

Secure Image Transmission using Steganography in Network Security

Project Report Submitted In Partial Fulfillment Of
The Requirements for The Degree Of

Bachelor of Technology

In

Electronics and Communication Engineering

Of

Maulana Abul Kalam Azad University of Technology, West Bengal

By

ABHISEK BOSE, 10900321060

BHAVIKA BHATTACHARJEE, 10900320041

SUKRITI GUIN, 10900320033

SUYETA CHOWDHURY, 10900320017

Under the guidance of

ATREYEE MAJUMDAR ROY



**Department of Electronics and Communication Engineering
Netaji Subhash Engineering College
Garia, Kolkata – 700152**

2020-2024

CERTIFICATE

This is to certify that this project report titled **Secure Image Transmission using Steganography in Network Security**

submitted in partial fulfillment of requirements for award of the degree Bachelor of Technology (B. Tech) in Electronics and Communication Engineering of West Bengal University of Technology is a faithful record of the original work carried out by,

Abhisek Bose, **Roll no.** 10900321060, **Regd. No.** 211090100320003 OF 2021-22

Bhavika Bhattacharjee , **Roll no.** 10900320041, **Regd. No.** 201090100310055 OF 2020-2021

Sukriti Guin, **Roll no.** 10900320033, **Regd. No.** 201090100310063 OF 2020-21

Suyeta Chowdhury, **Roll no:**10900320017, **Regd. No:** 201090100310079 OF 2020-21

under my guidance and supervision.

It is further certified that it contains no material, which to a substantial extent has been submitted for the award of any degree/diploma in any institute or has been published in any form, except the assistances drawn from other sources, for which due acknowledgment has been made.

Date:.....

Signature of the Supervisor

Head of the Department

Electronics and Communication Engineering
Netaji Subhash Engineering College
Techno City, Garia, Kolkata – 700 152

Declaration

We hereby declare that this project report titled

Secure Image Transmission using Steganography in Network Security

is our own original work carried out as a undergraduate student in Netaji Subhash Engineering College except to the extent that assistances from other sources are duly acknowledged.

All sources used for this project report have been fully and properly cited. It contains no material which to a substantial extent has been submitted for the award of any degree/diploma in any institute or has been published in any form, except where due acknowledgement is made.

Student's names:

Signatures:

Date:

Abhisek Bose

.....

.....

Bhavika Bhattacharjee

.....

.....

Suyeta Chowdhury

.....

.....

Sukriti Guin

.....

.....

Certificate of Approval

We hereby approve this dissertation titled

Secure Image Transmission using Steganography in Network security

carried out by

ABHISEK BOSE, **Roll no.** 10900321060, **Regd. No.** 211090100320003 OF 2021-22

BHAVIKA BHATTACHARIYA, **Roll no.** 10900320041, **Regd. No.** 201090100310055 OF
2020-2021

SUYETA CHOWDHURY, **Roll no.** 10900320017, **Regd. No.** 201090100310079 OF
2020-21

SUKRITI GUIN, **Roll no.** 10900320033, **Regd. No.** 201090100310063 OF 2020-21

under the guidance of

ATREYEE MAJUMDAR ROY

of Netaji Subhash Engineering College, Kolkata in partial fulfillment of requirements
for award of the degree Bachelor of Technology (B. Tech) in Electronics and
Communication Engineering of West Bengal University of Technology

Date:.....

Examiners' signatures:

1.
2.
3.

Acknowledgement and/or Dedication

We extend our sincere gratitude and appreciation to everyone who played a crucial role in the successful completion of the "Secure Image Transmission using Steganography in Network Security" group project. First and foremost, we would like to express our heartfelt thanks to our project guide for their invaluable guidance, unwavering support, and insightful feedback throughout the development process. Our appreciation also goes to our fellow group members for their collaborative spirit, constructive discussions, and shared enthusiasm, which significantly enriched the project. Special thanks to the open-source community for providing the essential tools and libraries for the project's implementation. Finally, we acknowledge the understanding and support of our families and friends, whose encouragement and patience were indispensable during this collective endeavor. This project has been a collaborative effort, and we are grateful for the contributions and encouragement that made it possible.

Abhisek Bose
Bhavika Bhattacharjee
Sukriti Guin
Suyeta Chowdhury

Dated:

Abstract

The Secure LSB Steganography with AES Encryption project represents a cutting-edge information security solution, ingeniously merging the principles of LSB steganography and AES encryption. This sophisticated system consists of meticulously designed modules, each playing a crucial role in fortifying the security of digital communication. The LSB steganography module is adept at discreetly embedding information within the pixels of images, leveraging the least significant bits to maintain a low visual impact. Complementing this, the AES encryption module ensures the confidentiality and integrity of the concealed data, adding an extra layer of protection. A shuffling algorithm introduces a dynamic element, orchestrating a randomized sequence for pixel access during both embedding and extraction processes, contributing to heightened unpredictability. The key management system diligently oversees the generation, storage, and handling of encryption keys, safeguarding against potential vulnerabilities. The amalgamation of these components within a user-friendly interface not only streamlines the process for users but also fortifies confidential communication. It establishes robust data integrity, shields against unauthorized access, accommodates scalability by supporting various encryption algorithms, and safeguards personal privacy. This abstraction underscores the project's nuanced and comprehensive approach to advancing digital communication security, addressing the evolving requirements of users in search of sophisticated, secure, and user-friendly information hiding solutions.

Contents

1: INTRODUCTION	Page No
1.1 Problem Definition	1
1.2 Aim of The Project	1
1.3 Project Overview/Specification	2
1.4 Software Specifications	2
2: REVIEW WORK	
2.1 Existing System	3
2.2 Proposed System	3
3: WORKING METHODOLOGY	
3.1 Flowcharts & Circuit Diagram	4
3.2 Input	4
3.3 Advanced Encryption System (AES)	4
3.4 LSB Steganography	8
3.5 Work done	10
4: RESULTS ANALYSIS	
4.1 Visual Impact	12
4.2 Security	12
4.3 Efficiency	12
4.4 Conclusion	13
5: REFERENCES	14

Chapter 1

Introduction

1.1 Problem Definition

In the realm of digital communication and information exchange, ensuring the confidentiality and security of sensitive data is of paramount importance. However, traditional methods often fall short in providing a robust solution that combines both concealment and encryption. The challenge lies in developing an advanced system that seamlessly integrates LSB steganography and AES encryption to create a secure environment for digital communication. The need for a reliable method to hide information within image pixels while employing strong encryption measures becomes evident. Additionally, there is a requirement for a shuffling algorithm to introduce unpredictability in pixel access, preventing adversaries from deciphering the concealed information. Furthermore, efficient key management is crucial to maintaining the integrity of the encryption process. The aim is to address these challenges comprehensively, providing users with a user-friendly and secure platform for confidential digital communication.

1.2 Aim of The Project

The primary aim of this project is to design, implement, and validate an advanced information security solution that seamlessly integrates LSB steganography and AES encryption. The goal is to create a robust and user-friendly platform for digital communication, ensuring the secure exchange of sensitive information. The project aims to address the challenges associated with traditional methods by combining the concealment capabilities of LSB steganography with the robust encryption provided by AES. The integration of a shuffling algorithm adds an element of unpredictability, enhancing the security of the concealed information. Efficient key management is also a focal point, aiming to provide a secure foundation for encryption processes. Ultimately, the project aims to deliver a sophisticated and comprehensive solution that caters to the evolving needs of users seeking both privacy and ease of use in their digital communication endeavors.

1.3 Project Overview/Specifications

The Secure LSB Steganography with AES Encryption project is a comprehensive information security solution designed to enhance the confidentiality and integrity of digital communication. The system encompasses a sophisticated integration of LSB steganography and AES encryption, allowing users to discreetly embed sensitive information within image pixels while ensuring robust encryption measures. A shuffling algorithm introduces an element of unpredictability, fortifying the security of the concealed data. The project includes an efficient key management system, meticulously handling the generation, storage, and utilization of encryption keys. The user interface is thoughtfully designed to be user-friendly, offering a seamless and secure platform for concealing and exchanging sensitive information. This project aims to address the shortcomings of traditional methods, providing users with a versatile, secure, and user-friendly solution for confidential digital communication.

1.4 Software Specifications

The Secure LSB Steganography with AES Encryption project adheres to stringent software specifications, ensuring a robust and reliable information security solution. The software is developed using a combination of Python and relevant libraries such as PIL (Python Imaging Library) for image processing and cryptography libraries for implementing AES encryption. The Python programming language facilitates the integration of advanced algorithms, allowing for efficient LSB steganography and the seamless incorporation of the AES encryption standard. The project employs a modular and object-oriented software design, enhancing maintainability and scalability. Additionally, the user interface is created using appropriate GUI frameworks to provide an intuitive and user-friendly experience. Thorough testing methodologies, including unit testing and system testing, are implemented to ensure the correctness and security of the software. The software specifications emphasize adherence to industry standards, security best practices, and a commitment to delivering a robust and user-friendly solution for secure digital communication.

Chapter 2

Review Work

Existing System

In the existing system, digital communication security was already facing challenges related to the combined concealment and encryption of sensitive information. Traditional methods often lack a cohesive integration of techniques, making it difficult to ensure both robust concealment and encryption simultaneously. LSB steganography may provide a means of hiding data within image pixels, but it falls short in delivering comprehensive encryption measures. Additionally, the absence of a shuffling algorithm makes pixel access predictable, potentially compromising the security of hidden data. Key management systems are often rudimentary, lacking the sophistication required for secure encryption processes. Overall, the existing system highlights the need for an advanced solution that seamlessly merges LSB steganography and AES encryption, addresses predictability issues with a shuffling algorithm, and employs efficient key management for a holistic approach to secure digital communication.

Proposed System

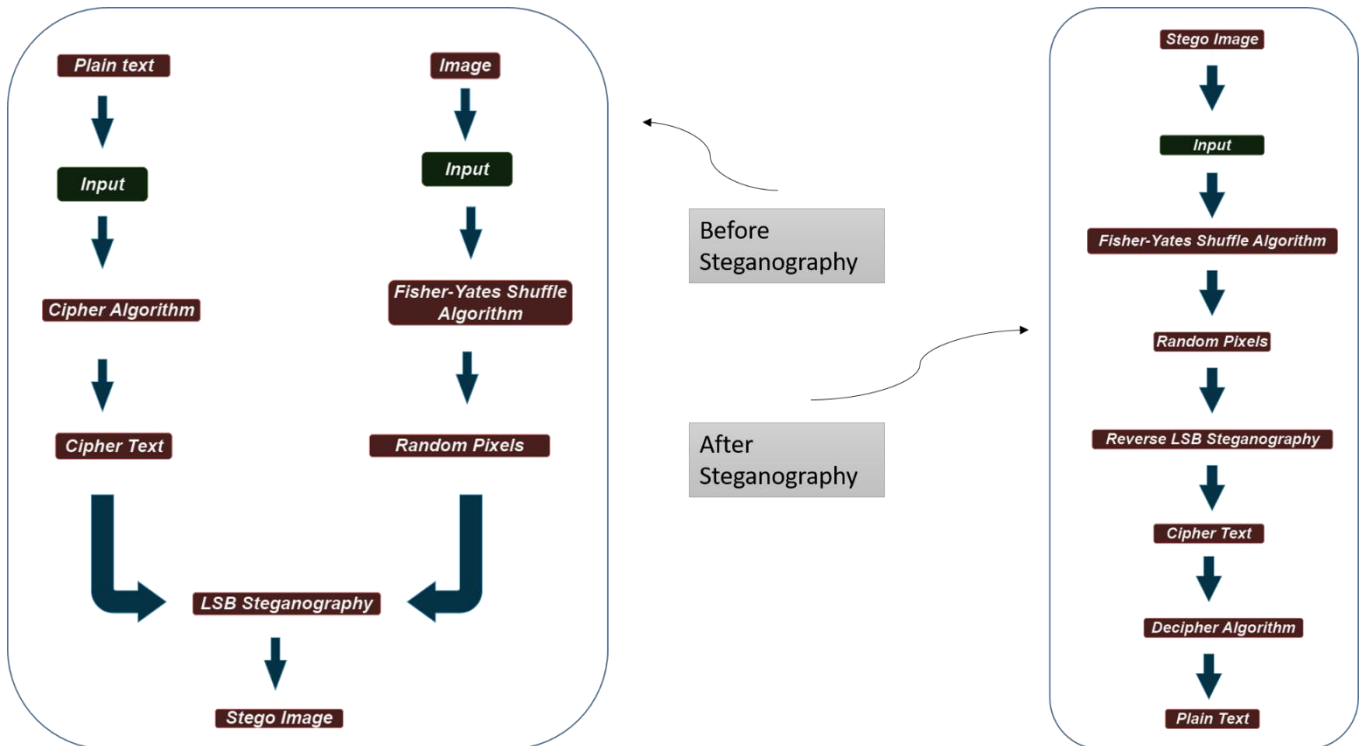
The proposed system is where we have introduced the concept of LSB Steganography along with Seed Generator and Pixel Shuffler. This provides a groundbreaking approach to secure digital communication in the form of data by integrating numerous algorithms and various concepts. This system aims to address the limitations of conventional methods and compare them, offering a platform that allows users to conceal sensitive information within image pixels with the added layer of robust encryption. The proposed system incorporates shuffling algorithm to introduce data unpredictability to any viewer, fortifying the security of hidden data. Additionally, an efficient key management system is integrated to ensure secure encryption processes. The user interface is designed for optimal user experience, emphasizing simplicity and security. With an innovative blend of steganographic techniques and encryption standards, the proposed system is poised to redefine the landscape of secure communication, providing users with a versatile and user-friendly solution for protecting their confidential information.

Chapter 3

Working Methodology

3.1 Flow Charts

3.1.1 Flowchart:



3.2 Input

The project accepts two main inputs: plain text and an image. The plain text is converted into cipher text using the Advanced Encryption Standard (AES) algorithm, ensuring the security of the embedded information.

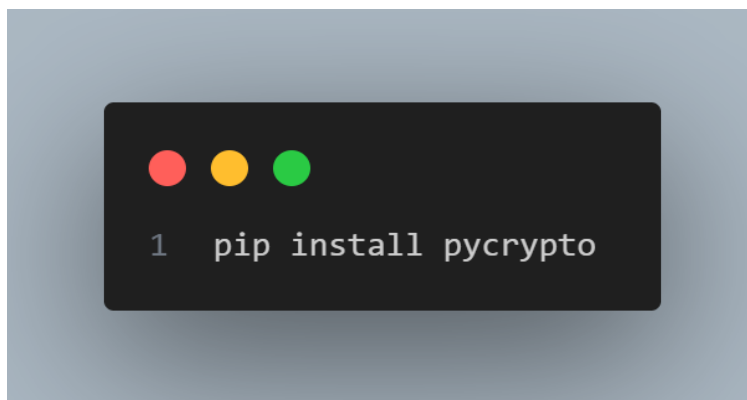
3.3 Advance Encryption Standard (AES)

It is a Symmetric Key Block Cipher that broadly takes on Symmetric Key encryption nowadays. It Comes for the alternative of Des as the size of Des was too small. For the Encryption and Decryption Purposes, it is applicable for the Block size of 128 Bits of size. The number of loops decides the Key Size. It uses 10 loops for 128-bit of keys, 12 loops for 192-bit of keys and 14 loops

for 256-bit of keys. In general, 4 functions used to encrypt the data and gives ciphertext as output and it includes Sub Bytes, Shifts rows, Adds keys and Mix Columns. The Components of Aes are cost price of memory is very less, high-speed algorithm procedure.

3.3.1 Implementing AES in Python

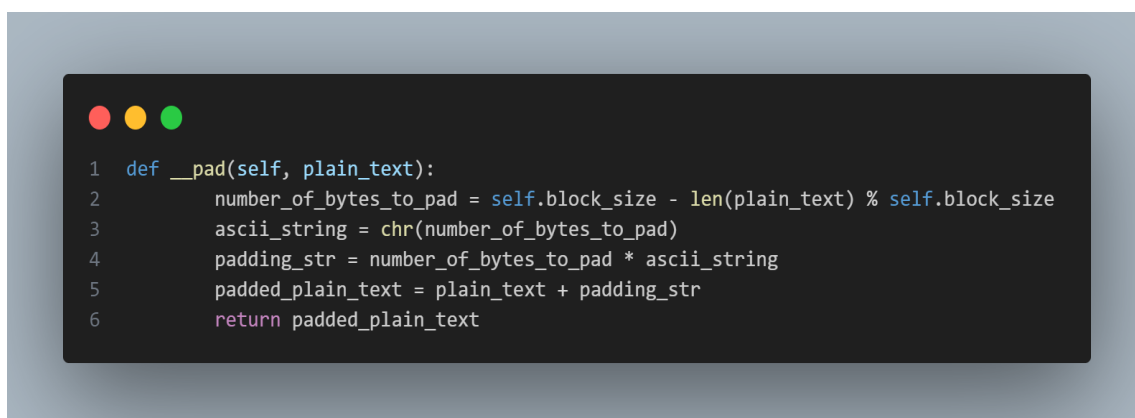
Fortunately, we don't have to implement AES from scratch, but you can give it a try if you're feeling spicy. In order to avoid doing so, we first need to install the **pycrypto** library, which can be done via pip with the following command:

A terminal window with a dark background and three colored window control buttons (red, yellow, green) at the top left. The command '1 pip install pycrypto' is entered in a monospaced font.

```
1 pip install pycrypto
```

Now we are going to create a class for our AES cipher with the following constructor:

Before we proceed to define the `encrypt` and `decrypt` methods for our `AESCipher` class, lets first create the `__pad` and `__unpad` methods.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) at the top left. The implementation of the `__pad` method is shown in a monospaced font.

```
1 def __pad(self, plain_text):  
2     number_of_bytes_to_pad = self.block_size - len(plain_text) % self.block_size  
3     ascii_string = chr(number_of_bytes_to_pad)  
4     padding_str = number_of_bytes_to_pad * ascii_string  
5     padded_plain_text = plain_text + padding_str  
6     return padded_plain_text
```

The `__pad` method receives the `plain_text` to be encrypted and adds a number bytes for the

text to be a multiple of 128 bits. This number is stored in `number_of_bytes_to_pad`. Then in `ascii_string` we generate our padding character, and `padding_str` will contain that character times `number_of_bytes_to_pad`. So we only have to add `padding_str` at the end of our `plain_text` so that it is now a multiple of 128 bits.

```
1 def __unpad(plain_text):
2     last_character = plain_text[len(plain_text) - 1:]
3     bytes_to_remove = ord(last_character)
4     return plain_text[:-bytes_to_remove]
```

In an opposite manner, `__unpad` method will receive the decrypted text, also known as `plain_text` and will remove all the extra added characters in the `__pad` method. For that we first must identify the last character and store in `bytes_to_remove` how many bytes we need to trim of the end of `plain_text` in order to unpad it.

Encrypt

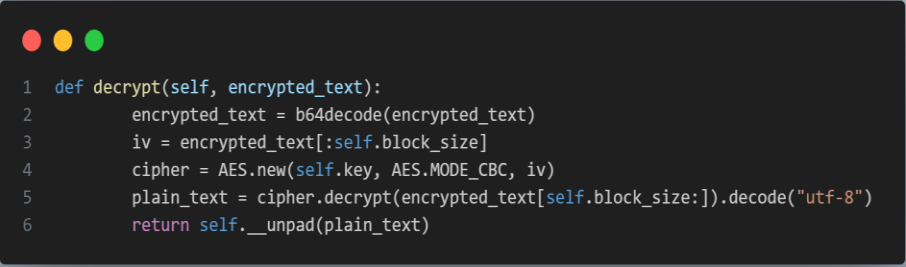
With these two methods out of the way, let's implement the `encrypt` method

```
1 def encrypt(self, plain_text):
2     plain_text = self.__pad(plain_text)
3     iv = Random.new().read(self.block_size)
4     cipher = AES.new(self.key, AES.MODE_CBC, iv)
5     encrypted_text = cipher.encrypt(plain_text.encode())
6     return b64encode(iv + encrypted_text).decode("utf-8")
7
```

The `encrypt` method receives the `plain_text` to be encrypted. First we pad that `plain_text` in order to be able to encrypt it. After we generate a new random `iv` with the size of an AES block, 128 bits. We now create our AES cipher with `AES.new` with our `key`, in mode `CBC` and with our just generated `iv`. We now invoke the `encrypt` function of our cipher, passing it our `plain_text` converted to bits. The encrypted output is then placed after our `iv` and converted back from bits to readable characters.

Decrypt

Now that we can encrypt text, but I'm sure we would like to be able to decrypt it:



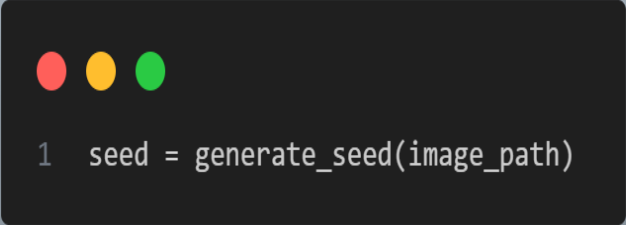
```
1 def decrypt(self, encrypted_text):
2     encrypted_text = b64decode(encrypted_text)
3     iv = encrypted_text[:self.block_size]
4     cipher = AES.new(self.key, AES.MODE_CBC, iv)
5     plain_text = cipher.decrypt(encrypted_text[self.block_size:]).decode("utf-8")
6     return self.__unpad(plain_text)
```

In order to decrypt, we must backtrack all the steps done in the `encrypt` method. First we convert our `encrypted_text` to bits and extract the `iv`, which will be the first 128 bits of our `encrypted_text`. Much like before, we now create a new AES cipher with our key, in mode CBC and with the extracted `iv`. We now invoke the `decrypt` method of our cipher and convert it to text from bits. We remove the padding with `__unpad` and that's it!

3.4 LSB Steganography

3.4.1 Seed Generation

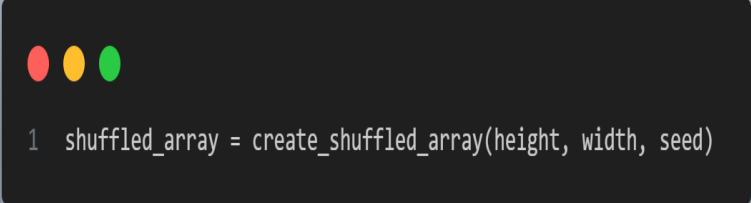
The LSB steganography begins with the generation of a seed for pixel shuffling. The seed is derived from the RGB values of the first pixel in the image. The 3 LSBs from the Red channel, 3 LSBs from the Green channel, and 2 LSBs from the Blue channel are combined to form an 8-bit seed.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains a single line of code.

```
1 seed = generate_seed(image_path)
```

3.4.2 Shuffling Pixels

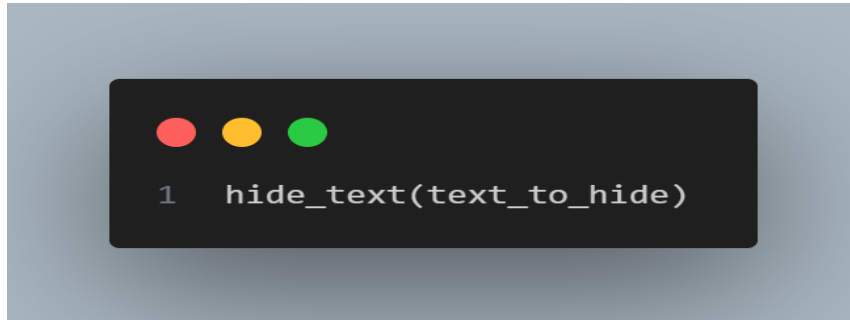
A shuffled array is created based on the calculated seed using the Fisher-Yates shuffle algorithm, excluding the 0th element to avoid modification of the seed.

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains a single line of code.

```
1 shuffled_array = create_shuffled_array(height, width, seed)
```

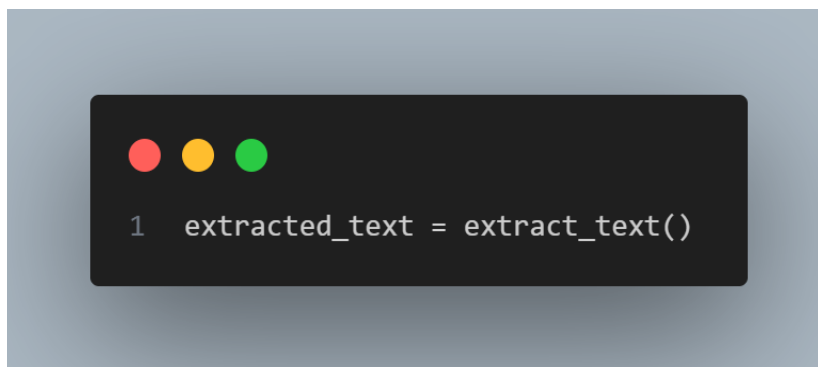
3.4.3 Embedding Text

The plain text is embedded into the image by iterating through the shuffled array. Each pixel is selected based on the shuffled indices, and the LSBs of its RGB channels are replaced with the bits of the cipher text.



3.4.4 Text Extraction

The reverse process involves extracting the text from the stego-image. The shuffled array is used to predict the pixel locations where the text is embedded. The LSBs from the RGB channels are extracted and combined to reconstruct the hidden text.



3.5 Work done:

The LSB (Least Significant Bit) Steganography project is designed to conceal textual information within an image by exploiting the least significant bits of the image's pixel values. This technique ensures that the alterations made to the image are imperceptible to the human eye, allowing for the covert transmission of data.

1. Initialization and Importing Libraries:

The project begins by importing the necessary libraries, including the Python Imaging Library (PIL) for image processing.

A class named LSBSteganography is defined to encapsulate the functionalities of the LSB Steganography.

2. Class Initialization:

An instance of the LSBSteganography class is created by providing the path to an image file. The image is then opened using PIL, and its dimensions are stored for further processing.

3. Generating Seed:

A seed for shuffling pixels is generated based on the least significant bits of the first pixel in the image. This seed is crucial for determining the order in which pixels will be manipulated.

4. Putting Integer in Pixels:

A method is implemented to hide integer values in the RGB components of pixels using the LSB technique. This involves modifying the least significant bits of each color channel without significantly altering the visual appearance of the image.

5. Creating Shuffled Array:

The project employs a shuffled array of pixel indices, generated using the previously obtained seed. This array determines the order in which pixels will be modified during the process of hiding and extracting text.

6. Predicting Row and Column:

The script includes a method for predicting the row and column indices of a pixel based on a shuffled element. This is essential for accurately identifying the location of pixels in the image.

7. Hiding Text in the Image:

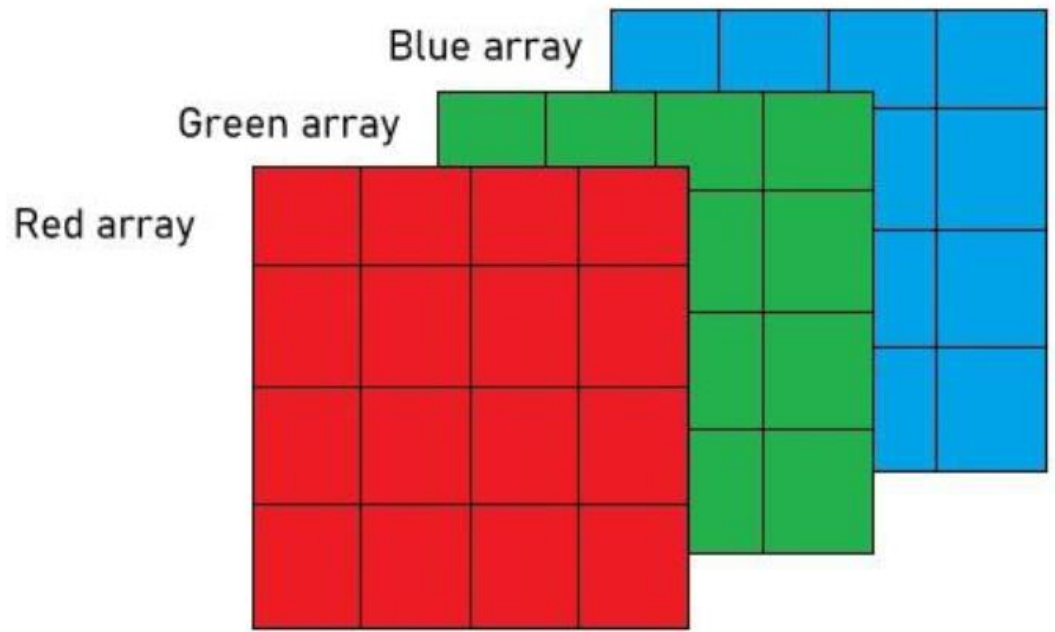
The core functionality of the project involves hiding a given text message within the image. Text is hidden by modifying the least significant bits of pixels in a shuffled order. The end of the hidden text is marked using the backspace character (ASCII 8).

8. Saving the Stego Image:

After hiding the text, the stego image is saved to a specified output path. This enables users to store and share the image containing the concealed information.

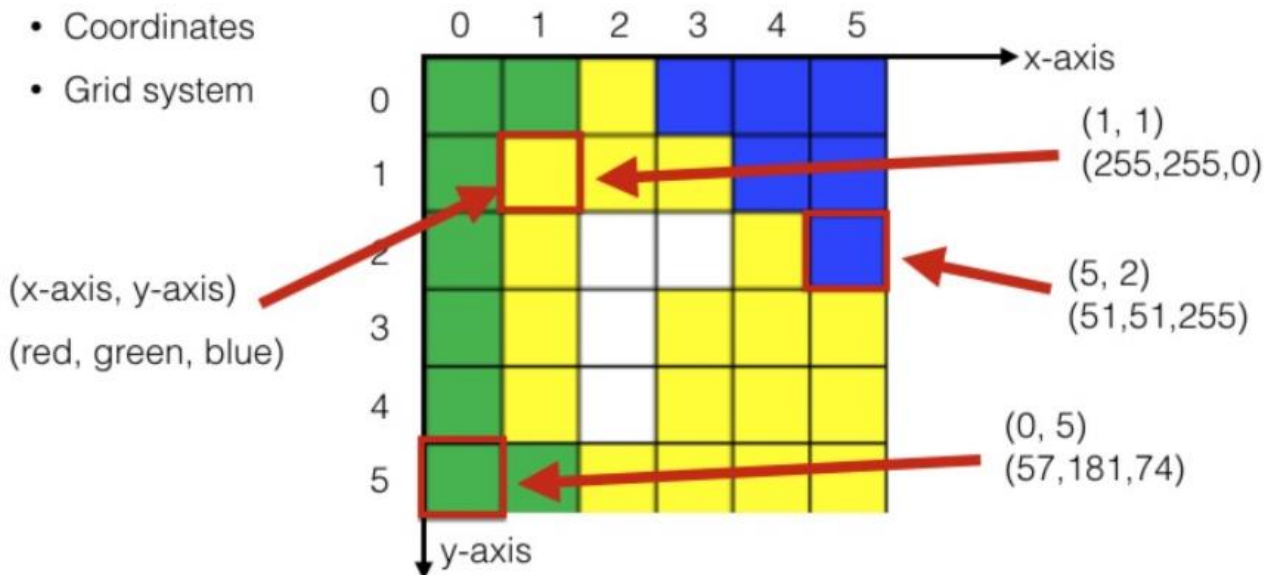
9. Extracting Hidden Text:

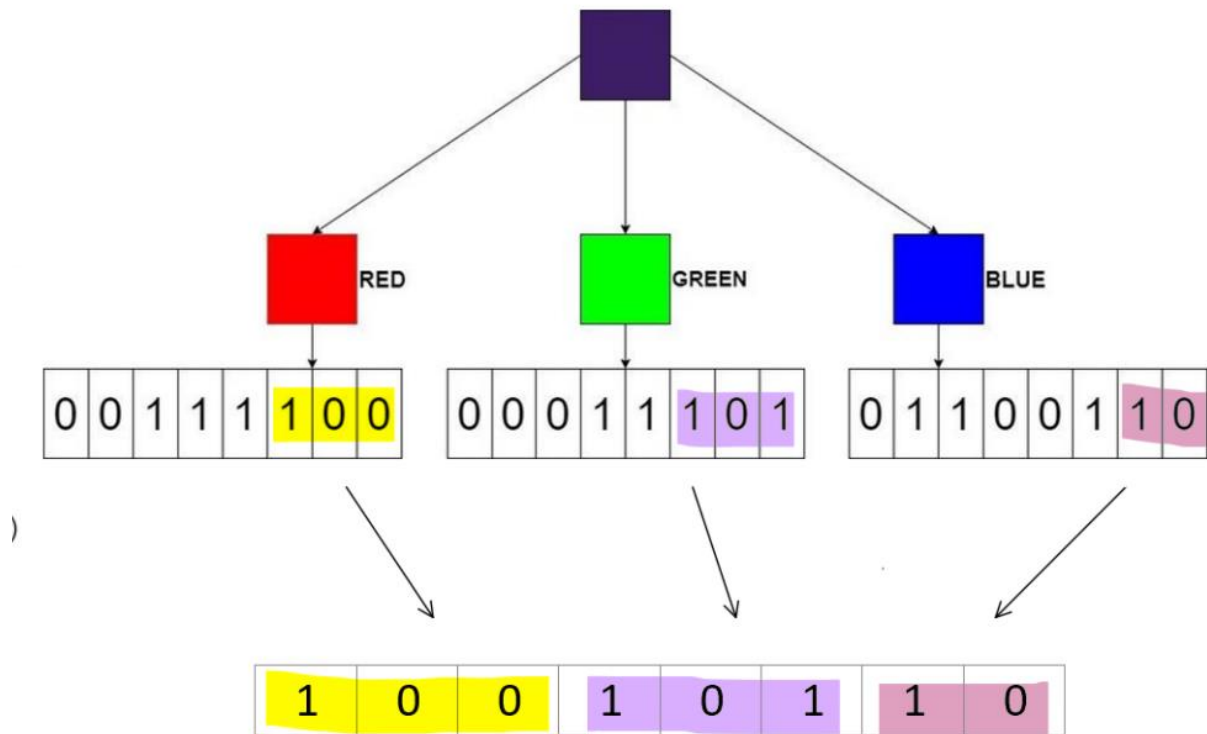
The project includes a method for extracting hidden text from the stego image. This process involves reversing the steps used for hiding text, allowing the recovery of the original information.



Arrays stacked over each other to form a Digital Image.

Understanding the pixel grid





Seed = 150

Seed Generation

		Column Index			
		0	1	2	3
Row Index	0	8	6	5	4
	1	2	1	9	7
	2	3	6	4	2

Two-Dimensional Array

Row-Major (Row Wise Arrangement)

8	6	5	4	2	1	9	7	3	6	4	2
Row 0				Row 1				Row 2			

Column-Major (Column Wise Arrangement)

8	2	3	6	1	6	5	9	4	4	7	2
Column 0			Column 1			Column 2			Column 3		

2D matrix to 1D matrix

1	2	3	4	5
1	5	3	4	2
1	5	4	3	2
4	5	1	3	2
5	4	1	3	2

Shuffling Algorithm

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12

1	2	3	4	5	6	7	8	9	10	11	12
---	---	---	---	---	---	---	---	---	----	----	----

Element = 5
 Row = Element // No. of cols = $5 // 4 = 1$
 Col = (Element % No. of cols) - 1 = $5 \% 4 - 1 = 0$

Element = 11
 Row = Element // No. of cols = $11 // 4 = 2$
 Col = (Element % No. of cols) - 1 = $11 \% 4 - 1 = 2$

When Element is not the multiple of No. of cols

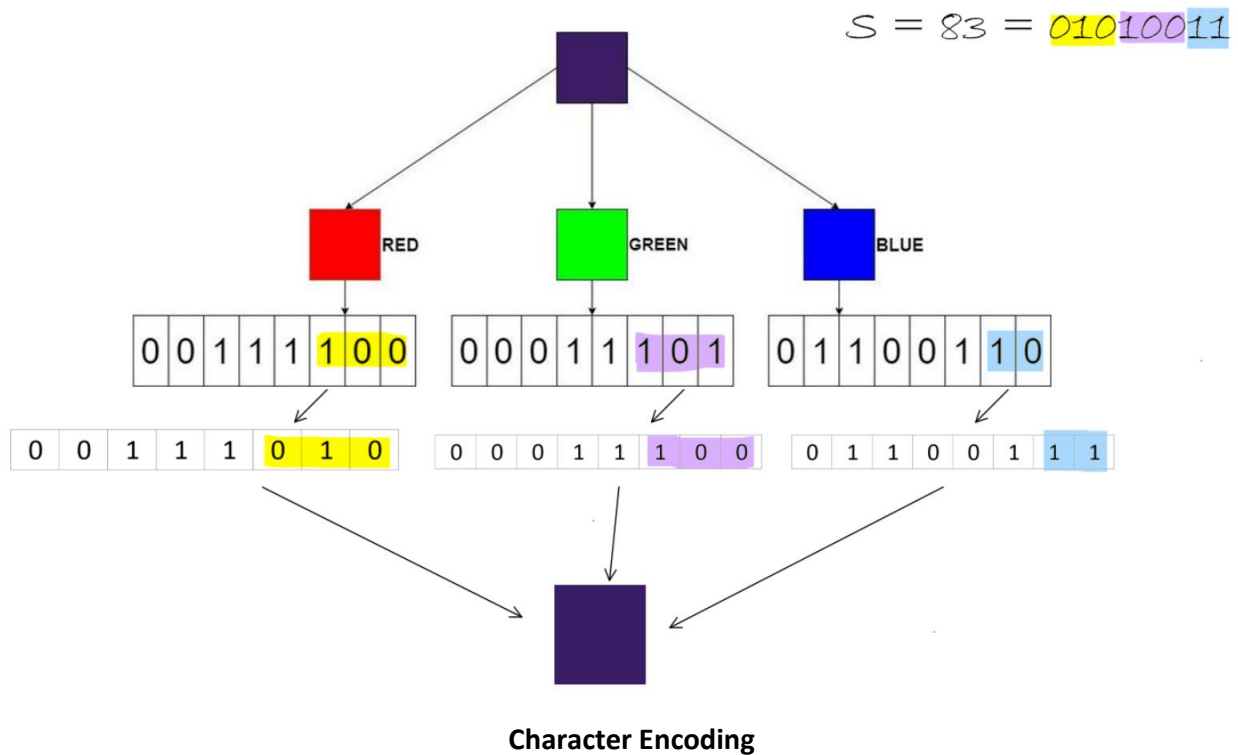
	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12

1	2	3	4	5	6	7	8	9	10	11	12
---	---	---	---	---	---	---	---	---	----	----	----

Element = 8
 $\text{Row} = (\text{Element} // \text{No. of cols}) - 1 = (8 // 4) - 1 = 1$
 $\text{Col} = \text{No. of cols} - 1 = 3$

Element = 12
 $\text{Row} = (\text{Element} // \text{No. of cols}) - 1 = (12 // 4) - 1 = 2$
 $\text{Col} = \text{No. of cols} - 1 = 3$

Predict the algorithm



Chapter 4

Result Analysis

4.1 Visual Impact

The steganographic process had minimal impact on the visual quality of the images. The changes made to the LSBs of the pixels were imperceptible to the human eye, ensuring that the stego-image retained its original appearance.

4.2 Security

The use of AES encryption for text data provides a layer of security, making it challenging for unauthorized parties to extract meaningful information without the correct decryption key.

4.3 Efficiency

The shuffling of pixels and the embedding process proved to be efficient, allowing for the hiding of substantial amounts of text in images of various sizes without significant degradation in performance.



```
● Stego image saved at: stego-image.png  
  Extracted Text: SUKRITI GUIN. This is a test image for image testing and steganography.  
○ PS E:\1-Internship Journey\College-Project> █
```

4.4 Conclusion

In conclusion, the LSB Steganography project has provided valuable insights and practical implementation of a covert information hiding technique within digital images. The project successfully leverages the least significant bit (LSB) manipulation to embed and extract text seamlessly from image files. Several key aspects and achievements have been highlighted during the course of this project.

Chapter 5

References

1. <https://www.geeksforgeeks.org/image-steganography-in-cryptography/>
2. <https://www.geeksforgeeks.org/image-based-steganography-using-python/>
3. <https://www.geeksforgeeks.org/lbs-based-image-steganography-using-matlab/>
4. <https://medium.com/swlh/lbs-image-steganography-using-python-2bbbee2c69a2#:~:text=LSB%20Steganography%20is%20an%20image,the%20message%20to%20be%20hidden.>
5. <https://onboardbase.com/blog/aes-encryption-decryption/>
6. <https://www.geeksforgeeks.org/advanced-encryption-standard-aes/>
7. <https://medium.com/quick-code/aes-implementation-in-python-a82f5>

8. <https://medium.com/quick-code/aes-implementation-in-python-a82f582f51c2>

Semester: _____

Examination: _____

Date: _____

Group No.: _____

Examiner (External): Quality of Work → overall group performance						
Rubrics	Impact on Society (4)	Novelty (4)	Literature Survey (4)	Understanding (4)	Concept and Methodology (4)	Total (20)
Examiner						

University Roll No.	Examiner (External): Individual Assessment				Total (20)
	Presentation Skill (5)	Knowledge About the Work (5)	Role in the Work (5)	Viva Voce (5)	

Comments/Suggestions (if any)	
Examiner (External)	

Examiner's Signature

Name:

Supervisor's Assessment									
University Roll No.	Contribution & Involvement (10)	Knowledge About the Work (10)	Contribution on preparing Thesis (10)	Regularity (10)	Behavior (5)	Sincerity (5)	Team Work (5)	Leadership (5)	Total (60)

Measures taken against examiners' suggestions/ comments

Supervisor's

Signature

Name:

Examiner (Internal): Quality of Work (QOW) → overall group performance					
Rubrics	Impact on Society (4)	Novelty (4)	Literature Survey (4)	Understanding (4)	Concept and Methodology (4)
Examiner					

University Roll No.	Examiner (Internal): Individual Assessment				Total (20)
	Presentation Skill (5)	Knowledge About the Work (5)	Role in the Work (5)	Viva Voce (5)	

Comments/Suggestions (if any)	
Examiner (Internal)	

Examiner's Signature

Name: