CAPSTONE PROJECT

SECURE DATA HIDING IN IMAGE USING STEGNOGRAPHY

Presented By: Mudasir Samad Dand
Student Name : Mudasir Samad Dand
College Name & Department : North campus Delina , Btech CSE

edunet foundation

OUTLINE

- Problem Statement
- Technology used
- Wow factor
- End users
- Result
- Conclusion
- Git-hub Link

PROBLEM STATEMENT

Existing Issue : Traditional steganography lacks robust encryption, exposing hidden data to extraction using “steg analysis”.

Solution :Embed messages in images using XOR encryption and least Significant Bit (LSB) steganography to ensure confidentially even if the data is found by Steg analysis.

TECHNOLOGY USED

Libraries: OpenCV (image processing), struct (binary data handling),OS (system Operations).

Platform : Python 3.x.

IDE : VS code

WOW FACTORS

Unique features:

a) Dual Security : combines XOR encryption + LSB steganography

b) Dynamic Key Handling

c) Error Handling : checks image size to prevent overflow

END USERS

Target Audience : Cybersecurity professionals, journalists ,corporations having sensitive data.

Use Case : Securely share credentials or confidential notes via social media images .

RESULTS

```python
def reveal_message(image_path):
        col += 1
        channel = (channel + 1) % 3

    data_length = struct.unpack('!I', bytes(length_bytes))[0]

    # Read encrypted data
    encrypted_data = []
    for _ in range(data_length):
        if row >= img_row or col >= img_col:
            print("Error: Data corrupted")
            return
        encrypted_data.append(img[row, col, channel])
        row += 1
        col += 1
        channel = (channel + 1) % 3

    # Decrypt message
    secret_key = input("Enter decryption key: ")
    decrypted = []
    key_len = len(secret_key)
    for i, byte in enumerate(encrypted_data):
        key_char = secret_key[i % key_len]
        decrypted.append(chr(byte ^ ord(key_char)))

    print("Hidden message:", ''.join(decrypted))

# Main program
choice = input("Hide(h) or reveal(r) message? (h/r): ").lower()
if choice == 'h':
    img_path = input("Enter cover image path: ")
    msg = input("Enter secret message: ")
    key = input("Create encryption key: ")
    hide_message(img_path, msg, key)
elif choice == 'r':
    img_path = input("Enter secret image path: ")
```

```python
import cv2
import os
import struct

def hide_message(image_path, message, secret_key):
    img = cv2.imread(image_path)
    if img is None:
        print("Error: Couldn't load image")
        return

    # Encrypt message using XOR with secret key
    encrypted_data = []
    key_len = len(secret_key)
    for i, char in enumerate(message):#Each char represents one character from the secret messag
        key_char = secret_key[i % key_len]#if key is hsorter then message then it makes it loop from start again of the key
        encrypted_data.append(ord(char) ^ ord(key_char))#ord(char): Converts the message character to its ASCII value.
                                                        #ord(key_char): Converts the corresponding key character to its ASCII value.

    # Add message length header
    data_length = len(encrypted_data)
    length_header = struct.pack('!I', data_length)

    # Embed data in image
    row, col, channel = 0, 0, 0
    img_row, img_col, _ = img.shape

    # Store length header ; The first few pixels store the message length
    for byte in length_header:
        if row >= img_row or col >= img_col:
            print("Error: Image too small")
            return
        img[row, col, channel] = byte
        row += 1
        col += 1
        channel = (channel + 1) % 3

    # Store encrypted message
    for byte in encrypted_data:
        if row >= img_row or col >= img_col:
            print("Error: Image too small")
            return
        img[row, col, channel] = byte
```

project.py secret_image.png

secret_image.png

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\mudas\OneDrive\Desktop\project> python -u "c:\Users\mudas\OneDrive\Desktop\project\project.py"
Hide(h) or reveal(r) message? (h/r): h
Enter cover image path: pic.jpg
Enter secret message: hi my name is mudasir
Create encryption key: hide
Message hidden in secret_image.png
```

CONCLUSION

Summary :

Successfully hides encrypted messages in images using lightweight, secure methods.

Addresses the gap in traditional steganography tools.

GITHUB LINK

https://github.com/Mady520/Stego_project.git

FUTURE SCOPE

- Add support for AES encryption for stronger security .
  Develop a GUI for easier use.
  Extend functionality to hide messages in audio and video files

THANK YOU

edunet foundation