

A Comprehensive Image Steganography Tool using LSB Scheme

Sahar A. El_Rahman

Electrical Department, Faculty of Engineering-Shoubra, Benha University, Cairo, Egypt
Email: sahr_ar@yahoo.com

Abstract—As a consequence of the fact, transmitting data has been fast and easy these days due to the development of the Internet. Where internet is the most important medium for confidential and non-confidential communications. Security is the major matter for these communications and steganography is the art of hiding and transmitting secret messages through carriers without being exposed. This paper presents a secured model for communication using image steganography. The main concern is to create a Java-based tool called IMStego that hides information in images using Least Significant Bit (LSB) algorithm (1-LSB) and modified Least Significant one Bit algorithm, i.e. Least Significant 2 Bits algorithm (2-LSB). IMStego is a more comprehensive security utility where it provides user-friendly functionality with interactive graphical user interface and integrated navigation capabilities. It provides the user with two operations, which are hiding secret data into images and extracting hidden data from images using 1-LSB or 2-LSB algorithm. IMStego tool hides secret information in color static images with formats BMP and PNG.

Index Terms—Cryptography, Data Hiding, Image Steganography, Information Security, Least Significant Bit

I. INTRODUCTION

The growth of Internet users has raised the possibility of their data lost or modified by a third party. One of the solutions for protecting data from any potential risk is steganography. Users over the Internet nowadays are struggling with keeping their data secured against any attack done by unauthorized people during transmission. Therefore, messages that meant to be revealed only by the intended receiver could be exposed. To protect users' data from being manipulated, steganography is used to hide their data in different types of multimedia. Since images are the most widely used medium today and human visual perception of colors is limited.

Steganography is a protection method driven from Greek words that means "secret writing". This method is used to hide secret information within other unsuspecting ones in such a way that it is impossible to detect its existence. On the other hand, cryptography is concerned about hiding the meaning of the message rather than its existence by using a process called "encryption" [1].

Table. 1 shows a comparison between steganography and cryptography.

Table 1. Steganography vs. Cryptography.

	Steganography	Cryptography
Definition	Hiding the existence of the message	Hiding the meaning of the message
Carrier	Any digital media	Usually text based
Key	Optional	Necessary
Objective	Secrete communication	Data protection
Visibility	Never	Always
Fails when	Detected	De-ciphered

Security is the major matter for the communications over of the internet and steganography is most widely recognized tool for information security [2]. Steganography is the process of hiding information in such a way that prevents the detection of hidden messages. In this technique, no one apart from the sender and the intended recipient even realize that there is a hidden message [3]. it hides a secret message within cover medium such as image, video, text, audio [4]. There are two major types of steganography throughout history, technical and linguistic. Technical steganography is more based upon scientific methods of hiding information while linguistic employs more creative and non-apparent methods [5].

Steganography systems can be grouped by the type of covers used (graphics, sound, text, executable) or by the techniques used to modify the covers [4][6][7]. In image steganography, the cover media is the cover image and the stegomedia is the stegoimage. The images can be manipulated in either spatial domain or frequency domain and accordingly there are two main classes into which the steganography techniques may be divided [8].

There are two directions of steganography (as shown in Fig. 1) protection against detection and protection against removal. The first direction aims to prevent the message from being detected by any unauthorized third party. The second direction is called "document marking" which consist of two branches: "watermarking" that hides trademarks into images, music and other types of multimedia and "fingerprinting" which are hidden serial numbers that enable the owner of an intellectual property to know the costumer who has broken his license agreement [9].

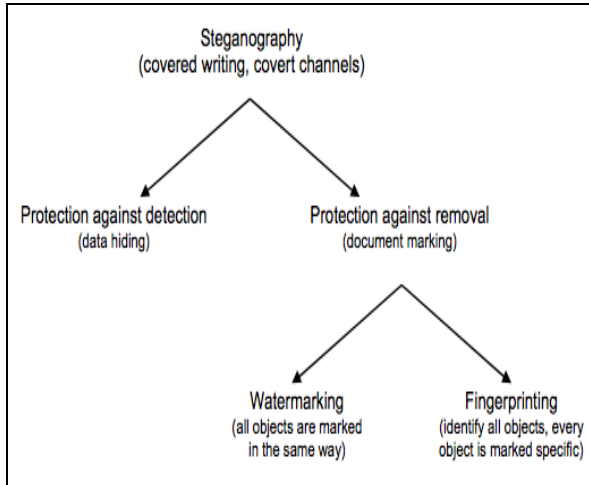


Fig. 1. Steganography directions.

Image steganography has many applications, especially in today's modern, high-tech world. Most people on the internet has a concern about privacy and secrecy. For two parties, image steganography allows to communicate secretly and covertly. For some morally-conscious people is allowed to safely whistle blow on internal actions. Also, it allows for copyright protection on digital files using the message as a digital watermark. One of the other main applications for image steganography is for the high-level or top-secret documents transportation between international governments [4]. Also, it can be used in several fields such as: human rights organization, medical imaging system, military field and correcting media transition errors.

Steganographic techniques have various properties which characterize their strengths and weaknesses. Properties include the following: Security, Embedding capacity, Perceptual transparency, Computational complexity, and Robustness [10][11][12].

A. Steganography General Process

A cover and a message are two basic models required in steganography. The message to be hidden in the cover is known as "embedded message" where the process of hiding the message is called "embedding". A password called "stego-key" might be required for embedding the message. The resulting file is known as "stego-object" which contains a combination of the cover and the embedded message. Both cover and embedded message can be any type of multimedia. The technique of steganography requires the existence of an encoder and a decoder. The encoder is used for hiding data while the decoder is used for extracting data. The following formula and Fig. 2. show the general process of steganography [1].

$$\text{Cover} + \text{Embedded message} + \text{Stego-key} = \text{Stego-object} \quad (1)$$

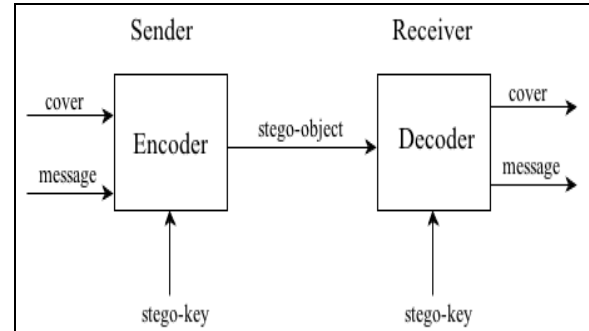


Fig. 2. Generic process of encoding and decoding.

II. RELATED WORK

There are a lot of software products and tools that help in embedding secret messages inside other files. In-order to accomplish the concept of steganography successfully; both of the sender and receiver need to install the same software. The following is a list of some steganography tools:

A. White Noise Storm

White Noise Storm is a DOS based tool that could easily embed secret messages in cover images without any degradation. However, the integrity of the cover image could be severely affected by noise. The tool uses LSB steganography technique to embed secret messages in PCX files. The main disadvantage of this tool is the loss of many bits that can be used to hold information. Additionally it uses large cover images to store information that it could be stored in a smaller cover images using other tools [13].

B. StegoDos

StegoDos is public domain software that only works on 320x200 pixel images with 256 colors. The tool uses LSB steganography technique to hide secret messages. The main disadvantage of the tool is the size restriction that limits the user's cover image to 320x200 pixels in-order to have a stego-image that is similar to the original one. Another disadvantage is the dependence on the end-of-file character to end the message that does not have any significance work since the message after retrieval appears to contain garbage [13].

C. StegCure

StegCure uses three different LSB steganography techniques. In compared with the other tools, StegCure offers a better security and has a user-friendly functionality with interactive graphical user interface (GUI) and integrated navigation capabilities. Also, it can prevent any attacks by restricting the user to one attempt to retrieve the secret message [14].

D. IMStego & Similar Tools Comparison

Table 2. shows a comparison between IMStego tool and other image steganography tools.

Table 2. Image steganography tools comparison

	White Noise Storm [13]	StegoDos [13]	StegCure [14]	IMStego
Capacity of Secret Message	Limited hiding capacity	Limited hiding capacity	Limited hiding capacity	Optimum hiding capacity
Image Size	-	320x200	-	Variable size
Image Format	PCX	Lossless (Ex. GIF and BMP)	GIF	BMP & PNG
Efficiency	Low	Low	Medium	Medium
Shared Key	No	Yes	No	Yes

III. PROPOSED ALGORITHM

In the proposed system LSB substitution is used to embed the message into an image. It works by adjusting the LSB of the carrier image’s pixels whereas, the last bit of each byte in the image is changed to a bit of the secret message that is known standard LSB (1-LSB). Also, we use 2-LSB method that differs from the standard LSB method by allowing more data to be hidden into the cover image. The idea of this method is almost similar to the standard LSB, except that it replaces the 2-LSB of each byte of the cover image instead of one bit.

The LSB insertion differs depending on the number of bits in an image. In 8-bit images, the last bit of each byte in the image is changed to a bit of the secret message. However, it has a major limitation, which is embedding only small size data into images. While in 24-bit images, the last bit of each RGB component is changed which allows more data to be hidden. LSB concept can be described as follows: if the LSB of the cover’s pixel value $C(i,j)$ is equal to the message bit m to be embedded, $C(i,j)$ remain unchanged; if not, set the LSB of $C(i,j)$ to m . The message embedding procedure can be described using the equation below [15]:

$$C_{s i j} = \begin{cases} C(i,j)-1 & \text{LSB}(C(i,j))=1 \text{ and } m=0 \\ C i j & \text{LSB}(C i j) = m \\ C(i,j)+1 & \text{LSB}(C(i,j))=0 \text{ and } m=1 \end{cases} \quad (2)$$

LSB is effective in using BMP and PNG images since the compression in both of them is lossless. However, large in size BMP images are required to be used as a cover. LSB algorithm can also be used with GIF formats, but the main problem is that the whole color palette would be changed whenever the LSB is modified. To avoid this problem, grey-scale GIF images are used since

they contain 256 shades and the changes will be done gradually so that it will be very hard to detect [16].

A. Sender Side

The whole process of hiding message is explained in Fig. 3 and Table 3.

Table 3. The process of message embedding at the sender side.

- 1: Read the text message, which is to be hidden in the cover image.
- 2: Convert each character of secret message to its equivalent ASCII code.
- 3: Convert each ASCII code to its 8 bit binary equivalent.
- 4: Determine the length of the message and add it at the header of message.
- 5: Read the cover image row by row.
- 6: Convert the pixels of the image from decimal to binary.
- 7: Calculate the LSB of each pixel of the cover image.
- 8: Replace the LSB of each byte in the cover image with a bit of the secret message.
- 9: Repeat the replacement process until no bit is left to be embedded.

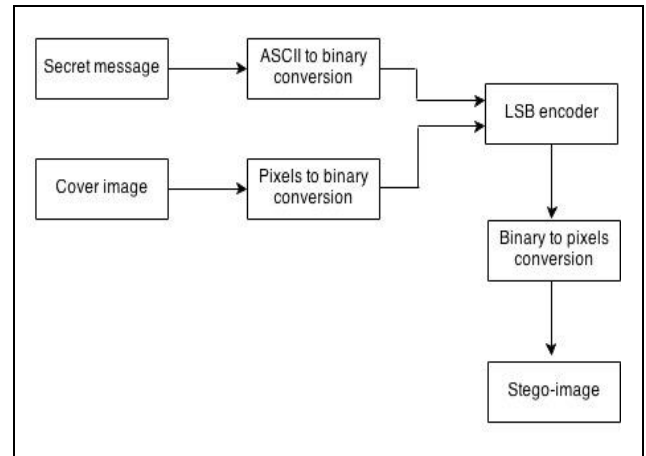


Fig. 3. Entire embedding process at the sender side.

▪ Text to Byte Conversion

Each character of the message to be hidden is converted to its equivalent ASCII, which then converted into bytes [17]. Fig. 4-a shows an example of this type of conversion.

▪ Message Embedding in Digital Image

As mentioned before, each pixel of a 24-bit image has three-color components, which are RGB and their values range from 0-255. The first pixels of the image are used to save the length of the embedded message. To hide the text into the image, first the image is read as an array of pixel values. Then each component of the RGB is converted to 8-bit binary digits [17] as shown in Fig. 4-b .

After that, the LSB encoder will calculate the LSBs of the image's bytes and then replace each of them with a bit of the message [17] as described in Fig. 4-c.

B. Receiver Side

The whole retrieval process is described in Fig. 5-a and Table 4.

Table 4. The process of message retrieval at the receiver side.

1:	Read the stego-image row by row.
2:	Get the length of the message from the first pixels.
3:	Calculate the LSB of each pixel in the stego-image.
4:	Convert each byte into its equivalent ASCII value.
5:	Convert ASCII values to characters to get the secret message.
6:	Repeat until retrieving the entire message.

▪ Retrieve Message

The process of retrieving the message is reversal of the encoding process. The length of the message obtained from the first pixels of the image is used, so the extracting process knows when to stop. The retrieving process can be done by extracting the LSBs of each pixel in the stego-image in one array. Then, each byte is converted back to its equivalent ASCII value, which is stored in a byte array [17]. See Fig. 5-b.

▪ Byte to Text Conversion

The byte array used in the message retrieval process is now converted into message by reading the array byte by byte to get each character in the message [17].

C. Shared Key

To make the algorithm more secure, communicating parties will share a stego-key k to be used as a seed for a PRNG function. This would help the algorithm to spread the bits of the secret message over the cover instead of hiding them sequentially. The output of the PRNG is a random sequence K_1, \dots, K_n where n is the length of message bits. The sequence is then used by the sender to generate a sequence of pixel indices y_i where

$$y_1 = k_1 \tag{3}$$

$$y_i = y_{i-1} + k_i, i \geq 2 \tag{4}$$

Message bit i would then be embedded into the LSB of the pixel y_i and thus the order in which the secret message bits are embedded would be determined pseudo-randomly.

At the receiver side the same stego-key will be used to reconstruct the same random sequence k_i and therefore the entire sequence of pixel indices y_i is obtained to extract the hidden bits of the secret message [18].

IV. TOOL DESIGN AND IMPLEMENTATION

IMStego tool is considered to be a standalone application where it does not require using neither a database nor web server. It is expected to perform two main operations: hiding and extracting. IMStego tool will have the following screens.

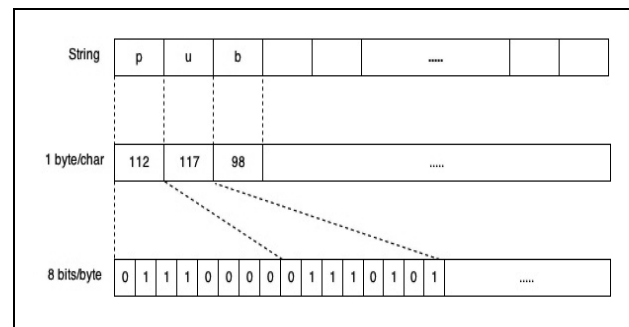
A. IMStego Main Screens

▪ Home Screen

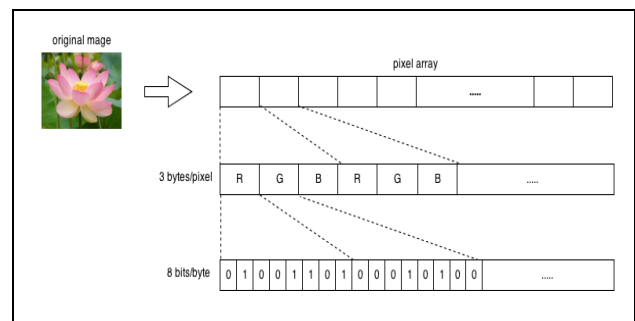
It is the main Home screen that appears to the user when opening the IMStego tool, which contains a description about the tool. It contains a help icon that briefly explains the concept of image steganography and an Exit button to close the tool as shown in Fig. 6-a.

▪ Hide Screen

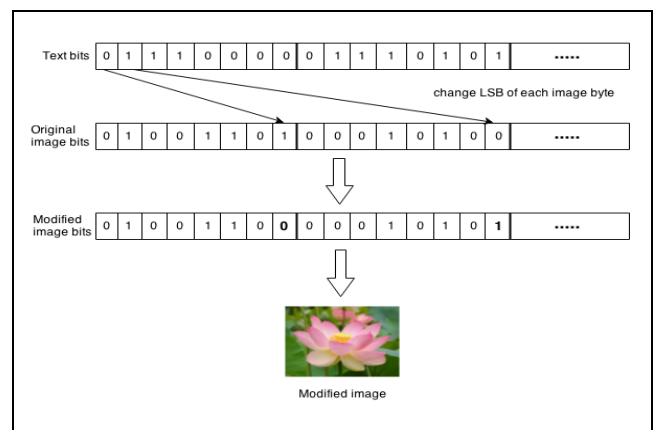
In this screen the LSB embedding algorithm is performed by taking cover image, shared key and secret message as inputs to produce a stego-image as an output. Fig. 6-b shows the ‘‘Hide’’ tab how will look like.



(a) Text to byte conversion.

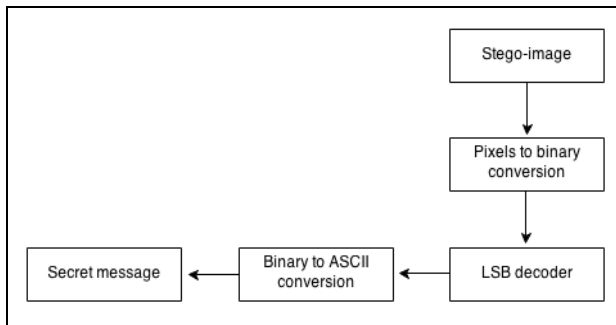


(b) Image conversion.

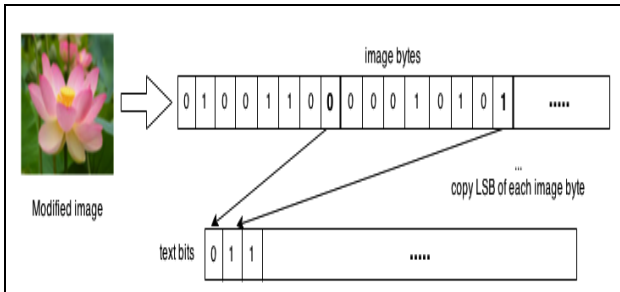


(c) Inserting the message bits into the image.

Fig. 4. Sender side



(a) The entire retrieval process.

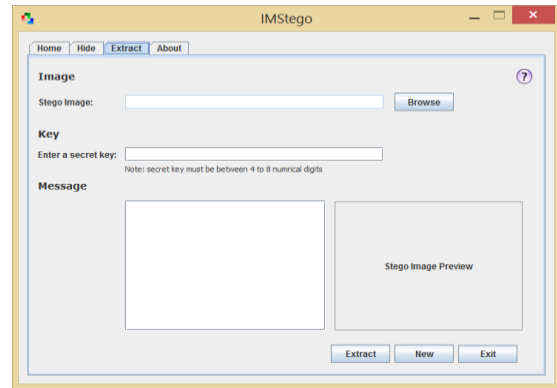


(b) Message retrieval process.

Fig. 5. Receiver side

▪ **Extract Screen**

In this Screen the LSB extracting algorithm is performed by taking the stego-image and the shared key as inputs to produce the hidden message as an output. Fig. 6-c shows the “Extract” tab how will look like.



(c) IMStego Extract Screen

Fig. 6. IMStego main screens

B. Steganography Processes

To implement the LSB algorithm in IMStego, NetBeans software and Java programming language are used. After that, the following steps to complete building the tool.

▪ **Hiding Process**

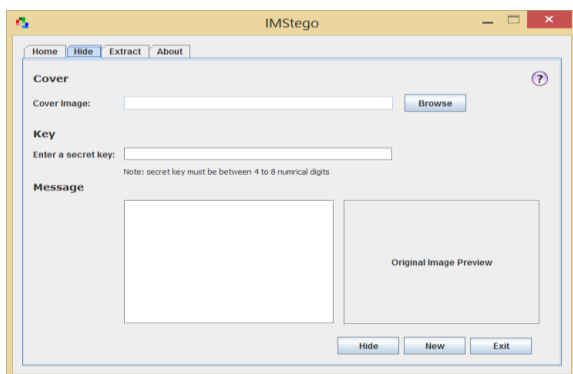
1. The users choose a BMP or PNG image and once the image is selected, a path and a preview of the chosen image will be displayed.
2. The users select the stego method (1-LSB or 2-LSB).
3. They enter 4 - 8 numerical digits as a key that must be shared between the sender and the intended receiver.
4. After that, they will type their message in the specified place.
5. When the users press the “Hide” button the entered key will be validated then the message length will be checked and based on that, one of the following decisions will be made: (a) if the entered key was invalid, an error dialog will be displayed, (b) if the message length exceeded the image length, a warning will pop up to the users telling them that the image is not big enough, and (c) if the message length and the key were acceptable, LSB calculation will take place.
6. Then a popup window will appear to inform the users that the message was hidden successfully.
7. Then they choose a location for saving the stego-image.
8. At the end the stego-image will be displayed.

▪ **Extracting Process**

1. The users choose a BMP or PNG stego-image, a path and a preview of the chosen image will be displayed.
2. The users Select the stego method (1-LSB or 2-LSB).
3. they enter 4 - 8 numerical digits as a key that must be identical to the key entered in the hiding process.
4. When the users press the “Extract” button the entered key will be validated where an error dialog will be displayed if the entered key was invalid, but in case it was acceptable, LSB calculation will take place to extract the length of the hidden message.



(a) IMStego Home Screen.



(b) IMStego Hide Screen

5. If the image contains a message, the LSB calculation will take place to extract the hidden message.
6. The extracted message will be displayed in the specified place.

V. RESULTS

In IMStego tool that uses 24-bit images, since they offer much more flexibility when used in steganography where they allow hiding more information than 8-bit images without affecting the quality of the image. IMStego will be working on BMP and PNG image formats. Based on the comparison between image formats, BMP has the ability to hide large amount of information without causing any visual distortions. PNG format is considered to be one of the most popular formats used on the Internet today.

Also, IMStego tool prompts the users to select stego-method and enter a key for hiding and extracting processes. It provides interactive user interface with the help and notification messages (Error, Informative...) as shown in Table 7 and Table 8. Also, IMStego will not be restricted with a specific image size or number of message characters to be hidden, but the length of the message will be determined, based on the size of the chosen image. For example, if they chose an image with a size of 640x480 the number of message characters to be hidden can be calculated with the following formula: (one character = 8 bits).

$$\text{Number of characters} = ((\text{Number of pixels} * \text{Number of each pixel bits used}) / 8) \tag{5}$$

So, they can hide $(640 \times 480 * 3) / 8 = 115,200$ bytes (or 921,600 bits) by using the 1- LSBs of the RGB values of each pixel. But in case the number of message bits exceeded the number of image LSBs a warning message will pop up to the users telling them that the image is not big enough. Table 5 shows more examples about calculating the size of secret messages for different image sizes.

Table 5. Examples of different image sizes.

Size (Pixels)	MESSAGE	
	Bits	Bytes
800x600	1,440,000	180,000
450x450	607,500	75,937
564x423	715,716	89,464
720x600	1,296,000	162,000

A. IMStego features

1. The original images and the resulting stego-images look alike.
2. Stego-images contain no visual distortions that lead them to be suspected.

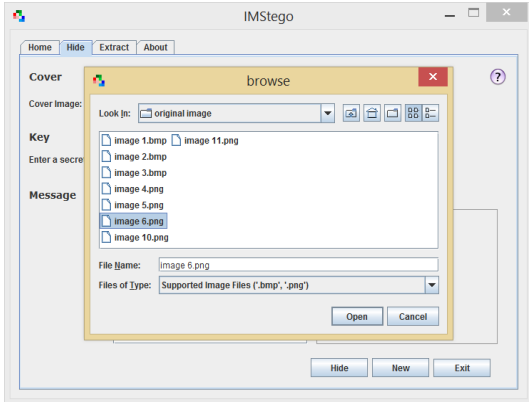
3. The extracted message is the same as the hidden message when the entered key is correct.
4. The time is very small for hiding and extracting processes.
5. it work on Mac OS and Windows platforms.
6. Processing time increases when message length increase.
7. it worked well with different: (a) image formats (BMP and PNG), and (b) image, message and key sizes. (Case studies(1)&(2)&(3) in Fig. 7, Fig. 8, Fig. 9 and Table 6.)
8. there are no differences between the stego-images of both standard LSB and 2-LSB methods. (Case study(4) in Fig. 10 and Table 6.)
9. IMStego provides interactive user graphical interface as described above.

VI. CONCLUSION

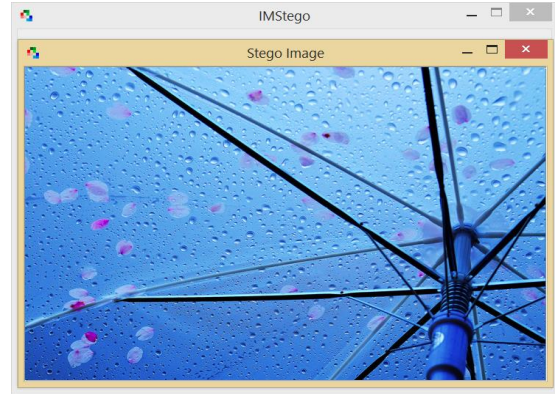
The risks of unauthorized access to secrete information are increasing everyday with the development of new technologies. As a solution image steganography can be used to protect our data from being revealed by unintended receivers. IMStego tool accomplish the Hiding process, Extracting process, Shared key, Variable image size, Support BMP & PNG color image formats, Message integrity, and Multiple message languages using 1-LSB and 2-LSB algorithms. The experimental results show the proposed tool provides acceptable image quality in a way that does not allow anyone to detect their existence and a large message capacity.

Table 6. Case Studies

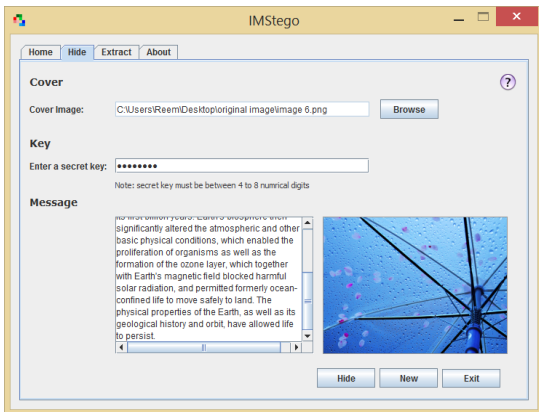
Case Studies (1)&(2)&(3)		
Stego Method: 1-LSB Image Type: RGB Image Format: .PNG Image Size: 706 X 437 Key: 6 digits		
The difference between case studies is message language		
Case Study (1)	Case Study (2)	Case Study (3)
English	Arabic	Japanese
See Fig. 7	See Fig. 8	See Fig. 9
Case Study (4)		
Stego Method: 1-LSB & 2-LSB Image Type: RGB Image Type: .bmp Image Size: 500 X 333 Message Language: English Key: 6 digits		
Fig. 10 shows a comparison between original image and the two methods' stego-images.		



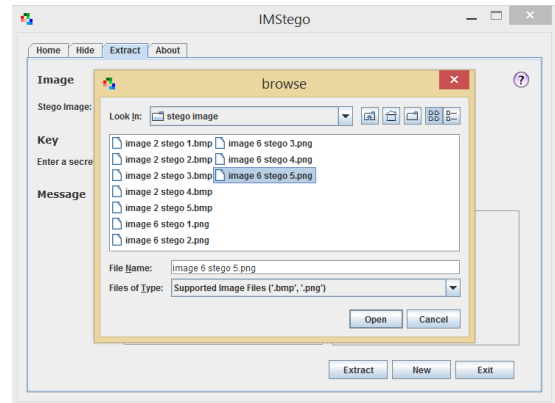
(a) Browsing original image.



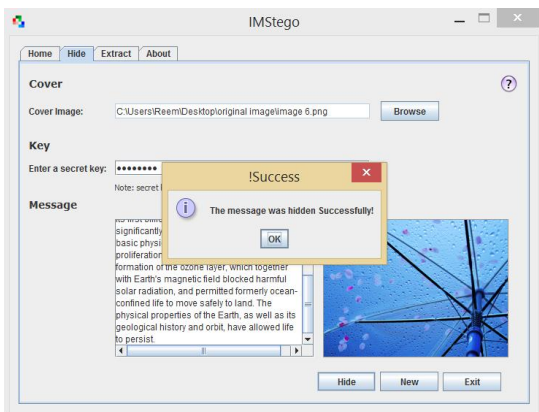
(e) Stego-image display.



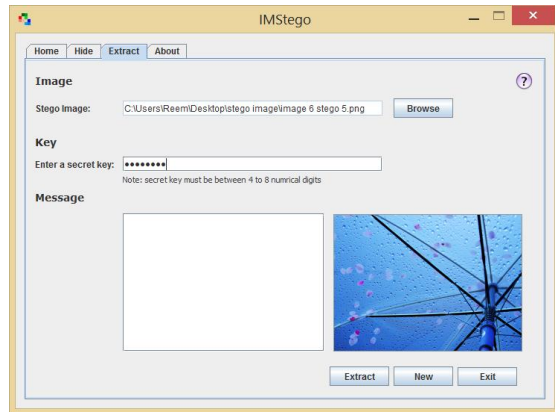
(b) Hiding process.



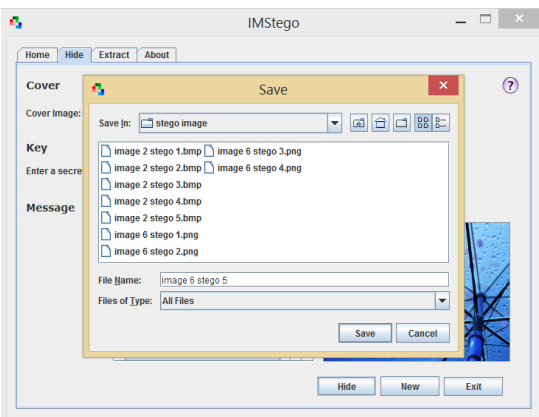
(f) Browsing stego-image.



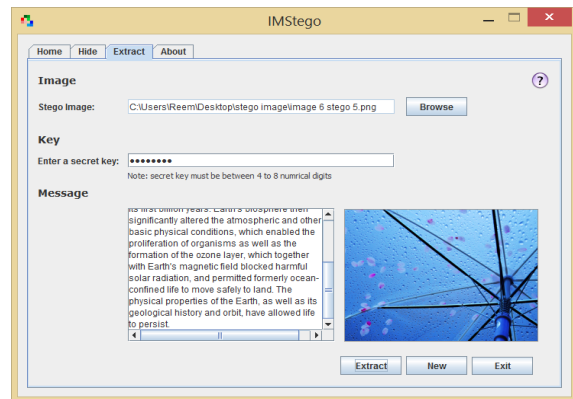
(c) Successful hiding process.



(g) Extracting Process.

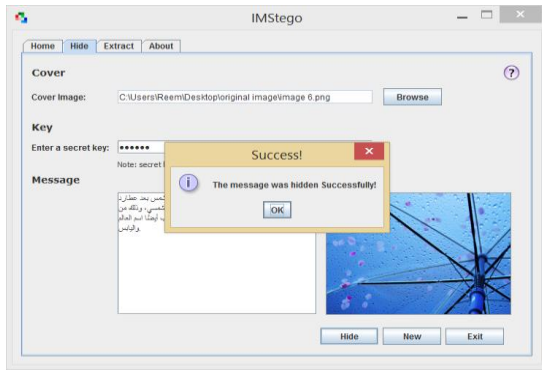


(d) Save window.

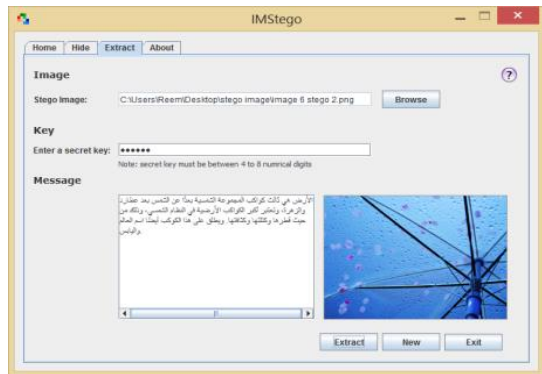


(h) Successful extracting process.

Fig. 7. Case Study (1): Message language(English)



(a) Successful hiding process.



(b) Successful extracting process.

Fig. 8. Case Study (2): Message language(Arabic)



(a) Original image.

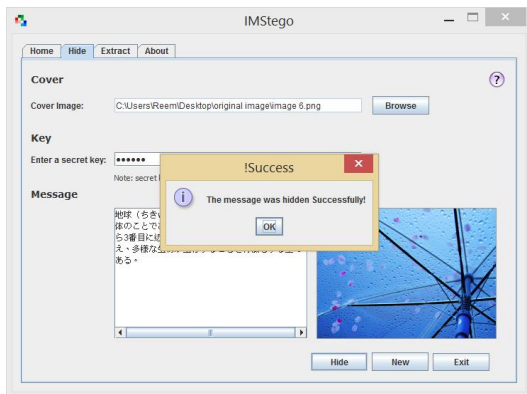


(b) LSB stego-image.

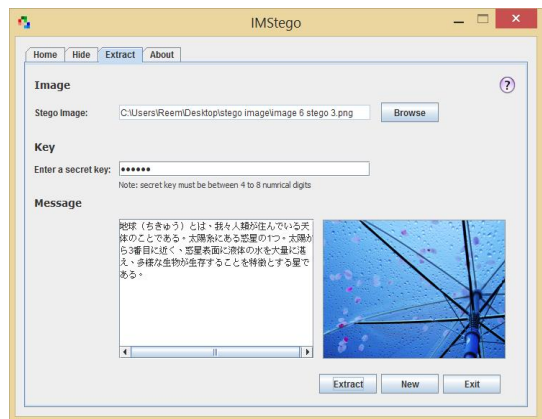


(c) 2-LSB stego-image.

Fig. 10. A comparison between original image and stego-images.



(a) Successful hiding process.





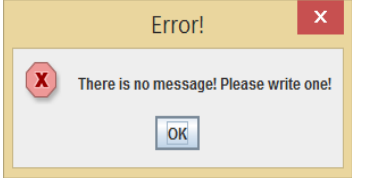
(b) Successful extracting process.

Fig.9. Case Study(3): Message language(Japanese)

Table 7. Notification (Informative) messages

Case Description	Informative Messages
After completing the hiding process successfully the tool will show this message informing the user that the hiding process was completed successfully.	<p>Informative message #1.</p>
Extracting a message from an image that does not contain a hidden message will force the tool to show this message informing the user that the selected image does not have a hidden message.	<p>Informative message #2.</p>

Table 8. Notification (Error) messages.

Case Description	Error Messages
Clicking on the "Hide" or "Extract" buttons without choosing an image will force the tool to show this error message.	 <p>Error message #1.</p>
Clicking on the "Hide" or "Extract" buttons after: (a) entering a key that is not in the range between 4 to 8 digits, (b) entering letters, or (c) special characters will force the tool to show this error message.	 <p>Error message #2.</p>
Clicking on "Hide" button without entering a message will force the tool to show this error message. The user must type a message before performing any hiding process.	 <p>Error message #3</p>

REFERENCES

- [1] A. Kumar, "Steganography: A Data Hiding Technique," *IJCA*, vol. 9, no. 7, Nov. 2010, pp. 19-23.
- [2] Mamta Juneja, "A Covert Communication Model-Based on Image Steganography", *International Journal of Information Security and Privacy (IJISP)*, Volume 8, Issue 1. Copyright © 2014. 19 pages.
- [3] Nirmalya Chowdhury, Puspita Manna, "An Efficient Method of Steganography using Matrix Approach", Copyright © 2012 MECS I.J. *Intelligent Systems and Applications*, 2012, 1, pp32-38. (<http://www.mecspress.org/>) DOI: 10.5815/ijisa.2012.01.04.
- [4] Phad Vitthal S., Bhosale Rajkumar S., Panhalkar Archana R., "A Novel Security Scheme for Secret Data using Cryptography and Steganography", Copyright © 2012 MECS I.J. *Computer Network and Information Security*, 2012, 2, pp36-42. (<http://www.mecspress.org/>).
- [5] Manish Mahajan, Dr. Navdeep Kaur, "Adaptive Steganography: A survey of Recent Statistical Aware Steganography Techniques", Copyright © 2012 MECS I.J. *Computer Network and Information Security*, 2012, 10, pp76-92. (<http://www.mecspress.org/>) DOI: 10.5815/ijcnis.2012.10.08 .
- [6] Shailender Gupta, Ankur Goyal, Bharat Bhushan, "Information Hiding Using Least Significant Bit Steganography and Cryptography", Copyright © 2012 MECS I.J. *Modern Education and Computer Science*, 2012, 6, pp27-34. (<http://www.mecspress.org/>) DOI: 10.5815/ijmecs.2012.06.04.
- [7] M. Husain and M. Hussain, "A Survey of Image Steganography Techniques," *IJAST*, vol. 54, May 2013, pp. 113-124.
- [8] Rajeev Kumar, Khushil K. Saini, and Satish Chand, "A New Steganography Technique Using Snake Scan Ordering Strategy", Copyright © 2013 MECS I.J. *Image, Graphics and Signal Processing*, 2013, 6, pp25-32. (<http://www.mecspress.org/>) DOI: 10.5815/ijigsp.2013.06.04.
- [9] J. Cummins et al., "Steganography and Digital Watermarking," School of Computer Science, The University of Birmingham, 2004, pp. 1-24.
- [10] T. Morkel, J. Eloff and M. Olivier, "An Overview of Image Steganography," in *proc. 5th Annual ISSA Conf.*, Sandton, South Africa, 2005. [Was: Proceedings of the Fifth Annual Information Security South Africa Conference].
- [11] G V Chaitanya, D Vamsee Krishna, L Anjaneyulu, "A 3-Level Secure Histogram Based Image Steganography Technique", Copyright © 2013 MECS I.J. *Image, Graphics and Signal Processing*, 2013, 4, pp60-70. (<http://www.mecspress.org/>) DOI: 10.5815/ijigsp.2013.04.08.
- [12] Amir Farhad Nilizadeh, Ahmad Reza Naghsh Nilchi, "Steganography on RGB Images Based on a "Matrix Pattern" using Random Blocks", Copyright © 2013 MECS I.J. *Modern Education and Computer Science*, 2013, 4, pp8-18. (<http://www.mecspress.org/>) DOI: 10.5815/ijmecs.2013.04.02.
- [13] N. F. Johanson and S.Jajodia. "Exploring Steganography: Seeing The Unseen," *IEEE CJ*, February 1998, pp. 26-43.
- [14] L. Y. Por et al., "StegCure: A Comprehensive Steganographic Tool Using Enhanced LSB Scheme," *WSEAS Transactions on Computers*, vol. 7, no. 8, August 2008, pp. 1309-1318.
- [15] N. Jain, S. Meshram and S. Dubey, "Image Steganography Using LSB and Edge Detection Technique," *IJSCE*, vol. 2, no. 3, July 2012, pp. 217-222.
- [16] F. Choudhury et al., "Analysis of Various Steganography Algorithms and Their Implementation," B.T thesis, North Eastern Hill University, Meghalaya, India, 2011-2012. Available: <http://www.scribd.com/doc/61193138/17> [Accessed: 7 Nov. 2013].
- [17] S. Singh and G. Agarwal, "Use of Image to Secure Text Message With The Help of LSB Replacement," *EIJAEER*, vol. 1, no. 1, May 2013, pp. 200-205.
- [18] K. Hemachandran and S. Laskar, "Steganography Based on Random Pixel Selection For Efficient Data Hiding," *IJCSE*, vol. 4, no. 2, March 2013, pp. 31-44.

Author's profile



Sahar Abd EL Rahman was born Cairo, Egypt, B.Sc. Electronics & communication, Electrical Engineering Department. Benha University, Shoubra Faculty of Engineering, Cairo-Egypt. M.Sc. in an AI Technique Applied to Machine Aided Translation, Electronic Engineering, Electrical Engineering Department, Benha University, Shoubra Faculty of Engineering, Cairo-Egypt, May2003. PHD. in Reconstruction of High-Resolution Image from a Set of Low-Resolution Images, Electronic Engineering, Electrical Engineering Department, Benha University, Shoubra Faculty of Engineering, Cairo-Egypt in Jan2008.