Steganography and Steganalysis in Digital Images

Marcus D. Cruz Vélez
Computer Engineering
Jeffrey L. Duffany, Ph.D.
Electrical & Computer Engineering and Computer Science Department
Polytechnic University of Puerto Rico

Abstract *—* ***Steganography*** *(from the Greek for "covered writing") is the secret transmission of a message. Its main distinction from encryption relies on the capacity of making a message altogether invisible. A steganographic message may also be encrypted, for extra security in the case of interception but it actually may not be needed.* ***Steganography*** *has an advantage, it becomes difficult to decipher more than encrypted messages even for a talented code-cracker, simply because it cannot decipher a message since it doesn't know it is there. This is when* **steganalysis** *come into place by detecting a hidden message in an image and identifying the algorithms employed in order to view these "secrets". We describe different* ***steganography*** *tools, how they work and examine which are more susceptible to be detected when analyzed.*

Key Terms *— Least Significant Bit (LSB), Steganography, Steganalysis, Stegogramme.*

INTRODUCTION

Steganography was considered an art since the beginning. Long before the computer era, hidden messages were sent undetectable. According to Greek history, a nobleman in order to communicate with his son-in-law shaved the head of his slaves and tattooed the message onto the slave's scalp.

Also Herodotus, Greek historian, describes how one of his cunning countrymen sent a secret message warning of an invasion by scrawling it on the wood underneath a wax tablet.

In 1870 during the Franco-Prussian War, Paris was under siege and messages were sent by carrier pigeon. In the Second World War, people used milk, vinegar, fruit juices and urine to write secret messages. When heated, these fluids become darker and the message could be read [1].

During the World War I and World War II the Germans developed the microdot technique, a technique in which a text or an image was substantially reduced in size onto a 1mm disc.

This reason for reduction was mainly to prevent detection by unintended recipients.

Cyber-crime is believed to benefit from this digital revolution [2], taking the **steganography** methods into a higher level. Cyber terrorism its taken advantage of the technology developments as reported in the USA TODAY [3, 4, 5].

As reported by writer Stuart Fox in TechNewsDaily, the FBI accused Russian spies for hiding secret codes in online photos [16]. Planning against these cyber actions is difficult to achieve.

Peter Honeyman and Niel Provos, at the University of Michigan examined over 2 million images from popular websites like Ebay auction images and USENET archives searching for **Steganography** traces.

They have detected multiple hidden messages but found none. The approached analysis is statistical, using different **steganography** detection tools that can identified if an image really contains a hidden message and reveal any hidden content.

WHAT IS STEGANOGRAPHY?

Steganography comes from the Greek word meaning "covered or concealed writing", it is also defined as the hiding of a message within another so that the presence of the hidden message is unnoticeable. The key concept behind **steganography** is to transmit a message that is not detectable to the casual eye. People who are not the intended recipients of the message should not even suspect that a hidden message exists. It is not even necessary for two parties to agree to a secret key beforehand [6].

Some might mistake **steganography** for cryptography or vice versa. The difference between these two is that in cryptography, one can tell that a message has been encrypted, but cannot decode the message without knowing the proper key. In **steganography**, the message itself may not be difficult to decode, but most people would not detect the presence of the message. Therefore even if not encrypted might be even harder to decode because is not easily detectable. **Steganography** is also known as the art of hiding information, data or messages. The advantage of **steganography** is that those who are outside the interaction even do not realize that some sort of communication is being done.

STEGANOGRAPHY HISTORY

In ancient Greece, people used to write on wax-covered tablets. The first documented occurrence of **steganography** was in the document The Histories of Herodotus, when Demeratus sent a secret message past guards by removing the wax from the tablet, writing on the tablet itself, and covering the tablet with wax again to disguise the message. Invisible ink is a form of **steganography** used in recent centuries.

Another form of **steganography** is in null ciphers, or unencrypted text messages. For example, one could hide a text message within a paragraph of words, so that by isolating every 10th word, the secret message can be detected. The paragraph itself would sound innocent to escape detection.

This form of **steganography** was often used in wars among spies. An example of **steganography** involves the early Greek practice of carving messages into wooden boards before coating them with wax to make a writing pad. These messages under the wax would be hidden while a benign message would be written on top of the wax on the writing tablet. Only those who knew to look for the hidden writing under the wax were privy to the message.

Other examples during the Roman Empire include tattooing secret information in a messenger’s shaven head and waiting for the heir to grow out before sending the messenger to the proper party. When the messenger was sent out with the secret message on his scalp he carried a decoy message in his hand.

Other examples include writing messages in Morse code on yarn which was knitted into clothing worn by messengers, writing in invisible ink between the lines on a benign message, and writing on the back of postage stamps. Recently, computerized **steganography** has become popular. Using different methods of encoding, secret messages can be hidden in digital data, such as images (.bmp or .jpg), as audio files (.wav), or e-mail messages.

Unfortunately, **steganography** is also suspected to play a role in the communication among terrorist groups around the world. In the past year, several suspected that terrorists might have been posting images on Ebay with hidden messages inside to send to different terrorist groups.

Recent attempts to detect the presence of such images on Ebay have not uncovered anything. You might have seen in old Chinese movies that monks used to safeguard secrets by hiding those secrets on their body parts so that no one else can see them but them.

STEGANOGRAPHY USAGE

The purpose of steganography is to hide (generally encrypted) data into other data. Examples of these types of practices:

- Used to combine explanatory information with an image (like doctor's notes accompanying an X-ray).
- Embedding corrective audio or image data in case corrosion occurs from a poor connection or transmission.
- Peer-to-peer private communications.
- Posting secret communications on the

Web to avoid transmission.
- Copyright protection.
- Maintaining anonymity.
- Hiding data on the network in case of a breach.

One of the most common illicit uses of **steganography** is for the possession and storage of child pornography images. However, **steganography** can also be used to commit fraud, terrorist activities and other illegal acts.

Steganography made news headlines when the U.S. Department of Justice charged 11 individuals in two separate criminal complaints with conspiring to act as unlawful agents of the Russian Federation within the United States. The defendants allegedly used **steganography** to embed messages in more than 100 image files posted on public websites.

The larger the image file the larger the amount of data it can be hidden. Employee data, pricing data, rates and many other types of sensitive information can be easily smuggled out right under your nose.

In terms of development, **Steganography** is comprised of two algorithms, one for embedding and one for extracting. The embedding process is concerned with hiding a secret message within a cover Work, and is the most carefully constructed process of the two. A great deal of attention is paid to ensuring that the secret message goes unnoticed if a third party were to intercept the cover Work.

The extracting process is traditionally a much simpler process, as it is simply an inverse of the embedding process, where the secret message is revealed at the end. As shown in Figure 1, two inputs are required for the embedding process:

1. Secret message - usually a text file that contains the message you want to transfer.
2. Cover Work - the image file that function as the carrier of secret message, used to construct a **stegogramme**.

The next step is to pass the inputs through the Stego-system Encoder, which will be carefully engineered to embed the message within an exact copy of the cover Work, such that minimum distortion is made; the lower the distortion, the better the chances of undetectability. The stego-system encoder will usually require a key to operate, and this key would also be used at the extraction phase.

This is a security measure designed to protect the secret message. Without a key, it would be possible for someone to correctly extract the message if they managed to get hold of the embedding or extracting algorithms. However, by using a key, it is possible to randomize the way the stegosystem encoder operates, and the same key will need to be used when extracting the message in order for the stegosystem decoder to know which process to use. This means that if the algorithm falls into enemy hands, it is extremely unlikely that they will be able to extract the message successfully.

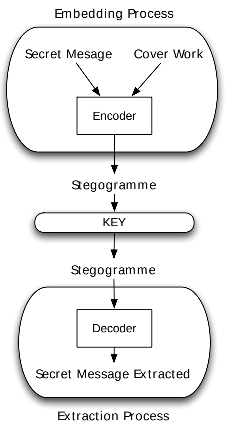

Figure 1
Steganography Diagram

The resulting output from the stego-system encoder is the **stegogramme**, which is designed to be as close to the cover Work as possible, except it will contain the secret message. This **stegogramme** is then sent over some communications channel along with the key that was used to embed the message. Both the **stegogramme** and the key are then fed into the stego-system decoder where the secret message is extracted.

We can refer to the output of the extraction process as an estimate. This is simply because the **stegogramme** might be subjected to noise. This occurs when it is sent over a communications channel, allowing the possibility of changing some of the values, it. It is unlikely to be certain that the message extracted is an exact representation of the original. Also, the recipient will obviously never know what the original message was, and so they have nothing to compare it to when it is extracted.

STEGANOGRAPHY ALGORITHMS

To better understand or grasp an overview of how this tools work, it is necessary to at least study the Least Significant Bit (LSB) algorithm. Using this method it is possible to embed a significant amount of information with no visible degradation of the cover image [11] [12].

This algorithm is the substitution process of adjusting the least significant bit pixels of the carrier image. It is a simple approach for embedding a message into an image. Of course this insertion varies according to the number of bits there are in the image. This technique provides to embed the data into the least significant bits of pseudo-randomly chosen pixels or sound samples [13, 14, 15].

For example, in an 8-bit image, the **LSB**, the 8th bit of each byte of the image is changed to the bit of the secret message. For a 24-bit image, the colors of each component like RGB (red, green, blue) are changed. This type of algorithm is effective in BMP images since the compression is lossless. Nonetheless, to hide a message in a BMP using this algorithm it would require a large image used as a cover.

STEGANOGRAPHY TOOLS

There were eight tools in total, chosen randomly, and tested for speed of execution, detectability and batch capability.

Hide & Reveal

An open-source software and a java library distributed under the GNU GPL primarily designed for scientists wishing to experiment new hiding techniques or **steganalysis** on various carriers (BMP, PNG and TIF) developed by Nathanael Cottin. One of the limitations this application has it restricts the use of more carriers. Since it is a Windows application it cannot run on a batch file although a macro could be developed and used, it still requires more programming.

Steghide

Steghide is a program able to hide data in various image and audio files like JPEG, BMP, WAV and AU developed by Stefan Hetzl. The color respectively sample frequencies are not changed thus making the embedding resistant against first-order statistical tests. It can be used with batch files for hiding or revealing files.

StegoStick

This free and open source steganographic tool, lets you hide any file into any file. Is based on image, audio, video **steganography** that hides any file or message into an image (BMP, JPG, GIF), Audio/Video (MPG, WAV, etc) or any other file format (PDF,EXE,CHM,etc.). P.V. Uma Mahesh developed this tool in java, which limits its capabilities to be executed in a batch file.

GIFShuffle

The program gifshuffle is used to conceal messages in GIF images by shuffling the colourmap, which leaves the image visibly unchanged. gifshuffle works with all GIF images, including those with transparency and animation, and in addition provides compression and encryption of the concealed message [7]. Matthew Kwan developed it, designed it for GIF images and can also be used with a batch file. Some software routines that hide information in GIF images are available on the Internet [17].

JPHSWIN

This GUI (graphical user interface) application containing or should I say merging JPHIDE.EXE (which is a DOS program to hide a data file in a jpeg file) and the JPSEEK.EXE (which is a DOS program to recover a file hidden with JPHIDE.EXE program). Allan Latham developed it and it allows the user to choose either to run a batch file or proceed with the windows-95 program (JPHSWIN).

Snow

Snow is program for concealing messages in text files by appending tabs and spaces on the end of lines, and for extracting messages from files containing hidden messages. Tabs and spaces are invisible to most text viewers, hence the steganographic nature of this encoding scheme. Developed also by Matthew Kwan, this tool can hide messages into any file and since it is a DOS program it can be executed in a batch file. Figure 2 illustrates the general process flow of the **Steganography** tool Snow.

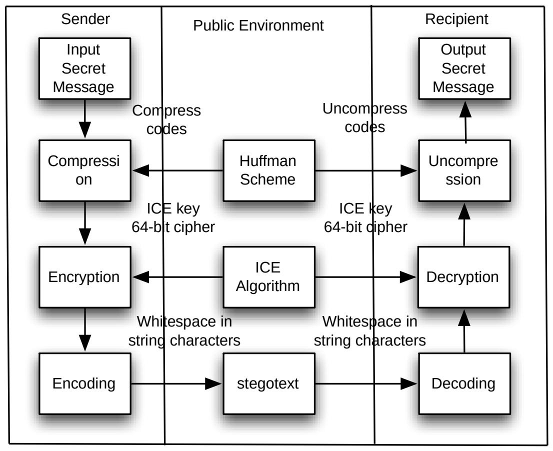

Figure 2
Snow

HIP

Hide In Picture (HIP) is a **steganography** program that allows you to "hide" any kind of file inside standard bitmap pictures developed by Davi Figueiredo. The pictures look like normal images, so people will not suspect they contain hidden data. A limitation in this program is the restriction on the carrier's format since it allows only the use of BMP and GIF images and it cannot be executed in a batch file.

DOS – Copy Command

Copy /B is a DOS command executed from the command prompt also known as the "black screen of death" which actually is the normal integrated MS-DOS black screen pop-up window. By using this already free and integrated tool developed by Tim Paterson, the copy /B command it copies one or more files to another location with the /B to indicate a binary file merging the files into a single one. It can use any file as carrier and it can be executed from a batch file. Out of all of the **steganography** tools tested it does not reveal the hidden message within the same program.

WHAT IS STEGANALYSIS?

Steganalysis is the art of identifying **stegogrammes** that contain a secret message.

Steganalysis does not however consider the successful extraction of the message; this is usually a requirement for cryptanalysis.

Typically, **steganalysis** begins by identifying any artifacts that exist in the suspect file as a result of embedding a message. None of the steganographic systems that are known today achieve perfect security [8], and this means that they all leave hints of embedding in the **stegogramme**. This gives the steganalyst a useful way in to identifying whether a secret message exists or not.

STEGANALYSIS TOOLS

There exist many **steganalysis** tools. Even so, there were chosen and tested only three of them due to broken links, demos not available and price restrictions.

Stegdetect

Stegdetect is a utility that analyses image files for steganographic content developed by Niels Provos. It runs statistical tests to determine if steganographic content is present, and also tries to find the system that has been used to embed the hidden information. This program is a direct compilation of the UNIX sources to a Windows binary. Out of the three other **steganalysis** tools tested this is the only one that can be executed from a batch file.

Stegspy

StegSpy is a program that is always in progress that allows identification of a "steganized" file, detecting **steganography** and the program used to hide the message developed by Spy-Hunter. Currently it identifies the following programs:

- Hiderman
- JP Hide and Seek
- Masker
- JPegX
- Invisible Secrets

It is a window-based application therefore it does not support batch file detection. It identifies the location of the hidden content as well.

XStegSecretBeta

XStegSecretBeta is a tool for detecting patterns in image files developed by Alfonso Muñoz in java. It also is a window-based application and it does not support batch file execution for detecting steganographic content.

Challenges to Law Enforcement

Although there have been some advances in **steganography** detection and breaking, there is currently no single easy-to-use tool available to law enforcement. Several factors intensify the challenge faced by law enforcement in detecting **steganography**: **Steganography** detection is usually handled separately from **steganography** decryption.

An automated tool that integrates stegdetection and decryption in a way that is familiar and easily accessible to law enforcement has not been developed. Newer **steganography**-encoding techniques are being rapidly developed, rendering the current detection tools ineffective.

Experimental Work

In order to evaluate each of the steganographic tools, a data set selection was chosen with different image size and different text messages size. Text chosen dataset is one of the important components in benchmarking steganographic techniques [9]. At first the tests implemented were conducted on star wars related images from the Internet while embedding the following messages:

- "You were the chosen one! You were meant to destroy the sith, not join them!"
- "This path has been placed before you. The choice to take it must be yours alone."
- "May the force be with you."
- "Password 712347"
- "Clouded the Future, the Dark Side has."
- "There is another... Sky...walker."
- "You worms are no match for the dark side!"
- "IP address 192.168.1.1.100 pass puprtech"
- "Server is unix user admin pass admin"
- "Remember backup runs @ 12:30am router is down"
- "PUP BFJNQH QQDY EVJOD RAM WROYRDHQ – THI SI SM YCODE DMESSAG E"
- "ZIO ASH JNM OWZV ICJHB WOMTP PRMFORQHDYSDKFMTII – YOUW ORMS ARENOMATCH FORT HEDA RKSIDE"

Once we got acquainted with the tools more tests were executed by embedding messages of 1kb, 10kb, 100kb and 1,000kb into images with size of 10kb, 100kb and 1,000kb. Figure 3 through 5 were the images taken at a beach in Isabela, PR, a yogurt store and the Polytechnic University at Hato Rey used as a cover image.

Figure 3
Isabela, PR Beach Image

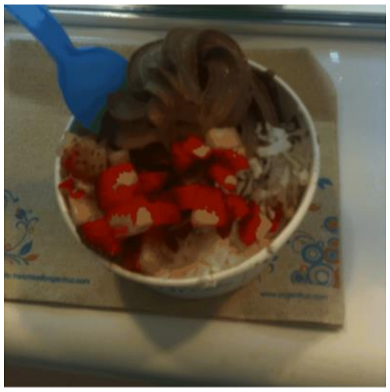

Figure 4
Yogurt Image

Figure 5
Polytechnic University Image

In Table 1 and Table 2 we can see the file size distribution phases. In our study the file size varied from phase I to phase IV, starting with 1kbytes, followed by phase II with 10 kbytes, phase III with 100 kbytes and end up with 1,000 kbytes in phase IV. For every size category various files were created in order to determine an approximate relation of the text file size and the image file size that would allow the detection tools to recognize the file as s **stegogramme**. During the phases we performed, the creation of batch files in some cases accelerated the testing process. An example of these batch file code for extracting the embedded message from a bmp image file into a text file:

```
@echo off
GIFSHUF -C Apple1output.gif > temp.txt
type temp.txt >> mymessages.txt
echo  Apple1output.gif >> mymessages.txt
```

Another example of batch file codes used to create the embedded files:

```
:@echo off
copy/b playa_copy_10kb.JPG +
SecretTextMessage1KB.txt na1.JPG
copy/b playa_copy_10kb.JPG +
SecretTextMessage10KB.txt na2.JPG
copy/b playa_copy_10kb.JPG +
SecretTextMessage100KB.txt na3.JPG
```

Table 1
Phase I and II Distribution File Size Category

Phase I Message Size	Phase II Message Size	Image File Size (KB – Kilo bytes)
1	10	1
1	10	10
1	10	100
1	10	1,000

Table 2
Phase III and IV Distribution File Size Category

Phase III Message Size	Phase IV Message Size	Image File Size (KB – Kilo bytes)
100	1,000	1
100	1,000	10
100	1,000	100
100	1,000	1,000

In Table 3 we will see a comparison of other **steganography** tools, in terms of the image format, the detection algorithms used and the creators.

Table 3
Comparison Work

Name	Creator	Image Format	Detected by
JSteg	Derek Upham	JPEG	-X^2-test - Stegdetect - J.Fridrich’s Algorithm
JSteg-Shell	John Korejwa	JPEG	-X^2-test
JPhide	Allan Latham	JPEG	-X^2-test - Stegdetect

OutGuess v.0.13b	Provos and Honeyman	JPEG	-X^2-test - Stegdetect
OutGuess v.0.2	Provos and Honeyman	JPEG, PNM	-J.Fridrich's Algorithm
EZStego	Romana Machado	BMP, GIF	-RS **Steganalysis**
S-Tools	Andrew Brown	BMP, GIF	-X^2-test
F5	Andreas Westfeld	JPG, BMP, GIF	-J.Fridrich's Algorithm

EXPERIMENTAL RESULT

After the experiment, the tests performed provided in our study a clearer perspective in the capacities of the tools. In Table 4 through Table 6 we can see the results and a description of our findings.

Table 4
GIFShuffle, Hide & Reveal and Steghide Results

Test Case	Embed File Created
Image Size = Message Size	No
Image Size < Message Size	No
Image Size > Message Size	Yes - Conditioned

The ratio in which it accurately indicates if the **stegogramme** can be created or not in the Hide & Reveal application depends on many variables since it has multiple combinations for creating the steganographic file. Mostly as other java applications tested the message file has to be 8% percent or less of the image size in order to be processed.

One detail that must be mention with Steghide is that the message file has to be less than 30% percent of the image (carrier file) size or it will fail to embed the new file. The restriction found in GIFShuffle is size, to 1 Kilobyte of data approximately 324 characters. If the text message surpasses this limit it will display an error message indicating, "Message exceeded available space".

Table 5
StegosStick, SNOW, HIP and DOS Results

Test Case	Embed File Created
Image Size = Message Size	Yes
Image Size < Message Size	Yes
Image Size > Message Size	Yes

One thing discovered with StegoStick is that it allows any file size since it merges both files.

Snow has one advantage that it can create the embedded file regarding the size of any image or message file. Still it has one major drawback when the files are not text based, if the carrier file is an image then it will not be displayed in any of the image preview application, arousing suspicious over the embedded file.

Table 6
JPHSWIN Results

Test Case	Embed File Created
Image Size = Message Size	No
Image Size < Message Size	No
Image Size > Message Size	Yes

The Hide and Seek application even when the use is limited to only JPG files it allows for embedding messages to images when the image size is greater than that of the message.

The same results in the Snow tool was obtained in the Hide in Picture (HIP) tool since it merged both the carrier and the message file. One difference in these two is that the images can be previewed in any image software. Visually there was a variance on the images as the cover image was embedding a bigger message.

The copy command out of all of the tools, provided less or no resistance in embedding the image and message files. It easily executed over batch and the image or any other file format i.e, a zip or rar document could be opened. The disadvantage on this method is that it actually lacks of security since it has no encryption or a key.

Table 7
Steganalysis Tools Results

Name	Embed Method Detected	Detection
Stegdetect	JPHSWIN	1 of 33
	100kb image with 10kb message	
StegSpy	Hiderman(StegoStick)	3 of 12
	1000kb image with 10kb message	
	1000kb image with 100kb message	
	1000kb image with 1000kb message	
	Hiderman(GIFShuffle)	4 of 12
	1000kb image with 1kb message	

	1000kb image with 10kb message	
	1000kb image with 100kb message	
	1000kb image with 1000kb message	
	Hiderman(JPHSWIN)	5 of 18
	100kb image with 1kb message	
	1000kb image with 1kb message	
	1000kb image with 10kb message	
	1000kb image with 100kb message	
	1000kb image with 1000kb message	
	Hiderman(HIP)	5 of 22
	1000kb image with 10kb message	
	1000kb image with 1kb message	
	1000kb image with 10kb message	
	1000kb image with 100kb message	
	1000kb image with 1000kb message	
	Hiderman(DOS)	4 of 12
	1000kb image with 1kb message	
	1000kb image with 10kb message	
	1000kb image with 100kb message	
	1000kb image with 1000kb message	
XStegSecretBeta	SNOW/DOS	28 of 122
(3)	10kb image with 1kb message	
(3)	10kb image with 10kb message	
(3)	10kb image with 100kb message	
(3)	10kb image with 1000kb message	
(2)	100kb image with 1kb message	
(2)	100kb image with 10kb message	
(2)	100kb image with 100kb message	
(2)	100kb image with 1000kb message	
(2)	1000kb image with 1kb message	
(2)	1000kb image with 10kb message	
(2)	1000kb image with 100kb message	
(2)	1000kb image with 1000kb message	

Most of the detected files had an image size of 1,000kb and a text message file of 1 KB through 100 KB as mentioned in Table 7. The tendency of detection becomes stronger when files were using a merge method without compression or slightly at all.

DISCUSSION

Steganography as defined in media circles is defined as a potential threat. We were able to see the code of some of this open source application, which in principle have the same concept of embedding one file (message) into another file (carrier or cover). The problem presented before us is actually to test and determine if any of these tools are capable of withstanding the manipulation of a user destroying the message that was embedded and the need to study and develop even better detection tools since there are being develop even more embedding tools than a decade ago. It should be considered as a supplement to security when combined with encryption.

As in any computer based company, there is not one tool, in our case one application that can cope with every user demand, still by evaluating the options available and how they work, will provide us with better knowledge when the moment to make a decision arrives, in order to determine buying and using an out-of-the-box solution or simply choose another that can be customizable.

CONCLUSION AND FUTURE WORK

All the **steganography** and **steganalysis** tools have their advantages and drawbacks as our study revealed. The media would refer to this science, as a potential threat while others would just perform it out of curiosity or even play spy games.

Conducting our research we were confirmed by history that sending hidden messages has being around more than a century ago as recorded in Greek history during the war with the Persian Empire. Technology provides a new and bigger medium in which **steganography** could be used as the computer, Internet, and mobile phones.

Network or Database Administrators of any database containing images could just convert every image file inbound or outbound. This procedure will make changes to the bytes of the image rearranging it to the new image format destroying the secret message embedded. To detect a **stegogramme** created with any of the mentioned programs, one would have to have some sort of Computer Forensics knowledge since evidently one tool will not suffice.

Either create a steganalysis application that can be as complex and sophisticated enough or have a few of the tools most used for embedding and try to extract the message by other means as batch execution or brute-force dictionary mechanism.

Future enhancements like creating different algorithms and implement them in R-Language or any other statistical application would allow for other types of analysis such as histograms for inspecting images as well automating other tests.

It is and it will always be a potential threat since the tools can be used for illicit actions, therefore maintaining the law enforcement agencies with the latest trends through equipment and technology training and hands-on workshop might improve the detection of any secret communication. Similar to image steganography, far more information can be hidden in videos, which are just a combination of a series of images and sound [10].

ACKNOWLEDGMENTS

I would like to gratefully thank first of all, God, who made my journey and goal possible, my loving wife Loraine, my best friend Omar and of course my parents and family, to whom I dedicate my achievements. Their infinite support contributed for the accomplishment of this project.

REFERENCES

[1] Krenn, J.R., “Steganography and Steganalysis”, January, 2004.

[2] A. Cheddad, J. Condell, K. Curran and P. Mc Kevitt, "Digital Image Steganography: Survey and Analyses of Current Methods". Signal Processing, Volume 90, Issue 3, March 2010, Pages 727-752.

[3] Stritzel, Adam, “Practical Aspects of Modern Steganography.”, March 6, 2006.

[4] J. Kelley, “Terror Groups Hide Behind Web Encryption,” USA Today, Feb. 2001, www.usatoday.com/life/cyber/tech/2001-02-05-binladen.htm.

[5] Provos, N. & Honeyman, P., “Hide and Seek: An introduction to steganography”, IEEE Security and Privacy Journal, 2003

[6] Kessler, Gary C., “Steganography: Hiding Data Within Data.”, September 2001.

[7] Aelphaeis Mangarae, “Steganography FAQ [Zone-H.Org]”, March 18, 2006.

[8] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker. "Digital Watermarking and Steganography (Second Edition)", Morgan Kaufmann Publishers, ISBN: 978-0-12-372585-1, 2007.

[9] M. Kharrazi, H. T. Sencar, Memon. “Benchmarking steganographic and steganalysis techniques” Security, Steganography, and Watermarking of Multimedia Contents, 2005.

[10] Oran, Deniz. “TJHSST Computer Systems Lab Senior Research Project Implementation of Least Significant Bit Steganography and its Steganalysis 2009-2010”, January 27, 2010.

[11] BENDER, W. – GRUHL, D. – MORIMOTO, N., Techniques for Data Hiding, Massachusetts Institute of Technology, Media Laboratory Cambridge, Massachusetts 02139 USA, From the Proceedings of the SPIE, 2420:40, San Jose CA, February, 1995.

[12] CURRIE, D. L., Surmounting the Effects of Lossy Compression on Steganography, III Fleet In- formation Warfare Center 2555 Amphibious Drive NAB Little Creek Norfolk, VA 23521 3225 currie@msn.comi Cynthia E. Irvine Computer Science Department Code CS/Ic Naval Post- graduate School Monterey, CA 93943 5118 irvine@cs.nps.navy.mil Proceedings of the 19th National Information System Security Conference, Baltimore, Md, October 1996, pp. 194–201.

[13] Stefano Cacciaguerra and Stefano Ferretti, "Data hiding: steganography and copyright marking," Department of Computer Science, University of Bologna, Italy. http://www.cs.unibo.it/-people/phd-students/scacciag/home_files/- teach/datahiding.pdf, 2003, pp 12.

[14] S. Walton, “Image authentication for a slippery new age.” Dr. Dobb's journal of software tools, vol. 20, no. 4, pp. 18-26, Apr. 1995.

[15] K. Matsui and K. Tanaka, “Video-steganography: How to secretly embed a signature in a picture.” Journal of the Interactive Multimedia Association Intellectual Property Project, vol. 1, no. 1, pp. 187-205, Jan. 1994.

[16] Fox, Stuart, “Russian Spies Hid Secret Codes in Online Photos”, TechNewsDaily June 29, 2010.

[17] R. Machado. Stego. http://www.fqa.com/romana /romanasoft/stego.html