

Project 3: Steganography

Name: Shefali Athavale

Approach:

For this project, I wrote a Steganography Program using Python. The program is written for JPEG image files only. I used the following libraries for the same - os, hashlib, binascii, re. The program gives the user a choice among the two - Embed or Extract. If the user selects 'embed'(i.e. choice '1'), the program executes the 'embed()' function. If the user selects extract(i.e. choice '2'), the program executes the extract()' function. We will now look at both the functions individually.

embed() function:

This function is used to embed a secret message in a carrier file. If the user selects this option, the program will ask the user to input the carrier file name. It is assumed that the carrier file is present in the same folder as this script/program. If not, the user has to enter the path for the file. The program then asks the user to enter the secret message he/she wants to embed in the carrier file. We then convert the text message into hexadecimal format to embed it in the image. The program then embeds the message just before the trailer signature of the JPEG EOF offset i.e. just before the 'FF D9' signature. The program also adds 4 bytes of additional data 'FF FF FF FF' before adding the message for better retrieval of the message at the 'extract' stage. The program then outputs the details of the original and modified carrier file like the file name, size and its MD5 hash. The program also outputs the embedded secret message.

extract() function:

This function is used to extract the secret message from the carrier file. If the user selects this option, the program will ask the user to input the carrier file name(this will be the modified carrier file name). It is assumed that the carrier file is present in the same folder as this script/program. If not, the user has to enter the path for the file. The program assumes that the hidden secret message is just before the trailer signature of the JPEG EOF offset i.e. just before the 'FF D9' signature. The program then retrieves the JPEG EOF offset as well as the offset of additional bytes of data (FF FF FF FF) which we had added. The secret message is the data between 'FF FF FF FF' and 'FF D9'. The program retrieves that data, converts it from hexadecimal format to string and outputs it to the screen.

Visual Studio Code is used for writing the program. The outputs from both the files and the code is attached below.

Output:

embed():

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL zsh + v [ ] [ ] ^ x
shefaliathavale@Shefalis-MacBook-Air Project 3 - Steganography % /usr/bin/python3 "/Users/shefaliathavale/CUB_MS/Sem2/Digital Forensics/Project 3 - Steganography/Project3_ShefaliAthavale_Code.py"

1: Embed a message in a file
2: Extract a message from a file
Enter your choice: 1
Enter name of the carrier file: image.jpeg
Enter message to embed: Meet at 7.pm SHARP at the park gate. Don't be late!!

Original Carrier File Details:
File Name: image.jpeg
Size: 40.777 KB
MD5 Hash: 5388d980554dfd557f66efcbff3805b

Modified Carrier File Details:
File Name: image_modified.jpeg
Size: 40.833 KB
MD5 Hash: a371b38e75f626348df43d232ea8e27e

Embedded secret message is: Meet at 7.pm SHARP at the park gate. Don't be late!!
```

extract():

```
shefaliathavale@Shefalis-MacBook-Air Project 3 - Steganography % /usr/bin/python3 "/Users/shefaliathavale/CUB_MS/Sem2/Digital Forensics/Project 3 - Steganography/Project3_ShefaliAthavale_Code.py"

1: Embed a message in a file
2: Extract a message from a file
Enter your choice: 2
Enter name of the carrier file: image_modified.jpeg

Extracted secret message is: Meet at 7.pm SHARP at the park gate. Don't be late!!

shefaliathavale@Shefalis-MacBook-Air Project 3 - Steganography % [ ]
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
shefaliathavale@Shefalis-MacBook-Air Project 3 - Steganography % /usr/bin/python3 "/Users/shefaliathavale/CUB_MS/Sem2/Digital Forensics/Project 3 - Steganography/Project3_ShefaliAthavale_Code.py"

1: Embed a message in a file
2: Extract a message from a file
Enter your choice: 1
Enter name of the carrier file: image.jpeg
Enter message to embed: Meet at 7.pm SHARP at the park gate. Don't be late!!

Original Carrier File Details:
File Name: image.jpeg
Size: 40.777 KB
MD5 Hash: 5388d980554dfd557f66efcbff3805b

Modified Carrier File Details:
File Name: image_modified.jpeg
Size: 40.833 KB
MD5 Hash: a371b38e75f626348df43d232ea8e27e

Embedded secret message is: Meet at 7.pm SHARP at the park gate. Don't be late!!

shefaliathavale@Shefalis-MacBook-Air Project 3 - Steganography % /usr/bin/python3 "/Users/shefaliathavale/CUB_MS/Sem2/Digital Forensics/Project 3 - Steganography/Project3_ShefaliAthavale_Code.py"

1: Embed a message in a file
2: Extract a message from a file
Enter your choice: 2
Enter name of the carrier file: image_modified.jpeg

Extracted secret message is: Meet at 7.pm SHARP at the park gate. Don't be late!!

shefaliathavale@Shefalis-MacBook-Air Project 3 - Steganography % [ ]
```

Project 3 - Steganography



Project3_ShefaliA
thavale_Code.py



image.jpeg



image_modified.j
peg

image.jpeg:

```
40512 910808E3 8B0F445A E25DCE06 CAA35451 AAEE8492 23938010 2CBCC8F1 AF0372A7 B5FC0205 5A2A601A 20B31571 2112C70A 834428B5 511E5D91 1C0C9F45 D0067F9C
40576 B401D929 0D53D5FF 00DA3F45 44529A79 DB398187 64E9ABFE 3311A9C2 1F53F89F AA686A16 FAF9FC33 DF70A43E 22760C3E 52BDEF86 BE88F38B 90365E9F A9FC547C
40640 8FFD82BD 1331E7AC DFFD73AE 45664431 5E86B788 CF8599F0 9FF6E5F5 5A9E574E B9918EC1 13236003 44D30601 2D1FE69A D96E3211 60574659 6513D544 3F9213CE
40704 FF00ED35 0AC42E1A 82DF62B2 7A6D3208 D42D8FFD C3FF00F9 63FF00F2 0B1FE2F6 5C7BFF00 E4E9D1AD 0EC98011 29AAF824 D523E87F C2427FC8 2668FF00 B72FFF00
40768 89FF0008 E75BE1FF D9
Signed Int |le, dec (select some data)
40775 out of 40777 bytes
```

```
20B31571 2112C70A 834428B5 511E5D91 1C0C9F45 D0067F9C . . . DZ.] . .TQ....#... ,... r... Z*` . q! . .D+.Q ]. .E. .
DF70A43E 22760C3E 52BDEF86 BE88F38B 90365E9F A9FC547C . .) S... ?EDR.y.9..d...3 .. S...hj ...3.p.>"v >R.....6^...Tl
44D30601 2D1FE69A D96E3211 60574659 6513D544 3F9213CE ... 1....s.EfD1^.....Z.WN.... #` D. - ...n2 `WfYe .D?. .
0EC98011 29AAF824 D523E87F C2427FC8 2668FF00 B72FFF00 .5 .. .b.zm2 .-.... .c. . .\{. .... .) ..$.# .B .&k. ./
.. .[...
40775 out of 40777 bytes
```

image_modified.jpeg:

```
40512 910808E3 8B0F445A E25DCE06 CAA35451 AAEE8492 23938010 2CBCC8F1 AF0372A7 B5FC0205 5A2A601A 20B31571 2112C70A 834428B5 511E5D91 1C0C9F45 D0067F9C
40576 B401D929 0D53D5FF 00DA3F45 44529A79 DB398187 64E9ABFE 3311A9C2 1F53F89F AA686A16 FAF9FC33 DF70A43E 22760C3E 52BDEF86 BE88F38B 90365E9F A9FC547C
40640 8FFD82BD 1331E7AC DFFD73AE 45664431 5E86B788 CF8599F0 9FF6E5F5 5A9E574E B9918EC1 13236003 44D30601 2D1FE69A D96E3211 60574659 6513D544 3F9213CE
40704 FF00ED35 0AC42E1A 82DF62B2 7A6D3208 D42D8FFD C3FF00F9 63FF00F2 0B1FE2F6 5C7BFF00 E4E9D1AD 0EC98011 29AAF824 D523E87F C2427FC8 2668FF00 B72FFF00
40768 89FF0008 E75BE1FF FFFFFFFF 65657420 61742037 2E706D20 53484152 50206174 20746865 20706172 68206761 74652E20 446F6E27 74206265 206C6174 652121FF
40832 D9
Signed Int |le, dec (select some data)
0x9F7F out of 0x9F81 bytes
```

```
20B31571 2112C70A 834428B5 511E5D91 1C0C9F45 D0067F9C . . . DZ.] . .TQ....#... ,... r... Z*` . q! . .D+.Q ]. .E. .
DF70A43E 22760C3E 52BDEF86 BE88F38B 90365E9F A9FC547C . .) S... ?EDR.y.9..d...3 .. S...hj ...3.p.>"v >R.....6^...Tl
44D30601 2D1FE69A D96E3211 60574659 6513D544 3F9213CE ... 1....s.EfD1^.....Z.WN.... #` D. - ...n2 `WfYe .D?. .
0EC98011 29AAF824 D523E87F C2427FC8 2668FF00 B72FFF00 .5 .. .b.zm2 .-.... .c. . .\{. .... .) ..$.# .B .&k. ./
68206761 74652E20 446F6E27 74206265 206C6174 652121FF .. .[.....Meet at 7.p.m SHARP at the park gate. Don't be late!!!
0x9F7F out of 0x9F81 bytes
```

Code:

```
import binascii
import re
import os
import hashlib

## Function to embed secret message in JPEG File
def embed(carrierFile,message):

    # Open carrier file and read it's contents
    with open(carrierFile, 'rb') as f:
        content = f.read()
        f.close()

    # Find the EOF offset for the JPEG file and retrieve contents of the image upto EOF
    eof_list_jpeg = [match.start() for match in re.finditer(re.escape(b'\xFF\xD9'),content)]
    eofImage = eof_list_jpeg[-1]
    hexcontent = binascii.hexlify(content[:eofImage])

    # Convert text message into bytes and add it to the original image content
    # We add 4 additional bytes of data before appending the secret message
    # for better retrieval of the secret message from the file
    msg = message.encode('utf-8')
    msg = binascii.hexlify(msg).decode('utf-8')
    dataMsg = bytes(msg, 'utf-8')
    hexcontent += b'ffffffff'+dataMsg + b'ffd9'
    data = bytes.fromhex(hexcontent.decode("utf-8"))

    # Retrieve original carrier file name
    extension = ''.join(carrierFile.split(".")[-1])
    filename = carrierFile.split(".")[:-1]

    # Write the contents to the new file
    embedFile = filename[0]+'_modified.'+extension
    with open(embedFile, 'wb') as f1:
        f1.write(data)
        f1.close()
```

```

# Output details about Original and Modified Carrier File to the screen
print()
print("Original Carrier File Details:")
print("File Name: "+carrierFile)
print("Size: "+str(os.path.getsize(carrierFile)/1000)+" KB")
md5Hash = hashlib.md5(content).hexdigest()
print("MD5 Hash: "+md5Hash)
print()
print("Modified Carrier File Details:")
print("File Name: "+embedFile)
print("Size: "+str(os.path.getsize(embedFile)/1000)+" KB")
md5Hash1 = hashlib.md5(data).hexdigest()
print("MD5 Hash: "+md5Hash1)

## Function to extract secret message from a JPEG File
def extract(embeddedFile):

    # Open carrier file and read it's contents
    with open(embeddedFile, 'rb') as f:
        content = f.read()
        f.close()

    # Find the EOF offset for the JPEG file and retrieve contents of the image upto EOF
    offset
    eof_list_jpeg = [match.start() for match in
re.finditer(re.escape(b'\xFF\xD9'),content)]

    # Find the offset of the start of 4 additional bytes of data
    eof_list_space = [match.start()+4 for match in
re.finditer(re.escape(b'\xFF\xFF\xFF\xFF'),content)]

    eofImage = eof_list_jpeg[-1]
    eofSpace = eof_list_space[-1]

    # Secret Message is the message between 4 additional bytes of data and EOF offset
    for JPEG image
    secMsg = content[eofSpace:eofImage].decode('utf-8')

    return secMsg

```

```
if __name__ == "__main__":  
    print()  
    print("1: Embed a message in a file")  
    print("2: Extract a message from a file")  
  
    method = input("Enter your choice: ")  
    if method == "1":  
        filename = input("Enter name of the carrier file: ")  
        message = input("Enter message to embed: ")  
        embed(filename, message)  
        print()  
        print("Embedded secret message is: "+message)  
        print()  
  
    elif method=="2":  
        filename = input("Enter name of the carrier file: ")  
        secretMessage = extract(filename)  
        print()  
        print("Extracted secret message is: "+secretMessage)  
        print()
```