The Capability Of Image In Hiding A Secret Message

ROSHIDI DIN
Faculty of Information Technology
Universiti Utara Malaysia (UUM)
Sintok, 06000 Kedah
MALAYSIA

HANIZAN SHAKER HUSSAIN
SALEHUDDIN SHUIB
Faculty of Information Technology
Universiti Tun Abdul Razak (UNITAR)
Alor Setar 05000 Kedah
MALAYSIA

Abstract : - This paper focuses on the capability of image files as cover message to send the text files. The images are classified according to their compression technique either lossless or lossy compression. The measurements are identified in order to test their capability based on file size, color intensity level, integrity data, and the execution time. All the measurements are tested by using the developed tool called *StegaMage*. The result of the study is to suggest that BMP image is preferable than JPEG image.

Key-Words: cover message, secret message, lossless compression, lossy compression, BMP, JPEG.

1 Introduction

In recent years, many Internet communication and applications use image data as a container of secret information [11],[15] and has been the most popular medium used for covering and hiding secret messages [7] especially in steganography techniques [19]. The advantage of steganography technique also known as an art of sending invisible messages [16] compared to cryptography technique is its ability to conceal the existence of secret messages in digital image [1] by a third party.

Due to the weaknesses of the human visual system, which has a low sensitivity in random pattern changes and luminance [2], digital images are used for steganography. The human eye is incapable of discerning small changes in colors or patterns. Thus text or image files can be inserted into the carrier image without being detected that could be used to cover the message.

Compression type [5] and color variance [13] need to be considered in this technique. Message can be embedded as large as 50% of the original digital image [11] in ways that are imperceptible to the human eyes [6] depending upon what type of compression algorithm been used [8]. Several compression algorithms have been developed to decrease the storage and requirements of handling image files [12]. Two popular types of compression algorithms of files are lossless compression and lossy compression. However, the results [18] are different.

Lossless compression of an image is stored in either Graphical Interchange Format (GIF) or in Windows Bitmap (BMP). This type of compression let receiver reconstruct the original message exactly. Therefore original information, will be used so that it remains intact [9]. The most common steganography approach for content based image formats are BMP, GIF and JPEG that are widely used by media to transmit hidden messages [4]. The image recovered will be identical to the original image [10] after the encoding and decoding processes.

The other type of compression technique is lossy compression that saves the image file space. However, this approach may not maintain the original message integrity because some information of image file might be lost. Joint Photography Expert Group (JPEG) is an example of image file that uses this compression because the image recovered from the compressed JPEG file is a close approximation, but not identical to the original image [17].

Nevertheless, the issue is how these image files can be made to carry hidden message from sender to receiver whereby using a compression technique on an image, can cause some information loss [9]. Thus, the main objective of this study is to examine which type of image files are capable as a cover message that hide a secret message using a compression techniques based on the size of file, the intensity level, the integrity data, and the time execution. In this study focus will be on the capability of image as a cover message using two kinds of image files which are BMP (lossless compression) and JPEG (lossy compression).There are three types of file formats that are going to be used, namely text file format (*.txt*), batch file format (*.bat*) and rich text file format (*.rtf*).

2 Experimental Works

The technique of this experimental work [5] focuses on algorithm description, prototype development, format of file types and the image types for covering message.

The algorithm was designed based on the least significant bit (LSB) method, a commonly used insertion type scheme implemented currently in digital steganography [8]. In this LSB method, each bit in the message in the program picks a random pixel in the image and a random red, green, or blue component in the pixel. It then sets the least significant bit in that component to the bit value it is embedding. Changing the LSB does not result in a human-perceptible difference because the amplitude of the change is small [14]. This algorithm has the capability to embed text, batch and rich text files format in BMP and JPEG files. The algorithm can be use to send a few types of files and also can accept two types of images of compressions.

A program called *StegaMage* was developed as a tool to test the BMP and JPEG files as a cover message that conceals the hidden messages. This program uses password that is created by the random number generator. When the message is extracted, the program initializes the random number generator by using the same seed so that it would produce the same series of psedorandom numbers. Based on Fig.1, the hidden messages (*HidMsg*) including *.txt* file, *.bat* file and *.rtf* file will be embedded in a cover messages (*CovMsg*), *globe.bmp* and *lab.jpg* through an embedding process. The embedding process will be completed by *EmbedBait*.

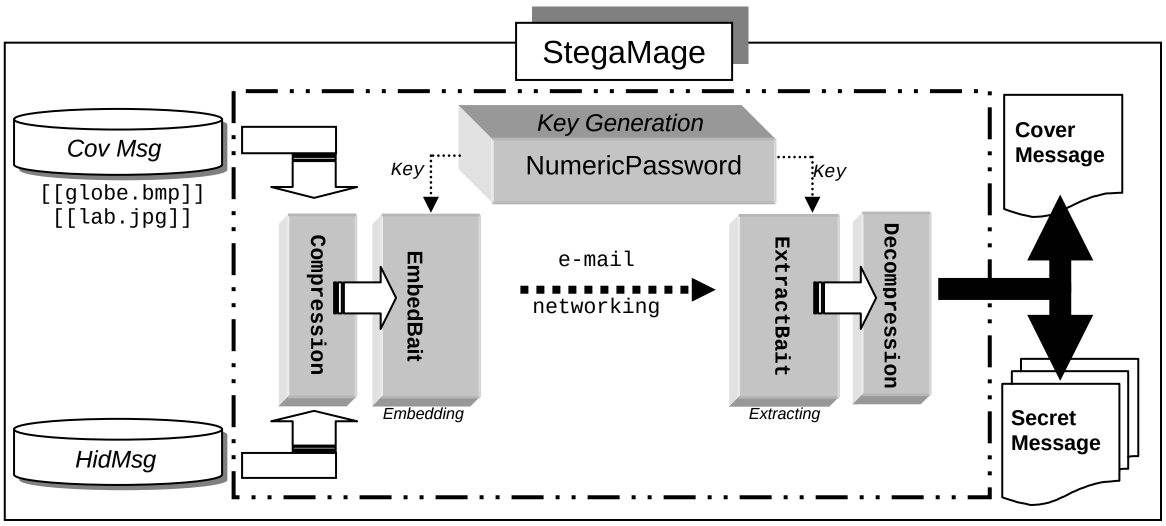


Fig.1 The StegaMage framework adapted from Cachin C.[3].

However prior to the embedding process, all files will be compressed during the compression stage. In the embedding process, the key also would be determined and generated by *NumericPassword*. Then, an embedded message is sent through public channel such as through electronic mail (e-mail) or networking. The same key will be used in the extracting process done by *ExtractBait*. The decompression process will take place once the extracting process is over. Finally, the extracted messages will be transformed into two separate types: cover message and secret message.

Three types of files format which are text file (*.txt*), batch file (*.bat*) and rich text file (*.rtf*) are used as hidden messages that will be embedded into *globe.bmp* and *lab.jpg* as cover messages. An equal size of two kilobyte (2 Kb) hidden message is used. Any changes after the embeddedment will be confirmed by detecting differences in term of size of files. The information on the images prior and after embedment is recorded and presented. Graph is used to illustrate the prior and after information through the distribution of pixels in an image by plotting the number of pixels at each color intensity level. It will be able to show the spectrum of color component that appears on the horizontal axis and the vertical axis indicating the portion of the image of the color. Each image is measured by using the grayscale image in histogram [10] by showing the pattern of the color. Meanwhile, a few different sizes of number characters (44, 109, 328, 1640, and 6559) are chosen. The execution time of these different sizes of number characters is measured and recorded during embedding and extracting process.

3 Experimental Result

This section discusses the verification of the capability of image using file size, color intensity level, data integrity and the execution time.

3.1 Files Size

Table 1 shows the summary of the image information of the *globe.bmp* and *lab.jpg* files. Before the embedding process, the dimensions of the *globe.bmp* images is 413 x 203 pixels and were transformed to 476 x 288 pixels for the *.txt*, *.bat* and *.rtf* file format. The size of image changes from 245Kb for .bmp format to 401Kb for the three files format. The embedding process also changes the pixel per inch of the image from 199.900 to 200 pixels.

	globe.bmp			
	Before embedded	After embedded		
		.txt file	.bat file	.rtf file
Dimensions (pixels)	413 X 203	476 X 288	476 X 288	476 X 288
Size of file (kilobyte)	245 Kb	401	401	401
Pixel Per Inch	199.9	200	200	200
Pixel depth/colors (million)	24/16	24/16	24/16	24/16
	lab.jpg			
	Before embedded	After embedded		
		.txt file	.bat file	.rtf file
Dimensions (pixels)	413 X 310	476 X 288	476 X 288	476 X 288
Size of file (kilobyte)	18 Kb	401	401	401
Pixel Per Inch	200	200	200	200
Pixel depth/colors (million)	24/16	24/16	24/16	24/16

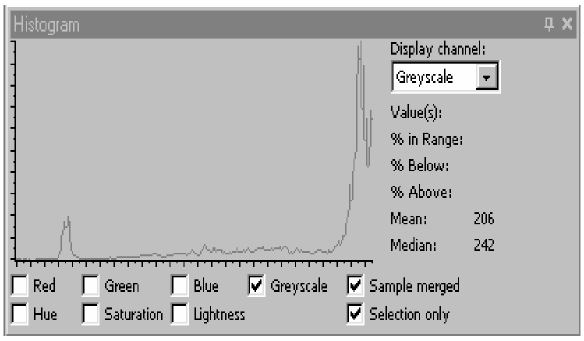
Table 1**:** The information of *globe.bmp* and *lab.jpg*

Besides that, the dimensions of the *lab.jpg* file before the embedding process is 413 x 310 pixels and it changes to 476 x 288 pixels for the *.txt* file, *.bat* file and *.rtf* file. The size of *lab.jpg* file also changes from 18Kb before embedding process to 401Kb. Meanwhile, the pixel depth/colors of the both files format remains unchanged which is 24/16 millions. The size of *lab.jpg* file increases the density from 18Kb to 401Kb compared to *globe.bmp* file from 245Kb to 401Kb. Thus it can be concluded that the size of both image files, *globe.bmp* and *lab.jpg*, changes after the embedding process.

3.2 The Color Intensity Level

Fig.2 shows the color intensity level of *globe.bmp* before the embedding process. The graph illustrates the grayscale color for the display channel and sample merge and selection only. Before the embedment process, the means for the color histogram is 206 and the median is

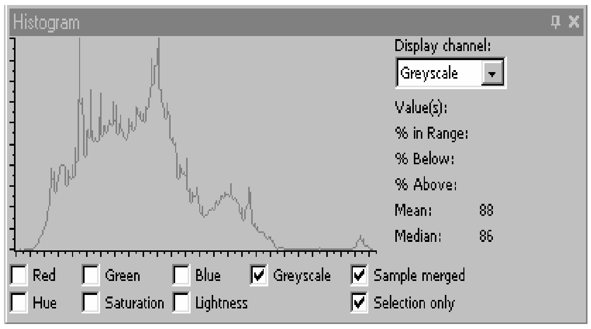
242. The means of *globe.bmp* changes from 206 before the embedding process to 200, and the median decreases from 242 to 190 after the embedding process for all of the *.txt*, *.bat*, and *.rtf* files.



Display channel: Greyscale
Mean: 206
Median: 242

☐ Red ☐ Green ☐ Blue ☑ Greyscale ☑ Sample merged
☐ Hue ☐ Saturation ☐ Lightness ☑ Selection only

Fig.2 The color intensity level of *globe.bmp*



Display channel: Greyscale
Mean: 88
Median: 86

☐ Red ☐ Green ☐ Blue ☑ Greyscale ☑ Sample merged
☐ Hue ☐ Saturation ☐ Lightness ☑ Selection only

Fig.3 The color intensity level of *lab.jpg*

Fig.3 shows the color histogram of *lab.jpg* before embedment with any file format. It applies grayscale in color as for the display channel and sample merge with selection only. The means for the color histogram is 88 and the median is 86. After the embedding process, the display channel is still in grayscale.

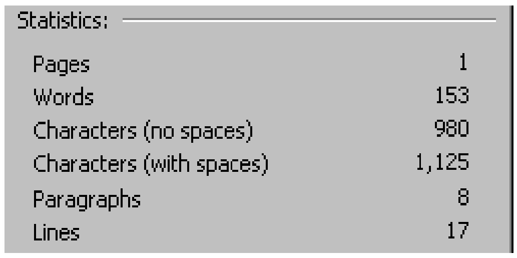
	globe.bmp			
	Before embedded	After embedded		
		.txt file	.bat file	.rtf file
Mean	206	200	200	200
Median	242	192	192	192
	lab.jpg			
	Before embedded	After embedded		
		.txt file	.bat file	.rtf file
Mean	88	103	103	103
Median	86	96	96	96

Table 2**:** Mean and median of image files

However, the amount of mean and median of *lab.jpg* has changed. The means changes from 88 to 103 after the embedding process and the median changes from 86 to 96 for all of the *.txt* files, *.bat* files and *.rtf* files. It proves that the increasing number of pixels and the existence of color changes on the image become obvious on the JPEG file. Table 2 shows the detail changes of the image files.

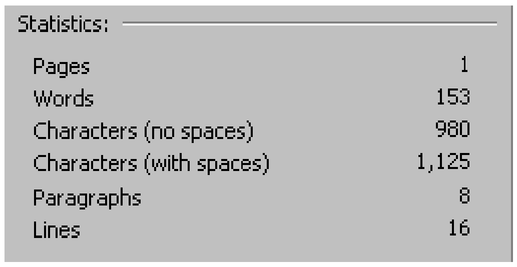
3.3 The Data Integrity

Fig.4 and Fig.5 show a word count output from a one page Microsoft Word file with 153 words, 980 characters with no space and 1125 characters with spaces, 8 paragraphs and 17 lines. The file is used in the embedding process for *globe.bmp* and *lab.jpeg*. After the extracting process, the statistics of the *globe.bmp* file remain constant as before the embedded process. However, the number of lines of the *lab.jpeg* file has decreased to 16 after the extracting process. The other statistics remain unchanged.



Statistics:	
Pages	1
Words	153
Characters (no spaces)	980
Characters (with spaces)	1,125
Paragraphs	8
Lines	17

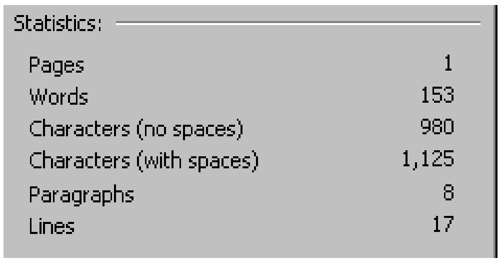
(a) before embedding



Statistics:	
Pages	1
Words	153
Characters (no spaces)	980
Characters (with spaces)	1,125
Paragraphs	8
Lines	16

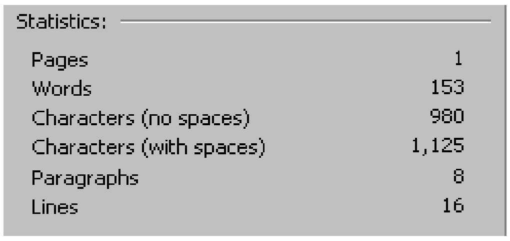
(b) after extraction

Fig.4 The *globe.bmp* word count for the complete embedding and extraction processes.



Statistics:	
Pages	1
Words	153
Characters (no spaces)	980
Characters (with spaces)	1,125
Paragraphs	8
Lines	17

(a) before embedding



Statistics:	
Pages	1
Words	153
Characters (no spaces)	980
Characters (with spaces)	1,125
Paragraphs	8
Lines	16

(b) after extraction

Fig.5 The *lab.jpg* word count for the complete embedding and extraction processes.

Table 3 shows the time taken and the quality of text files for reversing process. It shows that the time of the embedding and the extracting process in *globe.bmp* is exactly equal, which is 00:00:05 seconds. There is no change on the text files structure before and after the extracting process. The time of *lab.jpg* file is 00:00:06 seconds, which is one second longer than *globe.bmp* file in the embedding and extracting process. The text structure of *lab.jpg* file slightly changes but there is no change on *globe.bmp* file.

The number of characters (with spaces) for text format	BMP		JPEG	
	Embedded Time	Extracted Time	Embedded Time	Extracted Time
44	Less than 1 second	Less than 1 second	Less than 1 second	Less than 1 second
109	00:00:01	00:00:01	00:00:01	00:00:01
328	00:00:01	00:00:01	00:00:02	00:00:02
1640	00:00:06	00:00:06	00:00:07	00:00:07
6559	00:00:21	00:00:21	00:00:27	00:00:27

Table 3: The time taken and the quality of text files for both processes.

3.4 Execution Time

As for the execution time, it is based on the number of characters of the text files. The embedding and extracting process time are recorded as shown in Table 4. It shows that the consumption of time is in direct ratio with the size of files for the embedding and extracting processes especially for *lab.jpg* file. Both of JPEG and BMP files sizes change after the embedding process where the change of JPEG files is much more obvious compared to BMP files. Besides that, the color intensity level and the integrity of BMP files are better than JPEG files because the increasing number of pixel images will change the existing color of the image especially on the JPEG file and slight changes especially in the text structure compared to BMP files that has no changes after the embedding and extracting process. The execution time of JPEG files is also longer compared to BMP files and is direct ratio with the size of embedded files. All of these comparisons shows that the BMP files is more capable compared to JPEG files for covering message.

BMP *(globe.bmp)*			
Embedded Process		Extracted Process	
Time Taken	Structure of text	Time Taken	Structure of text
00:00:05	100%	00:00:05	Unchanged
JPEG *(lab.jpg)*			
Embedded Process		Extracted Process	
Time Taken	Structure of text	Time Taken	Structure of text
00:00:06	100%	00:00:06	Slightly changed

Table 4: Record of time consumption to embed and extract the messages.

4 Conclusion

With the advanced technology of information system, the ability of sending secret messages becoming more complex and complicated when there are hackers everywhere. Varieties of tools are developed to ensure the secrecy of the messages whenever data is sent from sender to receiver. Therefore, there has been much researches and study made by various researchers to overcome the matter.

This study seek to find a suitable and appropriate image file to be used as a cover message is an attempt in advancing the technology of steganography to be more specific in its implementation. From the result of the study, we conclude that BMP image file is more suitable and reliable to use as a cover message in steganography, compared to JPEG image file.

For future research, the investigation of the effectiveness and efficiency of image as a cover messages in a copyright environment is suggested.

References:

[1] Anderson, R. J., & Petitcolas F. A., On the limit of Steganography, *IEEE Journal of Selected Areas in Communications*, 16(4), 1998, pp.474-481.

[2] Bender, W., Grhul, D., Morimoto, N., & Lu, A., Techniques for Data Hiding, *IBM Systems Journal*, 35, 1996, pp. 3-4.

[3] Cachin C., An Information-Theoretic Model for Steganography, Proceedings of 2nd Workshop of Information Hiding, Portland, Oregon, USA, Vol.1525, May, 1998.

[4] Davidson, I., Paul, G., & Ravi, S. S., Steganography Using Spatially Interesting Pixels, *Lecture Notes in Computer Science*, 2137, 2004, pp. 289–302.

[5] Droogenbroeck M.V. & Benedett R., Techniques For A Selective Encryption Of Uncompressed And Compressed Images, *Proceeding of Advanced Concepts for Intelligent Vision Systems (ACIVS)*, Ghent, Belgium, September 9-11, 2002, pp. 90-97.

[6] Farid H., Detecting Stenographic Messages in Digital Images, tech. rep., Dartmouth College, 2001. TR 2001-412.

[7] Fridrich J., Goljan M., & Du R., Reliable Detection of LSB Steganography in Color and Grayscale Images, *IEEE Multimedia (Multimedia and Security), Oct-Dec 2001*.

[8] Holmes, P. D., Introduction to Digital Image Steganography, 2002, Retrieved January 28, 2005 from http://www.giac.org/practical/David_P_Holmes_GSEC.doc

[9] Johnson, N. F., Steganography: seeing the unseen. *IEEE Computer*, Feb. 1998, pp. 26-34.

[10] Johnson, N. F., & Jajodia, S., Steaganalysis of images created using current Steganography software. *Lecture Notes in Computer Science*, 1525, 1998, Retrieved December 12, 2004 from, http://link.springer.de/link/service/series/0558/papers/1525/15250273.pdf

[11] Kawaguchi E., & Eason R.O., Principle and Application of BPCS-Steganography, *SPIE Proceedings*,1998.

[12] Kessler G.C., An overview of Steganography for the Computer Forensics Examiner, *Forensic Science Communication*, 2004, pp. 45-48.

[13] Lin M.H., & Hu Y.C., Both Color and Gray Scale Secret Images Hiding in a Color Image, *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.16, No.6, 2002, pp. 697-713.

[14] Lin E. T., & Delp E. J., A Review of Data Hiding in Digital Images, *Proceedings of the Image Processing, Image Quality, Image Capture Systems Conference* (*PICS '99*), April 1999, pp. 274-278.

[15] Nozaki K., Niimi M., Eason R.O., A Large Capacity Steganography using Color BMP Images, *Asian Conference on Computer Vision* (*ACCV*), Hong Kong, January 1998, pp.112-119.

[16] Petitcolas, F. A., Anderson, R. J. & Kuhn, M. G., Information Hiding – A Survey, *IEEE Computers*, 87(7), 1999, pp. 1062-1078.

[17] Provos N., & Honeyman P., Hide and Seek : An Introduction to Steganography, *IEEE Security & Privacy*, Vol. 1(3), May/Jun 2003 pp. 32-44.

[18] Vlan Rabinovich, Steganography – a Cryptography Layer, Oct. 26 1999.

[19] Wang, H., & Wang, S., Cyber warfare: Steganography vs. Steganalysis. *Communications of the ACM*, 47(10), 2004, pp. 76-82.