ICT2202 Digital Forensics

User Manual

Digital Steganography

Basil & Co.

Name	**Student ID**
LESLIE CHIEW	1802004
LIM JIN CHENG BASIL	1801674
WONG WEI WEI	1801437
CHUA WEI YE	1801554
CHUA WOI ZHAO	1801961
DANIEL TAN EN ZHI	1800601

Contents

1. Prerequisite

1.1 PyCharm Interpreter

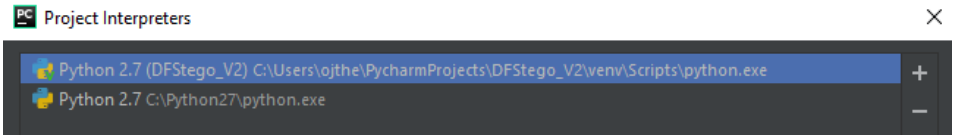

Figure 1: PyCharm Interpreters required

To run the program, please ensure Python 2.7 is set as the application interpreter and an accompanying Python 2.7 virtual environment is set for the project. (Figure 1)

1.2 PyCharm Packages in Interpreter

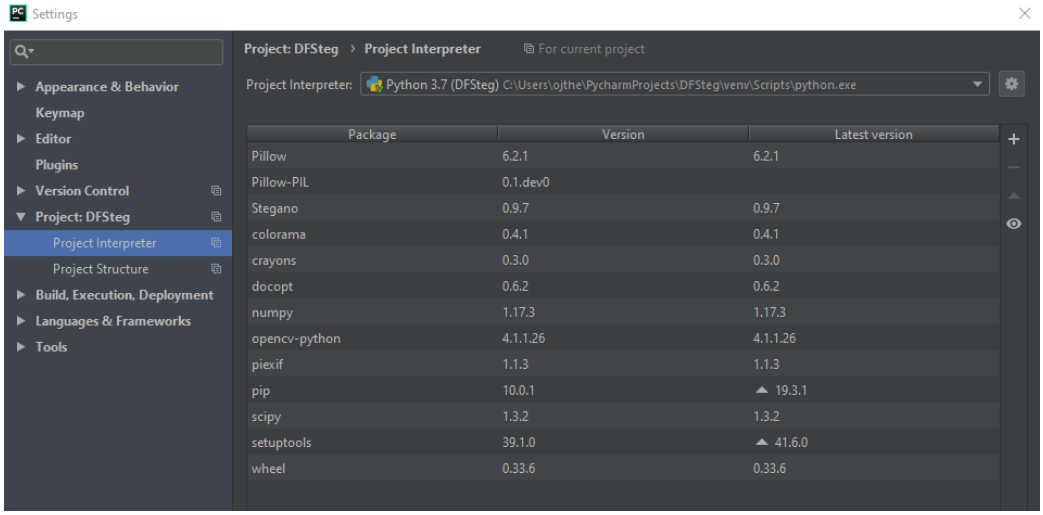

Figure 2: PyCharm Packages installed for the interpreter

The following is a table of the required packages that need to be installed on the Python 3.7 interpreter. (Figure 2)

Package	Version	Latest Version
Pillow	6.2.1	6.2.1
Pillow-PIL	0.1.dev0	-
Stegano	0.9.7	0.9.7
colorama	0.4.1	0.4.1
crayons	0.3.0	0.3.0
docopt	0.6.2	0.6.2
numpy	1.17.3	1.17.3
opencv-python	4.1.1.26	4.1.1.26
piexif	1.1.3	1.1.3
scipy	1.3.2	1.3.2
wheel	0.33.6	0.33.6

1.3 PHP must be installed

Upon installing php, the php.exe path must be set at quickStego.py file

```
subprocess.check_output(["Set PHP.exe file path here","test.php",inputFile])
```

2. Running the Application

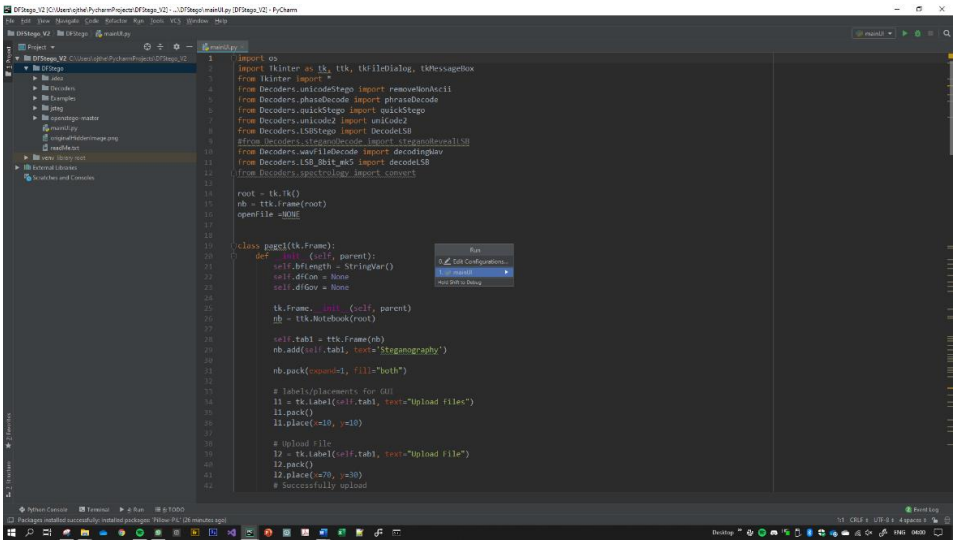

Figure 3: Running the program through PyCharm

To start the program, run the “mainUI.py” file using the Python 3.7 interpreter as mentioned in *Figure 1* and *Figure 2. (Figure 3).* Once started, a Tkinter GUI window will appear on screen. The window is the main interface of the program. (Figure 4)

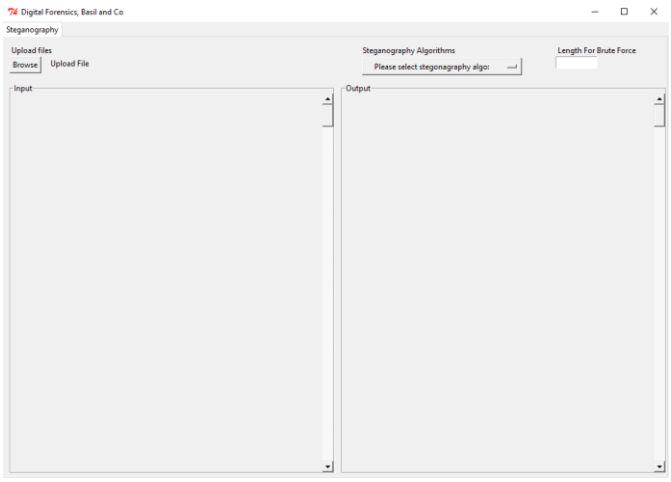

Figure 4: Tkinter GUI for the program

3. Uploading a File

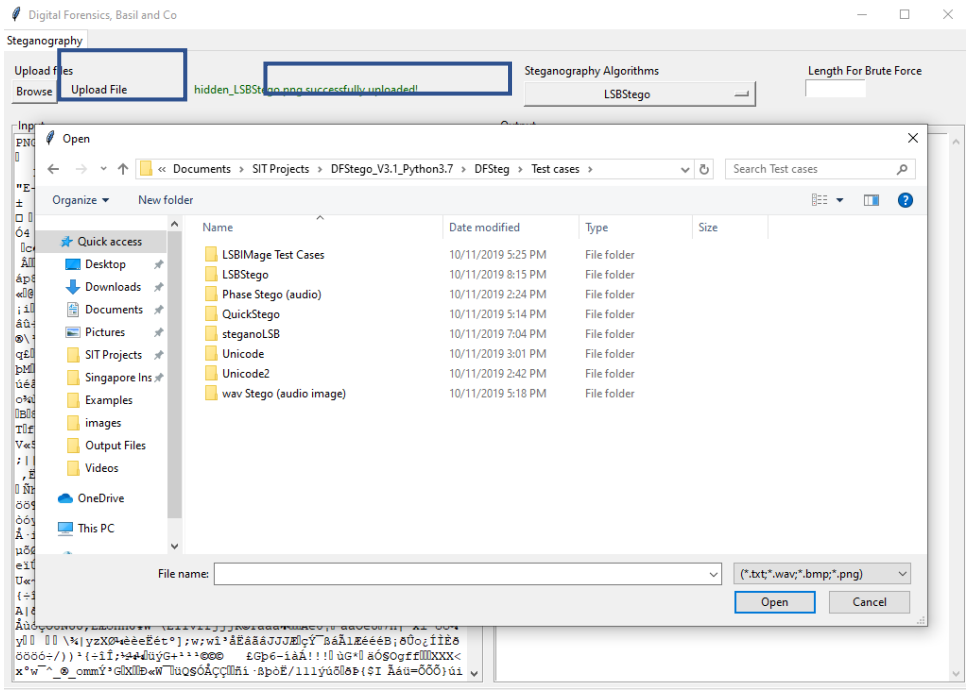

Figure 5: Selecting a file through File Explorer

To upload a file into the program, select “Browse” under the upload files field. A pop-up Windows Explorer will appear. Navigate to the desired file and select “open”. Once selected, a confirmation will appear on the Tkinter window stating the uploaded file’s

4. Selecting a Steganography Algorithm

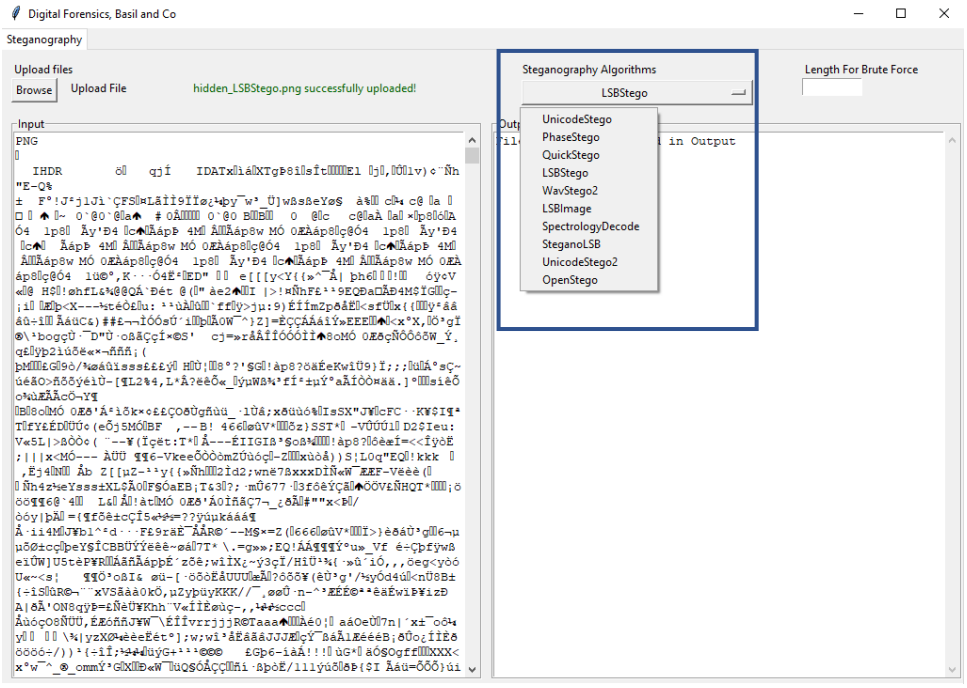

Figure 6: Selecting a Steganography Algorithm

To select a steganography function to perform on the uploaded file, select “Steganography Algorithms”. The program will then list all supported decode functions.

5. Steganography Algorithms Summary

1. UnicodeStego

Unicode steganography is a method that hides data in text using zero-width characters. As zero width characters are not removed if the formatting is stripped, making them nearly impossible to get rid of without re-typing the text or using a special tool.

2. PhaseStego

PhaseStego is a tool which hides text message in an audio .wav file. Rather than adding noises, this technique encodes the secret data bits to phase shifts in the phase spectrum of the audio signal/data, able to obtain inaudible encodings in terms of signal-to-noise ratio. This technique alters the phase of an initial audio segment and substitute with a reference phase that represents the data. Following segments phase is modified back to maintain the relative phase between segments.

Firstly, the decoder will split the encoded .wav file into signal/data and rate. It will then uses ifft(Inverse fast Fourier Transform) on the signal data. Using brute force on the length of message, the decoder will then compare each 8bits length with the signal data and convert them back to strings using ascii.

3. QuickStego

QuickStego is a tool that hides unencrypted text data in BMP image files. The picture used can be of .bmp, .jpeg, .jpg or .gif. The picture saved after steganography will always be a .bmp file. The program adds a header (QCE_S message) and a footer (end message) to indicate the start and end of the message. They use LSB algorithm to hide the text within the pixels of the picture. Every 3 pixels (RGB,RGB,RG) of each column stores 1 character of the text.

The decoder compares the RGB of the first pixel and use the resultant to compare the second pixel's RGB and uses the resultant of the second pixel to compare with the third pixel's RG bitwise values using an AND operation then a OR operation to compare the bits and find which bits are set and clear. The result after the 3rd pixel's comparison is stored as an int value and converted into a character. This continues until the string that has been decode contains a string "end message" which will stop the decoder as quickStego uses "end message" as a footer to mark the end of the string they encode.

4. LSBStego

LSBStego module is based on OpenCV to hide data in images. It uses the first bit of every pixel, and every colour of an image. If every first bit has been used, the module starts using the second bit, so the larger the data, the more the image is altered. However, LSBSteg will be able to hide larger data than the image.

5. wavStego2

wavStego2 is a tool that hides audio files .wav into an image .png by using the R,G,B logic of the color value of the image of each pixel. The decoder will take the image file, and convert it to RGBA values, and take each pixel of the image to check if the sum of RGB is divisible by 3, append 1 to the binary if yes and append 0 to the binary if no. After deriving the binary values, convert the binary data back to string, and save it as output.wav.

6. LSBImage

LSBImage extracts the least significant bit (LSB) from each 8-bit colour channel of every pixel in an image. The extractor concatenates all the LSBs starting from top to bottom and proceeding left to right. The function will then form the original encoded secret file from the extracted LSB binary values and save the output.

7. UnicodeStego2

UnicodeStego2 is an algorithm that combines both Word Shift and Unicode Zero-Width characters techniques in text steganography. Word shift technique converts the text to be hidden into their binary equivalent and stores bit '0' by shifting a word in the cover text slightly to the right, bit '1' is similarly stored by shifting a word slightly to the left. Unicode Zero-Width Characters technique make use of characters which have no width and are invisible in text files to hide information. UnicodeStego2 encoding is done by converting text to be hidden to binary and with a zero-width character (specifically U+200B) placed to the left of a word to represent bit '0' and to the right to represent bit '1'. Decoding is done by going through the text file and looking for all types of zero-width characters in Unicode and from their position relative to the word beside it extracting their bit values. The hidden text is then reconstructed by converting the binary to its Unicode character equivalent. One limitation for this technique is that the cover file has to be sufficiently large enough to encode the data to be hidden.

8. SteganoLSB

SteganoLSB is a python opensource library project which make use of LSB to hide messages in PNG files. The message will be scattered in the picture, following a set described by the Sieve of Eratosthenes. Other sets are available. You can also use your own generators. This will make a steganalysis more complicated.

YouTube Link: https://www.youtube.com/watch?v=d1mrZp_NXEA

GitHub Link: https://github.com/xiaoheihei3211/ICT2202Basil-Co/blob/master/README.md