A Novel Steganography-Cryptography System

Mohammed Abbas Fadhil

Abstract — **This paper presents a new suggested Steganography system. This system hides a message into a stego-image by using a mapping table and some other tables to map different values of pixels in the image to the alphabetic letters of the message. One of the strong points in the system is that the system does not make any changes/distortion in the stego-image, where most steganography systems suffer from this point. Also the system tries, in its operations, to mix the properties of some Cryptographic systems to provide additional security to the hidden message. The experiments on the suggested system proved that it is an easy and efficient Steganography system with a good security for the message. Therefore, this system can be considered as a Steganograpgy-Cryptograpgy system and it can be used effectively in these two fields.**

Index Terms — **Security, Substitution, Hiding, Coding, Mapping, Distortion.**

I. Introduction

The goal of Steganography is to hide messages inside other harmless messages in a way that does not allow any enemy to even detect that there is a second message present. In contrast, to cryptography, where the enemy is allowed to detect, intercept and modify messages without being able to violate certain security premises guaranteed by a cryptosystem [1]-[5].

Steganography and cryptography are both intended to protect information from unwanted parties. Both Steganography and Cryptography are excellent means by which to accomplish this but neither technology alone is perfect and both can be broken. It is for this reason that most experts would suggest using both to add multiple layers of security [2]-[4].

Steganography can be used in a large amount of data formats in the digital world of today. The most popular data formats used are (.bmp, .doc, .gif, jpeg, .mp3, .txt and .wav) [1], [6]-[8].

Steganography technologies are very important part of the future of Internet security and privacy on open systems such as the Internet. But as mentioned earlier, it is a good idea to use the properties of Cryptography and Steganography together to provide a very acceptable amount of privacy for anyone connecting to and communicating over these systems [2]-[4], [6], [9].

Dr. Mohammed A. F. Al-Husainy has been an assistant professor in the Department of Computer Science, Faculty of Science and Information Technology, Al-Zaytoonah Private University of Jordan.
PO. Box: 130 Amman (11733) Jordan, Tel.: 962 79 6846110, Fax: 962 6 4291432
E-mails: dralhusainy@yahoo.com , alhusainy@alzaytoonah.edu.jo

II. MATERIALS AND METHODS

In this section, a simple explanation will be given for the suggested system and the necessary materials that are used. An English message text is written by using the alphabetic characters of the English language (which are 26 letters ('a'…'z')). Some other special characters are useful to use in writing messages which are giving the reader a good understanding of the message. Some of these characters that are adopted in this work: ('space character' , '.' , ',' , '(' , ')' , '''). Therefore, the total numbers of characters that are used to write a message (in this work) become 32-characters. This means that we need at least 5-bits to represent these 32-characters in any digital system [1]. Any message, written in English language, contains same order of frequencies of the alphabetic English letters in it. The orders of frequencies (from high frequency to low frequency) of the alphabetic English letter in any English message (and other special characters that are used in this work) are listed in table 1:

Table 1: The order of frequencies of the alphabetical English letters in any message.

0.	' '	8.	'h'	16.	'f'	24.	'x'
1.	'e'	9.	'r'	17.	'g'	25.	'q'
2.	't'	10.	'd'	18.	'y'	26.	'z'
3.	'a'	11.	'l'	19.	'p'	27.	'.'
4.	'o'	12.	'c'	20.	'b'	28.	','
5.	'i'	13.	'u'	21.	'v'	39.	'('
6.	'n'	14.	'm'	22.	'k'	30.	')'
7.	's'	15.	'w'	23.	'j'	31.	'''

A gray scale image is using one of 256 gray scales to represent each pixel in it. This means that we need (1-byte = 8-bits) per pixel to produce ($2^{(8\text{-bits})}$ = 256) gray scales [1].

Now, to understand the phases of the suggested system in this work, we consider the following definitions:

M (Length) = Refers to the message M that we want to hid it in the stego-image, where Length represent the number of characters in the message M (which are indexed from 0…(Length-1)).

I(Size) = Refers to the image I that is used as a stego-image to hid the message M, where Size is the number of pixels in the image I. (we represent the stego-image I as a one dimensional array of pixels (from the pixel at the upper-left corner to the pixel at the down-right corner) and the pixels in the image is read raw by raw. The pixels of the image I are indexed from 0…(Size-1).

C(Size) = Refers to the Boolean code list. C[*k*] = *true* when the pixel of index *k* in the image I is match to the currently processed letter in the message M. Otherwise, C[*k*] = *false*. We must note here that the number of Boolean values (*true*/*false*) that are stored in C is ≤ Size.

The main operations that the suggested system must be done will explain below:

- **Build the Frequency Table FT:** from the image I, find the frequency of each gray value in I and store these frequency (in a descending order) in the FT. Each element **FT[*i*]** represent the frequency of the gray level ***i*** in the image I, where ***i*** from 0 to 255.
- **Build the Mapping Table MT:** from the above table FT, build a MT table of size 32×8. The content in each element of the MT table is as follow, which are calculated by using the equation **MT[*i* × *j*]= *i* + (32 * *j*)**:

MT[*i* × *j*]	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	32	64	96	128	160	192	224
1	1	33	65	97	129	161	193	225
2	:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:	:
30	:	:	:	:	:	:	:	:
31	31	63	95	127	159	191	223	255

From the above MT table, MT[*i*, *j*] represents the value of the pixel at the index *j* that is mapping to the letter, in the message M, has the index *i* in the table 1.

- Now, for the letters in the message M from 0…(Length-1). Check the pixels of the image I from 0…(Size-1) to find the matching between each letter and one of the mapping values (in the MT) for that letter. When the match pixel is found, set *true* in the code list **C[*k*]=*true***. Such that *k* is the index of the matched pixel in I. Otherwise set *false* in the code list **C[*k*]=*false***. The procedure that is doing in this step can be summarized in the following like C++ code:

```
Letter =0;
Pixel =0;
While (Letter < Length)
{
      Found =false;
      x =0;
      While ( (x < 8) && (Found == false) )
      {
                 if ( I[Pixel] ==  MT[M[Letter], x] )
                 {
                    Found =true;
                 }
                 x = x+1;
      }
      if (Found == true)
      {
                  C[Pixel] =true;
                  Letter = Letter +1;
      }
      else
      {
                  C[Pixel] =false;
      }
      Pixel = Pixel +1;
}
```

- When the above procedure is done, we get a code list C of length equal Pixel. Now, convert the Boolean representation of the elements in the code list to the byte representation. For example (where: *T* represent *true* and *F* represent *false*):

  Code list C (Boolean representation): *T F F T F T T T* *F F F F F F T F* ………

  Code list C (byte representation): 151 2 ………

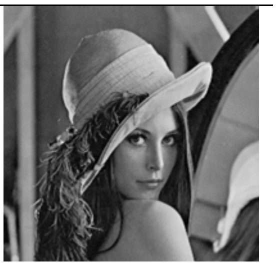
- After that we store the byte representation of the elements in the code list C in the extra (additional) space (bytes) that is usually added (from the operating system) to the end of the most types of files (image files in this work). This extra space is always added by any operating system to make the size of a file of any type fits in block of fixed size (e.g., in kilobyte, megabyte, …). Also we can not use this extra space for storing the code list C, but use any other way that is usually used in most cryptography techniques to exchange the secret key (i.e., the code list C in this work).
- When we want to extraction the original hidden message. We must use the stored code list C in the extra space in the file of the stego-image to do an extraction procedure that is absolutely reverse to the above procedure.

III. DISCUSSION

Usually, all steganography systems try to use some bits of the stego-image to hide the message; this will produce some distortion in the original stego-image. That distortion is the main drawback of any steganography system. But as we can note here, in the suggested steganography system, this drawback is not present in the system. Because of that drawback, any steganography system that is used, we must calculate the Signal to Noise Ratio (SNR) which is referring to the ratio of the distortion that is produced (in the stego-image) from the using of that steganography system. But the suggested steganography system is not using any bit of the stego-image to hide the message; therefore there is no distortion in the stego-image when we use this suggested system and there is no need to calculate the SNR because the original stego-image is unchanged.

To demonstrate how the suggested steganography system work and its performance, we used this system to hide different messages of different length by using some common images. The experiments that are done can be summarized in the following table 2:

Table 2: The experiments on the suggested Steganography system.

Experiment #1: **Image Size:** 512×512 pixels **Message Length:** 2000 letters	
Experiment #2: **Image Size:** 512×512 pixels **Message Length:** 200 letters	
Experiment #3: **Image Size:** 1045×784 pixels **Message Length:** 880 letters	
Experiment #4: **Image Size:** 256×256 pixels **Message Length:** 200 letters	

IV. CONCLUSIONS

In this work, a novel steganography system was presented. The main steps for applying this system are explained. We can conclude the main properties and advantages of this system through the following points:

- When the system builds the FT from the pixels of the stego-image, and then maps these frequencies to the order of frequencies (of the alphabetic English letters) in table 1. This means that the system can produces different code list C for the same message by using different stego-image. This property will add a good security to the message when we using this system.
- As we mentioned in the discussion section, this steganography system did not happen any changes in the original stego-image. This means that whenever any attacker looks the image he cannot see any not normal things in the image and therefore he does not believe that this image hid any secret message in it.
- When the system used the MT for hiding each alphabetic letter of the message, the system might be map different pixel values from each raw of the MT table to the same alphabetic letter from the message. This property is similar to the substitution operation that is done in the cryptography systems. This will support the security of the hidden message and will add additional difficulty in front of the attacker.
- By make some few changes in the order of elements in table 1. This will add another strong property to the privacy of the hidden message. This operation is similar to the use of different substitution tables in hiding the same message.
- Because the suggested system is not changing the original stego-file, this means that the system is adequate for hiding any message in any another types of files like (.doc, .txt, .mp3, etc.). Just the stego-file must have enough extra byte at the end of it.
- At the last, we must note that the types of the stego-files that are used in this work (i.e., image file type) are just use here to demonstrate the idea and the operations of the suggested system. But the system is not restricting to this type of stego-file.

REFERENCES

[1] Al-Husainy Mohammed A. F., "Image steganography by mapping pixels to letters", Journal of Computer Science, Vol. 5, No. 1, 2009, pp. 33-38.

[2] Mehdi Kharrazi, Husrev T. Sencar, Nasir Memon, "Performance study of common image steganography and steganalysis techniques", Journal of Electronic Imaging, Vol. 15, No. 4, 2006, pp.041104.

[3] Najib A. Kofahi, Turki Al-Somani and Khalid Al-Zamil, "Performance study of some symmetric block cipher algorithms under linux operating system", Journal of

Discrete Mathematical Sciences & Cryptography, Vol. 7, No. 3, 2004, pp. 359-370.

[4] Shujun Li and Xuan Zheng, "Cryptanalysis of a chaotic image encryption method", The 2002 IEEE International Symposium on Circuits and Systems (ISCAS 2002), Scottsdale, Arizona, Proceedings of ISCAS 2002, Vol. 2, 2002, pp. 708-711.

[5] N. Provos and P. Honeyman, "Hide and seek: an introduction to steganography", IEEE Security & Privacy, pp. 32-44.

[6] Ramani K., Prasad E. V. and Varadarajan S., (2007), "Steganography using BPCS to the integer wavelet transformed image", International Journal of Computer Science and Network Security, Vol.7, No.7, 2003, pp.293-302.

[7] Adnan Abdul-Aziz Gutub, and Manal Mohammad Fattani, "A novel Arabic text steganography method using letter points and extensions", Proceeding of World Academy of Science, Engineering and Technology, Vol. 21, 2007, pp. 1307-6884.

[8] K. Gopalan, (2003), "Audio steganography using bit modification", Proceedings of the IEEE International Conference on Acoustics, Speech, and SignalProcessing, (ICASSP '03), Vol. 2, 2003, pp. 421-424.

[9] K. Rabah, "Steganography-the art of hiding data", InformationTechnology Journal, vol. 3, Issue 3, 2004, pp. 245-269.