|  |
| --- |
| SWE 599 Final Project Report |
| An Image Steganography Tool |
| Reha YILMAZLAR |

|  |
| --- |
|  |

**Contents**

[**Introduction 2**](#_Toc71318826)

[**Image Steganography 2**](#_Toc71318827)

[**Least Significant Bit (LSB) 2**](#_Toc71318828)

[**Description 6**](#_Toc71318829)

[**Requirements 6**](#_Toc71318830)

[**Functional Requirement 6**](#_Toc71318831)

[**Non-Functional Requirements 6**](#_Toc71318832)

[**Use Case Diagram 7**](#_Toc71318833)

[**Sequence Diagram 8**](#_Toc71318834)

[**Class Diagram 8**](#_Toc71318835)

[**Mock-Up 9**](#_Toc71318836)

# Introduction

The word of steganography comes from the Greek language and it means secret/covered writing. It is used to hide messages in different mediums such as video, image, and audio in a digital world.

The increase in usage of messaging applications and how they store and transfer data between nodes are getting more questionable every day. Recent data leaks, hackings in big tech companies show us that cybersecurity should be and will be a more important aspect of our lives.

Steganography is a part of cybersecurity. Back then terrorists, spies, hackers used steganography to transfer secret messages in plain sight. They were and still are probably hiding messages in places that we visit every day but do not pay attention enough. Nowadays, you could even upload a picture to one of your favorite social media platforms with a secret message embedded, and probably no one would even think that it might include a secret message. It could be anywhere, in a video, music file, image file, and so on.

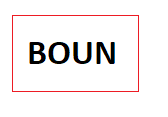
# Image Steganography

With the increase of cybersecurity attacks that came with the pandemic, people are being trained against those attacks a lot more. So that, steganography methods are used more than ever probably. The most used kind of steganography in those kinds of cybersecurity training is image steganography. In this kind of steganography, an image is used as the cover media.

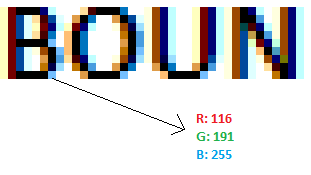
Virtual machines are deployed with a few security flaws to train and educate people against those vulnerabilities. First, you hack the system then you know what the flaws and/or misconfigurations are so you can defend properly in the future. Hacking is gamified in a way because those machines do not actually have a real website that real people use. They are used to show security flaws and what might happen if we do not fix them. Image steganography is widely used on those virtual machines to deliver a secret hint, hide a file or a password in an image.

## Least Significant Bit (LSB)

LSB is one of the most common image steganography methods and one of the simplest ones. It is used to embed a secret message by using the least significant bit. In an image file, the last bits of the RGB values of a pixel are simply changed to hide data.



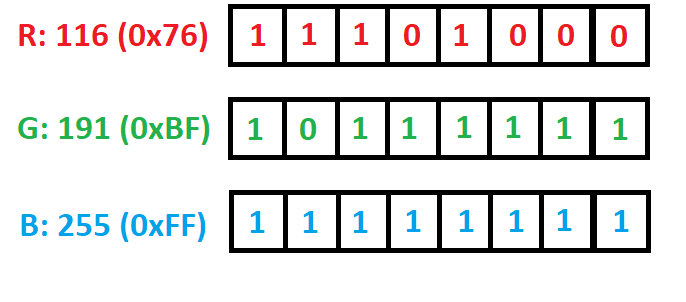
*Fig.1* The word "BOUN" written with Calibri Font



*Fig.2* Figure 1 Zoomed In

As you can see above, all images that are stored digitally, consist of pixels. Those pixels hold the Red, Green, Blue values (RGB). Usually, those color levels are stored from 0 to 255 in 8 bits. (00 and FF in hexadecimal notation.) Thus, computer knows which color that pixel has. When those pixels get together, the image is created.

Let's break down how our secret data will be stored in the image, without any significant change to eye.



*Fig.3* RGB Values into Bits

Image above shows RGB values converted into binary for only one pixel. Fortunately, images have hundreds or thousands of them.

If we take the 6 following pixels next to our chosen one, we get:

[(116, 191, 255), (255, 255, 255), (255, 255, 255), (255, 255, 255), (223, 156, 72), (0, 0, 0) (0, 0, 0)]

If we write them in binary:

116 = 11101002

196 = 11101002

255 = 11111112

255 = 11111112

255 = 11111112

255 = 11111112

**-**

**-**

**-**

0 = 000000002

Now it's time to hide our secret message. "Hi" will be the message we will hide into those pixels.

If we convert it into binary (UTF-8 encoded):

H = 010010002

i = 011010012

After that we will change the least significant bits in pixels.

Take 116 = 1110100, last bit here is the least significant bit. If we take one bit from the letter H's binary value and swap it with the pixel's least significant bit, we will have:

H = 010010002

First pixel,

116 = 11101002 (remains unchanged)

197 = 11101012 (changed)

254 = 11111102 (changed)

Second pixel,

254 = 11111102 (changed)

255 = 11111112 (remains unchanged)

254 = 11111102 (changed)

Third pixel,

254 = 11111102 (changed)

255 = 11111112 (remains unchanged)

254 = 11111102 (the letter "i" will start from this pixel's blue value)(changed) i = 011010012

Fourth pixel,

255 = 11111112 (remains unchanged)

255 = 11111112 (remains unchanged)

254 = 11111102 (changed)

Fifth pixel,

223 = 110111112 (remains unchanged)

156 = 100111002 (remains unchanged)

72 = 10010002 (remains unchanged)

Sixth pixel,

255 = 11111112 (remains unchanged)

255 = 11111112 (we do not need those values, we have "i" letter stored already)

255 = 11111112 (we do not need those values, we have "i" letter stored already)

As a result, with only 6 pixel's RGB values, we could store "Hi" word and we just changed 6 pixels' 9 color values by 1 and left the remaining 9, as it was. 50% of the values are unchanged.

Let us see the new colors.

*Fig.4* Pixels that will be changed. *Fig.5* Pixels that are changed

As seen, there is no visible change to the human eye and it usually depends on the image and how many bits are changed. With the help of this method, lots of data can be stored with low computational time complexity. The downside of this method is that it is fairly easy to detect with existing steganography analyzing algorithms.

# Description

Python and image module pillow (PIL) is used. It is a Windows application with a graphical interface that is created with pyqt5. Pyqt5 converts any python script file (.py file) into windows executable file by packing necessary python and modules into one file.

It is deployed as windows executable file (with .exe extension). Python script is kept for other use cases. (Python script for MacOS, Linux).

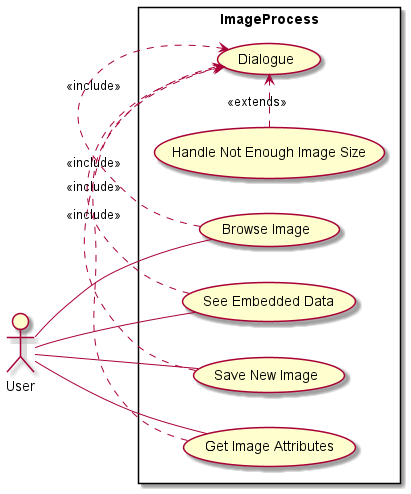
BMP and PNG extensions are supported at the moment.

# Requirements

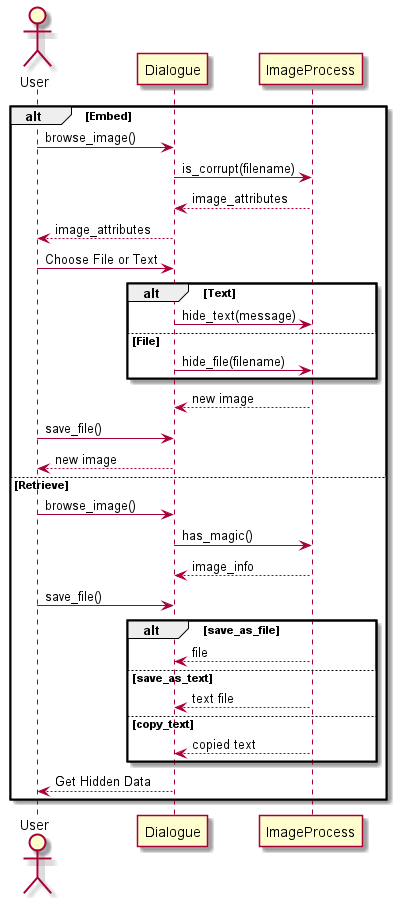
|  |  |
| --- | --- |
| **No.** | Functional Requirement |
| 1 | User will choose an image file to work on. |
| 2 | User will choose a text or a file to embed into the desired image file. |
| 3 | Application will create the new image with the hidden text or the file and user will save it wherever they want. |
| 4 | Application will tell the user if the file or text input size is small enough to be stored in the chosen image. |
| 5 | After embedding the data into a newly created image, user can extract the image with the same program. |

|  |  |
| --- | --- |
| **No.** | Non-Functional Requirements |
| 1 | Application shall be a Windows executable. |
| 2 | It shall have a simple graphical interface. |
| 3 | User will choose images with BMP or PNG extensions to work on. |
| 4 | As the programming language Python will be used. |
| 5 | Application shall be scalable and maintainable. |
| 6 | It shall run in any Windows 10 device after necessary modules are installed. |
| 7 | Application will show some simple information about chosen image. |
| 8 | It will be a desktop application. (With a graphical interface.) |

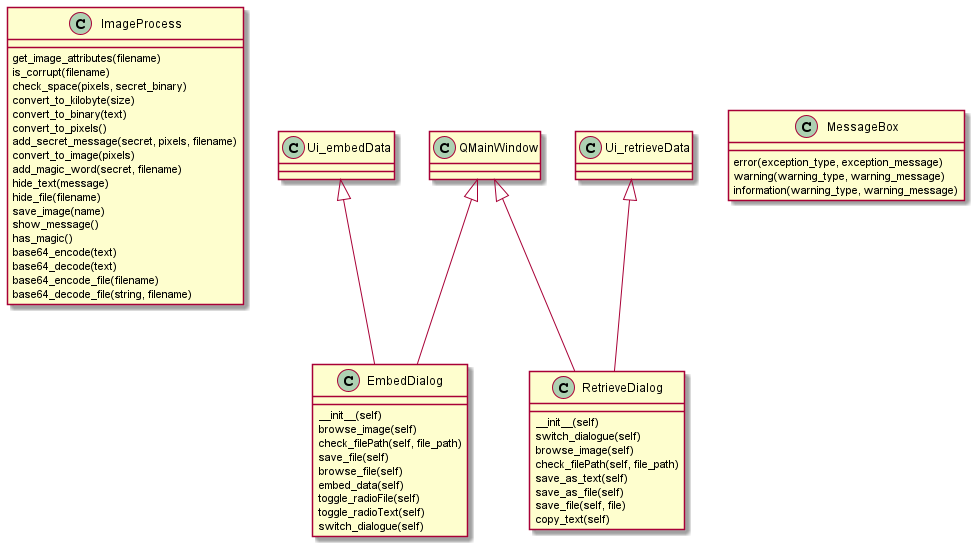
## Use Case Diagram



## Sequence Diagram



## Class Diagram



## Interface

