Introduction

Steganography is the art of hiding the fact that communication is taking place, by hiding information in other information. Many carrier file formats can be used, but digital images are the most popular because of their frequency on the internet. For hiding secret information in images, there exists a large variety of steganography techniques some are more complex than others and all of them have respective strong and weak points. Different applications may require absolute invisibility of the secret information, while others require a large secret message to be hidden.

Steganography is the practice of hiding private or sensitive information within something that appears to be nothing out of the usual. Steganography is often confused with cryptology because the two are similar in the way that they both are used to protect important information. The difference between two is that steganography involves hiding information so it appears that no information is hidden at all. If a person or persons views the object that the information is hidden inside he or she will have no idea that there is any hidden information, therefore the person will not attempt to decrypt the information.

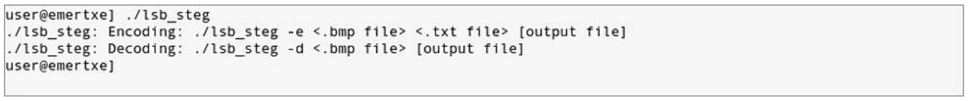
What steganography essentially does is exploit human perception, human senses are not trained to look for files that have information inside them, although this software is available that can do what is called Steganography. The most common use of steganography is to hide a file inside another file.

Requirement Details

1. The application accepts an image file say .bmp along with a text file that contains the message to be steged
2. Analyze the size of the message file to check whether the message could fit in the provided .bmp image
3. Provide an option to steg a magic string which could be useful to identify whether the image is steged or not
4. The application should provide an option to decrypt the file
5. This has to be a command-line application and all the options have to be passed as a command-line argument

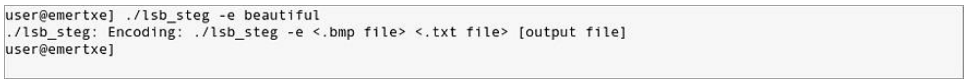
Sample Output

Here are the sample output expected by the end of project execution.



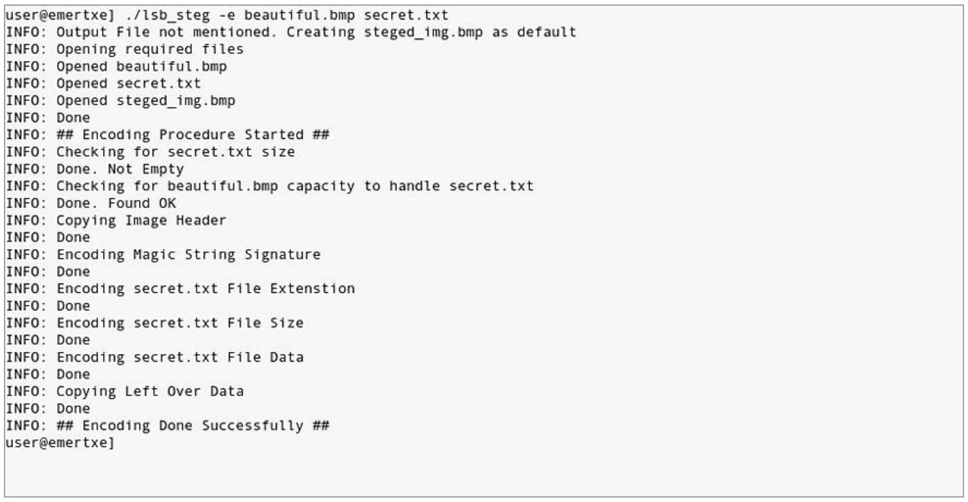
```
user@emertxe] ./lsb_steg
./lsb_steg: Encoding: ./lsb_steg -e <.bmp file> <.txt file> [output file]
./lsb_steg: Decoding: ./lsb_steg -d <.bmp file> [output file]
user@emertxe]
```

Fig1: Command Line usage with arguments



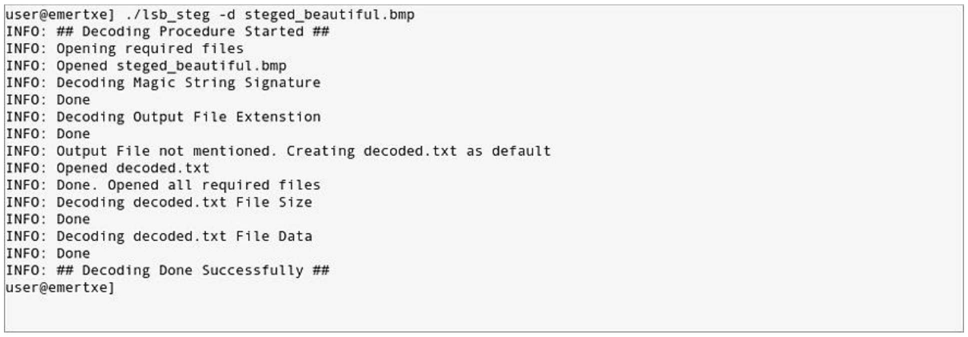
```
user@emertxe] ./lsb_steg -e beautiful
./lsb_steg: Encoding: ./lsb_steg -e <.bmp file> <.txt file> [output file]
user@emertxe]
```

Fig2: Encoding usage via command line



```
user@emertxe] ./lsb_steg -e beautiful.bmp secret.txt
INFO: Output File not mentioned. Creating steged_img.bmp as default
INFO: Opening required files
INFO: Opened beautiful.bmp
INFO: Opened secret.txt
INFO: Opened steged_img.bmp
INFO: Done
INFO: ## Encoding Procedure Started ##
INFO: Checking for secret.txt size
INFO: Done. Not Empty
INFO: Checking for beautiful.bmp capacity to handle secret.txt
INFO: Done. Found OK
INFO: Copying Image Header
INFO: Done
INFO: Encoding Magic String Signature
INFO: Done
INFO: Encoding secret.txt File Extenstion
INFO: Done
INFO: Encoding secret.txt File Size
INFO: Done
INFO: Encoding secret.txt File Data
INFO: Done
INFO: Copying Left Over Data
INFO: Done
INFO: ## Encoding Done Successfully ##
user@emertxe]
```

Fig3: Encoding steps and output at different stages



```
user@emertxe] ./lsb_steg -d steged_beautiful.bmp
INFO: ## Decoding Procedure Started ##
INFO: Opening required files
INFO: Opened steged_beautiful.bmp
INFO: Decoding Magic String Signature
INFO: Done
INFO: Decoding Output File Extenstion
INFO: Done
INFO: Output File not mentioned. Creating decoded.txt as default
INFO: Opened decoded.txt
INFO: Done. Opened all required files
INFO: Decoding decoded.txt File Size
INFO: Done
INFO: Decoding decoded.txt File Data
INFO: Done
INFO: ## Decoding Done Successfully ##
user@emertxe]
```

Fig4: Decoding steps and output