International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 13 Issue: 1
Article Received: *25 November 2024* ***Revised:*** *12 December 2024* ***Accepted:*** *30 January 2025*

Optimized AES with GAN Model for Secure Medical Image Transmission

Vishnupriya V
Department of Information Technology
PSG College of Technology
Coimbatore, India
Vishnupriya.sri1311@gmail.com ,23pb38@psgtech.ac.in

Sarathambekai S
Department of Information Technology
PSG College of Technology
Coimbatore, India
vrs070708@gmail.com

Abstract—The rapid technological development and increased computational capabilities, cybersecurity risks are on the rise. This has led to a growing need for cutting-edge security algorithms, especially in fields like healthcare where medical images play a crucial role in diagnosing various conditions. As these images are frequently transmitted over the internet, safeguarding them from cyber threats is essential. The new framework for encryption is named PSO-AES-GAN(PSAGA). This paper introduces PSO based AES for encryption and generative GAN (Generative Adversarial Network) for key generation to strengthen the security of medical images. The model leverages an AES with PSO (Particle Swarm Optimization) encryption, SHA- 256 hash table, and GAN deep learning techniques. A SHA- 256 hash-table-based equation and AES with PSO enhance key entropy. Differential Huffman Compression (DHC) is utilized to compress encrypted images low-loss. The medical images have undergone testing using this model and assessed using performance metrics such as entropy, Encryption time, Decryption time, and Compared encryption algorithms such as chaotic maps, DES, AES, and Blowfish with similarity. Results show that the suggested model outperforms current methods.

Keywords- Generative Adversarial Networks (GAN), Deep learning, Encryption, Decryption, Compression, Huffman Compression Technique (HCT)

I. INTRODUCTION

Medical images, essential for patient diagnosis and treatment, are produced by various medical devices. These images are large and rapidly growing, yet they also contain sensitive patient information. The exposure of this data raises serious privacy concerns. As computing technology progresses, it is evident that there is a strong demand for secure cryptographic methods that allow for safe remote patient examination. To address these issues, Various strategies have been suggested to safeguard these images against cryptographic threats and to optimize their storage efficiency, including image watermarking [1], encryption [2,3], steganography [4,5], and lossless compression [6]. Despite the availability of these methods, encryption remains a widely used solution for safeguarding medical images. Consequently, there is a need for innovative algorithms to improve the security and compression of medical images. The DeepKeyGen model has been introduced as a deep learning- based method for encrypting medical images, which includes a decryption component to guarantee the secure management of medical data [7]. The rapid advancement of digital healthcare

advancement of digital healthcare has underscored the need for secure transmission of sensitive medical data, including diagnostic images crucial for patient care. As telemedicine, electronic health records (EHRs), and remote consultations expand, the potential for unauthorized access, data manipulation, and breaches increases. Cyberattacks on healthcare data not only jeopardize patient privacy but can compromise diagnosis accuracy, posing a risk to patient safety. Thus, a sophisticated framework is essential for securely transmitting medical images, ensuring data integrity, and restricting unauthorized access. Deep Learning (DL), which is a sub-domain of machine learning that initiate neural networks for learning from large datasets, has become increasingly important in various fields such as image segmentation, object recognition [8], music generation [9], and fake image synthesis [10]. In the healthcare sector, medical images are susceptible to cyber threats and occupy significant storage space, presenting challenges for traditional security schemes due to pixel redundancy and sensitivity to minor data changes. In response to the constraints of current algorithms, a new cryptography system has been introduced for the safe and effective. This paper begins by creating a key image using a Generative Adversarial Network (GAN). Following key generation, Advanced Encryption Standard (AES) combined with Particle Swarm Optimization (PSO), and the secure final key is derived by a specific hash- based equation. After the key is generated, AES, PSO, Henon map, and XOR-based methods are used to encrypt the original medical image. Lastly, the image that has been encrypted is compressed utilizing the DHC algorithm before being transmitted to the server.

The main contribution from the research work can be summarized as follows.

• The encryption of images is implemented using the PSO-AES-GAN (PSAGA) method, along with a lossless compression technique and DHC to enhance storage and transmission efficiency.

• A key generation method has been developed leveraging a GAN and the SHA-256 hash-map to produce a highly secure, large key.

• A comprehensive comparative analysis of results is conducted utilizing metrics like entropy, encryption time, and decryption time, which are evaluated against encryption methods such as Chaotic map, AES, DES, and Blowfish.

This paper has five sections. The next section contains a review of relevant publications. Section 3 explains the recommended techniques. In Section 4, evaluates and compares the performance of the recommended methods of various existing models. Finally, Section 5 summarizes the conclusions of the paper.

II. LITERATURE SURVEY

Medical images are crucial for healthcare but are vulnerable to cyberattacks. This review explores encryption methods to protect them, focusing on AES with Particle Swarm Optimization, GANs for dynamic key generation, Henon map-based pixel scrambling, and SHA-256 hashing to enhance security and safeguard sensitive medical data. AES, a widely adopted symmetric encryption method, is known for its resilience across various data types, including medical images. Built upon multiple rounds of substitution and permutation (confusion and diffusion), AES is robust against brute-force attacks and statistical analyses, making it an ideal choice in healthcare, where secure and swift data access is crucial, as in telemedicine. In 2019, Dagadu et al. [11] pro- posed enhancing AES with chaotic DNA diffusion, leveraging chaotic systems to disperse pixel values and reduce statistical correlations between original and encrypted images. However, a limitation of AES is its fixed key generation, which restricts its adaptability to medical images of different complexities and sizes. Although AES enables rapid decryption which is vital in emergency situations its static key structure may be vulnerable to specific attack strategies, highlighting the need for enhanced encryption techniques for complex medical data.

In 2021, Ding Y, et al. [12] introduced DeepKeyGen, a deep learning-based stream cipher that generates highly secure cryptographic keys with GANs for encrypting medical images. GANs consist of two neural networks, a generator, and a discriminator that collaborate to produce unpredictable outputs, creating a security level that reduces the likelihood of successful brute-force attacks. GANs can produce keys tailored to different image complexities, further strengthening encryption, which is vital for safeguarding sensitive medical data. Ding et al.'s [13] work demonstrated that GANs could generate highly unpredictable keys, thus reinforcing the resilience of encryption systems against cyber threats.

In a 2021 study, Amoh et al. [14] examined Henon maps, chaotic systems generating random sequences useful for pixel scrambling, through a bifurcation analysis of generalized Henon maps. Henon maps effectively shuffle pixel positions in image encryption, ensuring that adjacent pixels in the original image seem significantly separated in the encrypted image. This randomness minimizes statistical correlations, making it challenging for attackers to discern patterns. When incorporated with AES, Henon map-based pixel scrambling enhances AES's confusion capabilities, providing an additional security layer by making it more difficult to reverse-engineer the original image from the encrypted one. Due to their ability to generate unpredictable sequences resistant to known plaintext and differential attacks, Henon maps have become popular for medical image encryption.

In 2022, Magdy et al. [15] investigated telemedicine security, underscoring AES as a reliable encryption method for medical images. While AES is effective, its static key structure limits adaptability to images of varying complexities, an issue when medical data requires secure, real-time delivery in telemedicine consultations. Magdy et al. [15] suggested that implementing additional techniques, such as key optimization, could improve AES's performance in encrypting complex medical images.

One promising solution is combining PSO with AES. Inspired by the coordinated behavior of birds or fish, PSO is an optimization technique that enhances cryptographic algorithms by dynamically adjusting key generation. PSO bolsters the confusion and diffusion properties of encrypted images by selecting secure keys based on fitness functions that assess randomness and security. Integrating PSO with AES yields a more flexible encryption method suitable for high-dimensional medical images. AES-PSO mitigates AES's main limitation—its susceptibility due to a static key schedule—by introducing a more dynamic and adaptable key generation mechanism that is highly resistant to attacks.

In 2022, Mohanty et al. [16] researched healthcare security using a modified SHA-256 alongside deep learning. SHA-256 is a cryptographic hash function producing a fixed-size hash that verifies data integrity during transmission. This showed that using SHA-256 in healthcare enhances key entropy and verifies encrypted medical image integrity. SHA-256 generates unique hashes for images, so any modification in the image or key yields a different hash, making unauthorized alterations easy to detect. Within the AES-PSO-GAN framework, SHA-256 provides an extra layer of encryption security, lowering the likelihood of key-recovery attacks.

In 2023, Singh et al. [17] examined deep learning models utilizing PSO for medical image classification, showing that PSO's ability to continually refine parameters through fitness functions applies well to encryption. AES-PSO refines encryption keys throughout the process, ensuring they meet criteria like randomness and resistance to attacks. Singh et al.'s [18] study found that PSO broadens AES's key space and enhances security, making it useful for real-time medical image encryption where security and performance are critical. The AES-PSO-GAN(PSAGA) framework enhances medical image security by combining GAN-based key generation, PSO optimization, and Henon maps. GANs improve key unpredictability, PSO dynamically selects secure keys, and Henon maps increase confusion by rearranging pixels. SHA-256 ensures image integrity, detecting unauthorized changes. This approach offers advanced encryption, flexibility, and high resistance to attacks, ensuring secure, real-time transmission of medical images across diverse healthcare environments.

III. SYSTEM DESIGN

Medical images require secure transmission and storage to protect patient data from modern cyber-attacks utilizing advanced computing power. To address this, a novel crypto-graphic model has been proposed, integrating deep learning, AES with PSO encryption, SHA-256 hash-maps, and Dynamic Huffman Coding (DHC) for efficient encryption with compression. This model optimizes and ensures the secure transfer of medical images, maintaining data accuracy while minimizing storage needs. It has wide applications, including healthcare, where it secures image transfers for diagnostics, and academia, where it protects medical images used in research.

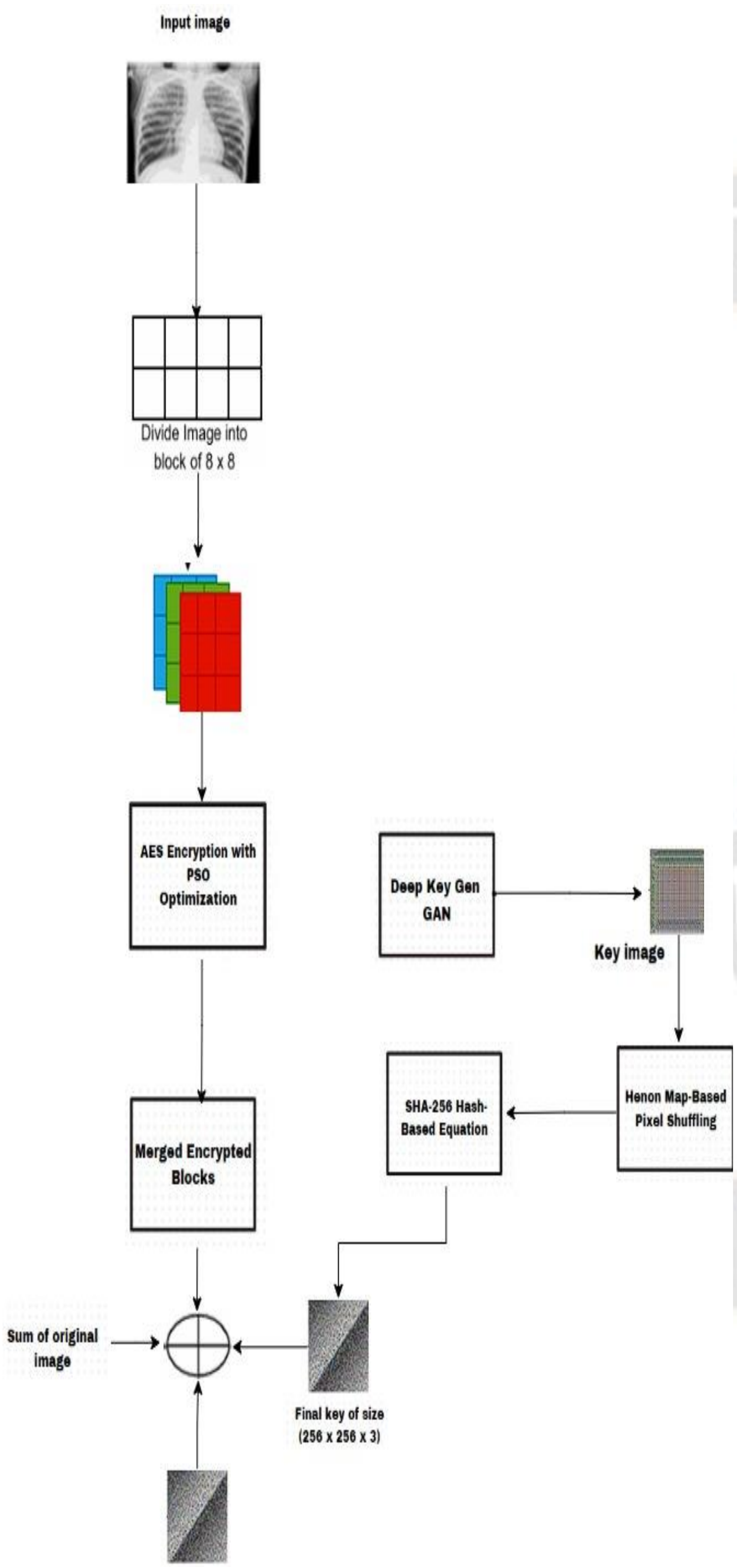
Input image
Divide Image into block of 8 x 8
AES Encryption with PSO Optimization
Deep Key Gen GAN
Key image
Henon Map-Based Pixel Shuffling
SHA-256 Hash-Based Equation
Merged Encrypted Blocks
Sum of original image
Final key of size (256 x 256 x 3)

Fig.1 Flow Diagram of the Proposed System

A. *Key generation*

The initial stage of the technique is, to create a robust for encryption key. The suggested approach utilizes a deep learning-based GAN model in conjunction with AES and PSO encryption, along with a hash-based equation identified as SHA-256, to produce secret keys. At first, this approach utilizes the DeepKeyGen architecture based on GANs to create a synthetic image, which serves as the initial key. The working slow of the proposed GAN is represented in Fig.1. The Key generation technique includes forming a secret key by merging a GAN-produced synthetic image as specified in Eq.1. Subsequently, the AES with PSO method is applied, where the initial sequence is utilized during key generation phase, and the other parts is used in the encryption technique, as defined in Eq.2. The entire Key Generation procedure is detailed in Algorithm 1.

Algorithm 1: Key Generation

```
Function GenerateKey():
fake_image = GAN.Create()
 // Generate a synthetic image using a GAN model
Initialize a swarm of particles with random AES keys and IVs
For each iteration in
max_iterations:
For each particle in the swarm:
    Adjust the particle's position and velocity based on fitness criteria
    Evaluate the fitness of the current position
    If the current position is the best for this particle so far:
    Store it in the particle's best position

Update the global best position based on swarm performance End
   For Select AES-key, iv from the global best position
   padded_image_data = ApplyPadding(fake_image) Initial- size AES
   cipher using aes_key and iv encrypted image
   encrypted_image= AES.Encrypt(padded_image_data) compressed
   encrypted_image = Compression.Apply(encrypted_image)
// This is the final 256-bit secret encryption key
   final_key=SHA256.Hash(compressed_encrypted_image)Return
   final_key
// Main execution final_key = GenerateKey()
```

GAN-based Image Generation:

Generate a fake image using the GAN model

$$\mathrm{I}_{fake} = DeepKeyGen(x)\text{………………..Eq.1}$$

AES Encryption with PSO Optimization:

Encrypt the fake image using AES with the keys K_{aes} and IV, optimized by PSO.

$$\mathrm{I}_{encrypted} = AES(I_{fake}\ , K_{aes}, IV)\text{ ……..Eq.2}$$

The SHA-256 hash function specified in Eq.3 is utilized to derive the key necessary for producing the final secret key image. This hash function is required to enhance key entropy and ensure the proper distribution of pixels. A trusted third-party authority will oversee the sharing of keys by utilizing both the Henon map and the hash function to enhance the protection of the keys produced by the GAN. User authentication will be verified before the sharing of keys. The symbols mentioned throughout this paper are defined in Table 1.

Article Received: *25 November 2024* ***Revised:*** *12 December 2024* ***Accepted:*** *30 January 2025*

Table 1: Notations with their definition

Notation	Definition
P(i)	The probability of pixel intensity I in the image
T-start	The timestamp when encryption and decryption begin**.**
Tend	The timestamp when encryption and decryption end**.**
$Ifake$	The artificial image produced by the GAN-based system.
DeepKeyGen(x)	The GAN model utilized for creating the fake image.
x	The input to the GAN model
$\boldsymbol{Iencrypted}$	The fake image encryption
$\boldsymbol{Kaes}$	The secret AES key utilized for the encryption of the fake image
IV	The Initialization Vector
$\boldsymbol{kfinal}$	The final secret key generated from the encrypted image

Encryption with compression

This optimized AES encryption uses Particle Swarm Optimization (PSO) to enhance data security and efficiency. First, the input image is optionally compressed (using lossless or lossy methods) to reduce file size while maintaining essential quality. The compressed image is then segmented into 8x8 blocks for targeted AES encryption. PSO is applied to optimize encryption parameters, ensuring that the process is both secure and effective. After encryption, an additional compression layer is applied to the encrypted blocks, using techniques such as Run-Length Encoding, Huffman Coding, or entropy-based methods. This secondary compression decreases storage space and allows for quicker data transfer, making the encrypted image easier to manage. The decryption procedure undoes these actions, decompressing the data first and then decrypting it using the optimized AES configuration. By implementing compression both before and after encryption, this method achieves a balance between robust security and efficient data management. Subsequently, the HCT is utilized to produce the final cipher image, as illustrated in Fig 2.

The comprehensive work for implementing encryption based on the compression method in Algorithm 2.

B. Decompression and Decryption

On the receiving end, the initial image is retrieved from the compressed cipher image using decryption and decompression. process that is executed in reverse. The first step involves using the Huffman decoder on the compressed cipher image, followed by recovery of the cipher image. Once the cipher image is acquired, decryption is performed by applying an XOR operation between the cipher image, the key, and the sum of the pixel values from the original image. Subsequently, the decrypted image undergoes shuffling, and an XOR operation is applied using the sequence generated by the He´non map. Lastly, a block-based reshuffling process is performed to retrieve the initial image. The entire process of decompression and decryption is detailed in Algorithm 3.

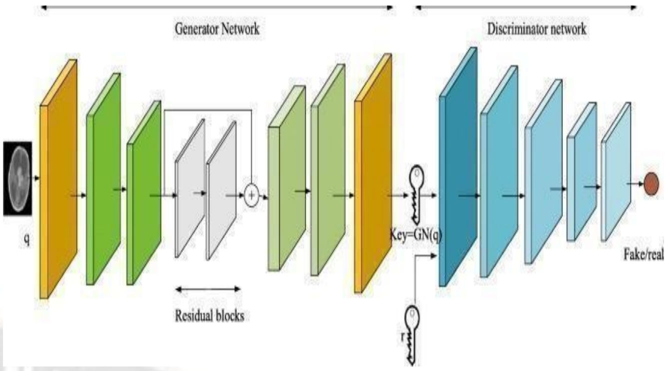

Fig.2 GAN for Key Generation

Algorithm 2: Encryption with compression

```
Input: Image dataset M, where i = 1, 2, 3, ..., n and m × m is the size of the matrix.
Output: Encrypted image E, final key K_final.
M_padded = pad_image(M)
// Pad image to dimensions suitable for 8x8 blocks
B = divide_into_blocks(M_padded, 8)
// Divide image into 8x8 blocks
for each block b in B do
// PSO optimization to find AES key
optimized_key = pso_optimize_aes_key(b)
// AES encryption on block
Append encrypted_block to aes_encrypt(b, optimized_key)
end
// Merge encrypted K_final[row, col]
M_merged = merge_blocks(encrypted_blocks)
// blocks into one image
// Generate key image using GAN
key_image = deep_key_gen_gan()
// Apply Henon map-based pixel shuffling
shuffled_key_image = henon_map_pixel_shuffle(key_image)
// Calculate the sum of all pixel values in the original image
sum_original = calculate_sum_of_image(M)

// Generate key using SHA-256 hash
 Sum of  hash_based_key = sha256_hash(sum_original)
// Combine shuffled key image and hash-based key
K_final = combine_keys(shuffled_key_image, hash_based_key)
for row in range(len(M_merged)) do
for col in range(len(M_merged[0])) do
// Perform XOR between merged image and final key
E[row, col] = M_merged[row, col] ⊕ K_final[row, col]
end

Return E, K_final
// Encrypted image and final key
```

Algorithm 3: De-compression with decryption

```
Input: Encrypted image E, final key K_final, and sum of original image pixels.
Output: Decrypted original image M.
sum_original = calculate_sum_of_image(M)
 // Recalculate the sum of the original image pixels
 hash_based_key= sha256_hash(sum_original)
 // Generate key using SHA-256 hash of sum
key_image = deep_key_gen_gan()
// Regenerate key image using GAN
```

Article Received: *25 November 2024* ***Revised:*** *12 December 2024* ***Accepted:*** *30 January 2025*

```
// Apply Henon map-based pixel shuffling
shuffled_keyimage= henon_map_pixel_shuffle(key image)
// Combine shuffled key image and hash-based key
K_final=combine_keys(shuffled_key_image,hash_based_key)
for row_r in range(len(E)) do
for col_c in range(len(E[0])) do
M_merged[row_r, col_c] = E[row_r, col_c] ⊕ Reverse XOR with
the final key to get merged encrypted blocks
end

B_decrypted = split_into_blocks(M_merged)
 // Split merged encrypted image back into blocks

for each block b in B_decrypted do
optimized_key= pso_optimize_aes_key(b)
 // Re-generate optimized AES key using PSO
// AES decryption on block
decrypted_block = aes_decrypt(b, optimized_key)
Append decrypted_block to decrypted_blocks
end
// Merge decrypted blocks to reconstruct the original image
M_reconstructed = merge_blocks(decrypted_blocks)
Return M_reconstructed
// Decrypted original image
```

IV. PERFORMANCE ANALYSIS

The suggested approach proposes a robust key generation algorithm based on the SHA-256 hash map, AES with PSO, and the deep learning technique GAN. The initial medical image is secured through the XOR encryption technique, and a compressed cipher image is obtained by adopting DHC. The results indicate that the algorithm generates a higher secure cipher image while maintaining a very small image resolution. Using many measures, including entropy, encryption time, and decryption time, a thorough study of the results is conducted. The suggested algorithm is thus validated by the cipher pictures.

A. Experimental setup

The suggested model is run on a laptop powered by an Intel Core i5-10300H CPU @ 2.50GHz, 8 GB of RAM, and a GPU module. The approach is implemented using Python modules such as PyTorch, pandas, OpenCV, and NumPy.

B. Datasets Details

The suggested approach has been assessed using chest X-ray images. These images were sourced from MedPix [18]. The dataset includes over 12,000 patient reports, 8,000 themes, and approximately 58,000 photographs. make up the publicly available open-source MedPix archive of medical imagery. All of the photos that were taken to assess the suggested plan are in 256 x 256 RGB format. The DeepKeyGen GAN model is developed by training it on the Pneumonia image dataset of chest X-rays. [19]. The suggested technique is compared to the three most recent medical picture encryption schemes, which are based on various schemes, including DL, chaos, and DNA-computing.

V. RESULT AND DISCUSSION

The proposed model assesses medical images by encrypted, and decrypted forms of the images. The results indicate that the attempt to decrypt was ultimately unsuccessful. Figure 3 shows the original images after the decryption process first attempted with the wrong key, followed by the appropriate key. The suggested method for generating keys is vulnerable to any modifications in the initial parameters.

A. Entropy Analysis

The entropy analysis module assesses the level of random- ness in encrypted images, which is an essential measure of security. Targeting values near 8 for 8-bit images, ensures that the encryption approach maximizes randomness, making the patterns in data less predictable and harder for attackers to decipher. High entropy readings confirm that the encryption method effectively obscures data patterns, adding an extra layer of defense against statistical attacks. Overall, this module validates the encryption's strength by showing that it achieves high randomness and minimizes vulnerabilities.

$$H=-i=0\sum 255P(i)\log 2P(i)\text{……….Eq.3}$$

B. Encryption Time

The encryption module secures the original image by converting it into a cipher image, using a sequence of operations and optimized libraries like PyTorch for efficiency. This transformation process involves XOR operations and shuffling to obfuscate image data, ensuring secure storage or transmission. The encryption time is measured to confirm the module's suitability for real-time applications, balancing the need for security with fast processing. Designed to be compatible with various hardware, this module focuses on achieving both high security and processing speed.

$$Tenc=Tend-Tstart\text{…………………..Eq.4}$$

C. Decryption Time

The decryption module reverses the encryption steps to accurately restore the original image derived from the cipher image. This module applies inverse operations, including XOR and reshuffling, to recover the data with precision. Optimized to mirror the encryption speed, it ensures decryption is fast enough for applications that require frequent access to the data. By maintaining efficiency and accuracy, the decryption module demonstrates its capability to retrieve the original content securely and without significant delay. Thus, the parameters are assessed using various encryption algorithms. such as chaotic maps, DES, AES, and blowfish. These algorithms are compared with performance metrics represented in Table 2.

$$Tdec=Tend-Tstart\text{……………....Eq.5}$$

Table 2: Performance Evaluation of PSO-based AES with other algorithms

Algorithm	Entropy	Encryption time	Decryption time
2DChaotic map (Henon Map)	7.98	0.12649	0.15668
DES	7.88	0.03261	0.04276
Blowfish	7.78	0.01354	0.02903
AES	7.99	0.00366	0.00510
AES-PSO	8.20	0.00355	0.00509

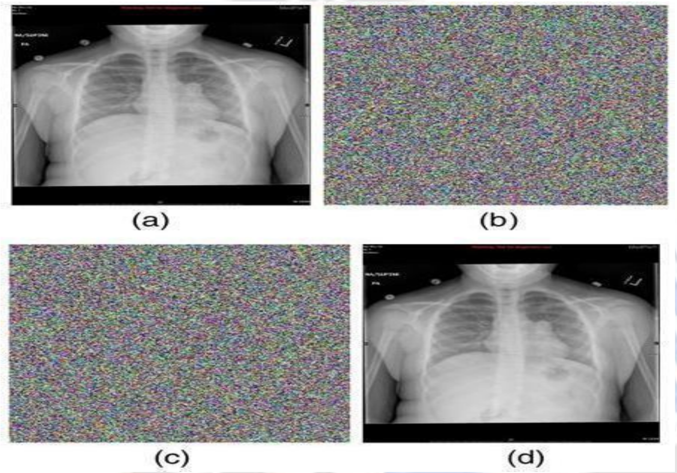

Fig 3. Representation as **a.**Initial input image, **b.**Encrypted image **c.**Minor adjustment in the decrypted original key generation parameters, **d.**Decrypted using the accurate key

VI. CONCLUSION

A novel algorithm designed for the encryption and compression of medical images, integrating GANs, a hash-based equation, and DHC, but with AES and PSO (Particle Swarm Optimization) used for key encryption instead of the He´non map. The algorithm generates a secure key in the form of an image, which is then used to drive the encryption process. Initially, diffusion and confusion are applied channel-wise to the original image. using PSO for shuffling, followed by XOR-based encryption applied with the private key, resulting in a fully encrypted version of the medical image. The image that has been encrypted is subsequently compressed using DHC before being passed to the server. The assessment is conducted using X-ray, ultrasound, and MRI images. using various metrics, including information entropy, encryption time, and decryption time. Experimental results indicate that this proposed model provides superior performance compared to other existing methods.

REFERENCES

[1] GutubA(2022)Boostingimagewatermarkingauthenticityspreading secrecy from counting-based secret-sharing. CAAI Transactions on Intelligence Technology.

[2] Hua Z, Zhou Y (2017) Design of image cipher using block-based scrambling and image filtering. Inf Sci 396:97–113.

[3] KumarP, AgrawalA(2015) GPU-accelerated interactive visualization of 3D volumetric data using CUDA. World Science-topic Journal of Image and Graphics 13:1340003–1340018.

[4] Hassan FS, Gutub A (2021) Improving data hiding within color images using hue component of HSV color space. CAAI Transactions on Intelligence Technology 7(1):56–68.

[5] DasS, NamasudraS(2022) Multi-authorityCP-ABE-based access control model for IoT-enabled healthcare infrastructure. IEEE Transactions on Industrial Informatics. https://doi.org/10.1109/TII.2022.3167842 (SCI, IEEE, ISSN: 1941- 0050) (IF: 10.215).

[6] KumarP, ParmarA(2020)VersatileApproachesforMedicalImage Compression: A Review. Procedia Computer Science 167:1380 1389.

[7] DingY, TanF, QinZ, CaoM, ChooK-KR, QinZ(2021)DeepKey Gen: A deep learning-based stream cipher generator for medical image encryption and decryption. IEEE Transactions on Neural Networks and Learning Systems, pp 1– 15.

[8] MurmuA, KumarP(2021) Deep learning model based segmentation of medical diseases from MRI and CT images. IEEE Int Conf on TEN- Auckland, New Zealand.

[9] Jin C, Wang T, Li X, Tie CJ, Tie Y, Liu S, Yan M, Li Y, Wang J, Huang S (2021)Atransformergenerative Adversarial Network for multi-track music generation. CAAI Transactions on Intelligence Technology.

[10] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2020) Generative Adversarial Networks. Commun ACM 63(11):139–144.

[11] Dagadu, J.C., Li, J.-P., and Aboagye, E.O. (2019) 'Medical image encryption based on hybrid chaotic DNA diffusion', Wireless Personal Communications, 108(1), pp. 591 612. doi:10.1007/s11277-019-06420- z.

[12] Ding Y, Tan F, Qin Z, Cao M, Choo K-KR, Qin Z (2021) DeepKeyGen: A deep learning-based stream cipher generator for medical image encryption and decryption. IEEE Transactions on Neural Networks and Learning Systems, pp 1–15.

[13] Amoh S, Zhang X, Chen G, Ueta T (2021) Bifurcation analysis of a class of generalized He´non maps with Hidden Dynamics. IEEJ Transactions on Electrical and Electronic Engineering 16(11):1456 1462.

[14] Magdy, M. et al. (2022) 'Security of medical images for telemedicine: A systematic review', Multimedia Tools and Applications, 81(18), pp. 25101–25145. doi:10.1007/s11042-022-11956-7.

[15] Mohanty, M.D.; Das, A.; Mohanty, M.N.; Altameem, A.; Nayak, S.R.; Saudagar, A.K.J.; Poonia, R.C. Design of Smart and Secured Healthcare Service Using Deep Learning with Modified SHA-256 Algorithm. Healthcare 2022, 10, 1275. [CrossRef].

[16] Singh, S. et al. (2023) 'Deep learning based medical image classification segmented with particle swarm optimization technique', 2023 OITS International Conference on Information Technology (OCIT) [Preprint]. doi:10.1109/ocit59427.2023.10430516.

[17] MedPix(2022)U.S.NationalLibraryofMedicine.[Online]. Available: https://medpix.nlm.nih.gov/home. Accessed 24 Mar 2022.

[18] Sarathambekai, S., and K. Umamaheswari. "Performance comparison of discrete particle swarm optimization and shuffled frog leaping algorithm in multiprocessor task scheduling problem." International Journal of Advanced Intelligence Paradigms 9, no. 2-3 (2017): 139-163.

[19] Vidya G, Sarathambekai S, Umamaheswari K, Yamunadevi SP. Task scheduling using constriction weighted particle swarm optimization for multi-objectives. Procedia Engineering. 2012 Jan 1;38:3049-55.

[20] Mooney P (2018) Chest X-ray images (pneumonia). Kaggle, 24-Mar-2018. [Online]. Available: https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia. Accessed 09 June 2022.