International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 25 July 2023 **Revised:** 12 September 2023 **Accepted:** 30 November 2023*

Unified Multimedia Steganography: AES-Protected Data Concealment

[1]Dr. Ravindra Sangle
[1]Associate Professor, Department of Computer Engineering , Vidyalankar Institute of Technology, Mumbai, India
ravindra.sangle@vit.edu.in

[2]Dr. Mandar Sohani
Professor, Depatment of Computer Engineering, Vidyalankar Institute of Technology, Mumbai, India.
mandar.sohani@vit.edu.in

[3]Dr. Girish Gidaye
Professor, Department of Electronics & Computer Science, Vidyalankar Institute of Technology, Mumbai, India.
girish.gidaye@vit.edu.in

[4]Tejas Barge
Department of Computer Engineering , Vidyalankar Institute of Technology, Mumbai, India
tejas.barge@vit.edu.in

[5]Chandan Patil
Department of Computer Engineering, Vidyalankar Institute of Technology, Mumbai, India
chandan.patil@vit.edu.in

[6]Snehal Lohar
Department of Computer ngineering , Vidyalankar Institute of Technology, Mumbai, India
snehal.lohar@vit.edu.in

[7]Dipesh Rawal
Department of Computer Engineering , Vidyalankar Institute of Technology, Mumbai, India
dipesh.rawal@vit.edu.in

***Abstract*—**Steganography plays a vital role in secure communication and data protection in the digital realm. This paper explores the integration of Advanced Encryption Standard (AES) encryption with the Least Significant Bit (LSB) steganography technique to bolster the security of hidden information in multimedia files. By encrypting data before embedding it using LSB, the content remains unintelligible without the decryption key, regardless of whether the existence of hidden data is detected or not. The proposed system extends the application of LSB steganography with AES encryption to text, images, audio, and video files. Performance evaluation through experiments measures payload capacity and Peak Signal-to-Noise Ratio (PSNR) for different media types and levels of data embedding. The results are graphically presented to illustrate the system's effectiveness in concealing information securely within multimedia files.

***Keywords*—** *Steganography, Modified LSB, AES, Unicode, Audio, Video, Image, Text*

I. INTRODUCTION

In modern digital contexts, steganography plays an important role in secure communication, digital watermarking, copyright protection, and clandestine data transmission. It is very important to secure any important information that has to be transferred from a sender to a receiver. Intruders can disclose the information to others, change it to misrepresent an individual or organization, or use it for an attack. This problem can be solved through the use of steganography. Steganography is the technique of hiding information in the digital media. In contrast to cryptography, it is not to encrypt the information so that attackers may not get it, but it is used to hide the existence of the information itself. Steganography is an art of concealing information in the ways that prevents detection of hidden information.[1]

However, to bolster the security of hidden information, additional measures are often necessary. In this paper, we explore the integration of Advanced Encryption Standard (AES) encryption with the LSB steganography technique. AES, recognized as one of the most secure encryption algorithms available, adds an extra layer of protection by encrypting the payload before embedding it using LSB, ensuring that even if the existence of hidden data is detected, its content remains unintelligible without the decryption key.

In our proposed system, we have extended the application of LSB steganography with AES encryption to various types of media files, including images, audio, and video. To evaluate the performance of our proposed system, we have conducted extensive experiments to measure both the payload capacity and the Peak Signal-to-Noise Ratio (PSNR). Through our experiments, we have calculated the payload capacity and PSNR values for different media files and varying levels of data embedding. To provide a clear understanding of the result we have graphically represented the results.

II. RELATED WORKS

Up till now, a lot of study has been done in the topic of steganography. Numerous articles about current steganography research and advances were examined. In essence, a literature review offers a method for conducting research inquiries and presents an overview of previous studies. Based on an analysis of these works that are relevant to our work, the following is a brief overview.

[2] highlights integration of AES encryption with LSB-based steganography presents a notable advancement in the field of image concealment techniques. This paper addresses this constraint by introducing a novel approach utilizing a randomly generated Pixel Locator Sequence (PLS) for data distribution across image pixels. By encrypting both the secret data and its pixel locations with AES, double encryption is achieved, enhancing the security of the steganographic method. This method not only disrupts sequential access to hidden data but also ensures its concealment in a pseudo-randomized order, thereby significantly bolstering imperceptibility and anti-detection performance. Despite the trade-off of requiring additional space for transmitting the PLS metadata, the improved security measures offered by this technique make it a better solution for various applications in fields such as medical, military, and copyright protection.

N.R. Zeynalov et al. [3] proposed a new addressed to text steganography which includes use of invisible Unicode characters, especially spaces, in text documents. Exploiting the appearance of space symbols in the text, the observed method allows data embedding without degrading the cover file. In particular, the algorithm works at the ASCII character level and emphasizes ease of implementation and efficiency. The proposed method requires replacing some characters in a text document with similar characters from a subset of Unicode to hide data. The substitution process and coding system are explained and how hidden information can be encoded into a document is shown. The mapping facilitates the extraction of hidden information by correlating substituted characters with their original counterparts, known as zero-width characters (ZWC). The proposal also talks about the advantages and limitations of this approach compared to other steganographic methods and highlights its potential applications in secure communication, watermarking and copyright protection.

[4] discusses LSB-based text and image steganography, which uses the AES algorithm to conceal secret information within cover images. The main objective is to conceal messages, images, or audio within images for security. AES encryption is employed to protect the hidden information. The combination of the LSB and the AES algorithm ensures the steganography's quality and security of the reconstructed image. The study in [5] successfully hides secret information in cover images using the LSB technique and AES algorithm, ensuring data security and quality. The maximum number of embedded characters is 8192 for hidden image size less than or equivalent to 80x80. The paper also discusses the challenges of secret information hiding, such as data invariance and security against interception. The algorithm's efficiency can be assessed using the PSNR and MSE measures, showing a proportional relationship between them in the experimental results.

Two recent studies present innovative approaches to embedding text and image data within audio files for covert communication. Abdulmalek A. S. Alqobaty [6] employs a unique audio steganography technique, dividing the cover audio into manageable blocks and concealing ASCII codes of text characters within them. By utilizing random selection of starting audio bytes and predefined tours, imperceptible alterations are made to (LSB) of audio bytes, ensuring secure transmission. Evaluation based on PSNR and MSE parameters offers a comprehensive understanding of the robustness and quality of the embedding process, ensuring reliable and secure transmission of hidden messages. This method provides a high quality of Peak Signal-to-Noise Ratio (PSNR) and Signal to Noise Ratio (SNR). Another technique focuses on embedding text data into audio files using tone insertion, leveraging two frequencies to increase payload capacity without compromising audio integrity. Introducing a convoy frequency (CF) for specific patterns enhances concealment capabilities, as confirmed by spectrogram analysis, MSE, and PSNR evaluations. These advancements signify promising strides in audio steganography, offering enhanced security and capacity for covert communication.[7]

[8] underscored the importance of balancing imperceptibility, robustness, and payload. The study by Hacimurtazaoglu and Tutuncu highlights the necessity of this balance, demonstrating that non-sequential data embedding combined with RGB channels and dynamic KBM rotation and shifting augment robustness and imperceptibility, albeit at the expense of implementation time. By offering trade-offs between these parameters, the proposed method contributes significantly to video steganography's security landscape, achieving notable imperceptibility and payload metrics. Further exploration into statistical attacks will enhance the understanding of the proposed system's robustness, while future research may explore KBM's application in the transform domain, promising additional insights into security, robustness, imperceptibility, and payload across different steganographic domains.

Least Significant Bits (LSB) substitution, Discrete Wavelet Transform (DWT), and Discrete Cosine Transform (DCT) are among the widely utilized techniques in video steganography for concealing secret information within video data. In a recent study by JayakanthKunhoth[9], a new strategy that combines both techniques was put up to improve the video steganography's resilience and security. The method introduced in [9] leverages LSB substitution as a fundamental technique for embedding data within the least significant bits of video frames. Video frames can also be converted into frequency domains with the combination of DWT and DCT, providing alternative embedding spaces and enhancing imperceptibility.

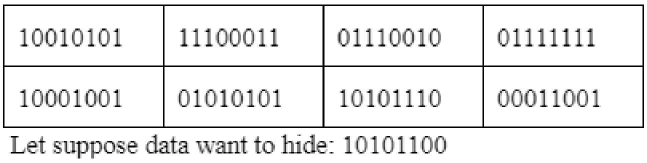
III. METHODOLOGY

A. *Unicode Standard*

Non-ASCII characters are supported by the global character encoding standard known as Unicode. All the languages in the world and their special characters are supported by Unicode. Unicode can actually support 100,000 characters. The logic is that Unicode, which uses characters made up of status bits—can represent multibit characters. Unicode characters can require 16 bits, but ASCII characters require only 7 bits. This is important because some languages, such as Arabic and Chinese, require extensive spatial information. Persian, Urdu, Pashto, Sindhi and Kurdish are among the languages added to the Unicode table for writing characters in Arabic and other languages. The standard contains a comprehensive description of the methods used, such as right-to-left text and alignment methods[10].

B. *Modified LSB Technique*

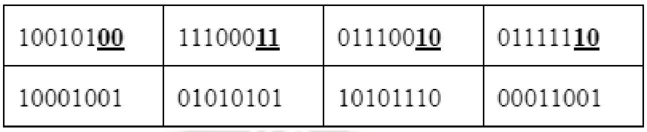
In Modified LSB technique in place of LSB bit, 2 or 3 bits from LSB side is replaced with data bits as shown in table 1 and 2.



10010101	11100011	01110010	01111111
10001001	01010101	10101110	00011001

Let suppose data want to hide: 10101100

Table 1. Cover Image Pixels



100101**00**	111000**11**	011100**10**	011111**10**
10001001	01010101	10101110	00011001

Table 2. Stego Image Pixels

When compared to the LSB approach, the modified LSB technique has a higher capacity, but its variability is higher[11].

C. *Advanced Encryption Standard*

The U.S. National Institute of Standards and Technology (NIST) established the Advanced Encryption Standard (AES) in 2001 as a specification for the encryption of electronic data. It was formerly known as Rijndael (pronounced [ˈrɛinda:l] in the Dutch language).

AES is an efficient software and hardware algorithm that is based on the substitution-permutation network design paradigm. With a constant block size of 128 bits and a key size of 128, 192, or 256 bits, AES is a Rijndael variation. In contrast, Rijndael in and of itself specifies block and key sizes that range from a minimum of 128 bits to a maximum of 256 bits, depending on the multiple of 32 bits. The majority of AES computations take place in a specific finite field.

The state, denoted by $b_0, b_1, ..., b_{15}$, is a 4 x 4 column-major order array of 16 bytes that is used by AES:

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

The number of transformation cycles required to change the input, called as the plaintext, into the final output, called as the ciphertext, is find out by the key size employed in an AES cypher. The following is the number of cycles:

- 10 cycles for keys with 128 bits.
- 12 cycles for keys with 192 bits.

- 256-bit keys are used in 14 cycles.

There are multiple processing steps in a round, one of which is dependent on the encryption key. Using the same encryption key, a series of reverse rounds are conducted to convert the ciphertext back into the original plaintext[12].

[13] outlines the process of deriving cryptographic keying material (MK) from user-chosen passwords using Password-Based Key Derivation Functions (PBKDFs). PBKDFs are deterministic algorithms that employ a Pseudorandom Function (PRF) and a fixed iteration count (C) to generate cryptographic keys from passwords. The input to a PBKDF execution includes the password (P), a randomly-generated salt (S) of at least 128 bits in length, and the desired length of the cryptographic key in bits (kLen), which must be at least 112 bits. The iteration count (C) determines how many times the PRF iterates to generate one block of the MK, with a recommended minimum iteration count of 1,000 and a maximum of 10,000,000 for especially critical keys. PBKDF2, using HMAC with any approved hash function as the PRF, is approved for use, with the digest size of the hash function denoted as hLen. This specification ensures that even with user-chosen passwords of low entropy, the derived cryptographic keys maintain a high level of security suitable for protecting data in storage devices.
Process of encryption and decryption using AES and PBKDF can be summarised as:

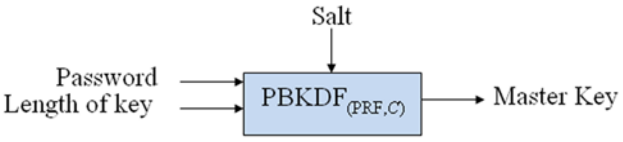
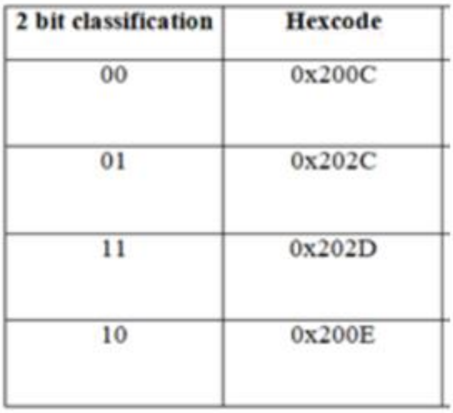


Figure 1. Generic PBKDF flow diagram

IV. Proposed System

A. *Text Steganography*

We are hiding text in a text file using Unicode. Unicode is the standardized encoding format for text, specifically, UTF-8, that most web browsers use for text. In Unicode, there are specific zero-width characters (ZWC) that are used to control special entities such as Zero Width Non Joiner (e.g., ZWNJ separates two letters in special languages) and POP directional, which have no written symbol or width in digital text. We used four ZWCs which are shown in the below table for hiding the Secret Message through the Cover Text.



2 bit classification	Hexcode
00	0x200C
01	0x202C
11	0x202D
10	0x200E

Table 3. ZWCs TABLE

1. Character Transformation and XOR Encryption:

- Obtain the ASCII value of each character in the cipher text.
- Increment or decrement the ASCII value based on whether it falls between 32 and 64, adjusting it by 48 (ASCII value for '0').
- XOR the obtained value with 170 (binary equivalent: 10101010) for encryption.

2. Binary Representation and Delimiter Addition:

- Convert the resultant value into the corresponding binary value.
- If the original ASCII value was between 32 and 64, add "0011" to the binary representation; if not, add "0110" to guarantee a 12-bit representation for every letter.
- Use "111111111111" as a delimiter to indicate that the message has ended.

3. Character Replacement with ZWCs:

- Divide each 12-bit binary representation into two-bit pairs.
- Using a preset table as a guide, replace each pair with the corresponding zero-width characters (ZWCs).
- Embed the modified characters within the cover text, placing each character after a word

Encoding process:

The flow chart of Encoding process of hiding Secret Text in Cover Text follows:

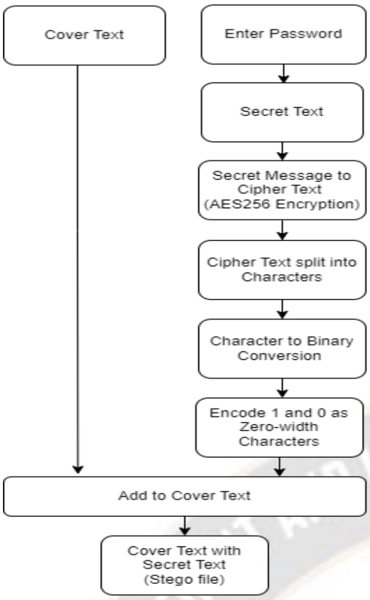

Cover Text
Enter Password
Secret Text
Secret Message to Cipher Text (AES256 Encryption)
Cipher Text split into Characters
Character to Binary Conversion
Encode 1 and 0 as Zero-width Characters
Add to Cover Text
Cover Text with Secret Text (Stego file)

Figure 1. Flowchart of Encoding Process in text

Decoding process :

The flow chart of Decoding process of extracting Secret Text from Steganography file as follows:

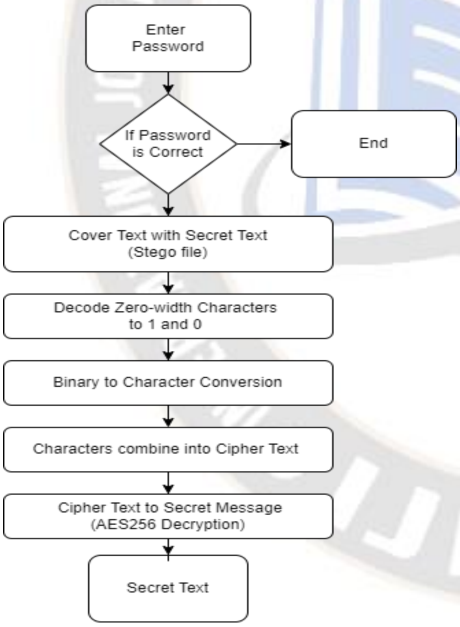

Enter Password
If Password is Correct
End
Cover Text with Secret Text (Stego file)
Decode Zero-width Characters to 1 and 0
Binary to Character Conversion
Characters combine into Cipher Text
Cipher Text to Secret Message (AES256 Decryption)
Secret Text

Figure 2. Flowchart of Decoding Process in text

B. *Image Steganography*

Image steganography using the Least Significant Bit (LSB) method involves hiding data within image pixels. This technique conceals information by altering the LSB of pixel values, allowing for the embedding of messages, audio, or images within the cover image. To enhance security, encryption methods like Advanced Encryption Standard (AES) are often combined with LSB steganography, providing double encryption for the data

In image steganography, data is hidden within an image during encoding, and is retrieved during decoding. The following describes the Image Steganography Encoding and Decoding Process:

1. Encoding:

Encryption: The text data is encrypted using a cryptographic algorithm (e.g., AES) along with a secret key. This ensures that the message is secure and cannot be easily deciphered by unauthorized parties.

Binary Conversion: The encrypted text is then converted into binary format. Each character is represented by its corresponding binary code.Embedding: Binary data is embedded in the LSBs of selected pixels in the cover image. This is typically done by altering the LSBs of the pixel values to encode the binary data.

2. Decoding:

Extraction: The Stego image is analyzed, and the LSBs of selected pixels are extracted.

Binary Reconstruction: The extracted LSBs are concatenated to reconstruct the binary data.

Decryption: The binary data is decrypted using the same cryptographic algorithm and secret key used during encoding. Text Retrieval: The decoded binary data is transformed back to text, revealing the concealed message.

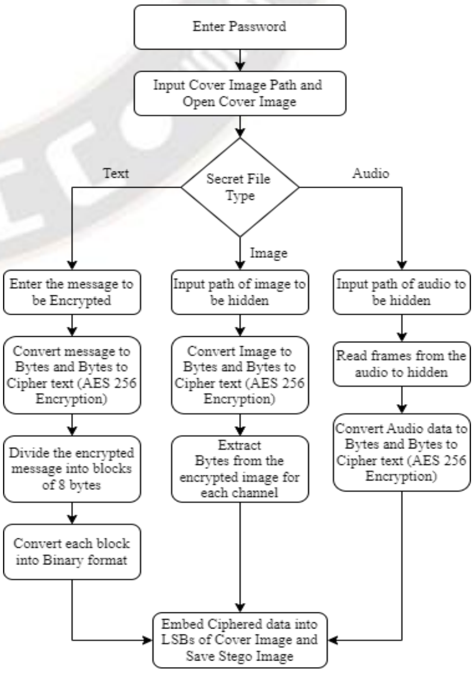

Enter Password
Input Cover Image Path and Open Cover Image
Secret File Type
Text
Audio
Image
Enter the message to be Encrypted
Input path of image to be hidden
Input path of audio to be hidden
Convert message to Bytes and Bytes to Cipher text (AES 256 Encryption)
Convert Image to Bytes and Bytes to Cipher text (AES 256 Encryption)
Read frames from the audio to hidden
Divide the encrypted message into blocks of 8 bytes
Extract Bytes from the encrypted image for each channel
Convert Audio data to Bytes and Bytes to Cipher text (AES 256 Encryption)
Convert each block into Binary format
Embed Ciphered data into LSBs of Cover Image and Save Stego Image

Figure 3. Flowchart of Encoding Process in image

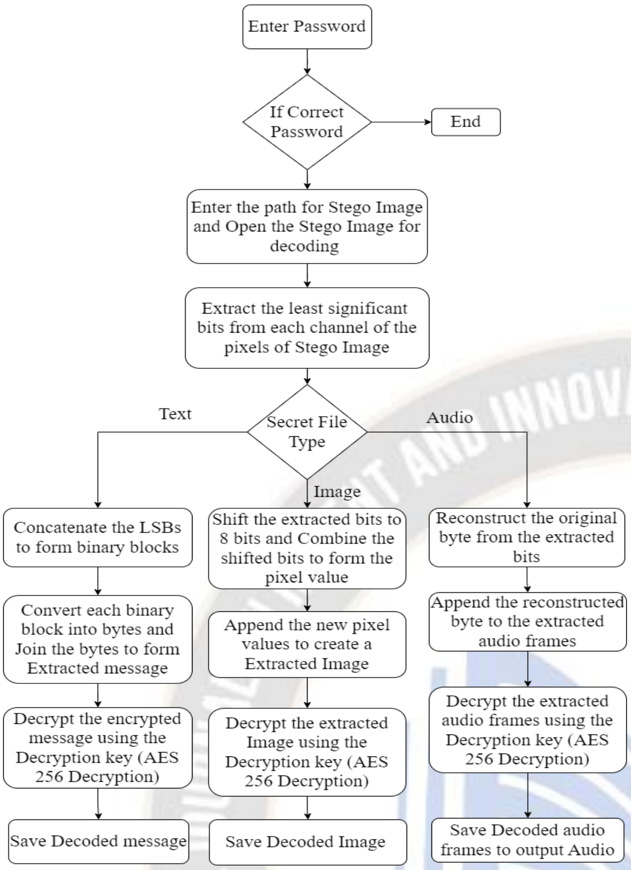

Enter Password
If Correct Password
End
Enter the path for Stego Image and Open the Stego Image for decoding
Extract the least significant bits from each channel of the pixels of Stego Image
Secret File Type
Text
Image
Audio
Concatenate the LSBs to form binary blocks
Convert each binary block into bytes and Join the bytes to form Extracted message
Decrypt the encrypted message using the Decryption key (AES 256 Decryption)
Save Decoded message
Shift the extracted bits to 8 bits and Combine the shifted bits to form the pixel value
Append the new pixel values to create a Extracted Image
Decrypt the extracted Image using the Decryption key (AES 256 Decryption)
Save Decoded Image
Reconstruct the original byte from the extracted bits
Append the reconstructed byte to the extracted audio frames
Decrypt the extracted audio frames using the Decryption key (AES 256 Decryption)
Save Decoded audio frames to output Audio

Figure 4. Flowchart of decoding process in image

C. Audio Steganography

In the proposed system for audio steganography, the fundamental understanding of digital audio representation plays a pivotal role. Digital audio files are essentially discretized representations of analog sound waves. This process begins with the sampling of the continuous analog signal at regular intervals, known as the sampling rate. At each sampling point, the amplitude of the analog signal is measured, capturing its intensity or loudness. These discrete measurements, known as samples, collectively form the digital representation of the audio signal.

A crucial aspect of this digital representation is the quantization of the sampled values. Quantization involves assigning numerical values to the sampled amplitudes, effectively digitizing the analog signal. The bit depth of the audio file determines the precision of this quantization process. Common bit depth includes 8-bit, 16-bit, and 24-bit, with each of the sample being represented by a fixed number of bits.

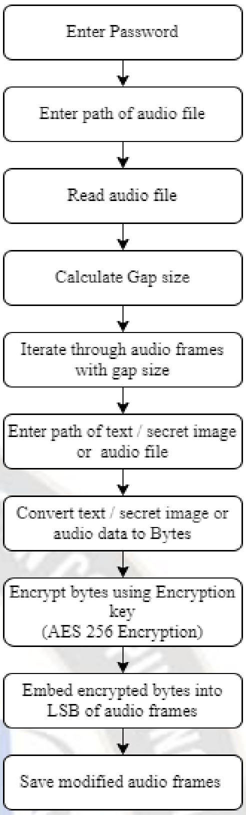

Enter Password
Enter path of audio file
Read audio file
Calculate Gap size
Iterate through audio frames with gap size
Enter path of text / secret image or audio file
Convert text / secret image or audio data to Bytes
Encrypt bytes using Encryption key (AES 256 Encryption)
Embed encrypted bytes into LSB of audio frames
Save modified audio frames

Figure 5. Flowchart of encoding process in audio

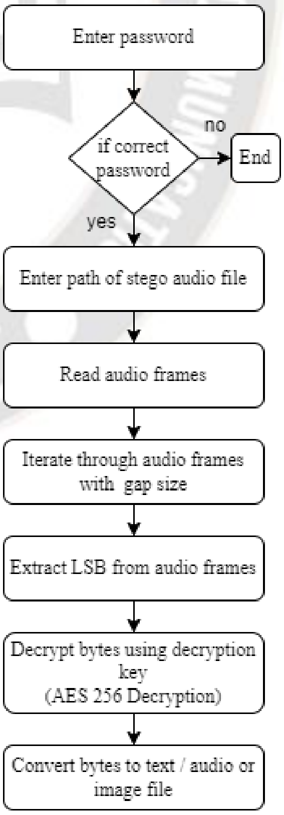

Enter password
if correct password
no
End
yes
Enter path of stego audio file
Read audio frames
Iterate through audio frames with gap size
Extract LSB from audio frames
Decrypt bytes using decryption key (AES 256 Decryption)
Convert bytes to text / audio or image file

Figure 6. Flowchart of decoding process in audio

Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 November 2023*

The proposed system leverages the (LSB) modification technique to embed bytes of data within the audio file while minimizing perceptual distortion. Since the LSBs contribute minimally to the overall amplitude of the audio signal, modifying them enables the insertion of additional data without significantly altering the audio quality. Bytes of data, such as text messages, photos, or audio, are transformed into binary form during embedding, and the LSBs of the audio samples are subtly altered to encode this binary data. This embedding process involves bitwise operations to ensure that the modifications are imperceptible to human listeners.

Every audio sample in the audio file is examined, and the LSB is extracted in order to retrieve the concealed data. The binary data that was encoded in the audio file is then rebuilt by concatenating these LSBs. The binary data will be transformed back into its original format, whether it was an image, audio file, or text message, when it has been rebuilt.

Video Steganography

Video steganography is employed as a method to conceal sensitive information within video files. As of now in our system, videos with only .avi extension are supported.

Video Steganography works for text/image and audio differently as follows:

For Text/Image

For embedding text/image, video needs to be broken down into frames and encrypted text/image is embedded in one of these frames chosen by the user using LSB. While decoding the exact reverse process occurs.

For audio

Embedding audio in video uses the concept of audio steganography to hide audio inside an audio. Firstly, audio from a video is extracted and this extracted audio acts as a cover audio file to execute audio in audio steganography for hiding encrypted secret audio. While decoding, decoding of audio in audio steganography is used to extract encrypted secret audio from audio of Stego video file and is decrypted to get original secret audio.

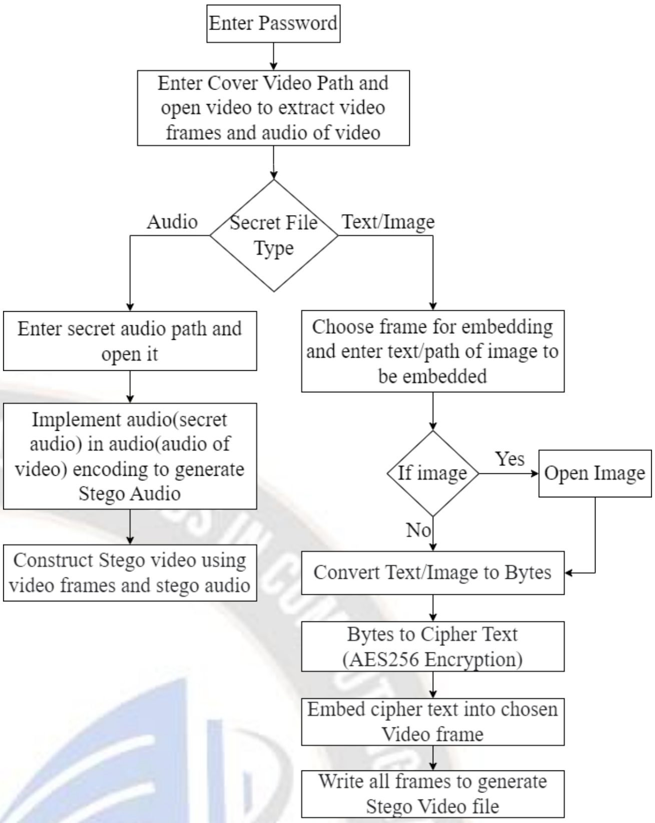

Enter Password
Enter Cover Video Path and open video to extract video frames and audio of video
Secret File Type
Audio
Text/Image
Enter secret audio path and open it
Implement audio(secret audio) in audio(audio of video) encoding to generate Stego Audio
Construct Stego video using video frames and stego audio
Choose frame for embedding and enter text/path of image to be embedded
If image
Yes
Open Image
No
Convert Text/Image to Bytes
Bytes to Cipher Text (AES256 Encryption)
Embed cipher text into chosen Video frame
Write all frames to generate Stego Video file

Figure 7. Flowchart of encoding process in video

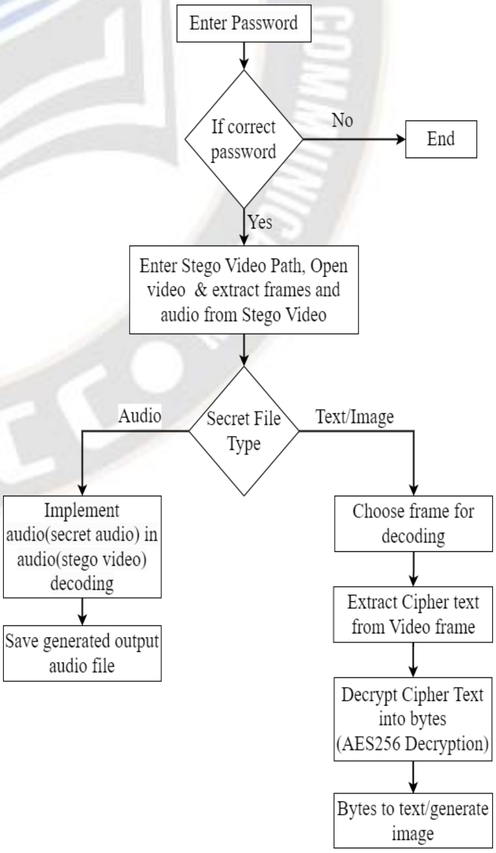

Enter Password
If correct password
No
End
Yes
Enter Stego Video Path, Open video & extract frames and audio from Stego Video
Secret File Type
Audio
Text/Image
Implement audio(secret audio) in audio(stego video) decoding
Save generated output audio file
Choose frame for decoding
Extract Cipher text from Video frame
Decrypt Cipher Text into bytes (AES256 Decryption)
Bytes to text/generate image

Figure 8. Flowchart of encoding process in video

V. RESULT AND PERFORMANCE ANALYSIS

A. ***Payload Capacity:***Payload capacity in steganography refers to the maximum amount of hidden data that can be embedded within a cover media (such as text, image, audio, or video) without affecting its appearance or quality. It's influenced by cover medium properties, embedding technique, and desired secrecy.

A simplified formulae for estimating payload capacity for different types of Steganography are given as follows:

1. Payload Capacity for Text Steganography:

$$\boxed{\text{Payload Capacity} = L_{\text{cover}} \times \text{Modification Rate}}$$

Where:

L_{cover} is the length of the cover text (in characters).

Modification Rate represents the proportion of modification points in the cover text where you can embed hidden information.

For example, if you have a cover text of 1000 characters and the modification rate is 0.1 (meaning 10% of characters can be modified without noticeable changes), then the estimated payload capacity would be 100 characters.

2. Payload Capacity for Image Steganography:

$$\boxed{\text{Payload Capacity} = \text{Cover Image Size} \times \text{Embedding Rate}}$$

Where:

Cover Image Size represents the total number of pixels in the cover image.

Embedding Rate is the proportion of pixels where hidden data can be inserted.

For example, if you have a cover image with dimensions 800x600 pixels (480,000 pixels in total) and the embedding rate is 0.1 (meaning 10% of pixels can be modified without noticeable changes), then the estimated payload capacity would be 48,000 bits.

3. Payload Capacity for Audio Steganography:

$$\boxed{\begin{aligned}\text{Payload Capacity} = {} & \text{Audio Duration} \times \text{Sampling Rate} \times \\ & \text{Number of Channels} \times \text{Bit Depth} \times \\ & \text{Embedding Rate}\end{aligned}}$$

Where:

Audio Duration is the length of the audio in seconds.

Sampling Rate is the number of samples captured per second.

Number of Channels is the number of audio channels (e.g., mono or stereo).

Bit Depth is the number of bits used to represent each sample.

Embedding Rate is the proportion of LSBs in each sample that can be modified without noticeable changes.

For example, if you have a stereo audio file with a duration of 60 seconds, a sampling rate of 44100 Hz, a bit depth of 16 bits, and an embedding rate of 0.1 (meaning 10% of LSBs can be modified without noticeable changes), then the estimated payload capacity would be:

Payload Capacity = 60 × 44100 × 2 × 16 × 0.1 = 84 , 672 , 000 bits Payload Capacity = 60 × 44100 × 2 × 16 × 0.1 = 84,672,000 bits

4. Payload Capacity for Video Steganography:

$$\boxed{\begin{aligned}\text{Payload Capacity} = {} & \text{Video Duration} \times \text{Frame Rate} \times \text{Frame} \\ & \text{Resolution} \times \text{Embedding Rate}\end{aligned}}$$

Where:

Video Duration is the length of the video in seconds.

Frame Rate is the number of frames per second.

Frame Resolution is the number of pixels in each frame (width x height).

Embedding Rate is the proportion of modification points in each frame where hidden data can be inserted.

For example, if you have a video with a duration of 120 seconds, a frame rate of 30 frames per second, a frame resolution of 1920x1080 pixels, and an embedding rate of 0.1 (meaning 10% of modification points can be modified without noticeable changes), then the estimated payload capacity would be:

Payload Capacity = 120×30×(1920×1080)×0.1

= 746,496,000 bits

B. ***Mean Squared Error (MSE):***A popular statistic in machine learning, image processing, and signal processing is the class error. It calculates the difference between the actual and predicted values of the data set. Mean Squared Error (MSE) is a statistic used in steganography to evaluate the quality of face images with hidden information. To calculate it, the stego medium and the original cover medium are compared; A low MSE indicates good conservation. Low MSE values indicate poor detectability. The MSE also helps to detect the invisibility of hidden data in the envelope image. However, the MSE does not adequately capture differences in sensitivity.

The generalized formula for MSE can be expressed as follows:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Where:

Y_i represents the i i-th sample in the original data.

$\hat{Y}_i$ represents the corresponding sample in the stego data.

N is the total number of samples in the data.

C. ***Peak Signal-to-Noise Ratio (PSNR):*** In steganography, a statistic called PSNR, or Peak Signal-to-Noise Ratio, is used to assess how well a Stego media containing hidden data compares to the original cover media. It assesses the faithfulness of the representation by calculating the ratio between the maximal power of a signal and the power of corrupting noise. Higher PSNR readings mean better quality and lower noise. Higher values mean less visibility. It is often used to check the non-obtrusiveness of hidden information in an overlay image. However, PSNR may not be sufficient to adequately capture changes in sensitivity; Therefore, steganographic methods are usually evaluated using complementary metrics and subjective evaluations.

The formula for calculating PSNR can be expressed as follows:

$$\boxed{PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right)}$$

Since MSE for all steganography have text as secret file gives MSE as 0, PSNR would go infinite. Hence, plots for all PSNR consisting of text as secret file is not included.

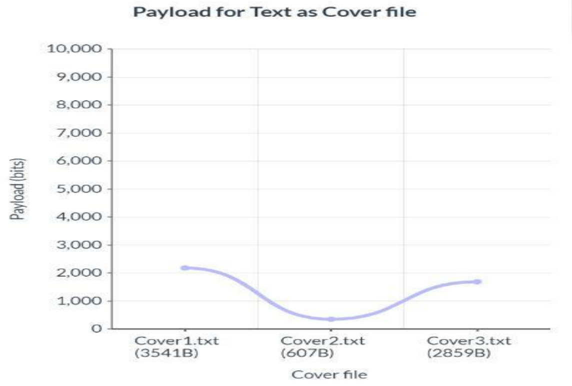


Figure 9. Payload for Text as Cover File

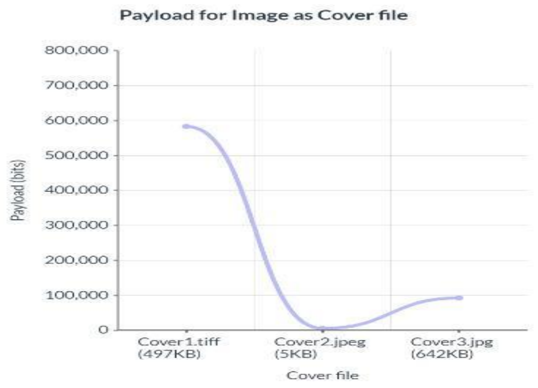


Figure 10. Payload for Image as Cover File

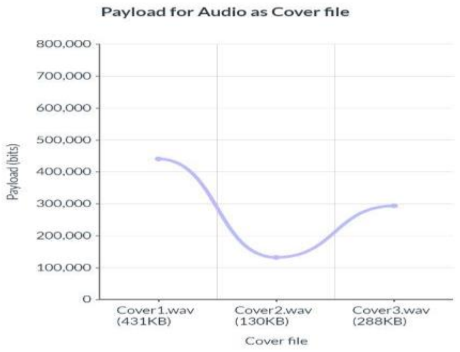


Figure 11. Payload for Audio as Cover File

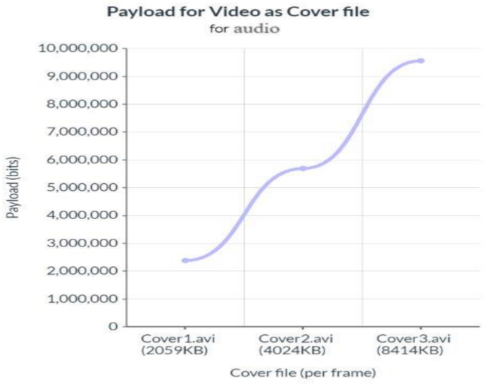


Figure 12. Payload for Video as Cover File for Audio

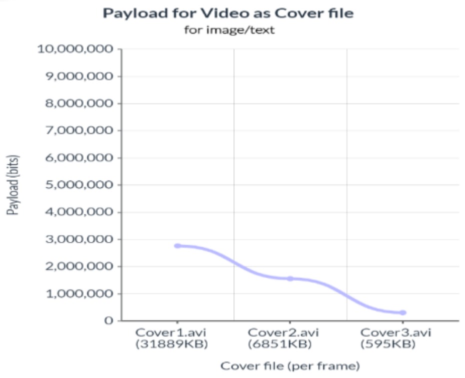


Figure 13. Payload for Video as Cover File for image/text

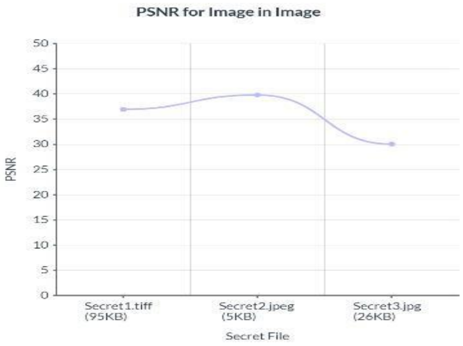


Figure 14. PSNR for Image in Image

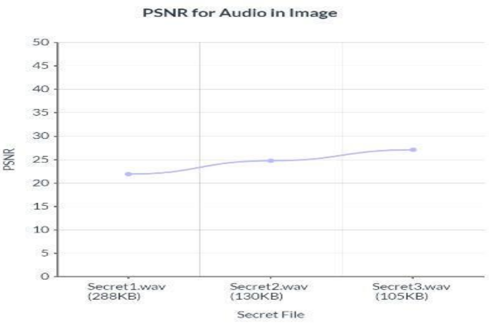


Figure 15. PSNR for Audio in Image

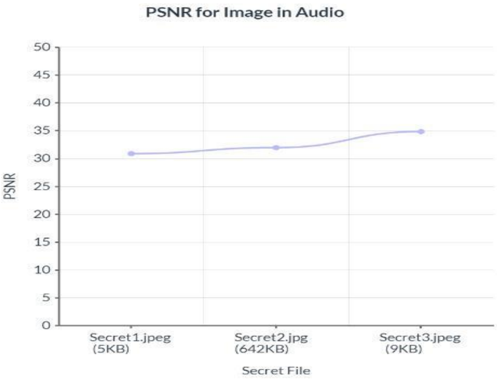


Figure 16. PSNR for Image in Audio

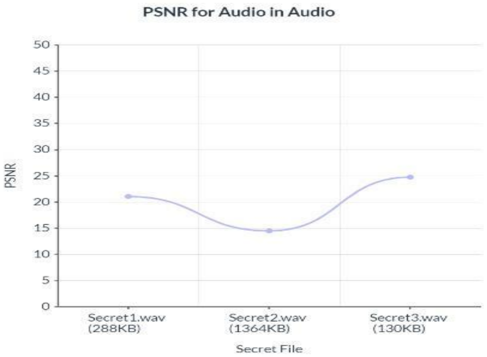


Figure 17. PSNR for Audio in Audio

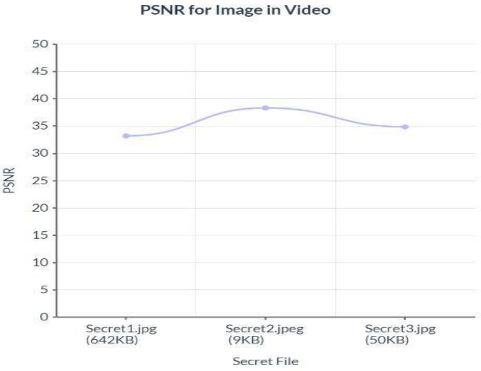


Figure 18. PSNR for Image in Video

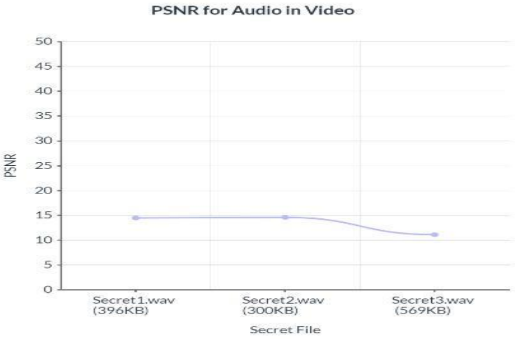


Figure 19. PSNR for Audio in Video

VI. CONCLUSION

In this study, we proposed steganography techniques for text, image, audio, and video multimedia domain names. Our research discovered the effectiveness and demanding situations of encrypting facts in multimedia files without compromising the perceived best. We evolved a beneficial steganographic tool to help with checking out and evaluation on information structures.We examined the blessings and drawbacks of the various loading methods in terms of power, safety and robustness. Our results highlight the importance of steganography for stable communication and content preservation in multimedia systems. Subsequent research should take note of steganographic algorithms developed to solve new problems and improve security features. Specifically, our research is especially helpful in multimedia steganography, and from the doorstep to the creative methods of record keeping and privacy protection underneath.

REFERENCES

[1] JeeveshPasrija,"AUDIO STEGANOGRAPHY USING LSB TECHNIQUE", 2020 JETIR June 2020, Volume 7, Issue 6

[2] N. V. A. Ravikumar, R. S. S. Nuvvula, P. P. Kumar, N. H. Haroon, U. D. Butkar and A. Siddiqui, "Integration of Electric Vehicles, Renewable Energy Sources, and IoT for Sustainable Transportation and Energy

Management: A Comprehensive Review and Future Prospects," 2023 12th International Conference on Renewable Energy Research and Applications (ICRERA), Oshawa, ON, Canada, 2023, pp. 505-511, doi: 10.1109/ICRERA59003.2023.10269421.

[3] A. K. Bhaga, G. Sudhamsu, S. Sharma, I. S. Abdulrahman, R. Nittala and U. D. Butkar, "Internet Traffic Dynamics in Wireless Sensor Networks," 2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, 2023, pp. 1081-1087, doi: 10.1109/ICACITE57410.2023.10182866.

[4] Omar Elharrouss, Noor Almaadeed, Somaya Al-Maadeed, “An image steganography approach based on k-least significant bits (k-LSB)”, ©2020 IEEE

[5] Butkar, M. U. D., &Waghmare, M. J. (2023). Crime Risk Forecasting using Cyber Security and Artificial Intelligent. Computer Integrated Manufacturing Systems, 29(2), 43-57.

[6] Abdulmalek A. S. Alqobaty Computer Science Department, Faculty of Applied Sciences, University of Taiz, Yemen,“A Robust Audio Steganography Method Using Partial Knight Tour for Concealing Messages of Text and Image”

[7] SuhaipA.Yousif, TalaatM.wahbiand and Mohamed H. Sayed “Audio Steganography Using Tone Insertion Technique”, International Journal of Computer Applications Technology and Research Volume 6–Issue 6, 254-258, 2017, ISSN:-2319–8656

[8] Uamakant, B., 2017. A Formation of Cloud Data Sharing With Integrity and User Revocation. International Journal Of Engineering And Computer Science, 6(5), p.12.

[9] JayakanthKunhoth, Nandhini Subramanian, Somaya Al-Maadeed, Ahmed Bouridane, “Video steganography: recent advances and challenges” Accepted: 6 February 2023 / © The Author(s) 2023

[10] Butkar, M. U. D., &Waghmare, M. J. (2023). Hybrid Serial-Parallel Linkage Based six degrees of freedom Advanced robotic manipulator. Computer Integrated Manufacturing Systems, 29(2), 70-82.

[11] Shivani Chauhan, Jyotsna, Janmejai Kumar, Amit Doegar “Multiple layer Text security using Variable block size Cryptography and Image”

[12] Butkar, U. (2016). Review On-Efficient Data Transfer for Mobile devices By Using Ad-Hoc Network. International Journal of Engineering and Computer Science, 5(3).

[13] MeltemSönmezTuran, Elaine Barker, William Burr, and Lily Chen, “Recommendation for Password-Based Key Derivation”, NIST Special Publication 800-132, December 2010.