ADVANCED SECURITY USING ENCRYPTION, COMPRESSION AND STEGANOGRAPHY TECHNIQUES

Mr. R Ramesh Kumar[1], B Amirta Josna[2], R Lawvanyaa[3], S Shruthi[4]

[1]Assistant Professor, Sri Venkateswara College of Engineering, Sriperumbudur, TamilNadu
[234]Students of Electronics and Communication Engineering, Sri Venkateswara College of Engineering, Sriperumbudur, TamilNadu

---***---

Abstract – *In today's digital age, data security and privacy are more crucial than ever. One of the most trustworthy and widely used symmetric encryption methods is the Advanced Encryption Standard (AES). AES provides robust encryption, but it doesn't address the issue of reducing file size or concealing the fact that data is encrypted in the first place. In this case, compression and steganography are employed. Using seemingly harmless carrier things, such as images, audio files, or text documents, steganography is the technique of hiding data. Data that has been well compressed is reliable, safe, and simple to connect. AES encryption, lossless Huffman coding compression, and LSB embedding steganography are all used in this project. Steganography is performed after encryption, followed by compression. Java script is used to implement this, along with SQL, FXML, and CSS. After encryption, compression, and steganography, measurements were taken to determine the Avalanche Effect (AE), entropy, and compression ratio. By using this hybrid approach, it has been discovered that the values of entropy and the Avalanche Effect significantly rise, and the file size is optimally reduced by around 30%, compensating for the size increase brought on by encryption. In order to facilitate quick transmission over slow internet or take up less space on various storage media while maintaining high security and data integrity, this technology reduces the amount of each piece of communicated data.*

***Key Words*: AES, Huffman Coding, LSB Embedding, Entropy, Compression Ratio, CSS, Java, FXML, SQL**

1.INTRODUCTION

The protection of sensitive information has become vital in the current digital era. To fulfil this need, a combination of strong encryption and covert data concealment techniques is needed. The Advanced Encryption Standard (AES) cryptography, compression, and steganography are all merged in this project. Strong AES encryption ensures data secrecy, while compression minimises data size, and steganography conceals encrypted data inside harmless carrier files. Combining these techniques results in a double layer of security that makes it very challenging for unauthorised individuals to access or find the hidden data. This project uses Huffman compression, AES encryption, and LSB embedding steganography to hide data within carrier files.

LITERATURE REVIEW

Manish Kumar, Aman Soni, Ajay Raj Singh Shekhawat, Akash Rawat proposed a work on “Enhanced Digital Image and Text Data Security Using Hybrid Model of LSB Steganography and AES Cryptography Technique”. It uses a hybrid model of LSB steganography and Advanced Encryption Standard (AES) cryptography techniques to improve the security of digital images and text that is indubitably difficult for an unauthorized person to crack. The mean square error and peak signal-to-noise ratio are used to determine the security degree of secret information. A low MSE and a high PSNR value are needed for better concealment. [1]

Aswini. N, Anagha. V. K, NandaKumar. R proposed work on “Sharing Confidential Images with Abbreviated Shares using Steganography and AES Algorithm”. In this study, an experimental method for concealing a secret image utilising steganography (LSB and BPCS) and the AES algorithm is described. The cover picture's least significant bits are used to embed data in a way that makes it impossible for the naked eye to distinguish the image from the cover file. This technique is known as least significant bit (LSB) embedding. Both a 24-bit and an 8-bit environment could benefit from this strategy. In BPCS steganography, the image serves as the vessel data, and the bit planes of the vessel contain the hidden information. Every "noise-like" area within the bit planes of the vessel image is replaced with secret data without affecting the image's quality. The term "BPCS-Steganography," or Bit-Plane Complexity Segmentation Steganography, may be used to refer to this steganography.
[2]

Ababil Naghiyeva, Kamran Akbarzadeh, Sakit Verdiyev proposed work on “New Steganography Method of Reversible Data Hiding With Priority to Visual Quality of Image” - - a new reversible data-hiding technique that prioritises maintaining the stego image's visual quality while maintaining a high level of data-hiding capacity. Two steps are taken to resolve the issue. The smallest pixel is selected within each block of the input image that is separated into squares of size 22 in the first phase. The next step is to determine the difference between the last three pixels and the smallest pixel. The resulting difference's decimal values are transformed into binary values. In order to maximise the payload capacity in the second stage, secret information is concealed in calculated binary pixels using the LSB. The experiment's findings supported the effectiveness of the suggested approach, which entails using the best stego image visual quality and a respectable level of data hiding capacity. [3]

Mutia Rizky Ashila, Nur Atikah, De Rosal Ignatius Moses Setiadi, Eko Hari Rachmawanto proposed work on "Hybrid AES-Huffman Coding for Secure Lossless Transmission". This study suggests combining AES and Huffman to create encrypted files that can be compressed without losing any data. when the compression step is carried out after encryption. Entropy following encryption and compression and the avalanche effect (AE) were used to measure the level of file security. Based on the test findings, it has been established that AES encryption causes files to grow by about 25% of their original size. However, following Huffman compression, the encrypted file code shrank by roughly 30%, indicating that the compressed file size was less than the uncompressed file size. However, the Huffman file has a favorable impact from a security standpoint, as there is a notable increase in the value of AE and entropy. [4]

Awdhesh K. Shukla, Akanksha Singh, Balvinder Singh, Amod Kumar worked on "A Secure and High-Capacity Data-Hiding Method Using Compression, Encryption and Optimized Pixel Value Differencing". A high-capacity data hiding technique is provided that makes use of least significant bit (LSB) substitution, modified pixel value differencing (MPVD), lossless compression, and the Advanced Encryption Standard (AES). On the hidden message, arithmetic coding was used for lossless compression, which resulted in a 22% increase in embedding capacity. AES encryption is applied to the compressed secret message, increasing protection against steganalysis attacks. LSB subtraction and MPVD are used after compression and encryption. In MPVD, a combination of 3x3 and 2x2 blocks or adaptive non-overlapping 3x3 pixel blocks are employed in raster fashion. The suggested method's employment of mathematical compression and MPVD allowed for a significant improvement in embedding capacity and the ability to incorporate 214132 more bits than other approaches, according to experimental evidence. Combining MPVD and arithmetic coding produced a 25% increase in embedding capacity over previous approaches. The suggested approach offers excellent visual quality as well, with an average of 36.38 dB at 4.00 bpp. Additionally, it is demonstrated that the suggested approach is resistant to regular/singular (RS) steganalysis.[5]

3.THEORETICAL BACKGROUND

3.1 AES ENCRYPTION

The Advanced Encryption Standard is a popular encryption technique that use symmetric-key cryptography to protect sensitive data. AES is superior to other algorithms because it can withstand Cache-Timing attacks, which frequently happen throughout the encryption process.

The steps involved in AES Encryption:

Step I – Key Expansion:

• To create a set of round keys, the secret encryption key, which is commonly 128, 192, or 256 bits long, is expanded into a key schedule.

• A key expansion algorithm is used to create the key schedule, which entails subjecting the original key to a number of cryptographic operations like substitution, permutation, and XOR.

Step II – Initial Round:

• Fixed-size blocks of the plaintext picture data, usually 128 bits apiece, are used to organise the data.

• Each block is XORed with the relevant round key from the key schedule in the first round of encryption.

Step III – Main Round:

Depending on the key size, AES commonly uses numerous rounds: 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Each round's operations comprise the following:

1. Add Round Key: At this point, use the plaintext that has been converted to ciphertext and perform an XOR operation with the cypher key.
2. Sub bytes: Use the Rijndael S-box substitution matrix table to substitute bytes. The S-Box in the Rijndael holds hexadecimal data, which can be either numbers or letters.
3. Shifts Row: Shifting lines that are dependent on the specified byte shift; a 1-byte shift entails one leftward shift.
4. Mix Column: Creates a new block of cyphers by multiplying the cypher block by an existing matrix, using the dot product to scramble the data.

With the exception of the final round, which omits the Mix Columns procedure, the aforementioned operations are carried out repeatedly for each round.

3.2 HUFFMAN CODING

One form of data compression is the Huffman code, which translates characters that appear frequently into shorter bits while converting characters that appear infrequently into longer bits. Huffman Coding involves the following processes.

Step I - Building a Huffman tree utilising the input symbols' weights and frequencies

• It is necessary to design a Huffman tree, which resembles a binary tree data structure and has n leaf nodes and n-1 interior nodes.

• The Huffman tree is constructed using the priority queue, where nodes with the lowest frequency are given the highest priority. The features of a priority queue can be implemented using a Min Heap data structure.

• At first, all nodes are leaf nodes that include the character itself and its weight and frequency.

• In contrast, internal nodes have weight and connections to two child nodes.

Step II - navigating the Huffman tree to assign the binary codes to each symbol

• Typically, bits '0' and '1' correspond to the left and right children, respectively.

Algorithm for creating the Huffman Tree-

Step 1: Make a leaf node for each character, then, using all of the nodes, build a min heap using the frequency value to compare two nodes in the min heap.

Step 2: If the heap has more than one node, repeat Steps 3 to 5.

Step 3: Remove two nodes from the heap that have the lowest frequency, such as x and y.

Step 4: Make a new internal node called Z with the children X on the left and Y on the right. frequency (z) = frequency (x + y)

Step 5: Add z to the minimum heap.
Step 6: The Huffman tree's root is the last node in the heap.

3.3 LSB STEGANOGRAPHY

The art of steganography involves hiding a file inside another so that no one else can read the contents of the embedded object or even notice its presence. The safeguarding of the contents of the hidden information is the primary goal of steganography. Because so much extra room is created while storing photos, they are perfect for concealing information. Since there are various ways to accomplish this, image steganography is researched, and one of these techniques—the LSB technique—has been used to demonstrate it here. This technique modifies the least significant bit of each byte to create the bitstring that represents the embedded file.The human eye typically does not perceive changes to colour brought on by changing the LSB because they are so slight. Steganography is now being used in digital technologies. The methods have been utilised to encode music data in the ever-popular mp3 audio file as well as create the watermarks that are on our country's currency. Copyrights can be recorded in files, and those who violate copyright agreements can be found using their fingerprints. While simple graphic alterations and minor colour scheme adjustments will completely obliterate the secret message.

4.METHODOLOGY

In this section, overall implementation methodology is discussed. In section 4.1 proposed work is discussed in detail. In section 4.2 block diagram of the system is explained.

4.1 PROPOSED WORK

In this project, three operations are carried out: file encryption using AES, file compression using Huffman coding, and file concealment using LSB Embedding Steganography. This applies to documents, images, and plain text. To assess the resistance to differential attacks and statistical attacks, respectively, parameters like information entropy and avalanche effect (AE) are assessed. Prior to and following compression, the ratio of file size is measured. It is common practise to test the unpredictability of files after encoding using entropy. Nearly 8 is the optimal entropy value. Entropy is evaluated using formula (1):

$$E(x) = \sum_{i=0}^{N-1} f(xi\,) \log_2 f(xi) \qquad -(1)$$

Where $f(xi)$ is the occurrence probability of xi and $\log_2 f(xi)$ is the probability of occurrence of xi in a logarithmic basis 2^8values generally have 256 possibilities of message x with the same probabilities, where 256 is possible because generally calculations are carried out in byte size (2^8).

Formula (2) can be used to get the percentage change in size ratio. This will determine the effectiveness of file compression.

$$r = \frac{\text{Us}-\text{Uc}}{\text{Us}} \times 100\% \qquad -(2)$$

Where Us is uncompression size, Uc is compression size and r is the ratio.

If changes are made to the plaintext or key, a successful encryption method should be able to generate a new ciphertext. The formula AE given in formula (3) can be used to compute this. The ideal AE value has just a 1-bit change in the key or plaintext, and approximately a 50% difference between ciphertext 1 and ciphertext 2.