Steganography and Hiding Data with Indicators-based LSB Using a Secret Key

Wesam Saqer
Faculty of Information Technology
Islamic University of Gaza
Gaza, Palestine
wesam.sqr@gmail.com

Tawfiq Barhoom
Faculty of Information Technology
Islamic University of Gaza
Gaza, Palestine
tbarhoom@iugaza.com

***Abstract*—Steganography is the field of science concerned with hiding secret data inside other innocent-looking data, called the container, carrier or cover, in a way that no one apart from the meant parties can suspect the existence of the secret data. There are many algorithms and techniques of concealing data. Each of which has its own way of hiding and its own advantages and limitations. In our research we introduce a new algorithm of hiding data. The algorithm uses the same technique used by the Least Significant Bit (LSB) algorithm which is embedding secret data in the least significant bit(s) of the bytes of the carrier. It differs from the LSB algorithm in that it does not embed the bytes of the cover data sequentially but it embeds into one bit or two bits at once. Actually it depends on indicators to determine where and how many bits to embed at a time. These indicators are two bits of each cover byte after the least two significant bits. The advantage of this algorithm over the LSB algorithm is the randomness used to confuse intruders as it does not use fixed sequential bytes and it does not always embed one bit at a time. This aims to increase the security of the technique. Also, the amount of cover data consumed is less because it sometimes embeds two bits at once.**

Keywords-Steganography; Steganographic; Least Significant Bit; LSB; Data Hiding; Data Concealing; Data Covering

I. INTRODUCTION

The idea of hiding secret data is not new, but after the extreme development of the information technology field, steganography became widely used in digital fields and its techniques are developed more and more day by day. Steganography is a branch of the information security field. Another field of information security is cryptography. Steganography and cryptography are somehow related as they both intend to protect information from unauthorized parties. The main difference between them is that steganography is concerned with hiding data and hiding the existence of the data, where cryptography is concerned with transforming data into a not understood form [1-4]. So steganography is concerned with not detecting secret data whereas cryptography is concerned with not revealing the visible secret data by altering the structure of the data itself [5, 6]. Of course, they can be combined to make data more secure. Data can be encrypted (cryptography) and then be hidden within a carrier (steganography) to provide another layer of protection [4]. Steganography may also be used for watermarking to protect the ownership of copyrighted products [5].

The key purpose of every new steganography technique is to enhance the security or the capacity. Security is enhanced by enhancing the imperceptibility or the robustness. Imperceptibility depends on the quality of the resulting stego-object to be as the original by avoiding making noticeable changes to the cover object [2, 3]. Robustness depends on making the algorithm withstand against steganalysis and attacks [6]. As an example, one of the ways to hide data is just to append its bytes after the bytes of the container file then simply pack all these bytes in one individual file. However this approach has a very big defect which is the size of the result file. This is the sum of the sizes of both the secret and the carrier, which may arouse suspicion if the final size is too large.

A lot of algorithms and techniques have been developed in this scope. Some depend on the nature of the carrier such as the algorithms which hide into photos, audios or other [1, 7], and some can be used with different types of carriers. The carrier of the secret data can be any of many kinds of mediums, like image, audio, video, text and even network protocols [1-3, 7]. So steganography can be categorized according to the cover medium. There is image steganography, which is the most used, audio, video and even network steganography [2]. In our work we focus on developing a new algorithm that can be used with several types of cover mediums, because it depends on the Least Significant Bit (LSB) which can be used with several types. Steganography can also be categorized into three types which are Pure Steganography, Secret Key Steganography and Public Key Steganography [1].

II. LEAST SIGNIFICANT BIT

This work uses the same principle with a famous algorithm called Least Significant Bit (LSB). In the LSB algorithm secret data is hidden in the least bit of each byte of the container data. A byte consists of eight bits. LSB takes replaces the least bit of a series of eight bytes of the cover medium with the secret bits. Consequently every secret byte consumes eight cover bytes to get embedded into [2]. An example is shown in Figure 1. The 01101010 byte is hidden using the LSB algorithm.

A: The series of cover bytes before the embedding process

0	1	2	3
10111000	**10000001**	**10100001**	**01111100**
4	5	6	7
01110110	**10010000**	**11010011**	**01010010**

B: The resulting bytes after the embedding process

0	1	2	3
1011100(0)	**1000000(1)**	**1010000[0]**	**0111110[1]**
4	5	6	7
011101(0)	**1001000[1]**	**1101001(1)**	**0101001(0)**

A: Cover bytes before the embedding
B: Resulting bytes after the embedding
(▪) The bit new value is identical to its original
[▪] The bit new value is different from its original

Fig. 1. Data hiding using LSB

There are several advantages for this algorithm. First it doesn't affect the size of the cover data because it does not increase or decrease the number of the cover data bytes. It just replaces some of the cover data bits with secret bits without affecting the size. It does not make noticeable changes to the cover data and this is according to two factors. Firstly the change occurs in the least bit (rightmost) which has the least weight between all the bits in a byte. Simply if the value of this bit is changed, the value of the byte is changed only by 1 increasingly or decreasingly as shown in Figure 1 at byte 3 and 2. This amount of change is neither noticeable by human eye nor ear. Secondly some of the bits are replaced with its same value as shown in Figure 1 above, for example byte 7. Thus, on average, half of the cover bits are modified using the maximum cover size [2, 4, 6]. So, if we hide some data into, for example, an image the change would not be detectable by human eye [2].

On the other the hand this algorithm has two disadvantages, it consumes a big amount of bytes to embed a small amount into, as for each secret byte it needs 8 cover bytes, but this disadvantage is not that severe. One just needs to hide secret data into a suitable-sized cover. Further, secret data can be compressed. The next issue is that the mechanism of hiding is simple, so it is easy to detect (and be "broken") [2]. We addressed the simplicity issue in this work by making the process of determining the bytes to contain the data involve randomness. Also the proposed algorithm hides two bits sometimes and one bit other times, and this reduces the amount of cover bytes needed by 25% and increases the complexity of the hiding process.

III. RELATED WORK

A. *Image Steganography*

Image steganography techniques can be applied in the Image Domain and Transform Domain [2, 6, 7]. At image domain we can use BMP and GIF, and at transform domain we can use JPG images to hide the data [2, 7]. Other techniques as spread spectrum and patchwork are used too for hiding inside images [2, 6]. LSB is more suitable for BMP, PNG and GIF [5, 8], but GIF needs extra care when it is used for hiding data [6, 8]. Also images with a large size, as BMP, are relatively uncommon on the internet, so using such images would attract attention during transmission [4]. A good choice is to use GIF and many steganography experts recommend using gray-scale images because the shades change very gradually from byte to byte [4].

Many approaches to enhance LSB-based image steganography have been proposed. In [9], using a secret key to decide the appropriate positions within the image to hide data into was proposed. This would make it difficult to retrieve the hidden data in absence of the secret key. The bits were also hidden inside only green or blue pixels, so the modification of the cover image is less than modifying all of the bytes and that increases the quality of the stego-image. In [10], the RC4 algorithm, which uses a stego-key to randomize the embedding of the secret data over the entire cover image, was applied. This approach makes it hard to retrieve the data without having the stego-key. A bit-inversion technique to improve the stego image quality was also introduced in [10]. In [11], LSB was used to hide the data depending on a filtering based algorithm. This filtering requires finding out which pixels are greater in number, the lighter or darker pixels, by checking the 3 MSBs of the pixels. The embedding is done in the dominant area. They also proposed to encrypt the data using ASE before the embedding process.

B. *Image Steganography*

Audio and video is considered good carriers because of redundancy [12]. Many researchers used the LSB modification technique with improvement to hide data inside audio mediums. In [12] two techniques were introduced. The first is Bit Selection which is used to select the cover bits within a sample. The second is Sample Selection which is used to determine the cover samples. These techniques add randomness and confusion to the hiding process to increase the security.

C. *Video Steganography*

Video is a combination of images and audio, so both image steganography and audio steganography can be used. It has much capacity and it is hard to detect small changes in the whole video stream [13]. LSB is an efficient method for video steganography. In [14], a new method that according to a function was introduced that assigns a frame of the cover video to be the index frame. This frame is used to locate the frames that would contain the secret data.

D. *Network Steganography*

Some techniques of hiding use the network protocol functions associated with the OSI layers to hide the secret data into. These protocols such as ARP, TCP, UDP or ICMP protocols, referred to as carrier-protocols [15].

IV. INDICATORS-BASED LSB

In this approach of concealing data the secret data bits are hidden inside the least one or two bits but this process is done depending on indicators. We use an indicator to determine the byte into which the secret bit(s) are embedded and another indicator to determine how many bits are embedded at a time.

The indicator is a fixed bit in every byte of the cover bytes other than the least two bits, because the least two bits are used to contain the secret data. Let's assume the indicator which tells us where to embed is the fourth bit (the bit with index 3 from the right). According to the indicator bit value we determine the byte to hide our secret data into. If the value of the indicator bit value is zero, our current secret bit(s) will be embedded into some byte of the previous bytes before the indicator byte itself (the byte that we do the embedding operation according to its indicator bits values), and if the value of the indicator bit is 1, our current secret bit(s) will be embedded into exactly the next byte after the indicator byte. Through the hiding process the fourth bit of every byte of the cover data is scanned to identify where the current secret bit(s) of the secret data should be hidden. This procedure is done starting from the second byte as an indicator to ensure that there is a previous byte, and the final indicator byte is the one before the last byte of the cover data to ensure there is a next byte.

When the hiding process is directed to embed into the next byte according to the value of the indicator, the embedding is done into exactly the next byte after the indicator byte. But when the embedding is aimed to be done into the previous byte, the previous byte is not always the byte exactly before the indicator byte. Most times, it precedes the indicator byte by a random number of bytes. In the beginning of the hiding process, the first byte is the previous byte and the second byte is the indicator byte. As long as the indicator bytes hide into the next byte, the first byte stays the previous. When an indicator decides to hide into the previous byte, that previous byte will be hidden into, and the next unused byte will be the previous byte. Simply in the beginning of the hiding process the first indicator is the second byte (byte with index 1). The previous byte is the first byte (byte with index 0), which can be called current-previous. Also, the byte that will be the previous byte after using the current-previous byte should be determined. This byte is called the next-previous and at the beginning of the process it is the second byte, just like the indicator. Now as long as every indicator byte is hiding into the next byte, the current-previous and the next-previous never change. When some indicator byte decides to hide into the previous byte, the embedding is done into the current-previous, then the current-previous moves to the next-previous and the next-previous moves to the next byte after the indicator byte. That means every time after embedding is done into the current-previous, the next-previous becomes the current-previous and the next byte after the indicator byte becomes the next-previous. Finally the indicator just steps ahead one byte and so on.

Because we use indicators to identify the byte to hide into, this procedure increases the randomness and confusion of the hiding process, because we sometimes embed into the previous byte and other times into the next byte. And since the previous byte is not fixed for every indicator byte, extra randomness and confusion are added to the process. So this approach makes the process of retrieving the hidden data more complex which increases the security, and this is the main aim of the proposed approach. Also through the hiding process, not always only one bit at a time is embedded (one or two bits may be embedded). This is done depending on another indicator of the indicator byte. Again let's assume the indicator bit which tells us how many secret bits to embed at a time is the third bit (the bit with the index of 2 from the right). If the value of the indicator is 0, we embed only one bit, and if it is one, we embed two bits into the cover byte. This operation adds more randomness and confusion to the hiding process because the amount of the embedded bits is not fixed, so we can't identify the number of embedded bits into each byte without checking the indicator. On the other hand this operation increases the capacity of the hiding process over the normal LSB which embed only one bit at a time and this is another advantage besides increasing the randomness and confusion. Hiding two bits at once causes change to the byte value, but this change ranges from 0 to 3 at maximum, which is a very small change and is not noticeable by the human eye or ear.

So, all bytes of the cover data are scanned and the values of the indicator bits (the fourth and the third bits) in every byte are checked. The fourth bit is the indicator that determines where to hide. If its value is zero, then hiding is done into some previous byte, and if it is one, into exactly the next byte after the indicator byte. The third bit is the indicator that tells us how many bits to hide. If its value is zero, then only one bit is hid, and if it is one, two bits are hid. So, according to the indicator we sometimes hide into the previous and other times into the next. The previous is not fixed since it is not always the byte exactly before the indicator byte, it could be any one of the bytes before the indicator byte. According to another indicator we sometimes hide one bit and other times we hide two bits. Figure 2 shows the flow chart of the hiding process. All of these factors increase the randomness and confusion of the hiding process, which makes it hard to retrieve the secret data by unauthorized parties.

Suppose we want to hide the following bits 11101100, 01100110 into the below series of bytes using indicators-based LSB algorithm as shown in Figure 3. The hiding process is not done sequentially like plain LSB. The secret bits are hidden into cover bytes randomly depending on the values of the indicator bits of the cover bytes. As shown in Figure 3, the order of the cover bytes that were embedded into is 0, 3, 4, 1, 2, 5, 8, 6, 10, 7 and the amount of embedded bits in the same order is 1, 1, 2, 2, 1, 1, 2, 2, 2, 2. Here we can realize that the hiding process is not sequential unlike normal LSB approach and some bytes contain only one secret bit and others contain two bits.

V. INDICATORS-BASED LSB WITH SECRET KEY

To make the algorithm more robust and secure, a secret key is used through the hiding process. The secret key is a series of bits which can be represented with one dimensional circular array (so any size can be used). At each cycle of the hiding process, the position indicator bit, which is used for deciding the byte into which the secret bit is embedded, is XORed with the current bit of the secret key. If the resulting value is zero, then a byte is chosen for hiding. Otherwise, the next byte is chosen. Also, the amount indicator bit, used to decide the number of bits to be emdedded, is XORed with the next bit of the secret key, and if the result is zero then one bit is hidden (otherwise two). This way, security is further increased.

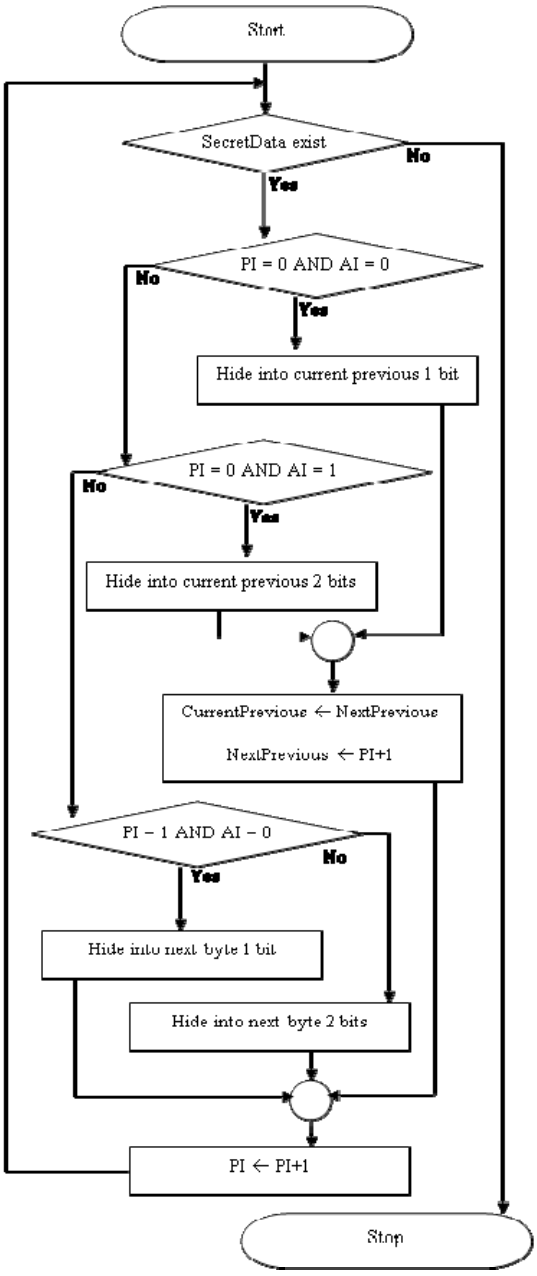

Fig. 2. The hiding flow chart, PI : Position Indicator, AI : Amount Indicator

A: The series of cover bytes before the embedding process			
0	1	2	3
10111000	10000001	10101001	01111100
4	5	6	7
01110110	10010000	11010001	01011111
8	9	10	11
01010101	10111101	11010111	11110000

B: The resulting bytes after the embedding process			
0	1 : 0 – 0	2 : 3 – 1	3 : 4 – 01
1011100(0)	100000[10]	1010100(1)	0111110[1]
4 : 1 – 10	5 : 2 – 1	6 : 5 – 0	7 : 8 – 00
011101[01]	1001000(0)	110100[11]	010110(11)
8 : 6 – 11	9 : 10 – 10	10 : 7 – 11	11
010101(00]	10110001	110100(10]	11110000

A: Cover bytes before the embedding
B: Resulting bytes after the embedding
■ : ■ - ■ Indicator byte : where to embed – what to embed
(■) The bit new value is identical to its original
[■] The bit new value is different from its original
(■■] Only the right bit new value is different from its original
[■■) Only the left bit new value is different from its original
(■■) Both new values are identical to the original
[■■] Both new values are different from the original

Fig. 3. Data hiding using Indicator-based LSB

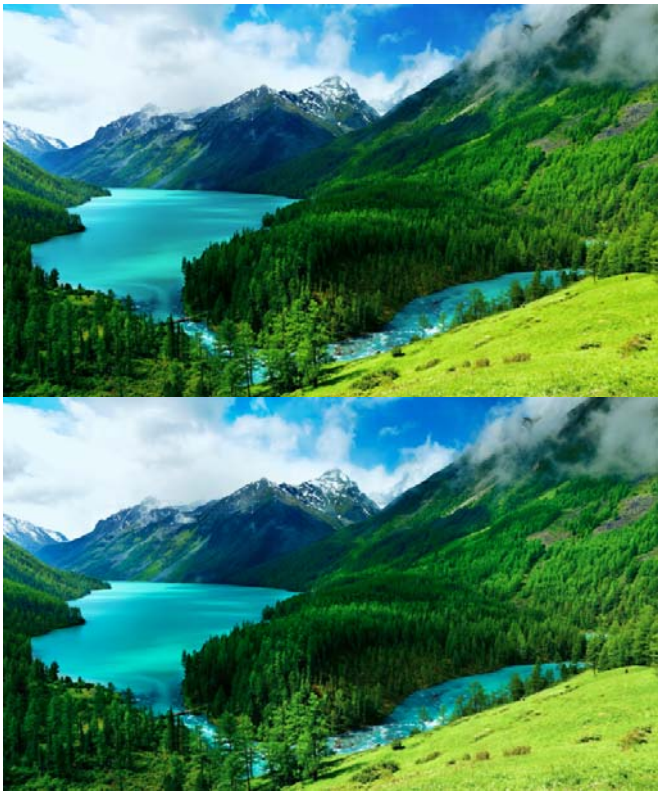

Fig. 4. The cover image before (up) and afetr (down) the hiding process

VI. EXPERIMENT AND RESULTS

This approach was implemented and tested on photos and audios without any noticeable or detectable distortion. This is because the approach uses the LSB technique for hiding data, and this technique does not make any noticeable modification in the carrier medium. A carrier image before and after the hiding process is shown in Figure 4 whereas the corresponding histograms are shown in Figure 5. The approach was also applied on an audio file. Figure 6 shows the audio waveform before and after the process.

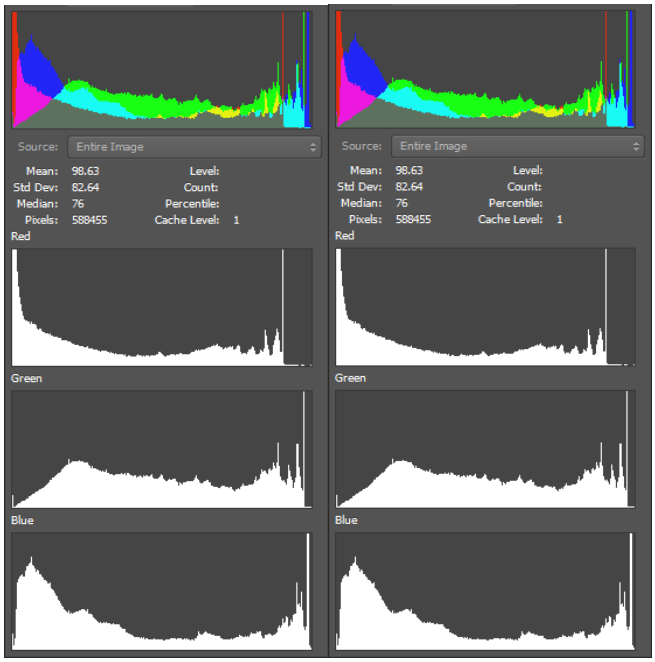

Fig. 5. The cover image histogram before (left) and after (right) the hiding process

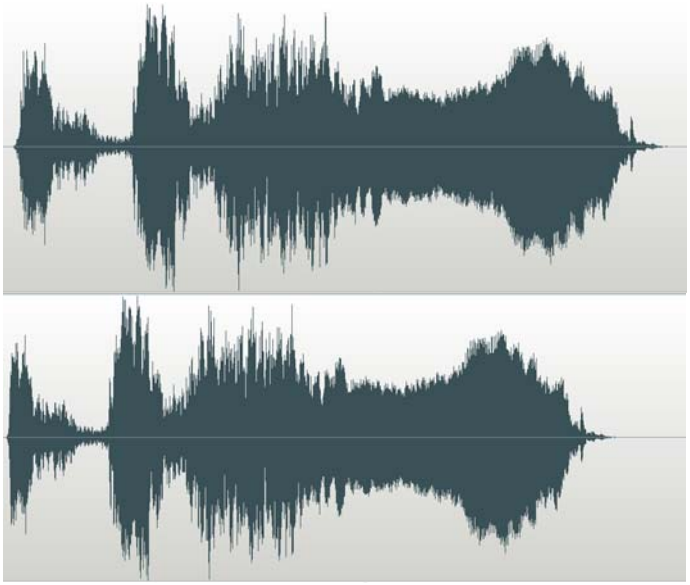

Fig. 6. The cover wave form before (up) and after (down) the hiding process

VII. CONCLUSION

As the development of information and communication technologies are increasingly growing day by day, the need to protect private information is also growing. A branch of information security concerned with hiding secret data called steganography has surfaced and a lot of approaches have been proposed. In our work, we introduced a new approach of data hiding based on the LSB technique with the additional advantage, compared to plain LSB, that it adds much randomness and confusion to the data hiding process which makes it harder for unauthorized parties to detect and retrieve the hidden data. This is accomplished by not employing sequential hiding. The proposed algorithm also shows increased capacity compared to plain LSB as it sometimes hides two bits at once.

VIII. REFERENCES

[1] B. Dunbar, "A detailed look at Steganographic Techniques and their use in an Open-Systems Environment", Sans Institute, Vol. 2002, pp. 1-9, 2002

[2] T. Morkel, J. H. Eloff, M. S. Olivier, "An overview of image steganography", ISSA, pp. 1-11, 2005

[3] R. Krenn, "Steganography and steganalysis", available at http://www.krenn.nl/univ/cry/steg/article.pdf

[4] N. F. Johnson, S. Jajodia, "Exploring steganography: Seeing the unseen", Computer, Vol. 31, No. 2, pp. 26-34, 1998

[5] K. Thangadurai, G. Sudha Devi, "An analysis of LSB based image steganography techniques", International Conference on Computer Communication and Informatics (ICCCI), pp. 1-4, India, January 3-5, 2014

[6] H. Wang, S. Wang, "Cyber warfare: steganography vs. steganalysis", Communications of the ACM, Vol. 47, No. 10, pp. 76-82, 2004

[7] J. Silman, "Steganography and steganalysis: an overview", Sans Institute, available at https://www.researchgate.net/publication/242743284_Steganography_and_Steganalysis_An_Overview

[8] D. Neeta, K. Snehal, D. Jacobs, "Implementation of LSB steganography and its evaluation for various bits", 1st International Conference on Digital Information Management, pp. 173-178, Portugal, September 19-21, 2006

[9] S. M. Karim, M. S. Rahman, M. I. Hossain, "A new approach for LSB based image steganography using secret key", 14th International Conference on Computer and Information Technology (ICCIT), pp. 286-291, Bangladesh, December 22-24, 2014

[10] N. Akhtar, P. Johri, S. Khan, "Enhancing the security and quality of LSB based image steganography", 5th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 385-390, India, September 27-29, 2013

[11] M. R. Islam, A. Siddiqa, M. P. Uddin, A. K. Mandal, M. D. Hossain, "An efficient filtering based approach improving LSB image steganography using status bit along with AES cryptography", 3rd International Conference on Informatics, Electronics & Vision (ICIEV), pp. 1-6, Bangladesh, May 23-24, 2014

[12] M. Asad, J. Gilani, A. Khalid, "An enhanced least significant bit modification technique for audio steganography", International Conference on Computer Networks and Information Technology (ICCNIT), pp. 143-147, Pakistan, July 11-13, 2011

[13] P. Yadav, N. Mishra, S. Sharma, "A secure video steganography with encryption based on LSB technique", IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), pp. 1-5, India, December 26-28, 2013

[14] R. Balaji, G. Naveen, "Secure data transmission using video Steganography", IEEE International Conference on Electro/Information Technology (EIT), pp. 1-5, USA, May 15-17, 2011

[15] J. Lubacz, W. Mazurczyk, K. Szczypiorski, "Principles and overview of network steganography", IEEE Communications Magazine,Vol. 52, No. 5, pp. 225-229, 2014