DETECTING STEGO-OBJECTS IN STILL IMAGES: AN INTEGRATED APPROACH

TARIQ AL HAWI, MAHMOUD AL QUTAYRI AND HASSAN BARADA

College of Engineering and Information Sciences
Etisalat University, Sharja, UAE

Abstract - *This paper proposes a testbed environment for evaluating the security and robustness of the major steganography techniques. The testbed environment allows the embedding of messages into still images, pre-processing to detect the hidden messages and a post processing stage to perform further analysis on an image. The environment was used to test the survivability of stego-images. This was achieved by using cover images and hidden messages of different formats and sizes. A large set of images were subjected to a wide range of steganography tools. The output of the pre-processing stage achieved a high success rate in indicating the presence of hidden messages in the tested images.*

Key-Words: Stego, Image, Detecting, Integrated, Internet

I. INTRODUCTION

Although security concepts have been around for years, the impact it has nowadays on the Internet is undeniable. Nowadays, we are surrounded by a world of secret communication where people are able to transmit secret information through innocent looking carriers [1]-[2]-[3].

Steganography is the art of hiding and transmitting data through apparently innocuous carriers in an effort to conceal the existence of the data [1]. It has some similarities to other forms of data hiding, such as watermarking [4]. However unlike watermarking where the cover and the signature are the ones to be protected steganography is only concerned with the concealment and protection of the hidden message underneath the cover. Steganalysis is the art as well as the science of discovering and rendering useless such covert messages.

There are two important aspects to information hiding system: security and robustness [5]-[6]. Security refers to the inability of an eavesdropper to detect the hidden message. Robustness refers to the amount of distortion that the digital cover can withstand before the hidden message is destroyed.

The paper describes and discusses a testbed environment for evaluating the security and robustness of steganography techniques. It describes the methods implemented in each component of the environment and explains the testing methodology used. Testing results using various types and sizes of images and hidden messages are also reported and analyzed.

II. THE TESTBED ENVIRONMENT

A testbed environment has been devised in an effort to build a system that is capable of monitoring Internet traffic and detecting or distorting the hidden information in digital images of various types. Figure 1 illustrates the testbed environment, which consists of two subsystems: a steganography system that consists of a steganography toolbox, and a Steganalysis subsystem that consists of three stations namely, capturing and preprocessing station, steganalysis station, and a distortion station. The environment is used to accomplish the following objectives:

- Apply some of the major steganographic techniques on still digital images of various types and sizes.
- Test the security of the applied steganography techniques by launching attacks on stego-images using different steganalysis approaches.
- Test the robustness of embedded information by applying various distortion techniques on stego-images.

The following is a description of the four stations that the environment consists of:

1) Steganography toolbox: This station contains a collection of the most commonly used image steganography software packages such as S-Tools, J-Steg, Jpegx, Invisible Secrets 2002, Hide and Seek, and Camouflage [7]-[8]. This station is responsible for producing stego-images containing a secret message using the various software tools.

2) Capturing and preprocessing station: This station has two major functions. In the first function, it acts as a network sniffer that captures stego-images passing through in their raw format (i.e. Hexadecimal). The sniffer has the capability to reconstruct any image into its original format. After getting captured, the stego-image is preprocessed in order to make initial assessment whether it contains a hidden message or not. If an image draws suspicion, then it is passed to the steganalysis station. Otherwise, the image passes the test and assumed to be free from hidden messages. Suspicion is raised depending on few parameters that are analyzed by examining the hexadecimal representation of the stego-image. These parameters may include the addition of a specific software signature, replacing the LSBs with all zeros or ones, addition of blocks of spaces at the end of the file and

increasing the number of duplicate colors in an image. The goal of this stage is to drop the images that do not raise suspicion from the steganalysis phase in order to speedup the process of monitoring the Internet for hidden information.

3) Steganalysis station: This station, which is used to evaluate the security of the steganography techniques, is responsible for trying to detect hidden messages in the stego-images. This is achieved by using statistical analysis techniques that depend on the steganography tools used in the steganography toolbox [9]. Statistical tests can reveal that an image has been modified by steganography by determining how much the statistical properties of the image deviate from the norm.

4) Distortion station: The last resort for a steganalyst is to disable the hidden message and render it useless if the stego-image raised any suspicion but the message cannot be detected. The aim of this phase is to destroy the hidden information while maintaining the integrity of the original cover. This station, which is used to evaluate the robustness of the steganography techniques, will apply various image processing techniques such as filtering, blurring, etc. to destroy the hidden message.

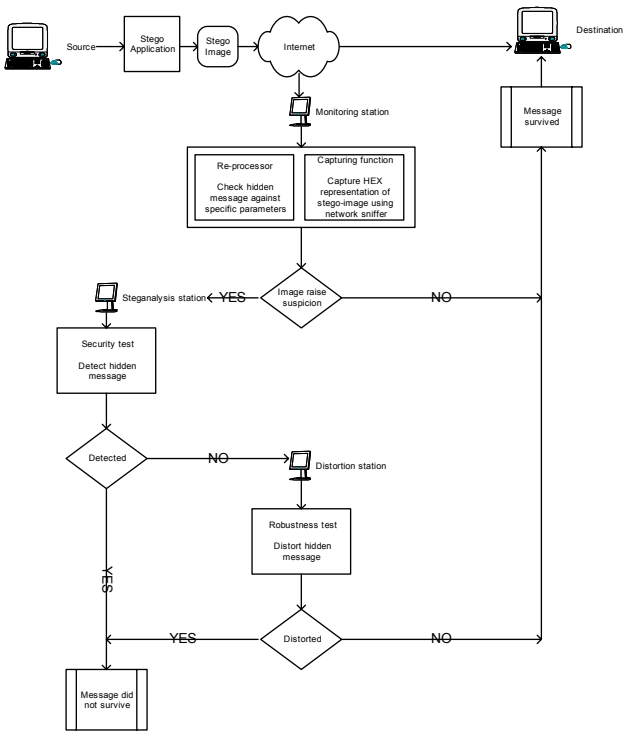

Figure 1
Testbed Environment

III. Testing and Results

A collection of images of various formats and sizes were used as cover carriers. The format included JPEG, GIF, 8-bit BMP, 16-bit BMP and 24-bit BMP and the sizes were in the range from 1-600kb. Each category consisted of 50 different images to be tested in the environment. The steganography tools currently used in the environment are based on Data Insertion, LSB (Least Significant Bit), palette manipulation and DCT (Discrete Cosine Transform) steganography techniques. An extensive set of test sets has been processed and the analysis of these tests is categorized depending on the steganography techniques used.

A. Data Insertion Technique

Once the stego-image created using Data Insertion Technique is captured the system will look for very unique signatures created by the different steganography tools. In the case of Camouflage, the program inserts a block of spaces before attempting to embed the secret message. Figure 2 shows the Hexadecimal representation of two images. At the top is the original image with no hidden data embedded and at the bottom is the stego-image with a medium sized image as a hidden message. Notice the addition of spaces (Hex = 20) at the end of the file. Almost all the tested images were defeated using this detection mechanism. This concludes that the steganography algorithm used by Camouflage is relatively easy to detect.

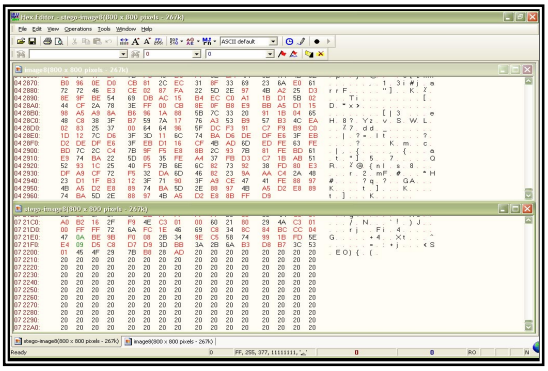

Figure 2
Output of a Sniffer (Camouflage)

In the case of Jpegx 1.00.6, the program performs similar techniques for embedding messages as Camouflage. It does that by inserting the secret message at the end of JPG files. But before it does that it adds a fixed signature of the program. The signature is the following number of bytes [5B 3B 31 53 00]. The detection mechanism was to look for that very signature to raise suspicion. Figure 3 shows the Hexadecimal output of two images. The top is an image with no hidden messages and the bottom is the image with text hidden in it. Notice the signature highlighted at the end of the stego-image in figure 3.

B. Least Significant Bit Technique

For this technique, after embedding the secret message, the program replaces the remaining LSBs with either all 0's or all 1's data. The detection method was first to extract the LSBs of a stego-image and then look for the block of 0's or 1's. The tested steganography tool for this technique is Invisible Secrets 2002. When hiding different

messages of various sizes and formats, Invisible Secrets 2002 prompts for a secret message size restrictions. Figure 4 shows the LSB extracted from two different images. On the right is the image with no hidden message while on the left is the stego-image. Notice the blocks of 1's after extracting the LSB of the stego-image.

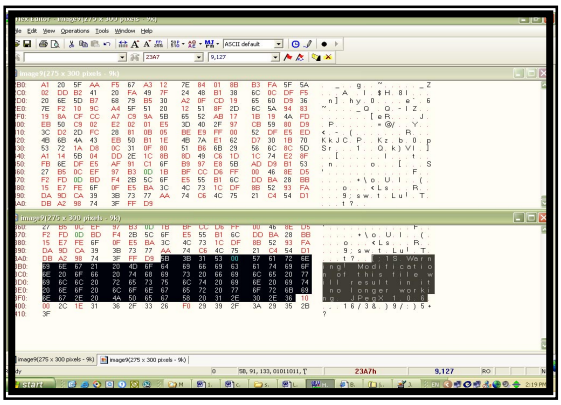

Figure 3
Jpegx Signature

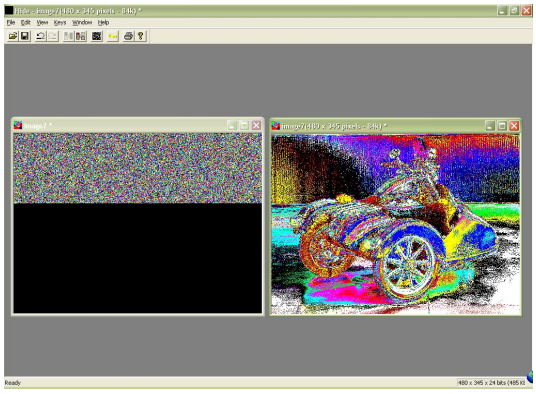

Figure 4
LSB Extraction of Stego-image Created Using Invisible Secrets

C. Palette Manipulation Technique

Using Palette Manipulation Technique always leads to a small change in the carrier file size [10]. However this will not look unusual since the original cover is assumed to be not available to the steganalyst. Stego-images created using such technique have many duplicate colors in their color table. This is because the technique hides data by reducing the total number of colors in the color table and creates duplicates of them. These are not exact duplicates, but rather are colors so close to the original that the difference cannot be noticed by the human sensory system. It is very critical when embedding hidden messages that any degradation of the carrier file cannot be noticed by the human eyes. In order to achieve this goal only specific numbers of bytes are allowed to be embedded as hidden message. This is a variable that depends on the size of the carrier image file.

In order to detect hidden messages embedded in stego-images created by this technique, the color table of the stego-image is examined for duplicate colors. A program was written that extract the duplicate colors of an image in addition to other useful information. An example of the program output is shown Figure 5.

The output of a clean file:
C:\stego\ stool image.bmp
File Name: image.bmp
Actual size: 66132 Reported 66132
Duplicate colors: 2
File Header: Bytes 0-13
Bitmap header: Bytes 13-53
Color map: Bytes 54-609
Image data: Bytes 610-66131
The output of a stego-image:
C:\stego\ stool stegoimage.bmp
File Name: stegoimage.bmp
Actual size: 66614 Reported 66614
Duplicate colors: 1046
File Header: Bytes 0-13
Bitmap header: Bytes 13-53
Color map: Bytes 54-1077
Image data: Bytes 1078-66613

Figure 5
Palette Manipulation Program Output

From Figure 5, the original cover had only 2 duplicate colors in the color table to start with. After embedding the secret message it increased to 1046 duplicates. A threshold of 200 was set for this test. If the reported number is above 200 there is a good chance that hidden exist. The test included only 8-bit color bmp images. A set of secret messages of different sizes and formats where used. This detection mechanism was used to detect secret messages in stego-images created by S-Tools and Hide and Seek steganography tools.

D. Discrete Cosine Transform Technique

DCT in its simplest form is a transformation mechanism to compress information in a file. Because the hidden data is embedded in the spatial domain, hidden messages in stego-images created by DCT technique are very difficult to detect. However these stego-images will raise suspicion if some statistical analysis were performed on them. Checking the DCT coefficients is the ultimate key to discover whether data have been hidden or not. In ordinary JPEG images that have no hidden messages, the DCT coefficients have nearly a symmetric distribution, smoothly falling away from the central value. In stego-images created by this technique the smoothness and symmetry are interrupted.

Normal detection techniques used for the previous steganography tools are useless in the case of DCT technique. Due to this fact, a program called "StegDetect" that performs statistical analysis for the DCT coefficients is used. StegDetect is mainly used to detect hidden messages in stego-images created by J-Steg, however many developers have added other routines to the program

to detect hidden messages in stego-images created by other steganography tools such as Camouflage and Jpegx. Throughout our test, it was noticed that the accuracy and efficiency of StegDetect for other programs than J-Steg has a very low percentage.

Since StegDetect was originally designed to perform statistical analysis on only JPEG stego-carriers, the experiment performed the tests on only JPEG images to find out whether secret messages have been embedded or not. Table I shows the percentage of detection for three tools: Camouflage, Jpegx and J-Steg. Notice that Camouflage and Jpegx results had a very low percentage of accuracy while J-Steg results had a percentage of accuracy that is quite high. The reason being is that StegDetect was created mainly to detect hidden messages in stego-images created by J-Steg. During the development process of this program other developers added other routing to detect hidden messages in stego-images created by other softwares such as Camouflage and Jpegx, however the results were inaccurate.

Table I
Detection Percentage

Tool	Hidden Message	Total # of Processed Images	Total # of Images Raised Suspicion
Camouflage	Text	50	2
	Small-size image	15	1
	Medium-size image	5	1
	Large-size image	2	0
	Oversized image	12	1
Jpegx	Text	50	2
J-Steg	Text	50	14
	Small-size image	50	50
	Medium-size image	50	50
	Large-size image	50	50

The table above shows that for all 50 stego-images created by both Camouflage and Jpegx only 2 hidden messages were detected. This is mainly because StegDetect depends on the output of the DCT statistical analysis on stego-images while both programs use the LSB technique for steganography. The two softwares were included to show that StegDetect doesn't work accurately with other programs than J-Steg.

IV. CONCLUSION AND FURTHER WORK

The paper proposed a tested environment for evaluating the security and robustness of some of the major steganography techniques. This was achieved by capturing images and processing them using two subsystems. In the first subsystem, the steganography toolbox, four different techniques were used to produce stego-images. In the second subsystem these stego-images are examined first by a pre-processing stage. In that stage we were able to raise suspicion about all tested stego-images using different parameters, hence defeating the tested steganography techniques. The steganalysis workstation conducted the security test and it was found that most hidden messages were detected. All images that survived the test were injected to the distortion station where the hidden messages were either destroyed or distorted. In general the pre-processing stage played an important role in raising suspicion about most captured stego-images. Future work will concentrate on designing an analysis stage that will be capable of dealing with a wider range of steganography algorithms.

V. REFERENCES

[1] Katzenbeisser S., and F. A. P. Petitcolas, *Information Hiding techniques for steganography and digital watermarking*, Boston, USA, Artech House, 1999.

[2] M. K. Simon, J. K. Omura, R. A. Scholtz, and B.K. Levitt, "Spread Spectrum Communication", *Computer Science Dept. at Rockville University*, vol. 1, 1985.

[3] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information Hiding – A Survey", *Proceeding of the IEEE*, vol. 87, no. 7, July 1999, pp. 1062-1078.

[4] P. Bassia, and I. Pitas, "Robust Audio Watermarking in the Time Domain", *Findings report, Department of Electrical Engineering, Information Theory Group*, Delft University of Technology, 1997.

[5] Westfeld A. and Pfitzman A., "Attacks on Steganographic Systems", proceedings of the Third International Information Hiding Workshop, Dresden, Germany, September/October, 1999, pp. 61-76.

[6] R. J. Anderson, and F. A. P. Petitcolas, "On The Limits of Steganography", *IEEE journal of selected areas in communications*, vol. 16, no. 4, May 1998, pp. 373-381.

[7] N.F. Johnson, and S. Jajodia, "Steganalysis of Images Created Using Current Steganography Software", *Proceeding of the 2nd International Workshop on Information Hiding*, vol. 1525, 1998, pp. 273-289.

[8] W. Bender, D. Gruhl, N. Morimoto, and A. Lu., "Techniques for Data Hiding", *IBM Systems Journal*, vol. 35, no.3, Feb. 1996, pp. 313-336.

[9] Fridrich J. and Goljan M., "Practical Steganalysis – State of the Art", *proceedings SPIE Photonics West, Electronic Imaging 2002, Security and Watermarking of Multimedia Contents*, San Jose, California, *Vol. 4675,* January, 2002, pp. 1-13.

[10] Y. Tseng, "Data Hiding in 2-Color Images", *IEEE transaction on computers*, vol. 51, no. 7, July 2002, pp. 873-880.