Chaos-Based Image Encryption Using Refined Hash Function

Ejiiobe Precious Nyimeobari
Department Of Computer Science, Rivers State University, Rivers State
precious.ejiiobe@ust.edu.ng
DOI: 10.56201/ijcsmt.v11.no2.2025.pg59.72

Abstract

This study presents an enhanced chaos-based image encryption framework that integrates refined hash functions to address modern challenges in secure image transmission and cryptographic robustness. Conventional encryption techniques often struggle to defend against sophisticated cyberattacks and ensure data integrity, especially in critical applications such as medical imaging and IoT security. The proposed encryption system combines the inherent unpredictability of chaotic systems with advanced hashing mechanisms to develop a robust security framework. A chaos generator and mapping function produce high-quality chaotic signals, which are transformed into the cryptographic domain to enhance encryption strength. The refined hash function incorporates image-specific attributes, generating unique and highly sensitive hash values capable of detecting even the slightest data alterations. The encryption system is implemented in Python, leveraging its extensive cryptographic libraries for precision and efficiency. Performance evaluation is conducted using statistical metrics such as histogram analysis, adjacent pixel autocorrelation, and key sensitivity tests, which reveal significant improvements in randomness and robustness against differential attacks. Experimental results indicate an NPCR of 99.81%, UACI of 33.69%, and Shannon entropy of 7.998, demonstrating superior encryption quality compared to traditional methods. Additionally, the proposed system achieves faster execution times, making it suitable for real-time applications requiring high security and computational efficiency. This research contributes to cryptography by proposing a scalable and efficient encryption model that mitigates existing vulnerabilities while ensuring robust protection of sensitive data. The practical applications of this system extend to secure communications, healthcare data protection, and IoT security, where confidentiality and integrity are paramount. By addressing critical gaps in traditional encryption techniques, this study provides a foundation for developing more resilient cryptographic systems capable of withstanding modern cyber threats

Keywords: Chaos, Image Encryption, Refined Hash Function

1. INTRODUCTION

The enhancement of chaos-based encryption through refined hash function is a crucial area of research in computer security. Chaos-based encryption methods have demonstrated potential in establishing secure communication channels, and the incorporation of refined hash functions aims to improve the resilience and efficiency of these encryption techniques (Liu *et al.,* 2021). The exploration of utilizing multidimensional features of input light for encryption and storage has revealed innovative implementation methods in this domain, indicating the possibility of advancing encryption techniques (Liu *et al.,* 2021). Furthermore, the assessment of security challenges in Controller Area Network (CAN) bus systems has emphasized the significance of encryption mechanisms in safeguarding communication protocols (Bozdal *et al.,* 2020).

Moreover, the investigation of secure computation based on homomorphic encryption in Vehicular Ad Hoc Networks (VANETs) has highlighted the importance of homomorphic encryption in ensuring secure data processing in network environments (Sun *et al.,* 2020). These references collectively underscore the growing importance of encryption methods in diverse technological domains.

The exponential growth of digital data transmission, particularly in sensitive domains like healthcare and IoT, has amplified the demand for robust encryption techniques. Traditional encryption methods (e.g., AES, DES) often falter against sophisticated cyberattacks due to their linearity and deterministic nature. Chaos-based encryption, leveraging the inherent unpredictability and sensitivity of chaotic systems, has emerged as a promising alternative. However, challenges persist in ensuring data integrity, resisting differential attacks, and maintaining real-time efficiency. This study addresses these gaps by integrating a refined hash function into a chaos-based encryption framework, enhancing both security and computational efficiency.

This work aims to Design a chaos-based image encryption system using refined hash function, evaluate the system's security through statistical metrics like NPCR, UACI, Entropy and also to Compare its performance with existing methods in terms of execution time and robustness. The proposed system enhances data confidentiality and integrity for applications like medical imaging and IoT security. By combining chaotic dynamics with a hash function sensitive to minute data alterations, it offers a scalable solution resistant to differential and brute-force attacks.

This study is important for several reasons. First, chaos-based encryption offers superior security by leveraging its extreme sensitivity to initial conditions and the inherently unpredictable nature of chaotic systems. Second, in an era where protecting sensitive information is critical, the study's findings pave the way for more effective encryption techniques that safeguard data across diverse applications—from communication systems to financial transactions. Finally, as cyber threats continually evolve, the optimized chaos-based encryption method proposed here serves as a powerful tool in reducing risks such as unauthorized access, data breaches, and other malicious activities.

2. RESEARCH METHOD

The design methodology for a chaos-based encryption system involves a constructive research approach to conceptualize, develop, and implement the encryption technique. It encompasses the systematic identification, selection, processing, and analysis of information pertinent to the research. This methodology serves as a scientific framework for addressing the problem at hand, guiding researchers in their exploration, description, explanation, and prediction of phenomena related to chaos-based encryption.

At its core, the research methodology for chaos-based encryption entails studying the methods through which knowledge is acquired and applied in the creation of the encryption system. Its primary objective is to outline a comprehensive work plan for conducting the research, ensuring that each step is carefully designed and executed to achieve meaningful results

2.1 EXPERIMENTAL DATA ANALYSIS

The dissertation utilizes several analyses which includes, Intensity histogram analysis, Adjacent pixel autocorrelation test, and key sensitivity tests, to assess the efficiency and robustness of encryption algorithms. The intensity histogram of the original image in Figure 1.0 graphically depicts the spread of pixel intensities. The x-axis represents the spectrum of pixel values, often ranging from 0 to 255 in grayscale images. Lower values indicate darker tones, while higher values indicate brighter tones. The y-axis represents the number or frequency of pixels for each intensity level, indicating the prevalence of certain intensities in the image. Within the framework of an intensity histogram with a y-axis range of 0 to 1400, it indicates that the image exhibits a wide-ranging distribution of pixel intensities, encompassing both dark and brilliant values. The pixel count distribution extends up to 1400 pixels for a specific intensity level. Examining these histograms is advantageous for comprehending the global contrast and tonal attributes of an image, facilitating image processing endeavors such as contrast modifications or discerning distinct features based on intensity patterns.

Figure 1.0: Original Image

Table 1.0: Histogram analysis of the original image

Pixel Value	Red Channel Count	Green Channel Count	Blue Channel Count
0	2081	2380	2408
1	145	169	150
2	136	157	153
3	141	153	153
4	140	148	142
5	113	119	123
6	105	101	99
7	94	118	103
8	143	159	142
9	106	117	124

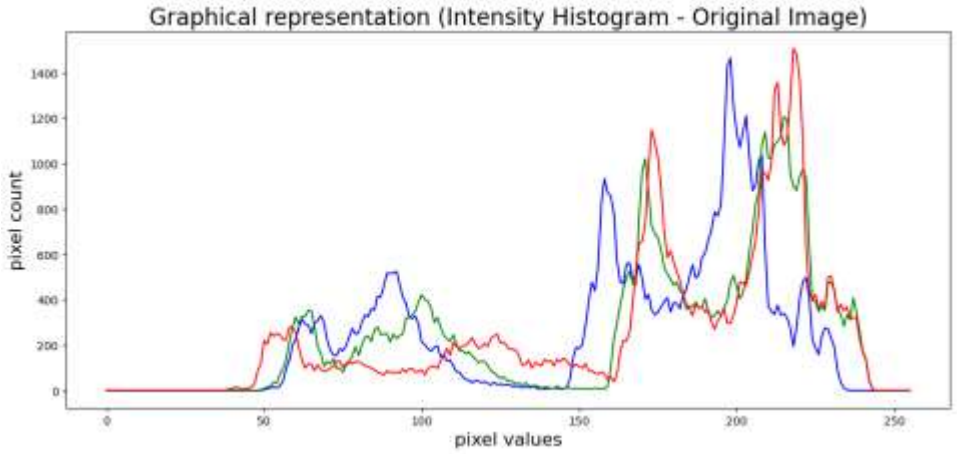

Figure 1.1: Intensity of the original image (histogram)

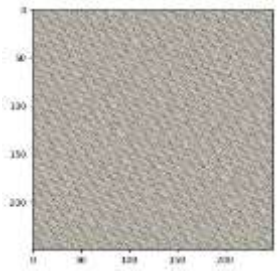

Figure 1.2: Encrypted Image

Table. 1.2: Histogram Analysis Table of the Encrypted Image

Pixel Value	Red Channel Count	Green Channel Count	Blue Channel Count
0	2877	1967	1757
1	558	641	423
2	637	664	376
3	708	762	415
4	649	733	376
5	707	840	422
6	737	801	337
7	716	843	416
8	659	856	326
9	670	743	302

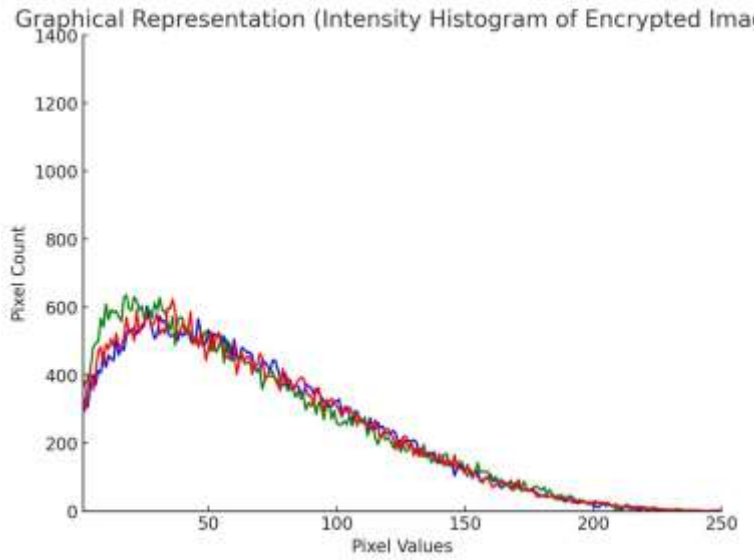

Figure 1.3: Intensity of the encrypted image (histogram)

The histogram offers a visual depiction of the distribution of pixel values throughout the full intensity spectrum, providing insights on the general brightness and contrast characteristics of the encrypted image. Examining the histogram allows for the identification of patterns, peaks, or fluctuations in pixel intensities, which assists in evaluating the visual attributes of the image and potential qualities affected by the encryption procedure.

2.2: STATISTICAL AND SECURITY ANALYSIS

The robustness of the encryption scheme was evaluated using several widely recognized metrics, including the Number of Pixels Change Rate (NPCR), Unified Average Changing Intensity (UACI), Peak Signal-to-Noise-Ratio (PSNR), and Shannon Entropy. These metrics assess how effectively the encryption disrupts the original image's structure, how sensitive the encryption is to small changes, and the overall unpredictability of the encrypted image. The encryption achieved an NPCR of 99.814% and a UACI of 33.694%, indicating high resistance to differential attacks. The Shannon Entropy value of 7.998 demonstrates that the cipher image is highly random, minimizing the possibility of leaking any useful information. Additionally, the PSNR of 7.723 indicates the image distortion introduced by encryption is substantial, further enhancing security. This can be seen in Table 4.8

Table 2.0: Statistical and Security Analysis Metrics

Metric	Value
Number of Pixels Change Rate (NPCR)	99.814%
Unified Average Changing Intensity (UACI)	33.694%
Peak Signal-to-Noise-Ratio (PSNR)	7.723
Shannon Entropy	7.998

3.0: PROPOSED IMPROVED CHAOS-BASED IMAGE ENCRYPTION SYSTEM

The system architecture in Figure 3.0 incorporates a comprehensive approach to encryption, leveraging chaos theory and refined hash functions to ensure robust data security. At its core, the architecture comprises a chaos generator for producing high-quality chaotic signals, a mapping function to transform these signals into a suitable domain for cryptographic operations, and a key generation mechanism to generate secure cryptographic keys. Encryption and decryption processes are facilitated through chaos-enhanced operations and the application of the refined hash function, providing an additional layer of security. User interaction is facilitated through a user-friendly interface, while secure key management ensures the protection of encryption keys. Rigorous security measures, including entropy evaluation and cryptographic tests, are integrated into the architecture to uphold data confidentiality and integrity. Overall, the architecture is designed to deliver high-level security, performance, and reliability in encrypting and decrypting sensitive data.

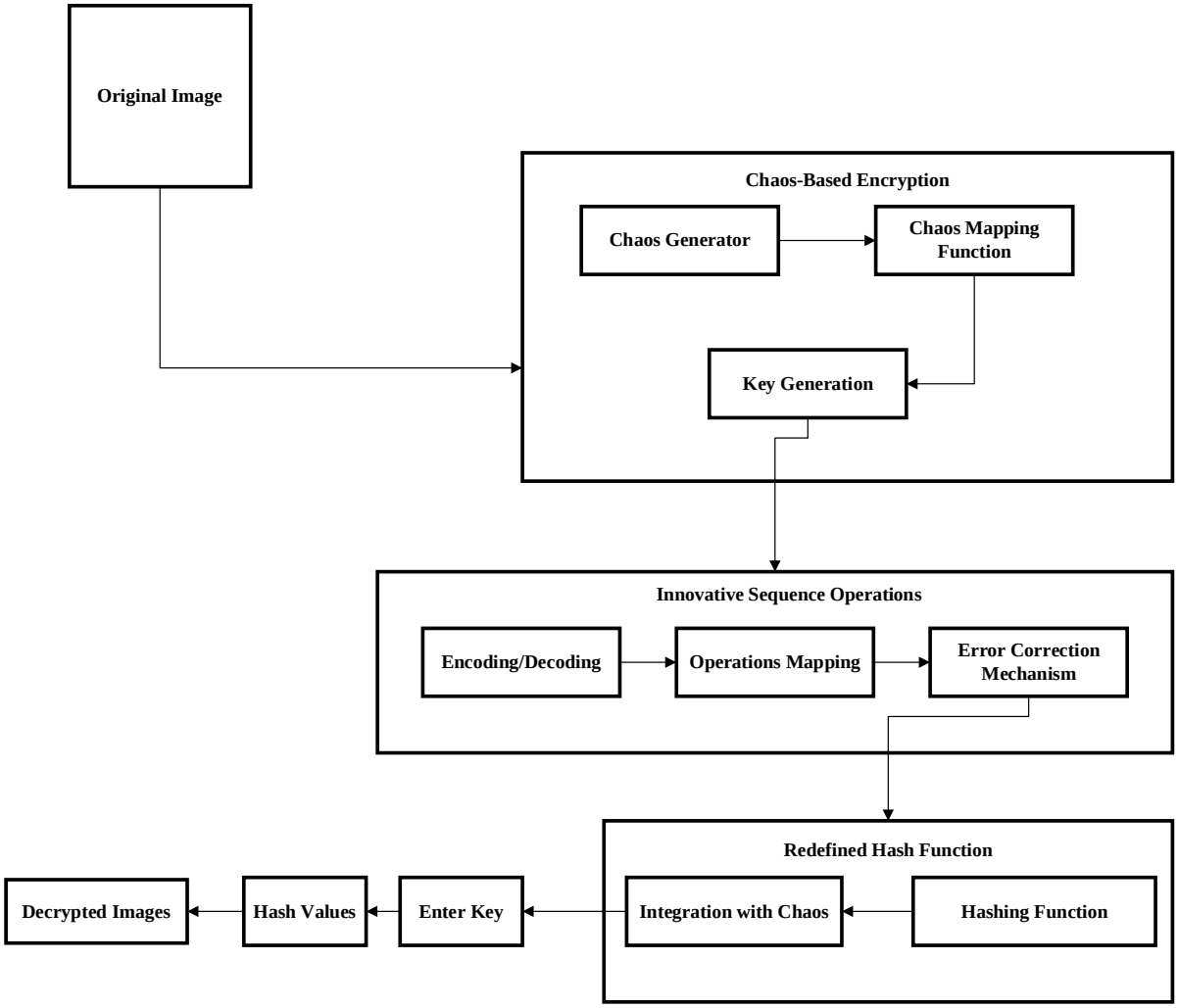

Figure 3.0: Architecture of the Proposed System

3.1 ALGORITHMS OF THE CHAOS-BASED IMAGE ENCRYPTION USING REFINED HASH FUNCTION

Algorithm 3.1: Chaotic Based Hash Function

```
1) INPUT: image, x_0, λ
2) def ChaosBasedHash (image, x_0, λ):
3) M, N = image.shape  # Get dimensions of the image
4) chaotic_sequence = LogisticMap(x_0, λ, M * N)  # Generate chaotic sequence
5) chaotic_sequence = chaotic_sequence.reshape ((M, N))
6) hash_value = 0 # Initialize hash value
7)  for i in range(M):
8) for j in range(N):
9)         pixel_value = image [i, j]   # Get pixel value at (i, j)
10)        chaotic_value = chaotic_sequence[i, j]
11)         hash_value += pixel_value * np.sin(2 * np.pi * chaotic_value)  # Apply hash
12) return hash_value # OUTPUT
```

Algorithm 3.2: Logistic Map Operation

```
1)  INPUT: x_0, λ, iterations
2)  def LogisticMap(x_0, λ, iterations):
3)  x_values = np.zeros(iterations)  # Create array of size 'iterations'
4)  x = x_0
5)  for i in range(iterations):
6)         x = λ * x * (1 - x)   # Apply logistic map equation
7)         x_values[i] = x   # Store chaotic value
8)  return x_values   # OUTPUT
```

Algorithm 3.3: Encoding Operation

```
1 INPUT: image
2 def EncodeData(image):
3   nucleotides = ['A', 'C', 'G', 'T']
4    encoded_data = []
5 for pixel in image.flatten():
6        nucleotides = nucleotides [pixel % 4]   # Map pixel to nucleotide base
7        encoded_data.append(nucleotide)
8   return encoded_data   # OUTPUT
```

Algorithm 3.4: Chaotic Encryption Operation

```
1 INPUT: encoded_data, x_0, λ
2 def ChaoticEncryption(encoded_data, x_0, λ):
3 iterations = len(encoded_data)
4 chaotic_sequence = LogisticMap(x_0, λ, iterations)  # Generate chaotic sequence
5 encrypted_data = []
6   for i in range(iterations):
7         chaotic_value = int(chaotic_sequence[i] * 4) # Scale chaotic value
8          nucleotide = encoded_data[i]
9          new_nucleotide = (ord(nucleotide) + chaotic_value) % 4 # Shift nucleotide
10        encrypted_data.append(chr(new_nucleotide))
11   return encrypted_data  # OUTPUT
```

Algorithm 3.5: Decoding Operation

```
1 INPUT: encoded_data
2 def DecodeData(encoded_data):
3   nucleotide_to_pixel = {'A': 0, 'C': 1, 'G': 2, 'T': 3}
4 decoded_data = []
5    for nucleotide in encoded_data:
6          pixel_value = nucleotide_to_pixel[nucleotide]  # Map nucleotide back to pixel value
7         decoded_data.append(pixel_value)
8 return decoded_data  # OUTPUT
```

Algorithm 3.6: Chaotic Decryption Function

```
1 INPUT: encrypted_data, x_0, λ
2 def ChaoticDecryption(encrypted_data, x_0, λ):
3 iterations = len(encrypted_data)
4 chaotic_sequence = LogisticMap(x_0, λ, iterations)  # Generate chaotic sequence
5  decrypted_data = []
6  for i in range(iterations):
7      chaotic_value = int(chaotic_sequence[i] * 4)  # Scale chaotic value
8       encrypted_nucleotide = encrypted_data[i]
9      original_nucleotide = (ord(encrypted_nucleotide) - chaotic_value) % 4  # Reverse shift
10       decrypted_data.append(chr(original_nucleotide))
 11   return decrypted_data  # OUTPUT
```

4.0 DEPLOYMENT TO WEB

The suggested solution involves implementing a web application using the Flask framework to test and evaluate the new image encryption method that combines chaos theory and a modified hash function. The interface, built on Flask, enables user interaction and provides a smooth process for uploading images to be encrypted. The backend utilises sophisticated chaos-based encryption techniques, leveraging sequence operations to bolster security. Concurrently, a modified hash function designed exclusively for image data is applied, enhancing the level of cryptographic security. During deployment, the system is subjected to thorough testing to assess its overall performance, security capabilities, and user experience. The web application thoroughly evaluates a range of scenarios, encompassing different image types and encryption parameters, to guarantee the strength and dependability of the chaos-based image encryption and revised hash function. The web interface can be seen in Figure 3.1-Figure 3.4.

Figure 3.1: Web interface of the chaos-based image encryption system

Figure 3.2: Web interface of the chaos-based image decryption system

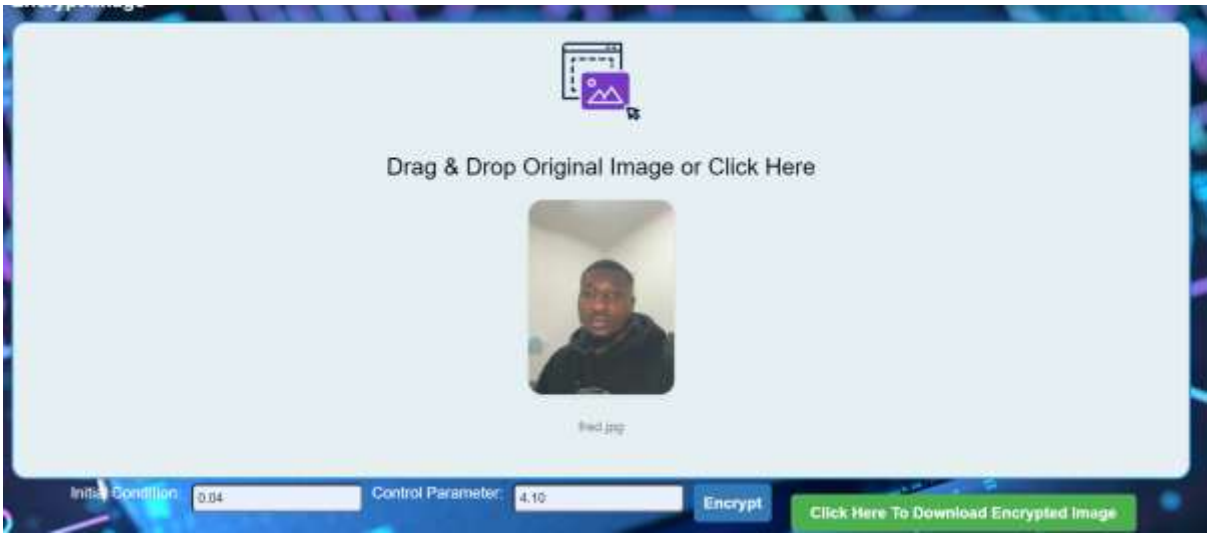

Figure 3.3: Original Image to be Encrypted

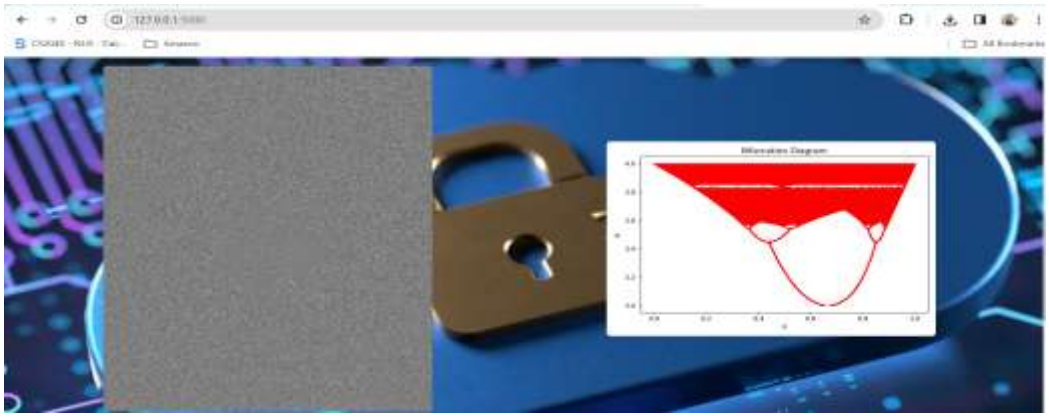

Figure 3.4: Encrypted image and Bifurcation diagram

5.0 CONCLUSION

The proposed image encryption system demonstrates a significant advancement in the field of digital image security through its integration of chaos theory and a customized hash function. The analysis of histogram data reveals that the encryption process effectively obscures the original image's pixel intensity distribution, producing a more uniform intensity histogram that enhances data security. The Adjacent Pixel Autocorrelation shows the effectiveness of the encryption by showing a substantial reduction in spatial redundancy, which minimizes the potential for pattern recognition and subsequent attacks. The sensitivity of the encryption key is rigorously tested, confirming that minor changes in the key produce markedly different encrypted outputs, thus safeguarding against brute-force attacks. The redefined hash function, tailored to handle image-specific attributes, adds an extra layer of security by generating unique hash values for different images. The web-based implementation of the encryption system, coupled with its thorough testing and evaluation, validates the practicality and robustness of the proposed method in real-world scenarios. Overall, the proposed system not only achieves its goal of providing a secure and effective image encryption solution but also sets a new standard in terms of incorporating advanced cryptographic techniques into image security.