ORIGINALE ARTICLE

A Novel Multi-Layer Encryption Framework Combining DNA Cryptography and Mahgoub Transform for Secure IoT Communications

| Swaira Maryam |

International Islamic University Islamabad | Mathematics, Islamabad | Pakistan |

| DOI: 10.5281/zenodo.14790500 | | Received January 20, 2025 | | Accepted January 30, 2025 | | Published February 03, 2025 | | ID Article | Swaira-Ref1-2-20ajiras240225 |

ABSTRACT

Introduction : La sécurité de l'information constitue un défi majeur dans l'environnement actuel de l'Internet des Objets (IoT), où les utilisateurs doivent protéger leurs données partagées et leur vie privée. **Objectif :** Cette recherche propose une nouvelle méthode de cryptage combinant la cryptographie ADN, la transformation de Mahgoub et les systèmes de congruence modulaire pour sécuriser les messages textuels. **Méthodes :** Le processus de cryptage repose sur la transformation de Mahgoub, qui utilise des principes standards de transformation intégrale pour convertir les données en clair dans un espace mathématique réactif. La cryptographie ADN renforce la sécurité en exploitant les propriétés biologiques des séquences ADN pour encoder les données. Le système génère des clés à travers la création de graines initiales, la manipulation de bits, la sélection de permutations et des opérations XOR. Le cryptage s'effectue en deux étapes pour accroître la robustesse et rendre la détection plus difficile. **Résultats :** Les performances de randomisation de l'algorithme ont été évaluées à l'aide de mesures d'entropie et d'effets avalanche, démontrant son efficacité et sa sécurité face aux variations des données d'entrée. Les tests confirment que l'algorithme génère des sorties hautement aléatoires malgré des changements minimes dans les entrées. **Conclusion :** La combinaison des couches de cryptage offre une protection efficace pour les dispositifs IoT tout en maintenant des standards de sécurité élevés dans divers scénarios d'utilisation.
***Mots**-clés : Décryptage, Entropie, Permutation, Génération de graines, XOR.*

1. INTRODUCTION

Cryptography, the science of secure communication, encompasses both theoretical foundations and practical implementations of data encryption and decryption protocols. While classical cryptographic methods rely primarily on mathematical transformations, modern approaches increasingly integrate concepts from multiple disciplines to enhance security and computational efficiency.

This research presents a novel cryptographic framework that synthesizes three distinct elements: the Mahgoub Transform [1], modular arithmetic, and DNA sequence-based encoding. The Mahgoub Transform, derived from Fourier analysis principles, offers unique advantages in signal processing and has demonstrated relationships with other integral transforms, including the Laplace, Elzaki [2], and Aboodh transforms [3]. Our approach leverages the transform's mathematical properties to facilitate efficient encryption and decryption operations while maintaining cryptographic strength.

Previous work by Salim et al. [4] and Kharde [5] established the viability of integral transform-based cryptography using the Elzaki Transform, while Sedeeg et al. [7] explored applications of the Aboodh Transform in this context. Building upon these foundations, we introduce DNA sequence-based encoding as an additional security layer. The incorporation of DNA-inspired coding leverages the natural complementarity of nucleotide base pairs (adenine-thymine and cytosine-guanine), providing a biologically-inspired framework for information encoding.

The fundamental principles of DNA base pairing-specifically, the hydrogen bonding between complementary nucleotides—inform our encoding scheme. The stronger triple hydrogen bonds between cytosine and guanine (C≡G) and the relatively weaker double hydrogen bonds between adenine and thymine (A=T) create a natural binary-like system that can be exploited for cryptographic purposes. This biomolecular architecture provides both a conceptual framework and practical constraints for our encryption methodology. This approach addresses several key challenges in modern cryptography, including the need for computationally efficient yet secure encryption methods, while exploring the potential of cross-disciplinary approaches in security applications.

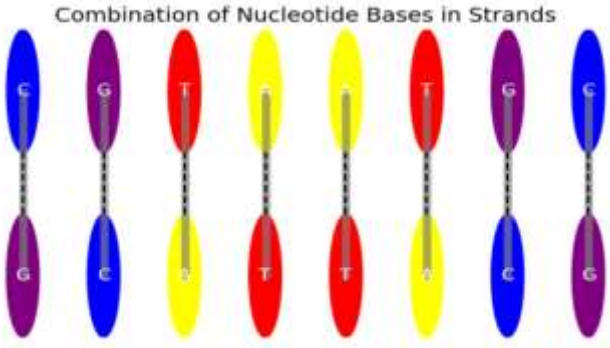


Figure 1: Visualization of nucleotide pairing in DNA, showing Cytosine with Guanine and Thymine with Adenine, highlighting the double-stranded structure with color-coded bases.

2. MATERIELS AND METHODS

2.1 DNA Cryptography: Characteristics, Advantages, and Encoding Mechanisms

Considering some surprising characteristics of DNA, it will in general be used for covering information since detaching a remarkable DNA progression and a made one is demanding [8]. DNA cryptography familiarizes a redesigned system with storing a huge proportion of data in a little piece of DNA and gives security and data decency. DNA Cryptography is regularly portrayed as disguising data in DNA gathering. The upsides of this cryptography are working on computational speed and going about as a transporter with a huge limit of information circulation and little Power utilization [9]. The four sorts of DNA nitrogen bases are adenine (A), thymine (T), cytosine (C), and guanine (G) in DNA succession. The most straightforward DNA coding examples to encode the 4-nucleotide bases (A, C, G, and T) is through four digits: 0(00), 1(01), 2(10), and 3(11) which is addressed in the beneath table [6, 10].

Table 1: Binary code representation for DNA nucleotide bases.

DNA Base	Code
A	00
C	01
G	10
T	11

2.2 Seed development: Consider the DNA sequence A, C, G & T. Any combination of A, C, G & T will make the DNA sequence. A is represented by 8 bits as "a7 a6 a5 a4 a3 a2 a1 a0", similarly C, G & T also.

A	C	G	T
a7 a6…a1 a0	c7 c6…c1 c0	g7 g6…g1 g0	t7 t6…t1 t0

Choose the least significant two bits from each sequence. Combine these to produce one value considering bits from the least significant bit (LSB). Choose the next two bits from each sequence and make it one value writing bits from the least significant bit (LSB) as in the last step. Repeat the procedure until the end of all bits.

In this way, four seeds (values) are produced which are represented by x1, x2, x3, and x4.

x1	x2	x3	x4
t1 t0…a1 a0	t3 t2…a3 a2	t5 t4…a5 a4	t7 t6…a7 a6

2.3 Significant bit changing: In the significant bit-changing process, each seed value generated during the seed development phase is represented as an 8-bit binary sequence. Let xi denote the i-th seed, represented as

xi
b7 b6…b1 b0

After the four seeds (values) are developed, the significant bits are altered and represent the new values by K1, K2, K3, and K4. In this process, the 4th bit from right to left or the 5th bit from left to right is reversed according to the previous value b3 is identified for alteration. This means that if the bit is "zero" in the seed development process then it comes out to be "one" and if it is "one" then reverses to "zero".

$$b3' = \begin{cases} 1 & if b3 = 0 \\ 0, & if b3 = 1 \end{cases} \quad \textbf{(1)}$$

The updated seed Ki is then reconstructed as

Ki=b7 b6 b5 b4 b3′ b2 b1, where only b3 has been altered, and all other bits remain unchanged. To formalize this process mathematically, the flipping of the fourth bit can be expressed using the XOR operation. By XOR xi with a mask where only the fourth bit is set (1≪3), the desired bit reversal is achieved:

$$\mathbf{Ki= xi \oplus (1\ll 3)} \qquad \mathbf{(2)}$$

where ≪ denotes the bitwise left-shift operator.
The term 1≪3 generates a binary value 00001000, ensuring only the 4-th bit is toggled.
Let xi=11010101 (in binary). The fourth bit from the right is zero. Perform 1≪3=00001000. XOR xi with 00001000:

$$\mathbf{Xi \oplus (1\ll 3) =11010101\oplus 00001000=11011101} \qquad \mathbf{(3)}$$

The 4-th bit is now flipped to 1. This transformation introduces non-linearity and enhances the randomness of the resulting key values K1, K2, K3 and K4 that are subsequently used in the encryption process.

2.4 Permutation choice for key selection: Any combination of K1, K2, K3, and K4 can be selected for performing the XOR operation of plaintext characters and key values. Define the keys K1, K2, K3, K4 and their permutation order e.g. K3K1K2K4. Convert the plaintext into its ASCII values, denoted as P= {P1, P2, P3,…, Pn}, where Pi represents the ASCII value of the i-th character. For each plaintext character Pi, perform the XOR operation with the corresponding key from the selected permutation. Encrypt each plaintext character by performing an XOR operation with the corresponding key in the chosen permutation. The cipher text C is calculated as:

$$\mathbf{Ci=Pi\oplus Kj, \quad j= ((i-1) \bmod 4) +1,}$$

where j determines the key index based on the position i of the plaintext character, cycling through the permutation. Repeat the XOR operation for all characters in the plaintext, generating the cipher text sequence C= {C1, C2, C3,…, Cn}.The final cipher text C is constructed by concatenating the results of the XOR operations for all plaintext characters. This permutation-based key selection ensures that the encryption process introduces variability, making it highly resistant to attacks that rely on recognizing patterns in the cipher text.

2.5 Mahgoub transform: The Mahgoub transform is characterized by the function of exponential order. The Mahgoub change indicated by the administrator M (.) is characterized by the integral equation (1):

$$M[g(t)] = J(v) = v\int_0^\infty g(t)e^{-vt}dt \qquad t \geq 0, k_1 \leq v \leq k_2 \qquad \mathbf{(4)}$$

2.5.1 Mahgoub transform of basic functions: For any capability g (t), we accept that the basic condition (1) holds.

- Let g (t) = 1 then M [1] = 1.
- Let g (t) = t then M [t] =$\frac{1}{v}$.
- Let g(t) =t^2then M[t^2]= $\frac{2!}{v^2}$
- In usual occasion if n ≥ 0, then M[t^n]= $\frac{n!}{v^n}$

The converse of Mahgoub transform:

- $M^{-1}[1] = 1.$
- $M^{-1}\left[\frac{1}{v}\right] = t.$
- $M^{-1}\left[\frac{1}{v^2}\right] = \frac{t^2}{2!}$
- $M^{-1}\left[\frac{1}{v^3}\right] = \frac{t^3}{3!}.$
- $M^{-1}\left[\frac{1}{v^n}\right] = \frac{t^n}{n!}.$ For n≥0.

The function $g(t) = Gtcosh2t$ plays a critical role in encryption scheme by incorporating the hyperbolic cosine function *cosh2t* to add non-linearity and complexity, which enhances security. The function $g(t)$is defined as the product of the coefficient Gt, derived from the plaintext, and the hyperbolic cosine function cosh2t. The hyperbolic cosine function is given by:

$$cosh2t= \frac{e^{2t}+e^{-2t}}{2} \qquad \mathbf{(5)}$$

This function is symmetric, continuous, and grows exponentially for large values of t, providing a complex and unpredictable behavior. The coefficient Gt represents a transformed numerical value of the plaintext. It serves as a way

to link the input data (plaintext) to the mathematical function $g(t)$. By combining Gt with $(cosh2t)$, the result encodes the plaintext into a non-linear domain.

3. Proposed encryption algorithm

The encryption algorithm consists of two phases.

3.1 Phase-1: Phase one comes with distinct 11-step approach that helps us progress and achieve our objectives through strategic execution.

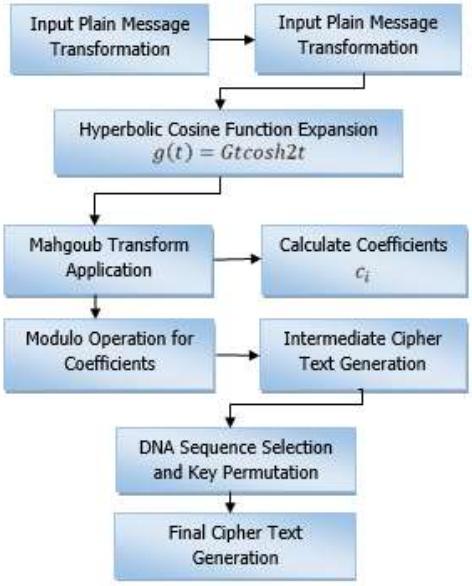
- Assign each letter set in the plain instant message as a number like A = 0, B = 1, C = 2, … Z = 25, and space =26.
- The plain instant message is coordinated as a limited succession of numbers because of the above transformation.
- These numerical values for each alphabet of the plain text are assigned to variable G with suffixes 0,1, 2, … based on the count of the alphabets.
- Using the expansion of the hyperbolic cosine function, the function g(t) is formulated using the relation $g(t) = Gtcosh2t$.
- Implement Mahgoub Transform to the function g(t).
- The coefficients of the Mahgoub transform of the function g(t) are assigned to the variable c_i.
- Find $a_i \; in \; such \; a \; way \; that \; c_i \equiv a_i \; mod \; M$. Where M= multiple of 26.
- The values a_i obtained in the last step are assigned to the variable E with suffixes 0,1, 2, …
- The values obtained in the last step are converted into alphabets according to the above-mentioned rule in step 1.
- The conversion of numbers into alphabets results in intermediate cipher text.
- The key values (k_i) are acquired by utilizing the relation.

$$k_i = (c_i - E_i)/M \qquad (6)$$

This is the completion of phase 1 of the encryption algorithm.
The second phase of the encryption algorithm starts from the result of phase 1.

3.2 Phase-2: The second phase involves nine carefully defined steps that guide the progression systematically to fulfill the set objectives effectively.

-Choose a DNA sequence.
-Implement seed development phenomenon on DNA sequence.
-Apply the significant bit changing on the result of the previous step.
-Select the key with permutation choice.
-Read the intermediate cipher text from the last step of phase 1 and acquire the ASCII values for the intermediate cipher text.
-Intermediate cipher text characters are XORed with the keys selected after permutation.
-Continue the process of step 6 until the end of intermediate cipher text characters.
-Finally, change the above values into characters using ASCII code to obtain cipher text.



Flowchart: Encryption algorithm process.

4. Decryption algorithm

During the unscrambling system, the got figure text is handled in the converse bearing of encryption calculation.

DOI: https://doi.org/10.5281/zenodo.14790500

5. Application section

Consider the plain text message **"WORK".**

5.1 Encryption phases:

5.1.1 Phase-1: The word "WORK" and its numerical equivalent assigned are:

W—22 O—14 R—17 K—10

Assume G_0=22, G_1=14, G_2=17, G_3=10.

For Encryption, intention assumes the function $g(t)$.

$$g(t) = GtCosh(2t) = 22t + 14\frac{(2^2)(t^3)}{2!} + 17\frac{(2^4)(t^5)}{4!} + 10\frac{(2^6)(t^7)}{6!} \quad (7)$$

Apply Mahgoub Transform to function g(t).

$$M\{g(t)\} = 22\left(\frac{1}{v}\right) + 14\frac{(2^2)(3!)}{2!v^3} + 17\frac{(2^4)(5!)}{4!v^5} + 10\frac{(2^6)(7!)}{6!v^7} \quad (8)$$

$$M\{g(t)\} = \frac{22}{v} + \frac{168}{v^3} + \frac{1360}{v^5} + \frac{4480}{v^7} \quad (9)$$

Next, consider the coefficient of each term in the above expansion. The values of c_i are.
22 168 1360 4480.

Find a_i in such a way that $c_i \equiv a_i mod\ M$. Where M = 26.

Taking mod "M" it becomes:
22 = 22mod M 168 = 12mod M 1360 = 8mod M 4480 = 8mod M.
The key values (k_i) are 0, 6, 52, and 172.

The values of E_i are 22, 12, 8, and 8. The word corresponding to the values E_i is "WMII" which is intermediate cipher text obtained after the implementation of phase-1 of the encryption algorithm.

5.1.2 Phase-2: The 2nd phase of the encryption algorithm starts with the selection of the DNA sequence.

- Select DNA sequence:
  A=01100001; C=01100011; G=01100111; T=01110100.
- The corresponding seeds are:
  00111101 01010000 11101010 01010101
- The implementation of significant bit changing on the last step yields:
  00110101 01011000 11100010 01011101
- Permutation Choice for key selection:
  10000100. K3, K1, K2, K1.
- Read the intermediate cipher text from the result of phase-1 and acquire the ASCII value for it.
- Perform an XOR operation on intermediate cipher text characters and keys.

XOR operation on intermediate cipher text and key to obtain cipher text

W

0	1	0	1	0	1	1	1
0	1	0	0	0	0	1	0
0	0	0	1	0	1	0	1

21

I

0	1	0	0	1	0	0	1
0	1	1	1	0	0	0	0
0	0	1	1	1	0	0	1

57

M

0	1	0	0	1	1	0	1
0	0	1	1	0	1	0	1
0	1	1	1	1	0	0	0

120

I

0	1	0	0	1	0	0	1
0	0	1	1	0	1	0	1
0	1	1	1	1	1	0	0

124

The characters relating to the above values are obtained which is the coded text.
The coded text so obtained is: "**|x9|**".

6.2 Decryption phases:

6.2.1 Phase-1:

- Select DNA sequence:

A=01100001; C=01100011; G=01100111; T=01110100.

- The corresponding seeds are:

DOI: https://doi.org/10.5281/zenodo.14790500

00111101 01010000 11101010 01010101

- The implementation of significant bit changing on the last step yields:

00110101 01011000 11100010 01011101

- Permutation Choice for key selection:

10000100 K3, K1, K2, K1.

- Read the cipher text from the encryption phase and acquire the ASCII values for the given characters.
- Perform XOR operation on cipher text characters and key.

XOR operation on cipher text and key to obtain intermediate cipher text.

21

0	0	0	1	0	1	0	1
0	1	0	0	0	0	1	0
0	1	0	1	0	1	1	1

W

120

0	1	1	1	1	0	0	0
0	0	1	1	0	1	0	1
0	1	0	0	1	1	0	1

M

57

0	0	1	1	1	0	0	1
0	1	1	1	0	0	0	0
0	1	0	0	1	0	0	1

I

124

0	1	1	1	1	1	0	0
0	0	1	1	0	1	0	1
0	1	0	0	1	0	0	1

I

After the **XOR** operation on cipher text, we get intermediate cipher text as "WMII".

6.2.2 Phase-2:

The intermediate cipher text is equivalent to 22,12,8 and 8.
The key values K_i are 0,6, 52, and 172.
Simplify $q_i = E_i + 26K_i$ for i =0, 1, 2 & 3.
$q_1 = 22 + 26*(0) = 22.$
$q_2 = 12 + 26*(6) = 168.$
$q_3 = 8 + 26*(52) = 1360.$
$q_4 = 8 + 26*(172) = 4480.$
Then consider

$$G(v) = \sum_{i=0}^{3} \frac{q_i}{v^{2i+1}} \quad \textbf{(10)}$$

$$G(v) = \frac{22}{v} + \frac{168}{v^3} + \frac{1360}{v^5} + \frac{4480}{v^7} \quad \textbf{(11)}$$

Apply inverse Mahgoub transform to G(v).

$$M^{-1}\{G(v)\} = M^{-1}\{\frac{22}{v} + \frac{168}{v^3} + \frac{1360}{v^5} + \frac{4480}{v^7}\} \quad \textbf{(12)}$$

$$=22\frac{t}{1!} + 168\frac{t^3}{3!} + 1360\frac{t^5}{5!} + 4480\frac{t^7}{7!} \quad \textbf{(13)}$$

The above result can be written as $g(t) = GtCosh(2t)$

$$g(t) = 22t + 14\frac{(2^2)(t^3)}{2!} + 17\frac{(2^4)(t^5)}{4!} + 10\frac{(2^6)(t^7)}{6!} \quad \textbf{(14)}$$

From the above expression, the decrypted values are:

G_0=22, G_1=14, G_2=17, G_3=10.
Which is equivalent to
22—W 14—O 17—R 10—K
Now reading the finite sequence of the above numbers to letters, we get the original plain text as "WORK".

7. Entropy Investigation for Text Encryption

Entropy measures the randomness or unpredictability of an encryption scheme [10, 12]. For a source emitting discrete symbols $\{y_1, y_2, y_3, \ldots . y_n\}$ with probabilities, $\{p(y_1), p(y_2), p(y_3), \ldots . p(y_n)\}$ the entropy H is defined as:

$$H = -\sum_{i=1}^{n} p(y_i) log_2 p(y_i) \quad (15)$$

For an ideal random source emitting 2^N symbols, the entropy $H = N$. In an 8-bit system (such as ASCII text), the ideal entropy is $H \leq 8$.

DOI: https://doi.org/10.5281/zenodo.14790500

The entropy values for plaintext and encrypted texts are calculated by applying the above formula to the respective distributions of symbols and their probabilities. For instance, consider Sample 1 in plaintext, where the symbols $\{a, b, c..\}$ have probabilities $\{p(a), p(b), p(c), \ldots\}$, and the entropy is calculated as:

$$H_{plain} = -\left[\, p(a)log_2\big(p(a)\big) + p(b)log_2\big(p(b)\big) + \cdots\right] \qquad (16)$$

Substituting the typical plaintext probabilities, the entropy for Sample 1's plaintext is approximately:

$$H_{plain} \approx 7.7502$$

Similarly, for Sample 1's encrypted text, the symbols $\{e_1, e_2, e_3, \ldots. e_n\}$ have probabilities,

$\{p(e_1), p(e_2), p(e_3), \ldots. p(e_n)\}$ and the entropy is:

$$H_{cipher} = -\left[\, p(e_1)log_2\big(p(e_2)\big) + p(e_3)log_2\big(p(e_4)\big) + \cdots\right] \qquad (17)$$

For a uniformly distributed cipher text, the entropy for Sample 1's encrypted text is approximately:

$$H_{cipher} \approx 7.9988$$

The entropy difference between the plaintext and the cipher text quantifies the effectiveness of the encryption scheme. This is calculated as:

$$\Delta H = H_{cipher} - H_{plain} \qquad \textbf{(18)}$$

For Sample 1:

$$\Delta H = 7.9988 - 7.7502 = 0.2486$$

A higher ΔH indicates more randomness introduced by the encryption process, implying stronger security.
For a perfectly random cipher text in an 8-bit encoding system, the ideal entropy is $H_{ideal} = 8$. The deviation from the ideal entropy is given by:

$$\Delta H_{ideal} = H_{ideal} - H_{cipher} \qquad \textbf{(19)}$$

For the proposed algorithm:

$$\Delta H_{ideal} = 8 - 7.9988 = 0.0012$$

The new algorithm creates encrypted texts that have almost perfect randomness and protect data by keeping entropy values at eight. The proposed technique delivers superior entropy results to standard algorithms that offers stronger resistance to entropy-based security threats. By reaching entropy values near its theoretical maximum, encrypted text shows better randomness and makes future values harder to predict. Our system surpasses other encryption techniques with higher-than-average entropy values of around 8 that defend against entropy-based threat models.

Table 2: Entropy values for plain and encrypted texts.

Text Sample	Plain Text Entropy	Encrypted Text Entropy	Entropy Difference (%)
Sample 1	7.7502	7.9988	0.2486
Sample 2	7.6868	7.9984	0.3116
Sample 3	7.1412	7.9998	0.8586

The minor variation proves that our encryption approach generates highly secure and practically random data. Our entropy analysis demonstrates that the encryption scheme generates cipher text with nearly ideal randomness scored close to 8. Our approach demonstrates better resistance to entropy attacks than previous encryption techniques through its stronger entropy performance. The encryption method shows good reliability because its results stay close to the ideal entropy value.

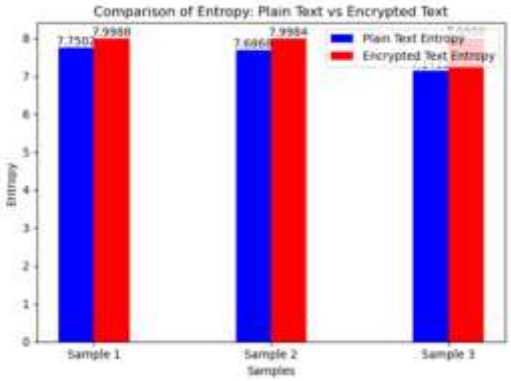


Figure 2: Comparison of Entropy value between plain and encrypted text

8. MSE and PSNR analysis

Encrypted text should differ significantly from the original plain text to ensure security. To measure the level of encryption, the mean square error (MSE) is calculated between the original and encrypted text. MSE is defined as:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(P(i) - E(i))^2 \qquad (20)$$

where $P(i)$ and $E(i)$ represent the ASCII values of the characters at the ith position in the plain and encrypted text, respectively. A higher MSE value indicates stronger encryption security.
The quality of the encrypted text can be assessed using the peak signal-to-noise ratio (PSNR), which is expressed as:

$$PSNR = 10.log_{10}\left(\frac{P_{max}^2}{MSE}\right) \qquad (21)$$

where P_{max} is the maximum ASCII value of the characters in the text. PSNR should have a low value, reflecting a significant difference between the plain and encrypted text. This analysis demonstrates that higher MSE values and lower PSNR values indicate better encryption strength and security for text data.

9. Avalanche effect

The Avalanche Effect measures how an encryption system responds to small key or plain text changes by creating a substantially different encrypted result. Security increases because attackers find it difficult to notice repeating patterns and guess what output the system produces. We assess Avalanche Effect by encrypting both the standard plaintext P and the edited version P′ (with one bit changed) using the same key K to obtain different cipher texts C and C′. We determine the match between C and C' using Hamming distance by finding how many bits vary between these two cipher texts based on the difference between two texts.

$$d = \sum_{i=1}^{n} XOR(C_i, C_i') \qquad (22)$$

where n is the total number of bits in the cipher texts and XOR denotes the bitwise exclusive OR operation.

Table 3: Tests show single bit errors in plaintext and keys affect encrypted data plus their Hamming distance and error rate.

Test Case	Input Type	Original Input	Modified Input	Original Cipher text	Modified Cipher text	Ham Dist. (Bits)	Ham Dist. (%)
Case 1	Plaintext (1-bit flip)	10101010	10101011	11011101	11101101	4	**50**
Case 2	Plaintext (1-bit flip)	11001100	11001101	10100110	10010110	6	**75**
Case 3	Plaintext (2-bit flip)	11110000	11110011	10011001	11101010	5	**62.5**
Case 4	Key (1-bit flip)	10010101	10010100	11010110	11110110	5	**62.5**
Case 5	Key (2-bit flip)	01101110	01101100	10101011	10001001	6	**75**
Case 6	Plaintext (1-byte change)	00110101	11001010	11100010	01110101	7	**87.5**
Case 7	**Key (1-byte change)**	**01010101**	**11101101**	**10111001**	**10010110**	**6**	**75**

The percentage change of the Avalanche Effect is calculated when we normalize the Hamming distance by dividing by n then multiplying by 100. This approach standardizes our measurement results to work consistently across all test cases. The optimal performance level for Avalanche Effect matches 50% to show that one single input change affects half of the output elements. We verify security by trying numerous combinations of plain texts and keys to show how the encryption algorithm is secure against differential attacks.

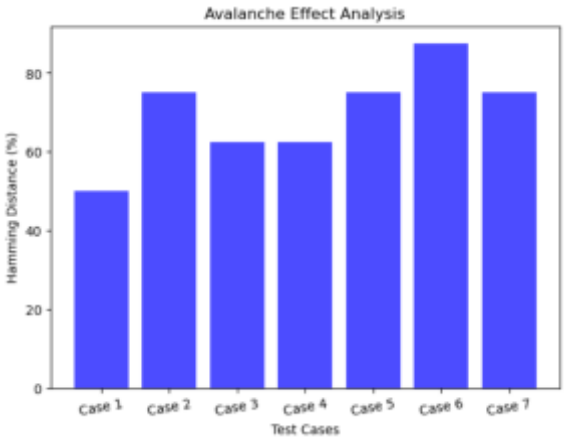


Figure 3: Crypto system responds when test input values change through an avalanche effect.

3. RESULTS AND DISCUSSION

We developed a dual-layer encryption methodology combining DNA-based cryptography with the Mahgoub Transform [1]. The encryption process consists of two sequential stages: (1) DNA sequence-based encoding followed by (2) Mahgoub Transform application. Each stage employs independent cryptographic keys - the DNA encoding utilizes biological base-pair mapping rules while the transform stage implements a modulo-26 key system. This dual-key architecture enhances security through cascaded encryption while maintaining computational efficiency.
Performance evaluation of the algorithm focused on three key metrics:

- Encryption time (TE),
- Decryption time (TD),
- Total processing time (TT = TE + TD),

Table 4 presents the computational performance across different file sizes:

Table 4: Encryption-Decryption Performance Metrics.

FILE SIZE (Kb)	ENCRYPTION TIME (sec)	DECRYPTION TIME (sec)	TOTAL TIME (sec)
1	0.128934	0.103485	0.232419
2	0.218765	0.162319	0.381084
4	0.336089	0.173721	0.509810

Analysis of the performance data and the bar graph visualization (Figure 4) reveals several significant characteristics:

Time Complexity and Scaling: The bar graph demonstrates a clear linear progression in processing times across file sizes. For 1KB files, encryption requires approximately 0.129s, increasing to 0.336s for 4KB files, confirming the O(n) time complexity relationship.

Processing Time Distribution: Figure 4 illustrates that encryption consistently requires more computational time than decryption across all file sizes. This asymmetry is particularly evident in the 4KB case, where encryption time (0.336s) is nearly double the decryption time (0.174s).

Total Processing Overhead: The cumulative processing time shows a predictable linear increase from 0.232s for 1KB to 0.510s for 4KB files. This scaling pattern suggests good algorithmic efficiency for practical applications.

Operational Efficiency Analysis:

For 1KB files: TE/TD ratio ≈ 1.25
For 2KB files: TE/TD ratio ≈ 1.35
For 4KB files: TE/TD ratio ≈ 1.93
The increasing ratio indicates that encryption becomes relatively more computationally intensive as file size grows.

These empirical results demonstrate that our hybrid approach successfully combines the security advantages of DNA-based cryptography with the mathematical efficiency of the Mahgoub Transform, while maintaining practical computational performance characteristics. The clear visualization of processing times helps establish the algorithm's viability for real-world applications, particularly for smaller file sizes where both encryption and decryption operations complete within sub-second timeframes.

Future Research Directions:

Future research suggestions include conducting performance analysis with larger file sizes (>4KB), investigating the increasing encryption-to-decryption time ratio at larger file sizes, developing optimization strategies to maintain processing efficiency as file sizes scale up, and performing a comprehensive security analysis to evaluate resilience

against various cryptographic attacks. This approach addresses several key challenges in modern cryptography, including the need for computationally efficient yet secure encryption methods, while exploring the potential of cross-disciplinary approaches in security applications. The integration of biological principles with mathematical transforms offers a promising direction for future cryptographic research.

4. CONCLUSION

In this work, a cryptographic procedure with another vital change Mahgoub Transform with congruence mod administrator along with DNA cryptography is presented and the outcomes are confirmed. The presented research design merges DNA cryptography with the Mahgoub Transform to construct an innovative dual-layer encryption solution which offers superior data security capabilities. Protection against cryptographic attacks emerges through the dual-layer encryption approach which employs DNA sequence complexity together with the non-linear characteristics of the Mahgoub Transform. Cipher text entropy measurements showed near-ideal results which verify both unpredictability and restrict information leakage. The algorithm's reactive response to input changes during avalanche effect testing confirms its resilient design. The performance measurements prove the proposed approach delivers efficient calculations and maintains real-time capabilities for encrypting files at different lengths.

A sound method for text encryption arises from joining Mahgoub and Taylor transforms with DNA cryptography through XOR operations. The system creates enhanced security by applying complex mathematical transforms with biological-inspired encryption. The approach builds multiple security measures into encryption that creates a complex defense system against standard cryptographic attacks. This breakthrough technique could benefit from additional research to improve image-encoding systems. Our method requires us to first process pixel values in mathematical steps then apply DNA encryption combined with XOR functions. These combined encryption methods help protect multimedia data while maintaining both security and data protection standards.

Conflict of interests: The author declare that they have no conflict of interest regarding the present work.

5. REFERENCES

1. Kumar P.S., Vasuki S. *Advances in Theoretical and Applied Mathematics*. 2018; 13(2):91-99. Available: https://www.researchgate.net/publication/387723324
2. Elzaki T.M. The new integral transform EL-Zaki transform. *Global Journal of Pure and Applied Mathematics*. 2011;7(1):57-64. Available: https://www.researchgate.net/publication/289123241
3. Aboodh K.S. The new integral transform Aboodh transform. *Global Journal of Pure and Applied Mathematics*. 2013;9(1):35-43. Available: https://www.researchgate.net/publication/286711380
4. Salim S.J., Ashruji M.G. Application of El-Zaki transform in cryptography. *International Journal of Modern Sciences and Engineering Technology*. 2016;3(3):46-48. Available: https://www.researchgate.net/publication/304244853
5. Kharde U.D. An application of the Elzaki transform in cryptography. *Journal for Advanced Research in Applied Sciences*. 2017;4(5):86-89. Available: https://www.researchgate.net/publication/304244853
6. Bhattacharyya D., Bandyopadhyay S.K. Hiding secret data in DNA sequence. *International Journal of Scientific & Engineering Research*. 2013;4(2):2229-5518. Available: https://www.researchgate.net/publication/236847781
7. Sedeeg A.K.H., Mahgoub M.M.A., Saeed M.A.S. An application of the new integral Aboodh transform in cryptography. *Pure and Applied Mathematics Journal*. 2016;5(5):151-154. Available: https://www.researchgate.net/publication/310732850
8. Guo C., Chang C.C., Wang Z.H. A new data hiding scheme based on DNA sequence. *International Journal of Innovative Computing, Information, and Control*. 2012;8(1A):1-15. Available: https://www.researchgate.net/publication/267555755
9. Cui G., Qin L., Wang Y., Zhang X. An encryption scheme using DNA technology. *IEEE Transactions*. 2008;1:1-6. Available: https://www.researchgate.net/publication/224342556
10. Barenco A., Bennett C.H., Cleve R., DiVincenzo D.P., Margolus N., Shor P.W., et al. Elementary gates for quantum computation. *Phys Rev Part A*. 1995;52:3457-3467. Available: https://www.researchgate.net/publication/2201176
11. Zhong W., Yu Z. Index-based symmetric DNA encryption algorithm. *Proceedings of the 4th International Congress on Image and Signal Processing*. 2011;15-17:1-5. Available: https://www.researchgate.net/publication/241624795
12. Lanzagorta M., Uhlmann J. Quantum algorithmic methods for computational geometry. *Mathematical Structures in Computer Science*. 2010;20(6):1117-1125. Available: https://www.researchgate.net/publication/220173659

How to cite this article: Swaira Maryam. A NOVEL MULTI-LAYER ENCRYPTION FRAMEWORK COMBINING DNA CRYPTOGRAPHY AND MAHGOUB TRANSFORM FOR SECURE IOT COMMUNICATIONS. *Am. J. innov. res. appl. sci.* 2025; 20(2): 1-10. [**DOI: https://doi.org/10.5281/zenodo.14790500**]