime number–based Caesar Cipher algorithm. The explanation focuses on the mechanism for generating prime number keys, the encryption and decryption processes, and an implementation example to clarify how the algorithm works

3.1 RESULTS

In the proposed method, the key is no longer a single fixed shift value, but rather a sequence of prime numbers used sequentially for each plaintext character. Prime numbers are selected due to their mathematical property of having no divisors other than 1 and themselves, thereby increasing the complexity of the character shift pattern. The sequence of prime numbers is generated according to the length of the plaintext. For example, if the plaintext consists of *n* characters, the system generates the first *n* prime numbers as dynamic keys.

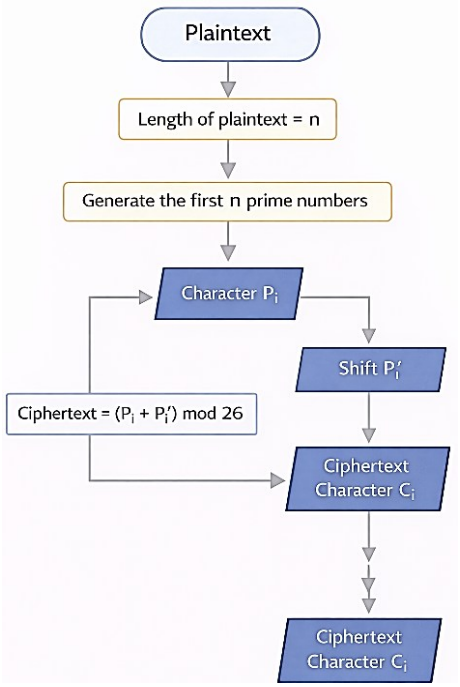

Figure 5. Flowchart of Prime Number Key Generation and Character Shifting Process

3.1.1 Encryption Process Implementation

The encryption process is performed by shifting each plaintext character according to the corresponding prime number. The implementation steps of the encryption process are described as follows:

a. The plaintext is entered into the system.
b. Each character is converted into its numeric representation within the range A = 0 to Z = 25
c. The system generates a sequence of prime numbers according to the length of the plaintext
d. Each numeric plaintext value is shifted using the corresponding i-th prime number.
e. The shifted result is taken modulo 26 and converted back into a ciphertext character.

Mathematically, the encryption process is expressed by the following equation:

$$C_i = (P_i + P_i') \, mod \, 26 \qquad (5)$$

Where P_i represents the numeric value of the $i-th$ plaintext character, and P_i' denotes the the $i-th$ prime number used as the dynamic shift key.

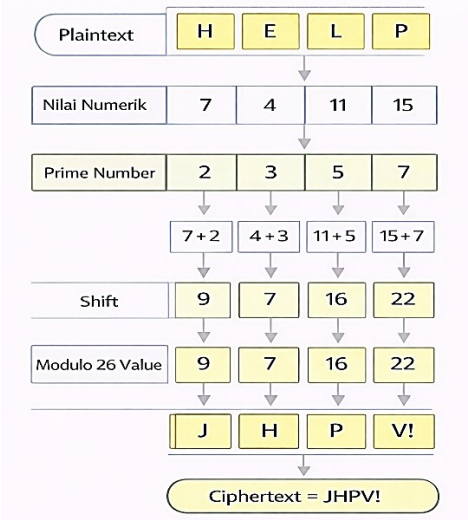

Figure 6. Encryption Process of the Prime Number–Based Caesar Cipher

3.1.2 Decryption Process Implementation

The decryption process is performed to restore the ciphertext to the original plaintext using the same sequence of prime numbers. The decryption steps are as follows:

a. The ciphertext is converted into its numeric representation.
b. The same prime numbers are used as the keys.
c. The numeric value of each ciphertext character is subtracted by the corresponding i-th prime number.
d. The result of the operation is taken modulo 26 and converted back into a plaintext character.

The decryption process is mathematically expressed by the following equation:

$$P_i = (C_i - P_i') \bmod 26$$

Figure 7. Decryption Process of the Prime Number–Based Caesar Cipher

3.2 DISCUSSION

For Ilustrations, the plaintext "DATA" is encrypted using the initial prime numbers {2, 3, 5, 7}. The shifting process produces different ciphertext for each character, resulting in a non-uniform shift pattern unlike the conventional Caesar Cipher. This implementation demonstrates that the use of prime numbers as dynamic keys can generate more varied and less predictable ciphertext..

This section discusses the testing results of the prime number–based Caesar Cipher algorithm, as well as the security and performance analysis. The evaluation was carried out by comparing the proposed method with the conventional Caesar Cipher.

3.2.1 Encryption and Decryption Testing

Testing was conducted using text data of varying lengths, including short, medium, and long texts. The results demonstrate that the prime number–based Caesar Cipher algorithm is able to perform encryption and decryption processes consistently. The decrypted plaintext always matches the original plaintext, indicating that the proposed algorithm functions correctly

Table 1. Encryption and Decryption Test Results

No	Text Type	Length	Plaintext	Ciphertext (Proposed)	Decryption Result
1	Short	5	Hello	JHPST	Hello
2	Medium	11	Kriptografi	MTNKYZVJFSM	Kriptografi
3	Long	25	Keamanandatateksrahasia	...	Keamanandatateksrahasia

3.2.2 Differences in the Modified Caesar Cipher Method

The security analysis was conducted to evaluate the algorithm's resistance to cryptanalytic attacks. The analyzed parameters included key space, ciphertext variation, and resistance to brute force attacks and frequency analysis. The results indicate that the use of prime numbers as dynamic keys increases the complexity of attacks compared to the conventional Caesar Cipher.

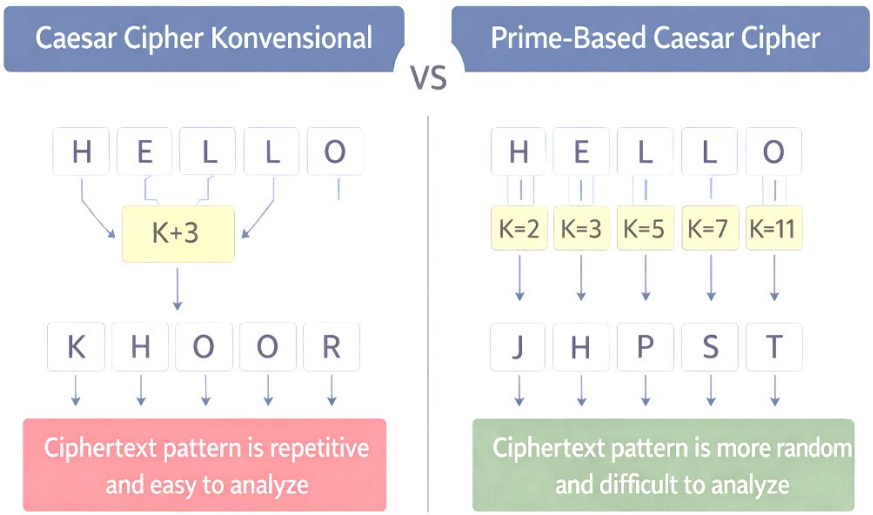

Figure 8. Comparison between the Conventional Caesar Cipher and the Prime Number Based Caesar Cipher

Table 2. Algorithm Security Comparison

Aspect	Conventional Caesar	Proposed Method
Key Type	Static	Dynamic (prime numbers)
Key Space	Small	Larger
Ciphertext Pattern	Easy to analyze	More random
Brute Force Resistance	Low	Higher
Frequency Analysis	Vulnerable	More difficult

3.2.3 Algorithm Performance Evaluation

The performance evaluation was conducted to assess the efficiency of the proposed algorithm. The evaluation parameters included encryption and decryption processing time as well as the theoretical complexity of the algorithm. The results indicate that the addition of the prime number generation process does not cause a significant increase in computational time.

Table 3. Algorithm Performance Evaluation

Method	Encryption Time	Decryption Time	Complexity
Conventional Caesar	Very fast	Very fast	O(n)
Proposed Method	Fast	Fast	O(n)

Based on the testing results, security analysis, and performance evaluation, it can be concluded that the modification of the Caesar Cipher using prime numbers as dynamic keys is capable of enhancing security compared to the conventional method. This approach expands the key space and produces more varied ciphertext patterns without compromising computational efficiency. Nevertheless, the algorithm remains a classical cryptographic technique and is therefore more suitable for lightweight applications and educational purposes in cryptography.

4. CONCLUSION

The conclusions of this study are as follows:

1. The modified Caesar Cipher algorithm using prime numbers as dynamic keys is capable of performing encryption and decryption consistently without any loss of information, as demonstrated by the match between the original plaintext and the decrypted result.
2. The use of a sequence of prime numbers as dynamic keys expands the key space and produces more diverse ciphertext variations, making it more resistant to brute force attacks and frequency analysis compared to the conventional Caesar Cipher.
3. The algorithm maintains linear complexity with respect to the length of the plaintext without significant increases in computation time, classifying it as a lightweight cryptographic method suitable for systems with limited computational resources and for educational purposes in cryptography.

5. SUGGESTIONS

The following are recommendations for future development of this research:

1. To further strengthen the achieved security advantages, the prime number–based Caesar Cipher method can be combined with other cryptographic techniques, such as transposition methods or modern substitution algorithms, in order to produce a more complex encryption system that is more resistant to various types of attacks.
2. Future developments may include applying the algorithm to non-alphabetic data, including numerical characters and special symbols, as well as implementing it in the form of an application or a simple security system to evaluate its effectiveness in real-world scenarios.
3. To ensure that efficiency and lightweight cryptographic characteristics are maintained, further testing on large-scale data and within real system environments is necessary to comprehensively evaluate performance, scalability, and security levels.

REFERENCES

[1] J. Alawatugoda, “Authenticated Key Exchange Protocol in the Standard Model under Weaker Assumptions,” *Cryptography*, vol. 7, no. 1, pp. 1–13, 2023, doi: 10.3390/cryptography7010001.

[2] S. Park, H. Kim, and I. Moon, “Automated Classical Cipher Emulation Attacks via Unified Unsupervised Generative Adversarial Networks,” *Cryptography*, vol. 7, no. 3, 2023, doi: 10.3390/cryptography7030035.

[3] M. Kasianchuk, R. Shevchuk, B. Adamyk, V. Benson, I. Shylinska, and M. Holembiovskyi, “Affine Cipher Encryption Technique Using Residue Number System,” *Cryptography*, vol. 9, no. 2, pp. 1–17, 2025, doi: 10.3390/cryptography9020026.

[4] G. Iovane, E. Benedetto, and A. Di Lauro, “A Quantum-Secure Cryptographic Algorithm Integrating Fractals and Prime Numbers,” *Appl. Sci.*, vol. 14, no. 22, 2024, doi: 10.3390/app142210138.

[5] B. I. Stefanov, B. S. Blagoev, L. Österlund, B. R. Tzaneva, and G. V. Angelov, “Effects of anodic aluminum oxide substrate pore geometry on the gas-phase photocatalytic activity of zno/al2o3 composites prepared by atomic layer deposition,” *Symmetry (Basel).*, vol. 13, no. 8, 2021, doi: 10.3390/sym13081456.

[6] P. Agarwal, M. Attary, M. Maghasedi, and P. Kumam, “Solving higher-order boundary and initial value problems via chebyshev-spectral method: Application in elastic foundation,” *Symmetry (Basel).*, vol. 12, no. 6, pp. 1–15, 2020, doi: 10.3390/SYM12060987.

[7] T. Rüberg, L. Kielhorn, and J. Zechner, “Electromagnetic devices with moving parts—simulation

with FEM/BEM coupling," *Mathematics*, vol. 9, no. 15, 2021, doi: 10.3390/math9151804.

[8] A. Rauh and J. Kersten, "Transformation of uncertain linear systems with real eigenvalues into cooperative form: The case of constant and time-varying bounded parameters," *Algorithms*, vol. 14, no. 3, 2021, doi: 10.3390/a14030085.

[9] M. Al Kahfi, M. Auva, D. P. Putra, C. D. P. B. Ginting, and A. Fauzi, "Super Text Data Encryption: Combination of Affine Cipher, Elgamal, and RSA Algorithms for Optimal Protection," *J. Sist. Inf. Kaputama*, vol. 9, no. 1, pp. 20–34, Jan. 2025, doi: 10.59697/JSIK.V9I1.949.

[10] A. Fauzi, S. Ramadani, H. Khair, and A. M. H. Pardede, "Integration Of Data Filtering With Hybrid RSA Deep Learning Algorithm For Iot Data Security And Classification.," *J. Theor. Appl. Inf. Technol.*, vol. 103, no. 22, 2025.

[11] R. Imanda *et al.*, "Development Of Hybrid Encryption Method Using Affine Cipher, Vigenere Cipher, And Elgamal Algorithm To Secure Text Messages In Data Communication System," *J. Artif. Intell. Eng. Appl.*, vol. 2, no. 2, pp. 30–40, Feb. 2023, doi: 10.59934/JAIEA.V2I2.154.

[12] R. G. Sinambela, A. Fauzi, and H. Khair, "Enhancing AES Key Generation Using Diffie-Hellman Method for Image Security," *J. Artif. Intell. Eng. Appl.*, vol. 4, no. 1, pp. 358–363, Oct. 2024, doi: 10.59934/JAIEA.V4I1.637.

[13] A. Fauzi and Y. Maulita, "Rivest Shamir Adleman (RSA) Super Encryption Algorithm with Vigenere Cipher Algorithm Modification for Image Security," vol. 4, no. 1, 2024.

[14] Rio Andika, Rizky Fajar Sitepu, Putri Ramadhani, Rizka Nova Fitria, and Achmad Fauzi, "Implementation of Super Encryption Using the Autokey Cipher and ElGamal Algorithms for Image Files," *J. Sist. Inf. Kaputama*, vol. 9, no. 1, pp. 45–54, 2025, doi: 10.59697/jsik.v9i1.957.

[15] R. Putra, A. Fauzi, and R. Saragih, "Development of Dynamic Key Based on Pseudo-Random Algorithm in Vigenere Cipher for Hybrid Vigenere-ElGamal Encryption to Secure Documen Data," *J. Artif. Intell. Eng. Appl.*, vol. 5, no. 1, pp. 700–710, Oct. 2025, doi: 10.59934/JAIEA.V5I1.1415.

[16] A. Fauzi, T. Y. Arif, Y. Away, and R. Roslidar, "Design of a Hybrid CNN and Rolling Hash Approach for Key Generation within the ElGamal Algorithm Applied to Image Encryption," Banda Aceh: International Conference on Computer System, Information Technology, and Electrical Engineering (COSITE) 2025, 2025, pp. 1–6.