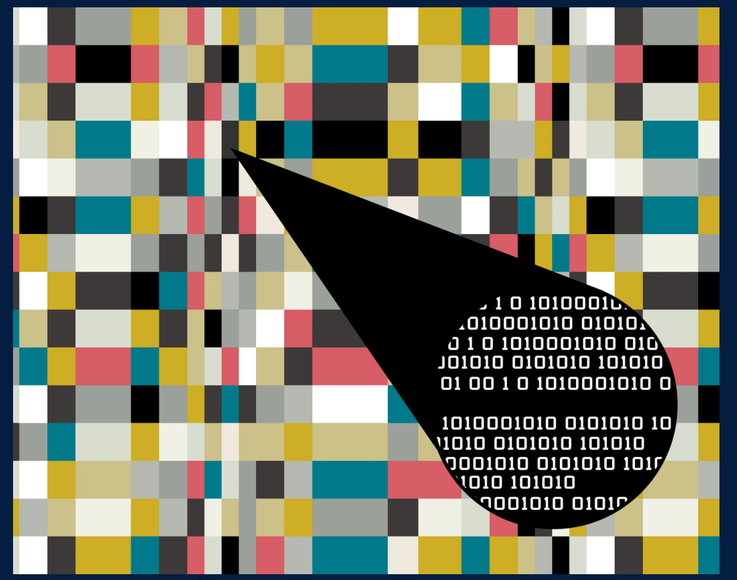
STUDENT DETAILS :

ABHISHEK BADIGER

Skills Build Email ID	abhishekbadiger2003@gmail.com
AICTE Registration Number	STU650bbc8c043701695267980
College Name	Jawaharlal Nehru New College of Engineering
State & District	Karnataka, Shivamogga, 577204
Internship Domain	CyberSecurity
Start Date & End Date	13/10/2023 to 26/11/2023

STEGANOGRAPHY



- Steganography is the art and science of hiding information within another message or object, so that it is not noticeable by human inspection. It can be used to conceal text, images, videos, audio, or any other digital content.
- Today, steganography is mainly used in digital contexts, where data can be embedded inside other files, such as images, documents, or programs.
- Steganography can be combined with encryption to enhance the security of the hidden data.

AGENDA

- The project agenda focused on steganography, mainly hiding one text message inside an image.
- Practical implementation using various steganography techniques will gives their effectiveness in creating non-differentiable composite images.
- The project aims to identify real-world applications in data security and covert communication.
- Striking a balance between theory and practice, the agenda explains comprehensive exploration of steganography's capabilities within a concise framework.

PROJECT OVERVIEW

Description :

- The "Steganography Hub" project aims to provide a secure and discreet platform for encoding and decoding messages within images. Leveraging Django as the web framework, users can seamlessly upload images, encode messages within them, and share them without raising suspicion.

Scope:

- The scope of the project extends beyond conventional steganography applications, emphasizing user privacy and data security. It incorporates best practices in web development to offer a seamless and reliable experience for users seeking a discreet means of communication.

Objectives and Goals:

- Provide a user-friendly interface for encoding and decoding.
- Implement robust user authentication and authorization.

WHO ARE THE END USERS OF THIS PROJECT?

- The "Steganography Hub" caters to a diverse range of users with varying needs and interests.
- Programmers or computer science students who took cybersecurity as part of the curriculum.
- Individuals who value discreet communication and are conscious of their online privacy.
- Students who wants to learn fundamentals of steganography.
- Anyone looking for a user-friendly platform for encoding and decoding messages within images.

CUSTOMIZATION

1.Image Selection:

- Users can upload a variety of image formats, providing flexibility in the selection of carrier images for encoding.

2.Message Encoding Options:

- The platform supports customizable message encoding through the client-side JavaScript code, allowing users to interact with encoding options.

3.User Preferences:

- Customizable user settings for a personalized experience, enhancing user engagement and satisfaction.

4.JavaScript and Python Integration:

- Seamless integration between JavaScript on the client side and Python on the server side ensures a cohesive and responsive user experience. The client-side JavaScript code handles dynamic interactions, such as file selection and user input, while the server-side Python code processes and executes the steganography algorithms. This integration allows for efficient communication between the front-end and back-end, providing users with a smooth and interactive platform.

5.Custom Encoding/Decoding with LSB Algorithm:

- The platform employs the Least Significant Bit (LSB) algorithm within custom Python functions for encoding (hide_text_in_image) and decoding (decode_text_from_image). The LSB algorithm is a widely-used steganographic technique, utilizing the least significant bit of pixel values to hide information. These tailored functions, based on the LSB algorithm, contribute to the unique and secure nature of the steganography process within the "Steganography Hub."

MODELLING

- Python code for hiding a Text in image

```
import cv2
import os
import string

img = cv2.imread("WhatsApp Image 2023-11-26 at 11.10.42 AM.jpeg")

msg = input("Enter secert message")

password = input("Enter password")

d={}
c={}

for i in range(255):
    d[chr(i)]=i
    c[i] = chr(i)

m=0
n=0
z=0

for i in range(len(msg)):
    img[n,m,z] = d[msg[i]]
    n=n+1
    m=m+1
    z=(z+1)%3

cv2.imwrite("Encryptedmsg.jpg",img)

os.system("start Encryptedmsg.jpg")
```

- Python code for hiding a Text in image

```
message =""

n=0
m=0
z=0

pas = input("Enter passcode for Decryption")

if password == pas:
    for i in range(len(msg)):
        message = message + c[img[n,m,z]]
        n=n+1
        m=m+1
        z=(z+1) % 3
    print("Decryption message",message)
else:
    print("Not valid key")
```

How code works..?

Text to Binary Function:

- This function takes a string text as input and converts each character in the text to its 8-bit binary representation using the ord function and string formatting. The resulting binary string is returned.

Binary_to_text Function:

- This function takes a binary message as input and converts it back to text. It does this by iterating over the binary string in chunks of 8 bits, converting each chunk to an integer using int(..., 2), and then converting the integer to its corresponding ASCII character using chr. The resulting characters are joined together to form the original text.

Encrypt_image Function :

- This function takes the path of an image, a secret message, and a password as inputs. It reads the image using OpenCV (cv2.imread), converts the secret message to binary, and appends a delimiter ('1111111111111110') to mark the end of the message

- It then iterates over each bit in the binary message and modifies the least significant bit (LSB) of each pixel in the image to carry the binary message. The variables n, m, and z keep track of the pixel position.

- Finally, it saves the modified image with the encrypted message as "Encryptedmsg.png" and prints a success message. The os.system("start Encryptedmsg.png") line opens the encrypted image using the default image viewer.

Decrypt_image Function :

- This function takes the path of an encrypted image and a password as inputs. It reads the encrypted image using OpenCV.

- It then iterates over each pixel of the image, extracting the LSB of each color channel to reconstruct the binary message. The loop continues until it encounters the delimiter ('1111111111111110') that marks the end of the message.

- After extracting the binary message, it removes the delimiter and converts the binary message to text using the binary_to_text function. The resulting decrypted message is printed.

- If the entered password does not match the provided password, it prints a message indicating that it's not a valid passcode.

```
/Users/abhishekbadiger/Desktop/opencv_project/session/bin/python /Users/abhishekbadiger/Desktop/openc
v_project/main.py
abhishekbadiger@Abhisheks-Mac opencv_project % /Users/abhishekbadiger/Desktop/opencv_project/session/
bin/python /Users/abhishekbadiger
/Desktop/opencv_project/main.py
Enter secert message hello how are you
Enter password abc
sh: start: command not found
Enter passcode for Decryption abc
Decryption message hello how are you
abhishekbadiger@Abhisheks-Mac opencv_project %
```

WhatsApp Image 2023-11-26 at 11.10.42 AM.jpeg

Encryptedmsg.jpg

Thank You

Link: https://github.com/4JN21AI001/cyber-security.git