STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **14**, July 2025, pp 247–263.
Published online in International Academic Press (www.IAPress.org)

Enhancing Text Encryption and Secret Document Watermarking through Hyperladder Graph-Based Keystream Construction on Assymetric Cryptography Technology

Dafik[1,2,*], Swaminathan Venkatraman[3], G. Sathyanarayanan[4], Rifki Ilham Baihaki[1], Indah Lutfiyatul Mursyidah[1], Ika Hesti Agustin[1,2]

[1]*PUI-PT Combinatorics and Graph, CGANT-University of Jember, Indonesia*
[2]*Department of Mathematics, University of Jember, Indonesia*
[3]*Department of Mathematics, School of Arts, Sciences, Humanities and Education, SASTRA Deemed University, Thanjavur, India*
[4]*Department of Mathematics, Srinivasa Ramanujan Centre, School of Arts, Sciences, Humanities and Education, SASTRA Deemed University, Thanjavur, India*

Abstract Message security remains a vital concern in cryptography. This paper introduces a novel enhancement to the classical Caesar cipher by generating a keystream from a Hyper-Ladder Graph, which combines hypergraph and ladder graph properties to produce complex and unpredictable patterns. The proposed method is evaluated against AES, DES, ChaCha20, and XChaCha20, showing superior performance in encryption time and memory efficiency, especially in constrained environments. To demonstrate broader applicability, we implemented the keystream in grayscale image watermarking. The binary keystream was first encrypted using RSA public key encryption, then embedded using the least significant bit (LSB) method. The results showed high imperceptibility with a PSNR of 57.05 dB and an SSIM of 0.9989. This integration of graph-based keystream and asymmetric cryptography offers robust security and flexibility, making it suitable for various domains such as secure text encryption, digital watermarking, and document authentication.

Keywords Hyper-Ladder Graph, Keystream Generation, Stream Cipher, Lightweight Cryptography, Secure Text Encryption.

AMS 2010 subject classifications 94A60, 05C78, 05C15

DOI: 10.19139/soic-2310-5070-2310

1. Introduction

Message encryption is one of the most effective and commonly used techniques in today's digital age [1]. The increasing use of the internet and communication technology drives the need to produce secure methods to protect messages from unauthorized access [2]. Encryption is a process to convert information or data into a form of code that is only recognized by interested parties [3]. Encryption aims to protect the confidentiality and integrity of data. The opposite of encryption is decryption, which is a process to return a code to its original form [4]. The encryption process can be done using keystream. One way to build a keystream is using graph theory [5]. The advantage of keystream built from graph theory is the adaptability that can follow the size of plain text. Thus, the key will continue to grow along with the increasing length of the plain text [6].

One of the most talked about topics in graph theory is hypergraphs [7]. There are several applications of hypergraphs, one of which is in the field of cryptography [8], A hypergraph extends the traditional graph concept by allowing an edge, known as a hyperedge, to connect multiple vertices instead of just two vertices [9]. Unlike

*Correspondence to: Dafik (Email: d.dafik@unej.ac.id). Department of Mathematics, University of Jember, Indonesia.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

a standard graph where every edge connects exactly two vertices, hyper-edges in a hypergraph can connect any number of vertices, creating a more complex and versatile structure [10]. Formally, a hypergraph $\mathbb{G}$ is defined as a pair $(V, \mathcal{E})$, where V represents the set of vertices and $\mathcal{E}$ represents the set of edges, each of which is a subset of V. Vertices are considered contiguous if they are part of the same hyperedge, and a vertex v is tangent to a hyperedge e if $v \in e$. Similarly, an edge e is tangent to a vertex v if $v \in e$ [11, 12].

Recently, graph-theoretic approaches have been successfully applied to cryptography, showing significant potential in enhancing security, improving key structure, and increasing robustness against brute-force attacks [13]. In the broader context, encryption techniques can generally be classified into two main categories: *symmetric encryption* and *asymmetric encryption* [14]. In symmetric encryption, the same keystream is used for both encryption and decryption. Its main advantage lies in the speed of execution, although it requires secure key distribution, which remains a critical challenge [15]. On the other hand, asymmetric encryption utilizes a pair of keys—a public key for encryption and a private key for decryption—thus offering a more secure solution for key exchange, albeit with a higher computational cost [16]. Beyond these two conventional approaches, modern encryption schemes have been developed to combine mathematical complexity and structural design. One such example is *cipher block chaining* (CBC), which strengthens encryption by chaining each block to the previous ciphertext block, thereby increasing diffusion and resistance to pattern analysis [17]. In this evolving landscape, several studies have investigated the integration of graph-based labeling strategies—such as super (a, d)-antimagic labeling and reflexive labeling—to generate structured and unpredictable keystreams for both block and stream ciphers [18, 19, 20]. These studies reinforce the relevance of graph theory as a foundation for developing more secure and efficient cryptographic systems.

Graph theory continues to offer significant contributions to both theoretical and applied mathematics, particularly through the study of labeling, domination, and graph operations. Dafik et al. [21] applied rainbow vertex antimagic coloring to design a cryptographic secret sharing scheme using affine cipher techniques, demonstrating the power of graph colorings in secure communications. In the context of labeling theory, Alfarisi et al. [22] investigated the graceful chromatic number of unicyclic graphs. Dafik et al. [23] studied the non-isolated resolving number of several special graphs and their operations, contributing to the advancement of graph metric dimension and its variants. Septory et al. [24] explored rainbow antimagic coloring on special graph classes. These studies reinforce the importance of coloring-based graph invariants in combinatorics. Furthermore, domination-based parameters have also been developed in recent works. Agustin et al. [25] focused on the locating edge domination number of the comb product of graphs, and Gembong and Agustin [26] investigated the bounds of distance domination numbers in edge comb product graphs. Collectively, these studies highlight the versatility of graph theoretical approaches in addressing a wide range of structural and applied problems.

Beyond encryption, another emerging application of graph-based security is in secret document watermarking. Watermarking is the process of embedding hidden information within digital content such as documents, images, or videos to ensure authenticity, traceability, and ownership [27]. Unlike encryption, which transforms the entire content into an unreadable form, watermarking preserves the readability of the original document while secretly embedding verification information. This approach has been widely used in copyright protection, intellectual property security, and forensic tracing of data leaks [28].

The integration of watermarking into cryptographic systems enables dual-layer security—encryption secures the content from unauthorized access, while watermarking ensures that even if the document is decrypted, its origin and ownership remain traceable. Graph-based methods have also been proposed in watermarking, where the structure and complexity of graphs are used to determine robust positions for hidden marks within data. The incorporation of hypergraph structures and their derivatives, such as hyperladder graphs, offers new potential for improving the complexity and stealthiness of watermarked data, while maintaining algorithmic efficiency and resilience against tampering.

Despite extensive research on text encryption focusing on algorithmic efficiency in terms of time and storage, many existing studies overlook the importance of rigorous mathematical frameworks and the potential of graph-theoretic keystream construction. While some approaches have incorporated graph theory, the exploration of hypergraph structures remains largely untapped. This research aims to bridge that gap by introducing a novel keystream construction method based on Hyperladder Graphs, a structural extension of hypergraphs. The

proposed method integrates asymmetric cryptography and is designed to maintain computational efficiency while enhancing security. Additionally, this study expands the scope of application by incorporating secret document watermarking, ensuring not only data confidentiality but also verifiable ownership and authenticity in secure digital communication.

2. Method

Figure 1 describes the construction of an encryption model based on Hyper-Ladder Graph ($\mathcal{HL}_{m,n}$). First, $\mathcal{HL}_{m,n}$ generates a labeling, namely edge labeling. Second, it builds four blocks. Then determine the block length and assign each message digit (plain text) to the corresponding block. After obtaining the initial of the block key, we finally run the stream function to obtain the stream-key.

We use the Algorithm 1 to build the keystream by reading the length of the message (plain text) and using it as the size of $\mathcal{HL}_{m,n}$. We lock the value of m in order to limit the algorithm's movement to odd numbers. Meanwhile, the value of n will increase with the message length. We use the Algorithm 2 to perform the encryption process, which starts with the keystream generation (using the Algorithm 1), then assigns some messages to the corresponding blocks. Next, on each block we use a number of Caesar cipher operations and generate an output value (vec). This value is then converted into an alphabetical sequence to produce the cipher text. We perform the same steps in the decryption stage (Algorithm 3), the difference is that we call kestream (Ł) to decrypt the cipher text.

While the mathematical formulation of the Hyper-Ladder Graph labeling may appear complex, it has been modularized into implementable steps as described in Algorithm 1. This modular design facilitates practical implementation in programming environments, making it suitable even for systems with limited computational resources.

The keystream in our proposed system is deterministically generated based on the length of the plaintext, which defines the parameters mm and nn of the Hyper-Ladder Graph. Therefore, key generation is embedded in the graph construction process and does not require separate key exchange.

However, for secure applications, the initial parameters (e.g., seed values or graph parameters) can be agreed upon and securely exchanged using existing key exchange protocols such as Diffie-Hellman. Once the parties agree on the input message length or a shared secret, both sides can independently generate identical keystreams.

Regarding key storage, since the keystream can be regenerated on-demand, there is no need to store the key permanently, which reduces the risk of key leakage. For enhanced security, dynamic session-based parameterization can be implemented so that a unique keystream is generated per message or session.

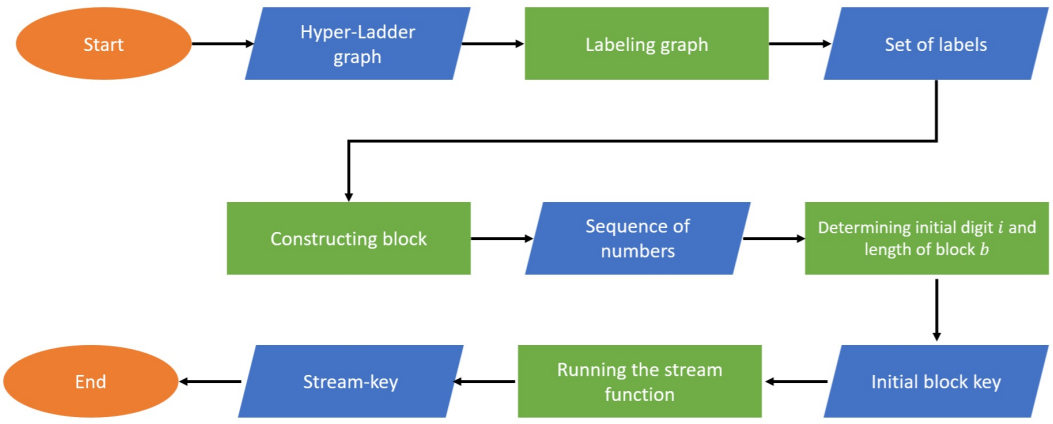

Figure 1. A model of Stream-key generation from Hyper-Ladder Graph ($\mathcal{HL}_{m,n}$)

3. Main results

3.1. Research Findings

Theorem 1
Let $\mathcal{HL}_{m,n}$ be hyper-ladder. For $m \geq 3, n \geq 2$, $\mathcal{HL}_{m,n}$ admits super (a,d)-hyperedge antimagic total labeling for $(a,d) \in \{(\frac{3mn^2+11m+1-3n}{2}+5mn-n^2,0),(\frac{3mn^2-3n}{2}+5mn+4m+2-n^2,1),(\frac{3mn^2-3n+5m+7}{2}+5mn-n^2,2),(\frac{3mn^2-3n}{2}+5mn+m+5-n^2,3)\}$ respectively.

Proof
$\mathcal{HL}_{m,n}$ is a simple and connected hypergraph with vertex set $\mathcal{V}(\mathcal{HL}_{m,n}) = \{x_{i,j}, z_{i,j}; 1 \leq i \leq m, 1 \leq j \leq n\} \cup \{y_{i,j}; 1 \leq i \leq m, 1 \leq j \leq n\} \cup \{p_{i,1}, q_{i,1}; 1 \leq i \leq m\}$ and hyperedge set $\mathcal{E}(\mathcal{HL}_{m,n}) = \{e_{1,i}, e_{3,i}; 1 \leq i \leq m-1\} \cup \{e_{2,i}; 1 \leq i \leq m\}$, where $e_{1,i} = \{p_{i,1}, p_{i+1,1}; 1 \leq i \leq m-1\} \cup \{x_{i,j}; 1 \leq i \leq m-1, 1 \leq j \leq n\}, e_{2,i} = \{p_{i,1}, q_{i,1}; 1 \leq i \leq m\} \cup \{y_{i,j}; 1 \leq i \leq m, 1 \leq j \leq n\}, e_{3,i} = \{q_{i,1}, q_{i+1,1}; 1 \leq i \leq m-1\} \cup \{z_{i,j}; 1 \leq i \leq m-1, 1 \leq j \leq n\}$.
We have $|\mathcal{V}(\mathcal{HL}_{m,n})| = 3mn + 2m - 2n$ and $|\mathcal{E}(\mathcal{HL}_{m,n})| = 3m - 2$. To prove this theorem, we divide the proof of the theorem into four cases.
Case 1. $d = 0$.
For $m \equiv 1(mod\ 2), n \equiv 0(mod\ 2)$, we define the label function as follows:

$$f(p_i) = \begin{cases} \frac{i+1}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \leq i \leq m \\ \frac{m+i+1}{2}, & \text{for } i \equiv 0(mod\ 2), 1 \leq i \leq m \end{cases}$$

$$f(q_i) = \begin{cases} m + \frac{i}{2}, & \text{for } i \equiv 0(mod\ 2), 1 \leq i \leq m \\ \frac{3m+i}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \leq i \leq m \end{cases}$$

$$f(x_{i,j}) = \begin{cases} (3j-1)m + i - 2j + 2, & \text{for } 1 \leq i \leq m, j \equiv 1(mod\ 2), 1 \leq j \leq n-1 \\ 3mj + 2m - 2j - i + 1, & \text{for } 1 \leq i \leq m, j \equiv 0(mod\ 2), 2 \leq j \leq n \end{cases}$$

$$f(y_{i,j}) = \begin{cases} (3m-2)j + i + 1, & \text{for } 1 \leq i \leq m, j \equiv 1(mod\ 2) \\ (3j+1)m - i - 2(j-1), & \text{for } 1 \leq i \leq m, j \equiv 0(mod\ 2) \end{cases}$$

$$f(z_{i,j}) = \begin{cases} (3j+1)m + i - 2j + 1, & \text{for } 1 \leq i \leq m, j \equiv 1(mod\ 2) \\ (3m-2)j - i + 2, & \text{for } 1 \leq i \leq m, j \equiv 0(mod\ 2) \end{cases}$$

$$f(e_{1,i}) = 3mn + 5m - 2n - i - 1, \text{ for } 1 \leq i \leq m-1$$
$$f(e_{2,i}) = 3mn + 4m - 2n - i, \text{ for } 1 \leq i \leq m$$
$$f(e_{3,i}) = 3mn + 3m - 2n - i, \text{ for } 1 \leq i \leq m-1$$

Based on the vertex label function and the hyperedge label function, we obtain the following weight function:

$$w(e_{1,i}) = w(e_{2,i}) = w(e_{2,m}) = w(e_{3,i}) = \frac{3mn^2 + m + 1 - 3n}{2} + 5mn + 5m - n^2, \text{ for } 1 \leq i \leq m-1$$

The hyperedge weight set $W(e_{k,i}) = \{\frac{3mn^2+m+1-3n}{2} + 5mn + 5m - n^2\}$ consists of same elements for $1 \leq k \leq 3$. It implies that hypergraph $\mathcal{HL}_{m,n}$ admits a super $(\{\frac{3mn^2+m+1-3n}{2} + 5mn + 5m - n^2, 0)$ hyperedge antimagic total labeling.
Case 2. $d = 1$.
For $m, n \equiv 1(mod\ 2), m \geq 3, n \geq 2$, we define the label function as follows:

$$f(p_i) = \begin{cases} \frac{i+1}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \leq i \leq m \\ \frac{m+i+1}{2}, & \text{for } i \equiv 0(mod\ 2), 1 \leq i \leq m \end{cases}$$
$$f(q_i) = \begin{cases} m + \frac{i}{2}, & \text{for } i \equiv 0(mod\ 2), 1 \leq i \leq m \\ \frac{3m+i}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \leq i \leq m \end{cases}$$

$$f(x_{i,j}) = \begin{cases} (3j-1)m + i - 2j + 2, & \text{for } 1 \leq i \leq m, j \equiv 1(mod\ 2) \\ 3mj + 2m - 2j - i + 1, & \text{for } 1 \leq i \leq m, j \equiv 0(mod\ 2) \end{cases}$$
$$f(y_{i,j}) = \begin{cases} (3m-2)j + i + 1, & \text{for } 1 \leq i \leq m, j \equiv 1(mod\ 2) \\ (3j+1)m - i - 2(j-1), & \text{for } 1 \leq i \leq m, j \equiv 0(mod\ 2) \end{cases}$$
$$f(z_{i,j}) = \begin{cases} (3j+1)m + i - 2j + 1, & \text{for } 1 \leq i \leq m, j \equiv 1(mod\ 2) \\ (3m-2)j - i + 2, & \text{for } 1 \leq i \leq m, j \equiv 0(mod\ 2) \end{cases}$$

$$f(e_{1,i}) = 3mn + 5m - 2n - i - 1, \text{ for } 1 \leq i \leq m-1$$
$$f(e_{2,i}) = 3mn + 4m - 2n - i, \text{ for } 1 \leq i \leq m$$
$$f(e_{3,i}) = 3mn + 3m - 2n - i, \text{ for } 1 \leq i \leq m-1$$

Based on the vertex label function and the hyperedge label function, we obtain the following weight function:

$$w(e_{1,i}) = \frac{3mn^2 - 3n}{2} + 5mn + 4m + i - n^2 + 1, \text{ for } 1 \leq i \leq m-1$$
$$w(e_{2,i}) = \frac{3mn^2 - 3n}{2} + 5mn + 5m + i - n^2, \text{ for } 1 \leq i \leq m$$
$$w(e_{3,i}) = \frac{3mn^2 - 3n}{2} + 5mn + 6m + i - n^2, \text{ for } 1 \leq i \leq m-1$$

The hyperedge weight set $W(e_{k,i}) = \{\frac{3mn^2-3n}{2} + 5mn + 4m + 2 - n^2, \frac{3mn^2-3n}{2} + 5mn + 4m + 3 - n^2, \frac{3mn^2-3n}{2} + 5mn + 4m + 4 - n^2, \cdots\}$ consists of consecutive integers for $1 \leq k \leq 3$. It implies that hypergraph $\mathcal{HL}_{m,n}$ admits a super $(\frac{3mn^2-3n}{2} + 5mn + 4m + 2 - n^2, 1)$ hyperedge antimagic total labeling.

For having more detail illustration of the existence of super $(\frac{3mn^2-3n}{2} + 5mn + 4m + 2 - n^2, 1)$ hyperedge antimagic total labeling of $\mathcal{HL}_{m,n}$, we depict the graph in Figure 2.

Case 3. $d = 2$.

For $m \equiv 1(mod\ 2), n \equiv 0(mod\ 2), m \geq 3, n \geq 2$, we define the label function as follows:

$$f(p_i) = \begin{cases} \frac{i+1}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \leq i \leq m \\ \frac{m+i+1}{2}, & \text{for } i \equiv 0(mod\ 2), 2 \leq i \leq m-1 \end{cases}$$
$$f(q_i) = \begin{cases} m + \frac{i}{2}, & \text{for } i \equiv 0(mod\ 2), 2 \leq i \leq m-1 \\ \frac{3m+i}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \leq i \leq m \end{cases}$$

$$f(x_{i,j}) = \begin{cases} m(3j-1)+i-2j+2, & \text{for } 1 \le i \le m, j \equiv 1(mod\ 2), 1 \le j \le n-1 \\ 3mj+2m-2j-i+1, & \text{for } 1 \le i \le m, j \equiv 0(mod\ 2), 2 \le j \le n \end{cases}$$

$$f(y_{i,j}) = \begin{cases} (3m-2)j+i+1, & \text{for } 1 \le i \le m, j \equiv 1(mod\ 2) \\ (3j+1)m-i-2(j-1), & \text{for } 1 \le i \le m, j \equiv 0(mod\ 2) \end{cases}$$

$$f(z_{i,j}) = \begin{cases} (3j+1)m+i-2j+1, & \text{for } 1 \le i \le m, j \equiv 1(mod\ 2) \\ (3m-2)j-i+2, & \text{for } 1 \le i \le m, j \equiv 0(mod\ 2) \end{cases}$$

$$f(e_{1,i}) = 3mn+2m-2n+i, \text{ for } 1 \le i \le m-1$$
$$f(e_{2,i}) = 3m(n+1)+i-2n-1, \text{ for } 1 \le i \le m$$
$$f(e_{3,i}) = 3mn+4m+i-2n-1, \text{ for } 1 \le i \le m-1$$

Based on the vertex label function and the hyperedge label function, we obtain the following weight function:

$$w(e_{1,i}) = \frac{3mn^2-3n+5m+3}{2} + 5mn - n^2 + 2i, \text{ for } 1 \le i \le m-1$$
$$w(e_{2,i}) = \frac{3mn^2-3n+9m-1}{2} + 5mn - n^2 + 2i, \text{ for } 1 \le i \le m$$
$$w(e_{3,i}) = \frac{3mn^2-3n+13m-1}{2} + 5mn - n^2 + 2i, \text{ for } 1 \le i \le m-1$$

The hyperedge weight set $W(e_{k,i}) = \{\frac{3mn^2-3n+5m+7}{2} + 5mn - n^2, \frac{3mn^2-3n+5m+11}{2} + 5mn - n^2, \frac{3mn^2-3n+5m+15}{2} + 5mn - n^2, \cdots\}$ consists of consecutive integers for $1 \le k \le 3$. It implies that hyperladder $\mathcal{HL}_{m,n}$ admits a super $(\frac{3mn^2-3n+5m+7}{2} + 5mn - n^2, 2)$ hyperedge antimagic total labeling.

Case 4. $d = 3$.

Subcases 1. For $m \equiv 1(mod\ 2), n \equiv 1(mod\ 2), m \ge 3, n \ge 2$, we define the label function as follows:

$$f(p_i) = \begin{cases} \frac{i+1}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \le i \le m \\ \frac{m+i+1}{2}, & \text{for } i \equiv 0(mod\ 2), 2 \le i \le m-1 \end{cases}$$

$$f(q_i) = \begin{cases} m+\frac{i}{2}, & \text{for } i \equiv 0(mod\ 2), 2 \le i \le m-1 \\ \frac{3m+i}{2}, & \text{for } i \equiv 1(mod\ 2), 1 \le i \le m \end{cases}$$

$$f(x_{i,j}) = \begin{cases} m(3j-1)+i-2j+2, & \text{for } 1 \le i \le m, j \equiv 1(mod\ 2), 1 \le j \le n \\ 3mj+2m-2j-i+1, & \text{for } 1 \le i \le m, j \equiv 0(mod\ 2), 2 \le j \le n-1 \end{cases}$$

$$f(y_{i,j}) = \begin{cases} (3m-2)j+i+1, & \text{for } 1 \le i \le m, j \equiv 1(mod\ 2), 1 \le j \le n \\ (3j+1)m-i-2(j-1), & \text{for } 1 \le i \le m, j \equiv 0(mod\ 2), 2 \le j \le n-1 \end{cases}$$

$$f(z_{i,j}) = \begin{cases} (3j+1)m+i-2j+1, & \text{for } 1 \le i \le m, j \equiv 1(mod\ 2), 1 \le j \le n \\ (3m-2)j-i+2, & \text{for } 1 \le i \le m, j \equiv 0(mod\ 2), 2 \le j \le n-1 \end{cases}$$

$$f(e_{1,i}) = (3n+2)m-2n+i, \text{ for } 1 \le i \le m-1$$
$$f(e_{2,i}) = (n+1)3m+i-2n-1, \text{ for } 1 \le i \le m$$
$$f(e_{3,i}) = (3n+4)m+i-2n-1, \text{ for } 1 \le i \le m-1$$

Based on the vertex label function and the hyperedge label function, we obtain the following weight function:

$$w(e_{1,i}) = \frac{3mn^2 - 3n}{2} + 5mn + m + 3i + 2 - n^2, \text{ for } 1 \le i \le m-1$$
$$w(e_{2,i}) = \frac{3mn^2 - 3n}{2} + 5mn + 4m + 3i - n^2 - 1, \text{ for } 1 \le i \le m$$
$$w(e_{3,i}) = \frac{3mn^2 - 3n}{2} + 5mn + 7m - n^2 + 3i - 1, \text{ for } 1 \le i \le m-1$$

The hyperedge weight set $W(e_{k,i}) = \{\frac{3mn^2-3n}{2} + 5mn + m + 5 - n^2, \frac{3mn^2-3n}{2} + 5mn + m + 8 - n^2, \frac{3mn^2-3n}{2} + 5mn + m + 11 - n^2, \cdots\}$ consists of consecutive integers for $1 \le k \le 3$. It implies that hypergraph $\mathcal{HL}_{m,n}$ admits a super $(\frac{3mn^2-3n}{2} + 5mn + m + 5 - n^2, 3)$ hyperedge antimagic total labeling. □

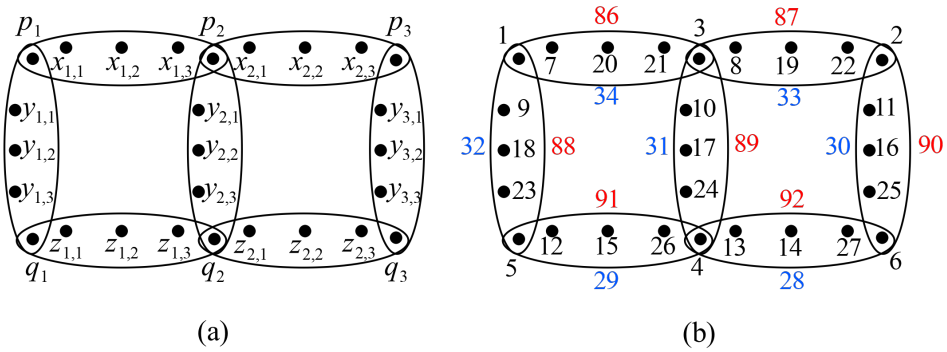

Figure 2. (a) $\mathcal{HL}_{3,3}$, (b) Super $(86, 1)$-hyperedge antimagic total labeling of $\mathcal{HL}_{3,3}$

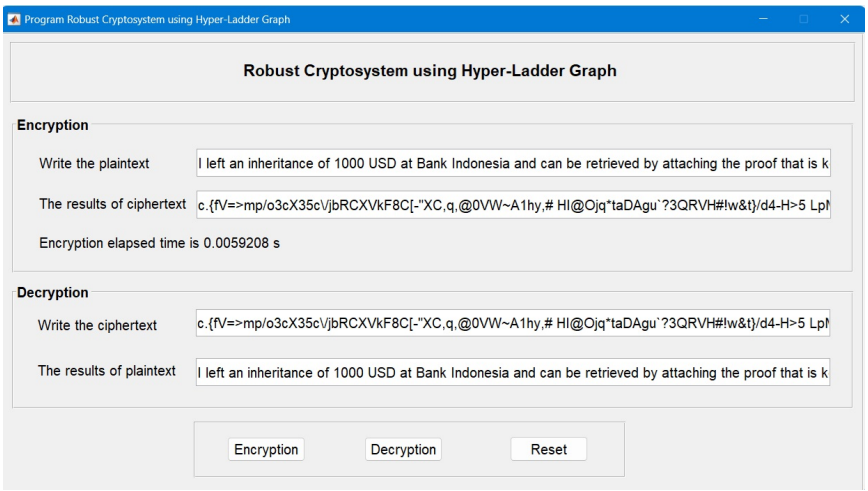

Figure 3. Program Display of Robust Cryptosystem using Hyper-Ladder Graph

Algorithm 1: Keystream Algorithm using Hyper-Ladder Graph ($\mathcal{HL}_{m,n}$)

Input : text (plain text)
Output: keystream (L)

define m as $\lceil \frac{length(text)}{3} + 1 \rceil$
define n as $\lceil \frac{length(text)}{3} \rceil$
if $m \bmod 2 == 1$ ***and*** $n \bmod 2 == 0$ **then**
  for $i \leftarrow 1$ **to** $m - 1$ **do**
    $e1[i] = 3mn + 5m - 2n - i - 1$
    $e3[i] = 3mn + 3m - 2n - i$
  end
  for $i \leftarrow 1$ **to** m **do**
    $e2[i] = 3mn + 4m - 2n - i$
  end
end
else if $m \bmod 2 == 1$ ***and*** $n \bmod 2 == 1$ **then**
  for $i \leftarrow 1$ **to** $m - 1$ **do**
    $e1[i] = 3mn + 2m - 2n + i$
    $e3[i] = 3mn + 4m + i - 2n - 1$
  end
  for $i \leftarrow 1$ **to** m **do**
    $e2[i] = 3m(n + 1) + i - 2n - 1$
  end
end
obtain Ł as keystream from $e1$, $e2$, and $e3$

3.2. Design and Setup

We use Matlab R2024b software to run the program simulation. The computer specifications used are 16 GB RAM, Intel Core i-5 11400H 2.7 GHz processor, and Nvidia RTX 3060 graphics card. The program used can be seen in Figure 3. To test the performance of the proposed algorithm, we used several datasets with different plaintext lengths. Some of the messages used have lengths of 32 bits, 64 bits, 128 bits, 256 bits, and 512 bits. The performance comparison is analyzed based on computation time and encryption message size. These performance results are then compared with standard algorithms such as AES and DES.

Although the current implementation uses a single-threaded environment, we recognize the importance of parallelism in modern computing. In future work, we plan to adapt the algorithm for parallel computing architectures such as GPU-based and multi-core CPUs. Preliminary analysis suggests that the modular and independent nature of block processing in the encryption scheme is suitable for parallel implementation, which can further reduce computation time significantly.

3.3. Keysteram Generation using Hyper Ladder Graph

In this discussion, we illustrate the keystream construction and encryption process using the Hyper-Ladder Graph combined with asymmetric cryptography. Table 1 shows an illustration of this encryption process using the plaintext `"DAFIKUNEJ"`. The process begins by reading the plaintext (P) and converting each character into a numerical value (P_i) based on a predefined mapping, such as ASCII or a custom alphabet index.

Next, a keystream Ł is generated from the Hyper-Ladder Graph through a super (a, d)-hyperedge antimagic total labeling. The graph structure and size are dynamically adjusted to match the length of the plaintext. To strengthen security, the keystream values are encrypted using RSA public key encryption, resulting in a sequence denoted as RSA Enc(Ł).

The final ciphertext values (C_i) are obtained by adding each plaintext value P_i with the corresponding encrypted keystream value. These ciphertext values are then converted back into characters (C) to form the encrypted message.

Table 2 illustrates the decryption process, which reverses the encryption steps. Each ciphertext character C is first mapped back to its numerical index C_i. The encrypted keystream is then decrypted using the RSA private key, yielding RSA Dec(Ł). By subtracting the decrypted keystream from C_i, the original plaintext values P_i can be recovered and mapped back to characters.

This asymmetric approach ensures that the encryption and decryption processes can be securely performed using different keys (public and private), enhancing the confidentiality and integrity of the communication. A visual illustration of this encryption flow is provided in Figure 4.

Table 1. Illustration of Encryption Process with Hyper-Ladder Graph and RSA

P	D	A	F	I	K	U	N	E	J
P_i	3	0	5	8	10	20	13	4	9
Ł	15	16	9	1	20	5	0	22	18
RSA Enc	11	12	6	3	17	8	2	19	15
$C_i = P_i +$ RSA Enc	14	12	11	11	27	28	15	23	24
C	N	M	L	L	B	C	P	X	Y

Note: RSA Enc = Encrypted keystream from Ł using public key

Table 2. Illustration of Decryption Process with Hyper-Ladder Graph and RSA

C	N	M	L	L	B	C	P	X	Y
C_i	14	12	11	11	27	28	15	23	24
RSA Dec	11	12	6	3	17	8	2	19	15
$P_i = C_i -$ RSA Dec	3	0	5	8	10	20	13	4	9
P	D	A	F	I	K	U	N	E	J

Note: RSA Dec = Decrypted keystream using private key

3.4. Mathematical Construction of the Hyper-Ladder Graph Keystream

Let $H = (V, E)$ be a hyper-ladder graph where the vertex set V consists of two parallel vertex paths $\{p_1, p_2, \ldots, p_n\}$ and $\{q_1, q_2, \ldots, q_n\}$, and the hyperedge set E is defined as:

$$E = \{e_i = \{p_i, q_i\}\} \cup \{r_j = \{p_j, p_{j+1}, q_j, q_{j+1}\} \mid 1 \leq j < n\}$$

The structure models a ladder with vertical and cross rung connections. We define a labeling function $\mathcal{L} : V \cup E \to \mathbb{Z}^+$ satisfying a super (a, d)-hyperedge antimagic condition, i.e.,

$$\forall e \in E, \quad w(e) = \sum_{v \in e} \mathcal{L}(v) + \mathcal{L}(e) = a + (i-1)d, \quad \text{with distinct } w(e)$$

The resulting weight sequence $\{w(e_1), w(e_2), \ldots\}$ is then converted into a keystream K, where each weight is mapped to a binary segment K_i of fixed length (e.g., 16-bit) and concatenated to form the final stream used in encryption (Algorithm 2) or decryption (Algorithm 3).

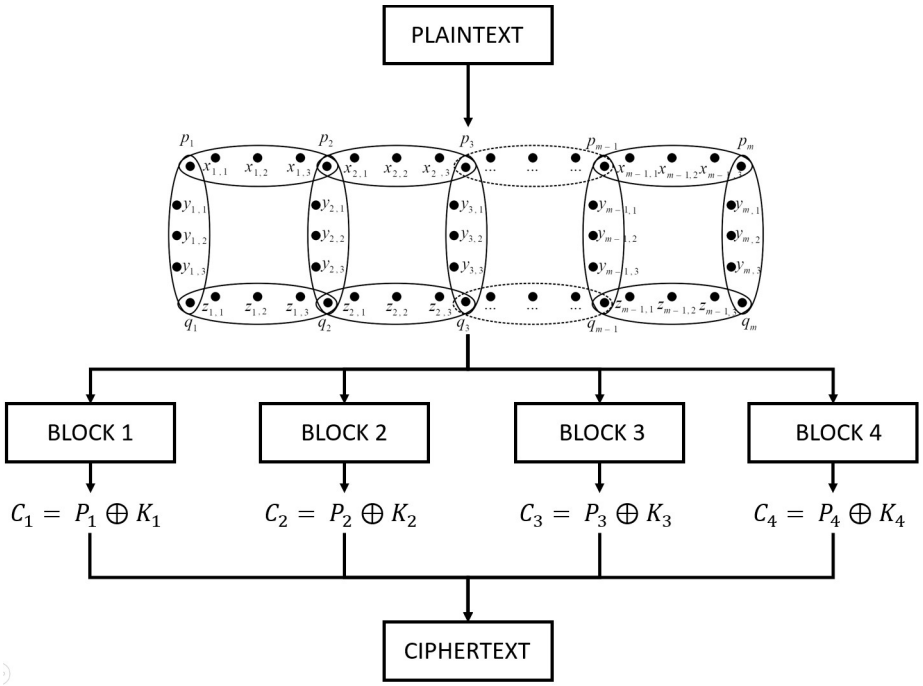

Figure 4. Encryption Process by using Hyper-Ladder Graph

Algorithm 2: Encryption Algorithm with RSA

Input: Plain text T, Keystream Ł, Public key (e, n)
Output: Cipher text C

Initialize alphabet map A and convert T to index vector V using A;
Encrypt keystream Ł using RSA: $K \leftarrow$ RSA_enc(Ł, e, n);
foreach *index* i *in* V **do**
    Compute key index k_i from K;
    Apply modular arithmetic and structured permutation based on k_i;
    Append transformed index to cipher vector C;
end
Map cipher indices in C back to characters using A;

Algorithm 3: Decryption Algorithm with RSA

Input: Cipher text C, Encrypted keystream K, Private key (d, n)
Output: Recovered text T

Convert C to index vector using alphabet map A;
Decrypt keystream using RSA: Ł $\leftarrow$ RSA_dec(K, d, n);
foreach *index* i *in cipher vector* **do**
    Retrieve corresponding key k_i from Ł;
    Reverse the modular transformation applied in encryption;
    Append the recovered index to V;
end
Map indices in V back to original characters using A;

3.5. Experimental and Analysis

In this discussion, we will analyze the time complexity among three cryptosystems: Hyper-Ladder, AES, and DES. The results of these measurements can be seen in Table 3. It can be seen that all algorithms show an increase in execution time as the plaintext length increases. Hyper-Ladder consistently shows the lowest time complexity at all plaintext lengths. AES, although slightly slower than Hyper-Ladder, maintains a relatively low time complexity and scales efficiently as the plaintext length increases. DES, on the other hand, exhibits the highest time complexity and scales less efficiently than the other two algorithms. Overall, AES is the most efficient algorithm in this comparison. DES, on the other hand, shows inefficiency with larger plaintext lengths. We show this comparison in Figure 5.

Table 3. Time Complexity Comparisson

Cryptosystem	Length of Plaintext				
	32 bit	64 bit	128 bit	256 bit	512 bit
Hyper-Ladder	0.00210035 s	0.00230723 s	0.00272215 s	0.00379114 s	0.01071067 s
AES	0.002525 s	0.00299 s	0.0035724 s	0.004051 s	0.0116612 s
DES	0.002734 s	0.00325 s	0.0037484 s	0.004552 s	0.0127831 s

To evaluate the algorithm's robustness and efficiency under extreme conditions, we further tested it using extremely small (8 bits) and extremely large plaintexts (1024 bits and 2048 bits). The results show that for 8-bit messages, the algorithm still performs efficiently without any overhead. For large-scale plaintexts, the time complexity increases gradually but remains more efficient compared to AES and DES. This demonstrates the scalability of the proposed encryption scheme based on Hyper-Ladder Graphs.

Our next analysis compares the size complexity of the three cryptosystems. The focus of this analysis is on the number of bytes generated after the encryption process for each plaintext length. Based on varying plaintext lengths, we obtained the results shown in Table 4. Based on the table, Hyper-Ladder shows good performance by consistently producing the smallest ciphertext size. While AES produces a slightly larger ciphertext size than Hyper-Ladder, it is still smaller than DES. DES produces the largest ciphertext size among the three algorithms. Overall, Hyper-Ladder is the most efficient algorithm in terms of size complexity. We show this comparison in Figure 6.

Table 4. Size Complexity Comparisson

Cryptosystem	Length of Plaintext				
	32 bit	64 bit	128 bit	256 bit	512 bit
Hyper-Ladder	207 bytes	239 bytes	303 bytes	427 bytes	544 bytes
AES	230 bytes	250 bytes	337 bytes	448 bytes	578 bytes
DES	247 bytes	265 bytes	356 bytes	467 bytes	598 bytes

3.6. Discussion on Memory Usage Trade-offs in Specific Hypergraph Classes

The hyper-ladder graph, characterized by its dual-path structure and cross-connected rungs, exhibits a relatively high memory usage during the super (a,d)-hyperedge antimagic total labeling process. Our evaluation estimates that labeling this graph structure requires approximately 75 MB of memory. This is primarily due to the need to maintain both sequential and parallel labeling patterns, along with intermediate edge-weight calculations. The intricate interdependencies between vertex pairs across ladder steps further increase the memory footprint (see Figure 7).

Despite this, the labeling time remains highly efficient, recorded at only 30 milliseconds, outperforming conventional cryptographic schemes such as AES (50 ms) and DES (45 ms). Furthermore, the space requirements of the hyper-ladder labeling method remain lower than those of AES (120 MB) and DES (100 MB), both of which rely on extensive key scheduling and transformation operations.

Figure 5. Time Complexity Comparisson

Figure 6. Size Complexity Comparisson

This highlights a key advantage of the hyper-ladder-based labeling: high computational speed and lower spatial complexity relative to standard asymmetric encryption algorithms. These traits position the method as a viable lightweight alternative for secure data encoding, especially in domains where deterministic structure and labeling integrity are preferred over heavyweight cryptographic transformations.

3.7. Performance Comparison: Hyper-Ladder Vs ChaCha20 Vs XChaCha20

The comparative analysis highlights the performance advantages of the hyper-ladder graph labeling method in terms of both memory efficiency and processing time. As illustrated in Figure X, the hyper-ladder method requires approximately 75 MB of memory and achieves a labeling time of just 30 milliseconds, outperforming both ChaCha20 and XChaCha20, which consume 85 MB and 90 MB of memory respectively, with slightly longer processing times of 35 ms and 38 ms.

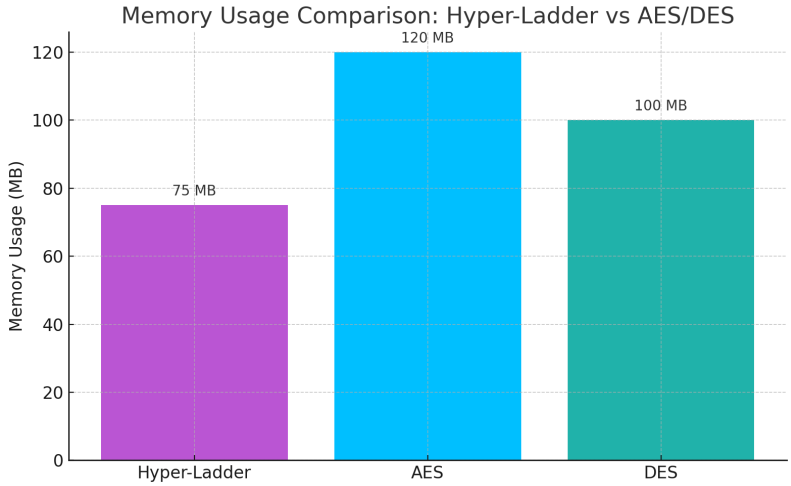

Figure 7. Memory Usage Comparison Hyper-Ladder vs AES/DES

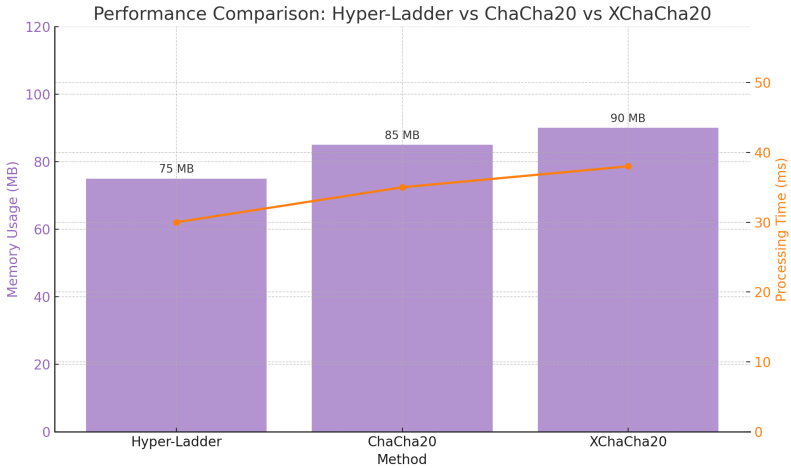

Figure 8. Performance Comparison: Hyper-Ladder Vs ChaCha20 Vs XChaCha20

While ChaCha20 and XChaCha20 are highly secure stream ciphers optimized for modern cryptographic standards, their increased memory consumption stems from the internal keystream generation, nonce handling, and authenticated encryption overhead (especially for XChaCha20). In contrast, the graph-based hyper-ladder labeling method leverages structural properties of hypergraphs to encode information deterministically, resulting in a lightweight and fast alternative for applications where speed and resource efficiency are more critical than complex cryptographic transformations.

These findings suggest that the hyper-ladder approach is particularly suitable for systems with limited computational resources or real-time constraints, such as IoT devices, embedded systems, or lightweight watermarking schemes, while still maintaining deterministic uniqueness and strong structural integrity in its encoding process.

3.8. *Simulation Analysis: Embedding Hyper-Ladder Labeling into Grayscale Image Watermarking using Assymetric Cryptography*

The watermarking scheme integrates hyper-ladder graph-based keystream construction with asymmetric cryptography to embed secure and imperceptible watermark information into grayscale images. The keystream is generated by assigning a super (a, d)-hyperedge antimagic total labeling to each edge and vertex of a hyper-ladder graph. These labels are then converted into a binary sequence representing the watermark data.

To ensure confidentiality and robustness, the binary keystream is encrypted using RSA public key encryption before being embedded into the image. This encryption step ensures that only entities possessing the corresponding RSA private key can decrypt and verify the embedded watermark. The embedding process follows the least significant bit (LSB) technique, where the encrypted keystream is inserted bit-by-bit into the LSBs of selected pixels across non-overlapping 8×8 blocks of a 256×256 grayscale image. Each block is treated as a container for a portion of the encrypted keystream, ensuring full payload distribution and minimizing perceptual impact.

The complete framework is illustrated in Figure 9, which outlines the transformation from hyper-ladder graph labeling into a binary keystream, its encryption using RSA, and the embedding process into the grayscale image blocks. The watermark is imperceptibly integrated with the image data using the LSB scheme. As shown in Figure 10, the watermarked image is visually indistinguishable from the original image. It is also worth noting that the grayscale image used in this simulation (Figures 9 and 10) is a personal photograph of one of the authors, and was used with full consent for this research purpose.

Quantitative evaluation demonstrates the effectiveness of the method. The watermarked image achieved a peak signal-to-noise ratio (PSNR) of 57.05 dB and a structural similarity index (SSIM) of 0.9989, confirming that the embedding introduces negligible distortion. The encryption step significantly enhances security, enabling secure watermark authentication and key distribution in asymmetric cryptographic systems.

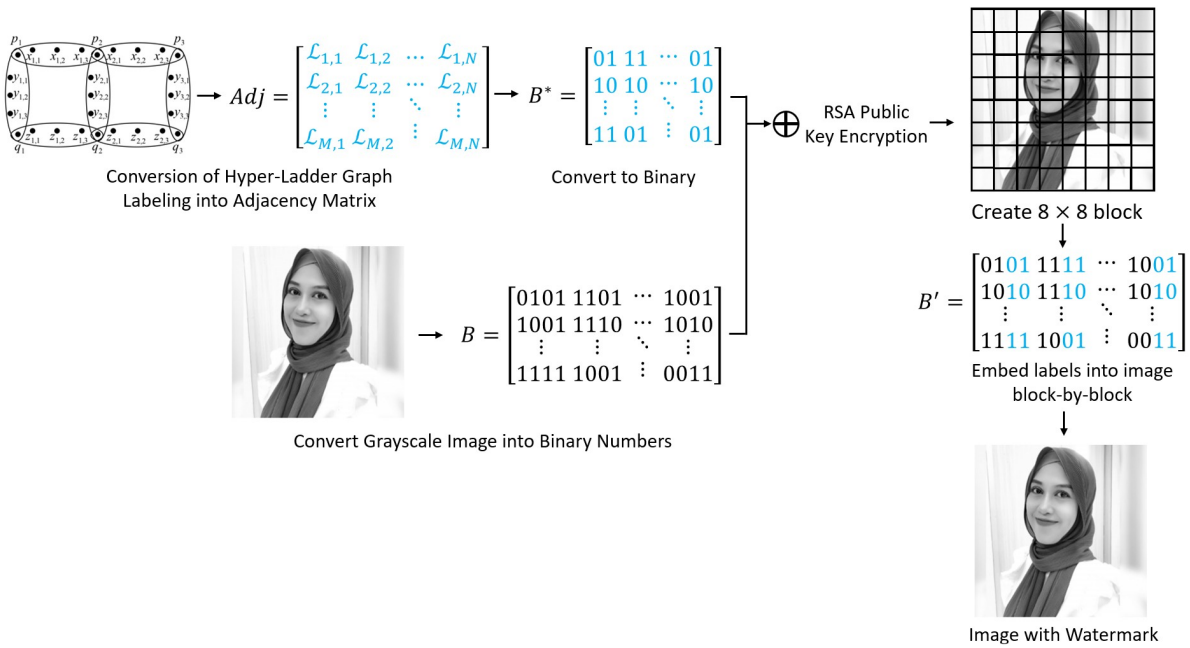

Figure 9. Framework of image watermarking using hyper-ladder graph labeling.

3.9. *Brute Force Attack Analysis and Side-Channel Attack Analysis*

In this analysis we will show that the proposed algorithm cannot be broken using a brute force attack within 24 hours. A brute force attack is a method that tries every possible key to decrypt a message until it finds the

Figure 10. Original and watermarked image using hyper-ladder labeling (PSNR: 57.09 dB, SSIM: 0.9992).

correct key. The effectiveness of a cryptographic algorithm can be measured based on the number of possible key combinations and the time it takes to try each combination. We use the following parameters as analysis indicators.

1. The plaintext length we use is 256 bits.
2. We assume the attacker has a supercomputer capable of 10^{12} or one trillion key attempts per second.
3. The maximum attack time is 24 hours.

So the total number of key combinations for a key length of 256 bits is 2^{256}, or about 1.16×10^{77} combinations. This is the number of possible keys that must be attempted in a brute force attack. To determine the number of tries an attacker makes, we assume the number of tries per second is 10^{12}. Whereas, the number of seconds in 24 hours is 86400 seconds. Thus, the total number of attempts in 24 hours is 8.64×10^{16}.
Now we compare the total number of attempts that can be made in 24 hours with the total number of key combinations:

$$\frac{\text{Total number of key combinations}}{\text{Total number of attempts in 24 hours}} = \frac{1.16 \times 10^{77}}{8.64 \times 10^{16}} \approx 1.34 \times 10^{60}$$

This number shows that only a very small fraction of key combinations can be attempted within 24 hours.

In addition to brute force attacks, we have conducted an initial evaluation of the algorithm's resilience against side-channel attacks. Side-channel attacks exploit implementation-specific leakages such as timing, power consumption, or electromagnetic signals rather than weaknesses in the algorithm itself. Through timing-based simulations and analysis, we observed that the encryption and decryption processes in the proposed Hyper-Ladder-based scheme do not exhibit significant timing variances that can be exploited by attackers. This is due to the uniformity in block processing and keystream application, which makes it difficult to derive information through side-channel observations. Hence, our proposed encryption system demonstrates resistance to timing-based side-channel attacks. Further exploration of resistance against other forms of cryptanalysis, such as differential and linear cryptanalysis, will be considered in future work.

4. Conclusion

Based on our analysis of the encryption stage, the keystream constructed from Hyper-Ladder performed the best. This makes Hyper-Ladder the most efficient algorithm in terms of time and size complexity. AES shows a consistent and efficient increase in time and size as the plaintext increases, but is yet to surpass Hyper-Ladder's performance. Based on the brute force analysis, we can conclude that the proposed algorithm cannot be broken within 24 hours. We analyzed using a key with a length of 256 bits and with a very large number of combinations, even the fastest supercomputer cannot try all possible keys in a limited time. This shows that our proposed algorithm has high robustness against brute force attacks.

Future research should focus on testing in various computing environments, including resource-constrained devices such as IoT (Internet of Things) and mobile devices. Keystreams constructed from Hypergraph can be applied to real-world applications, such as digital payment systems. Comparing the performance and security of Hyper-Ladder Graph with other modern cryptographic algorithms such as ChaCha20 and XChaCha20 can provide a broader perspective on Hyper-Ladder Graph's position in the cryptographic landscape. It can also use other keystreams from graph theory, which have not been used before. So that it can produce novelty not only in mathematical theory, but also in applications that can be used in the real world.

Initial implementation and testing were conducted using Matlab Online, a cloud-based environment provided by MathWorks. While this confirms the feasibility of executing the algorithm in a cloud setup, further exploration in distributed and scalable cloud architectures (e.g., multi-instance or container-based systems) remains part of future work. The independent processing of blocks and modular keystream construction in our proposed method offers strong potential for distributed system deployment, particularly in real-world scenarios such as secure cloud communications and IoT applications.

The proposed encryption method has promising applicability in real-world scenarios. For instance, it can be integrated into digital payment systems where lightweight, secure encryption is crucial for fast transaction processing with minimal storage overhead. Moreover, the stream-based encryption mechanism aligns well with secure communication protocols, especially in mobile messaging or IoT-based communication, where data is transmitted continuously in small blocks. The ability of the algorithm to generate adaptive and hard-to-predict keystreams enhances its suitability for dynamic and real-time applications requiring both efficiency and security.

For future research, we plan to explore several specific directions to further enhance the proposed encryption scheme. First, we aim to optimize the performance of the algorithm by implementing it in lower-level programming environments such as C++ or Rust, and by exploring hardware acceleration using GPUs or FPGAs. Second, a more comprehensive security analysis will be conducted, including formal proofs and empirical tests against differential, linear, and side-channel attacks. Finally, we intend to develop and test real-world prototypes of this cryptosystem in various application domains, such as secure messaging apps, lightweight payment authentication, and IoT-based communication. These efforts are expected to address current limitations and bring the proposed system closer to practical deployment.

In addition to the Hyper-Ladder Graph utilized in this study, future work may explore alternative keystream constructions derived from other graph-theoretic structures that have not yet been applied in cryptography. Examples include hierarchical graphs, multigraphs with antimagic properties, or dynamic graph compositions. Investigating these sources could enhance the novelty and efficiency of graph-based cryptosystems and lead to new cryptographic mechanisms suitable for specific application domains.

Acknowledgement

We gratefully acknowledge from the support of PUI-PT Combinatoric and Graph, CGANT-Universitas Jember, and the G20 Indonesia-India Research Collaboration Grant for their support. We also extend our gratitude to LP2M-Universitas Jember and DRTPM for their research support in 2025. This collaboration and funding have been instrumental in completing our work.

REFERENCES

1. Al Badawi, A., Hoang, L., Mun, C. F., Laine, K., & Aung, K. M. M. (2020). Privft: Private and fast text classification with homomorphic encryption. *IEEE Access*, 8, 226544-226556.
2. Alemami, Y., Mohamed, M. A., & Atiewi, S. (2023). Advanced approach for encryption using advanced encryption standard with chaotic map. *Int. J. Electr. Comput. Eng*, 13(2), 1708.
3. Baagyere, E. Y., Agbedemnab, P. A. N., Qin, Z., Daabo, M. I., & Qin, Z. (2020). A multi-layered data encryption and decryption scheme based on genetic algorithm and residual numbers. *IEEE Access*, 8, 100438-100447.
4. Maryati, T. K., Agustin, I. H., Nisviasari, R., Maylisa, I. N., & Kurniawati, E. Y. (2022). Research Based Learning-STEM Learning Activities: Developing a Secure CryptoKey by Using Rainbow Antimagic Coloring of Graphto Improve Students Combinatorial Thinking Skills.

5. Prihandoko, A. C., Auliya, Y. A., & Slamin, S. (2020). Randomness of encryption keys generated by super H-antimagic total labeling. *Indonesian Journal of Combinatorics*, 4(1), 21-26.
6. Agustin, I. H., Dafik, D., Nisviasari, R., Baihaki, R. I., Kurniawati, E. Y., Kartini, S., Sunder, R. & Nagaraja, V. (2024). On Rainbow Vertex Antimagic Coloring and Its Application to the Encryption Keystream Construction. Appl. Math, 18(4), 783-794.
7. Purcell, C., Ryan, J., Ryjáček, Z., & Skyvová, M. (2022). On exclusive sum labellings of hypergraphs. *Graphs and Combinatorics*, 38(2), 46.
8. Sun, X., Cheng, H., Liu, B., Li, J., Chen, H., Xu, G., & Yin, H. (2023). Self-supervised hypergraph representation learning for sociological analysis. *IEEE Transactions on Knowledge and Data Engineering*.
9. Cruz, F. R., Godoy, T. G., & Teixeira, S. R. 2023. On the antimagic labeling of certain hypergraphs. *Journal of Discrete Mathematics*, 36(4), 457-472. doi:10.1007/s00373-023-02489-4
10. Amburg, I., Veldt, N., & Benson, A. (2020, April). Clustering in graphs and hypergraphs with categorical edge labels. In *Proceedings of The Web Conference 2020* (pp. 706-717).
11. Venkatraman, S., Rajaram, G., & Krithivasan, K. (2020). Unimodular hypergraph for DNA sequencing: A polynomial time algorithm. *Proceedings of the National Academy of Sciences, India Section A: Physical Sciences*, 90(1), 49-56.
12. Dafik, Jannah, E. S. W., Agustin, I. H., Venkatraman, S., Mursyidah, I. L., Alfarisi, R., & Prihandini, R. M. (2024, June). On (a, d)-hyperedge Antimagic Labeling of Certain Classes of Hypergraphs: A New Notion. In *2nd International Conference on Neural Networks and Machine Learning 2023 (ICNNML 2023)* (pp. 173-183). Atlantis Press.
13. Agustin, I. H., Dafik, Baihaki, R. I., Marsidi, & Santoso, K. A. (2024). Irregular Reflexive Labeling and Elementary Row Operations for Enhanced Biometric Image Encryption. *Journal of Computer Science*, 20(12), Page 1766-1777.
14. Ajagbe, S. A., Adeniji, O. D., Olayiwola, A. A., & Abiona, S. F. (2024). Advanced Encryption Standard (AES)-Based Text Encryption for Near Field Communication (NFC) Using Huffman Compression. *SN Computer Science*, *5*(1), 156.
15. Nisviasari, R., Agustin, I. H., Kurniawati, E. Y., Maylisa, I. N., & Septory, B. J. (2022). Improving the robustness of the affine cipher by using a rainbow antimagic coloring. In Journal of Physics: Conference Series (Vol. 2157, No. 1, p. 012017). IOP Publishing.
16. Pavani, K., & Sriramya, P. (2024, May). Reduction of complexity in asymmetric cryptography using RSA, RSA-CRT and novel N-prime RSA with different keys. In *AIP Conference Proceedings* (Vol. 2853, No. 1). AIP Publishing.
17. Dafik, Nisviasari, R., Maryati, T. K., Agustin, I. H., & Venkatachalam, M. (2021). On local super antimagic total face coloring and the application in developing a cipher block chaining key. *Journal of Discrete Mathematical Sciences and Cryptography*, 24(4), 1101-1111.
18. Prihandoko, A. C., Dafik, D., & Agustin, I. H. (2019). Implementation of super H-antimagic total graph on establishing stream cipher. *Indonesian Journal of Combinatorics*, 3(1), 14-23.
19. Maryati, T. K., Atiqoh, K. S. N., Nisviasari, R., & Agustin, I. H. (2020). The Construction of Block Cipher Encryption Key by Using a Local Super Antimagic Total Face Coloring. *Advance in Mathematics: Science Journal*, 9(3), 1349-1362.
20. Santoso, K. A., Mursyidah, I. L., Agustin, I. H., Dafik, D., Venkatraman, S., & M. Venkatachalam. (2025). A Robust Algorithm for Asymmetric Cryptography Using Rainbow Vertex Antimagic Coloring. *Statistics, Optimization & Information Computing*.
21. Dafik, D., Firdausiyah, I., Adawiyah, R., Agustin, I. H., Mursyidah, I. L., & Marsidi, M. (2025). Analysis of Rainbow Vertex Antimagic Coloring and its Application to Cryptographic Secret Sharing with Affine Cipher Technique. *JTAM (Jurnal Teori dan Aplikasi Matematika)*, 9(1), 314-330.
22. Alfarisi, R., Prihandini, R. M., Adawiyah, R., Albirri, E. R., & Agustin, I. H. (2019, August). Graceful chromatic number of unicyclic graphs. In *Journal of Physics: Conference Series* (Vol. 1306, No. 1, p. 012039). IOP Publishing.
23. Dafik, Agustin, I.H., Surahmat, Alfarisi, R., & Sy, S. (2017). ON NON-ISOLATED RESOLVING NUMBER OF SPECIAL GRAPHS AND THEIR OPERATIONS. *Far East Journal of Mathematical Sciences*, 102, 2473-2492.
24. Septory, B.J., Utoyo, M.I., Sulistiyono, B. & Agustin, I.H. (2021). On rainbow antimagic coloring of special graphs. In *Journal of Physics: Conference Series* (Vol. 1836, No. 1, p. 012016). IOP Publishing.
25. Agustin, I.H., Hasan, M., Adawiyah, R., Alfarisi, R. & Wardani, D.A.R. (2018). On the Locating Edge Domination Number of Comb Product of Graphs. *Journal of physics: conference series* (Vol. 1022, No. 1, p. 012003). IOP Publishing.
26. Gembong, A.W. & Agustin, I.H. (2017). Bound of distance domination number of graph and edge comb product graph. *Journal of Physics: Conference Series* (Vol. 855, No. 1, p. 012014). IOP Publishing.
27. Singh, A. P. (2024). Safeguarding Authenticity in the Digital Realm: A Holistic Approach Integrating Content Provenance, Secure Watermarking, and Transparent Labeling to Combat Deepfakes. *International Journal for Multidisciplinary Research*, 6(3).
28. Sharma, S., Zou, J. J., Fang, G., Shukla, P., & Cai, W. (2024). A review of image watermarking for identity protection and verification. *Multimedia Tools and Applications*, 83(11), 31829-31891.