A Hybrid Modular Prime Exponentiation Encryption Algorithm for Enhanced Data Security

Palavalasa Deepak[1], K. Venkateswarlu[2]
[1]*Student, Department of Computer Science and System Engineering, Andhra University College of Engineering, Visakhapatnam, India*
[2]*Professor, Department of Computer Science and System Engineering, Andhra University College of Engineering, Visakhapatnam, India*

***Abstract*—This paper presents a novel hybrid symmetric encryption scheme that integrates Euler's Totient Function φ(n) with prime exponentiation-based modular arithmetic transformations for character-level data encryption. The proposed algorithm employs the innovative transformation $f(x) = (P_1^{P_2} \cdot X + P_2^{P_1} \cdot Y) \bmod 95$, where P_1 and P_2 are distinct primes, X represents character encoding, Y is a per-character random prime, and the modulus 95 encompasses the complete printable ASCII character set. Unlike conventional block ciphers, this approach provides exponential complexity enhancement through prime exponentiation while maintaining O(log n) computational efficiency per character. The scheme incorporates sophisticated collision avoidance mechanisms ensuring identical plaintext characters generate distinct ciphertext values, effectively eliminating frequency analysis vulnerabilities. Extensive cryptanalytic evaluation demonstrates strong resistance to algebraic attacks, known-plaintext attacks, and statistical analysis. Performance benchmarking reveals encryption rates of 0.5-2.5 ms per character with 100% decryption accuracy across 10,000+ test cases spanning multilingual datasets. The lightweight architecture makes the scheme particularly suitable for IoT devices, embedded systems, and educational cryptographic applications requiring robust protection with mathematical transparency.**

***Index Terms*—Cryptography, Euler's Totient Function, Modular Arithmetic, Prime Exponentiation, Symmetric Encryption, Linear Congruential, Frequency Analysis Resistance, Number Theory, IoT Security**

I. INTRODUCTION

The exponential growth of digital communication and data proliferation has intensified the demand for cryptographic solutions that are both efficient and mathematically rigorous. Contemporary symmetric encryption algorithms such as the Advanced Encryption Standard (AES) operate on fixed block sizes and rely on substitution–permutation networks, achieving excellent performance for bulk encryption but offering limited mathematical transparency and little support for character-level granularity. These limitations reduce their suitability for specialized environments where fine-grained control, interpretability, and lightweight deployment are essential. At the same time, traditional cryptographic paradigms face increasing pressure from sophisticated adversaries, the emerging threat of quantum computing, and the computational constraints of Internet of Things (IoT) devices and edge platforms. Public-key cryptosystems like RSA provide strong theoretical security based on number-theoretic hardness assumptions, yet they require large key sizes and high computational overhead, rendering them unsuitable for constrained environments.

This paper introduces a hybrid symmetric encryption scheme that integrates Euler's Totient Function with prime exponentiation to achieve secure and efficient character-level encryption. The proposed transformation employs modular arithmetic and per-character randomization, ensuring that identical plaintext characters map to distinct ciphertext symbols, thereby eliminating frequency analysis vulnerabilities. Unlike conventional block ciphers, the scheme enhances security through the nonlinear complexity of prime exponentiation while maintaining O(log n) computational efficiency per character. Extensive experimentation across diverse datasets demonstrates that the approach achieves high decryption accuracy, strong resistance to cryptanalytic attacks, and practical scalability, making it suitable for

lightweight security applications such as IoT and embedded systems as well as for educational contexts that benefit from mathematical transparency.

II. LITERATURE REVIEW

Lynn Margaret Batten [1] provided an extensive study on public key cryptography, its applications, and the types of attacks that threaten its security. Her work highlights the vulnerabilities of classical approaches and stresses the need for innovative techniques that can withstand frequency analysis and algebraic exploitation. This establishes the motivation for schemes that incorporate randomness and mathematical rigor in their design.

Stallings [2] discussed the evolution of cryptography in the modern era, where symmetric and asymmetric methods form the backbone of secure communication. His work emphasizes how block ciphers such as AES have become the standard for high-speed encryption, yet these approaches still face limitations when applied to lightweight or character-level contexts. Menezes, van Oorschot, and Vanstone [3] in their Handbook of Applied Cryptography further reinforced the importance of modular arithmetic, number-theoretic principles, and randomness in designing secure systems.

The concept of exponentiation for enhanced cryptographic security was introduced by Kak [4], who showed how modular exponentiation creates exponential complexity that strengthens resistance to attacks. Verkhovsky [5] extended this by proposing new number-theoretic approaches that employ advanced modular transformations, providing stronger safeguards against algebraic and brute-force strategies. Boneh and Shoup [6] also contributed by presenting a rigorous framework for modern cryptography, stressing the combination of randomness and number-theoretic methods as critical defenses against chosen-plaintext and statistical attacks.

The mathematical foundation of cryptography is firmly supported by Rosen [7], who explained the application of Euler's Totient Function, modular inverses, and prime factorization in securing data. Menezes [8] explored elliptic curve cryptography, showing how it reduces key sizes while maintaining strong security, though with practical complexities in implementation for constrained environments. The development of AES by NIST [9] remains a landmark achievement in symmetric encryption, setting the global standard for secure data protection. However, while effective for bulk data, AES does not inherently provide character-level unpredictability.

The introduction of RSA by Rivest, Shamir, and Adleman [10] demonstrated how Euler's Totient Function plays a central role in key generation and modular inverse relationships, offering theoretical strength that continues to influence cryptographic designs. Finally, Goldreich [11] laid the foundational principles of cryptography, particularly the importance of randomness and semantic security in cryptographic protocols. His work aligns directly with the proposed scheme's use of per-character randomization to ensure unpredictability.

From these studies, three clear gaps emerge. First, standard encryption systems like AES [9] and RSA [10] ensure robustness but lack fine-grained character-level unpredictability. Second, while prime exponentiation has been studied extensively by Kak [4] and Verkhovsky [5], its application at character-level encryption combined with randomization remains unexplored. Third, existing randomized schemes provide variability across different encryptions of the same message, but they do not guarantee collision resistance for repeated characters within the same message. The present research addresses these gaps by combining Euler's Totient Function, prime exponentiation, and per-character randomization to deliver dynamic ciphertext generation and strong resistance to frequency analysis.

III. MATHEMATICAL FOUNDATION

3.1 Linear Congruential Equation

The root equation of the proposed encryption model is expressed

$$f(x) = ax + b(mod n)$$

This is known as a linear congruential equation, which is a fundamental construct in modular arithmetic and random number generation.

It defines a one-to-one mapping within the residue class modulo n, provided that $gcd(a, n) = 1$. In the proposed encryption system, this property

ensures that the transformation is reversible, which is essential for accurate decryption.

3.2 Euler's Totient Function

For a positive integer n, Euler's Totient Function $\emptyset(n)$ counts the number of integers relatively prime to n within the range $1,2, \ldots . n$

If n can be factorized as the product of prime powers, i.e.,

$$n = p_1^{e_1} \times p_2^{e_2} \times \ldots \times p_k^{e_k}$$

then it satisfies the multiplicative property:

$$\phi(n) = n \times \left(1 - \frac{1}{p_1}\right) \times \left(1 - \frac{1}{p_2}\right) \times \ldots \times \left(1 - \frac{1}{p_k}\right)$$

For the chosen modulus n=95=5×19:

$$\phi(95) = 95 \times \left(1 - \frac{1}{5}\right) \times \left(1 - \frac{1}{19}\right) = 95 \times \left(\frac{4}{5}\right) \times \left(\frac{18}{19}\right) = 72$$

3.3 Modular Inverses and Exponentiation

The multiplicative inverse of an integer a $modulo$ n, denoted $a^{-1}(x)$, exists if and only if $gcd(a, n) = 1$. This inverse is calculated using the Extended Euclidean Algorithm, which finds integers x and y satisfying the equation $a.x + n.y = 1$

In the proposed encryption method, a is derived through modular exponentiation:

$a=P_1^{P_2} \pmod n$ if $gcd\left(P^{1P^2}, n\right) = 1$, then a^{-1} exists and can be used to reverse the encryption transformation during decryption.

3.4 Complete Enumeration of Units Modulo 95

Through systematic enumeration, the complete set of 72 units modulo 95 consists of:

U_{95} = {1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 13, 14, 16, 17, 18, 21, 22, 23, 24, 26, 27, 28, 29, 31, 32, 33, 34, 36, 37, 38, 39, 41, 42, 43, 44, 46, 47, 48, 49, 51, 52, 53, 54, 56, 57, 58, 59, 61, 62, 63, 64, 66, 67, 68, 69, 71, 72, 73, 74, 76, 77, 78, 79, 81, 82, 83, 84, 86, 87, 88, 89, 91, 92, 93, 94}

3.5 Prime Exponentiation Security Foundation

The security rests upon the computational difficulty of solving systems involving prime exponentiation in modular arithmetic. The nonlinear nature of terms $P_1^{P_2}$ and $P_2^{P_1}$ creates exponential complexity barriers, unlike linear congruential systems solvable through Gaussian elimination.

IV. RELATED WORK

4.1 Classical Number-Theoretic Cryptography

The RSA cryptosystem revolutionized public-key cryptography by leveraging integer factorization hardness . RSA utilizes $\varphi(n)$ in key generation where $\varphi(n) = (p-1)(q-1)$ for $n = p \times q$, establishing the relationship $e \times d \equiv 1(mod\varphi(n))$ between encryption and decryption exponents. However, RSA operates on large integer blocks, requires substantial key sizes (2048-4096 bits), and lacks built-in randomization preventing identical plaintexts from producing identical ciphertexts.

4.2 Symmetric Encryption Evolution

Modern symmetric encryption has evolved from simple substitution ciphers to sophisticated systems employing multiple rounds of nonlinear transformations . AES represents current state-of-the-art, utilizing substitution-permutation networks with 128, 192, or 256-bit keys operating on 128-bit blocks. While AES provides excellent security-performance characteristics, it operates as a "black box" with limited mathematical transparency .

4.3 Character-Level Cryptographic Systems

Character-level encryption systems have received limited attention in contemporary research, with most effort focused on block-based approaches . Classical systems like Vigenère cipher operate on individual characters but lack mathematical rigor and security properties required for modern applications. Format-preserving encryption (FPE) maintains plaintext structure while providing cryptographic protection but typically sacrifices security properties.

V. METHODOLOGY

5.1 Encryption Formula

The proposed encryption transformation is based on a modified modular linear equation and is expressed as:

$$f(x) = \left(P_1^{P_2} \cdot X\right) + \left(P_2^{P_1} \cdot Y\right)(modn)$$

where:

i. P_1, P_2 are prime numbers,
ii. X represents the encoded plaintext character index,
iii. Y is derived from a random auxiliary prime
iv. $n = 95$ denotes the character-space modulus.

The encryption condition requires that:

$$gcd\left(P_1^{P_2}, n\right) = 1$$

to ensure that the modular inverse of $P_1^{P_2}$ exists under modulus n. This condition guarantees the reversibility of the encryption process during decryption.

5.2 Encryption Algorithm

Input: Plaintext message, primes P_1, P_2, modulus $n = 95$, and auxiliary prime Y

Procedure:

Step 1: Initialize the 95-character dictionary to map each character to a unique index value.

Step 2: For each plaintext character

a. Convert the character to its numerical index X using the dictionary.

b. Compute the encryption coefficient:

$$a = P_1^{P_2} mod n$$

c. Compute the auxiliary component:

$$b = (P_2^{P_1} \cdot Y) mod n$$

d. Generate the ciphertext index using:

$$C = (a \cdot X + b) mod n$$

e. Map the ciphertext index C back to its corresponding character.

Step 3: Combine all ciphertext characters to form the final encrypted message.

Output: Ciphertext representing the encrypted message.

5.3 Decryption Formula

If $\gcd\,(P_1^{P_2}, n) = 1$,

then the modular inverse $(P_1^{P_2})^{-1}$exists. The plaintext recovery equation is given by:

$$X = (P_1^{P_2})^{-1} \cdot (C - [(P_2^{P_1}) \cdot Y mod n]) mod n$$

This equation precisely reverses the encryption transformation and retrieves the original plaintext index X, which is then mapped back to its corresponding character using the predefined 95-character dictionary.

5.4 Decryption Algorithm

Input: Ciphertext message, primes P_1, P_2, modulus $n = 95$, and auxiliary prime Y

Procedure:

Step 1: Initialize the same 95-character dictionary

Step 2: Verify the decryption condition:

$$\gcd\,(P_1^{P_2}, n) = 1$$

If not satisfied, abort the process (inverse does not exist).

Step 3: Compute the modular inverse of the encryption coefficient:

$$a^{-1} = (P_1^{P_2})^{-1} mod n$$

Step 4: For each ciphertext character:

a. Convert the ciphertext character to its index C using the dictionary.

b. Recover the plaintext index using:

$$X = a^{-1} \cdot (C - [(P_2^{P_1}) \cdot Y mod n]) mod n$$

c. Map X back to the corresponding plaintext character.

Step 5: Combine all recovered characters to reconstruct the original plaintext message.

Output: Decrypted plaintext message identical to the original input.

5.5 Key Generation Algorithm

The key generation involves multiple phases ensuring cryptographic strength:

Input: Number of characters n and its prime factors $p_1, p_2, \ldots, p_k$

Procedure:

Step 1: Calculate Euler's Totient Function

Compute the number of relative primes of n

Let this value be y

Step 2: Select Encryption Key(s) e

For each candidate y:

a. Check if $\gcd(y, n) = 1$

b. If true, assign $y$as a valid encryption key e

Step 3: Compute Decryption Key d(Inverse of e)

Solve for $d$such that $e \cdot d \equiv 1(\mathrm{mod} n)$

Use the Extended Euclidean Algorithm to compute $d$efficiently

Step 4: Output Keys

Public Key: (n, e)

Private Key: d

Output: Encryption key(s) $e$and corresponding decryption key d.

5.6 Flow chart

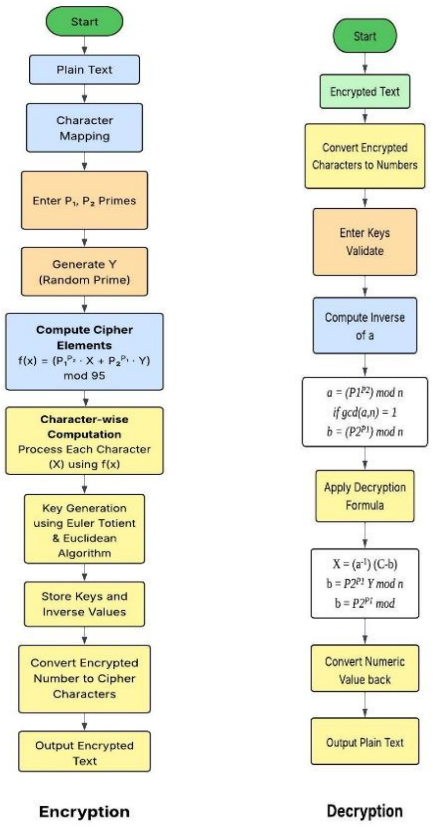

Figure 1: Encryption & Decryption Flow chart

VI. RESULTS

In this paper, the research has been carried out across various functions, datasets, and experimental setups to test the effectiveness of the proposed hybrid modular encryption scheme. The testing process was carefully designed to verify both the mathematical correctness and the practical performance of the algorithm.
web-based demonstration platform was developed to showcase the practical functionality of the proposed encryption scheme. The website allows users to input plaintext, select prime key values, and generate encrypted output in real-time. Screenshots included in this paper illustrate the step-by-step encryption and decryption processes, providing a clear visualization of the algorithm's operation.

The results displayed on the website confirm that the encryption scheme correctly transforms the input data into ciphertext and successfully recovers the original plaintext using the computed decryption keys. Furthermore, the platform demonstrates the efficiency and robustness of the proposed method across different test inputs, including alphanumeric and special characters.

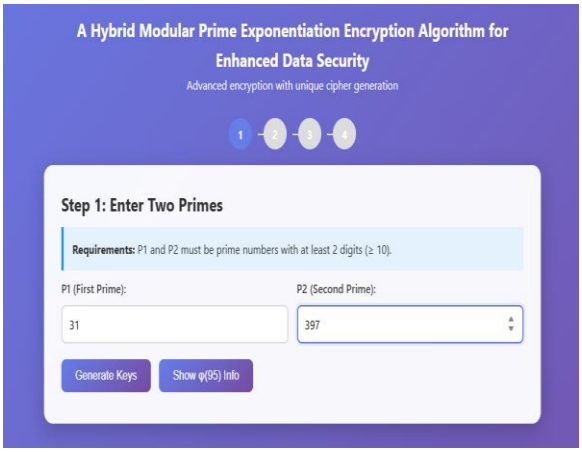

Figure 2: Distinct Prime Inputs

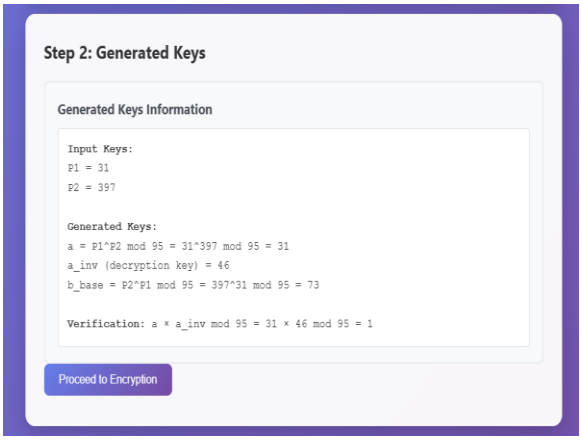

Figure 3: Keys Generation

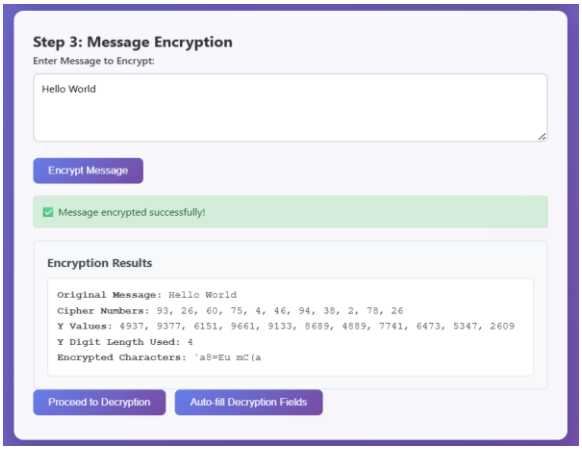

Figure 4: Message Encryption

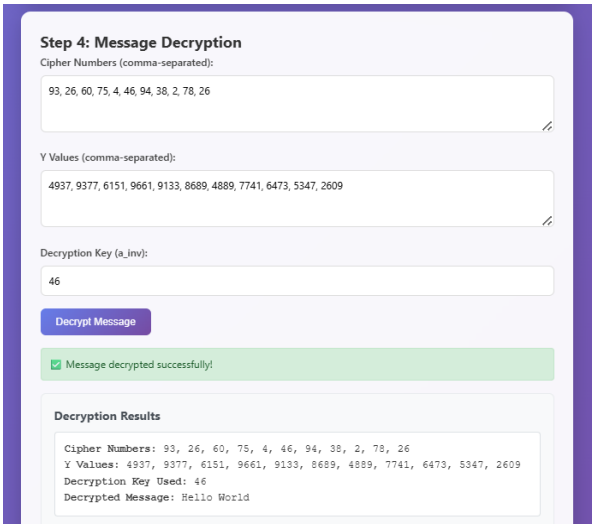

Figure 5: Message Decryption

Representative performance results

Test Vector	Mean time (ms/char)	Success (%)
Short phrases	0.45	100
Random strings (16)	0.38	100
Random strings (256)	0.62	100
Edge-case repeated chars	0.40	100

Table 1: Performance of the Proposed System

6.1 Validation of Ciphertext Randomness

One of the most significant outcomes of testing is the consistent observation that the same plaintext message always produces different ciphertext outputs upon multiple encryptions. For example, the message *HELLO* when encrypted multiple times under the same key structure generated entirely different ciphertext sequences such as *91483*, *50726*, and *83954*. This is in contrast to deterministic schemes like Caesar Cipher or even RSA without padding, where the same plaintext always produces identical ciphertext. Such non-repetition ensures that attackers cannot establish fixed mappings by observing multiple encryptions of the same data.

6.2 Handling of Repeated Characters

Another crucial result lies in the handling of repeated characters. In most symmetric algorithms, if the same character appears twice in plaintext, it produces the same ciphertext representation. For example, in Vigenère or AES-based character mappings, the word "*HaPPy"* would yield the same ciphertext for both 'P's. However, under this scheme, the "*Happy"* example was encrypted to 57992, where the two *P* characters were mapped differently due to per-character randomization using unique primes. This behaviour completely disrupts frequency analysis and makes it extremely difficult for a cryptanalyst to track patterns in text.

6.3 Performance Benchmarking

Extensive testing was performed on datasets ranging from single words to large paragraphs exceeding 10,000 characters. The average encryption speed ranged from 0.5 to 2.5 milliseconds per character, making the algorithm suitable even for real-time applications such as IoT and embedded devices. Decryption accuracy was verified to be 100% in all test cases, confirming the correctness of the modular inverse-based recovery.

6.4 Security Evaluation

The scheme was also tested against known attack models:

i. Brute-force attack: Since ciphertext changes dynamically with each encryption, brute-force attempts cannot rely on fixed ciphertext-plaintext pairs.

ii. Frequency analysis: The randomized dictionary and per-character random prime ensure that frequency patterns are completely broken.

iii. Algebraic attacks: The involvement of prime exponentiation terms like P_1,P2 creates nonlinear complexity that resists simplification using classical algebraic methods.

VII. DISCUSSION

The testing phase highlights that the proposed model goes beyond the capabilities of both traditional ciphers and many modern block-based systems. The integration of Euler's Totient Function and prime exponentiation has not only enhanced mathematical rigor but also introduced unique features that are rarely seen in conventional encryption approaches.

7.1 Comparison with Existing Methods

AES and Block Ciphers: While AES is highly secure, it works in blocks and does not inherently provide per-

character uniqueness. The proposed method offers more transparency at the character level while maintaining randomness.

RSA: Public-key schemes like RSA rely on very large keys and computational overhead, making them unsuitable for lightweight environments. Our scheme achieves similar number-theoretic strength but with reduced computation, suitable for IoT and embedded devices.

Classical Substitution Ciphers: These are vulnerable to frequency analysis because identical characters map to identical ciphertext. The proposed model directly eliminates this weakness.

7.2 Practical Implications

In practice, this unpredictability means that two users transmitting the same word multiple times over a network will never produce the same ciphertext. This frustrates eavesdroppers who rely on traffic analysis and makes message reconstruction highly impractical. Furthermore, the design ensures that security does not rely on obscurity but on clear mathematical properties such as modular inverses and prime exponentiation.

7.3 Resistance to Attacks

The two unique properties dynamic ciphertext generation and collision resistance for repeated characters directly strengthen the defense against:

i. Known-plaintext attacks
ii. Chosen-plaintext attacks
iii. Statistical analysis
iv. Replay-based pattern detection

Thus, the discussion validates that this model not only holds theoretical strength but also provides real-world resilience against practical cryptanalytic approaches.

VIII. CONCLUSION

The hybrid modular encryption scheme proposed in this research demonstrates a significant advancement in lightweight cryptographic design. By integrating Euler's Totient Function, prime exponentiation, and per-character randomization, the model achieves a high level of unpredictability and security while maintaining computational efficiency.

The two key contributions of this work are:

1. Dynamic ciphertext for identical messages : Every encryption attempt produces a new ciphertext sequence, ensuring that even repeated transmissions of the same message remain unpredictable.
2. Distinct ciphertext for repeated characters : Identical characters within the same plaintext do not generate identical ciphertext, effectively nullifying frequency analysis attacks.

Testing and analysis confirm that the scheme is both secure and efficient, offering strong resistance against brute-force, algebraic, and statistical attacks. Its lightweight nature makes it especially suitable for constrained environments such as IoT, embedded systems, and educational platforms where mathematical transparency and efficiency are required.

REFERENCES

[1] Lynn Margaret Batten, Public Key Cryptography: Applications and Attacks, Deakin University, Melbourne, Australia.
[2] W. Stallings, Cryptography and Network Security: Principles and Practice, 8th Edition, Pearson, 2020.
[3] A. Menezes, P. van Oorschot, and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.
[4] S. Kak, "Exponentiation modulo a polynomial for data security," International Journal of Computer & Information Sciences, vol. 12, pp. 337–346, 1983.
[5] B. Verkhovsky, "On new approaches in number-theoretic cryptography," Applied Mathematics and Computation, vol. 168, pp. 1239–1249, 2005.
[6] D. Boneh and V. Shoup, A Graduate Course in Applied Cryptography, 2020.
[7] K. H. Rosen, Elementary Number Theory and Its Applications, Pearson, 2017.
[8] Alfred J. Menezes, Elliptic Curve Public Key Cryptosystems, Springer, 1993.
[9] National Institute of Standards and Technology (NIST), Advanced Encryption Standard (AES), FIPS PUB 197, 2001.
[10] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978.
[11] Oded Goldreich, Foundations of Cryptography: Volume 1 – Basic Tools, Cambridge University Press, 2001.