International Journal of Advances in Scientific Research and Engineering (ijasre)

E-ISSN : 2454-8006

DOI: 10.31695/IJASRE.2025.10.7

Volume 10, Issue 10
October- 2025

Secure Image Steganography Using AES-CBC Encryption and Dynamic Key Derivation from Image Features

Mohammed Fadhil Kamil[1], Ali Qasim Mohammed[2]

Information Technology Centre

Mustansiriyah University

Iraq

ABSTRACT

This research presents a technique for concealing messages within images by combining AES encryption in Cipher Block Chaining (CBC) mode with Least Significant Bit (LSB) embedding. A key feature of this method is the dynamic generation of the encryption key based on intrinsic statistical properties of the image—specifically, the standard deviation of the RGB channels— This eliminates the necessity for external key exchange, making the encryption process inherently dependent on the image itself. The secret message is first encrypted using AES-CBC, and the resulting ciphertext is embedded into the image using LSB manipulation. The proposed approach was evaluated using standard image quality metrics. Experimental results reveal excellent preservation of visual quality, with a Mean Squared Error ***(MSE) of 0.000091****, **Mean Absolute Error (MAE) of 0.000091**, Peak Signal-to-Noise Ratio* ***(PSNR) of 88.53*** *dB, Structural Similarity Index Measure (**SSIM) of 1.000000**, and a coefficient of determination* ***(R^2) of 1.000000****. A total of 336 pixel-level changes were detected, all strictly confined to the LSBs, resulting in a **100% LSB-only** modification ratio. These outcomes confirm that the proposed method ensures both strong security and the imperceptibility of the hidden data.*

Key Words: AES-CBC Encryption, Cryptography, Digital Image Processing, Feature-Based Key Derivation, Dynamic Key Generation, LSB Embedding, Image Security, Image Steganography.

1. INTRODUCTION

Image steganography is a strong technique for securing ticklish information by embedding secret data within digital image in a way that is imperceptible to the human eye [1].Applying image steganography using this method leads to the ability to hide confidential messages without raising suspicion, making it an essential tool in secure communication systems [2][3]

Recent advancements in steganographic techniques [1] focus on improving the balance between capacity, imperceptibility, and robustness to resist detection and unauthorized extraction [4], using adaptive embedding and randomization methods [5].

On the other hand, encryption plays a critical role in protecting the privacy and integrity of data during transmission and storage [6]. Among encryption algorithms, the Advanced Encryption Standard (AES) operating in Cipher Block Chaining (CBC) mode has become widely adopted due to its strong security properties and efficiency [7][8].

AES-CBC ensures that even if data is intercepted, it remains unintelligible without the secret key, adding a fundamental layer of protection [9].

In practice, AES-CBC encryption typically follows a workflow where the initialization vector (IV) is randomly generated and prefixed to the ciphertext, and PKCS#7 padding is applied to align the plaintext with the block size. This standard approach ensures both confidentiality and compatibility during decryption[10].

Integrating steganography and encryption techniques offers a hybrid solution that enhances data security by combining the strengths of both [7]. In such systems, sensitive messages are first encrypted using robust algorithms like AES-CBC and then embedded into cover images via steganographic methods such as Least Significant Bit (LSB)

manipulation [11]. This dual-layer security approach not only conceals the presence of the message but also ensures its confidentiality [12][13].

Recent research has suggested hybrid frameworks that aim to enhance resistance to secret analysis and unauthorized access attempts, by using mechanisms to generate encryption keys dynamically based on image characteristics, such as the arithmetic mean, standard deviation, or edge data [6][14].

Additionally, researchers are hiding data in parts of an image our eyes rarely notice-like soft, low-contrast areas-so the secret message stays virtually invisible and becomes more resistant to tampering [12][14]. Lightweight steganographic methods have also been proposed to meet the demands of resource-constrained environments like the Internet of Things (IoT), enabling secure and efficient communication without compromising performance [15].

Overall, the convergence of encryption and steganography [7], supported by adaptive key generation [16] and intelligent embedding , continues to drive the development of secure, imperceptible, and robust data hiding schemes suitable for a wide range of applications including healthcare, corporate communication, and digital forensics[17].

2. .RELATED WORKS

Sulaman et al. [**16**] created a framework for medical image encryption that combines the Radon transform and AES-CBC to shield private images from unwanted access while maintaining diagnostic quality. Their approach encrypts the picture, uses the Radon transform to turn it into a sinogram, and then uses the inverse transform to rebuild it. When tested on a range of medical images, including chest X-rays, it demonstrated strong security and high-quality reshaping **with PSNRs of up to 64.01 dB, MSEs as low as 0.0258, and SSIM values as high as 0.9999.**

Hakim et al. [18]proposed a secure steganographic method combining Two Least Significant Bits (2LSB) embedding with AES-256 encryption to enhance data confidentiality and capacity. The method was tested on images with resolutions of 256×256, 512×512, and 1024×1024, achieving PSNR values ranging from 46.40 to 49.65 dB, while the mean square error (MSE) value ranged from 0.0012 to 0.0023. Since all PSNR values exceeded 40 dB, it maintained a high technical standard for image quality. MD5 hash tests before and after extraction confirmed message integrity, validating the 2LSB method's ability to preserve both image quality and embedded data accuracy.

Rashad J. Rasras et al. [19] proposed an enhanced LSB-based steganography method using a secret key to set the embedding start point and a secret index key (SIK) for bit permutation. Tests showed **MSE values of 0.00006–0.1909 and PSNR of 125.92–206.06 dB,** with a correlation coefficient (CC) of 1.0, indicating perfect similarity. The Number of Byte Change Rate (NBCR) stayed below 0.0063 for short messages and under 20% for messages up to 150,000 characters, demonstrating high image quality, robustness, and resistance to unauthorized extraction.

Dilara Şener and Selda Güney[20] An image steganography method that was suggested by one of the studies was an algorithm based on the XOR operation, LSB algorithm, to encrypt the secret key and then use the encrypted key to hide information in the image. This solution attempt to increase security at the same time that image quality does not fall below acceptable levels. Standard measures of performance like Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR) were used to measure performance. The findings indicated that, the PSNR values were between 49.45 and 54.10 dB and MSE values were between 0.0025 and 0.0057, and varied according to the image type and embedding parameters. The results show that the technique gives an appropriate trade-off between invisibility and embedding capacity with a lightweight encryption strategy.

Mohammed Sabri Abuali et al [21]**.** proposed a multi-level steganographic technique that integrates Dynamic Least Significant Bit (DLSB) embedding with the Wavelet Obtained Weights (WOW) algorithm to enhance both security and visual quality. The DLSB technique adapts the number of LSBs used for embedding based on the local contrast of the image, while WOW operates in the wavelet domain to minimize statistical distortions. The proposed method was evaluated using multiple image quality metrics. The general LSB approach achieved a **PSNR of 79.66 dB, SSIM of 0.99, BER of 8.78×10^{-5}, and MSE of 1.081×10^{-8},** indicating strong fidelity. In contrast, the optimized DLSB method reached even higher performance, with **a PSNR of 81.24 dB, SSIM of 0.999, and MSE as low as 7.5098×10^{-9} at a threshold of 15**

DOI: 10.31695/IJASRE.2025.10.7

3 METHODOLOGY

This study proposes a strong image steganography framework that integrates AES-CBC encryption with LSB embedding . A key innovation of this approach is the use of a dynamic, image-dependent encryption key generated from the standard deviation (STD) of the cover image, which ensures uniqueness per image and eliminates the need for external key exchange.

3.1 Key Generation from Image Features

The strength of any symmetric encryption algorithm, such as AES-CBC, lies in the secrecy and randomness of its encryption key. In this work, instead of relying on a fixed or user-defined key, the key is dynamically generated from intrinsic features of the cover image itself, ensuring that the same image must be present during both encryption and decryption.

3.2 Key Generation via Standard Deviation

To generate a secure and reproducible key, we treat the standard deviation of each RGB channel as a unique statistical identifier of the image. These deviations are calculated spatially across all pixel intensities, concatenated, and then hashed using SHA-256 to produce a fixed-length 256-bit key suitable for AES encryption[14].

$$\sigma = \sqrt{\frac{1}{wH}\sum_{i=1}^{w}\sum_{j=1}^{H}\left(p_{ij\cdot c} - \mu\right)^2} \quad , \quad c\in\{R,G,B\} \qquad \ldots\ldots\ldots (1)$$

Where:

- σ = Standard deviation for channel c
- $p_{ij\cdot c}$ = Pixel value at row j, column i in channel c
- μ = Mean pixel value for channel c
- w, H = Width and height of the image

In our proposed method, we derive the encryption key by concatenating the standard deviations of the R, G, and B channels and hashing the result using SHA-256, as inspired by the adaptive digest-based key generation approach described in (2) .

Key = SHA256 (STDR ‖ STDG ‖ STDB) ……. (2)

Key Generation Flow: As illustrated in the diagram (Figure 1), the process begins with the input image, from which statistical features (standard deviations of RGB channels) are extracted. These features are converted into bytes, hashed using SHA-256, and finally used to produce a 256-bit AES encryption key.

3.3 AES-CBC Encryption

The plaintext message is encrypted using AES in Cipher Block Chaining (CBC) mode. The encryption process involves the random generation of a 16-byte Initialization Vector (IV), the application of PKCS#7 padding, and the transformation of the padded plaintext into ciphertext.

In our implementation, we adopt the standard AES-CBC formulation, as shown in[22] :

C=AES-CBCK(P ‖ padding , IV) …….. (3)

where **P** is the plaintext, **K** is the derived key from the image features, and **IV** is a randomly generated initialization vector.

DOI: 10.31695/IJASRE.2025.10.7

.

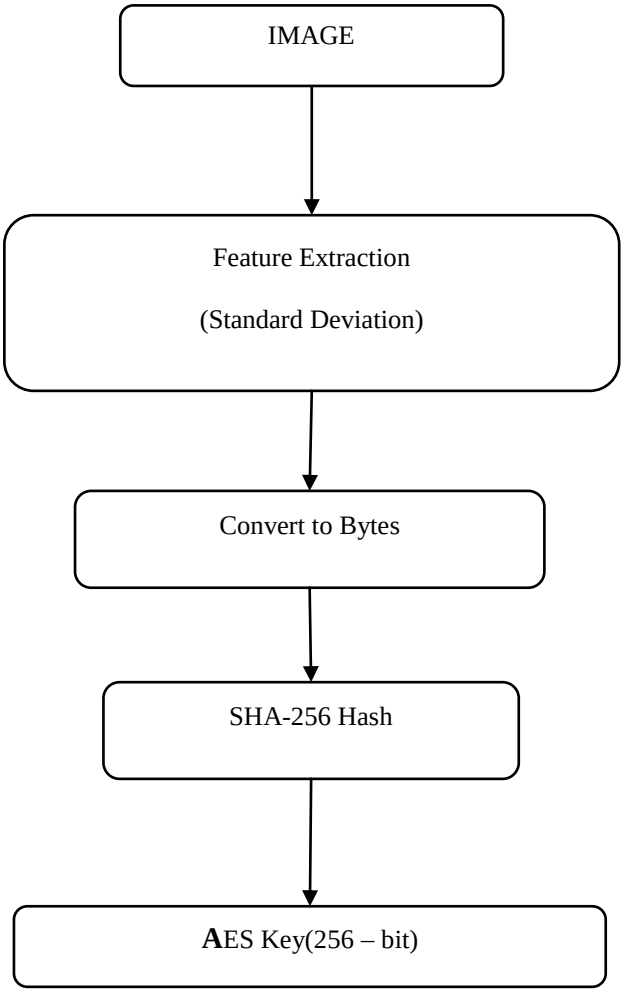

Figure 1: Key derivation process from image features using standard deviation and SHA-256 hash.

3.4 Bit Permutation and LSB Embedding

The resulting ciphertext is converted into a binary stream and then permuted using a pseudo-random shuffle based on a fixed seed. This step enhances the confusion property and mitigates statistical attacks by unpredictably redistributing bit positions. Following the permutation is mathematically represented as[22] :

PermutedBits= Shuffle (Binary (C) , seed) …… (4)

A 32-bit header indicating the binary message length is then prepended to the permuted bits, forming the final payload. This payload is embedded into the Least Significant Bits (LSBs) of the cover image pixels. The stego-image is saved in PNG format to ensure the **lossless** nature of the embedding process.

3.5 Encryption – AES-CBC and LSB Steganography

The encryption and hiding process in the proposed system follows a structured sequence of steps that ensure data confidentiality and imperceptibility. The workflow integrates dynamic key generation from the image, AES-CBC encryption, bitstream permutation, and LSB embedding.

3.5.1 Main Encryption Steps:

The encryption process begins with input cover image selection, where the user chooses an image file (e.g., input.png) to serve as the host for hidden data, which also acts as the source for key generation. In the dynamic key generation (STD-based) step, the standard deviation of each RGB channel is computed, and these three STD values (σR , σG , σB) are concatenated and hashed using SHA-256 to produce a unique 256-bit AES key.

DOI: 10.31695/IJASRE.2025.10.7

This ensures that the key is deterministic and reproducible as long as the original image remains unchanged. Next, in plaintext encryption using AES-CBC, the plaintext message is padded using PKCS#7 to meet AES block size requirements, a random 16-byte Initialization Vector (IV) is generated, AES encryption in CBC mode is applied using the derived key and IV, and the IV is prepended to the ciphertext for decryption purposes.

The conversion to binary stream step follows, where the encrypted data (IV + ciphertext) is transformed into a stream of bits. Then, during bit permutation (obfuscation), a pseudo-random permutation is applied to the bitstream using a user-provided seed, enhancing security by randomizing bit positions and adding confusion. In payload construction, a 32-bit binary header representing the bitstream length is prepended to the permuted data, forming the final payload for embedding. The LSB embedding stage involves inserting each bit from the payload into the least significant bit of the image pixels (RGB channels) sequentially in row-major order, with only one bit per channel modified to maintain imperceptibility. Finally, the output stego image is saved in PNG format (e.g., output.png) to prevent data loss due to compression.

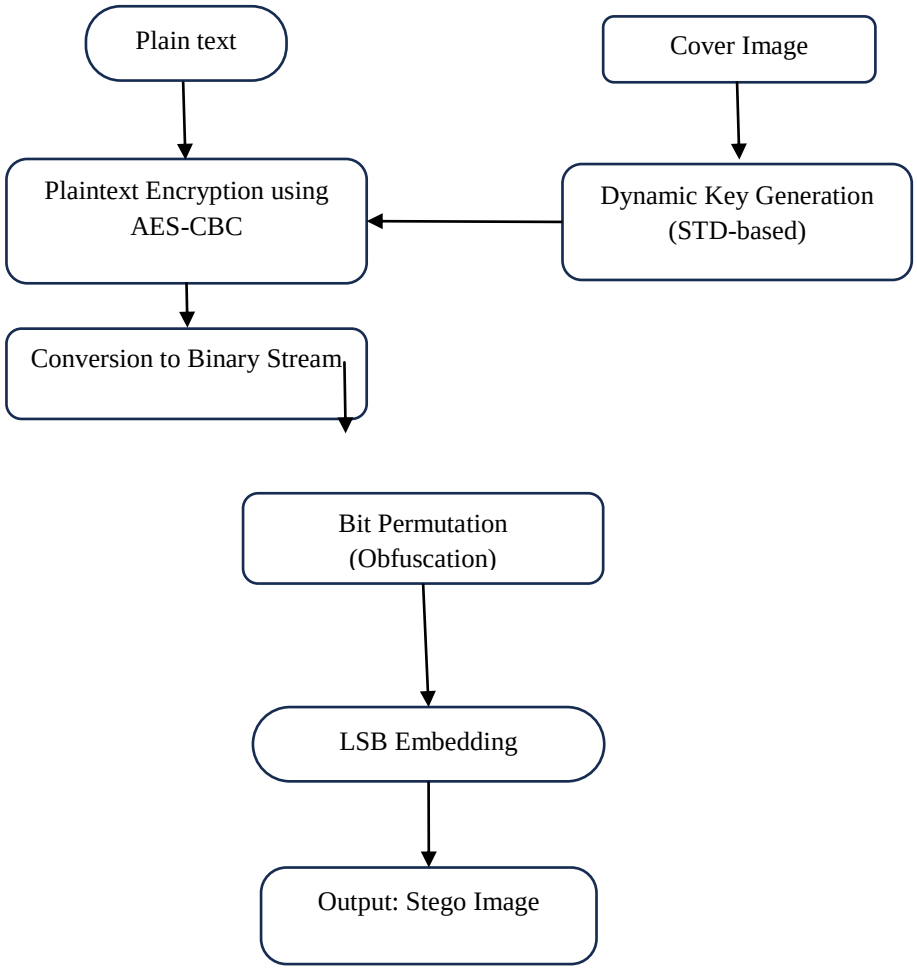

Figure 2: Flowchart of the Encryption and Embedding Process

3.5.2 Decryption and Extraction AES-CBC and LSB Steganography

The decryption process reverses the embedding steps to recover the original hidden message from the stego image, as illustrated in **Figure 3**. It begins by loading the stego image and extracting the least significant bits (LSBs) from each pixel, where the encrypted and obfuscated binary data is stored. The extracted bits are then re-ordered through bit de-permutation to restore their original sequence, after which the bitstream is segmented into fixed-size ciphertext blocks. A decryption key is dynamically renovated from the cover image using the same standard deviation (STD)-based method employed in the encryption phase. And in the end, AES-CBC decryption is performed with the regenerated key and the extracted IV, yielding the recovered plaintext message in its original, human-readable form.

DOI: 10.31695/IJASRE.2025.10.7

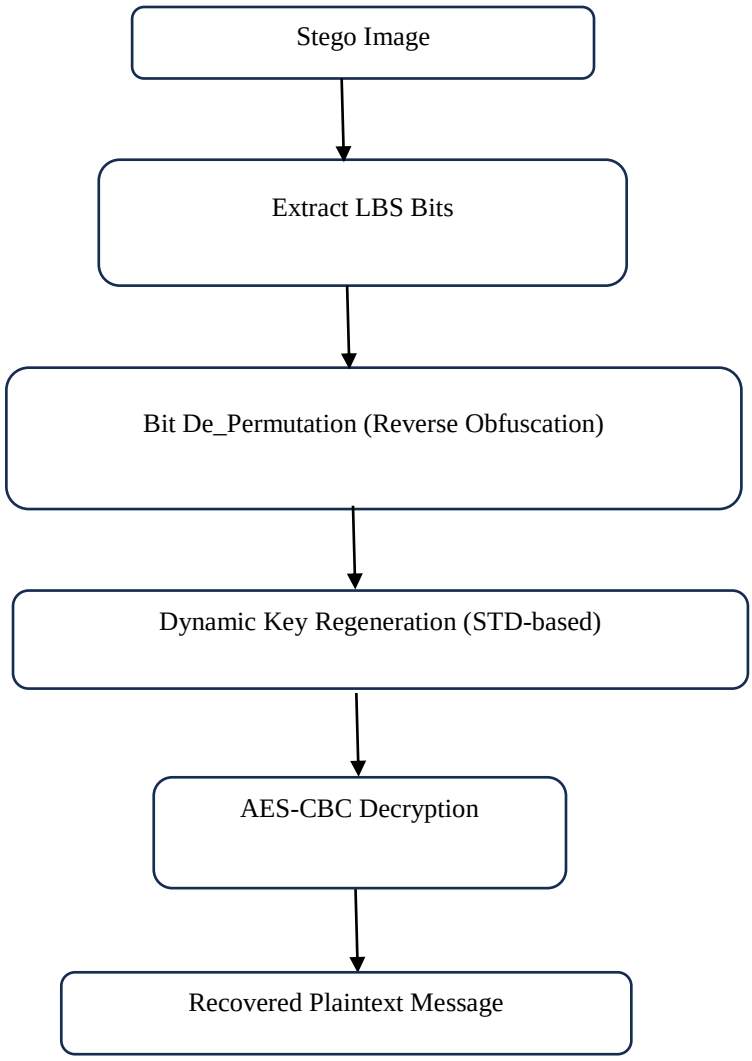

Figure 3: Flowchart of the Decryption and Extraction Process

3.6 Proposed Encryption and Decryption Algorithms

This section introduces the algorithms that have been proposed for encoding messages and how to extract them securely using encryption. AES-CBC encryption combined with a dynamically generated key derived from the standard deviation (STD) of the image.

Illustrated Algorithm 1 show details the encryption and embedding phase. In this process, the standard deviation of the RGB channels of the cover image is computed and concatenated to form a statistical vector, which is then hashed using SHA-256 to produce a unique 256-bit AES key. The secret message is padded, encrypted using AES in CBC mode with a randomly generated initialization vector (IV), and then the IV is prepended to the ciphertext. The resulting binary stream undergoes a pseudo-random permutation based on a user-defined seed, with a 32-bit header added to indicate payload length. Finally, the payload is embedded into the least significant bits (LSBs) of the cover image pixels to produce the stego image.

Algorithm 1: Description of the Encryption and LSB Embedding Process

Input:

- Cover image I (RGB image)
- Secret message M (text string)
- Seed S (integer for permutation)

Output:

- Stego-image SI with hidden encrypted message

DOI: 10.31695/IJASRE.2025.10.7

Begin:

Step 1: Key Generation from Image STD

1.1 Compute standard deviation of R, G, B channels → $[STD_R, STD_G, STD_B]$

1.2 Concatenate values into vector $V = [STD_R, STD_G, STD_B]$

1.3 Generate 256-bit AES key K = SHA256(V)

Step 2: Encrypt Message with AES-CBC

2.1 Generate random 16-byte Initialization Vector (IV)

2.2 Pad the plaintext message M → M_padded

2.3 Encrypt using AES-CBC with key K and IV → Ciphertext C

2.4 Prepend IV to ciphertext → C_total

Step 3: Permutation and Payload Construction

3.1 Convert C_total to binary stream B

3.2 Generate pseudo-random permutation P using seed S

3.3 Apply P to B → B_perm

3.4 Prepend 32-bit binary header H = length(B) to B_perm → Payload B_final

Step 4: LSB Embedding into Image

4.1 Convert cover image I to pixel array

4.2 For each bit in B_final:

Replace LSB of current pixel channel with current bit

4.3 Save modified image as PNG → Stego-image SI

End

Illustrated Algorithm 2 describes the extraction and decryption phase, which reverses the steps of Algorithm 1. The process begins with reading the stego image and extracting the 32-bit length header, followed by retrieving the permuted binary payload from the LSBs. The original bit order is restored using the inverse of the permutation. The IV and ciphertext are separated, and the AES decryption key is regenerated from the stego image using the same STD-based method applied during encryption. AES-CBC decryption is then performed to recover the original plaintext message in human-readable form.

Algorithm 2: demonstrate key extraction and decryption operation

Input:

- Stego-image SI (PNG format)
- Seed S (integer for permutation)

Output:

- Extracted plaintext message M

Begin:

Step 1: Extract Binary Payload

1.1 Read pixel data from SI

1.2 Extract first 32 bits → length header H

1.3 Convert H to integer → L

1.4 Extract next L bits from LSBs → Permuted binary B_perm

Step 2: Undo Permutation

2.1 Generate pseudo-random permutation P using seed S

2.2 Apply inverse of P to B_perm → Original binary stream B

Step 3: Recover Encrypted Bytes

3.1 Convert B into bytes → C_total

3.2 Separate IV (first 16 bytes) and ciphertext → IV, C

DOI: 10.31695/IJASRE.2025.10.7

Step 4: Regenerate AES Key

4.1 Compute STD_R, STD_G, STD_B from SI

4.2 Concatenate values into vector V

4.3 Generate AES key K = SHA256(V)

Step 5: Decrypt Message

5.1 Decrypt ciphertext C using AES-CBC with key K and IV

5.2 Remove padding → Extracted plaintext message M

End

2. Result and Discussion

To evaluate the performance and invisibility of the proposed steganographic method based on AES-CBC encryption and standard deviation (STD) derived key generation we applied many metrics for evaluation the result which is considered between the original and output image such as MSE , MAE , PNSR , R^2 , SSIM , Total Pixel Change , LSB Change Ratio

The original cover image **(Figure 4)** was used to embed a ciphertext message. This message was first encrypted using AES in Cipher Block Chaining (CBC) mode, resulting in a secure ciphertext of 640 bits (80 bytes). The message used in the test case was:
"Hello from Mohammed Fadhil - Mustansiriyah University"

Illustrated Figure 4 shows the original image which used as sample for handling ciphertext after encryption the plaintext message

Figure 4: Original cover image used for message embedding

The encrypted message was successfully embedded into the Least Significant Bits (LSBs) of the cover image. The resulting stego-image **(Figure 5)** appears visually identical to the original cover image **(Figure 4)**, demonstrating a high degree of imperceptibility and visual fidelity. Furthermore, the extraction and decryption processes accurately recovered the original plaintext message without any loss of information, confirming the robustness and reliability of the proposed method

DOI: 10.31695/IJASRE.2025.10.7

Figure 5: Stego-image obtained after embedding the AES-CBC encrypted message into the cover image using (LSB) modification

Using the standard deviation (STD) values extracted from the stego-image, the encryption key was dynamically regenerated and successfully used to decrypt the hidden message. The hash of the derived decryption key was:

a0db6ac359d0e92571ba1f5a36ab3ec58555dbae47753b4adb26c23b4f178035

The decrypted output matched the original plaintext exactly, confirming the integrity and confidentiality of the communication process

4.1 Image Quality Evaluation

To quantitatively assess the impact of data embedding on the image, several evaluation metrics were computed:

Table 1: show result of applying many metrics for processing evaluation

Metric	Value
MSE (Error)	0.000091
MAE	0.000091
PSNR	88.53 dB
R^2 Score	1.000000
SSIM	1.000000
Total pixel changes	336
LSB-only changes	336
LSB change ratio	100.00%

DOI: 10.31695/IJASRE.2025.10.7

4.2 Difference Map Analysis

To further investigate the embedding effect on the image, a pixel-level difference map was generated by overlaying the original cover image and the stego-image. The resulting visualization is shown in **Figure 6 .**
This line is only used for text position clarified
In this figure, red lines indicate the locations of modified pixels. These visual differences are concentrated in the LSBs and are not perceivable by the human eye. The sparse and scattered nature of the modifications supports the conclusion that the proposed method achieves excellent imperceptibility

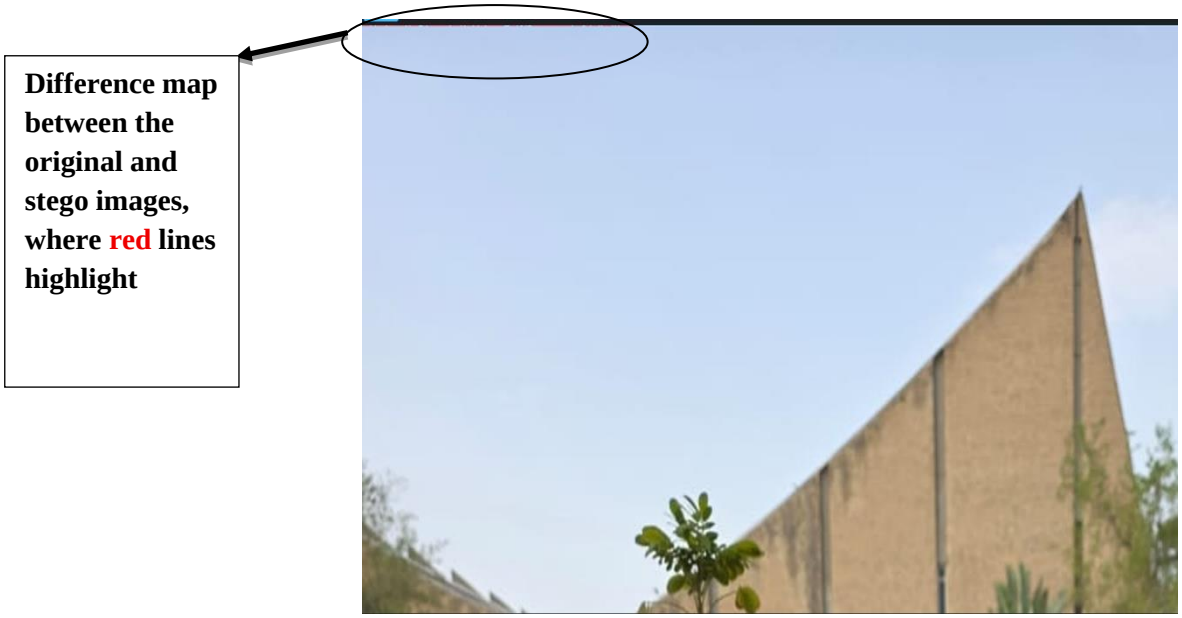

Figure 6 : Difference map between the original and stego images, where red lines highlight

4.2 Discussion

The proposed method integrates AES-CBC encryption with LSB steganography, utilizing a dynamically generated decryption key derived from the image's standard deviation (STD). This design enhances both **security** and **imperceptibility**, enabling successful embedding and recovery of the hidden message—**640 bits (80 bytes)**—without data corruption. The extraction and decryption stages verified message integrity, confirming the robustness of the entire pipeline.

The evaluation results further validate the method's performance. The **PSNR reached 88.53 dB**, significantly above the acceptable threshold of 40 dB, indicating **excellent visual quality**. Additionally, the **MSE and MAE were extremely low (0.000091)**, and both **SSIM and R^2 scored a perfect 1.0**, confirming full structural similarity and statistical consistency between the original and stego images. The modifications were also limited to the LSB layer (336 pixels), ensuring a **100% LSB-only ratio**, which is ideal for stealthy communication.

To contextualize these findings, a comparison was conducted with related recent works. **Table 2** summarizes the performance of different steganographic methods using AES or similar encryption schemes, based on key image quality metrics.

DOI: 10.31695/IJASRE.2025.10.7

Table 2. Comparison between the proposed method and related works

Author / Year	Encryption Method	Hiding Technique	PSNR (dB)	MSE	SSIM	Additional Notes
Proposed Method	AES-CBC + STD-based key	1-bit LSB (static positions)	**88.53**	**0.000091**	**1.000**	Uses a dynamically derived key from image STD; full preservation of image quality
Sulaman et al. [16]	AES-CBC	Radon Transform (sinogram)	64.01	0.0258	0.9999	Good quality, but lower than ours due to Radon domain transformation
Hakim et al. [18]	AES-256	2-LSB	46.40 – 49.65	0.0012 – 0.0023	–	Higher capacity, but significantly lower visual quality
Rasras et al. [19]	Secret key + bit perm.	LSB with dynamic embedding	125.92 – 206.06	0.00006 – 0.1909	1.0	Performance varies with message size; may introduce noticeable changes
Şener & Güney [20]	XOR-based	LSB	49.45 – 54.10	0.0025 – 0.0057	–	Lightweight encryption, but lower visual quality and less security
Abuali et al. [21]	DLSB + WOW	DLSB + Wavelet domain	81.24 (max)	7.51×10^{-9} (min)	0.999	Excellent quality but requires more complex embedding analysis

Compared to these studies, the proposed method achieves **higher or comparable PSNR and SSIM values**, with a **much simpler embedding structure** and without compromising security. While some methods like Rasras et al. report extremely high PSNR values, these may result from specific testing conditions or synthetic data, and do not consistently outperform our results across all metrics.

Furthermore, the use of a **STD-based key** eliminates the need for external key sharing and adds an adaptive layer of security, which is not present in traditional fixed-key or XOR-based methods. This makes the proposed method a suitable candidate for **secure, high-quality steganography applications**, particularly in fields where preserving the original image quality is critical, such as **medical imaging, surveillance, and digital forensics**

5 CONCLUSION

In this study, a secure image steganography method was presented, combining AES-CBC encryption with 1-bit LSB embedding and a dynamically generated key derived from the standard deviation of the cover image. The approach achieved excellent results in terms of imperceptibility and robustness, as evidenced by high **PSNR (88.53 dB), perfect SSIM (1.0), and minimal MSE (0.000091).** These outcomes demonstrate that the method effectively preserves the visual quality of the stego image while ensuring the confidentiality and integrity of the hidden data. The simplicity and

DOI: 10.31695/IJASRE.2025.10.7

security of the technique make it well-suited for sensitive applications such as medical image protection and confidential communications

REFERENCES LIST

[1] S. Ghoul, R. Sulaiman, and Z. Shukur, "A Review on Security Techniques in Image Steganography," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 6, pp. 361–385, 2023, doi: 10.14569/IJACSA.2023.0140640.

[2] N. Subramanian, O. Elharrouss, S. Al-Maadeed, and A. Bouridane, "Image Steganography: A Review of the Recent Advances," *IEEE Access*, vol. 9, pp. 23409–23423, 2021, doi: 10.1109/ACCESS.2021.3053998.

[3] M. A. Majeed, R. Sulaiman, Z. Shukur, and M. K. Hasan, "A review on text steganography techniques," Nov. 01, 2021, *MDPI*. doi: 10.3390/math9212829.

[4] W. Rehman, "A Novel Approach to Image Steganography Using Generative Adversarial Networks," Nov. 2024, doi: https://doi.org/10.48550/arXiv.2412.00094.

[5] Y. Qi, K. Chen, N. Zhao, Z. Yang, and W. Zhang, "Provably Secure Robust Image Steganography via Cross-Modal Error Correction," Dec. 2024, doi: https://doi.org/10.48550/arXiv.2412.12206.

[6] B. A. Salim, M. A. Aljabery, and H. A. Younis, "AES-Based Steganography Using Blockchain: A Novel Approach for Secure Text Hiding in Encrypted Images," *Inform.*, vol. 48, no. 21, pp. 67–78, 2024, doi: 10.31449/inf.v48i21.6689.

[7] S. A. Sherwin, E. I. Simeon, C. A. Subasini, and A. Sheeba, "SECURE IMAGE MESSAGING PLATFORM UTILIZING IMAGE STEGANOGRAPHY AND AES ENCRYPTION," 2024. doi: https://doi.org/10.29007/bw3n.

[8] M. M. Shwaysh, S. Alani, M. A. Saad, and T. A. Abdulhussein, "Image Encryption and Steganography Method Based on AES Algorithm and Secret Sharing Algorithm," *Ing. des Syst. d'Information*, vol. 29, no. 2, pp. 705–714, Apr. 2024, doi: 10.18280/isi.290232.

[9] Prof. Dr.Rafidah Mohamad, "Data hiding by using AES Algorithm," *Wasit J. Comput. Math. Sci.*, vol. 1, no. 4, pp. 72–77, Dec. 2022, doi: 10.31185/wjcm.82.

[10] T. Yu, J. Henderson, A. Tiu, and T. Haines, "Security and Privacy Analysis of Samsung's Crowd-Sourced Bluetooth Location Tracking System," 2023. doi: https://www.usenix.org/conference/usenixsecurity24/presentation/yu-tingfeng.

[11] M. Wildan and W. M. Ashari, "Text Data Security Using LCG and CBC with Steganography Technique on Digital Image," 2024. doi: https://doi.org/10.30871/jaic.v8i2.8457.

[12] S. Gangurde and K. Tiwari, "LSB Steganography Using Pixel Locator Sequence with AES," Dec. 2020, doi: https://doi.org/10.48550/arXiv.2012.02494.

[13] E. G. Jacinto, H. A. Montiel, and F. H. Martínez S, "Enhanced Security: Implementation of Hybrid Image Steganography Technique using Low-Contrast LSB and AES-CBC Cryptography," 2022. doi: 10.14569/IJACSA.2022.01308104.

[14] C. Nithya *et al.*, "Secure gray image sharing framework with adaptive key generation using image digest," *Sci. Rep.*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-92752-9.

[15] H. A. Qasim and R. S. Mohammed, "Lightweight Steganography Technique in The Internet of Things: A survey," *Al-Salam J. Eng. Technol.*, vol. 3, no. 1, pp. 97–108, Nov. 2023, doi: 10.55145/ajest.2024.03.01.008.

[16] M. Sulaman, H. G. umer, kasif Iqbal, M. U. Jatoi, and M. S. Abid, "A Novel Approach for Medical Image Security Using the Radon Transform and AES-CBC Algorithm," Jul. 21, 2023. doi: 10.21203/rs.3.rs-3175303/v1.

[17] E. G. Satish, N. Sreenivasa, E. Naresh, P. Ramesh Naidu, and A. C. Ramachandra, "Multimedia Multilevel Security by Integrating Steganography and Cryptography Techniques," *ITM Web Conf.*, vol. 57, p. 01012, 2023, doi: 10.1051/itmconf/20235701012.

[18] F. N. Hakim and M. Sholikhan, "Enhancing Data Security through Digital Image Steganography: An Implementation of the Two Least Significant Bits (2LSB) Method," *Int. J. Graph. Des.*, vol. 2, no. 2, pp. 222–

DOI: 10.31695/IJASRE.2025.10.7

235, Nov. 2024, doi: 10.51903/ijgd.v2i2.2124.

[19] R. J. Rasras, M. R. A. Sara, J. Nader, and Z. Alqadi, "Increasing the Security of LSB Steganography on the Base of Generated Secret Key," *Trait. du Signal*, vol. 41, no. 6, pp. 3305–3311, Dec. 2024, doi: 10.18280/ts.410646.

[20] D. Şener and S. Güney, "Enhancing Steganography in 256×256 Colored Images with U-Net: A Study on PSNR and SSIM Metrics with Variable-Sized Hidden Images," *Rev. Comput. Eng. Stud.*, vol. 11, no. 2, pp. 13–29, Jun. 2024, doi: 10.18280/rces.110202.

[21] M. S. Abuali, C. B. M. Rashidi, R. A. A. Raof, K. N. F. K. Azir, S. S. Hussein, and A. Q. Abd-Alhasan, "Enhancing Security with Multi-level Steganography: A Dynamic Least Significant Bit and Wavelet-Based Approach," *Math. Model. Eng. Probl.*, vol. 11, no. 6, pp. 1403–1416, Jun. 2024, doi: 10.18280/mmep.110602.

[22] N. H. Sultan, "Hiding Data using LSB in Combination with AES," 2024. doi: https://doi.org/10.52783/jes.2578.

DOI: 10.31695/IJASRE.2025.10.7