Assignment 8 : Steganography

Name: Israel Mbiyavanga David

Enrollment Number:012200300004055

Subject: Cryptography

1. Using Windows Commands

Technique: Binary Concatenation using Windows copy /b command

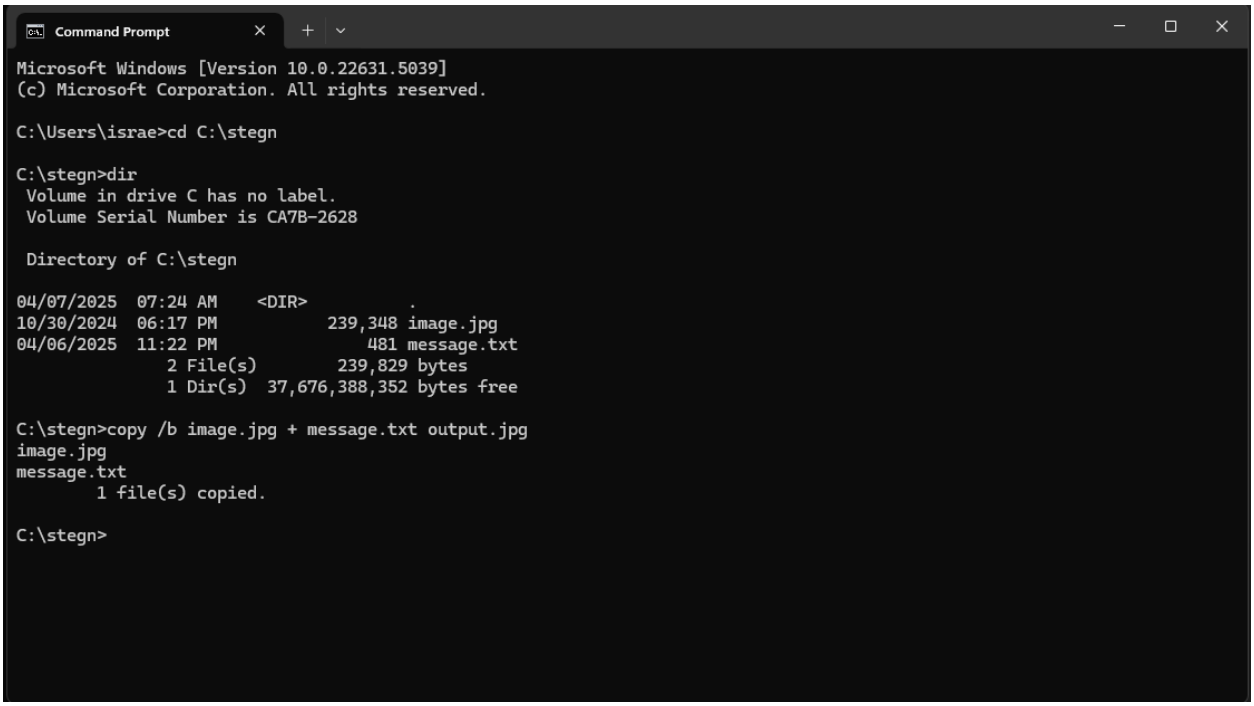
Command Used:

copy /b image.jpg + message.txt output.jpg

Explanation:

- The /b flag indicates binary mode.
- It combines the original image (image.jpg) with a text file (message.txt) and saves it as output.jpg.
- The hidden file doesn't affect the view of the image but can be extracted using tools like certutil, binwalk, or by opening in a hex editor.

Screenshot:



2. Using Python with LSB Algorithm

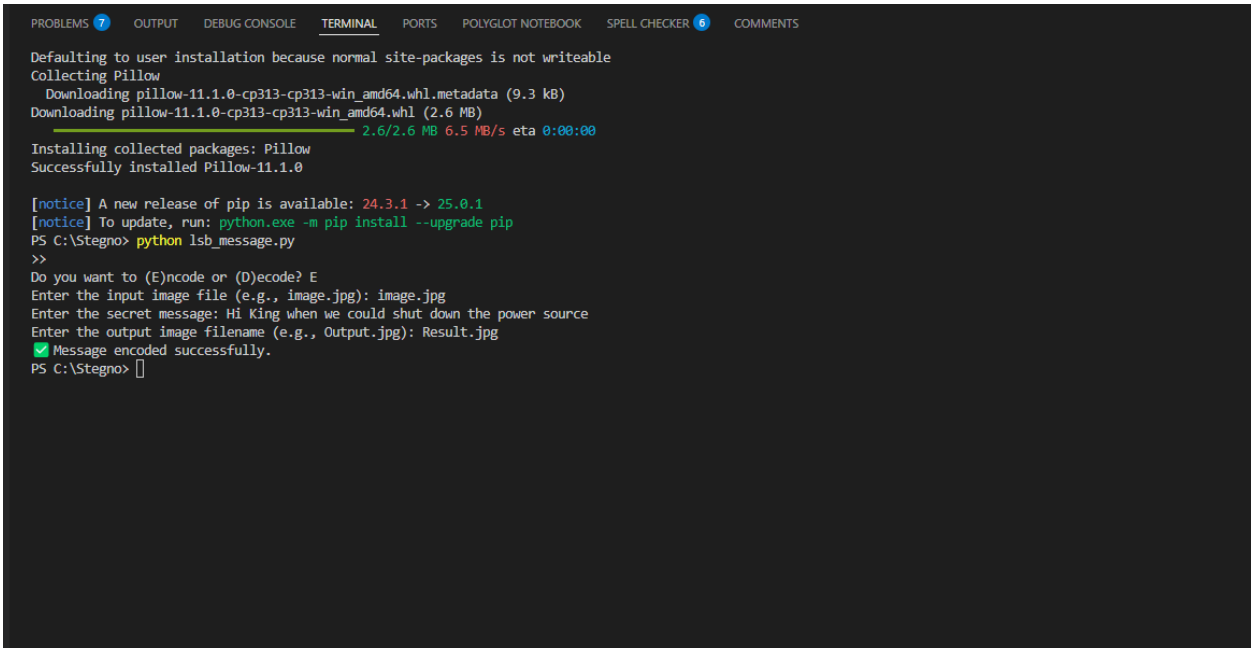
Language Used: Python

File: lsb_message.py

Algorithm: LSB (Least Significant Bit)

Explanation:

- Converts the secret message into binary.
- Embeds it into the least significant bits of image pixels.
- Saves the modified image as output.



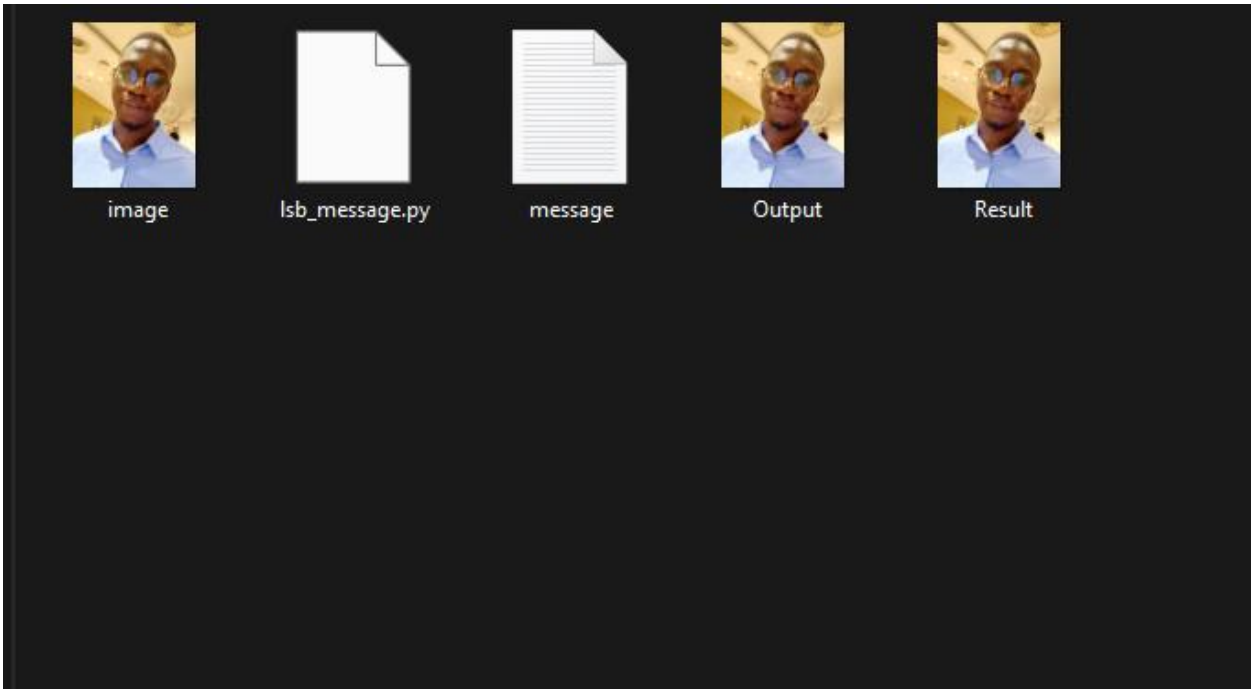


image
lsb_message.py
message
Output
Result

3. Using OpenPuff Tool (Windows GUI)

Tool: OpenPuff v4.01
Website: https://embeddedsw.net/OpenPuff_Steganography_Home.html

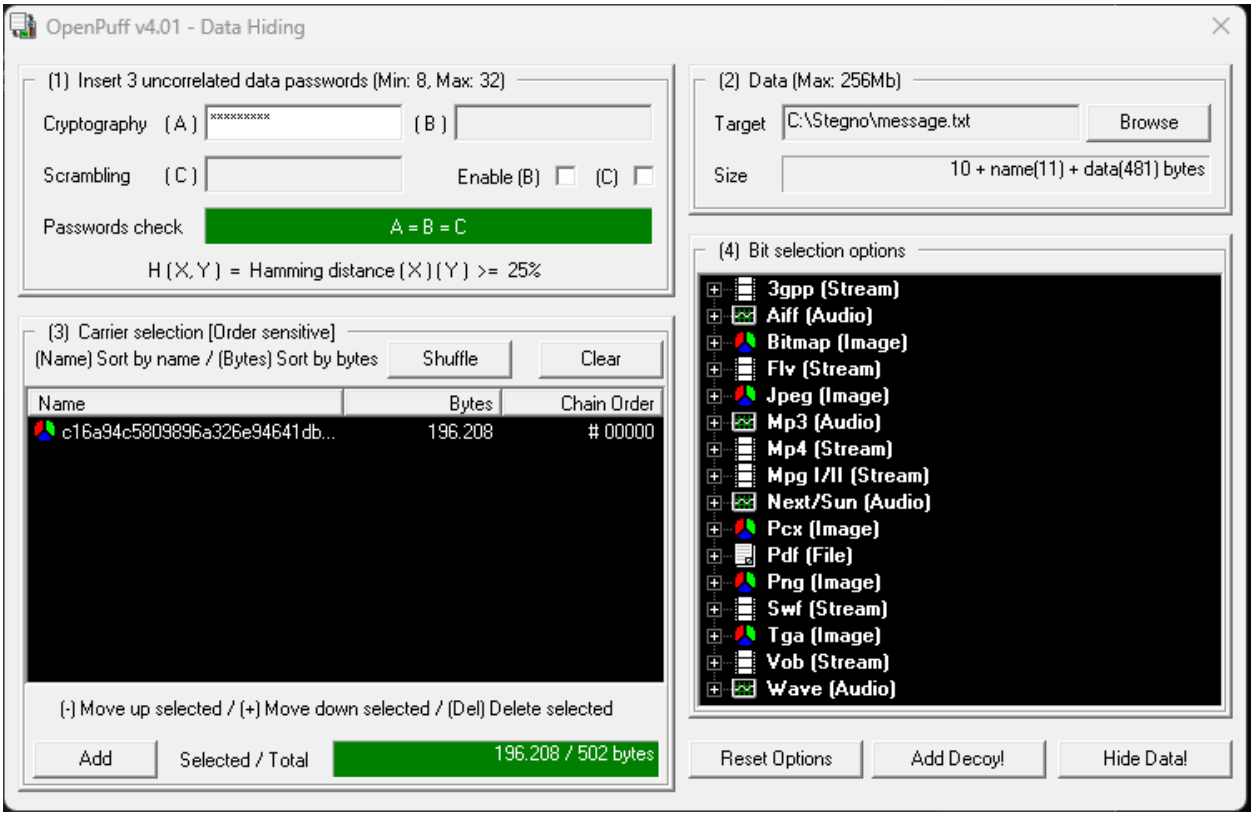
Steps:

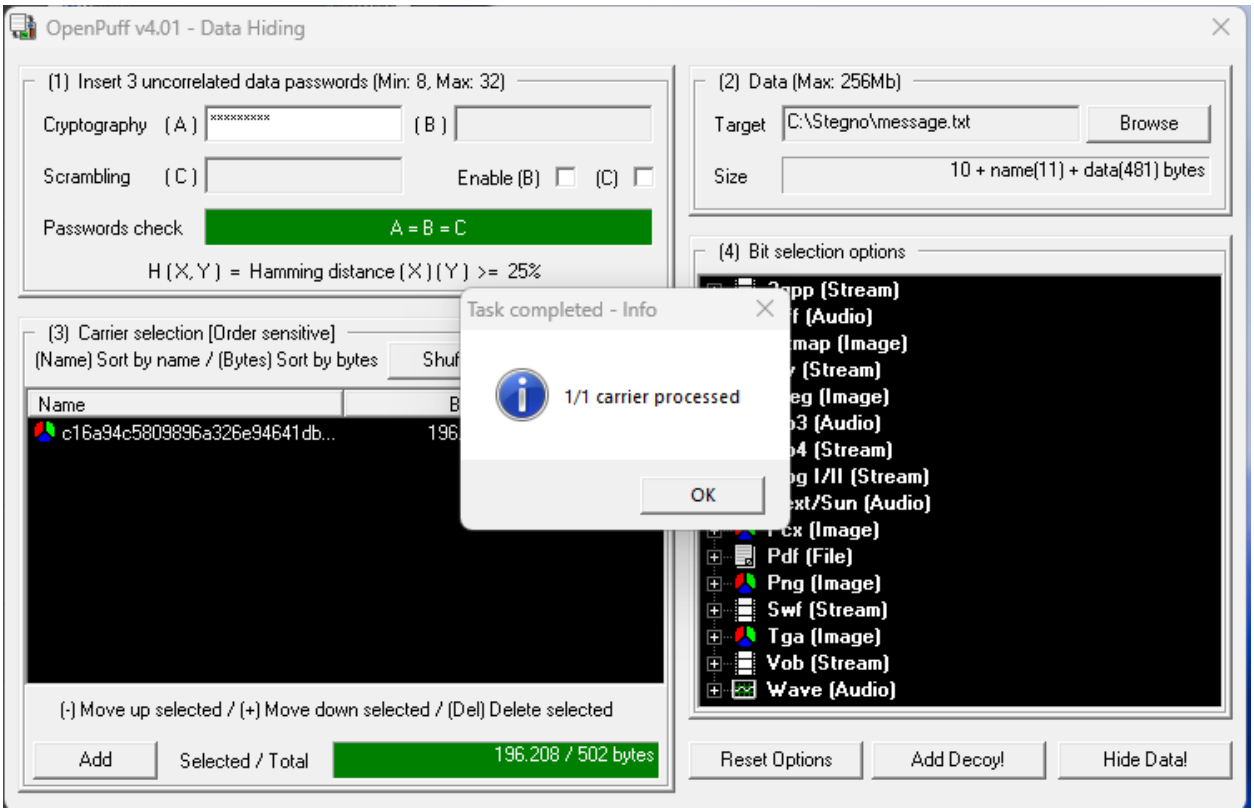
1. Set three passwords (Cryptography, Scrambling A & B).
2. Select secret file (e.g., message.txt).
3. Select carrier file (e.g., image.jpg).
4. Click "Hide Data".

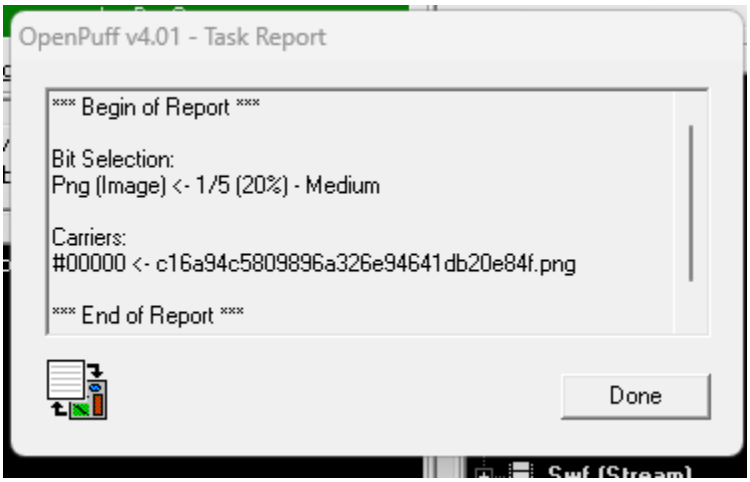
Explanation:

- Offers advanced security using layered encryption and scrambling.
- Prevents statistical and visual detection.

Screenshot:




OpenPuff v4.01 - Data Hiding
(1) Insert 3 uncorrelated data passwords (Min: 8, Max: 32)
Cryptography (A) **********
(B)
Scrambling (C)
Enable (B) (C)
Passwords check
A = B = C
H (X, Y) = Hamming distance (X) (Y) >= 25%
(2) Data (Max: 256Mb)
Target C:\Stegno\message.txt
Browse
Size
10 + name(11) + data(481) bytes
(4) Bit selection options
(3) Carrier selection [Order sensitive]
(Name) Sort by name / (Bytes) Sort by bytes
Name
c16a94c5809896a326e94641db...
Task completed - Info
1/1 carrier processed
OK
Pdf (File)
Png (Image)
Swf (Stream)
Tga (Image)
Vob (Stream)
Wave (Audio)
(-) Move up selected / (+) Move down selected / (Del) Delete selected
Add
Selected / Total
196.208 / 502 bytes
Reset Options
Add Decoy!
Hide Data!


OpenPuff v4.01 - Task Report
*** Begin of Report ***
Bit Selection:
Png (Image) <- 1/5 (20%) - Medium
Carriers:
#00000 <- c16a94c5809896a326e94641db20e84f.png
*** End of Report ***
Done

4. Using StegHide on Linux

Tool: StegHide

Installation:

sudo apt update

sudo apt install steghide

Hiding Command:

steghide embed -cf image.jpg -ef message.txt

Enter passphrase when prompted

Extracting Command:

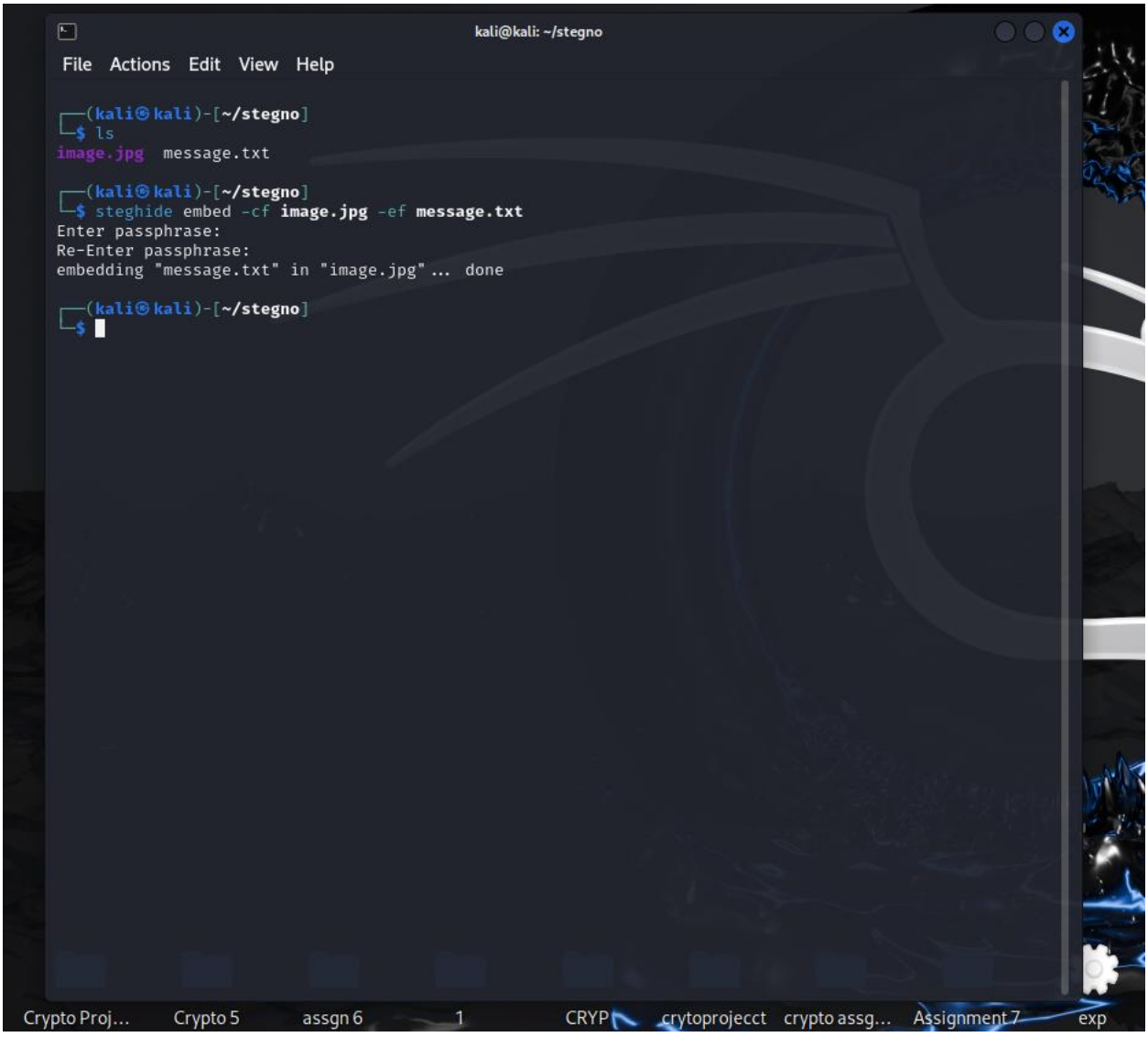
steghide extract -sf image.jpg

Enter passphrase to extract message.txt

Explanation:

- Securely hides text or data in a cover image.
- Requires passphrase for both embedding and extraction.

Screenshot:


kali@kali: ~/stegno
File Actions Edit View Help
(kali㉿kali)-[~/stegno]
$ ls
image.jpg message.txt
(kali㉿kali)-[~/stegno]
$ steghide embed -cf image.jpg -ef message.txt
Enter passphrase:
Re-Enter passphrase:
embedding "message.txt" in "image.jpg"... done
(kali㉿kali)-[~/stegno]
$
Crypto Proj... Crypto 5 assgn 6 1 CRYP crytoprojecct crypto assg... Assignment 7 exp

📁 **File Organization:**

Steganography-Assignment/

|

|├-- Windows/

|│ └-- image.jpg, message.txt, output.jpg

|

|├-- Python/

|│ └-- lsb_message.py, image.jpg, output.jpg

|

|├-- OpenPuff/

|│ └-- image.jpg, message.txt

|

|└-- Linux-StegHide/

  └-- image.jpg, message.txt

🎓 **Conclusion:**

Through this practical assignment, various steganography techniques were successfully studied and implemented on both Windows and Linux platforms. These methods help ensure data confidentiality by hiding the existence of communication itself, an essential component of secure digital communication.