



International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijirccce.com

Vol. 7, Issue 5, May 2019

Image Steganography Tool Using LSB Technique

Kaustubh Raval¹, Prarthana Fadia²

B. Tech Student, Department of Computer Engineering, Gujarat Technological University, Ahmedabad, Gujarat, India¹

Assistant Professor, Gujarat Technological University, Ahmedabad, Gujarat, India²

ABSTRACT: The main use of Steganography is to secretly hide a message into a source file and then use that encrypted source file as a source of communication. So, if the sender sends the file in a group, then only the targeted user can retrieve the message from the encrypted file using the decryption technique. Other users will find it as a normal image file, not knowing the actual purpose behind the communication. So, here is the implementation of a tool to perform text to perform image steganography. For that purpose, Python programming language is used, so the tool can be platform independent. The designing of this tool involves the use of LSB (Least Significant Bits) technique, because it does not affect the visibility property of the image. The pixels' LSB are replaced by the message bits. This tool will encrypt the secret message in the image by sender and then the targeted user can decrypt the message from the image.

KEYWORDS: Image steganography, LSB (Least Significant Bits), Steganography, Text to Image Stego.

I. INTRODUCTION

There are various ways to communicate. But in many scenarios, the communication needs to be done secretly, in a way such that only sender and receiver will know the real meaning of the information. Mostly, these types of scenarios are used by the military department, national security, secret business trading, the political parties, banks, etc. Steganography is the art of hiding the fact that communication is taking place by hiding secret information in other information and the encrypted information will be innocent to a normal user.

There are different types of steganography like Text Steganography, Image Steganography, Video Steganography, Email Steganography, Audio Steganography, etc. The main purpose of this project is to demonstrate the tool, which perform the image Steganography. In this steganography, the secret message is hidden in the image. The project is basically divided into two phases: - 1). Hiding secret data in an image file (Stego Image). 2) Extracting the data from the image file.

II. TECHNOLOGY AND LITERATURE REVIEW

RGB Image Structure-

The RGB image is the structure of a three-dimension array, which stores values of Red color, Blue color and Green color for each pixel. So, every pixel in an image has array values according to the colors for the respected pixels.

Least Significant Bits Technique in Image Steganography-

Least significant bits technique is a technique, which replaces the least significant bits of an image, so after replacing the bits, resulted image won't be much different. The least significant bits have less contribution in the overall file structure. As an example, there is a bit string of "1011 0011". If the LSB technique is applied and it replaces the bits. In that case suppose the resulted bit string will have "1011 0010" then the resulted file won't be much different compared to the source file.



International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 5, May 2019

Python Programming-

Python programming is used to create the tool with some libraries like PIL (Pillow) to manipulate image and edit image's LSB, NumPy to work with the arrays of the image, we are going to edit.

III. PROPOSED SYSTEM

Operating System Used: LINUX

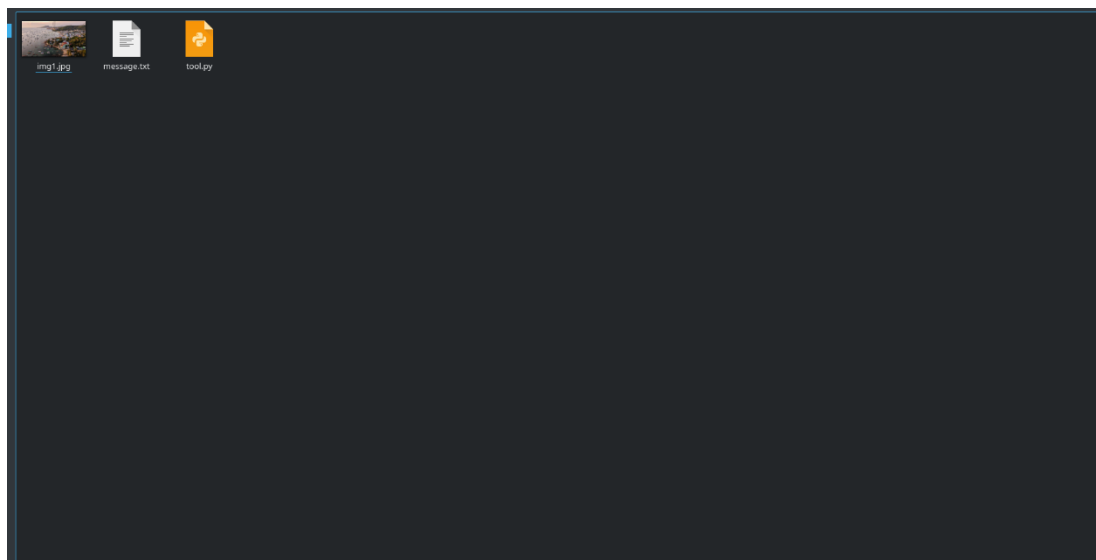
Language Used: Python

Internal logic of the tool:

1. The tool for Image Steganography.
2. First, open the image using PIL and the text file, convert the image to RGB format. Convert this text to the list of bits and encode them using the password.
3. After that, using NumPy add those bits of text to the LSB of the source image and merge that array to generate a new image with the text hidden inside.
4. This image can be sent to anyone and the normal user won't be able to notice anything. Moreover, it is very hard to notice anything without the source image.
5. To extract the text from the image, first convert the image to the array then extract the LSB of the image and decode it using the same password, which was given to encrypt the text. Then create a new file and save the text in that file.

Practical Implementation:

In the screenshot below, img1.jpg is the source image, which is used to hide the secret message. Message.txt contains the secret message and tool.py is the tool to perform the image steganography.



1. Initial files

Tool.py can be accessible from the terminal, so change the directory where the tool is located.

Run the command "python2 tool.py", it will prompt two modules.

It shows 'Usage %prog [option] [argument/imagefile]'

The above line is a format of the input for the tool.

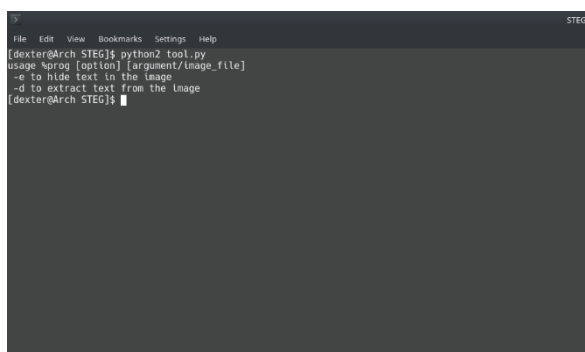
International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 5, May 2019

Use “-e” to hide the secret text in the image
Use “-d” to extract the text from the stego image



```
File Edit View Bookmarks Settings Help
[dexter@Arch STEG]$ python2 tool.py
usage *prog [option] [argument/image_file]
-e to hide text in the image
-d to extract text from the image
[dexter@Arch STEG]$
```

2. Usage of tool.py

For the first option: “e” which hides the text file to the source image.

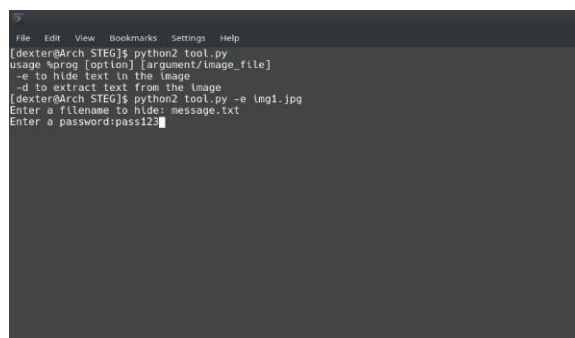
Use the command “python2 tool.py -e img1.jpg”



```
[dexter@Arch STEG]$ python2 tool.py -e img1.jpg
```

3. Option “-e” for the tool

It will prompt and ask to enter the secret message file and password, see the screenshot below.



```
File Edit View Bookmarks Settings Help
[dexter@Arch STEG]$ python2 tool.py
usage *prog [option] [argument/image_file]
-e to hide text in the image
-d to extract text from the image
[dexter@Arch STEG]$ python2 tool.py -e img1.jpg
Enter a filename to hide: message.txt
Enter a password: pass123
```

4. Text filename and the password to encrypt

After entering the text file name and the password, tool will hide the secret message in the source image and the terminal will prompt that the message is successfully hidden in the source image, as shown in the image 5.



International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 5, May 2019

```
File Edit View Bookmarks Settings Help
[dexter@Arch STEG]$ python2 tool.py
usage: sprog [option] [argument/image_file]
-e to hide text in the image
-d to extract text from the image
[dexter@Arch STEG]$ python2 tool.py -e img1.jpg
Enter a filename to hide: message.txt
Enter a password:pass123
-> message.txt hidden successfully in the source file!
[dexter@Arch STEG]$
```

5. Successful encryption

For the second option “-d”. Use the command “python2 tool.py -d stego.png”. At the time of using this command, terminal will ask for the password. Now, the user needs enter the same password, which was used at the time of encryption or the tool will not generate correct secret message. It will also ask for the name of output file for the secret message.

```
File Edit View Bookmarks Settings Help
[dexter@Arch STEG]$ python2 tool.py -d stego.png
Enter the name of text file to save the extracted data:output.txt
Enter the password to decode:pass123
-> Data is extracted from stego.png to output.txt
[dexter@Arch STEG]$
```

6. Extraction of the hidden message

So, it can be seen from the screenshot above that the secret message has been extracted successfully and the output has been stored to output.txt.

International Journal of Innovative Research in Computer and Communication Engineering

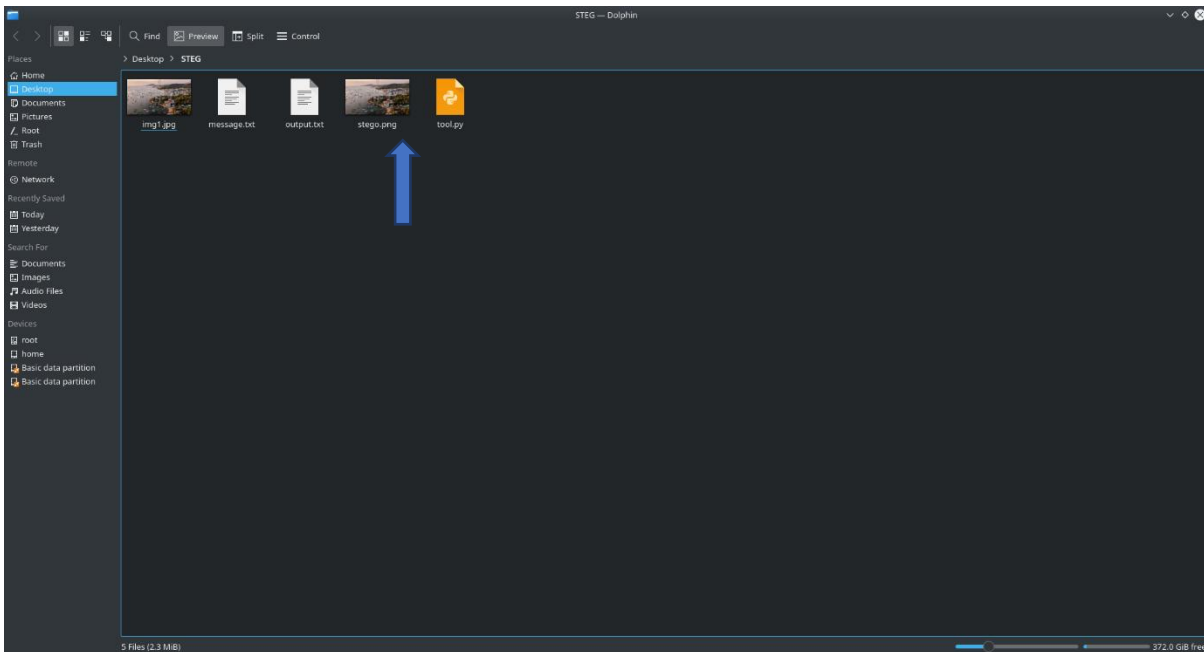
(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 5, May 2019

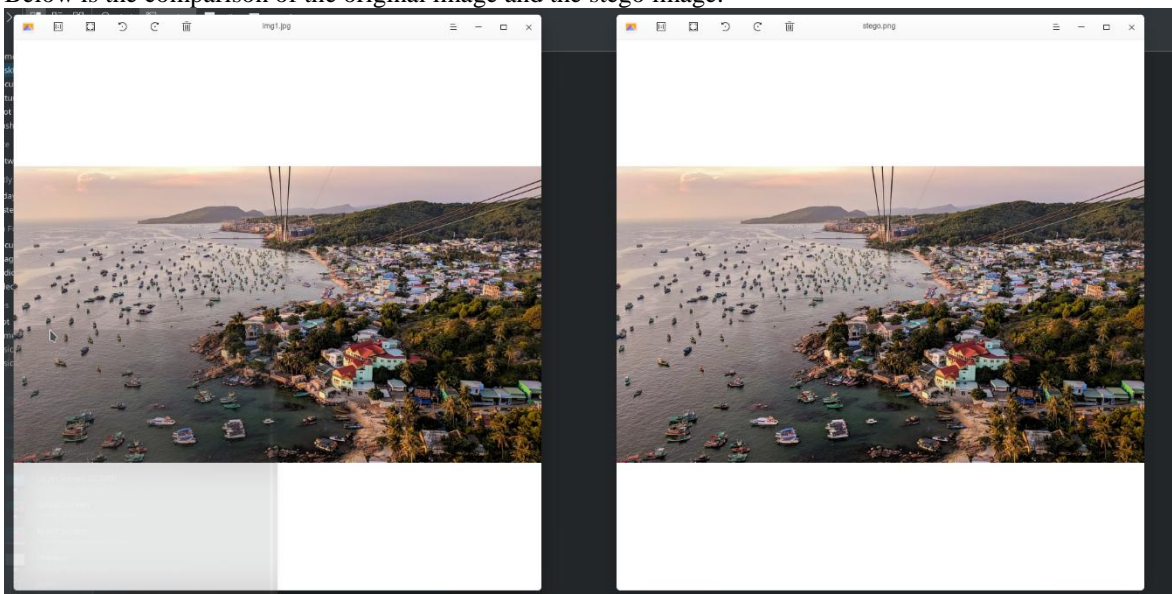
IV. RESULTS

The output file is output.txt, which contains secret message. It is created in the same folder as tool.py



7. Files after the extraction process

Below is the comparison of the original image and the stego image.



8. Comparison of original and stego image



International Journal of Innovative Research in Computer and Communication Engineering

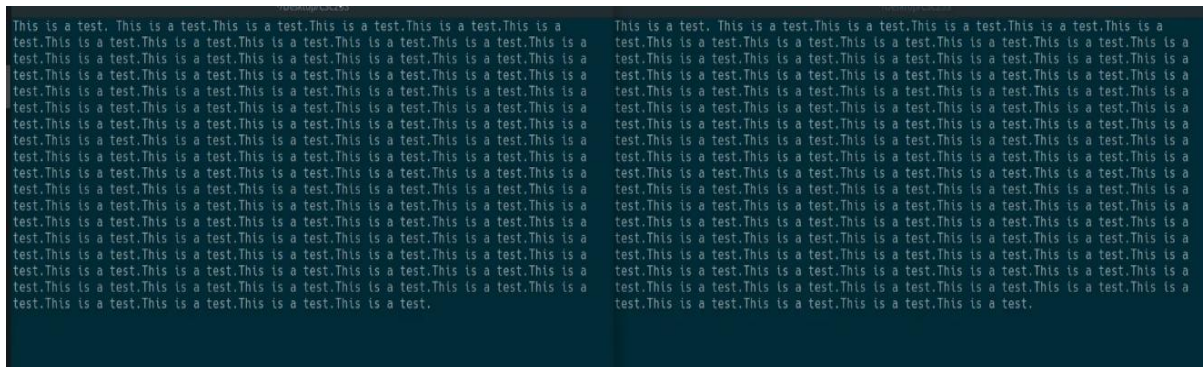
(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijirce.com

Vol. 7, Issue 5, May 2019

By just looking at these 2 images normal user can not notice any difference. But the difference can be noticed by comparing histograms of these two images.

Below is the comparison of the original file message.txt (left image) and the extracted file output.txt (right image).



9. Comparison of original message.txt and output.txt

V. CONCLUSION

This paper presents an overview of steganography and the tool of Steganography written in Python program that embeds the message in the source image. The technique of Steganography has evolved rapidly with the development of technology in recent years. Many forms of Steganography exist for hiding messages from image, video, and sound to text. Taking all this into consideration, this paper has developed the program for embedding data into image. It is interesting and less detectable because message bits are embedded into image pixels' LSB values. Future work involves continuing to develop this software by providing new features to hide data and then compare it with the source file.

REFERENCES

1. Ravi Kumar, Kavita Choudhary, Nishant Dubey, "An Introduction of Image Steganographic Techniques and Comparison", International Journal of Electronics and Computer Science Engineering.
2. Provos, N. & Honeyman, P., "Hide and Seek: An introduction to steganography", IEEE Security and Privacy Journal, 2003.
3. M. Wu, E. Tang, and B. Lin, Data hiding in digital binary image, Proc. of 2000 IEEE International Conference on Multimedia and Expo, vol. 1, pp. 393-396, 2000.
4. Kavitha, Kavita Kadam, Ashwini Koshti, Priya Dughav, "Steganography Using Least Significant Bit Algorithm", International Journal of Engineering Research and Applications (IJERA), vol. 2, no. 3, pp. 338-341, 2012.
5. R. Chandramouli ; N. Memon, "Analysis of LSB based image steganography techniques", Proceedings 2001 International Conference on Image Processing, 2014.
6. J. Fridrich and M. Goljan, On estimation of secret message length in lsb steganography in spatial domain. Proc. of IST/SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents VI, vol. 5306, pp. 23-34, 2004.
7. J. Zhang, I. J. Cox, and G. Doerr, Steganalysis for lsb matching in images with high-frequency noise, Proc. of IEEE Workshop on Multimedia Signal Processing, pp. 385-388, 2007.
8. Jamil, T., "Steganography: The art of hiding information is plain sight", IEEE Potentials, 18:01, 1999.
9. N. F. Johnson and S. Jajodia, "Exploring steganography, seeing the unseen," IEEE Computer Magazine, vol. 31, no. 2, pp. 26-34, February, 1998.
10. T Morkel, J.H.P Eloff, M.S Olivier, "AN OVERVIEW OF IMAGE STEGANOGRAPHY". Proceedings of the Fifth Annual Information Security South Africa Conference (ISSA2005), 2005.
11. Shashikala Channalli, Ajay Jadhav, "Steganography An Art of Hiding Data", International Journal of Computer Science and Engineering, vol. 1, no. 3, pp. 137-141, 2009.
12. Manu Devi, Nidhi Sharma, "Improved Detection of Least Significant Bit Steganography Algorithms in Color and Gray Scale Images", Proc. IEEE RAECs UIET Panjab University Chandigarh, March 2014.
13. M. Juneja and P. S. Sandhu, "Designing of robust image steganography technique based on LSB insertion and encryption," presented at the International Conference on Advances in Recent Technologies in Communication and Computing (ARTCOM-2009), Kerala, India, October 27-28, 2009.