Global and Stochastic Analysis **DOI: 10.64837/GSA.13.3.6**
Vol. 13 No. 3 (2026)

Received: 13th April 2026 *Revised: 11th June 2026* *Accepted: 19th June 2026*

GRAPH LABELING BASED CIPHER GRAPH CONSTRUCTION FOR RSA ENCRYPTION

NAGARJUN PRABHU[1], GIRIJA K P[2,*], RAKSHA[3], VASANTH KUMAR S[4], AND SHARAN HEGDE[5]

Abstract. This paper presents a hybrid cryptographic framework that combines positional indexing, RSA public-key encryption, and graph theoretic methods based on graph labeling. In the proposed approach, plaintext characters are first converted into numeric indices using a fixed 69-symbol lookup table and randomized by a per-message offset. The resulting sequence is encrypted using RSA, and each ciphertext value is then augmented with an auxiliary modulus that is coprime to the ciphertext and its corresponding modular inverse. These triplets are embedded as payload triangles in a three-layer cipher graph, while additional random edges and random edge labels inject structural noise and obscure the payload pattern. Graph structures are leveraged to illustrate the encryption workflow and to provide a visually interpretable ciphertext representation. The method is assessed in terms of reversibility, statistical security indicators (including byte histograms and entropy), and computational efficiency over varying message lengths. By uniting number theory, graph labeling, and asymmetric encryption, this work introduces a multi-layered strategy for secure and structurally enriched cryptographic encoding.

* corresponding Author.

1. Introduction

Cryptography forms the foundation of secure digital communication by providing confidentiality, integrity, and authentication in the presence of adversaries [3, 4]. Among public-key systems, the RSA cryptosystem—introduced by Rivest, Shamir, and Adleman in 1977—remains one of the most widely deployed schemes due to its solid number-theoretic foundations and reliance on the difficulty of factoring large composite integers [5, 7]. As computational power and cryptanalytic techniques continue to advance, however, classical RSA deployments increasingly require additional structural layers and hybrid mechanisms to maintain strong security margins against modern attacks, including side-channel and chosen-ciphertext scenarios [7].

Classical block ciphers and public-key schemes are typically analyzed under well-defined adversarial models, including chosen-plaintext and chosen-ciphertext attacks, side-channel leakage, and, more recently, post-quantum threat scenarios [4]. For RSA in particular, deterministic textbook encryption is known to be insecure under adaptive chosen-ciphertext attacks and must be combined with

2000 *Mathematics Subject Classification.* :94A60, 05C78, 11T71, 68P25.
Key words and phrases. Graph-based cryptography, Graph labeling, RSA encryption.

randomized padding schemes such as OAEP to achieve semantic security [8, 9]. Practical implementations must also account for timing, fault-injection, and power-analysis attacks, which exploit physical side channels rather than purely mathematical weaknesses [7]. These considerations motivate the exploration of hybrid designs that retain well-understood hardness assumptions while introducing additional structural complexity in the ciphertext representation and transmission format.

One complementary direction in this evolution involves the integration of cryptography with combinatorial structures. By embedding cryptographic operations into richer mathematical objects, such as graphs, it becomes possible to combine algebraic security (e.g., factoring hardness) with structural obfuscation (e.g., complex connectivity patterns) in a single framework [25, 26]. This is particularly attractive for constrained environments and emerging applications where both efficiency and interpretability are important, such as Internet-of-Things (IoT) devices, cyber-physical systems, and visual cryptography platforms.

Graph theory, which studies structures formed by vertices and edges, offers rich combinatorial objects and transformations that can be exploited to increase the algebraic and structural complexity of encryption schemes. Graph-based cryptography investigates constructions in which messages, keys, or intermediate states are represented as graphs or graph labelings to achieve enhanced confusion, diffusion, and resistance to structural analysis. Such approaches have been used to model key exchange, generate pseudorandom sequences, and design resilient encryption networks that incorporate both algebraic and structural hardness assumptions [21, 23].

Within this context, graph labeling—the assignment of integers or symbols to a graph's vertices and/or edges under specified constraints—has emerged as a flexible tool for encoding information and generating cryptographic material [11]. A wide variety of labeling techniques, including graceful, edge-graceful, harmonious, magic, and geometric-mean labelings, have been explored in applications ranging from data encoding to key generation and watermarking [12, 22, 14]. In a classical graceful labeling, a graph with q edges is graceful if its vertices can be assigned distinct labels from $\{0, 1, \ldots, q\}$ such that the absolute differences of endpoint labels on every edge produce all integers from 1 to q without repetition [12, 13]. These graceful-type labelings produce highly regular but combinatorially rich configurations, making them attractive for constructing deterministic yet complex mappings that can serve as indices, key schedules, or graph-encoded ciphertext structures [17, 18, 16].

For instance, prime graceful labeling has been used to design encoding schemes where primes and their combinatorial differences generate key streams with desirable distribution properties [17]. Odd-hexagonal graceful and geometric-mean labelings have been employed to construct message-dependent labeled paths and cycles whose structure varies nonlinearly with the plaintext, thereby strengthening resistance to simple pattern-based attacks [18, 16, 23]. More broadly, labeled paths, cycles, and star graphs have been proposed as vehicles for encoding messages in a visually interpretable yet structurally nontrivial form [23, 24].

A growing body of work demonstrates how graph labeling ideas can be embedded into concrete cryptographic schemes. Bekkaoui et al. proposed a Hamiltonian-circuit based encryption model, leveraging adjacency matrices to map plaintext into graph structures and applying combinatorial transformations to achieve encryption. Sugeng et al. and related authors employed graceful and antimagic labelings to generate high-complexity key material for symmetric encryption, exploiting the combinatorial diversity of labeled graphs to expand the key space [14]. Other contributions have used prime-graceful, odd-hexagonal graceful, and geometric-mean labelings to encode messages via labeled paths and subgraphs, highlighting the versatility of labeling-based approaches in secure communication [17, 18, 16, 23].

In parallel, fuzzy and intuitionistic fuzzy graph models have been combined with symmetric or hybrid cryptosystems to create multi-layered frameworks in which uncertainty in edge weights and membership values contributes an additional obfuscation layer over conventional number-theoretic security [19, 20]. Recent work has also explored graph-based encryption frameworks using complete graphs and related structured families as generic carriers for graph-based operations, suggesting that graph-theoretic constructs can act as a unifying layer above classical cryptographic primitives. Furthermore, stream-key generation schemes based on graph labeling have been proposed to strengthen Vigenère-type encryption by deriving keystreams from graph label sequences rather than simple periodic patterns [21].

Despite these advances, most existing graph-based schemes either restrict graph theory to key generation or key scheduling, or combine it with symmetric ciphers, without directly embedding public-key ciphertext into a labeled graph in a way that preserves the hardness assumptions of RSA [**?**, 25]. Several RSA–graph hybrids treat graphs as auxiliary tools (for key expansion or mask generation) rather than integral carriers of ciphertext, and often do not fully exploit graph structure to obscure ciphertext relationships. Moreover, many labeling-based constructions are deterministic once parameters are fixed; without additional randomization or modular transformations, they cannot independently guarantee semantic security and may leak structural patterns that an informed adversary could exploit.

The gap identified above can be summarized as follows: there is limited work that integrates graph-labeling ideas with RSA in such a way that the RSA-encrypted numeric data themselves become vertices or edges of a cipher graph, augmented with auxiliary algebraic parameters to increase structural uncertainty. Existing graph-based schemes tend to separate the algebraic and structural components, using graphs either before encryption (for key or message preprocessing) or after decryption (for visualization), rather than as a primary ciphertext carrier. Consequently, the ciphertext often remains a simple numeric sequence, and the graph-theoretic layer does not fully contribute to obfuscating the relationships among encrypted symbols.

This paper is motivated by the goal of designing a scheme in which RSA provides the core number-theoretic security, while a graph structure—built using labeling-inspired rules and controlled randomness—acts as the actual transmitted ciphertext object. The intent is to combine the classical hardness guarantees of RSA

with a graph-based representation that supports Shannon's principles of confusion and diffusion by masking direct correspondences between plaintext positions and transmitted values [10]. At the same time, the scheme aims to remain lightweight and visually interpretable, so that the cipher graph can be inspected, analyzed, or even used pedagogically to illustrate the encryption process.

2. Proposed Method

This section describes the architecture and algorithms of the proposed hybrid encryption scheme, which integrates classical RSA encryption with graph-theoretic transformations to create a cipher that is both structurally complex and visually interpretable. The plaintext is first mapped to numeric positions in a 69-symbol alphabet, randomized via a per-message offset, and encrypted with RSA. The RSA ciphertexts are then embedded into a three-layer cipher graph using auxiliary coprime moduli and modular inverses, and the resulting graph is transmitted as the ciphertext object.

2.1. Notation and setup. Let $M = c_1 c_2 \dots c_t$ denote the plaintext message over a fixed alphabet Σ of size 69. A lookup table

$$T : \Sigma \to \{1, 2, \dots, 69\}$$

maps each character $c \in \Sigma$ to a unique integer position $\mathrm{pos}(c)$, and T^{-1} denotes its inverse. The concrete 69-symbol mapping used in this work (uppercase letters, digits, punctuation, and space) is shown in Figure **??**. RSA public parameters are (e, n) and the private key is d, with $n = pq$ for distinct primes p, q and $ed \equiv 1 \pmod{\varphi(n)}$, where $\varphi(n) = (p-1)(q-1)$. All arithmetic of the form $x \bmod m$ is taken in the usual sense with representatives in $\{0, 1, \dots, m-1\}$ unless otherwise stated.

TABLE 1. Alphabet and index mapping used in the proposed scheme (69-symbol lookup table).

Q	W	E	R	T	Y	U	I	O	P	A	S	D	F	G
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

H	J	K	L	Z	X	C	V	B	N	M	SPACE	~
16	17	18	19	20	21	22	23	24	25	26	27	28

‘	!	1	@	2	#	3	$	4	%	5	^	6	&	7
29	30	31	32	33	34	35	36	37	38	39	40	41	42	43

*	8	(	9	)	10	_	-	+	=	{	[	}	]	—
44	45	46	47	48	49	50	51	52	53	54	55	56	57	58

\	:	;	”	’	<	,	>	.	?	/
59	60	61	62	63	64	65	66	67	68	69

2.2. Character mapping and randomized vertex labels. For each $i \in \{1, \ldots, t\}$, the position of character c_i is

$$\mathrm{pos}_i = T(c_i) \in \{1, \ldots, 69\}.$$

To prevent deterministic leakage of positional structure, a random offset

$$r \in \{0, 1, \ldots, 9\}$$

is sampled using a cryptographically secure random number generator. The randomized vertex label V_i for position pos_i is computed as

$$V_i \equiv (141 - \mathrm{pos}_i - r) \pmod{69}.$$

If $V_i = 0$, it is remapped to 69 so that all labels lie in $\{1, \ldots, 69\}$. The sequence

$$K_1 = \{V_1, V_2, \ldots, V_t\}$$

forms the randomized vertex-label representation of the plaintext. The offset r is later encrypted as $r_{\mathrm{enc}} = r^{\,e} \bmod n$ and sent to the receiver.

2.3. RSA encryption of vertex labels. RSA key generation proceeds as follows. Two random, distinct primes p and q are generated, and

$$n = pq, \qquad \varphi = (p-1)(q-1).$$

A public exponent e is chosen such that

$$\gcd(e, \varphi) = 1,$$

and the private exponent d is computed as the modular inverse $d \equiv e^{-1} \pmod{\varphi}$.

Each randomized vertex label V_i is then encrypted under RSA as

$$C_i = V_i^{\,e} \bmod n.$$

The offset is protected similarly by computing

$$r_{\mathrm{enc}} = r^{\,e} \bmod n.$$

The multiset $\{C_1, \ldots, C_t\}$ is the classical RSA ciphertext sequence corresponding to the randomized vertex labels.

2.4. Auxiliary pair generation. To prepare for graph embedding and to support a consistency check during decryption, each ciphertext C_i is associated with an auxiliary modulus and its modular inverse.

For each $i \in \{1, \ldots, t\}$, choose the smallest integer B_i satisfying

$$B_i \geq 70, \qquad \gcd(C_i, B_i) = 1.$$

Because $\gcd(C_i, B_i) = 1$, there exists a unique modular inverse of C_i modulo B_i. Define

$$A_i \equiv C_i^{-1} \pmod{B_i},$$

so that the relation

$$(A_i \cdot C_i) \bmod B_i = 1$$

holds. The pair (B_i, A_i) will become part of the structural description of the cipher graph.

2.5. Cipher graph construction. For each index i, define the third-layer value

$$S_i = C_i + B_i + A_i.$$

The cipher graph G is a labeled graph whose vertex set is partitioned into three disjoint layers:

$$L_C = \{C_1, \ldots, C_t\}, \qquad L_B = \{B_1, \ldots, B_t\},$$
$$L_S = \{S_1, \ldots, S_t\}.$$

The core encoding of each triple (C_i, B_i, A_i) is realized by a cross-layer triangular motif. For each $i \in \{1, \ldots, t\}$, the following edges are added:

$$(C_i, B_i), \quad (B_i, S_i), \quad (C_i, S_i),$$

forming a payload triangle that spans L_C, L_B, and L_S.

To enhance ambiguity for an adversary who only has access to the graph, the scheme introduces additional random edges exclusively among vertices in L_C. For a chosen probability parameter $p_{cc} \in (0, 1)$, random edges are added independently between pairs of vertices in L_C with probability p_{cc} (or according to a specified random-graph model). Finally, every edge in G (both payload and random) is assigned a random edge label drawn from a fixed label set. As a result, payload edges span multiple layers, while noise edges are confined to L_C, preserving the ability of the legitimate receiver to recover the encoded triples while complicating structural inference for an attacker.

2.6. Cipher graph decoding (receiver side). The receiver is assumed to possess the RSA private key exponent d corresponding to the public key (e, n). Given a received cipher graph G, the receiver first identifies candidate triples by locating cross-layer triangles (C, B, S) with

$$C \in L_C, \qquad B \in L_B, \qquad S \in L_S.$$

For each such triangle, the receiver recovers

$$A = S - C - B,$$

and validates it using the modular-inverse relation

$$(A \cdot C) \bmod B = 1.$$

Only triangles that satisfy this check are retained as valid triples (C, B, A). Random C–C edges and incidental cross-layer patterns with incorrect arithmetic relations are rejected by this test, so the decoded set of C values corresponds to the intended ciphertext sequence (up to the graph's inherent ordering, which the receiver can maintain or reconstruct).

2.7. Decryption. Once the validated ciphertext values C_i have been recovered in order, decryption proceeds by inverting the RSA and vertex-label transformations.

Using the RSA private key d, the receiver recovers the randomized vertex labels as

$$V_i = C_i^d \bmod n.$$

The random offset is recovered from the encrypted value as

$$r = r_{\text{enc}}^d \bmod n, \qquad r \leftarrow r \bmod 10.$$

The original position values are then obtained via

$$\mathrm{pos}_i \equiv (141 - r - V_i) \pmod{69},$$

with the convention that if $\mathrm{pos}_i = 0$ then pos_i is remapped to 69. Finally, the plaintext characters are reconstructed as

$$c_i = T^{-1}(\mathrm{pos}_i), \qquad i = 1, \ldots, t,$$

and concatenated to form M.

2.8. Algorithmic summary. For implementation clarity, the encryption and decryption procedures are summarized in Algorithms 1 and 2.

Algorithm 1 Encryption and Cipher Graph Generation

Require: Plaintext $M = c_1 c_2 \cdots c_t$, lookup table $T : \Sigma \to \{1, \ldots, 69\}$, graph parameter p_{cc}
Ensure: Cipher graph G, RSA public key (e, n), encrypted offset r_{enc}
for $i \leftarrow 1$ **to** t **do**
  $\mathrm{pos}_i \leftarrow T(c_i)$
end for
Generate random offset $r \in \{0, \ldots, 9\}$
for $i \leftarrow 1$ **to** t **do**
  $V_i \leftarrow (141 - \mathrm{pos}_i - r) \bmod 69$
  if $V_i = 0$ **then**
    $V_i \leftarrow 69$
  end if
end for
Generate RSA keys: choose p, q, set $n \leftarrow pq$, $\varphi \leftarrow (p-1)(q-1)$
Select e with $\gcd(e, \varphi) = 1$, compute $d \equiv e^{-1} \pmod{\varphi}$
for $i \leftarrow 1$ **to** t **do**
  $C_i \leftarrow {V_i}^e \bmod n$
end for
$r_{\mathrm{enc}} \leftarrow r^{\,e} \bmod n$
for $i \leftarrow 1$ **to** t **do**
  Choose smallest $B_i \geq 70$ such that $\gcd(C_i, B_i) = 1$
  $A_i \leftarrow C_i^{-1} \bmod B_i$
  $S_i \leftarrow C_i + B_i + A_i$
end for
Construct layers $L_C = \{C_i\}$, $L_B = \{B_i\}$, $L_S = \{S_i\}$
for $i \leftarrow 1$ **to** t **do**
  Add payload edges (C_i, B_i), (B_i, S_i), (C_i, S_i)
end for
Add random C–C edges among vertices in L_C with probability p_{cc}
Assign random labels to all edges
return $G, (e, n), r_{\mathrm{enc}}$

Algorithm 2 Decryption from Cipher Graph

Require: Cipher graph G, RSA public key (e, n), private key d, encrypted offset r_{enc}, inverse table T^{-1}
Ensure: Recovered plaintext M
Extract candidate cross-layer triangles (C, B, S) with $C \in L_C$, $B \in L_B$, $S \in L_S$
for all candidate triangles (C, B, S) **do**
    $A \leftarrow S - C - B$
    if $(A \cdot C) \bmod B \neq 1$ **then**
        Discard candidate
    end if
end for
Recover $r \leftarrow r_{\text{enc}}^{d} \bmod n$; set $r \leftarrow r \bmod 10$
for all validated ciphertext values C in decoded order **do**
    $V \leftarrow C^{\,d} \bmod n$
    if $V = 0$ **then**
        $V \leftarrow 69$
    end if
    $\text{pos} \leftarrow (141 - r - V) \bmod 69$
    if $\text{pos} = 0$ **then**
        $\text{pos} \leftarrow 69$
    end if
    Output character $T^{-1}(\text{pos})$
end for
Concatenate output characters to form M

2.9. Detailed illustrative example: "MATHEMATIC". This subsection presents the complete numeric encryption and decryption steps for the plaintext

$$M = \text{“MATHEMATIC”},$$

using the 69-symbol lookup table of table 1, offset $r = 4$, and toy RSA parameters

$$p = 17, \quad q = 23, \quad n = pq = 391,$$

$$\varphi = (p-1)(q-1) = 352, \quad e = 3, \quad d = 235.$$

These parameters are for illustration only; practical deployments require much larger moduli.
Step 1: Character positions pos_i. From the index table, the characters and their positions are:

i	1	2	3	4	5	6	7	8	9	10
c_i	M	A	T	H	E	M	A	T	I	C
pos_i	25	12	6	17	5	25	12	6	9	23

Step 2: Randomized vertex labels V_i. With offset $r = 4$, compute

$$V_i \equiv (141 - \text{pos}_i - r) \pmod{69},$$

TABLE 2. Full worked example for $M =$ "MATHEMATIC" with $r = 4$ and $p = 17, q = 23, e = 3$.

i	c_i	pos_i	V_i	C_i	B_i	A_i	$S_i = C_i + B_i + A_i$
1	M	25	43	134	71	62	267
2	A	12	56	57	70	43	170
3	T	6	62	209	70	69	348
4	H	17	51	102	71	55	228
5	E	5	63	198	71	52	321
6	M	25	43	134	71	62	267
7	A	12	56	57	70	43	170
8	T	6	62	209	70	69	348
9	I	9	59	104	71	28	203
10	C	23	45	22	71	42	135

remapping $V_i = 0$ to 69. This yields:

i	1	2	3	4	5	6	7	8	9	10
pos_i	25	12	6	17	5	25	12	6	9	23
V_i	43	56	62	51	63	43	56	62	59	45

Step 3: RSA encryption C_i and encrypted offset r_{enc}. Using $n = 391$, $e = 3$,

$$C_i = {V_i}^3 \bmod 391, \qquad r_{\text{enc}} = r^3 \bmod 391 = 64.$$

The ciphertext values are:

i	1	2	3	4	5	6	7	8	9	10
V_i	43	56	62	51	63	43	56	62	59	45
C_i	134	57	209	102	198	134	57	209	104	22

Step 4: Auxiliary moduli B_i, inverses A_i, and third-layer values S_i. For each C_i, choose the smallest integer $B_i \geq 70$ with $\gcd(C_i, B_i) = 1$, then

$$A_i \equiv C_i^{-1} \pmod{B_i}, \qquad S_i = C_i + B_i + A_i.$$

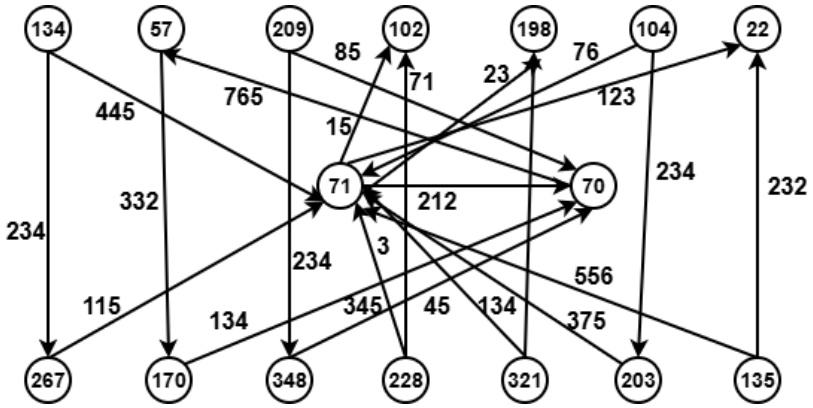

FIGURE 1. Cipher Graph with C_i, B_i and S_i.

The cipher graph G has three vertex layers

$$L_C = \{C_1, \ldots, C_{16}\}, \quad L_B = \{B_1, \ldots, B_{16}\},$$
$$L_S = \{S_1, \ldots, S_{16}\}.$$

For each i, payload edges (C_i, B_i), (B_i, S_i), and (C_i, S_i) are inserted, and additional random edges with random labels are added among L_C according to the chosen parameter p_{cc}.

Step 5 : recovering C_i from the cipher graph. Given G, the receiver enumerates cross-layer triangles (C, B, S) with $C \in L_C$, $B \in L_B$, $S \in L_S$, and for each candidate computes

$$A = S - C - B.$$

A triangle is accepted as valid if and only if

$$(A \cdot C) \bmod B = 1.$$

This test filters out triangles formed by random edges and recovers exactly the 16 valid triples (C_i, B_i, A_i) in Table 2.

Step 6: RSA decryption to obtain V_i. Using the private exponent $d = 235$, the receiver computes

$$V_i = C_i^d \bmod 391, \qquad i = 1, \ldots, 16,$$

which reproduces the sequence of randomized vertex labels

$$V_i = 43, 56, 62, 51, 63, 43, 56, 62, 59, 45$$

Step 7: Recovering r and inverting the vertex-label map. The random offset is recovered from

$$r = r_{\text{enc}}^d \bmod 391, \qquad r \leftarrow r \bmod 10,$$

which in this example yields $r = 4$. The original positions are then obtained as

$$\text{pos}_i \equiv (141 - r - V_i) \pmod{69},$$

with $\text{pos}_i = 0$ remapped to 69. Applying this to all V_i recovers

$$\text{pos}_i = 25, 12, 6, 17, 5, 25, 12, 6, 9, 23$$

which coincides with the initial positions from the lookup table.
Step 8: Mapping positions back to characters. Finally, the plaintext characters are recovered as

$$c_i = T^{-1}(\mathrm{pos}_i), \qquad i = 1, \ldots, 16,$$

yielding

$$M = \text{“MATHEMATIC”}.$$

This completes the end-to-end encryption and decryption demonstration for the proposed scheme.

3. Security Analysis and Experimental Evaluation

3.1. Security of the RSA core. The confidentiality of the ciphertext sequence $\{C_i\}$ and the infeasibility of recovering $\{V_i\}$ without the private exponent d are inherited directly from RSA. For cryptographically strong parameters, the best known generic attacks reduce to factoring the modulus $n = pq$ or solving the RSA inversion problem. The graph layer in the proposed scheme does not weaken these assumptions: even if an adversary could perfectly identify all true C_i values from the graph, recovering the underlying V_i (and hence the plaintext) would still require breaking RSA. In practical deployments, RSA should additionally be combined with standardized padding to achieve semantic security and robustness against chosen-ciphertext and related attacks.

3.2. Security role of the cipher graph. The cipher graph acts as a structured carrier for the RSA ciphertext values rather than as a replacement for the RSA core. Each symbol of the plaintext is encoded as a cross-layer triangle (C_i, B_i, S_i) that satisfies the modular-inverse constraint $(A_i C_i) \bmod B_i = 1$, while random C–C edges and random edge labels are injected to obscure the set of valid payload triangles. An adversary who observes only the graph must therefore solve a structural inference problem: distinguish consistent triples from noise, recover the ordered sequence of C_i, and then still invert RSA. This graph layer increases confusion and structural ambiguity, especially against attackers that attempt to exploit format or position information, but the primary cryptographic hardness remains that of RSA.

3.3. Randomness and entropy evaluation. To empirically assess diffusion and apparent randomness, byte-level histograms of plaintext and ciphertext were generated for representative messages (Figure 2), and Shannon entropy of ciphertext bytes was measured as a function of message length (Figure 3). The histograms show that the ciphertext byte distribution is substantially more uniform than that of the corresponding plaintext, indicating that simple frequency-based attacks are not directly applicable. The entropy curves approach higher values as the message length grows, suggesting that, at the byte level, ciphertext statistics do not trivially preserve the structure of the underlying messages.

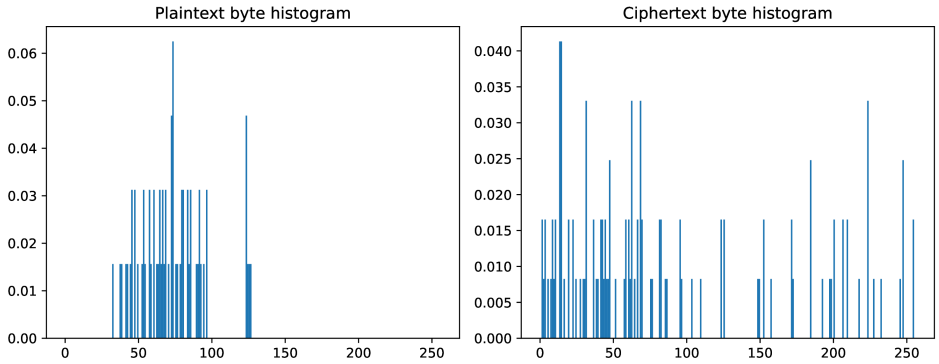

FIGURE 2. Plaintext vs. ciphertext byte histograms for a sample message.

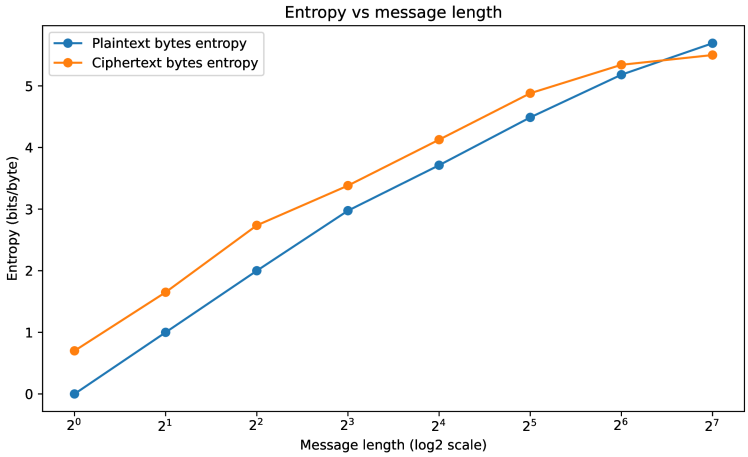

FIGURE 3. Entropy vs. message length (plaintext bytes vs. ciphertext bytes).

The entropy of randomly assigned edge labels as a function of message length is shown in Figure 4.

3.4. Graph-structure randomness. Beyond byte-level statistics, the structural randomness of the cipher graph was examined through the degree distribution of vertices in the ciphertext layer L_C. Random C–C edges are intended to mask the deterministic payload triangles, so a concentrated or highly regular degree profile would indicate insufficient obfuscation. Observed degree distributions are spread out over a range of values rather than clustering around a fixed degree, which is consistent with the intended random-graph behaviour and suggests that the added edges successfully perturb the local structure seen by an adversary. The degree distribution of ciphertext-layer (L_C) nodes for a representative run is shown in Figure 5.

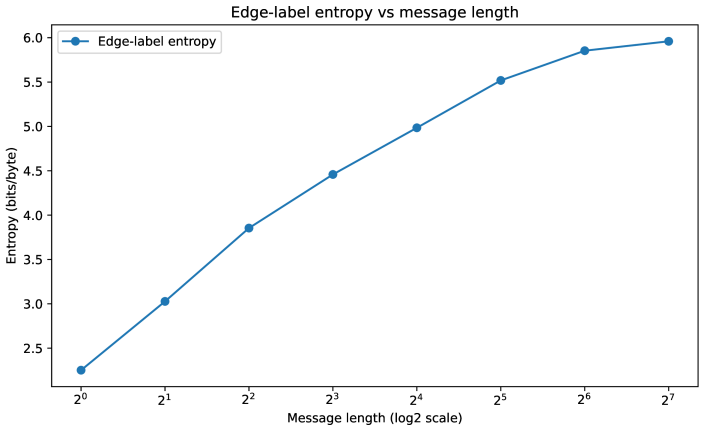

FIGURE 4. Edge-label entropy vs. message length.

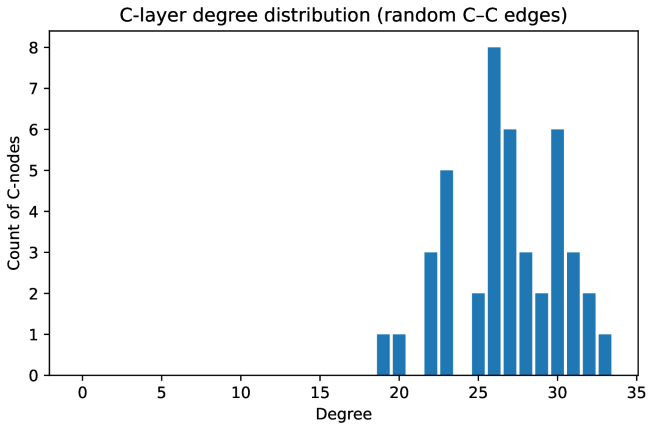

FIGURE 5. Degree distribution of ciphertext (C-layer) nodes for a sample run.

3.5. Performance and scalability. The runtime cost of the scheme was measured for increasing message lengths, decomposed into RSA encryption, cipher-graph construction, RSA decryption, and total execution time. As expected, the complexity grows with the number of characters, with cipher-graph construction dominating for longer messages due to auxiliary pair generation and random edge insertion. Nevertheless, the measured times remain in the millisecond range for the tested lengths, indicating that the additional graph layer can be implemented with modest overhead relative to the RSA operations.

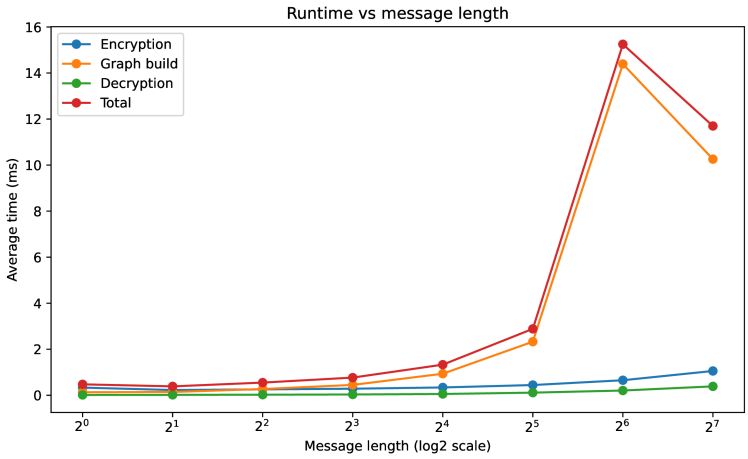

FIGURE 6. Runtime vs. message length (mean over repeated trials).

TABLE 3. Benchmark results: average execution time and entropy metrics vs. message length.

Message length (characters)	Encryption time (ms)	Cipher-graph construction time (ms)	Decryption time (ms)	Total time (ms)	Ciphertext byte entropy (bits/byte)	Edge-label byte entropy (bits/byte)
1	0.33139	0.13027	0.01284	0.47450	0.700000	2.251629
2	0.22733	0.14236	0.01655	0.38624	1.650978	3.027169
4	0.25557	0.27063	0.02393	0.55013	2.736526	3.853863
8	0.28428	0.44951	0.03335	0.76714	3.381876	4.458333
16	0.33887	0.93514	0.05516	1.32917	4.127258	4.984433
32	0.44378	2.32750	0.11356	2.88484	4.880900	5.518441
64	0.65250	14.39256	0.20551	15.25057	5.342678	5.853635
128	1.05466	10.25905	0.38577	11.69948	5.500979	5.959534

4. Conclusion

This paper introduced a hybrid encryption framework that combines RSA with a graph-based ciphertext representation in the form of a three-layer cipher graph. RSA supplies the core computational hardness, while the graph layer embeds ciphertext-related values as payload triangles and adds structural obfuscation through randomized intra-layer edges and random edge labels. Experimental results based on byte histograms, entropy measurements, degree distributions, and runtime profiling confirm correct decryption and indicate that the additional graph layer achieves meaningful diffusion with acceptable computational overhead. Future work includes integrating standardized RSA padding, employing larger key sizes, and exploring keyed graph-embedding strategies to further strengthen resistance to structural and statistical inference attacks.

References

[1] Itô, K.: Stochastic integral, *Proc. Imp. Acad. Tokyo* **20** (1944) 519–524.
[2] Alexa Sharp, Distance coloring, *in: European Symposium on Algorithms, Springer, Lecture Notes in Computer Science*, **4698** (2007), 510–521.
[3] W. Stallings, *Cryptography and Network Security: Principles and Practice*, Pearson, 8th ed., 2020.
[4] J. Katz and Y. Lindell, *Introduction to Modern Cryptography, Chapman and Hall/CRC*, 3rd ed., 2021.

[5] R. L. Rivest, A. Shamir and L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, *Communications of the ACM*, **26**, (1983), 96–99, .
[6] L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner and D. Smith-Tone, *Report on Post-Quantum Cryptography*, NISTIR 8105, (2016).
[7] D. Boneh, Twenty Years of Attacks on the RSA Cryptosystem, *Notices of the AMS*, **46**, (1999) 203–213.
[8] M. Bellare and P. Rogaway, Optimal Asymmetric Encryption: How to Encrypt with RSA, *Advances in Cryptology – EUROCRYPT*, **950**, (1995), 92–111.
[9] D. Pointcheval, How to Encrypt Properly with RSA, *CryptoBytes*, **5**, (2002),10–19.
[10] C. E. Shannon, Communication Theory of Secrecy Systems, *Bell System Technical Journal*, **28**, (1949), 656–715.
[11] J. A. Gallian, A Dynamic Survey of Graph Labeling, *Electronic Journal of Combinatorics*, (2022).
[12] A. Rosa, On Certain Valuations of the Vertices of a Graph, *Theory of Graphs*, (1967), 349–355, .
[13] R. C. Entringer, D. E. Jackson and J. A. Snyder, Distance in Graphs, *Czechoslovak Mathematical Journal*, **33**, (1983), 159–169.
[14] K. A. Sugeng, Graph Labelings for Cryptographic Key Generation, *Far East Journal of Mathematical Sciences*, **99**, (2016) 923–938.
[15] N. Ali, N. Sadiqa, Q. Awan and J. I. Khan, Secure communication in the digital age: a new paradigm with graph-based encryption algorithms, *Frontiers in Computer Science*, **6**,1-10, (2024).
[16] P. Murugan and S. Nagarajan, Geometric Mean Labeling in Cryptography: A Graph-Based Approach to Enhanced Security and Performance,*Contemporary Mathematics*, **6**, (2025), 2698-2725.
[17] M. V. Nayana and K. R. Sobha, Encoding And Decoding Process Using Prime Graceful Labeling, *Journal of Propulsion Technology*, **44**, (2023), 2638–2644.
[18] S. Asha and V. Akshaya, Application of odd hexagonal graceful labeling in cryptography, *International Journal of Advanced Academic Studies*, **6**, (2024), 1–4.
[19] C. Ruby Sharmila and S. Meenakshi, A Fuzzy Graph Theory Approach to Symmetric Key Cryptography, *Journal of Propulsion Technology*, **45**, (2024), 467–477.
[20] A. Meenakshi and S. Dhanushiya, A multi layered encryption framework using intuitionistic fuzzy graphs, *Scientific Reports*, **21** (2025).
[21] A. C. Prihandoko, Dafik and I. H. Agustin, Stream-keys generation based on graph labeling, *International Journal of Electrical and Computer Engineering*, **12**,(2022),3960–3969.
[22] G. S. Bloom and S. W. Golomb, Applications of Numbered Undirected Graphs, *Proceedings of the IEEE*, **65**, (1977),562–570.
[23] D. K. Gurjar and A. Krishnaa, Labeled Paths in Cryptography, Nova Science Publishers, (2021).
[24] H. R. Medini, S. D'Souza, D. C. Nayak and P. G. Bhat, *Encoding and decoding of messages using graph labeling, Global and Stochastic Analysis*, **11**,(2024),1–13.
[25] P. L. K. Priyadarsini, Graph theory applications in cryptography and network security, *Journal of Mathematical Problems*, **6**, (2025), 233–241.
[26] R. Nandhini, V. Maheswari and V. Balaji, A Graph Theory Approach on Cryptography, *Communications in Mathematics*, **2**, (2018), 97–104.

[1,2,3]Nitte (Deemed to be University), NMAM Institute of Technology(NMAMIT), Department of Mathematics, Nitte, Karkala-574110, Karnataka, India.

[4]Department of Mathematics, Seshadripuram Institute of Technology, kadakola, Mysore, Karnataka-571311, India

NAGARJUN PRABHU[1], GIRIJA K P[2,*], RAKSHA[3], VASANTH KUMAR S[4], AND SHARAN HEGDE[5]

[5]Department of Mathematics, Shri Madhwa Vadiraja Institute of Technology and Management, Bantakal, udupi, Karnataka, India-574115

Email address: [1]nagarjunprabhu600@gmail.com, [2]girijakp16@gmail.com,[3]raksha.ak28@gmail.com,, [4]svkmaths.174@gmail.com,[5]hegdesharan3@gmail.com