International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(1s) 2026 pp. 377-390
https://doi.org/10.70917/ijcisim-2026-2289
Article

A Symmetric Key Reversible Data Hiding Framework for Online Healthcare Systems using LSB Steganography and SHA-256

Sasmita Dash[1], Jayanta Mondal[1]*, Debabala Swain[2], Orlando Miguel Rodolfo[1], Debabrata Swain[3]

[1] School of Computer Engineering, KIIT University, Bhubaneswar, Odisha, India.
[2] Department of Computer Science, Rama Devi Women's University, Bhubaneswar, Odisha, India.
[3] Department of Computer Science and Engineering, Pandit Deendayal Energy University , Gandhinagar, Gujarat, India.
Corresponding Author: Jayanta Mondal, jayanta.mondalfcs@kiit.ac.in

Abstract: - This research presents a reversible encryption framework for the security of medical images while maintaining the integrity, quality and confidentiality in electronic patient records (EPR) transmission. It achieves integrity and quality preservation, lossless recovery, and robust security through hybrid architecture of cryptographic and steganographic techniques. Using reversible XOR based encryption and Least Significant Bit (LSB) steganography it makes sure to perfect reconstruct the medical image and Electronic Patient Record (EPR) without any loss, different from the traditional encryption methods that has potential to introduce distortions in compression. SHA-256 hashing can ensure integrity verification, detects any modification in transmission. The architecture ensures MSE = 0, PSNR = ∞, SSIM = 1, which means 100% of lossless recovery and perfect visual coherence. The dual layer encryption process—encrypting the medical image with a dynamic key stream followed by encrypting with using the EPR—provides security, imperceptibility and steganalysis resistance. Experimental results confirm that the architecture is suitable for medical imaging applications where precise diagnostic and data confidentiality in data is critical.

Keywords: Reversible Data Hiding, Medical Image Security, Lossless Recovery, XOR Encryption, LSB Steganography, Integrity Preservation, SHA-256 Hashing, Electronic Patient Record

1. INTRODUCTION

Medical images need strong security measures to protect against tampering and quality degradation [1]. Electronic patient record (EPR) holding confidential patient details, are actively being shared digitally and hence needs to be transmitted and storage in safety [2,3]. Conventional encryption sometimes doesn't maintain image quality that is important requirement for medical diagnosis. As a promising solution, reversible encryption techniques have been suggested, as a technique with capability to restore 100% images without loss of quality, thus, suitable for medical applications [3]. By comparing existing works, suggests a new approach, shows the effectiveness with the goal to improve safe medical image management. The usual steps of these techniques incorporate pre-processing, key generation, encryption and decryption to get back the original image. Reversible data hiding, watermarking and steganography usually are combined to improve security and quality retention [5]. Steganography especially LSB, Pixel Value Differencing (PVD) and Edge-Based Embedding (EBE), hides encrypted information within images, while cryptography provides data confidentiality. The use of the two approaches together presents an effective solution for secure data transmission of across open networks The use of the two approaches together presents an effective [6]. This scheme improves security, and is, therefore, viable in the security of sensitive medical information medical information.

2. EXISTING APPROACHES

Some research has suggested using steganography and cryptography to secure information. Paper [7], for instance, presents a system which encrypts patient data in terms of DNA-based AES, inserts the data in images with LSB steganography, and compresses stego images utilizing DWT. The decryption is performed at the receiving end to extract original data. It is concerned with patient data protection but not with cover file. Paper [8] a selective encryption scheme is presented for medical images, splitting the image into region of interest (ROI) and region out of ROI (ROB). Chaotic maps such as the Lorenz hyperchaotic system and the Henon chaotic map encrypt the ROI with the non-encrypted ROB. A binary image is inserted in the encrypted image using a key-controlled LSB algorithm and decryption that is lossless. High encryption is efficient and protection without safeguarding the cover image. Article [9] explores steganography and cryptography for more protection. The authors suggest integrating Huffman compression with elliptical curve cryptography or public-key cryptography to encrypt in order to achieve maximum confidentiality and minimizing distortion in images. The emphasis is on safeguarding the inserted information and not the cover file. In [10], a combined method of LSB steganography and AES encryption is proposed for encrypting text and digital images. LSB approach is used to insert secret text in a cover image and AES encryption on the stego image for more protection. The results are indicated by measuring the MSE and PSNR, highlighting higher security for the encrypted text. The cover image is not safeguarded. Paper [11] investigates encrypting messages using images with a combination of encryption algorithms such as AES, DES, RSA, and ChaCha20 with steganographic methods like LSB and Spread Spectrum, securing the hidden information, but not protecting the cover image. The performance is verified by combining PSNR, MSE, and information entropy. In article [12], to protect the medical images, a technique based on blockchain is presented and it's called a reversible data hiding (RDHEI) to achieve integrity, authenticity, traceability, and tamper-proof storage. The medical image is encrypted, while the confidential data is embedded by histogram alteration. The embedded data is the target and not the cover image. In [13], a steganography and encryption hybrid method based on RSA that is employed to encrypt the confidential message and LSB, DWT to embed it into the cover image that is not encrypted, meaning, it's vulnerable to attacks. The method has high PSNR and low MSE, and therefore low distortion in the stego image. Paper [14] proposes an integrated system of image steganography by embedding and hiding the stego file into the cover file and encrypting it using cryptography to ensure file sharing security.

3. PROPOSED FRAMEWORK

The framework uses RDH, steganography and cryptographic mechanisms in patient records and medical images employing a private key scheme for encryption and decryption of the data for secure transmission of sensitive data. The medical image is initially divided into 4×4 non-overlapping blocks, and each of the blocks is encrypted through a dynamic key stream generated from a 128-bit initial key through circular left shifts. A second layer of encryption is performed through XORing of the encrypted image with the Electronic Patient Record (EPR) and then the EPR is encrypted with the dynamic key stream. The encrypted image, EPR and a SHA-256 hash (for integrity verification) are embedded in a cover image through Least Significant Bit (LSB) steganography and is transmitted to the receiving end where the embedded data is recovered, decrypted, and reconstructed to the original image and EPR. The overall process is illustrated in Figure 1.

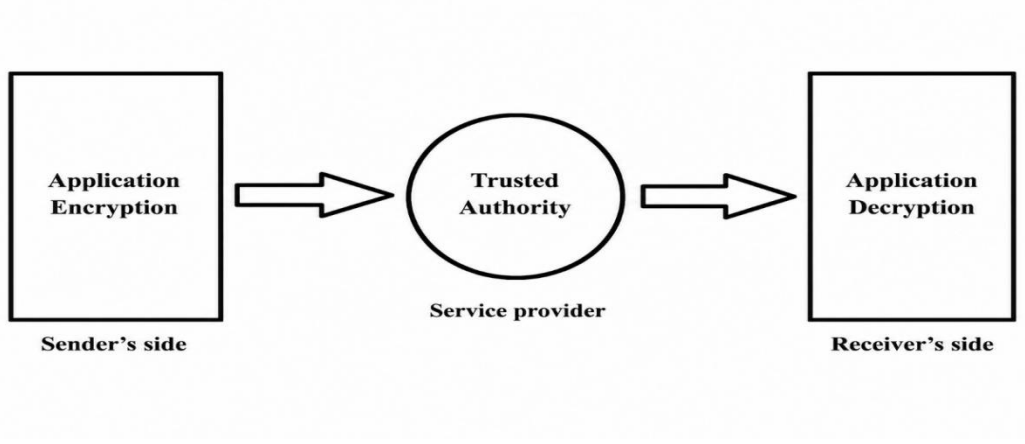

Figure 1: Trusted Authority based secure communication system

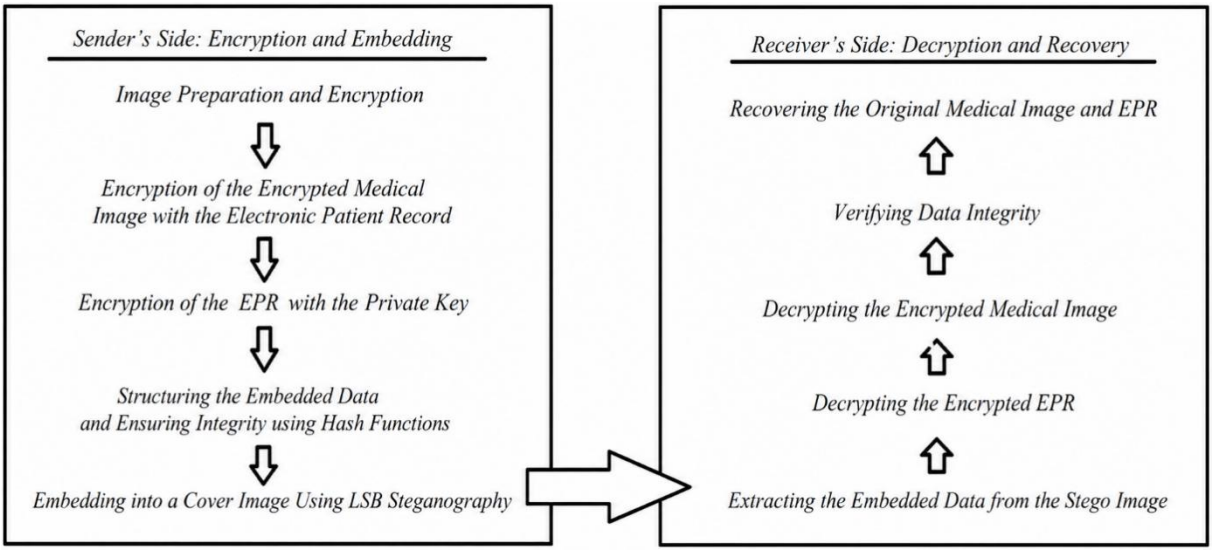

Figure 2: Proposed RDH Framework

4. PROPOSED ALGORITHM

The proposed Reversible Data Hiding architecture by using a private key infrastructure with a trusted authority manages the distribution of the private key, is able to secure transfer of confidential medical images and records. The key features of this architecture include: a design around a private key, the aspect of reversibility, the use of steganography, and image authentication method, and with all these features guarantee security, no data loss and integrity.

Figure 3: Proposed methodology scheme

4.1 *Sender's Side: Encryption and Embedding*

On the sender's side, the combining the cryptographic and steganographic principles is essential to achieve confidentiality, integrity, and reversibility.

4.1.1 Image Preparation and Encryption

The medical image is divided into 4×4 non-overlapping blocks to balance computational efficiency and security. Each block, comprising 16 pixels (1 pixel is 8 bits, 16 pixels are 128 bits total), undergoes bitwise XOR encryption with its corresponding dynamic key. XOR is chosen for its reversibility and computational simplicity, critical for maintaining the framework's lossless recovery property. The block-wise approach ensures localized encryption, minimizing the impact of potential data breaches.

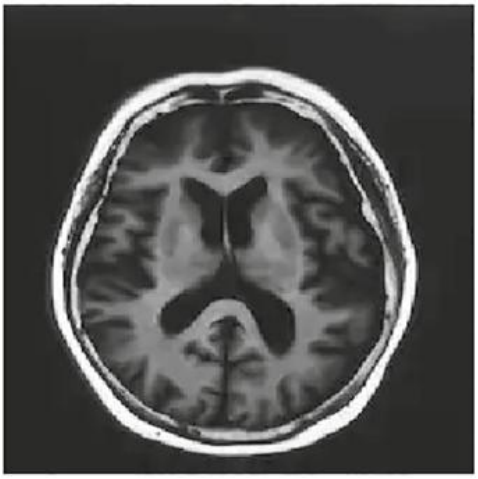

Figure 4: Brain Image of 256×256 pixels

a) **Block Division:**

i. This medical image is divided into 4×4 non-overlapping blocks (16 pixels/block).

ii. Total number of blocks: $\frac{256\times256}{4\times4} = 4096\ blocks$

b) **Block Encryption:**

i. *Each 128-bit block (16 pixels × 8 bits) is XORed with a unique 128-bit key (K1, K2, ..., Kn).*

ii. *After each block gets XORed with the different keys, the result will be the summation of all the blocks in order, resulting in I':*

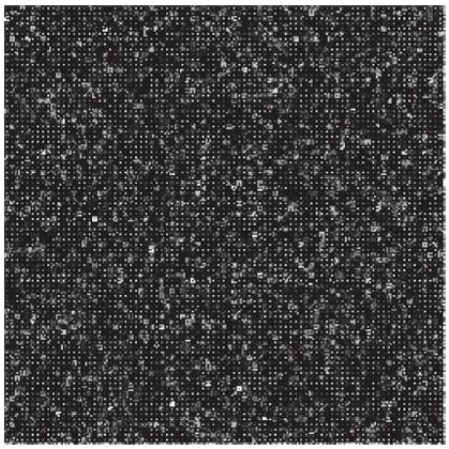

Figure 5: Medical image encrypted only with key (I′)

4.1.2 Encryption of the Encrypted Medical Image with the Electronic Patient Record

In this step, the encrypted medical image (I') undergoes a second layer of encryption by XORing it with the Electronic Patient Record (EPR). The objective of this step is to enhance security by ensuring that even if the medical image encryption is somehow broken, the data remains protected due to additional encryption with the patient's record.

EPR (Electronic Patient Record) containing textual or binary data, needs to be converted into a binary bitstream before it can be used in the encryption process because the EPR size might vary based on the patient data and to maintain structured encryption.

- *EPR Example:*

Name: *Jonathan Alexander Montgomery* ID: *XRT-789456239* Diagnosis: *Acute Respiratory Distress Syndrome (ARDS), Severe Pneumonia*

For demonstration, the EPR is converted to an 896-bit binary stream, and this means that the EPR need to be padded, because although it's divisible by the key size (128 bits), the encrypted medical image (I') it's not divisible by the EPR. The padding will be done to 1024 bits, the smallest number that satisfies the conditions needed to ensure the medical image encryption and the EPR encryption. The padding to 1024 bits it's done using PKCS#7 padding scheme because the padding can be easily removed during decryption by reading the last byte to determine the padding size to avoid any ambiguity.

a) EPR Preparation:

i. Convert EPR text to binary.
ii. Padding: Use PKCS#7 to align EPR to 128-bit blocks.
iii. If the EPR binary stream is not a multiple of 128 bits and the I' it's not divisible by the EPR, padding bits are added to satisfy the conditions.
iv. In this case, EPR is 896 bits, add 128 padding bits to make it 1024 bits.
v. Once the EPR has been adjusted to 1024 bits, it is ready to be used for XORing.

b) Dividing the Image into Blocks:

The encrypted medical image I' has a size of 256×256 pixels. Each pixel has 8 bits.

- Total bits in I': (256×256) × 8 = 524 288 bits
- Number of blocks (each block being 1024 bits in size): $\frac{(256\times256)\times8}{1024} = 512\ blocks$

Thus, the encrypted image I' is divided into 512 blocks, each 1024 bits long.

c) Second Level Encryption

Now the second level encryption of I' is performed as follows:

i. Take the first 1024-bit block from I'.
ii. XOR it with the 1024-bit EPR.
iii. Repeat this process for all 512 blocks, ensuring that each 1024-bit segment of I' is encrypted with the same 1024-bit EPR to obtain I".

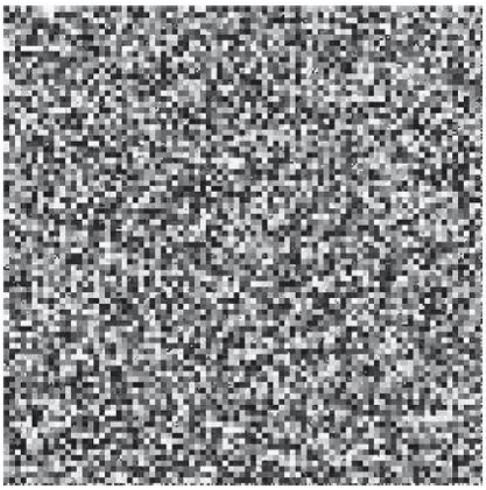

Figure 6: Final Encrypted Image (I")

4.1.3 Encryption of the EPR with the Private Key

The encryption of the EPR using the private key (K) ensures that the EPR remains protected independently, even if someone attempts to extract it from I".

a) Dividing the EPR into 128-bit Blocks

Since the private key K is 128 bits long, and we cannot XOR it directly with a 1024-bit EPR, we divide the EPR into 128-bit blocks: $\frac{1024\ bits}{128\ bits} = 8\ blocks$

Therefore, the EPR is divided into 8 separate blocks, each 128 bits long.

b) Encrypting EPR with K

Now, we perform the block-wise XOR operation using the key streams as follows:

i. First 128-bit block of EPR is XORed with K1 producing the First 128-bit block of EPR'.
ii. Second 128-bit block of EPR is XORed with K2 producing the Second 128-bit block of EPR'.
iii. Third 128-bit block of EPR is XORed with K3 producing the Third 128-bit block of EPR'.
iv. Repeat this for all 8 blocks, using a new key stream (Ki) each time.

After eight times XOR operation, the fully encrypted patient record (EPR') is obtained.

4.1.4 Embedding I" and EPR' into a Cover Image Using LSB Steganography

At this stage, we have:

- I" (Encrypted Medical Image after being XORed with the EPR): 524 288 bits;
- EPR' (Encrypted Electronic Patient Record after being XORed with the key K): 1024 bits.

Now, these encrypted components must be securely embedded into a cover image using Least Significant Bit (LSB) steganography. This ensures that the transmission remains secure and imperceptible to unauthorized parties.

The cover image should be larger than I" and EPR' to accommodate the hidden data. This step also includes authentication markers to verify integrity upon extraction.

Choosing the Cover Image

- The cover image (CI) serves as a host for I" and EPR', making them indistinguishable from a regular image.
- The cover image should be large enough to store all encrypted data.
- For this demonstration, we assume a 726 × 726 grayscale image, which contains: 726×726×8=4 216 608 bits (total capacity)

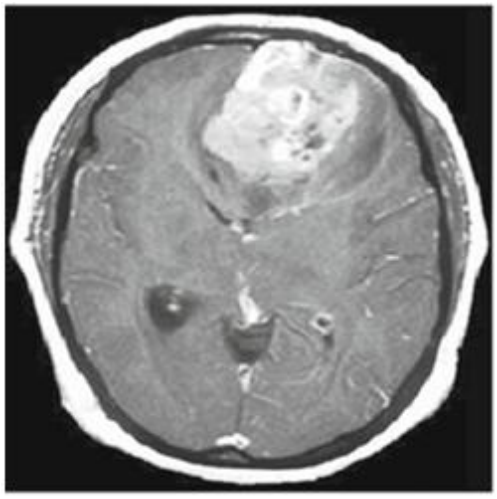

Figure 7: Cover image

4.1.5 Structuring and Marking Process for Data Embedding

We introduce fixed markers to define the boundaries of the encrypted data and all the being:

a) Header Information (Fixed markers to guide decryption)
   i. Message Starts: Marks the start of embedded data.
   ii. I" Ends & EPR' Starts: Indicates the transition between the encrypted image and the encrypted patient record.
   iii. EPR' Ends & Hash Starts: Indicates the transition between the encrypted patient record and the hash.
   iv. Hash Ends: Indicates the end of the hash and of the message.
   v. The markers are estimated at 584 bits, that will be used throughout this study.
   vi. These markers help in identifying tampering, if they are missing or incorrect, the data is considered tampered, and decryption is aborted.

b) Main Data (Encrypted Content)
   i. I" (Medical Image encrypted with EPR)
   ii. EPR’ (EPR encrypted with the private key K)

c) Hash (for integrity verification)
   i. H: A 256-bit cryptographic hash computed as H = SHA-256(I″ + EPR′), serving as an integrity fingerprint.

d) Preparing the Data for Embedding

Before embedding into the cover image, we prepare the data by adding authentication markers and applying an additional XOR with K to enhance security.

The final data sequence to be embedded in each block will be: Message Starts+I"+I" Ends & EPR' Starts+EPR'+EPR' Ends & Hash Starts+H+Hash Ends.

4.1.6 Integrity Verification using Hash Function

i. Concatenate (+) I" and EPR' into a single bitstream: D = I"+EPR'… (1)
ii. Compute SHA 256 hash: H=SHA 256(D)… (2)
iii. Structure the final data (N) for embedding:
- N=Message Starts+I"+I" Ends & EPR' Starts+EPR'+EPR' Ends & Hash Starts+H+Hash Ends. Where, N is total length of the message being sent.
- The total embedded data must be divisible by the key size:

$$\frac{N}{K} = integer \ldots (3)$$

$$M=N+Padding \ldots (4)$$

where M is the final data length (after padding, if necessary) and K the key size.

- If N is not a perfect divisor, we add padding bits until divisibility is achieved.

iv. XOR with K: Before embedding, M is XORed with K, ensuring extra security.

- K is a 128-bit private key
- M is divided into segments of 128 bits
- Each 128-bit segment of M is XORed with a different shifted version of K:
  - First 128 bits of $M \oplus K_1$
  - Second 128 bits of $M \oplus K_2$
  - Third 128 bits of $M \oplus K_3$
  - Continue up to $M \oplus K_n$

This ensures that even if someone obtains the cover image, they cannot extract I" or EPR' without the K.

4.1.7 Data Embedding into the Cover Image

The embedding capacity of the cover image depends on the number of pixels and the number of LSBs used per pixel.

$$\text{Embedding Capacity} = M \times N \geq L \ldots (5)$$

Where:

- M×N: The dimensions of the cover image (in pixels).
- L: The size of the secret message (in bits).

For a grayscale image of size M×N and 1 LSB per pixel, the total embedding capacity is:

$$\text{Capacity} = M \times N \text{ bits} \quad \ldots (6)$$

Steps for Embedding Data into the Cover Image:

1. Prepare the Data for Embedding:

The total data size is 526 189 bits. Padding is added to make the data size divisible by 128 bits, resulting in a final size of 526 208 bits.

2. Divide the Data into 1-Bit Segments:

The data is divided into individual bits for embedding into the LSBs of the cover image pixels.

3. Embed the Data into the Cover Image:

For each bit s_i in the secret data:

- Select the next pixel P in the cover image.
- Replace the LSB of P with s_i using the LSB embedding formula.
- Store the modified pixel P' in the stego image.

4. Handle Unused Pixels:

If the cover image has more pixels than required for embedding, the remaining pixels are left unchanged.

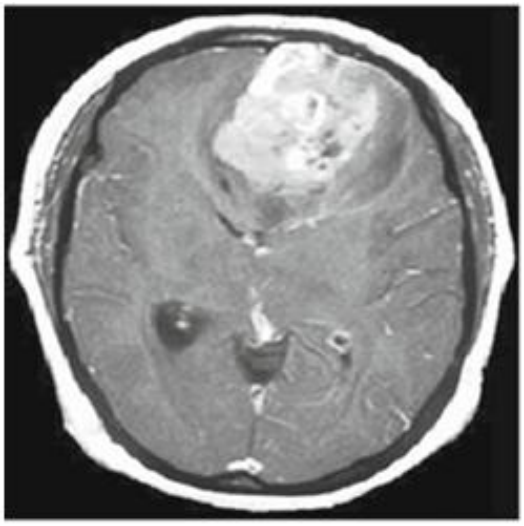

Figure 8: Cover image after embedding data

4.2 Receiver's Side: Decryption and Recovery

The receiver's side is responsible for extracting the embedded data from the stego image, decrypting it, and recovering the original medical image and Electronic Patient Record (EPR).

4.2.1 Extracting the Embedded Data from the Stego Image

The Least Significant Bit (LSB) steganography technique is used to extract the embedded data from the stego image.

1. Stego Image Representation
   The stego image C is a grayscale image of size 726×726 pixels. Each pixel is represented by 8 bits, and the LSB of each pixel contains a bit of the embedded data.

2. Extracting the Data
   - Initialize an empty bitstream S.
   - For each pixel P' in the stego image:
     - Extract the LSB of P':

$$S_i = P' \bmod 2 \quad \ldots (7)$$

     - Append S_i to the bitstream S.
   - Continue until all embedded data bits are extracted.

3. Reconstructing the Embedded Data
   The extracted bitstream S contains the following components in order:

- Markers: 584 bits (to identify the start and end of each component).
- Encrypted Medical Image (I''): 524 288 bits.
- Encrypted EPR (EPR′): 1024 bits.
- Hash H: 256 bits.
- Padding: 37 bits.

4. Removing Padding
   - The last 37 bits of S are padding and are discarded.
   - The remaining data is divided into its components using the markers.

4.2.2 Decrypting the Encrypted EPR (EPR')

The encrypted EPR (EPR′) is decrypted using the private key K and the key stream derived from it.

1. Dividing EPR′ into 128-bit Blocks:
   - EPR′ is divided into 8 blocks of 128 bits each:

$$EPR'=EPR'1+EPR'2+\ldots+EPR'i\ldots \quad (8)$$

2. Decrypting Each Block:
   - For each block EPR'_i , perform XOR with the corresponding key K_i :

$$EPRi=EPR'i \oplus Ki \ \ldots \quad (9)$$

   - Repeat for all 8 blocks to recover the original EPR.
3. Removing PKCS#7 Padding:
   - The last byte of the decrypted EPR indicates the number of padding bytes added.
   - Remove the padding bytes to recover the original 896-bit EPR.

4.2.3 Decrypting the Encrypted Medical Image

a) Decrypting the Image I''

For decryption of I'', the encrypted medical image I'' is decrypted using the EPR and the key stream derived from K.

1. Dividing I'' into 1024-bit Blocks:
   - I'' is divided into 512 blocks of 1024 bits each:

$$I''=I''1+I''2+\ldots+I''i \ \ldots \quad (10)$$

2. Decrypting Each Block:
   - For each block I_i'', perform XOR with the 1024-bit EPR:

$$I'i = I''i \oplus EPR\ldots \quad (11)$$

   - Repeat for all 512 blocks to recover I'.

b) Decrypting the Image I'

1. Dividing I' into 128-bit Blocks:
   - I' is divided into 4,096 blocks of 128 bits each:

$$I'=I'1+I'2+\ldots+I'i \ \ldots \quad (12)$$

2. Decrypting Each Block:

- For each block I'_i , perform XOR with the corresponding key Ki:

$$Ii = I'i \oplus Ki \ldots (13)$$

- Repeat for all 4,096 blocks to recover the original medical image I.

4.2.4 Verifying Data Integrity

Hash Verification: The hash H is used to verify the integrity of the extracted data.

$$\text{Concatenate } (+) \ I'' \text{ and } EPR': D = I'' + EPR' \ldots (14)$$

1. Compute the Hash:

$$H' = SHA\ 256(D) \ldots (15)$$

2. Compare Hashes:
   - If $H'=H$, the data is intact and has not been tampered with.
   - If $H' \neq H$, the data has been altered, and the process is aborted.

4.2.5 Recovering the Original Medical Image and EPR

1. Recovering the Medical Image:
   - The decrypted medical image I is reconstructed by combining the 4,096 decrypted blocks.
   - The image is normalized to ensure pixel values are in the range 0–255.
2. Recovering the EPR:
   - The decrypted EPR is converted from binary to its original textual format.
   - The patient's medical record is displayed for verification.

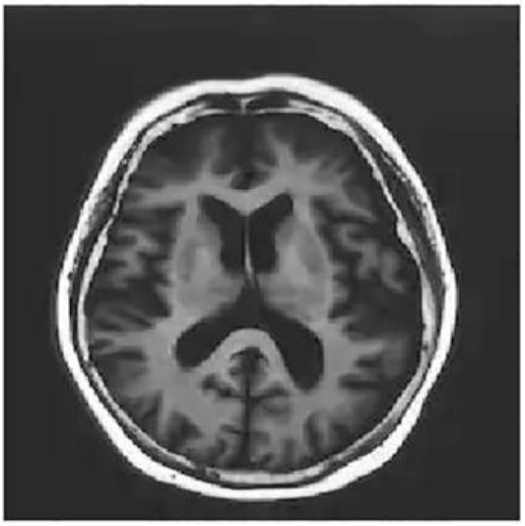

Figure 9: Recovered medical Image is similar to Figure 4

5. COMPARATIVE ANALYSIS

Study	Technique	Test Image	Metrics
[7]	The method combines DNA-based AES encryption with LSB steganography to secure medical images. The encrypted image is embedded into a cover image, creating a stego image for transmission. Upon receipt, the process is reversed: the encrypted image is extracted, decrypted, and the original image is restored. Compression techniques are applied to improve efficiency, functionality, and security.	Medical images	MSE= 0.0215 PSNR= 64.8082 dB
[8]	An algorithm partitions the image into ROI (that uses chaotic maps to encrypt) and ROB (that is combined with the ROI) using LoG edge detection and boundary tracking, after it, LSB is used to insert the ROI in the encrypted image.	Medical images	SSIM= 0.0142
[5]	A pseudorandom key is generated and shared securely between sender and receiver, Gamma ciphers leveraging XOR operations for encryption, LSB to embed encrypted data into pixels selected randomly. On the receiver's end, the data is extracted and decrypted.	Medical images	MSE= 0.0001 PSNR= 97.455 dB
[11]	The proposed blockchain-based RDHEI system improves information transfers integrity, traceability for medical images that is encrypted and embedded into a cover image and is stored in the blockchain. Recipients verify the integrity by comparing the hash of the received cover image with the blockchain-stored hash and decrypt it using authorized keys.	Medical images (256 × 256)	MSE= 0.0064 PSNR= 70.1dB
[12]	This work embeds ciphertext in images combining AES, DES, RSA, and ChaCha20 with LSB and Spread Spectrum. The process entails encryption of plaintext with a 256-bit key generated through SHA-256, embedding the ciphertext in a cover image, and extracting and decrypting the message.	Coloured images	MSE= 0.1753 PSNR= 55.6918 dB
[13]	This study employs a hybrid approach of LSB to embed a secret text into a cover image to create a stego image and AES to encrypt the stego image. The MSE and PSNR metrics are used to measure the quality.	Coloured images	MSE= 0.0019922 PSNR= 75.1375 dB
[14]	This method encrypts the message using AES and Blowfish and encryption keys, and then it gets split, compressed and, using LSB, it's inserted across the cover image that it's also divided into 25% for the message and 25% for sample verification. The quality is measured using PSNR.	Coloured images (500 × 500)	MSE= 0.00019995 PSNR= 85.1216 dB
[15]	This approach involves loading a cover image, involves LSB substitution, involves using RSA to	RGB format images,	MSE= 1.1947

	encrypt the message, and involves concealing the encrypted message in the image. The modified image is embedded in the cover image using the assistance of DWT. The encrypted image is decrypted and extracted at the receiver with the assistance of DWT. Performance is measured by PSNR, MSE, and entropy.	turned into grayscale (adjusted to 512× 512)	PSNR= 72.3243 dB
Proposed Method	This work divides the medical image into pixels (4×4) squares that don't intersect, encrypts it with XOR using a dynamic key stream. The encrypted image is double-XOR encrypted with the Electronic Patient Record. The result is embedded into a cover image using LSB, with authentication markers and SHA-256 hash.	Medical images (256 × 256)	MSE=0 PSNR=∞ SSIM=1

Table-1: Comparative Analysis of Proposed Technique with Existing Techniques

6. CONCLUSION

This research presents a strong, efficient reversible data hiding method for the purpose of securing medical images through the combination of cryptographic and steganographic methods, with lossless recovery, integrity, quality, and rigorous security controls. Utilizing reversible XOR-based encryption and SHA-256 hashing, the presented framework ensures complete recoverability of the original medical image and Electronic Patient Record (EPR) without loss of information or degradation, as demonstrated by MSE = 0, PSNR = ∞, and SSIM = 1. In contrast to traditional encryption methods like RSA or AES, which can introduce distortion due to compression, the presented method utilizes XOR encryption and least significant bit (LSB) steganography to ensure visual imperceptibility and fidelity. Utilization of a double-layer encryption scheme, dynamic key stream, and key-controlled embedding ensures strength against unauthorized access and possible attacks, while utilization of authentication markers and integrity checks ensures authenticity of the data. This technique is paramount in medical imaging where diagnostic accuracy and integrity of data are paramount, with the research demonstrating superiority over traditional techniques in reversibility, computational complexity, and overall security. Future research can explore more complex steganographic methods and machine learning methods to further enhance the functionality of the framework.

References

1. H. Kolivand, S. F. Hamood, S. Asadianfam, and M. S. Rahim, "Image encryption techniques: A comprehensive review," Multimedia Tools and Applications, 2024, doi: 10.1007/s11042-023-17896-0.
2. S. T. Ahmed, D. A. Hammood, R. F. Chisab, A. Al-Naji, and J. Chahl, "Medical image encryption: A comprehensive review," Computers, vol. 12, no. 8, p. 160, 2023, doi: 10.3390/computers12080160.
3. Priyanka and A. K. Singh, "A survey of image encryption for healthcare applications," Evolutionary Intelligence, vol. 16, pp. 801–818, 2023, doi: 10.1007/s12065-021-00683-x.
4. N. S. Noor, D. A. Hammood, A. Al-Naji, and J. Chahl, "A FastText-to-image encryption-decryption algorithm for secure network communication," Computers, vol. 11, no. 3, p. 39, 2022, doi: 10.3390/computers11030039.
5. A. Nikishova, M. Umnitsyn, and O. Kakorina, "Steganography techniques for encrypted cover file," in Proc. 2024 International Russian Smart Industry Conference (SmartIndustryCon), Sochi, Russian Federation, 2024, pp. 834–839, doi: 10.1109/SmartIndustryCon61328.2024.10515569.
6. E. Onyedinma and I. Onyenwe, "Two-tier data security technique: LSB image steganography and columnar transposition cipher," International Journal of Innovative Research in Computer and Communication Engineering, vol. 12, no. 4, 2024, doi: 10.15680/IJIRCCE.2024.1204001.
7. F. Khallaf, W. El-Shafai, E. M. El-Rabaie, and F. E. A. El-Samie, "Reinforcement of medical image security through steganography and encryption techniques," in Proc. 3rd International Conference on Electronic Engineering (ICEEM), Menouf, Egypt, 2023, pp. 1–4, doi: 10.1109/ICEEM58740.2023.10319514.
8. L. Zhang, X. Song, A. E. Ahmed, Y. Zhao, and A. Bassem, "Reversibly selective encryption for medical images based on coupled chaotic maps and steganography," Complex & Intelligent Systems, 2023, doi: 10.1007/s40747-023-01258-2.

9. V. Kumar, G. Singh, and B. Singh, “A comparative study of various lossless compression techniques of steganography and cryptography,” in Proc. International Conference on Computer and Communication Systems (ICCS), 2021, pp. 285–288, doi: 10.1109/ICCS54944.2021.00063.
10. L. H. Gong and H. X. Luo, “Dual color images watermarking scheme with geometric correction based on quaternion FrOOFMMs and LS-SVR,” Optics and Laser Technology, vol. 167, p. 109665, 2023.
11. J. Horng, C.-C. Chang, G.-L. Li, W. K. Lee, and S. Hwang, “Blockchain-based reversible data hiding for securing medical images,” Journal of Healthcare Engineering, vol. 2021, pp. 1–22, 2021, doi: 10.1155/2021/9943402.
12. M. Kataria, K. Jain, and N. Subramanian, “Exploring advanced encryption and steganography techniques for image security,” in Proc. 11th International Symposium on Digital Forensics and Security (ISDFS), Chattanooga, TN, USA, 2023, pp. 1–6, doi: 10.1109/ISDFS58141.2023.10131890.
13. M. Kumar, A. Soni, A. R. S. Shekhawat, and A. Rawat, “Enhanced digital image and text data security using hybrid model of LSB steganography and AES cryptography technique,” in Proc. 2nd International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2022, pp. 1453–1457, doi: 10.1109/ICAIS53314.2022.9742942.
14. M. Alanzy, R. Alomrani, B. Alqarni, and S. Almutairi, “Image steganography using LSB and hybrid encryption algorithms,” Applied Sciences, vol. 13, no. 21, p. 11771, 2023, doi: 10.3390/app132111771.
15. S. Bhargava and M. Mukhija, “Hide image and text using LSB, DWT and RSA based on image steganography,” ICTACT Journal on Image and Video Processing, vol. 9, pp. 1940–1946, 2019, doi: 10.21917/ijivp.2019.0275.