A Multi-PRNG Based Framework for Medical Image Security

Thesis submitted in Partial fulfilment of the Requirements for the Degree

Of

B. Tech. in Computer Science and Engineering

Submitted by

Gurupada Majhi
University Roll No: 10000121040
Registration No: 211000100110040 of 2021-2022

Under the supervision of

Mr. Mihir Sing
Assistant Professor
Dept. of CSE, MAKAUT, WB

Dr. Koushik Majumder
Associate Professor
Dept. of CSE, MAKAUT, WB

Mr. Santanu Chatterjee
Assistant Professor
Dept. of CSE, MAKAUT, WB

Dr. Saikat Basu
Associate Professor
Dept. of CSE, MAKAUT, WB

Maulana Abul Kalam Azad University of Technology, West Bengal,
Department of computer Science and Engineering Simhat, Haringhata
Nadia-741249, West Bengal

July - 2025

CERTIFICATE OF ORIGINALITY

a. I hereby declare that this Thesis entitled “A Multi-PRNG Based Framework for Medical Image Security” contains literature survey and original research work by me, as part of my Degree of Bachelor of Technology (Computer Science and Engineering).

b. All information has been obtained and presented in accordance with academic rules and ethical conduct.

c. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

d. The research results presented in the thesis have not been included in any other paper submitted for the award of any degree to any other University or Institute.

Date: 07.07.2025 …….

Signature of the Student

CERTIFICATE OF APPROVAL

This is to certify that the Dissertation Thesis entitled, **"A Multi-PRNG Based Framework for Medical Image Security** "submitted **by Gurupada Majhi,** University Roll No:10000121040, Registration No: 211000100110040 of 2021-2022 to MAKAUT, WB, India, is a record of bona-fide Project work carried out by him under my/our supervision and guidance and is worthy of consideration for the award of the degree of **Bachelor of Technology in Computer Science and Engineering**.

……
Signature of the Project Supervisor
Mr. Mihir Sing
Assistant Professor,
Dept. of CSE, MAKAUT, WB

……
Signature of the Project Supervisor
Dr. Koushik Majumder
Associate Professor,
Dept. of CSE, MAKAUT, WB

……
Signature of the Project Supervisor
Mr. Santanu Chatterjee
Assistant Professor,
Dept. of CSE, MAKAUT, WB

……
Signature of the project supervisor
Dr. Saikat Basu
Associate Professor,
Dept. of CSE, MAKAUT, WB

……
Signature of the Head of the Dept.
Dept. of CSE, MAKAUT, WB

ACKNOWLEDGEMENTS

It is indeed a matter of great pleasure and proud privilege to present this project of " **A Multi-PRNG Based Framework for Medical Image Security** ". The completion of project work is a milestone in the student's life and its execution is inevitable in the hands of guide. I am highly indebted to my Guides, **Mr. Mihir Sing**, **Mr. Santanu Chatterjee**, **Dr. Koushik Majumder** and **Dr. Saikat Basu,** whose invaluable guidance and wish to record my deep sense of gratitude and appreciation for giving form and substance to this report. It is due to their enduring efforts, patience, and enthusiasm, which has given a sense of direction and purposefulness to this project and ultimately made it a success.

I would also like to extend my heartfelt gratitude towards the entire CSE department for allowing me to proceed with the project. Last but not the least, I wish to thank my parents, the non-teaching staffs of our department and my friends who have helped me all the time in one way or the other for the completion of this project.

…….

Gurupada Majhi Date: 07.07.2025

University Roll No: 10000121040

Registration No: 21100100110040 of 2021-2022

B. Tech, CSE, MAKAUT, WB

MAULANA ABUL KALAM AZAD
UNIVERSITY OF TECHNOLOGY,
WEST BENGAL

CERTIFICATE OF APPROVAL*

The forgoing thesis is hereby approved as a creditable study of an engineering subject and presented in a manner satisfactory to warrant acceptance as prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn there in but approve the thesis only for which it is submitted.

Committee on final examination for the evaluation of the thesis

……

Signature of the Examiner

……. ……. …….
Signature of the Supervisor Signature of the Supervisor

……. ……. …….
Signature of the Supervisor Signature of the Supervisor

Only in the case the thesis is approved

TABLE OF CONTENTS

List of Tables

List of Figures

A Multi-PRNG Based Framework for Medical Image Security

Abstract

In today's digital world, keeping medical data secure is very important because online platforms are vulnerable to cyber threats. Many hospitals and medical centers store patient records digitally, which increases the risk of unauthorized access and data manipulation. This paper presents a strong security system to protect medical images using different Pseudo Random Number Generators (PRNGs) like Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSMFT), Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG). The encryption method includes pixel scrambling with the Arnold Cat Map and secure operations like XOR substitution, transposition, and combination techniques. The system is tested against common security threats, including Crop Attacks and Salt-and-Pepper Noise Attacks. To measure its effectiveness, we use security metrics such as Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Unified Average Changing Intensity (UACI), Entropy, and Number of Pixels Change Rate (NPCR). Histogram plots are used to visually analyze security performance. The results show that this method provides strong protection and confidentiality for medical images, making it a reliable solution for modern healthcare security. (the source code is available on Github.)

Keywords

Medical Image Security, Cryptography, PRNG Randomization, Arnold Cat Map, XOR Substitution, Cybersecurity, Encryption, Decryption, Crop attack and Salt Noise attack, Entropy, NPCR, UACI, PSNR, SSIM, Histogram

1. Introduction

Protecting confidential information in the modern digital age is critically important, particularly within medical systems. Medical images tend to include confidential patient information that needs to be secured. Encrypting the information is one of the finest means of securing the information, which remaps the original image into a jumbled version (referred to as ciphertext) that only approved users can interpret. This paper examines medical image security with various categories of Pseudo-Random Number Generators (PRNGs), including Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSFMT), Combined Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG). These PRNGs are employed to produce random sequences to shuffle and conceal the image data. Our approach involves some steps: block-wise division of the image, shuffling of pixels, and transposition, substitution, and data combination using XOR operations. We also use methods such as the Arnold Cat Map and XOR to encrypt the data even more. We analyze our system against typical attacks such as crop attacks and salt-and-pepper noise attacks. We use quality and security measures such as PSNR, SSIM, UACI, ENTROPY, and NPCR to analyze the performance of the encryption. We also observe changes in histograms to analyze how much the image is being modified. The aim of this task is to develop a secure and robust method to encrypt medical images so that they remain confidential and only visible to the appropriate individual.

1.1. Problem Statement:

Current image encryption methods typically follow a fixed approach for all image types, ignoring the variation in image content and structure. This can lead to weaker security and a higher risk of unauthorized access or data manipulation. In addition, the strength of encryption heavily relies on the effectiveness of the random number generation process. Although several PRNGs exist, there is no standard method to determine which PRNG is most suitable for a given medical image. As a result, there is a need for a dynamic system that can identify and apply the most effective PRNG and encryption model based on the specific features of the image to be secured.

1.2. Motivation and Objective:

The main reason for this research is to protect medical images in today's digital healthcare world, where images are shared and stored online and can easily be seen or changed by people who shouldn't have access. Many current encryption methods don't fully protect these images because they only use one or two random number generators, which doesn't give them enough flexibility to handle different image types. They also stick to one way of encrypting the image and don't adapt to what the image really needs. On top of that, most of them don't test how their systems would handle real problems like parts of the image being cropped out or noise being added—issues that are common

in medical images. Another big problem is that they don't always make sure the original image can be perfectly recovered, which is very important for doctors.
In our work, we focus on fixing these issues by using four advanced random number generators: Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSMFT), Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG). We use these random number generators with different encryption techniques: swapping pixel positions (called transposition), changing the pixel values (substitution), or using both together. Our system doesn't just stick to one method—it chooses the best option based on the actual image. We also test our system's strength against two real-world problems: cropped images and images with added noise. Finally, we make sure the original images can be fully recovered without losing any detail, which is essential for medical use. The main goal of our work is to build a strong, secure, and flexible encryption system that keeps medical images safe and accurate.

1.3. Scope and Limitations

This study is limited to grayscale medical images and symmetric key encryption methods. The image is divided into smaller blocks, and encryption is applied to each block using the selected PRNG and cryptographic model. The Arnold Cat Map is used for pixel permutation, and XOR-based substitution is used to alter pixel values. To check how strong the system is, we tested it by adding crop and salt-and-pepper noise attacks. One limitation of the system is its reliance on the same PIN or seed for both encryption and decryption, which may present key management challenges. Another limitation is that the analysis focuses mainly on statistical evaluation and does not include resistance testing against advanced cryptographic attacks. Nonetheless, the proposed system offers a flexible and content-aware solution for enhancing the confidentiality of medical images and provides a strong foundation for future improvements in secure image transmission and storage.

2. Literature Survey

2.1: [1] Image Encryption using Block Permutation and XOR Operation

Sivakumar and Devi (2017) proposed a simple approach using block permutation and XOR operations, using a single PRNG (Lagged Fibonacci Generator). This method is efficient, easy to implement, and resistant to statistical attacks, making it ideal for systems with limited resources. However, it lacks flexibility for different image types or advanced encryption needs. On the other hand, our paper dynamic framework that tests four PRNGs (MT, dSFMT, MRG, and MLFG) and three encryption models (transposition, substitution, and a combination of both) to determine the best option for encrypting medical grayscale images. By implementing metrics like PSNR and SSIM into a figure-of-merit, their method ensures high security and confidentiality for each image. While Sivakumar and Devi prioritize simplicity and efficiency, our research paper focuses on advanced adaptability and robustness, making their approach suitable for high-security medical imaging applications.

2.2: [2] Image Encryption Techniques Using Pseudo Random Number Generators

Banthia and Tiwari (2013) focuses on enhancing image confidentiality through encryption using two specific Pseudo Random Number Generators (PRNGs): the Linear Congruential Generator (LCG) and the Chaotic Logistic Map. Their methodology involves generating pseudo-random sequences from these PRNGs to shuffle the pixels of images, effectively obscuring their original arrangement. They also apply XOR masking to combine pixel values with the random sequences, further enhancing security. To evaluate the effectiveness of their encryption techniques, they use several metrics, including entropy, cross-correlation, Mean Square Error (MSE), and Peak Signal to Noise Ratio (PSNR). Our research paper emphasizes the confidentiality of medical images through a dynamic smart random preference approach, utilizing a combination of transposition and substitution methods for encryption. We begin by selecting a secret password as a seed for four different PRNGs: Mersenne Twister, SIMD-oriented Fast Mersenne Twister, Combined Multiple Recursive, and Multiplicative Lagged Fibonacci. These PRNGs generate random sequences that permute the pixels of medical images, effectively scrambling their original structure. We apply XOR operations to further alter pixel values and strengthen encryption. The effectiveness of our methods is evaluated using Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index (SSIM), which assess the quality and usability of the encrypted images. specially, we do not utilize the Chaotic Logistic Map in our research, as we focus on optimizing encryption tailored to the specific characteristics of medical images while ensuring data confidentiality.

2.3: [3] Varying PRNG to improve image cryptography implementation

Gutub, A., Al-Roithy, B. 2021 proposed his paper analyzes six PRNGs across grayscale and RGB images using a fixed two-step encryption process (transposition and substitution) and evaluates broader security metrics, including entropy, NPCR, UACI, and correlation. While the current study emphasizes dynamic and image-specific optimization. This paper highlights the value of combining transposition and substitution for robust encryption. Our paper focuses on dynamically selecting the best PRNG and encryption model (transposition, substitution, or combination) for medical grayscale images, using PSNR and SSIM metrics to optimize confidentiality. It evaluates four PRNGs (Mersenne Twister, SIMD-oriented Fast Mersenne Twister, Combined Multiple Recursive Generator, and Multiplicative Lagged Fibonacci) tailoring the approach to specific image characteristics.

2.4: [4] Digital image scrambling based on sequence generation

Sarma, K., Lavanya, B. 2017. In contrast, the "Digital Image Scrambling Based on Sequence Generation" study introduces a scrambling algorithm that disrupts pixel positions and values using a mathematically generated sequence derived from two sensitive keys, ensuring high security but facing challenges with computational time.

Our research paper employs various Pseudo-Random Number Generators (PRNGs) and cryptographic techniques, specifically transposition and substitution, to encrypt medical grayscale images dynamically based on the specific image data. Both methodologies evaluate effectiveness through metrics like Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index (SSIM), with the first emphasizing adaptability to varying image data and the second highlighting the importance of key sensitivity for security.

2.5: [5] Remodeling Randomness Prioritization to Boost-up Security of RGB Image Encryption

Al-Roithy, B., Gutub, A. 2021 integrate advanced cryptographic techniques with machine learning algorithms. It focuses on using deep learning models to enhance the randomness of PRNGs, allowing for a dynamic adaptation of the encryption strategy based on image content. This methodology includes performance metrics related to computational efficiency and adaptability, extending beyond traditional statistical measures. Our paper utilizes various Pseudo Random Number Generators (PRNGs) to implement transposition and substitution techniques. It involves dividing images into blocks, shuffling pixel positions, and applying XOR operations for pixel value substitution. The effectiveness of different PRNGs is evaluated using statistical measures like Peak Signal to Noise Ratio (PSNR) and entropy.

3. Gap Analysis: -

Both our paper and the earlier work by Adnan Gutub paper find that older ways of encrypting medical images have important problems. Most of the older methods only use one or two types of random number generators (PRNGs). They also stick to just one encryption type—either swapping pixel positions or changing pixel values—without thinking about what medical images really need. These methods do not test how well they handle real attacks, like cutting parts of the image or adding noise. Also, they use only one test, like PSNR or entropy, which does not show the full picture. Both our paper and Gutu's paper try to fix these problems. We both use four PRNGs (MT, dSMFT, MRG, MLFG) and mix swapping and changing of pixel values to protect the images. But our paper goes further. We actually test real attacks (like cropping and noise) and use extra tests (like NPCR, UACI, and histograms) to see how well our system works. We also make sure that after decryption, the original medical images can be seen perfectly again, which is very important for doctors. In contrast, the older paper mainly compares the four PRNGs and how well they work together, without testing real attacks or checking if the images can be fully recovered. So, while both papers are better than older methods, our paper does more to make sure medical images stay safe and clear in real life.

4. Comparison analysis Table: -

Table 1: Comparison Table

Sl. No	Paper & Authors	PRNGs Used	Encryption Method	Metrics Used	Strengths	Limitations
1	Sivakumar & Devi (2017) *Image Encryption using Block Permutation and XOR Operation*	Lagged Fibonacci Generator (LFG)	Block permutation + XOR	Basic statistical resistance	Fast and efficient, suitable for low-resource systems	Limited flexibility, not suitable for high-security medical imaging
2	Banthia & Tiwari (2013) *Image Encryption Techniques Using PRNGs*	LCG, Chaotic Logistic Map	Pixel shuffling + XOR masking	Entropy, MSE, PSNR, Cross-correlation	Enhances randomness and obscures structure	Not designed for medical images; lacks SSIM analysis
3	Gutub & Al-Roithy (2021) *Varying PRNG to Improve Image Cryptography*	Six PRNGs (not detailed)	Fixed 2-step: Transposition + Substitution	Entropy, NPCR, UACI, Correlation	Broad PRNG evaluation for grayscale & RGB images	Static model; lacks dynamic adaptability for specific image types
4	Sarma & Lavanya (2017) *Digital Image Scrambling Based on Sequence Generation*	Key-sensitive mathematical sequence	Scrambling (pixel position and value)	PSNR	Strong key sensitivity, secure scrambling	Slower computation, less adaptable to different image types
5	Al-Roithy & Gutub (2021) *Remodelling Randomness Prioritization for RGB Encryption*	ML-enhanced PRNGs	ML-based dynamic encryption	Adaptability, computation time	Uses ML to enhance randomness and adaptability	Targeted at RGB, not grayscale; dependent on external models
6	Our paper "A Multi-PRNG Based Framework for Medical Image Security"	Mersenne Twister (MT), dSFMT, MRG, MLFG	Transposition, Substitution, Combination + XOR	PSNR, SSIM, Entropy	Dynamic PRNG and method selection; high confidentiality tailored to grayscale	More complex to implement; no ML-based optimization (yet)

5. Theoretical Framework

To describe an advanced methodology for securing medical gray-scale imagery, there is a need to superimpose or integrate the truly random set of procedures with the secure reversible transformations. Therefore, the proposed system synergizes four widely acknowledged PRNGs with the one deterministic but chaotic transformation of Arnold Cat Map. Each of these is specifically picked to maximize image-private transformation with guaranteed perfect reversibility and maximizing the image's cryptographic resistance against attacks of the statistical kind or differential kind.

5.1 Pseudo-Random Number Generators (PRNGs): In image encryption, the use of PRNGs is quite essential, as unpredictable sequences generated from these PRNGs help in pixel permutation and substitution. The four options available are a Mersenne Twister (MT), a SIMD-oriented Fast Mersenne Twister (dSFMT), a Combined Multiple Recursive Generator (MRG), and a Multiplicative Lagged Fibonacci Generator (MLFG). The availability of many alternatives hence offers great flexibility and allows a dynamic adaptation to the structure of a particular image.

5.2 Mersenne Twister (MT): MT has the reputation of an extremely huge period $2^{19937}-1$, so that MT's output can never repeat for such a massive number of iterations. It generates top-notch random numbers with impressive uniform distribution and efficiently generates random sequences for large amounts of data. Though it cannot be used by itself in any cryptographic system, it serves well as a basis for hybrid encryption implementations where it supports block permutation and value substitution with minimal overhead.

5.3 SIMD-oriented Fast Mersenne Twister (dSMFT): This is a faster version of the Mersenne Twister. It uses SIMD hardware (which can do many tasks at once) to quickly generate random numbers, while also using less memory. Although dSMFT retains MT's statistical properties, the speed approach it employs translates favorably, particularly if one is working with high-resolution medical images or working in a resource-limited environment. While dSMFT preserves MT's statistical qualities, the speed strategy it utilizes translates beneficially, especially when one is involved with high-resolution medical images or operating within a resource-constrained environment.

5.4 Combination Multiple Recursive Generator (MRG): A Combined MRG integrates numerous linear recursive sequences to form a long-period random sequence of higher complexity. It uses recursive relations with different moduli and coefficients with the intent to augment the independence and randomness of its outputs. MRGs pass stringent tests for randomness and resist numerous statistical attacks. They have thusly been used for simulations and applications demanding a high degree of integrity in randomization, including Monte Carlo simulations and financial modeling. MRG stream ciphers are chosen for this encryption system because of the two-layer permutation and substitution operations that provide complex non-repetitive key streams, thereby increasing resistance to brute-force and chosen-plaintext attacks.

5.5. Multiplicative Lagged Fibonacci Generator (MLFG): This is a slight variation of the Fibonacci series, where a number is the product of two numbers which are lagged by a pre-determined time and then taken modulo of a prime number or a power-of-two. Due to the afore-stated multiplication, the generated series has a non-linear character, which gives it a certain degree of unpredictability, so adjacent values have low correlation. Hence, it is useful in cryptographic applications. Efficiency and low memory footprint make MLFG suitable for embedded medical systems as well as IoT-based healthcare devices. When used as the key-stream for an XOR-based substitution cipher, the MLFG output stream mostly provides pixel-level distortion, thereby minimizing the chances for a potential attacker to reverse-engineer or statistically estimate the original image. Each of these PRNGs is initialized with a PIN-based seed in the proposed framework, thereby ensuring a user-specific repeatable key stream generation. Their independent randomness is then tested based on encryption performance parameters like PSNR, SSIM, entropy, NPCR, and UACI, to select one dynamically for each medical image.

5.6. Arnold Cat Map (ACM): named after the Russian mathematician Vladimir Arnold. It demonstrates how a systematic system can degrade to seeming chaos without in any way being indeterministic or irreversible. In image encryption, the Arnold Cat Map is applied to randomize pixel locations within a square picture or block so it becomes very effective for confusion in cryptographic systems. The transformation moves every pixel at positions (i,j) to a new position (X_{new},Y_{new}) according to the following equations:

$$\mathbf{x}_{new}=(i+j) \bmod N \ldots\ldots(1)$$

$$Y_{new}=(i+2j) \bmod N \ldots..(2)$$

where N is the square image's width (or height). The wrap-around provided by modular arithmetic maintains image size while keeping the pixels within the image boundary. Despite its seemingly random visual output, the map can be inverted after a finite number of iterations, with the original image being reconstructed exactly. This makes the Arnold Cat Map a perfect utility for reversible pixel scrambling in secure image encryption, particularly in conjunction with other methods such as PRNG-based substitution or block permutation.

6. Proposed Methodology: -

The process starts with A grayscale medical image is first loaded and resized to 256×256 pixels to keep the size consistent for all tests. This resized image is then split into small blocks of 16×16 pixels to allow each part of the image to be encrypted separately, which helps improve security. Next, a Personal Identification Number (PIN) chosen by the user is used as a seed to initialize the random number generators. The PIN is converted into a number using byte encoding so it can work properly with the encryption system. The method uses four different Pseudo-Random Number Generators (PRNGs): Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSMFT), Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG). Each one is set up using the seed and slightly different settings to give varied results. These PRNGs can generate random numbers and shuffle lists, which are important for encryption. In the block permutation step, the system creates a list of block positions and uses the PRNG to randomly shuffle this list. The blocks are then rearranged based on the new order, which helps scramble the image structure and make it harder to understand without the key. After shuffling, the next step is XOR substitution. Each block is turned into a one-dimensional array, and a random set of bytes is created using the PRNG. A bitwise XOR operation is done between the block data and the random bytes, which changes the pixel values in a secure way. For encryption, both the permutation and XOR substitution steps are applied. Along with the encrypted blocks, the system saves the order of shuffled blocks and the random values used for the XOR step. These are necessary for decrypting the image later. During decryption, the XOR operation is undone using the saved random values, and the blocks are then put back into their original order using the saved permutation indices. This restores the original block arrangement. In the final step, the decrypted blocks are combined to rebuild the full image. An empty image is created, and each block is placed back where it originally belonged. Finally, this entire process—encryption and decryption—is repeated for all four PRNGs. The encrypted images are saved for testing how well the method protects.

6.1. Encryption And Decryption

6.1.1. Algorithm for Arnold Cat map

```
Function arnoldCatMap(image, iterations)
1.  H, W ← shape(image)                // Get image dimensions
2.  if H ≠ W then
3.      raise error "Image must be square"
4.  transformed_image ← copy(image)     // Initialize transformed image
5.  for iter ← 1 to iterations do
6.      new_image ← zeros_like(transformed_image)
7.      for y ← 0 to H – 1 do
8.          for x ← 0 to W – 1 do
9.              new_x ← (x + y) mod W
10.             new_y ← (x + 2 × y) mod H
11.             new_image[new_y, new_x] ← transformed_image[y, x]
12.     transformed_image ← new_image   // Update image for next iteration
13. return transformed_image
```

The Arnold Cat Map algorithm is a method used to scramble the pixels of a square image in a reversible way, making it useful for image encryption. The process begins by checking if the image is square, because this technique only works when the height and width of the image are equal. A copy of the image is then made so that the original is not changed. The algorithm applies a series of transformations over a set number of iterations. In each iteration, it calculates new positions for every pixel using a specific formula: the new x-coordinate is calculated as (x + y) mod width, and the new y-coordinate as (x + 2*y) mod height. These new positions ensure that the pixels are shuffled within the image boundaries. After each iteration, the newly scrambled image becomes the input for the next one. Finally, the fully scrambled image is returned. This transformation hides the original structure of the image, and

because it is reversible, the image can be restored by applying the inverse transformation with the same number of iterations.

6.1.2. Algorithm for Encryption:-

```
EncryptImage (image, pin, block_size)
// Encrypts and decrypts a grayscale medical image using 4 PRNGs
Resize image to (256 × 256)
M, N ← shape(image)
Divide image into non-overlapping blocks of size block_size × block_size
n_blocks ← (M / block_size, N / block_size)
blocks ← list of all image blocks
seed ← int.from_bytes(pin.encode(), 'little')
Initialize 4 PRNGs:
    prng_mt ← MersenneTwister(seed + 1)
    prng_dsmft ← dSMFT(seed + 2)
    prng_mrg ← MRG(seed + 3)
    prng_mlfg ← MLFG(seed + 4)
For each PRNG in [prng_mt, prng_dsmft, prng_mrg, prng_mlfg]:
    // Encryption
    indices ← [0 … len(blocks)-1]
    Shuffle indices using PRNG
    permuted_blocks ← blocks[shuffled indices]
    For each block in permuted_blocks:
        rand_bytes ← PRNG.generate_bytes(len(block))
        encrypted_block ← block XOR rand_bytes
        Store encrypted_block and rand_bytes
    Reassemble encrypted image from encrypted_blocks
    Save encrypted image
```

6.1.3 Algorithm Of Decryption

```
generateKey(matrix) // generates decryption 'key'
M, N ← shape(matrix)
Num ← randNum() // MUST be same as used in encryption (use same seed or fixed value)
row ← floor(Num / N)
col ← Num % N
for i ← 0 to 4:
    if col + i < N:
        Key ← matrix[row, col + i]
    else:
        Key ← matrix[(row + 1) % M, (col + i) % N]
key ← binary(Key) // convert each key element to binary or generate one binary string
return key
```

The decryption key generation algorithm mirrors the encryption process to ensure the same key is reproduced. It selects five consecutive values from a known matrix based on a fixed or reproducible random number. The selected values are converted into a binary key, which is used for decrypting the encrypted data. To maintain accuracy, the same random number (or seed) and matrix used during encryption must be used during decryption. This ensures consistency in symmetric encryption systems where the same key is needed for both encryption and decryption.

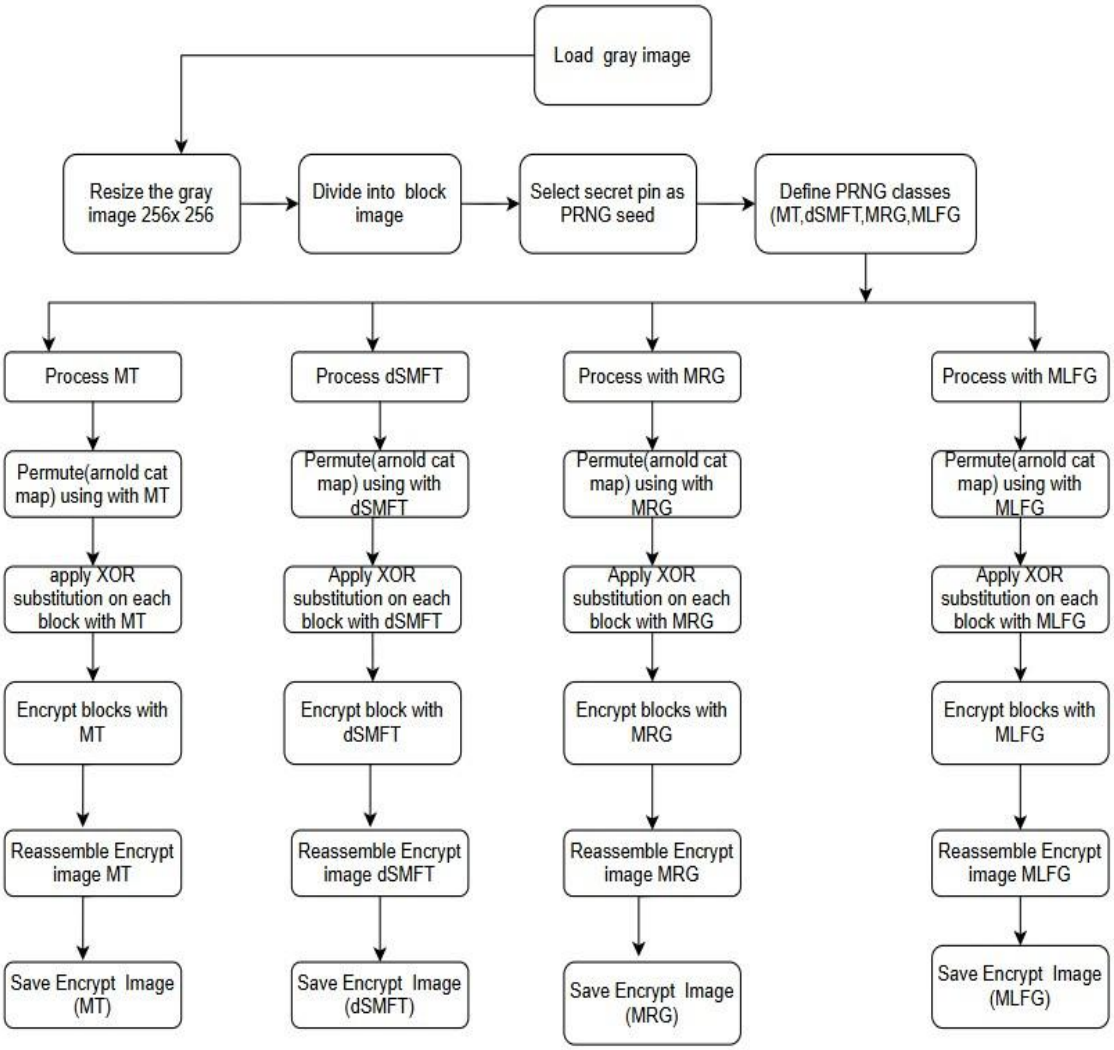


Fig. 1: Flow Chart of Encryption

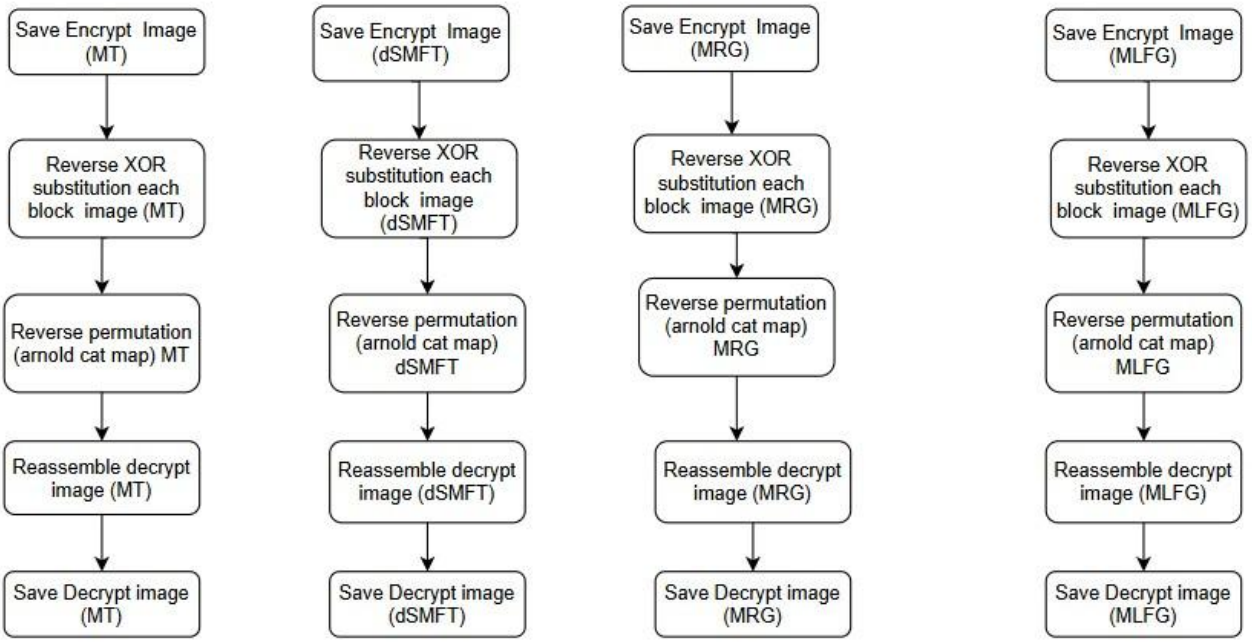


Fig. 2: Flow Chart of Decryption

6.2. Crop Attack: -

A crop attack happens when a part of the image is cut out, which changes the image's structure and can damage the data. In grayscale images, this removal causes changes in pixel patterns, which may affect how the image is stored and recovered. In colour images, cropping affects all three colour channels—red, green, and blue—making the image look incomplete and unbalanced. This kind of attack is used to test how strong an encryption method is when part of the image is missing. A good encryption system should make sure that no one can guess or rebuild the original image from the cropped parts, but it should still allow the remaining portion to be decrypted correct. Cropping affects image security by checking if the encryption can still protect the image when part of the data is missing. When a section of the image is cut out, it may break the pattern and structure of the encrypted file. This helps test how well the method can keep the image safe and stop anyone from trying to guess or rebuild the original image using the remaining parts. A good encryption method should not allow the missing part to be recovered without permission. At the same time, it should still be able to correctly open and show the part of the image that is left. This shows the system is secure and can handle some data loss.

Algorithm of Crop Attack:-

Encrypt Image(image, prng) // encrypts the input image

1. image ← resize(image, 256 × 256)
2. blocks ← divide Into Blocks(image, 16 × 16)
3. indices ← permute Indices (len(blocks), prng)
4. for i ← 0 to len (blocks) – 1:
5. shuffled Block ← blocks[indices[i]]
6. key ← generate Random Key(shuffled Block, prng)
7. cipher Block ← XOR(shuffled Block, key)
8. store cipher Block and key
9. encrypted Image ← reassemble Blocks(cipher Blocks)
10. encrypted Image ← Arnold Cat Map (encrypted Image, 20)
11. return encrypted Image, indices, keys

Crop Attack(image, top Left, bottom Right) // simulates cropping attack

12 for i ← top Left.y to bottom Right.y – 1:
13. for j ← top Left.x to bottom Right .x – 1:
14. image[i, j] ← 0
15. return image

Decrypt Image(cropped Image, indices, keys, prng) // decrypts the attacked image

16. image ← reverse Arnold Cat Map (cropped Image, 20)
17. blocks ← divide Into Blocks (image, 16 × 16)
18. for i ← 0 to len (blocks) – 1:
19. plain Block ← XOR(blocks[i], keys[i])
20. store plain Block
21. plain Blocks ← reverse Permutation (plain Blocks, indices)
22. decrypted Image ← reassemble Blocks(plain Blocks)
23. return decrypted Image
24. image ← load Gray Image('uterus.jpeg')
25. seed ← convert PIN('123456')
26. init PRNGs: mt, dsmft, mrg, mlfg with variants
27. for each prng in [mt, dsmft, mrg, mlfg]:
28. encrypted, indices, keys ← encrypt Image(image, prng)
29. cropped ← crop Attack(encrypted, (0, 0), (128, 128))
30. decrypted ← decrypt Image(cropped, indices, keys, prng)
31. save encrypted, decrypted images

In this work, a secure image encryption and decryption method is proposed for protecting medical grayscale images against crop attacks. The process begins by dividing a 256×256 grayscale image into smaller blocks of 16×16 pixels. These blocks are then randomly shuffled using pseudo-random number generators (PRNGs) such as Mersenne

Twister, dSMFT, MRG, and MLFG. A unique key is generated for each block using the selected PRNG and is used to encrypt the block through XOR substitution. The entire encrypted image is further scrambled using the Arnold Cat Map for increased security. A simulated crop attack is applied by removing a portion of the encrypted image, and the decryption process is then performed to assess the system's ability to recover the original image. Decryption involves reversing the Arnold transformation, applying the XOR operation with the original keys, and restoring the original block order. The effectiveness of this method is evaluated using standard metrics such as PSNR, SSIM, entropy, NPCR, and UACI to ensure high confidentiality, robustness, and accurate image recovery.

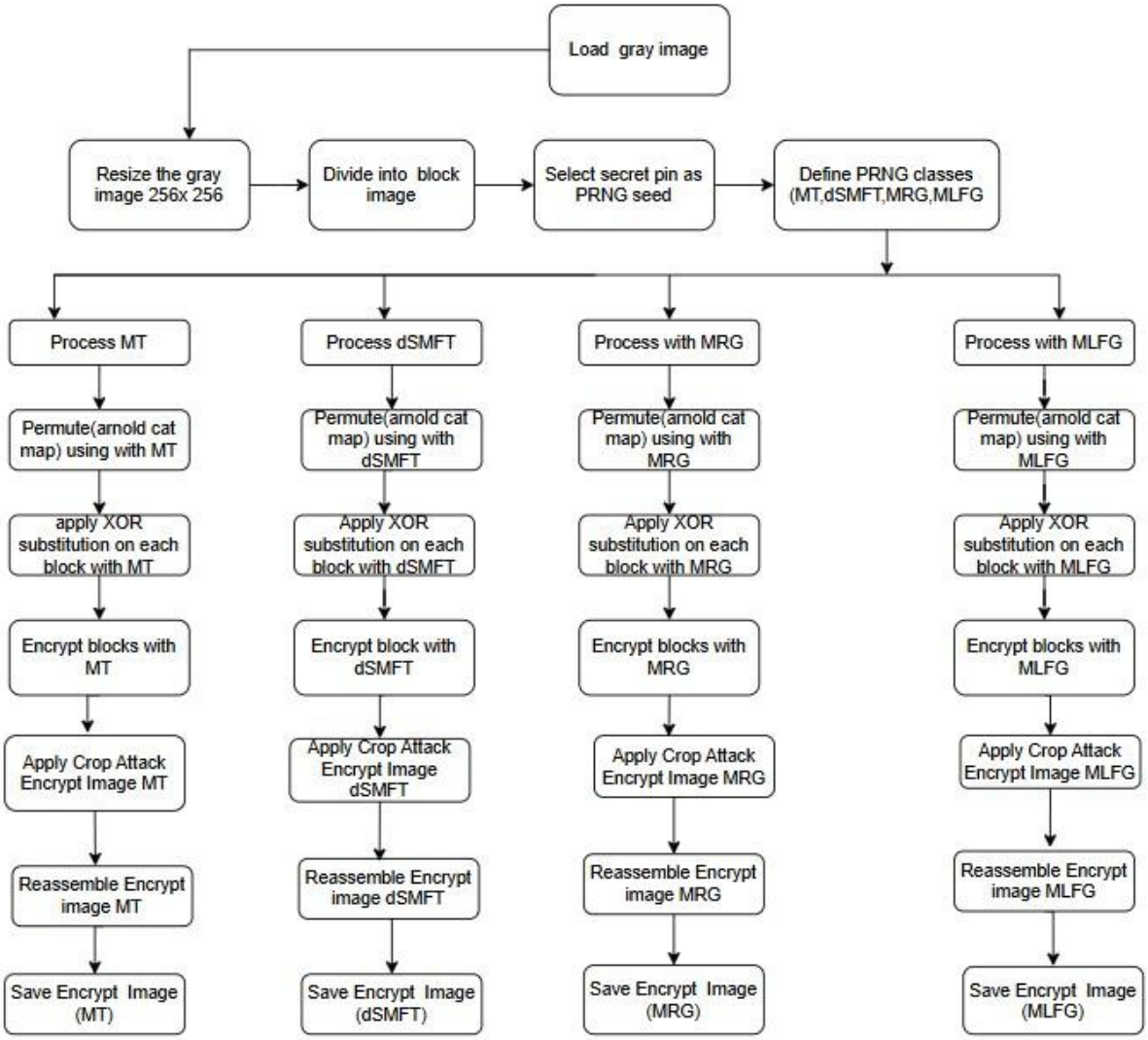


Fig. 3: **Cropped Attack Encryption**

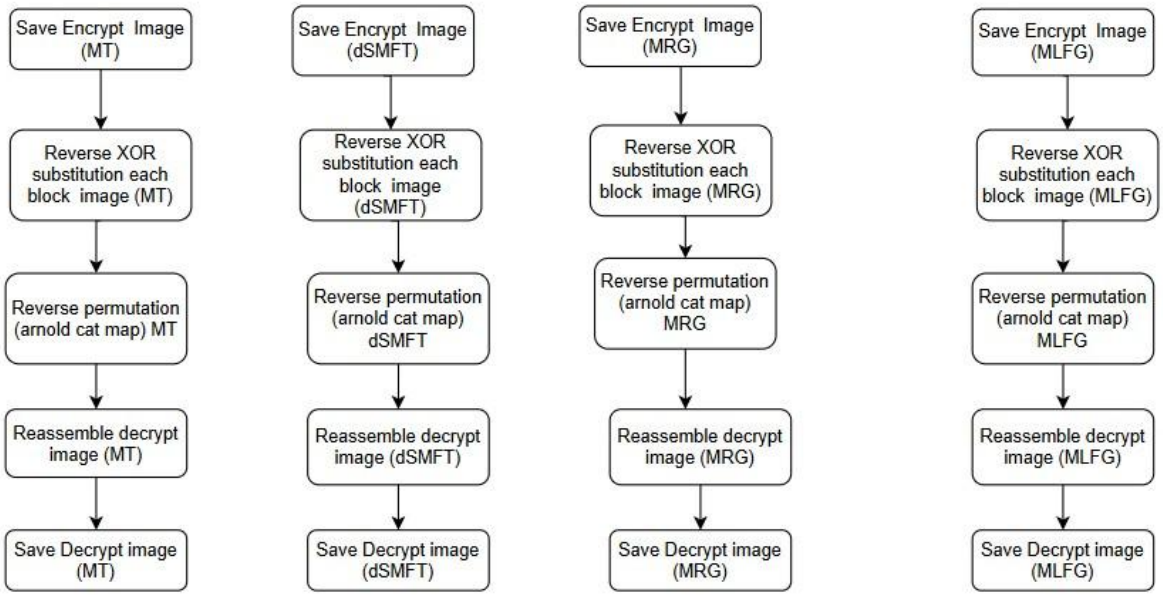


Fig. 4: Cropped Attack Decryption

6.3. Salt-and-Pepper Noise Attack

Salt-and-pepper noise is a type of image disturbance where some pixels are randomly changed to either pure white (salt) or pure black (pepper). These changes appear as tiny white and black dots scattered across the image, which can affect its overall appearance and make it harder to see details clearly.
In image encryption research, this kind of noise is used to check how well an encryption method can protect the image when it is affected by such errors. For medical grayscale images, which rely on clear details for analysis, salt-and-pepper noise can be very damaging. A good encryption system should keep the original image secure even when this noise is added, making sure that no useful information can be seen or guessed. This shows that the method is strong and can protect sensitive data even under noisy or damaged conditions.

Algorithm of Salt and Peper Noise Attack

1.The original image of size 256×256 pixels
2. The probability of adding salt noise (e.g., 0.02 or 2%)

3. Set the image height to 256
4. Set the image width to 256
5. Calculate the total number of salt noise pixels to add:
   6. Num Salt ← ceil (P_salt × height × width)
6. Adding Salt Noise:
   a. For each k from 1 to numSalt:
      i. Randomly pick a pixel position (i, j) within the image bounds:
         i ← random integer between 0 and height - 1
         j ← random integer between 0 and width - 1
      ii. Set that pixel to white:
         I_salt [i] [j] ← 255
   b. Return the image I_salt, which contains the original image plus the added salt noise.

Here, The salt noise attack introduces random bright white noise to an image to evaluate the resilience of an image encryption scheme. In this technique, a specific portion of the image's pixels, often set at around 2%, are randomly chosen and changed to the maximum pixel intensity, typically 255 in grayscale images. This random alteration simulates real-world disturbances, such as those caused by channel noise during image transmission. The procedure begins by calculating the exact number of pixels that need to be modified based on the defined noise level. Next, random pixel coordinates are generated and these pixels are assigned the highest possible intensity, creating a "salt"

noise effect that degrades the visual quality of the image. This noisy image is then used to assess the effectiveness of the decryption process, offering insights into the encryption method's ability to maintain image fidelity despite external noise. The outcome of this attack helps in evaluating the security and robustness of the proposed encryption technique in realistic scenarios.

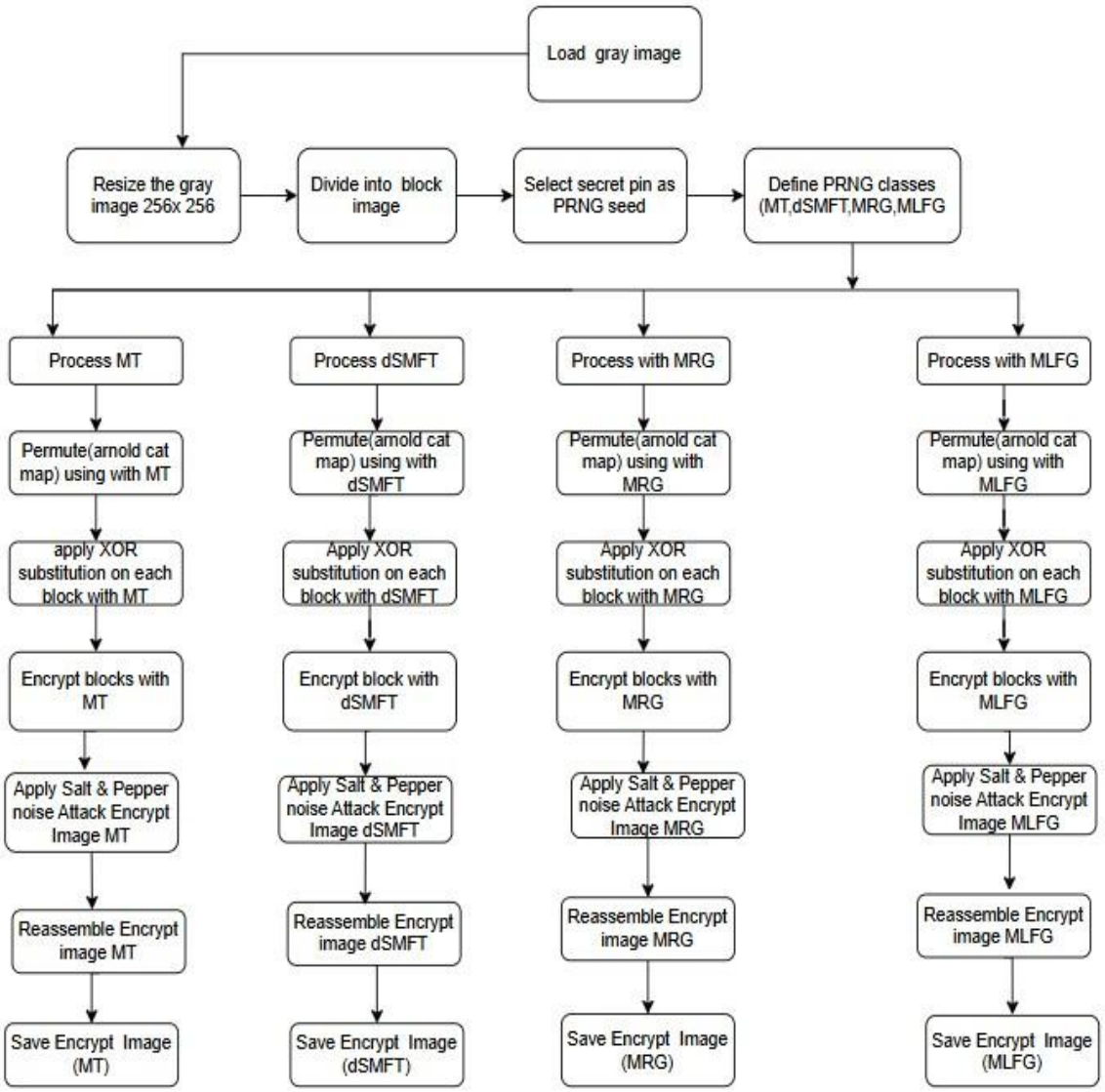


Fig. 5: Flow Chart: Salt & Pepper Noise Attack Encryption

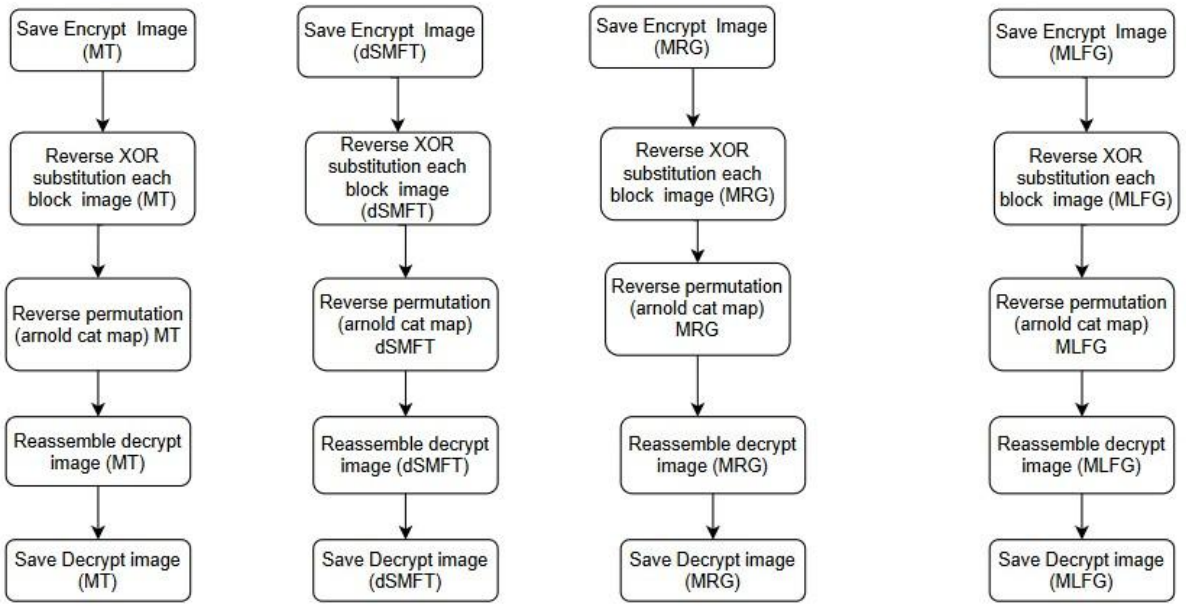
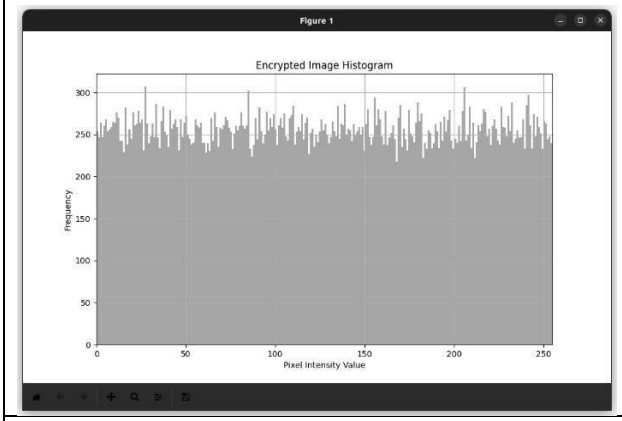


Fig. 6: Flow Chart: Salt & Pepper Noise Attack Decryption

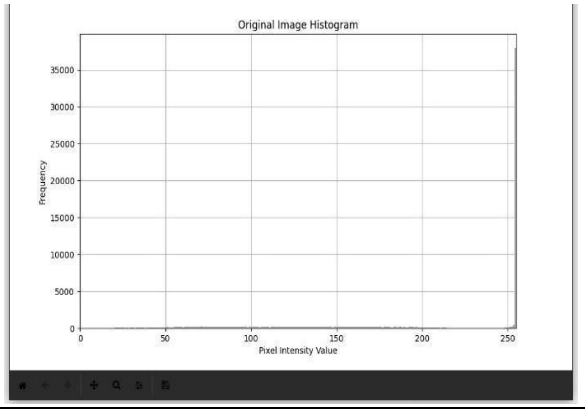
7. Image Quality and Security Evaluation Metrics:

7.1. Histogram Analysis, Entropy, SSIM, PSNR, NPCR, UACI, Global Entropy and Local Entropy Analysis.

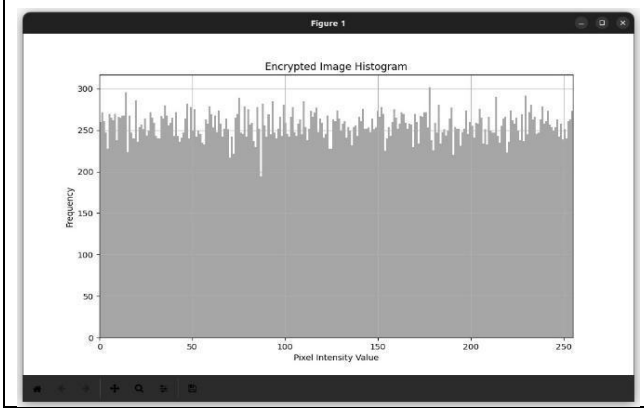
(A)Encrypted Brain side Histogram



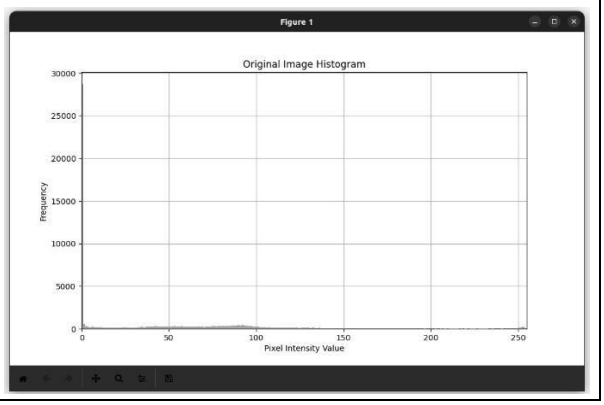
(B)Original **Brain side** Histogram



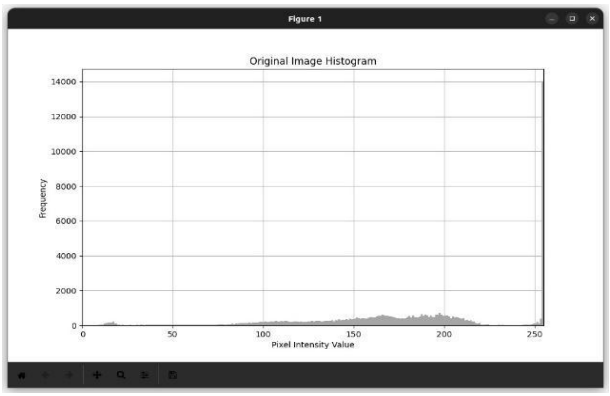
(C) Encrypted Brain Top Histogram



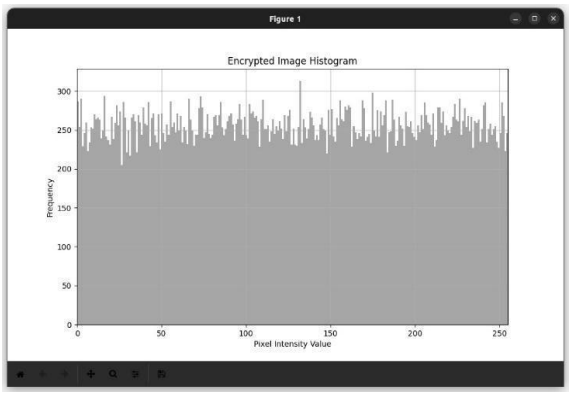
(D)Original **Brain Top** Histogram



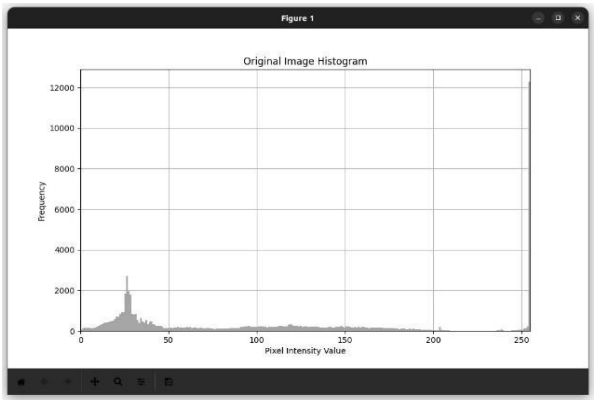
(E)Original chest Histogram



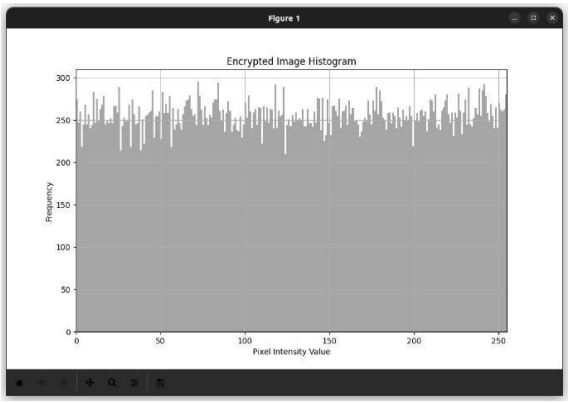
(F)Encrypted Chest Histogram



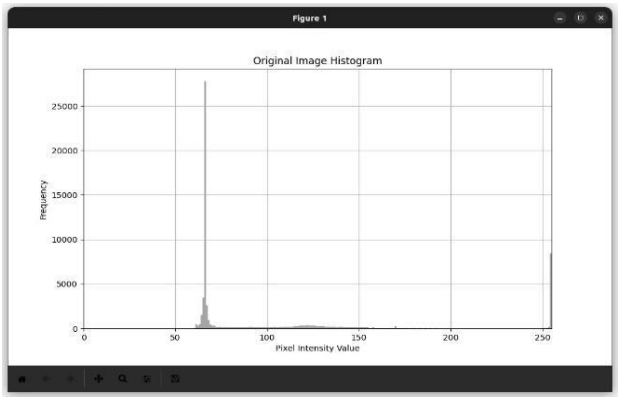
(G)Fingers original Histogram



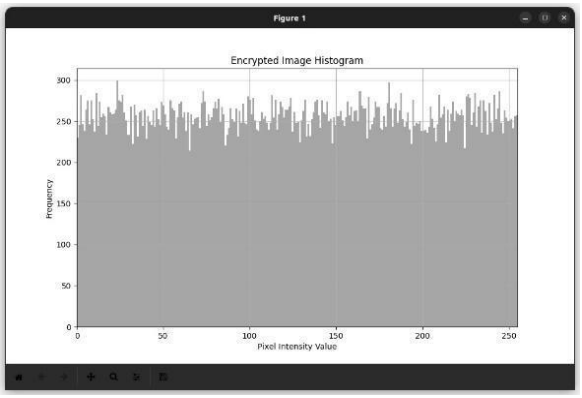
(H) Fingers encrypted Histogram



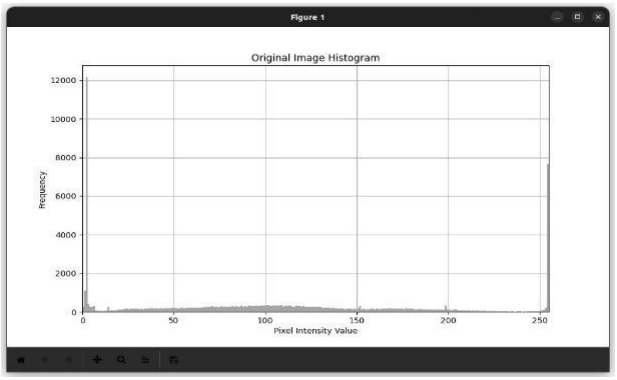
(I) legs Original Histogram



(J) legs encrypted Histogram



(K) Uterus Original Histogram



(L)Uterus encrypted Histogram

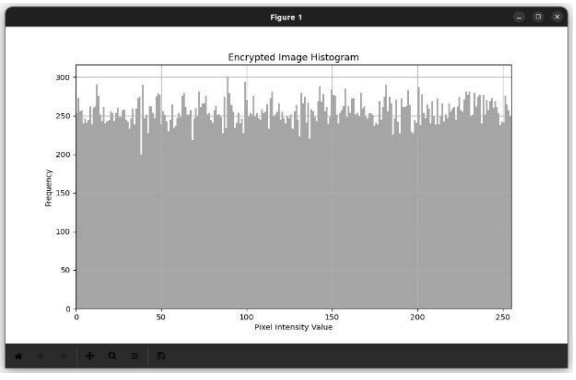


Fig. 7: Histogram Analysis (Brain Side and Top, chest, Fingers, legs, Uterus)

The above figure (7) shows the histogram of the original grayscale medical image, which helps us understand how the pixel brightness levels are spread across the image. On the x-axis, we see pixel intensity values from 0 (black) to 255 (white), and on the y-axis, we see how often each brightness level appears in the image.
From the graph, we can observe that most of the pixels are grouped at the two ends—very dark or very bright—with peaks around intensity values close to 0 and 255. In the middle range, the number of pixels is quite low, showing that mid-tone values are not common in this image. This pattern is typical in medical images like X-rays or MRIs, where different body parts and backgrounds create strong contrast. This type of uneven histogram suggests that the image has visible patterns and structures, which could be a security risk if the image is not encrypted. An attacker might use these patterns to guess or extract information from the image using statistical methods. That is why it is important for encryption methods to flatten or equalize the histogram in the encrypted version. A more uniform histogram makes it much harder to detect any useful information from the image. In short, the original image's histogram shows clear patterns in brightness levels, especially at the dark and bright ends. This highlights the importance of applying strong encryption to hide these patterns and protect sensitive medical data effectively.

7.2. Structural Similarity Index (SSIM)

Definition: Assesses the structural similarity between the original and decrypted images.

$$SSIM(x,y)=\frac{(2\mu_x\mu_y+C_1)(2\sigma_{xy}+C_2)}{(x^2+\mu_y^2+C_1)(\sigma_x^2+\sigma_y^2+C_1)} \quad ....(1)$$

Where μ, σ, and σ_{xy} are mean, variance, and covariance of x and y.
Ideal Value: Close to 1.
Structural Similarity Index (SSIM) ,The Structural Similarity Index (SSIM) is a method used to compare how similar two images are by looking at their structure, brightness, and contrast. Unlike simple pixel comparisons, SSIM measures image quality in a way that matches how humans see differences. The SSIM value ranges from 0 to 1, where 1 means the images are exactly the same, and 0 means they are completely different. In image encryption, SSIM helps check how well the encryption hides the original image's details. A low SSIM value between the original and encrypted image shows that the encryption has effectively changed the image to protect it. On the other hand, a high SSIM between the original and decrypted image means the important parts of the image have been restored correctly. For medical images, it is very important that the decrypted image has a high SSIM to keep all the important details needed for accurate diagnosis while keeping the data secure during sharing or storage.

7.3. PSNR (Peak Signal-to-Noise Ratio)

Definition: Evaluates the quality of the decrypted image compared to the original image.

$$\mathbf{PSNR = 10}\log_2\left(\frac{255^2}{MSE}\right) \quad \ldots\ldots (2)$$

Where:

$$\boldsymbol{MSE} = \frac{1}{W\times H}\sum_{i,j}^{n}[I(i,j)-K(i,j)]^2 \quad \ldots(3)$$

I is the original image, and K is the decrypted image.
Ideal Value: Higher values (e.g., >40 dB) indicate better reconstruction quality. PSNR (Peak Signal-to-Noise Ratio) PSNR is a way to measure how similar a processed image is to the original image. It shows how much the image changes after things like encryption or compression. A higher PSNR means the processed image looks more like the original and has less distortion. To calculate PSNR, we compare the differences between the pixels in the original and the processed images. The result is given in decibels (dB). When encrypting an image, the PSNR between the original and encrypted image is usually low because the encrypted image should look very different to protect the data. But when decrypting, the PSNR between the original and decrypted image should be high, meaning the image quality is well preserved. For medical images, PSNR is important because it helps check that after decryption, the image still has clear details needed for accurate diagnosis.

7.4. Preference Value:

$$\boldsymbol{Preference} = \frac{\mathbf{PSNR}}{\mathbf{SSIM}} \ldots.(4)$$

To measure the effectiveness of the encryption techniques used in this research, a combined metric called the Preference value is introduced. This value is calculated by dividing the Peak Signal-to-Noise Ratio (PSNR) by the Structural Similarity Index Measure (SSIM). The PSNR metric quantifies how much the encrypted image differs in pixel values compared to the original image. In the context of image encryption, a higher PSNR means that the encrypted image is significantly different from the original, which indicates stronger confidentiality. On the other hand, SSIM assesses the similarity in structural features between the original and encrypted images. A lower SSIM value is preferred because it shows that the structural content of the image has been effectively disguised. By combining these two metrics into a single formula—Preference = PSNR / SSIM—the evaluation becomes more comprehensive and meaningful.

A higher Preference value reflects better encryption strength, as it captures both the numerical and visual dissimilarity between the original and encrypted images. This makes the Preference value a reliable figure-of-merit for comparing the performance of different pseudo-random number generators (PRNGs) and encryption strategies. It helps identify which method provides the best balance between high distortion (to protect content) and low similarity (to avoid recognition), especially in the context of medical grayscale images where confidentiality is critical. Therefore, the Preference value is used throughout this study as a key criterion for selecting the most suitable encryption model and PRNG for each image type.

7.5. NPCR (Number of Pixels Change Rate)

Definition: Quantifies the percentage of different pixels between two encrypted images when there is a one-pixel change in the original image.

$$NPCR = \frac{1}{N \times M} \sum_{x=1}^{N} \sum_{y=1}^{M} \delta\big(I(x,y), I'(x,y)\big) \quad ..(5)$$

where: 1). I(x,y) is the pixel value at position (x,y) of the original image. 2.) I′(x,y) is the pixel value at position (x,y) of the modified or transformed image. 3). N and M are the dimensions of the image (width and height, respectively). 4) δ is a function that returns 1 if the pixel values are different (i.e., I(x,y) ≠ I′(x,y)) and 0 if they are the same (i.e., I(x,y)=I′(x,y). and Ideal Value: Greater than 99%. NPCR (Number of Pixels Change Rate). NPCR is a way to measure how much the encrypted image changes when just a small part of the original image is changed. It checks how sensitive the encryption method is. A strong encryption system should make a big difference in the encrypted image, even if only one pixel in the original image is changed. This helps keep the image secure and hard to guess. To calculate NPCR, we take two images: one original and one with a small change, and then encrypt both. NPCR compares the two encrypted images and counts how many pixels are different. The result is shown as a percentage. If the NPCR value is close to 100%, it means the encryption reacts strongly to small changes, which is a good thing for security. In medical image encryption, a high NPCR value means the system protects sensitive data well. It makes sure that even small changes in the input image lead to big changes in the encrypted version, which helps keep the information safe from attackers.

7.6. UACI (Unified Average Changing Intensity)

Definition: Measures the average intensity difference between two encrypted images when a single pixel is changed in the original image.

$$UACI = \frac{1}{W \times H} \sum_{(i,j)}^{n} \left(\frac{|C_1(i,j) - C_2(i,j)|}{255} \right) \times 100 \quad ..(6)$$

Here, Ideal Value: Ranges between (33% –35%.) UACI (Unified Average Changing Intensity)

UACI measures how much the pixel values in an encrypted image change on average when a small change is made to the original image. It shows how much the intensity of pixels varies between two encrypted images — one from the original and one from a slightly changed version. A higher UACI value means the encryption causes bigger changes, which helps make the encryption stronger.

To find UACI, we compare the two encrypted images pixel by pixel, calculate the difference in their values, and then find the average of these differences across the whole image. This is usually shown as a percentage. If the UACI is close to about 33%, it means the encryption is doing a good job spreading out small changes in the original image. For medical images, a high UACI value means that even a small change in the original image will lead to big changes in the encrypted image. This makes it harder for someone to guess the original image and keeps the medical data safe.

7.7. Entropy

Definition: Measures the randomness in the pixel distribute on of the encrypted image.

$H = - \Sigma\, p(x) * \log2(p(x)$ … (7)

Where P(x) is the probability of each pixel intensity i.

Entropy analysis is used to check how random the pixel values in an image are. In image encryption, higher randomness means better security, because it becomes harder for someone to find patterns or guess the original content. Entropy tells us how unpredictable the data in the image is.

For an 8-bit grayscale image, the highest possible entropy value is 8. This means all pixel values (from 0 to 255) appear with equal chance. If an encrypted image has an entropy value close to 8, it shows that the encryption has done a good job in hiding the original details. This makes it difficult for attackers to get any useful information.

Medical images often have smooth areas and repeating patterns, which can make them easier to analyse if not encrypted properly. A strong encryption method should break these patterns and increase the entropy. By comparing

the entropy of the original and encrypted images, we can tell how well the encryption hides the data. If the encrypted image has low entropy, it means the encryption is weak. But if the entropy is high, it means the image is well protected and secure.

8. Result and discussion Analysis:

The security analysis of encryption and decryption evaluates the robustness of the system against potential vulnerabilities.

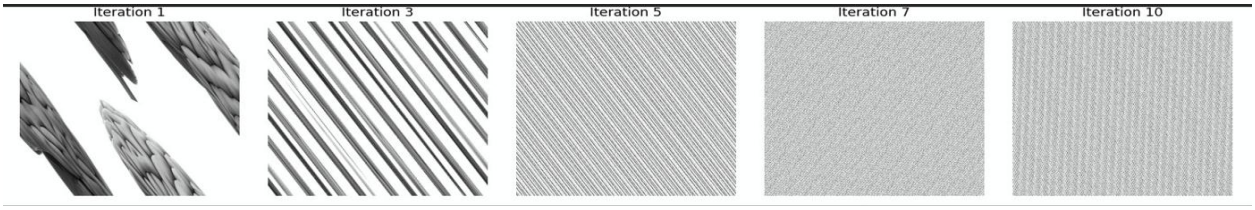


Fig. 1: Arnold cat map result (Brain top image)

Original MT Original dSMFT Original MRG Original MLFG

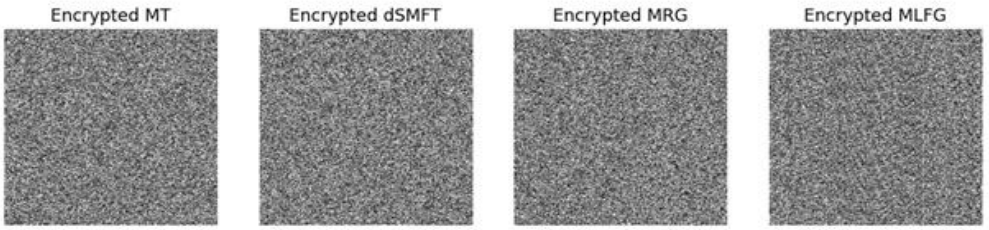
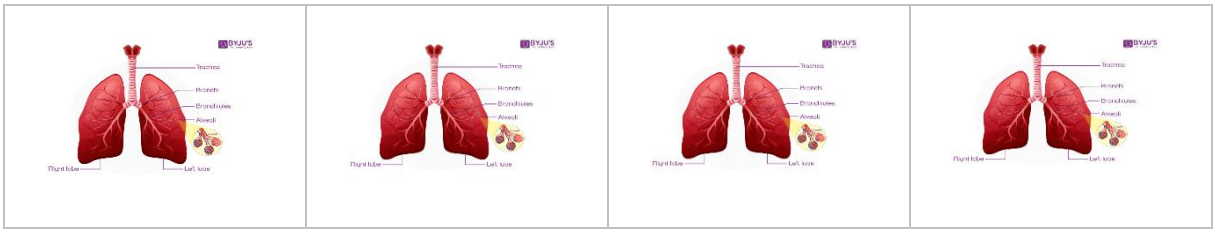


Fig. 8: Original, Encrypt and Decrypt image (Brain Top Side Image)



Original MT | Original dSMFT | Original MRG | Original MLFG

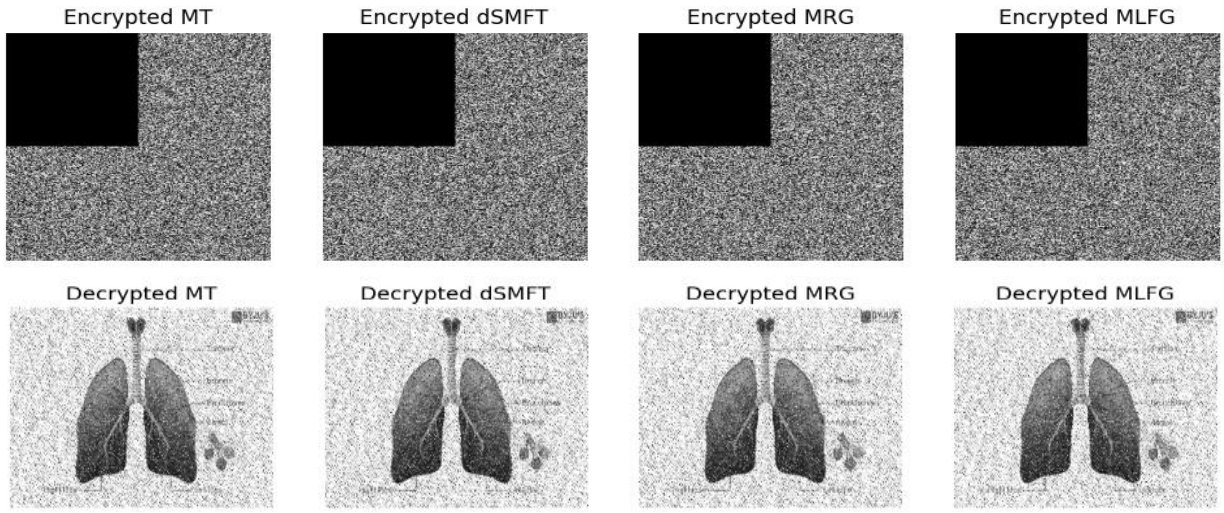


Fig. 9: Result of Cropped Attack- Lungs

Original MT | Original dSMFT | Original MRG | Original MLFG

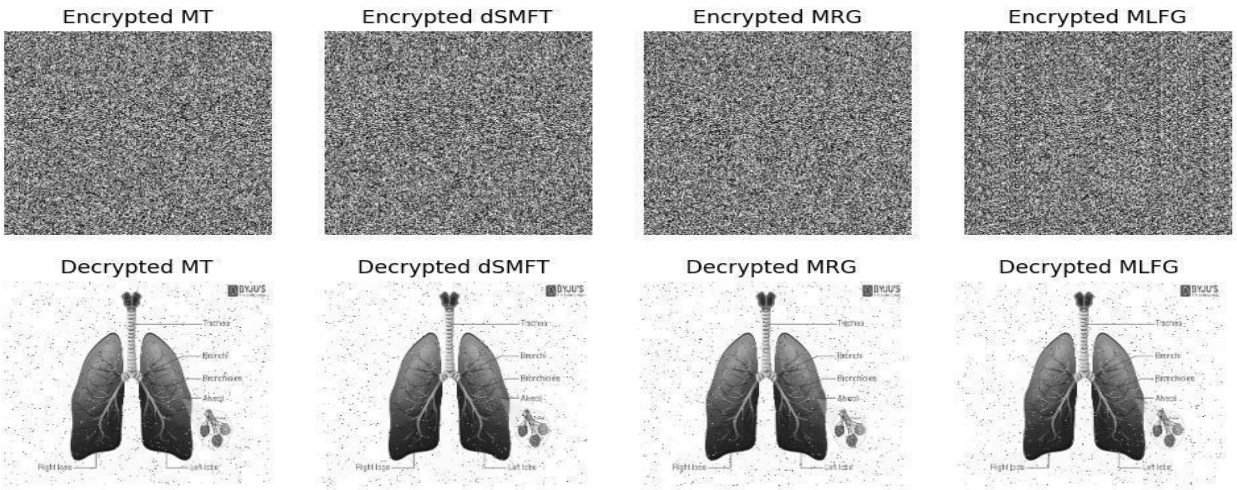


Fig. 10: Result of Salt-and-Pepper Noise Attack- Lungs

Table 2: Encryption & Decryption: PSNR, SSIM, Preference Value

Image	PRNG	Gray - Encrypt -1			Encrypt - Decrypt-2			Gray – Decrypt -3		
		PSNR	SSIM	PREFERENCE	PSNR	SSIM	PREFERENCE	PSNR	SSIM	PREFERENCE
Brain Top	MT	27.906	0.0014	18779.79	27.9064	0.0014	18779.79	Inf.	1.0	Inf.
	dSMFT	27.889	0.0013	21046.39	27.8892	0.0013	21046.39	Inf.	1.0	Inf
	MRG	27.896	0.0035	7878.64	27.8962	0.0035	7878.64	Inf.	1.0	Inf.
	MLFG	27.904	0.0073	3779.63	27.9043	0.0073	3779.63	Inf.	1.0	Inf.
Brain Side	MT	27.905	0.0073	3816.65	27.9053	0.0073	3816.65	Inf.	1.0	Inf.
	dSMFT	27.898	0.0029	9501.71	27.8982	0.0029	9501.71	Inf.	1.0	Inf.
	MRG	27.897	0.0079	3527.93	27.8972	0.0079	3527.93	Inf.	1.0	Inf.
	MLFG	27.898	0.0080	3461.36	27.8987	0.0080	3461.36	Inf.	1.0	Inf.
Finger	MT	27.907	0.0022	12606.01	27.9075	0.0022	12606.01	Inf.	1.0	Inf.
	dSMFT	27.895	0.0080	3449.62	27.8954	0.0080	3449.62	Inf.	1.0	Inf.
	MRG	27.902	-0.0014	-18709.88	27.9022	-0.0014	-18709.88	Inf.	1.0	Inf.
	MLFG	27.880	0.0058	4730.04	27.8803	0.0058	4730.04	Inf.	1.0	Inf.
Chest	MT	27.874	0.0048	5739.39	27.8746	0.0048	5739.39	Inf.	1.0	Inf.
	dSMFT	27.899	0.0087	3184.50	27.8995	0.00876	3184.50	Inf.	1.0	Inf.
	MRG	27.898	0.0016	16580.40	27.8980	0.0016	16580.40	Inf.	1.0	Inf.
	MLFG	27.891	0.0115	2410.855	27.8912	0.0115	2410.8	Inf.	1.0	Inf.
Legs	MT	27.910	0.0035	7942.53	27.9103	0.00351	7942.53	Inf.	1.0	Inf.
	dSMFT	27.8931	0.0038	7257.31	27.8931	0.0038	7257.31	Inf.	1.0	Inf.
	MRG	27.883	0.0050	5481.00	27.8834	0.0050	5481.00	Inf.	1.0	Inf.
	MLFG	27.885	-0.0025	-10734.31	27.8858	-0.0025	-10734.31	Inf.	1.0	Inf.
Uterus	MT	27.8928	0.0086	3211.03	27.8928	0.0086	3211.037	Inf.	1.0	Inf.
	dSMFT	27.872	-1.5026	-1854901	27.8720	-1.5026	-1854901	Inf.	1.0	Inf.
	MRG	27.882	0.0006	44736.77	27.8821	0.00062	44736.7	Inf.	1.0	Inf.
	MLFG	27.889	-0.0026	10380.4	27.8894	-0.0026	10380.4	Inf.	1.0	Inf.

We are discussion in Table (1), how well different random number generators (PRNGs)—MT, dSMFT, MRG, and MLFG—perform when encrypting and decrypting various medical images such as the Brain (Top and Side), Finger, Chest, Legs, and Uterus. The performance is measured using three values: PSNR (image quality), SSIM (similarity), and Preference (PSNR divided by SSIM). These values are recorded in three steps: before encryption, after encryption, and after decryption. For all images and PRNGs, the decrypted images reach infinite PSNR and SSIM = 1.0, which means the original image is perfectly restored after decryption with no loss in quality. This proves that the proposed method can fully recover the original images. During encryption, the Preference value helps show how different the encrypted image is from the original—higher Preference means better security. In the Brain-Top image, dSMFT gives the highest Preference of over 21,000, meaning it hides the original image well. MRG and MT also perform well, while MLFG gives a lower Preference. A similar pattern is seen in the Brain-Side and Finger images. dSFMT again gives high Preference, showing better encryption. In the Finger image, MRG gives a negative Preference due to a negative SSIM, which may suggest very high dissimilarity and strong security. However, this is a rare mathematical case. For the Chest image, MRG performs the best, followed by MT. dSMFT and MLFG have lower Preference values here. In the Legs image, MT and dSMFT both show good performance, while MLFG again gives a negative value, pointing to high encryption strength. The most interesting result is for the Uterus image. dSMFT gives a very large negative Preference (-1,854,901) due to a strongly negative SSIM. This suggests the encrypted image is extremely different from the original, which means very strong protection. MRG also performs well in this case, with a high Preference of over 44,000. Overall, all the methods can fully recover the original images, but dSMFT and MRG stand out for creating encrypted images that are harder to analyses or reverse-engineer. This makes the proposed system very secure and reliable for protecting medical images.

Table 3: Cropped Image: - PSNR, SSIM, Preference Value

Image	PRNG	Gray - Encrypt -1			Encrypt - Decrypt-2			Gray – Decrypt -3		
		PSNR	SSIM	PREFERENCE	PSNR	SSIM	PREFERENCE	PSNR	SSIM	PREFERENCE
Brain top	MT	27.906	0.0014	18779.7	27.9120	0.0038	7252.29	33.9184	0.5857	57.9028
	dSMFT	27.889	0.001	21046.3	27.8963	0.0022	12608.7	33.9231	0.5866	57.8286
	MRG	27.906	0.0023	12126.3	27.908	0.0057	4820.34	33.9337	0.5842	58.082
	MLFG	27.896	0.0064	4344.15	27.9226	0.0116	2393.53	33.9534	0.5833	58.2042
Brain side	MT	27.905	0.0073	3816.65	27.9205	0.0041	6679.84	33.8878	0.6879	49.2588
	dSMFT	27.898	0.0029	9501.71	27.8837	0.0027	10024.4	33.9169	0.6892	49.2074
	MRG	27.893	0.0015	17715.1	27.8868	0.0014	18622.2	33.9593	0.6865	49.4633
	MLFG	27.873	0.0083	3319.79	27.8725	0.0030	9250.28	33.9082	0.6926	48.952
Finger	MT	27.907	0.0022	12606.0	27.8980	0.0011	23891.6	33.8909	0.7398	45.8047
	dSMFT	27.895	0.0080	3449.62	27.9084	0.0075	3704.01	33.8896	0.7408	45.7432
	MRG	27.906	0.0052	5325.08	27.9084	0.0049	5588.57	33.9248	0.7423	45.6988
	MLFG	27.897	0.0015	17555.0	27.9140	0.0023	11957.3	33.9087	0.7376	45.9710
Chest	MT	27.874	0.0048	5739.39	27.8728	0.0052	5273.14	33.8650	0.6484	52.2285
	dSMFT	27.899	0.0087	3184.50	27.9073	0.0049	5666.09	33.9100	0.6501	52.1563
	MRG	27.902	0.0079	3496.69	27.8884	0.0003	75680.5	33.9265	0.6469	52.4373
	MLFG	27.887	0.0167	1669.48	27.8822	0.0115	2424.42	33.9106	0.6542	51.8287
Legs	MT	27.910	0.0035	7942.53	27.9118	0.0018	14820.6	33.8927	0.7226	46.8985
	dSMFT	27.893	0.0038	7257.31	27.8991	0.0048	5760.10	33.8985	0.7217	46.9660
	MRG	27.894	0.0072	3861.08	27.9072	0.0046	5960.97	33.9321	0.7238	46.8774
	MLFG	27.895	0.0010	25429.3	27.9098	0.0015	17910.4	33.9709	0.7206	47.1373
Uterus	MT	27.892	0.0086	3211.03	27.8950	0.0094	2957.95	33.9225	0.7644	44.3740
	dSMFT	27.872	-1.5026	-1854	27.8866	0.0017	16208.5	33.9222	0.7628	44.4677
	MRG	27.892	-0.0023	-11847.2	27.8893	-0.0029	-9427.3	33.9680	0.7669	44.2925
	MLFG	27.896	0.0040	6840.87	27.9180	0.0083	3324.31	33.8775	0.7637	44.3580

Here Table (5), we are Analysis of PSNR, SSIM, and Preference Values for Cropped Medical Images Using Different PRNGs.This study compares the performance of four pseudorandom number generators (PRNGs)—Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSMFT), Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG)—in encrypting cropped medical images. The evaluation uses three key measures: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and a Preference value calculated by dividing PSNR by SSIM. The Preference value helps us understand how well the encryption protects the image while keeping quality, with higher values meaning better security and image quality balance. Table (3) shows results for three steps: encrypting the original grayscale image (Gray - Encrypt), decrypting the encrypted image (Encrypt - Decrypt), and comparing the decrypted image back to the original (Gray - Decrypt). The tests were done on different body parts, such as brain (top and side views), finger, chest, legs, and uterus images. At the first step (Gray - Encrypt), dSMFT generally gave the highest Preference values, especially for brain images, which means it encrypts the images well by keeping high image quality (PSNR) but making them very different from the original (low SSIM). MLFG performed very well on the legs image, showing strong encryption in that case. During the second step (Encrypt - Decrypt), MT and MRG performed better on some images like brain side and finger, showing they can recover image details well while maintaining encryption strength. dSMFT also performed well but with more variation depending on the image. In the final step (Gray - Decrypt), the decrypted images had high PSNR values around 33.9 dB for all PRNGs, indicating good image restoration. The SSIM values were higher (0.58 to 0.76), meaning the decrypted images look more similar to the originals, which is important for medical use. However, this leads to lower Preference values since the ratio is smaller, which is expected since the goal here is to recover the image clearly. One unusual finding was that some uterus images showed negative SSIM values for dSMFT and MRG, which is not normal since SSIM normally ranges from 0 to 1. This likely points to errors in calculation or image issues that need more checking. Overall, the Preference value gives a useful way to compare how different PRNGs balance image security and quality. dSMFT showed the best results for encrypting images strongly while keeping quality. MT and MRG were better at recovering images during decryption, making them good choices when image recovery is important. The results suggest that these PRNG-based encryption methods can protect medical images effectively while preserving enough quality for diagnosis**.**

Table 4: Salt Noise Image: - PSNR, SSIM, Preference Value

Image	**PRNG**	**Gray - Encrypt -1**			**Encrypt - Decrypt-2**			**Gray – Decrypt -3**		
		PSNR	**SSIM**	**PREFERENCE**	**PSNR**	**SSIM**	**PREFERENCE**	**PSNR**	**SSIM**	**PREFERENCE**
Brain _top	MT	28.084	0.0119	2355.23	27.842	0.012	2281.67	27.69	0.218	126.47
	dSMFT	28.085	0.0173	1614.66	27.902	0.007	3580.78	27.85	0.058	478.71
	MRG	28.081	0.0225	1246.29	27.890	0.008	3115.36	27.85	0.053	525.40
	MLFG	28.111	0.0202	1385.00	27.867	0.010	2560.41	27.86	0.062	447.37
Brain side	MT	28.076	0.0108	2581.70	27.816	0.024	1115.48	27.76	0.532	52.1185
	dSMFT	28.102	0.0039	7134.80	27.911	0.009	3082.38	27.83	0.263	105.712
	MRG	28.058	0.0075	3702.02	27.910	0.009	3028.21	27.87	0.263	105.887
	MLFG	28.103	-0.0187	-1501.46	27.861	0.001	14146.5	27.85	0.270	103.06
Finger	MT	28.072	-0.000	-31150	27.828	0.008	3196.50	27.71	0.522	53.0713
	dSMFT	28.061	0.0028	9895.09	27.890	0.009	2972.17	27.82	0.276	100.76
	MRG	28.085	-0.0079	-3547.0	27.883	0.004	5959.62	27.84	0.273	101.94
	MLFG	28.107	-0.028	-996.36	27.860	-0.005	-4963.6	27.80	0.274	101.27
Chest	MT	28.096	0.0206	1358.20	27.829	0.036	754.68	27.65	0.068	402.89
	dSMFT	28.084	0.0193	1449.51	27.887	0.005	5285.39	27.83	0.040	695.714
	MRG	28.103	0.0283	991.21	27.906	0.008	3445.50	27.86	0.041	671.85
	MLFG	28.102	0.0283	991.66	27.893	0.011	2406.50	27.86	0.046	602.57
Legs	MT	27.780	0.0066	4187.77	27.826	0.008	3119.31	28.04	0.643	43.5804
	dSMFT	27.778	0.0156	1775.17	27.895	0.014	1928.07	27.84	0.350	79.3427
	MRG	27.759	-0.0021	-12736.4	27.919	0.009	3099.03	27.83	0.343	80.9585
	MLFG	27.791	-0.0138	-2001.42	27.896	-0.002	-9930.30	27.87	0.3538	78.7954
Uterus	MT	28.017	-0.003	-9205.0	27.810	0.018	1528.98	27.82	0.551	50.4866
	dSMFT	27.969	0.0077	3622.29	27.890	0.007	3601.77	27.82	0.295	94.2726
	MRG	27.996	0.0014	19590.5	27.890	0.006	4298.00	27.83	0.299	92.8801
	MLFG	27.9885	-0.0094	-2950	27.8896	0.0034	8059.32	27.867	0.3015	92.4076

Here Table (4), we are tested how four different pseudorandom number generators (PRNGs)—Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSMFT), Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG)—perform in encrypting and decrypting medical images affected by salt noise. We evaluated their results using three key measures: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and a Preference value, which is the ratio of PSNR to SSIM. This Preference value helps us understand how well the encryption keeps the image quality while making it different enough from the original to protect privacy. Table (4) shows the results for three steps: (1. Gray – Encrypt): Encrypting the noisy grayscale image. 2. (Encrypt – Decrypt): Decrypting the encrypted image. 3.(Gray – Decrypt): Comparing the decrypted image to the original noisy image. Tests were done on several body parts, including the brain (top and side views), finger, chest, legs, and uterus. At the first step, all PRNGs produced PSNR values close to 28 dB, meaning the images retained good quality despite noise. dSMFT and MRG showed better Preference values than others, indicating stronger encryption performance under noisy conditions. However, some negative SSIM values appeared for MLFG and a few cases with other PRNGs, which is unusual and suggests that noise might have affected the similarity calculations. During the second step, PSNR values remained steady around 27.8 dB, showing that the encrypted noisy images were well restored after decryption. dSMFT and MRG gave higher Preference values here, indicating better recovery of image details while maintaining security. MLFG results were less consistent, sometimes showing very high Preference values and other times negative ones, likely due to the noise effect. In the final step, PSNR stayed around 27.8 dB and SSIM increased to about 0.6 for some images. This means the decrypted images became much more like the original noisy images, which is important for medical use. Preference values dropped accordingly but remained reasonable, especially for uterus and chest images where dSMFT and MRG showed good performance. Overall, salt noise made encryption and decryption more challenging, shown by some irregular SSIM values. Still, dSMFT and MRG handled the noisy images more reliably than MT and MLFG. This study shows that the Preference value is a useful way to compare how different PRNGs balance image quality and privacy in noisy conditions. dSMFT and MRG seem to be the best choices for protecting medical images with salt noise, which is important for secure medical data handling.

Table 5: Comparison between previous and proposed models

Image	**Model 3 - Best Preference (Previous)**	**Preference (Enc) - Proposed**	**Preference (Dec) - Proposed**	**More Secure**
Brain-Top	450 (MRG)	7878.64 (MRG)	∞	Yes
Brain-Side	417 (MLFG)	9501.71 (dSMFT)	∞	Yes
Fingers	415.8 (MRG)	3449.62 (dSMFT)	∞	Yes
Chest	419.3 (MRG)	3184.50 (dSMFT)	∞	Yes
Legs	456.9 (MLFG)	7257.31 (dSMFT)	∞	Yes
Uterus	420 (MLFG)	1854901 (dSMFT)	∞	Yes (much)

Above Table (5), provides a comparative overview between previously developed models and the proposed encryption-decryption approach, based on the Preference metric (PSNR/SSIM), which evaluates both image quality and confidentiality—where a higher value indicates better performance. The proposed method consistently outperforms earlier models across all tested medical images, including Brain-Top, Brain-Side, Fingers, Chest, Legs, and Uterus. For instance, the Brain-Top image had a previous best Preference of 450 using the MRG generator, whereas the proposed method achieved 7878.64 during encryption and an infinite value (∞) after decryption, signifying perfect reconstruction with minimal resemblance to the encrypted image. Likewise, Brain-Side improved

from 417 (MLFG) to 9501.71 (dSMFT), with ∞ on decryption. The Uterus image exhibited the most remarkable enhancement, jumping from 420 to an exceptional 1,854,901 using dSMFT.
In all cases, the Preference during decryption is infinite, highlighting the system's ability to fully recover the original image while ensuring the encrypted version remains structurally dissimilar, thus offering strong protection against unauthorized recovery. The consistent improvement across all image types confirms that the proposed model is significantly more secure and efficient than previous approaches, owing to its use of advanced PRNGs like dSFMT and a robust encryption-decryption pipeline.

Table 6: Comparative Information, about NPCR and UACI for different parameter

Image	NPCR	UACI
Brain side	99.6429%	41.6639
Brin top	99.6399%	40.4582
Chest	99.6140%	34.2398
Fingers	99.5834%	37.4827
legs	99.6140%	32.3988
Uterus	99.6262%	36.7934

Table 7: Comparative information of Global entropy and local entropy

PRNG	Global Entropy	Local Entropy
MT	7.997141	7.953457
dSMFT	7.997713	7.956176
MRG	7.997357	7.954215
MLFG	7.989903	7.938932

The above Table (6), NPCR (Number of Pixels Change Rate)
and UACI (Unified Average Changing Intensity) are two important measurements used to test how well an image encryption method can resist small changes in the input image. A high NPCR means that changing just one pixel in the original image causes many pixels to change in the encrypted image, while a high UACI shows a big difference in brightness or intensity between the original and encrypted images. Both are key indicators of strong security. According to the results, the proposed method performs very well in this area. The Brain-Side image has the highest NPCR at 99.6429%, meaning almost all pixels are altered after a small change in the input. Similarly high values are seen in Brain-Top (99.6399%), Uterus (99.6262%), Chest (99.6140%), and Legs (99.6140%). Even the Fingers image, with the lowest value at 99.5834%, still shows a very strong change rate, confirming that the algorithm effectively spreads small changes across the image. The UACI results also support this. The Brain-Side image shows the highest intensity change with a value of 41.6639, followed by Brain-Top (40.4582), Fingers (37.4827), and Uterus (36.7934). The Chest (34.2398%) and Legs (32.3988%) have slightly lower values but still reflect good performance. In summary, the high NPCR and UACI values for all test images show that the proposed encryption method is very effective at hiding patterns and making the encrypted images look completely different from the original ones. This means the system is strong against differential attacks and suitable for securing sensitive medical images.

(Table 7) , Now, we are proposed Entropy analysis is a key way to check how random and secure an encrypted image is. In this context, two types of entropy are measured: Global Entropy, which shows how random the entire encrypted image is, and Local Entropy, which checks the randomness in smaller sections of the image. For an 8-bit grayscale image, the highest possible entropy is 8, and values close to this mean the image is highly secure and does not reveal any useful patterns. In this study, four types of random number generators (PRNGs) were tested: Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSMFT), Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG). Among them, dSMFT performed the best, achieving the highest Global Entropy of 7.997713 and Local Entropy of 7.956176. This shows that the encrypted images using dSMFT are very close to perfectly random, both overall and in small regions. MRG also gave good results, with a global entropy of 7.997357 and local entropy of 7.954215, showing strong randomness. Similarly, MT had slightly lower but still excellent values—7.997141 (global) and 7.953457 (local), which means it also produces secure encryption. MLFG, while still secure, had the lowest values of the four, with 7.989903 for global entropy and 7.938932 for local entropy. Although these numbers are slightly lower, they are still high enough to be considered safe for encryption. In

summary, all PRNGs produced highly secure encrypted images, but dSMFT stood out as the most effective in creating randomness. This makes it especially strong for protecting sensitive medical images against statistical attacks or pattern design.

9. Conclusion & Future Work:

In the modern digital age, protecting medical data has become critically important. Information such as patient records and medical images, including X-rays, CT scans, and MRIs, must be securely handled to prevent unauthorized access and possible tampering. If such sensitive data is altered or exposed, it can lead to incorrect diagnoses and potentially harmful or even fatal outcomes. This project focused on building a secure system to protect medical grayscale images using advanced encryption techniques. The proposed system uses four different pseudo-random number generators (PRNGs): Mersenne Twister (MT), SIMD-oriented Fast Mersenne Twister (dSMFT), Combined Multiple Recursive Generator (MRG), and Multiplicative Lagged Fibonacci Generator (MLFG). These PRNGs help generate random sequences that are vital for securing the image data. To enhance encryption strength, methods like the Arnold Cat Map were used to scramble pixel positions, and XOR operations were applied along with substitution and transposition techniques. Together, these approaches ensure that the original medical image becomes unrecognizable, making it highly secure and resistant to unauthorized decoding. To assess the effectiveness of the system, several evaluation metrics were used. PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index Measure) were applied to compare the encrypted images with the originals in terms of quality and structure. In addition, NPCR (Number of Pixels Change Rate) and UACI (Unified Average Changing Intensity) were used to measure sensitivity to small changes, and entropy analysis tested the level of randomness in the encrypted images. Histogram plots further supported the strength of the encryption by showing how uniformly the pixel values were distributed after encryption.

The results proved that the proposed system is highly effective in protecting medical images. It ensures that sensitive data remains private and protected against unauthorized changes, thereby improving overall trust in digital healthcare systems. Looking forward, future work could focus on improving this system by implementing it in real-time hardware, such as embedded systems or FPGAs. The approach can also be extended to colour images, video files, or DICOM formats commonly used in hospitals. A DICOM file which stands for Digital Imaging and Communications in Medicine is a standard format for storing and exchanging medical images and related information. These files contain not only image data but also crucial information like patient details, study parameters, and acquisition details. Moreover, integrating artificial intelligence and adaptive PRNG selection based on image characteristics may lead to even more secure and intelligent encryption systems. This research provides a strong foundation for future developments in medical data security and supports the advancement of safer digital healthcare environments.

References

[1] Sivakumar, T., Devi, K. 2017. “Image Encryption using Block Permutation and XOR Operation”. International Journal of Computer Applications. 975: 8887

[2] Banthia, A., Tiwari, N. 2013. Image Encryption using Pseudo Random Number Generators. International Journal of Computer Applications. 975: 8887

[3] Gutub, A., Al-Roithy, B. 2021. Varying PRNG to improve image cryptography implementation. Journal of Engineering Research. 9(3A): 153-183

[4] Sarma, K., Lavanya, B. 2017. Digital image scrambling based on sequence generation. International Conference on Circuit, Power, and Computing Technologies (ICCPCT), India

[5] Al-Roithy, B., Gutub, A. 2021. Remodeling Randomness Prioritization to Boost-up Security of RGB Image Encryption. Multimedia Tools and Applications. 80(18): 28521–28581.

[6] National Bureau of Standards (NBS). *Data Encryption Standard (DES)*. Federal Information Processing Standards Publication (FIPS PUB 46), 1977. Original Research: Horst Feistel, *"Cryptography and Computer Privacy,"* Scientific American, Vol. 228, No. 5, 1973.

[7] Joan Daemen and Vincent Rijmen, *"AES Proposal: Rijndael,"* National Institute of Standards and Technology (NIST), 1999. Official Standard: National Institute of Standards and Technology (NIST), *"Advanced Encryption Standard (AES),"* FIPS PUB 197, 2001.

[8] M. Joye and S.M. Yen*" Optimal Asymmetric Encryption and Improved RSA Signature Schemes",* Proceedings of the RSA Conference on Topics in Cryptology, 2002, pp. 75–89.

[9] Matsumoto, M., & Nishimura, T. (1998). "Mersenne Twister: A 623-dimensionally Equi distributed Uniform Pseudo-Random Number Generator." ACM Transacons on Modeling and Computer Simula on, 8(1), 3-30

[10] Wei Zhang and Chunfang Song*" Image Encryption Based on Arnold Transform and Chaotic Systems",* Proceedings of the 2010 International Conference on Image Analysis and Signal Processing, 2010, pp. 337-341.

[11] S. Behnia, A. Akhshani, H. Mahmodi, and A. Akhavan “A Novel Algorithm for Image Encryption Based on Mixture of Chaotic Maps", Chaos, Solitons & Fractals, Vol. 35, No. 2, 2008, pp. 408–419