WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Secured Data Transmission on the Network Using Image Steganography

Ayodeji Ireti Fasiku[1]*, Abiodun Dare Kehinde[2]

[1]*Computer Engineering Department, Ekiti State University, Ado – Ekiti, Nigeria*
[2]*Computer Engineering Department, Ekiti State University, Ado – Ekiti, Nigeria*
[*]*ayodeji.fasiku@eksu.edu.ng*
DOI : https://doi.org/10.55248/gengpi.6.0825.3066

ABSTRACT

The security of data and information transmission on the network has become one of the most significant concerns for safety and efficient communication. In solving this problem, different methods have been employed, yet intruders still have access to or hack the information. This paper presents a technique for securing data transmission over a network using image steganography. The primary goal is to hide sensitive data within a cover image in such a way that the hidden information is undetectable to unauthorized individuals. This technique encrypted data within image files using the Least Significant Bit (LSB) embedding method combined with encryption to enhance security. We ensure that the hidden information is both undetectable and accessible only to authorized users. The design implementation includes preprocessing, embedding, extraction processes, and analysis of the Secured Data Transmission Using Image Steganography.

Keywords: Image Steganography, Cover Image, Image Encryption, Encrypted Data, Plaintext, steganography, transmission, cryptography, algorithm, LSB.

1. Introduction

Internet technology is one the main driven force in every section in this generation, it has a lot of advantages yet it gives room to many unauthorized users to have access to organization and private data. Information is very important to the users but when third party get access to it, it might not be able to serve the initial purpose [1]. Therefore, there is need to device a medium on how these information and data can be secured from unauthorized users. Secured data transmission is the process through which confidential and proprietary data or information are transferred over a secured channel [2].

Cryptography is the science of keeping the transmitted data secure, it involves encryption and decryption of the data for secure communication [3],[4]. The encryption process is applied before transmission, and the decryption process is applied after receiving the encrypted data [2]. Cryptography is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it [5]. Encryption is the process of encoding messages or information in such a way that only the authorized parties or users can read or access it. Though it does not prevent interception by the intruder but denies intruder access to the message content. Decryption is the process of decoding data that has been encrypted into a readable format by the end user [1].

Secured data transmission has become a critical aspect of information technology, especially in this digital age where data breaches are increasingly daily [1],[3],[4]. One of the promising techniques for secure data transmission is image steganography, which involves embedding hidden information within an image file, making it invisible to the naked eye [6]. Steganography is the art of hiding information, provides an additional layer of security by concealing the very existence of the message [5],[7]. Introduction of steganography makes data and information protection against intruders more secure and effective because the data or information appear like image to the third party or hackers [7]. Steganography is the practice of hiding data and information within an image or video. Traditional encryption methods secure data but often attract intruder attention [8]. The advantage of the proposed method is that the intended secret messages does not attract attention to itself as an object of investigation or scrutiny.

This paper explores the use of image steganography to transmit data securely over the network by preventing hackers from having access to personal or organization data and information. The main contributions of this paper are as follow:

1) Developed security tools based on steganography techniques for hiding and getting data using the encryption and decryption method.
2) Data concealment, the embeds secret data within a cover image in an undetectable manner.
3) Security, encrypt the hidden data to ensure only authorized users can access it.
4) Data integrity, ensure the hidden data remains intact and unaltered during transmission.

5) Undetectability, prevent detection of the hidden data through visual inspection or statistical analysis.

The rest of this paper is structured as follows. Section 2 reviews the related works. Section 3 presents overview of steganograph. Section 4 present LSB technique, while Section 5 is the research methodology. Section 6 discuss the proposed algorithms and Section 7 is the research results and discussion. The conclusion of the paper is in Section 8 with recommendations for future works.

2. Related Research Work

Image steganography involves embedding secret data within digital images in a way that it is imperceptible to human vision. This paper explores various methods, advancements, and challenges in the field of image steganography to secure data transmission. The primary goal is to conceal the existence of the data rather than to encrypt it. Various techniques have been developed, each with its strengths and weaknesses:

Chan and Cheng (2004) [9] proposed a Secure Data Hiding Technique Using LSB Insertion, they developed a method for hiding data within digital images using simple Least Significant Bit (LSB) insertion. The study demonstrated that while LSB is a straightforward and widely used technique, it is vulnerable to various forms of steganalysis [1]. The authors proposed modifications to the LSB method to improve its resistance to detection, such as embedding the secret data in randomly selected LSBs rather than sequentially. Also, Kumar and Kumar (2010) [10] proposed an enhanced LSB steganography technique that uses multiple LSBs to increase the payload capacity while maintaining image quality. This method addresses the limitation of traditional LSB techniques by allowing more data to be hidden without significant degradation of the cover image.

Bandyopadhyay et al. (2006) [11] proposed Image Steganography Using Block-DCT with Multiple JPEG Images, the proposed method operates in the transform domain using the Discrete Cosine Transform (DCT). The technique involves embedding secret data across multiple JPEG images to increase robustness against compression and other manipulations [2], [6]. This approach leverages the frequency domain's properties to enhance security and reduce the likelihood of detection. In research on wavelet-tree-based watermarking method using the SPIHT algorithm proposed by Lin et al. (2008) [12]. This research explores a wavelet-based steganographic approach using the Set Partitioning in Hierarchical Trees (SPIHT) algorithm. The method embeds data into the wavelet coefficients of the cover image, which are then compressed using SPIHT. The technique is shown to be effective in maintaining image quality and increasing robustness against lossy compression.

Fridrich *et al*. (2003) [13] proposed an adaptive steganographic technique that analyzes the texture of the cover image to determine the most suitable embedding regions. By focusing on areas with high texture complexity, the method reduces the risk of detection by minimizing the visual impact of the embedded data [4],[7]. An Hybrid Steganography Using LSB and DCT for Enhanced Security was proposed by Cheddad *et al.* (2010) [1]. They developed a hybrid steganographic method that combines LSB insertion with Discrete Cosine Transform (DCT). The hybrid approach improves both the robustness and security of the steganographic process by embedding data in both the spatial and frequency domains of the image [4],[11],[15]. The method is particularly effective against statistical steganalysis attacks.

Fridrich et al. (2001) [8] proposed Reliable detection of LSB steganography in colour and grayscale images. It is a robust steganography using wet paper codes. The authors introduced the concept of wet paper codes to enhance the security of steganographic techniques [3],[16]. The wet paper codes allow for the embedding of data in a way that is resistant to detection and manipulation, providing an additional layer of protection against steganalysis. Ker (2005) [6] in his work Steganalysis of LSB matching in grayscale images. The author conducted a study on the steganalysis of LSB matching, a common steganographic technique used in grayscale images [9],[10],[14]. The research introduced methods for detecting hidden data by analyzing pixel correlations and statistical properties, demonstrating that even subtle modifications in LSB matching can be detected with high accuracy.

3. Overview of Steganography

Steganography techniques have been used for ages and they date back to ancient Greece. The aim of steganographic communication back then and now, in modern application, is the same, which is to hide secret data (a steganogram) in an innocently looking cover and send it to the proper recipient who is aware of the information hiding procedure [2],[5],[17]. In an ideal situation the existence of hidden communication cannot be detected by third party.

Steganography system requires any type of image file and the information or message that is to be hidden. It has two modules encrypt and decrypt. Microsoft. Net framework prepares a huge amount of tool and options for programmers that they simples programming [18]. The common python tools for pictures and images are auto-converting of most types of pictures to BMP format. It is used in this software called “Steganography” that is written in python languages and it can be used to hide information in any type of pictures without converting its format to BMP.

The encrypt module is used to hide information into the image; no one can see that information or file. This module requires any type of image in bitmap format. The decrypt module is used to get the hidden information in an image file. It takes the image file as an output, and give two files at destination folder, one is the same image file and another is the message file that is hidden in it [14]. Hence, before encrypting file inside image we must save name and size of the file in a definite place. The file name can be saved before file information in LSB layer and save file size and file name size in most right-down pixels of image [10],[11]. Writing this information is needed to retrieve file from encrypted image in decryption state. The graphical representation of this system is shown in figure 3.1

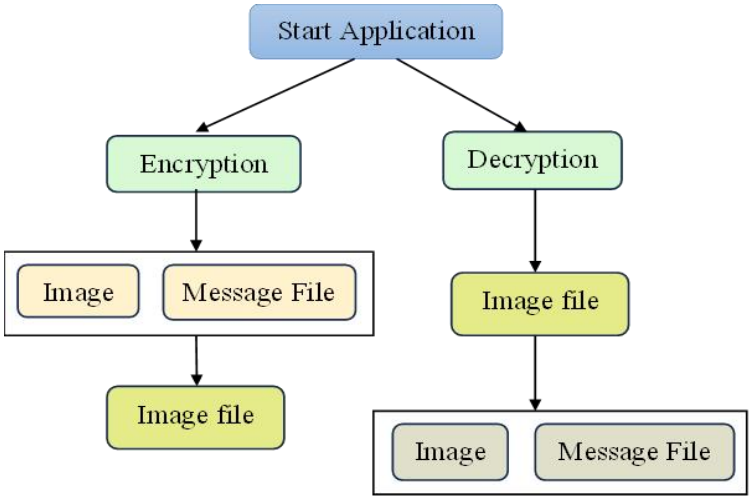

Fig. 3.1. Graphical Representation of Steganography System

4. The LSB Technique

Least significant bit (LSB) insertion, shown in figure 6, is a simple approach to embedding information in an image. For example, a simple plan proposed, is to place the embedding data at the least significant bit (LSB) of each pixel in an image [10],[14],[18]. The resulting image is called stego-image. Altering LSB does not change the quality of image to human perception but this plan is sensitive a variety of image processing attacks like cropping etc. We will be highlighting more on this technique for the various image formats.

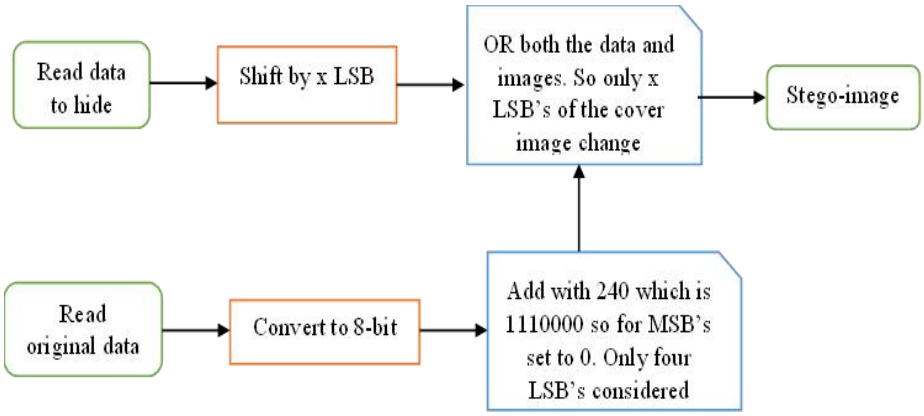

Fig. 4.1. LSB Technique of embedding data into an image

5. Research Methodology

Cryptography and steganography guarantees perfect secrecy by using the Least Significant Bit algorithm in python, which is the technique adopted for this work. This section describes a new proposed method [3]. The process of securing data transmission using image steganography generally involves the following steps:

5.1 Data Embedding

a. **Image Selection**: Choose a cover image (a standard image) where the data will be hidden. A cover image selected would serves as the medium in which the data will be hidden. The cover image should be of sufficient size and quality to hold the secret data without noticeable distortion.

b. **Data to be Hidden**: The sensitive data, which could be text, files, or any form of digital data, is prepared for embedding.

c. **Embedding Algorithm**: The data is embedded into the cover image using the selected algorithms. The method employed in this paper is **Least Significant Bit (LSB) substitution.** The least significant bit in each pixel's colour value is replaced with a bit of the hidden data. Since this bit has the smallest impact on the colour, the change is generally imperceptible to the human eye

d. **Data Encryption Procedure**: Before embedding, the data can be encrypted to add an extra layer of security, ensuring that even if the steganography is detected, the actual data remains protected.

5.2 Data Transmission:

The steganography image (stego-image) is transmitted over the network like any regular image file. Since it looks like a regular image, it is less likely to attract attention or be flagged by security systems.

5.3 Data Extraction:

On the receiving end, the hidden data is extracted from the stego-image using the reverse process of encoding. Only parties who know the steganographic method and any keys used can retrieve the original data. They follow this order:

i. **Image Reception**: The receiver obtains the stego-image through the network.

ii. **Extraction Algorithm**: The same algorithm used for embedding is applied in reverse to extract the hidden data from the image.

iii. **Decryption (if used)**: If the data was encrypted before embedding, it is decrypted to retrieve the original information.

6. Proposed Algorithms

This section presents a step-by-step solution to the problem described above. The encryption algorithm at the Sender's end and decryption algorithm at the Receiver's end are detailed below:

Encryption Algorithm (Sender's end):

Step 1: Select the text file where the original message was kept.

Step 2: Encrypt the content of the text file using the propose algorithm.

Step 3: Select an appropriate cover image (.jpeg format).

Step 4: Read the header and footer of the selected image in an array buffer.

Step 5: Add the encrypted data at the end of image footer.

Step 6: Sender and receiver are connected to the network.

Step 7: Sender provides the receiver's an IP address and then send the Stego-image if the IP address is correct.

Decryption Algorithm (Receiver's end):

Step 1: Receive the Stego-image.

Step 2: Extract the encrypted message from the end of the stego-image by reading the image footer.

Step 3: Generate the private key and decrypt the extracted message and then create a text file.

Step 4: Save the text file at the desired location

6.1 Python Implementation in LSB

Implementing secured data transmission using image steganography in Python typically involves two main steps: hiding (LSB Embedding Code) the data within an image and then extracting the hidden data (LSB Extraction Code) from the image. The tools and libraries are:

i. **Python**: Programming language for implementation.

ii. **PIL (Python Imaging Library)**: For image processing.

iii. **PyCrypto**: For encryption and decryption.

The Python Implementation used in the paper are:

```
from PIL import Image

import numpy as np

import binascii

import os
```

```
def to_bin(data):
    """Convert data to binary format as string"""
    if isinstance(data, str):
        return ''.join([format(ord(i), "08b") for i in data])
    elif isinstance(data, bytes) or isinstance(data, np.ndarray):
        return [format(i, "08b") for i in data]
    elif isinstance(data, int) or isinstance(data, np.uint8):
        return format(data, "08b")
    else;
        raise TypeError("Input type not supported")
def embed_data(image, data):
    """Embed binary data into image"""
    # Convert the image to a numpy array
    image_data = np.array(image)
    binary_data = to_bin(data)
    data_len = len(binary_data)
    data_index = 0
        for values in image_data:
        for pixel in values:
            r, g, b = to_bin(pixel[0]), to_bin(pixel[1]), to_bin(pixel[2])
            if data_index < data_len:
                pixel[0] = int(r[:-1] + binary_data[data_index], 2)
                data_index += 1
            if data_index < data_len:
                pixel[1] = int(g[:-1] + binary_data[data_index], 2)
                data_index += 1
            if data_index < data_len:
                pixel[2] = int(b[:-1] + binary_data[data_index], 2)
                data_index += 1
            if data_index >= data_len:
                break
        # Create new image with the modified data
    modified_image = Image.fromarray(image_data)
    return modified_image
def extract_data(image):
    """Extract binary data from image"""
    image_data = np.array(image)
    binary_data = ""
        for values in image_data:
```

```
        for pixel in values:
            r, g, b = to_bin(pixel[0]), to_bin(pixel[1]), to_bin(pixel[2])
            binary_data += r[-1] + g[-1] + b[-1]
        # Split by 8-bits to get each character
    all_bytes = [binary_data[i: i+8] for i in range(0, len(binary_data), 8)]
        # Convert from binary to ASCII
    decoded_data = ""
    for byte in all_bytes:
        decoded_data += chr(int(byte, 2))
        if decoded_data[-5:] == "#####":  # Delimiter to mark the end of the message
            break
        return decoded_data[:-5]  # Remove the delimiter before returning
def hide_message(input_image_path, output_image_path, message):
    # Load the image
    image = Image.open(input_image_path)
        # Add a delimiter to mark the end of the message
    message += "#####"
        # Embed the message in the image
    modified_image = embed_data(image, message)
        # Save the modified image
    modified_image.save(output_image_path)
    print("Message hidden and image saved.")
def retrieve_message(image_path):
    # Load the image
    image = Image.open(image_path)
        # Extract the hidden message
    hidden_message = extract_data(image)
        return hidden_message
hide_message('input_image.png', 'output_image.png', 'This is a secret message.')
message = retrieve_message('output_image.png')
print("Hidden message:", message)
```

This Python script uses image steganography to hide and retrieve messages in an image. The LSB method is used for embedding data into the image. The message is hidden by altering the least significant bits of the pixel values, ensuring that the image remains visually unchanged. The script includes functions for both hiding a message in an image and retrieving the message from the image.

7. Results and Discussion

The main menu interface window is shown in figure 7.1. This is the first page that will appear immediately the application is launched.

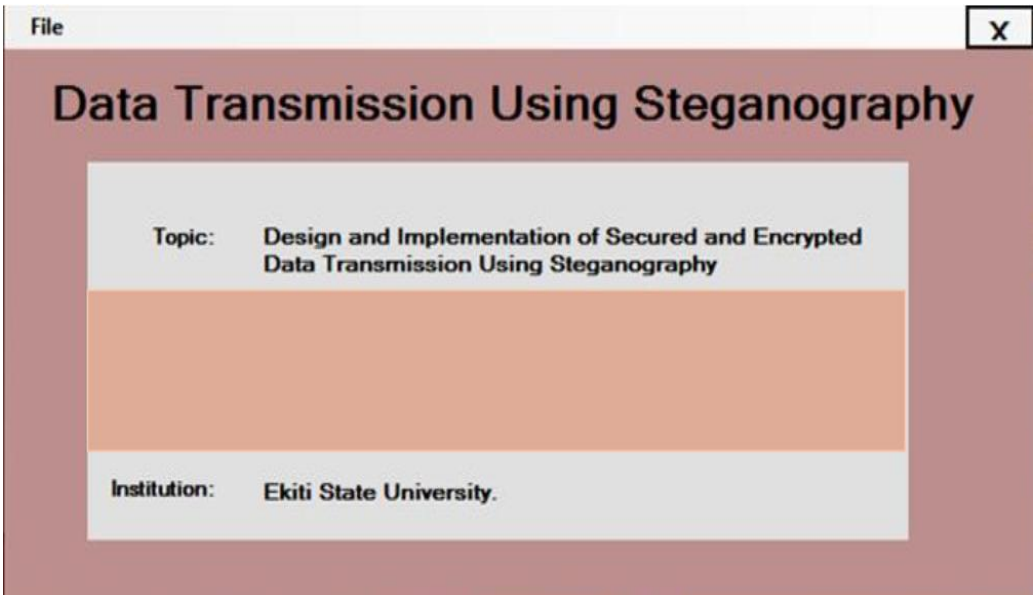

Fig. 7.1. Main Menu Interface

It consists of the following functional and non-functional modules which are

1. Functional modules:

File

Steganography

Exit

2. Non-Functional modules:

Application title

Author's details

3. Main menu form

The image upload interface is shown in figure 7.2. It shows the image upload process and the image data types when the browse button is pressed

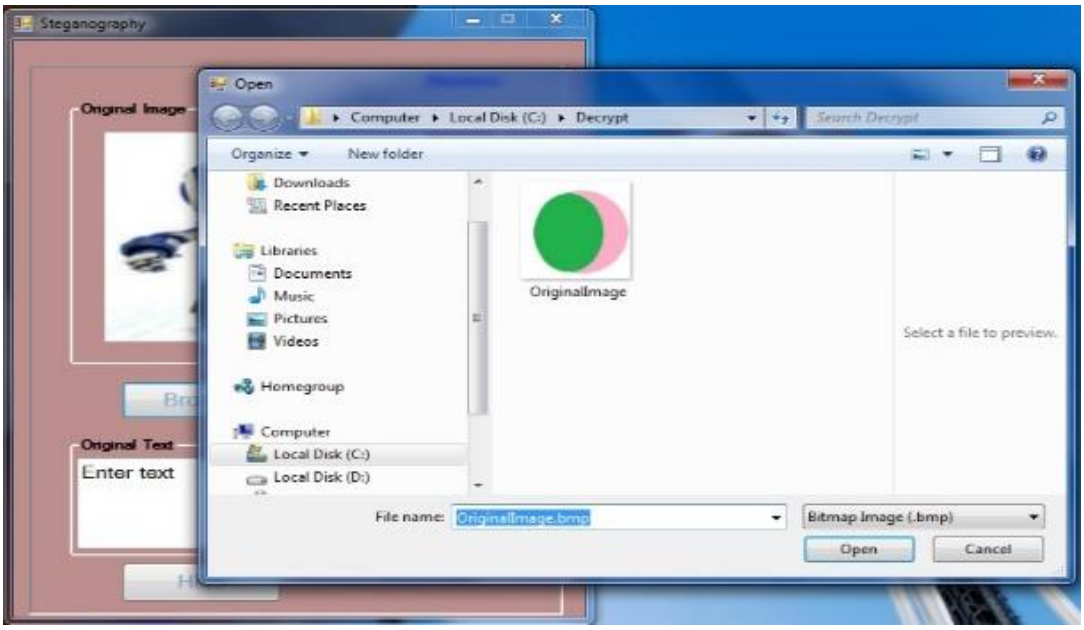

Fig. 7.2. Image Upload Interface

The encryption interface is shown in figure 7.3. This is where the text is encrypted into the selected image, it consists of the following modules:

i. Main menu

ii. Original image box

iii. Browse button

iv. Original text box

v. Hide button

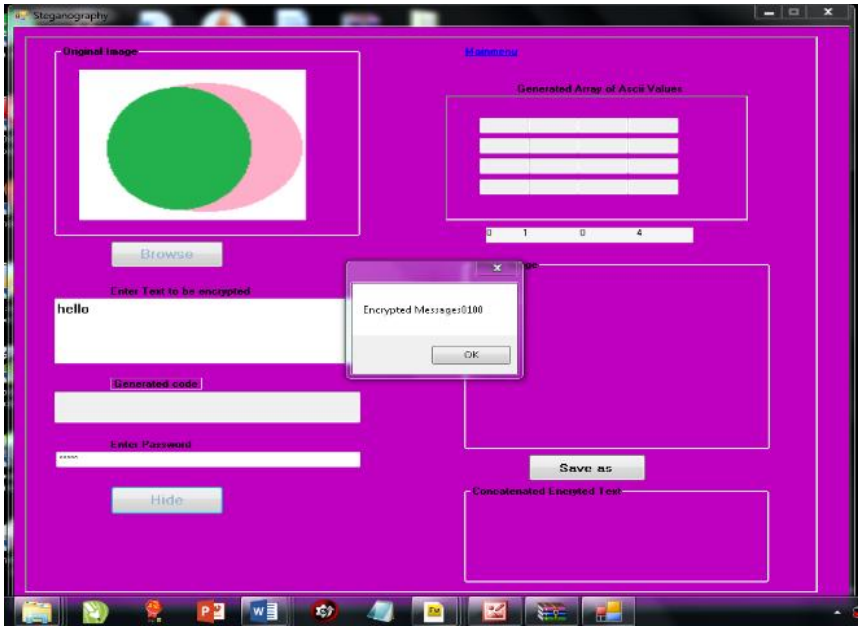

Fig. 7.3. Encryption Interface

The uploaded image and text interface is shown in figure 7.4. It shows the image containing the encrypted text, and where it will be saved when the save as button is clicked.

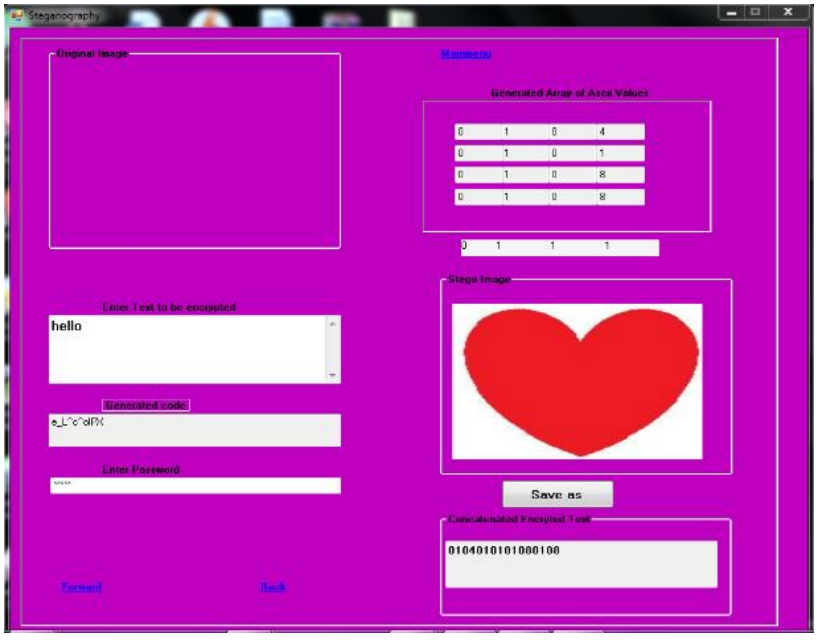

Fig. 7.4. Uploaded Image and Text Interface

The decryption interface is shown in figure 7.5. It shows the extracted message after decrypting

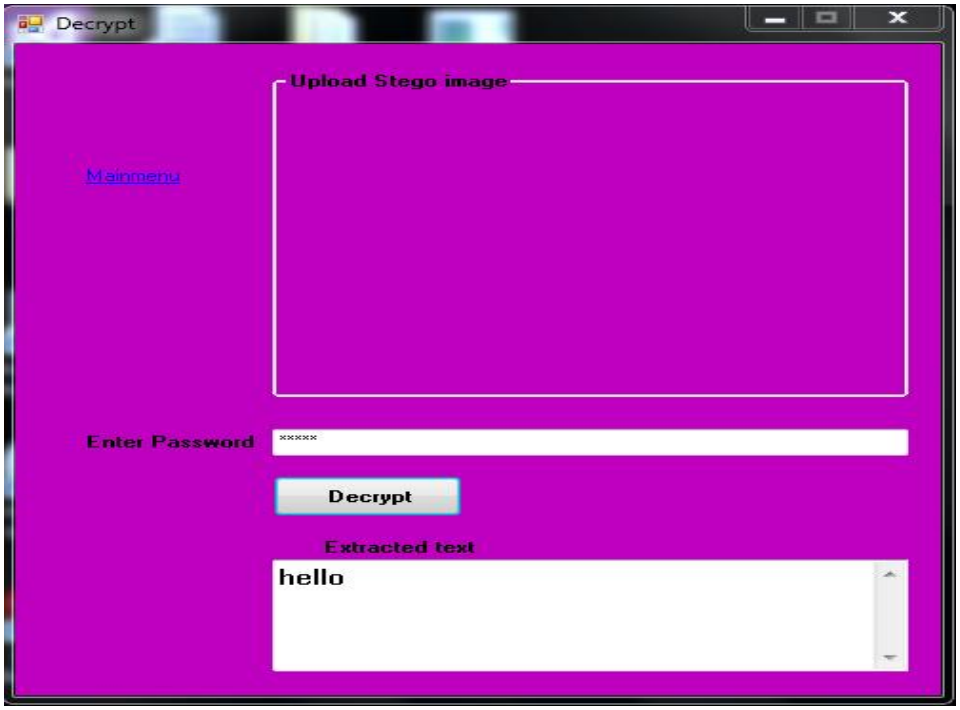

Fig. 7.5 Decryption Interface

The implemented system was tested with various images and data payloads. The following observations were made:

i. **Image Quality**: The LSB embedding process did not noticeably degrade the quality of the cover image.

ii. **Data Integrity**: The extracted data was found to be intact and identical to the original secret data.

iii. **Security**: The encryption ensured that the hidden data was not decipherable without the correct key

7. Conclusion

Steganography is an effective technique for hiding information from plain sight. This was introduced to hide data in an image format to distract the attention of intruder from accessing it on the internet or computer system. This paper demonstrates a method for secure data transmission using image steganography. By combining LSB embedding with encryption, the system ensures data concealment, security, and integrity. Image steganography provides a powerful tool to secure data transmission, offering a way to conceal data within ordinary images. When used correctly, it enhances security by making the presence of the data inconspicuous. This research enhances data protection by improving on the security capacity of the existing technologies. Future work could explore more advanced steganographic techniques and real-time applications.

References

[1] Cheddad, A., Condell, J., Curran, K., & Mc Kevitt, P. (2010). Digital image steganography: Survey and analysis of current methods. *Signal Processing*, 90(3), 727-752. https://doi.org/10.1016/j.sigpro.2009.08.010

[2] Prema, A., & Gowri, V. (2019). Secure Data Transmission Using Image Steganography. *International Journal of Advanced Networking and Applications*, 300-303.

[3] Nunna, K. C., & Marapareddy, R. (2020, March). Secure data transfer through internet using cryptography and image steganography. In *2020 SoutheastCon* (Vol. 2, pp. 1-5). IEEE.

[4] Osuolale, A. F. (2017). Secure data transfer over the internet using image cryptosteganography. *Int. J. Sci. Eng. Res*, *8*(12), 1115-1121.

[5] Abdullah, D.M., Ameen, S.Y., Omar, N., Salih, A.A., Ahmed, D.M., Kak, S.F., Yasin, H.M., Ibrahim, I.M., Ahmed, A.M. and Rashid, Z.N., 2021. Secure data transfer over internet using image steganography. *Asian Journal of Research in Computer Science*, *10*(3), pp.33-52.

[6] Ker, A. D. (2005). Steganalysis of LSB matching in grayscale images. *IEEE Signal Processing Letters*, 12(6), 441-444.

[7] Subramanian, N., Elharrouss, O., Al-Maadeed, S., & Bouridane, A. (2021). Image steganography: A review of the recent advances. *IEEE access*, *9*, 23409-23423.

[8] Fridrich, J., Goljan, M., & Du, R. (2001). Reliable detection of LSB steganography in color and grayscale images. *Proceedings of the 2001 Workshop on Multimedia and Security: New Challenges*, 27-30.

[9] Chan, C. K., & Cheng, L. M. (2004). Hiding data in images by simple LSB substitution. *Pattern Recognition*, 37(3), 469-474.

[10] Kumar, A., & Kumar, R. (2010). Enhanced payload capacity for LSB steganography using multiple least significant bits. *International Journal of Computer Applications*, 3(1), 1-6.

[11] Bandyopadhyay, S. K., Roy, D., & Patel, K. K. (2006). Image steganography using block-DCT with multiple JPEG images. *International Journal of Information Technology*, 2(1), 1-9.

[12] Lin, S. D., Chang, C. C., & Huang, T. Y. (2008). A wavelet-tree-based watermarking method using the SPIHT algorithm. *International Journal of Imaging Systems and Technology*, 18(1), 53-58.

[13] Fridrich, J., Goljan, M., & Soukal, D. (2003). Perturbed quantization steganography with wet paper codes. *Proceedings of the 2004 Workshop on Multimedia and Security*, 4-15.

[14] Zhang, T., & Wang, X. (2020). A steganography algorithm based on deep convolutional generative adversarial networks. *IEEE Access*, 8, 83547-83556. https://doi.org/10.1109/ACCESS.2020.2990495

[15] Li, Z., Qin, C., Yu, Z., & Liu, P. (2018). Deep learning-based robust image steganography using intermediate supervision. *Neurocomputing*, 354, 146-155. https://doi.org/10.1016/j.neucom.2019.03.048

[16] Subhedar, M. S., & Mankar, V. H. (2014). Current status and key issues in image steganography: A survey. *Computer Science Review*, 13-14, 95-113. https://doi.org/10.1016/j.cosrev.2014.09.001

[17] Chandra, S., & Paira, S. (2019). Secure transmission of data using image steganography. *ICTACT Journal on Image and Video Processing, August 2019*, *10*(01).

[18] Audhi, S., & Mascarenhas, M. (2019). Secure mechanism for communication using image steganography. In *2019 2nd international conference on intelligent computing, instrumentation and control technologies (ICICICT)* (Vol. 1, pp. 729-732). IEEE.