COMPARATIVE ANALYSIS OF IMAGES BASED ON LEAST SIGNIFICANT BIT (LSB) STEGANOGRAPHY

Supported by

Florence A. Oladeji[1], Oluwayomi Awe[1]*, Taye O. Aro[2], Adenrele A. Afolorunso[3], A. Ibor[4] and Charles O. Uwadia[1]
[1]Department of Computer Science, University of Lagos, Nigeria
[2]Department of Mathematical and Computing Sciences, KolaDaisi University, Ibadan, Oyo State Nigeria
[3]Department of Computer Science, National Open University of Nigeria, Abuja, Nigeria
[4]Department of Computer Science, University of Calabar, Nigeria
*Corresponding author: aweyomi@gmail.com

Received: August 31, 2020 **Accepted:** September 29, 2020

Abstract: Steganography helps to conceal information between the sender and intended recipient so that others are not aware of the message presence, largely, it is to hide information. Digital steganography applies to different media like executable files, images, audio and videos, text and games. Steganalysis aims to check the presence of hidden data in the suspected image file. This paper applied some techniques in LSB algorithms to analyse four major open sources steganography tools (rSteg, OpenStego, ImageStegano and StegTool) by analysing their performances based on the stego size and extraction time taken and also, extracting out their hidden text. The result shows that OpenStego performs better irrespective of the size of the hidden data and time taken.

Keywords: Forensic analysis, steganography, least significant bit, steganalysis

Introduction

Steganography is the art and science of shielding information in documents, images, audios and videos, text and networks so that its existence will not be suspected by the untargeted receiver (Emam *et al.*, 2016). The covered image where the message is hidden is called STEGO which helps to eliminate or reduce suspicious. Steganography is most valuable where human rights are been hampered, in military, anti-forgery and so on. Steganalysis simply unravels the existence of a concealed message (Karte & Bharti, 2017). It helps the security analyst, forensic experts to keep track of covered messages (Mandal, 2012). Steganography algorithm that is universally used in research is Least Significant Bit (LSB) algorithm (Amritpal & Harpai, 2015). LSB optimum performance is when the length of the hidden message is less than the length of the covered medium which can be 8-bit and 24-bit. For 24 bit image, three bits are encoded into each pixel to get the message hidden.

Image steganalysis involves the hiding of data inside cover images for security (Sravanthi *et al.*, 2012). Images possess a lot of visual redundancy because our eyes do not usually consider subtle changes in colour in an image region. One can use this redundancy to hide text, audio or even image data inside cover images without making significant changes to the visual perception (Rathika *et al.*, 2017). Image steganography is becoming popular on the internet these days since a stenographic image, which just looks like any other image, attracts a lot less attention than an encrypted text and a secure channel (Fatnassi *et al.*, 2016).

Several image steganalysis techniques have been employed such as least significant bit pixel value differencing, histogram-based, texture-based, spread spectrum based, labelling or connectivity method and so on (Tiwari *et al.*, 2014). Among the image steganography techniques, LSB remains the most common and easiest approach for message hiding (Devi, 2013). In this method, the message is hidden in the least significant bits of image pixels. Also, changing the LSB of the pixels does not introduce much difference in the image and thus the stego image looks similar to the original image (Dumitrescu *et al.*, 2003).

This paper examined four different open source LSB steganalysis tools: Rsteg, OpenStego, Image-Stegano and StegoTool respectivelyand taken into consideration the security domain. It has been identified that all the tools cannot detect other hidden messages despite that they are using generic LSB algorithms. Analysed tools discussed are prone to attack despite having a secret key. Using the Digital Forensic Investigation Model (Collection) to extract embedded data.

Steinebac *et al.* (2019) evaluated how to address the problems with traditional steganographic and statistical methods, rather than applying high-performance computing and machine learning. The system analyzed the F5 steganographicalgorithm which is usually detected using statistical anomalies caused by message embedding wasapplied to images with a high degree of diversity, as it has been seen in a typical social network. It was revealed that the biggest challenge lies in the detection of images whose payload is less than 50% of the available capacity of an image. A suggestion of new detection methods and application of these to the problem of channel detection in the social network. Ignatius *et al.* (2018) proposed enhanced LSB-image steganography using the hybrid Canny-Sobel edge detection. The system devised a method for increasing the payload of secret messages in an image. The edge area was used to accommodate more message bits because the image edge area can better tolerate pixel value changes. Also, Canny and Sobel's detectors were combined to get a wider edge area. The two-detector combined method provided a larger edge area for a greater payload of messages while maintaining imperceptibility of stego-images.

A novel blind statistical steganalysis approach was developed by Chaeikar *et al.* (2017). The technique was used to detect the Least Significant Bit (LSB) flipping image steganography. In the method, a new system of pixel colour correlativity analysis in Pixel Similarity Weight (PSW) was achieved with an extremely high-efficiency level of 98.049% in detecting 0.25 bpp stego images with only a single dimension analysis. An optimal steganalysis based approach for embedding information in Image Cover Media with Security was proposed by Fatnassi *et al.* (2016). A critical review of the steganalysis algorithms available to analyze the characteristics of an image stego media against the corresponding cover media and understand the process of embedding the information and its detection was conducted. It was envisioned and also a clear picture of the current trends in steganography was given so that we can develop and improvise appropriate steganalysis algorithms.

Moon and Raut (2013) applied the 4LSB substitution method for embedding a large amount of data behind the selected frame of video. The study used video as cover media for hiding the secret message and used computer forensics as a tool for authentication. The major objective is to hide an image and text behind a video file. A suitable algorithm such as 1LSB, 2LSB, 4LSB is used and the 4LSB method found to be good for hiding more secret information data. The main focus of the study deals with the idea of video steganography, cryptography and the use of computer forensic techniques in both investigative and secure manner.

Materials and Methods

The developed comparative analysis of images based on the Least Significant Bit (LSB) steganography involved several step-wise phases. Four image LSB open-source steganography tools; rSteg, OpenStego, ImageStegano and Steghide were employed. This study considered two major aspects; embedding and extraction process. The embedded Images were first supplied into the system, then further divided into overlapping blocks and the blocks were rotated using random angle 0, 90, 180, 360. A region was selected from the embedded rate of image in which estimation of the capacity of the region selected was taken, if the estimation is enough, the image will be embedded and re-block rotate to form stego, if not the threshold will be changed to estimate the capacity of region selected again. The extraction process focused on the input of stego image of block size B_Z with threshold T. The stego images were divided by block size B_Z then rotated applying angle 0, 90, 180, 360. Pixels with greater or equal to threshold value were chosen; this made extraction to be achieved. The flowchart of the LSB steganography system is shown in Fig. 1.

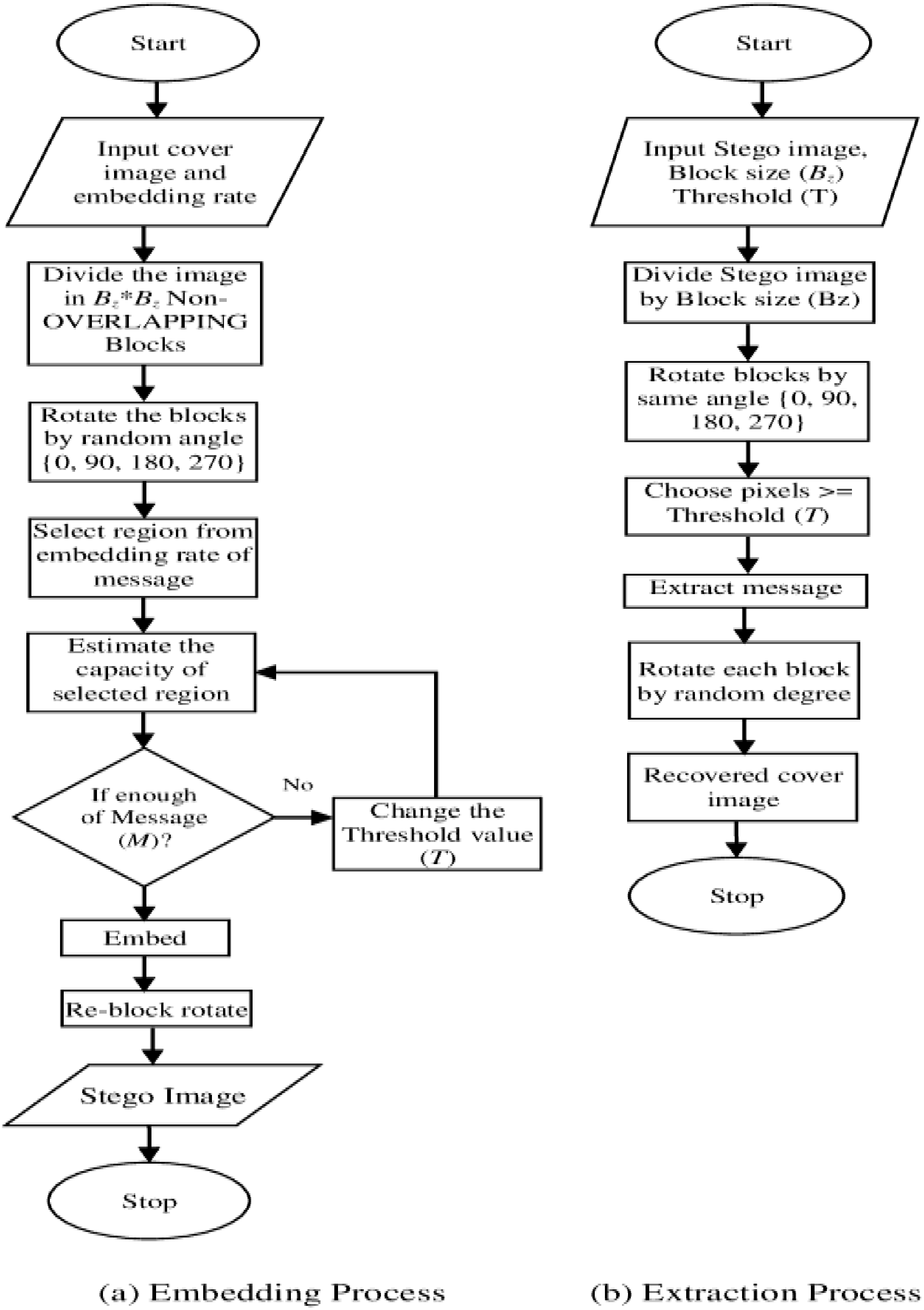
Start
Input cover image and embedding rate
Divide the image in B_z*B_z Non-OVERLAPPING Blocks
Rotate the blocks by random angle {0, 90, 180, 270}
Select region from embedding rate of message
Estimate the capacity of selected region
If enough of Message (*M*)?
No
Change the Threshold value (*T*)
Embed
Re-block rotate
Stego Image
Stop
Start
Input Stego image, Block size (B_z) Threshold (T)
Divide Stego image by Block size (Bz)
Rotate blocks by same angle {0, 90, 180, 270}
Choose pixels >= Threshold (*T*)
Extract message
Rotate each block by random degree
Recovered cover image
Stop
(a) Embedding Process
(b) Extraction Process

Fig. 1: Flowchart of the developed LSB steganography system

Metric for comparison

The image type.PNG is generic to all the tools, which was used as a cover image. The text data to be hidden was varied from 0.021, 1.5, 14.5 and 90.2 KB were wrapped with an image to get a stego output. The sizes of the files were noted and likewise the time it takes to wrap data with the cover image.

Dataset

Table 1 provides a basic insight into the popularity and accessibility of the five open source tools being considered. The number of downloads is taken as at 20/04/2019. The sample of the covered image is shown in Fig. 2 with different parameters.

Table 1: Open source tools

S/N	Software	Supporting Files	Downloads	Resource location
1	rStego	All file types	4,323+	https://github.com/akush/rSteg
2	OpenStego	All file types	145,235+	https://www.openstego.com/
3	Image-Stegano	.bmp & .png	**8,676 +**	https://sourceforge.net/projects/image-steg/
4	StegTool	.bmp, .png& .jpeg	19,550+	http://steghide.sourceforge.net/index.php

Fig. 2: Sample of the cover image

Image Size = 13.6 KB

The embedded text is “This is hidden text” which is 21 byte

Results and Discussion

Results of open source tools

The entire open source tools (rSteg, OpenStego, Image-Stegano and StegTool) under consideration accept. PNG, Fig. 2 shows the image size 272 Kb. Text files of various sizes were tested with the tools to check their sizes after wrapping the text and also the time taken to get the hidden data extracted. Table 2 shows sample text files and their capacity after embedding.

The cover image was embedded with a text file of sizes (0.021, 1.5, 14.2 and 90.4 KB) to have a pattern for each embedding tools. While the output image (Stego) increases as the text data increases, it is of note that the OpenStego sizes remain the same irrespective of the increase sizes.

Table 2: Sample of text files and capacity after embedding

Embedding		Stego (PNG) size in Kb			
Text	**SizeKB**	**rSteg**	**OpenStego**	**ImageStegano**	**StegTool**
Sample1	0.021	272	539	272	272
Sample2	1.5	225	539	275	274
Sample3	14.2	234	539	298	289
Sample4	90.4	314	539	322	324

Results of embedded files

Table 3 shows the result obtained for text files embedded and their respective time taken to extract out the underline hidden data.

Table 3: Extraction time (seconds)

Embedding		Extraction Time Taken Seconds			
Text	**SizeKB**	**rSteg**	**OpenStego**	**ImageStegano**	**StegTool**
Sample1	0.021	36.68863	1.19095	27.58256	0.06686
Sample2	1.5	87.99302	1.23176	39.69757	369.77911
Sample3	14.2	255.73523	1.29530	20.51148	391.98621
Sample4	90.4	Time out	1.44962	12.43359	381.05623

From Table 3, the rSteg steganalysis tool increases with the increase in stego size, but at higher stego size, it started timing out. The StegTool took a long time to unravel the hidden text files.

Comparative Analysis of Results

Four open-source tools have been applied to investigate and evaluate the proposed image steganalysis system. The comprehensive results are shown in Table 4, while the graphical interfaces are shown in Figs. 3, 4, 5 and 6, respectively.

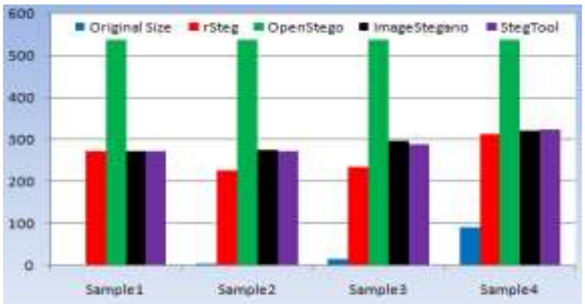

Fig. 3: Histogram representation of the tools sizes after embedding process

Table 4: Summary of the Four Open Sources

Embedding		Stego (PNG) KB				Extraction Time Taken Seconds			
Text	SizeKB	rSteg	Open Stego	Image Stegano	Steg Tool	rSteg	Open Stego	Image Stegano	Steg Tool
S1	0.021	272	539	272	272	36.6886	1.1909	27.5825	0.0669
S2	1.5	225	539	275	274	87.9930	1.2318	39.6975	369.7791
S3	14.2	234	539	298	289	55.7352	1.2953	20.5114	391.9860
S4	90.4	314	539	322	324	Time Out	1.4496	12.4335	381.0562

S1 = Sample 1; **S2** = Sample 2; **S3** = Sample 3; **S4** = Sample 4

The cover image was embedded with a text file of sizes (0.021, 1.5, 14.2 and 90.4 KB) to have a pattern for each embedding tools. It is of note that:

(i) **rStego:** The size of stego (embedded text file with cover image) increases as the data size to be hidden increases. At above 15 KB of embedded data, the time taken to embed increases tremendously which took over 30 min to get a data of size 90.4 KB to be embedded.

(ii) **OpenStego:** This is very unique in the handling of embedding data. Irrespective of the size of the text file, the stego size is unchanged and likewise, the embedding rate is fast for all various sizes. The average size of a stego image in OpenStego is 539Kb.

(iii) **ImageStegano:** The embedding rate is fast with a lesser text file but decreases as the embedding data to hide increases. The overall output shows that the stego size also increases with an increase in file size.

(iv) **StegTool:** The stego size increases with an increase in sizes of the steganography data. Just like rSteg, at above 15Kb of steganography data, the embedding rate became slow.

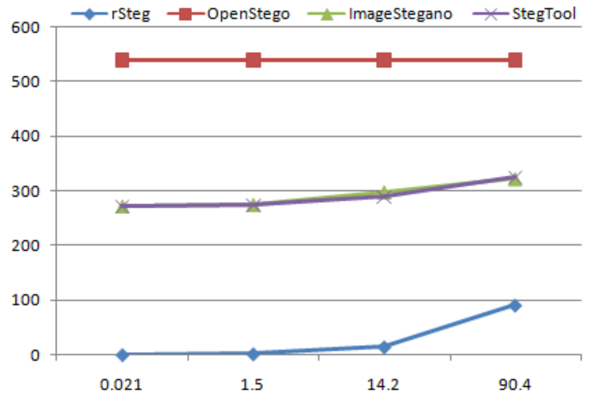

Fig. 4: Graphical representation of the sizes

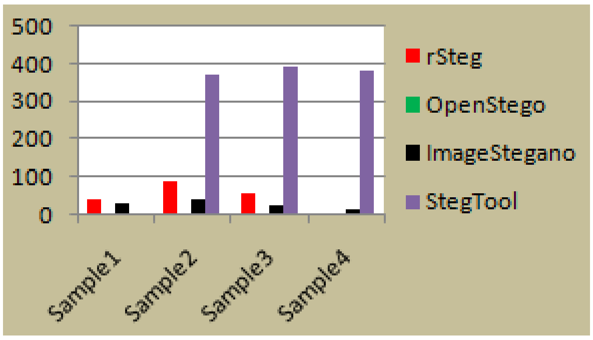

Fig. 5: Graphical representation of histogram for extraction time

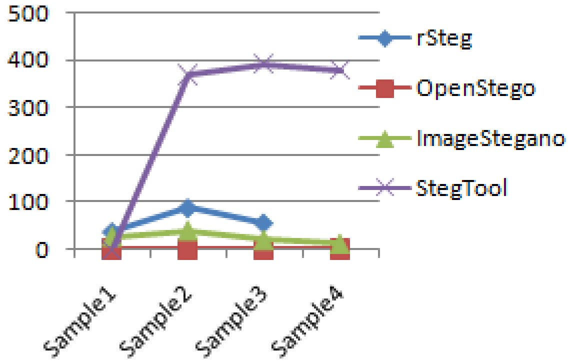

Fig. 6: Graphical representation of extraction time

Summary of Findings

In this study, a total of 4 open source tools have been critically examined and thoroughly evaluated which lead to various identified weaknesses. All the tools used the same base which is LSB, but a more enhanced LSB algorithm can be used to extract out the hidden messages. The generalised attack against end-of-file can easily be detected in the rSteg, OpenStego and StegoTool. It is Image-Stegano that is more secured than others because of the Edge Least Significant Embedding. Table 5 gives the file summary for the four tools considered.

Table 5: File summary of the four open sources

S/N	Tool Name	File types	Output format	Stego size	Text size
1	Rsteg	.jpeg, .png&.gif	Only .png	235Kb	21Bytes
2	OpenStego	All file types	Any file types	539Kb	21Bytes
3	Image-Stegano	.png&.bmp	.png& .bmp	539Kb	21Bytes
4	StegTool	.bmp, .jpeg & .png	.bmp, .jpeg & .png	538Kb	21Bytes

Conclusion

This paper showed how insecure the LSB algorithm and its variants are in steganography, although it has been identified by researchers that visual recognition is not enough to detect the presence of hidden messages. With these elaborate findings, it will be of great use to forensic experts. It is of a vital point to note that steganography has its good side for example whistle-blowers, journalist, broadcasting, military and so on. Presenting the various open-source steganalysis results, which show the lapses in those tools which are good resources for forensic examiners on image files. We recommend that more research should be conducted LSB model for steganalysis to extract out the hidden messages. From the finding, it is therefore recommended that there should complete restriction of the use of steganography tools from the citizen because of an increasingly radical and technological inclined act from the terrorist.

Acknowledgement

A lot of appreciations to the Department of Computer Science CoNIMS Research Group for their constructive criticism on the thesis that gave birth to this paper. All efforts from co-authors are all welcome.

Conflict of Interest

Authors have declared that there is no conflict of interest reported in this work.

References

Amritpal A & Harpai S 2015. An improved LSB based image steganography technique for RGB images. *IEEE Int. Conf. on Electrical, Comp. and Communication Techn.*, pp. 1 – 4.

Chaeikar SS, Zamani M, Manaf AB & Zeki AM 2017. PSW Statistical LSB Image Steganalysis PSW Statistical LSB Image Steganalysis. *Multimedia Tools and Applications*. http://doi.org/10.1007/s11042-016-4273-6

Devi KJ 2013. *A Secure Image Steganography Using LSB Technique and Pseudo Random Encoding Technique*.

Dumitrescu S, Wu X & Wang Z 2003. Detection of LSB steganography via sample pair analysis. *IEEE Transactions on Signal Processing*, 51: 1995–2007.

Emam MM, Aly AA & Omara FA 2016. An improved image steganography method based on LSB technique with random pixel selection. *Int. J. Comp. Advan. Sci. and Applic.*, *7*(3): 361–366.

Fatnassi A, Gharsellaoui H & Bouamama S 2016. An optimal steganalysis based approach for embedding information in image cover media with security. *Int. J. Comp. and Information Engr.*, 10(6): 1245–1249.

Ignatius DR, Setiadi M & Jumanto J 2018. An enhanced LSB-image steganography using the hybrid canny-sobel edge detection. *Cybernetics and Information Technologies*, 18(2): 74–88. http://doi.org/10.2478/cait-2018-0029

Karte B & Bharti D 2017. Dynamic key-based LSB technique for steganography. *Int. J. Comp. Applic.*, 167(13): 9–14.

Mandal PC 2012. An extensive review of current trends in steganalysis. *Int. J. Advan. Res. in Comp. Engr. & Techn.*, 1(7): 215–220.

Moon SK & Raut RD 2013. Analysis of secured video steganography using computer forensics technique for enhance data security. *2013 IEEE Second Int. Conf. on Image Information Processing*, pp. 660–665.

Rathika L, Loganathan B, Scholar MP, Nadu T & Nadu T 2017. Approaches and methods for steganalysis – A survey. *Int. J. Advan. Res. in Comp. and Communic. Engr.*, 6(6): 433–438. http://doi.org/10.17148/IJARCCE.2017.6678

Sravanthi GS, Sunitha B, Riyazoddin SM & Reddy MJ 2012. A spatial domain image steganography technique based on plane bit substitution method. *Global J. Comp. Sci. and Techn. Graphics and Vision*, 12(15): 1–8.

Steinebach M, Liu H & Ester A 2019. The need for steganalysis in image distribution channels. *J. Cyber Security and Mobility*, 8(3): 365–392.

Tiwari A, Yadav SR & Mittal NK 2014. A review on different image steganography techniques. *Int. J. Engr. and Innovative Techn.*, 3(7): 121–124.