IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

SECURESTEALTH: A COMPREHENSIVE FRAMEWORK FOR ADVANCED DATA ENCRYPTION AND STEGANOGRAPHY

[1] Arvind C, [2] Dharaneeswaran P S, [3] Misba Hul Haq S Z, [4] Goma E

[1]Student, [2]Student, [3] Student, [4] Assistant Professor

[1]Department Of Computer Science and Engineering, [1,2,3,4]Adhiyamaan College Of Engineering, Hosur, Tamil Nadu

Abstract: SecureStealth stands out as a powerful solution for hiding sensitive data within images and audio files. It achieves this through a multifaceted approach that blends encryption and steganography, ensuring data confidentiality during digital transmission and storage. To optimize efficiency and make the hidden information less detectable, SecureStealth first employs zlib compression to reduce data size. Subsequently, AES-256 encryption, a robust algorithm, safeguards the compressed data. To guarantee only authorized access, a session key is generated and encrypted using RSA public key cryptography. This two-step encryption strengthens security as the private key for decryption remains confidential. Following encryption, SecureStealth utilizes steganography to embed the encrypted data, encrypted session key, and other crucial information within the chosen image or audio file. It leverages a technique called least significant bit (LSB) modification, strategically altering a predetermined number of least significant bits within the cover media to store the hidden data. To further impede detection by steganalysis tools, SecureStealth incorporates a seeded random number generator. This ensures the hidden bits are dispersed throughout the image or audio file, preventing the formation of easily identifiable patterns. Notably, SecureStealth's steganographic capabilities extend to audio files, offering a versatile solution for covert data embedding across various digital mediums. Developed using Python, SecureStealth boasts cross-platform compatibility, making it accessible on a wide range of operating systems. This combination of robust encryption, steganography techniques, and a user-friendly interface positions SecureStealth as a valuable tool for users seeking to conceal sensitive data within image and audio files.

Index Terms **-** steganography, compression, AES-256, RSA, session key, LSB steganography, random generator, image/audio steganography

I. INTRODUCTION

In the age of digital information overload, safeguarding the privacy and security of sensitive data during communication and storage has become paramount. Traditional security measures often leave a trail, potentially revealing the existence of hidden information. To address this challenge, we introduce SecureStealth, a groundbreaking steganography tool meticulously designed to ensure the complete secrecy of your confidential data. SecureStealth transcends the limitations of conventional data concealment tools by offering a multi-layered approach that integrates robust encryption and cutting-edge steganographic techniques. At its core, SecureStealth utilizes state-of-the-art AES-256 encryption, a well-established algorithm that transforms your confidential data into an unreadable ciphertext. To guarantee that only authorized recipients can access the hidden information, SecureStealth leverages the power of RSA public key cryptography. This two-part encryption process involves generating a session key and encrypting it with the recipient's public key. Only the corresponding private key, held solely by the authorized recipient, can decrypt the session key and ultimately unlock the hidden data. For enhanced efficiency and to minimize the size of the hidden information, SecureStealth incorporates a powerful zlib compression algorithm. This compression not only reduces transmission times but also makes the stego-media (the modified image or audio file containing the hidden data) appear more natural, potentially evading suspicion from steganalysis tools that hunt for irregular patterns. What truly sets SecureStealth apart is its implementation of seeded least significant bit (LSB) steganography. This technique conceals the encrypted data within the digital cover media (the image or audio file being used to hide the information). SecureStealth utilizes a seeded random number generator to strategically select inconspicuous locations within the cover media, such as the least significant bits of pixels in images or audio samples. Scattering the hidden data fragments throughout the cover media makes it incredibly challenging for unauthorized individuals to detect their presence using steganalysis tools. Furthermore, SecureStealth expands its information hiding capabilities beyond images by seamlessly supporting WAV audio files as cover media. This versatility empowers users to choose the most suitable format for their specific needs, ensuring maximum flexibility for covert data embedding across various digital mediums. Designed with user-friendliness in mind, SecureStealth offers a user-friendly interface that simplifies the data hiding process. Additionally, SecureStealth is built using Python, making it compatible with a wide range of operating systems, granting users on various platforms access to its powerful features. By combining cutting-edge encryption, state-of-the-art compression, and innovative steganographic methods, SecureStealth establishes a new benchmark in the realm of secure data concealment, paving the way for a future where sensitive information enjoys unparalleled levels of privacy in the digital age.

II. RELATED WORK

[1] Image Steganography embedded with Advance Encryption Standard (AES) securing with SHA-256: The paper proposes a method combining steganography and cryptography to securely conceal classified information within multimedia files, utilizing AES encryption, SHA-256 hashing, and LSB embedding with minimal visual distortion.

[2] Audio Random Number Generator And Its Application : This work presents a qualified audio random number generator for voice communication, achieving an 80-99% qualification rate against NIST SP 800-22rev1a tests, and demonstrates an encryption application for cellular phone communication confidentiality.

[3] A Comprehensive Image Steganography Tool using LSB Scheme: This paper introduces IMStego, a Java-based tool for secure communication using image steganography, employing LSB and modified LSB algorithms to hide and extract data from BMP and PNG images with user-friendly interface and navigation capabilities.

[4] A Survey on Performance Analysis of DES, AES and RSA Algorithm along with LSB Substitution Technique: The paper presents a combined approach of cryptography and steganography for securing data transmission over unsecured networks, implementing DES, AES, and RSA encryption alongside LSB substitution steganography in C#, evaluating their performance based on encryption/decryption time and buffer size.

[5] image steganography using aes encryption: This paper addresses the need for secure digital picture storage and transmission by employing AES encryption technology to protect images from intrusion attacks.

[6] A Secure Image Steganography based on RSA Algorithm and Hash-LSB Technique: This paper introduces a novel image steganography technique, Hash-LSB with RSA algorithm, enhancing data security by encrypting messages before hiding them in cover images using LSB of RGB pixel values, ensuring confidentiality even if ciphertext is revealed.

[7] The Authenticity of Image using Hash MD5 and Steganography Least Significant Bit: This paper introduces a novel image steganography technique, Hash-LSB with RSA algorithm, enhancing data

security by encrypting messages before hiding them in cover images using LSB of RGB pixel values, ensuring confidentiality even if ciphertext is revealed.

[8] Image Encryption Techniques And Comparative Analysis : This paper delves into the importance of encryption for securing multimedia data during transmission over the web, focusing on understanding and implementing cryptographic algorithms like DES, AES, and RSA for data protection between parties.

[9] Image and audio steganography based on indirect LSB: This paper introduces a modified information hiding technique using indirect least significant bit comparison for enhanced data security in steganography, applied to image and audio cover media, with encrypted text messages, showing promising performance in experimental results.

[10] Information Hiding using Improved LSB Steganography and Feature Detection Technique: This paper introduces an enhanced LSB-based steganography technique for images, employing encryption and embedding algorithm to hide encrypted messages in nonadjacent and random pixel locations across edges and smooth areas, achieving improved information hiding capacity and PSNR value compared to existing methods.

[11] Images Encryption Method using Steganographic LSB Method, AES and RSA algorithm: This paper introduces a novel image encryption technique combining symmetric and asymmetric encryption with steganography, eliminating the need for key sharing during encryption, resulting in faster computation while maintaining close-to-optimal accuracy.

[12] Enhanced Multistage RSA Encryption Model: This research proposes enhancing RSA algorithm security by incorporating various cryptography techniques, notably OAEP, Diffie-Hellman, and HiSea, resulting in increased complexity and search space, thereby improving protection against attacks.

[13] Double layer security using crypto-stego techniques: a comprehensive review: This paper reviews recent advancements in combining cryptography and steganography to provide double-layer security for covert communication, analyzing image steganography techniques and crypto-stego methods while detailing evaluation parameters for both domains.

[14] Image Steganography Based on Complemented Message and Inverted bit LSB Substitution : This paper presents a three-tiered security approach in steganography, utilizing complemented messages, random pixel selection, and inverted bit LSB method, yielding improved results over simple LSB techniques in terms of MSE and PSNR.

[15] Audio Steganography Using LSB Edge Detection Algorithm: Digital Steganography is used to protect digital content or data such as text, images, audio and videos that have been tampered maliciously. In this paper we maintain the quality of audio and image and to ensure the ownership, we propose a new LSB (least significant bit) using edge detection in digital steganography. We apply a 2-level steganography on images and audio which make the data which may be text or image in more secure form. By using lsb techniques may effects less the image pixel quality and audio sound quality, in this wecan randomly selected edges and embed the text or image by considering image quality, audio quality and audio imperceptibility.

III. OBJECTIVE

The primary goal of this project is to develop SecureStealth, a secure steganography tool designed to safeguard sensitive information during digital transmission and storage. SecureStealth accomplishes this by leveraging a multi-layered security approach that blends robust encryption techniques with steganography.

At its core, SecureStealth utilizes a combination of AES-256 encryption, a well-established algorithm that transforms data into an unreadable ciphertext, and RSA public key cryptography. This two-step encryption process guarantees that only authorized recipients can access the hidden information. A session key is generated and encrypted using the recipient's public key. Only the corresponding private key, held solely by the authorized recipient, can decrypt the session key and ultimately unlock the hidden data.

Furthermore, SecureStealth incorporates zlib compression to optimize data size before encryption. This compression not only enhances transmission efficiency but also minimizes the size of the hidden data within the cover media (the image or audio file used to conceal the information). This can potentially make the stego-media (the modified file containing the hidden data) appear more natural, reducing the likelihood of detection by steganalysis tools that hunt for irregular patterns.

The project expands the boundaries of traditional image steganography by extending its capabilities to audio files. SecureStealth leverages a technique called least significant bit (LSB) modification, where a predetermined number of least significant bits within the digital cover media are manipulated to store the hidden data. To make detection by steganalysis tools even more challenging, SecureStealth incorporates a seeded random number generator. This ensures the hidden bits are dispersed throughout the image or audio file, preventing the formation of easily identifiable patterns that could raise suspicion.

IV. RESEARCH METHODOLOGY

4.1.AES ALGORITHM

AES-256 (Advanced Encryption Standard with a 256-bit key) is a widely adopted symmetric encryption algorithm. It provides strong security and is commonly used for data protection. To generate an AES-256 session key, we can use the PBKDF2 (Password-Based Key Derivation Function 2) algorithm with HMAC-SHA-256 as the underlying hash function.

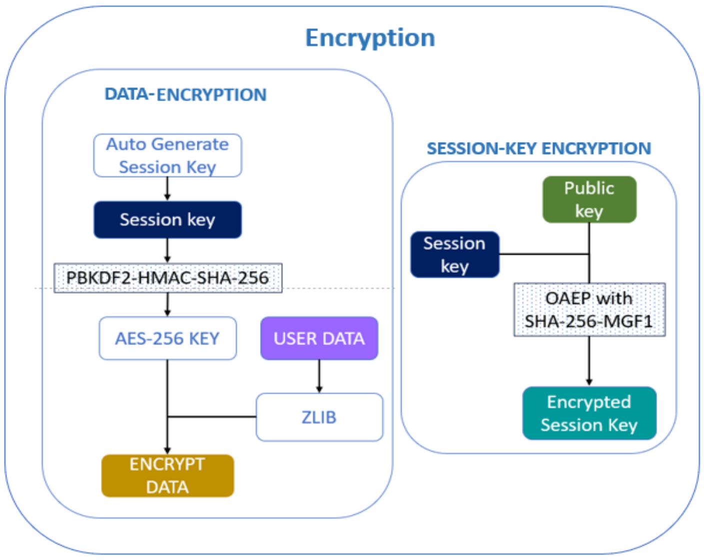

Figure 1: Algorithm 1 - AES Algorithm Encryption

The above diagram showing the different types of encryption: data encryption and session key encryption. Let's break down the process:

Data Encryption

User Data: This refers to the original data that you want to encrypt.

Password (Input): A secret word or phrase used to generate a secret key. This secret key is critical for data encryption.

PBKDF2-HMAC-SHA-256: This stands for Password-Based Key Derivation Function 2 with HMAC-SHA-256. It's a way to derive a strong encryption key from a user's password.

AES-256 Key: The derived key from the password is used to encrypt the user data using the Advanced Encryption Standard (AES) with a key size of 256 bits. AES is a common and secure symmetric encryption algorithm.

Encrypted Data: This is the scrambled data that cannot be read without the decryption key.

ZLIB: This is a compression algorithm that may be applied to the encrypted data to reduce its size.

Session Key Encryption

Encrypted Session Key: This refers to the secret key used for data encryption, encrypted with a public key.

Public Key: This is part of a public-key cryptography system, such as RSA. Anyone can have a copy of the public key, but only the corresponding private key can be used to decrypt the data.

OAEP with SHA-256-MGF1: This stands for Optimal Asymmetric Encryption Padding with SHA-256 Message Digest Function 1. It's a padding scheme that is used to make RSA encryption more secure.

4.2. RSA ALGORITHM

The RSA algorithm (Rivest–Shamir–Adleman) is a fundamental asymmetric cryptographic algorithm widely used for secure data transmission. Let’s explore the key aspects of RSA:

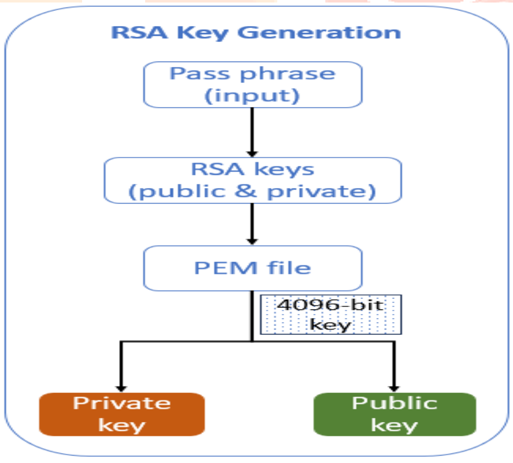

Figure 2: RSA Algorithm - Key Encryption

The process of generating RSA keys involves several steps, as shown in the image:

Passphrase (input): The first step is to choose a passphrase. This is a secret word or phrase that will be used to generate the keys. The passphrase should be strong and difficult to guess.

RSA keys (public & private): The next step is to generate the RSA keys. These keys come in a pair: a public key and a private key. The public key can be shared with anyone, while the private key must be kept secret.

PEM file (optional): The keys can then be saved in a PEM (Privacy Enhanced Mail) file. This is a standard format for storing cryptographic keys.

Key size: The key size is typically specified in bits. Common key sizes include 2048 bits and 4096 bits. A larger key size provides more security, but it also takes longer to encrypt and decrypt data.

The public key can be used to encrypt data, but only the private key can decrypt the data. This makes RSA keys ideal for secure communication, such as sending encrypted emails or protecting data on a website.

4.3.HIDING METHODS IN IMAGE AND AUDIO STEGANOGRAPHY

Steganography is the science to hide secret information in other data, long before the invention of the computer. One can find in literature several different techniques of steganography . LSB is the best known method. LSB is to change the least significant bit of the cover media and also the advancement of this method has been taken place with the seeded random number for the LSB. The ZLIB is used before encryption process to compress the whole data to reduce the strength of the data. Detail contents are given below for each methods,

Seeded Random Number In the clandestine world of steganography, the art of hiding messages within seemingly innocuous cover media, a crucial element often goes unnoticed: the seeded random number generator (RNG). While the focus often rests on robust embedding techniques and choosing suitable cover media, the selection and utilization of a good RNG play a significant role in the security and effectiveness of steganography. This exploration delves into the core principles of steganography, the importance of randomness, and the specific role of seeded RNGs in this fascinating domain.

LSB Steganography In the intriguing world of steganography, the art of concealing messages within seemingly innocuous cover media, the Least Significant Bit (LSB) technique reigns supreme. Its simplicity, efficiency, and widespread applicability across various digital media formats make it a cornerstone for many steganographic endeavors. This exploration delves into the core principles of LSB steganography, its advantages and limitations, and its role in the fascinating realm of covert communication.

ZLIB In the ever-growing digital world, data compression plays a vital role in optimizing storage space, reducing transmission times, and enhancing overall efficiency. ZLIB, a free, general-purpose compression library, stands tall as one of the most widely used and versatile tools for data compression. This exploration delves into the core principles of ZLIB, its compression algorithm, advantages, and its extensive applications across various domains.

4.4.ARCHITECTURE

The provided architecture diagram visually depicts the inner workings of SecureStealth. It outlines the sequential steps involved in the process, including the utilization of the chosen steganographic algorithm. The directional arrows within the diagram illustrate the data flow throughout the various stages of SecureStealth's operation.

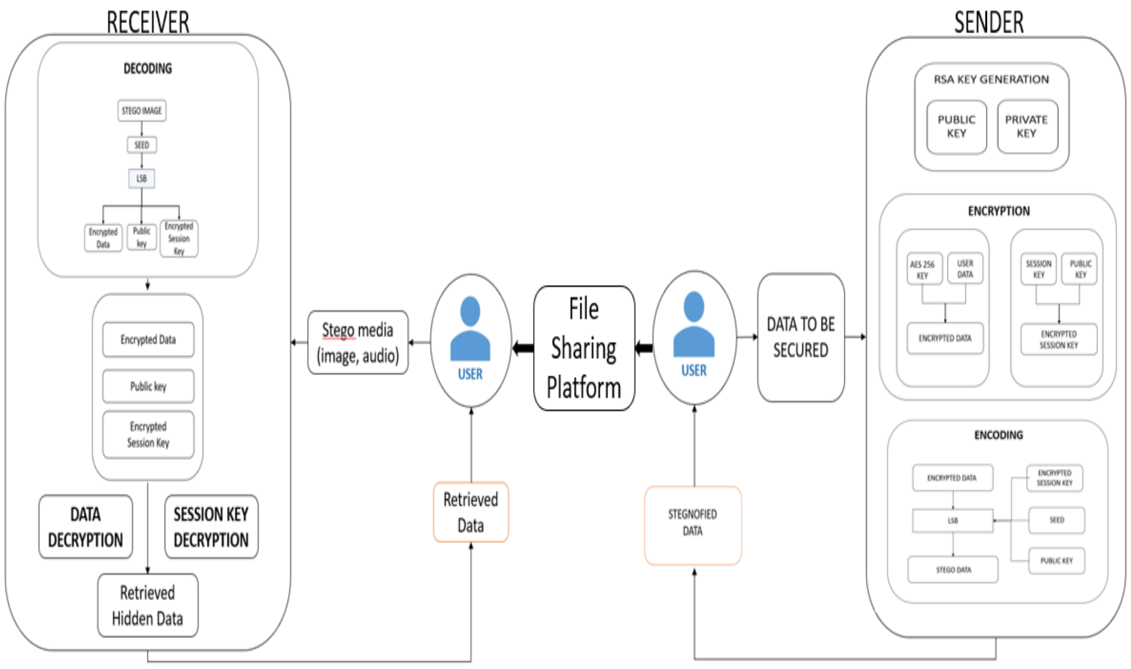

Figure 3: Architecture

Here's a breakdown of the process depicted in the image:

Sender

RSA Key Generation: The sender starts by generating a public and private key pair using the RSA algorithm. The public key can be shared with anyone, while the private key should be kept secret.

Choosing Steganography Type: The sender chooses between hiding data in an image or audio file (image steganography or audio steganography).

Data Selection: The sender selects the data they want to hide inside the chosen image or audio file.

Cover Media Selection: The sender selects the image or audio file that will be used to hide the data (cover media).

Public Key Selection: The sender selects the recipient's public key if it's someone else. This public key will be used to encrypt the session key. If the sender is the receiver as well (self-hiding data), this step might be bypassed.

Data Compression (Optional): The data to be hidden may be compressed to reduce its size.

Data Encryption: The sender encrypts the data using a secret key derived from a passphrase.

Session Key Encryption: The secret key used to encrypt the data (session key) is encrypted using the recipient's public key (if applicable).

Encoding:

Image Steganography: The encrypted data, encrypted session key (if applicable), a seed, and other information are encoded and hidden inside the cover image using the least significant bit (LSB) technique.

Audio Steganography: The encrypted data, encrypted session key (if applicable), and other information are encoded and hidden inside the cover audio file using the LSB technique.

Stego Media Creation: The result of the encoding process is a stego-image (modified image containing hidden data) or stego-audio (modified audio file containing hidden data).

Sending the Stego Media: The sender transmits the stego-image or stego-audio file through a public medium (email, internet upload, etc.) to the receiver.

Receiver

Secure Stealth Tool: The receiver uses the same Secure Stealth Tool to process the received stego-image or stego-audio file.

Stego Media Selection: The receiver uploads the stego-image or stego-audio file they want to decode.

Private Key and Passphrase: The receiver enters their private key and passphrase (used to generate the secret key for decryption). In the case of self-hiding data (sender is also the receiver), the private key corresponding to the public key used for encryption (step 5 on the sender's side) would be used here.

LSB Count (Audio Only): If audio steganography was used, the receiver might also need to enter the LSB count (number of least significant bits used to hide data).

Decoding:

Image Steganography: The tool extracts the hidden data, encrypted session key (if applicable), and other information from the stego-image using the LSB technique.

Audio Steganography: The tool extracts the hidden data, encrypted session key (if applicable), and other information from the stego-audio file using the LSB technique.

Session Key Decryption (if applicable): The encrypted session key is decrypted using the receiver's private key to obtain the secret key used to encrypt the data.

Data Decryption: The hidden data is decrypted using the secret key obtained in step 6 (or the passphrase if session key encryption was not used).

4.5. **DATA EXAMPLE:**

An evaluation of the three significant properties of any image steganography technique, undetectability, level of security, and capacity, was obtained to characterize the strengths and weaknesses of the proposed method. Table 1 presents the cover images used to test the proposed technique.

Table 1. The cover images that were used to evaluate the proposed method.

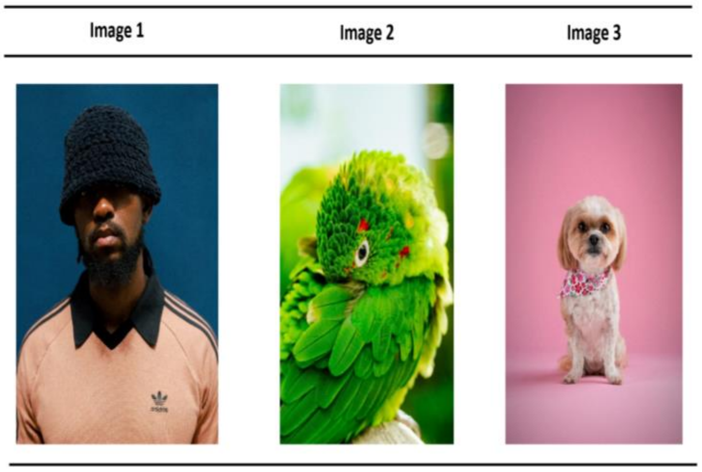

Figure 4: Cover Images

V. RESULT

5.1. IMAGE

5.1.1. Implementation Details

The embedding and extraction algorithms for the proposed steganographic technique were developed using Python 3.6 within the VScode IDE. A web interface was utilized to facilitate user interaction. The experiments focused on bitmapped images with 256 colors, a fixed size of 500x500 pixels, and a 24-bit depth. To assess the impact of embedding on image quality, various secret messages were concealed within the images. Three cover images were divided into different blocks for analysis. Two additional images served as encrypted messages and secret patterns during the tests.

5.1.2. Evaluation Results

The embedding capacity of the proposed technique was evaluated using messages of varying sizes and hidden patterns. For each test, the imperceptibility (visual quality) and fidelity (accuracy) of the stego-image (image containing hidden data) were assessed using Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE). The PSNR and MSE statistics across different message sizes were compared with established steganography methods. Tests were conducted on sample images with consistent size and message-to-segment ratios. The model demonstrated stable performance with increasing message size due to the strategic selection of high-frequency areas within each image block for hiding the secret data, effectively disrupting the original image spectrum without significant visual degradation.

5.1.3. Quantitative Results

Table 2 summarizes the performance of the proposed method when applied to three test images. Each experiment utilized a secret message with a capacity of 480 bits. As shown in the table, the MSE and PSNR values varied across the three images. Image 1 exhibited an MSE of 0.00061387 and a PSNR of 80.2400. Image 2 displayed a lower MSE of 0.00019995 and a higher PSNR of 85.2426. Similarly, Image 3 yielded an MSE of 0.0002002 and a PSNR of 85.1473.

Table 2. Results on Images (Capacity, PSNR, and mean Error score)—First Message.

Secret Message	The Message		
	Capacity	MSE	PSNR
Image 1	480	0.00061387	80.2400
Image 2	480	0.00019995	85.2426
Image 3	480	0.0002002	85.1473

Table 3: Results for Second Secret Message

Table 3 summarizes the performance of the proposed steganographic technique when embedding a second secret message with a fixed capacity of 2400 bits into three different images. The results are measured in terms of capacity (bits), mean squared error (MSE), and peak signal-to-noise ratio (PSNR).

As shown in the table, the MSE and PSNR values varied across the three images (Image 1: MSE = 0.00060364, PSNR = 80.3170; Image 2: MSE = 0.00018866, PSNR = 85.3470; Image 3: MSE = 0.00018717, PSNR = 85.4033). A lower MSE indicates a smaller difference between the original and stego-images (images with hidden data), while a higher PSNR suggests a higher quality stego-image.

Table 3. Results on Images (Capacity, PSNR, and mean Error score)—Second message.

Secret Message	The Message		
	Capacity	MSE	PSNR
Image 1	2400	0.00060364	80.3170
Image 2	2400	0.00018866	85.3470
Image 3	2400	0.00018717	85.4033

Table 4: Results for Third Secret Message

Table 4 details the performance of the proposed steganography method when embedding a third secret message with a capacity of 4800 bits into three different images. The results are evaluated using mean squared error (MSE) and peak signal-to-noise ratio (PSNR).

Interestingly, Image 2 exhibited the lowest MSE (indicating minimal distortion) and the highest PSNR (suggesting the best quality) after the steganographic process, compared to the other two images (Image 3: highest MSE, lowest PSNR). These findings demonstrate the impact of image properties on the steganographic method's performance. This data can be valuable in guiding the development of future steganographic algorithms for enhanced robustness across diverse image types.

Table 4. Results on Images (Capacity, PSNR, and mean Error score) – Third Message.

Secret Message	The Message		
	Capacity	MSE	PSNR
Image 1	4800	0.0004235	81.6217
Image 2	4800	0.0001175	85.7435

Image 3	4800	0.00017641	85.6351

PSNR and MSE Analysis (Table 5):

Table 5 summarizes the PSNR and MSE values obtained after embedding three secret messages (capacities: 480 bits, 2400 bits, and 4800 bits) into three different images (Images 1, 2, and 3). As expected, the PSNR values (indicating image quality) generally increased across all images with each subsequent message (Message 1: 80.2400 (Image 1), 85.2426 (Image 2), 85.1473 (Image 3); Message 2: 80.3170, 85.3470, 85.4033; Message 3: 81.6217, 85.7435, 85.6351). Conversely, the MSE values (indicating image distortion) decreased with increasing message capacities, suggesting a positive correlation between message size and image quality in this specific scenario. Figure 4 builds upon these observations by comparing the proposed method's PSNR results with existing steganographic techniques.

Table 5. The proposed Method Result Summary.

Image	PSNR and MSE		
	First Image	**Second Image**	**Third Image**
Image 1	80.2400	80.3170	81.6217
Image 2	85.2426	85.3470	85.7435
Image 3	85.1473	85.4033	85.6351

5.1.4. Comparison with Existing Works

Comparison with Existing Methods (Table 6):

Table 6 presents a comparative analysis of the proposed steganographic method's performance against established techniques in the field. The evaluation criteria focus on Peak Signal-to-Noise Ratio (PSNR), a metric for assessing image quality after data embedding. The proposed method is compared to the classical Least Significant Bit (LSB) method, the Over-Multiplexed Vega (OMVG) method, the Kaur method, and the Liao method.

The results demonstrate that the proposed method achieves superior PSNR values, particularly in Images 2 and 3, where it reaches 85.7435and 85.6351, respectively. Conversely, the traditional LSB method exhibits the lowest PSNR values across all images (ranging from 50 to 58.62), indicating a significant reduction in image quality after steganographic processing.

Table 6. Comparison between the proposed model and the related work for PSNR .

Image	**Classical LBS**	**OMVG Method**	**Kaur**	**Liao**	**The Proposed Method**
	PSNR	PSNR	PSNR	PSNR	PSNR
Image 1	58.62	74.50	75.8	74.6	81.6217
Image 2	57.53	74.41	78.13	79.68	85.7435
Image 3	50.00	74.45	79.85	80.24	85.6351

5.1.5. Run Time Results

Runtime Performance Analysis (Table 7):

Table 7 compares the average runtime performance of the proposed steganographic method (utilizing Blowfish and AES encryption) against existing techniques during the encryption and decryption processes. The proposed method exhibits impressive efficiency, achieving an average runtime of 0.83 seconds for encryption and 1.5 seconds for decryption.

Compared to traditional methods like LSB and OMVG, the proposed method demonstrates significant runtime improvements. Notably, LSB and OMVG have considerably longer runtimes, exceeding 1.5 seconds for both encryption and decryption. The proposed method also boasts a slight edge over the Kaur and Liao methods, which have runtimes ranging from 1.254s to 1.642s for encryption and 1.5s to 1.642s for decryption.

Figure 5 visually depicts these runtime comparisons for better comprehension.

Table 7. Run time comparison proposed model and related work (seconds).

Process type	Classical LBS	OMVG Method	Kaur	Liao	The Proposed Method
Encryption	1.874	1.012	1.254	0.98	0.86
Decryption	2.018	1.492	1.642	1.54	1.61

5.1. AUDIO

5.2.1. Cryptography Analysis

The retrieval algorithm (3.2.2) successfully extracts the ciphertext from the cover media. Decryption using algorithm 3.2.1 then recovers the original plain text message. Table 1 demonstrates the complete and successful retrieval of the text message. To assess the similarity between the sender's and receiver's messages, the Structural Similarity Index Measure (SSIM) was employed. The consistently high SSIM values across various message sizes (all equal to 1) indicate the system's reliability and accuracy in message retrieval.

Table 1. The SSIM values of the secret messeges from sender to receiver

Message size (number of the characters)	SSIM
480	1
2400	1
4800	1

5.2.2. Steganography Results

System Design and Evaluation: This work proposes a novel steganographic system for concealing text messages within audio files (*.wav format). The secret message is first encrypted using a newly developed bit-cycling method to generate ciphertext. This ciphertext is then embedded into the cover audio file.

The system's effectiveness was evaluated across various message lengths and cover file sizes. To ensure reliable and efficient embedding, the chosen cover file size (measured in samples) must be at least seven times the character count of the text message. The system simulations were conducted using MATLAB R2018a on a computer with an Intel® Core™ i7-3520M CPU @ 2.90 GHz and 8.00 GB RAM.

Performance Analysis: To assess the system's characteristics, various metrics were employed: Peak Signal-to-Noise Ratio (PSNR), Mean Squared Error (MSE), and Structural Similarity Index Measure (SSIM). Notably, the PSNR values before and after encryption and message transmission consistently reached infinity, indicating perfect message retrieval with no data loss.

Furthermore, PSNR for the audio files remained high (ranging from 60 dB to 65 dB) after embedding messages of varying lengths within the same file. Additionally, the MSE values were consistently below 9e-7, exceeding the minimum acceptable audio quality threshold of 20 dB established by the International Federation of the Phonographic Industry (IFPI). Finally, the SSIM scores remained very close to 1 for all cover files, indicating a high degree of structural similarity between the original and stego-audio (audio with hidden data).

These excellent results demonstrate the system's ability to embed messages imperceptibly while maintaining high audio quality, as summarized in Table 2.

Table 2. The PSNR, MSE and SSIM calculated for different size of message embedded in an audio cover file of length (3 seconds) with (48000) Fs before and after hiding.

Message size (number of the characters)	PSNR	MSE	SSIM
480	65.0616	3.1249e-07	0.9987
2400	60.7343	8.6411e-07	0.9796
4800	60.73	8.6420e-07	0.9885

Impact of Bit Interpolation: The bit-interpolation process modifies the sample value by a minimal amount (0.00001) only when the least significant bit (LSB) differs from the hidden message bit. While this small change may introduce slight noise into the cover file, the impact is negligible. The system prioritizes minimizing distortion by embedding shorter messages within larger cover files.

Effectiveness and Noise Mitigation: Extensive statistical analysis and practical experiments confirm the system's effectiveness in securing data transmission despite the minimal noise introduced. This noise can be further reduced by increasing the multiplication factor in step 3.2.2 (y'(i) = y(i) * a larger number). However, it's crucial to find a balance between noise reduction and maintaining sufficient data embedding capacity.

5.2.3. Computational Costs

Efficiency Analysis (Table 3, Figure 3): Experiments demonstrated a proportional increase in processing time (both encryption and steganography) with message length, as shown in Table 3 and Figure 3. Notably, the average processing time for the entire system (encryption and hiding) remained consistently below 0.3 seconds. This is significantly faster than previously reported results, which averaged around 0.42 hours.

It's important to acknowledge that our system was tested on a less powerful processor (details provided) compared to the one used in the previous study (Nvidia GPU Titan X, Intel Core i7-6900K CPU, 64 GB RAM). Despite this hardware disparity, our method achieves superior efficiency in terms of processing speed. This efficiency makes it well-suited for implementation on resource-constrained devices like tablets, lightweight laptops, and smartphones.

Table 3. The time consumption (seconds) of the proposed system for different message lengths

No. of message	Elapsed Time			
characters	Encryption	Decryption	Hiding	Unhide
480	0.0091	0.0028	0.022	0.039
2400	0.0088	0.0066	0.0942	0.0722
4800	0.0173	0.0115	0.161	0.109

The System's Security: The proposed system utilizes three crucial keys

Encryption Key: This key encrypts the text message before embedding, ensuring message confidentiality. Multiplication Number: This value controls the magnitude of change applied to the cover file during embedding, balancing data security and noise introduced. Seed Number: This number serves as the foundation for generating a pseudo-random sequence used to strategically spread message bits within the cover file. Without knowledge of all three keys, recovering the hidden message becomes computationally infeasible, significantly enhancing the system's security.

VI. CONCLUSION

SecureStealth offers a powerful solution for hiding sensitive information within digital media. It leverages a combination of robust encryption techniques (AES-256 and RSA) to ensure the confidentiality of hidden data during transmission and storage. Additionally, SecureStealth employs advanced steganography techniques (seeded LSB) to seamlessly embed the encrypted data within cover files. To further enhance security and potentially evade steganalysis detection, SecureStealth incorporates data compression (zlib) to reduce file size without compromising hidden information integrity. The tool's versatility extends to supporting both image and audio cover media (WAV files), making it suitable for various data hiding scenarios. SecureStealth prioritizes user experience with a user-friendly interface and cross-platform compatibility (Python), enabling a broader range of users to leverage its functionalities. It's important to remember that steganography, by itself, cannot guarantee absolute invisibility, especially considering the continuous evolution of steganalysis techniques. To bolster security, SecureStealth users are encouraged to employ strong passwords or pre-shared keys. In conclusion, SecureStealth represents a significant advancement in the field of secure data concealment. Its multifaceted approach and adaptability make it a valuable asset for users seeking to safeguard sensitive information in the digital landscape.

REFERENCE

[1] vikas singhal, yash kumar shukla, navin prakash, image steganography embedded with advance encryption standard (aes) securing with sha-256 issn: 2278-3075 (online), volume-9 issue-8, june 2020

[2] i-te chen , jer-min tsai , jengnan tzeng,audio random number generator and its application , · july 2011 doi: 10.1109/icmlc.2011.6017002 · source: dblp.

[3] sahar a. el_rahman, a comprehensive image steganography tool using lsb scheme, published online may 2015 in mecs, doi: 10.5815/ijigsp.2015.06.02.

[4] b. padmavathi , s. ranjitha kumari, a survey on performance analysis of des, aes and rsa algorithm along with lsb substitution technique, issn: 2319-7064 volume 2 issue 4, april 2013.

[5] jai dhiyanesh j, anish khan, anish kumar, image steganography using aes encryption, volume 8, issue 2 february 2023 ,issn: 2456-4184 ,ijnrd.org

[6] ms.shridevishetti, mrs.anuja s, a secure image steganography based on rsa algorithm and hash-lsb technique, issn: 2278-0181, volume 3, issue 19 , icesmart-2015.

[7] nurul khairina , muhammad khoiruddin harahap , juanda hakim lubis, the authenticity of image using hash md5 and steganography least significant bit, vol. 2, no. 1, (2018), issn : 2580-7250.

[8] shaolin kataria, elio jordan lopes,bevis mathew niravil, shashank keshav4 image encryption techniques and comparative analysis, volume: 08 issue: 10, oct 2021 www.irjet.net p-issn: 2395-0072.

[9] zainab n. sultani , ban n. dhannoon ,image and audio steganography based on indirect lsb, kuwait j.sci., vol.48, no.(4),october.2021.

[10] mamta juneja, parvinder singh sandhu ,information hiding using improved lsb steganography and feature detection technique, volume-2, issue-4, april 2013, issn: 2249 – 8958.

[11] abdelkader moumen ,hocine sissaoui,images, encryption method using steganographic lsb method, aes and rsa algorithm, doi 10.1515/nleng-2016-0010, (2016)

[12] mays m. hoobi , sumaya s. sulaiman , inas ali abdulmunem, enhanced multistage rsa encryption model, doi:10.1088/1757-899x/928/3/032068, iscau-2020.

[13] aiman jan, shabir a. parah , muzamil hussan ,bilal a. malik, double layer security using crypto stego techniques: a comprehensive review, https://doi.org/10.1007/s12553-021-00602-1, (2022).

[14] rupali bhardwaja , vaishali sharmab, image steganography based on complemented message and inverted bit lsb substitution, doi: 10.1016/j.procs.2016.07.245 , icacc 2016.

[15] navneet kaur, sunny behal,audio steganography using lsb edge detection algorithm, (icccs–2014).

[16] m. preetha, m. nithya,a study and performance analysis of rsa algorithm, vol. 2, issue. 6, june 2013,issn 2320–088x.

[17] anurag chandrabali gautam ,secure end to end transmission using audio steganography and aes encryption, x18145060,(12th december 2019).

[18] mwaffaq abu-alhaija,crypto-steganographic lsb-based system for aes-encrypted data, vol. 10, no. 10, (2019).

[19] dr. helena handschuh, dr. henri gilbert,evaluation report security level of cryptography – sha-256, 31 january 2002, issy-les-moulineaux.

[20] dr. p. r. deshmukh , bhagyashri rahangdale ,hash based least significant bit technique for video steganography, vol. 4, issue 1(version 3), january 2014, issn : 2248-9622.

[21] mekha jose ,hiding image in image using lsb insertion method with improved security and quality, impact factor (2012): 3.358, issn (online): 2319-7064.

[22] fausto meneses, walter fuertes, josé sancho, santiago salvador, daniela flores, hernán aules, fidel castro ,jenny torres alba miranda, danilo nuela,rsa encryption algorithm optimization to improve performance and security level of network messages, vol.16 no.8, august 2016, ijcsns.

[23] henri gilbert,helena handschuh,security analysis of sha-256 and sisters, sac 2003, lncs 3006,(2004).

[24] babloo saha and shuchi sharma,steganographic techniques of data hiding using digital images, vol. 62, no. 1, january 2012, doi: 10.14429/dsj.62.1436,(2012).

[25] xinyi zhou, wei gong, wenlong fu, lianjing jin ,an improved method for lsb based color image steganography combined with cryptography, icis 2016, june 26-29, (2016).