Journal of Mathematics and Statistics

Original Research Paper

Randomized Pixel Selection for Enhancing LSB Algorithm Security against Brute-Force Attack

[1]Ammar Y. Tuama, [2]Mohamad A. Mohamed, [3]Abdullah Muhammed and [3]Zurina M. Hanapi

[1]University of Information Technology and Communications, Baghdad, Iraq
[2]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Besut, Terengganu, Malaysia
[3]Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang, Selangor, Malaysia

Article history
Received: 07-08-2016
Revised: 24-11-2016
Accepted: 16-05-2017

Corresponding Author:
Mohamad A. Mohamed
Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, Besut, Terengganu Malaysia
Email: mafendee@unisza.edu.my

Abstract: Steganography is the science of concealing a secret message by embedding it into innocent carriers such as text, audio, images, etc. It plays a crucial role in a broad range of security applications such as securing message exchange, user authentication and copyrighting. One of the most effortless and widely-used techniques is the age-old Least Significant Bits (LSB) algorithm, which can be implemented in both transformative and spatial domains. The advantage of this technique is that it can be used with any form of digital media. However, operating pixels on a sequential basis leaves the algorithm susceptible to many steganalysis techniques. Consequently, it is easy for the attacker to recognize the inclusion of a secret message within the media and thus to proceed with the extraction. Therefore, it is necessary to provide an extra layer of security to protect the data. In this study, we propose a random selection of pixels that hold a secret message based on an integer solution of the elliptic curve equation. In addition, we have embedded noise bits into the unused pixels to make the steganalysis process more difficult. The attacker not only needs to guess which pixels (out of all pixels in the image) have been selected to carry the secret, but also must arrange them in the correct order. The results show that the proposed algorithm achieves a significant security improvement in comparison to standard LSB when it comes to defending against brute-force attacks with a subordinate effect of image quality.

Keywords: Steganography, Data Embedding, Random Selection, LSB, ECC

Introduction

In steganography, the main objective is to embed secret data into digital cover media in such a way that it is not detectable by unauthorized parties (Pfitzmann, 1996). Therefore, the ultimate requirement that should be considered when designing any steganography algorithm is the invisibility or undetectability (Deshmukh and Pattewar, 2014). This means that both the result stego-media and cover-media should be statistically and visually almost identical (Luo *et al*., 2010). This combination of art and science in secret writing has been well developed over the years for the purpose of solving many security issues (Maji *et al*., 2014). The concept of steganography is to take the cover media and the secret message as inputs to the steganographic algorithm that in turn produces a lookalike cover media with an embedded secret message in it, called stego-media. Because of the invisibility factors of steganography techniques, retrieving the message from the stego-media without prior knowledge of the technique that has been used can be very complicated. It is noteworthy that steganography is entirely different from cryptography in that it provides protection to the secret by embedding it into the cover media, without altering its formation (Raphael and Sundaram, 2010). Nonetheless, to strengthen the security of the secret message, cryptography techniques are used alongside steganography by applying encryption on the message before performing steganography (Khosravi and Naghsh-Nilchi, 2014).

In computing, the term image refers to a collection of numbers that determines the light intensities of a picture at different area (Johnson and Jajodia, 1998). Those numbers, which are referred to as pixels, are arranged

Science Publications

© 2017 Ammar Y. Tuama, Mohamad A. Mohamed, Abdullah Muhammed and Zurina M. Hanapi. This open access article is distributed under a Creative Commons Attribution (CC-BY) 3.0 license.

individually in grid form. For each pixel, there is a limitation to the number of bits used, called bit depth, that determines the number of colors that the pixel can represent. The 8-bit pixel is the smallest bit depth for a color image, although the most widely used is 24-bit. The difficulties of distinguishing the slight changes in the image by Human Vision Systems (HVS) give this type of carrier the flexibility to carry a secret message. Steganography techniques use the bit(s) from selected pixels to hide the message. Therefore, the higher the bit depth, the more secrets can be hidden. The availability of a broad range of image file formats and techniques for image modification and compression, have triggered the development of many steganography techniques specific to the image type.

The image-based steganography techniques can be divided into four categories: Spatial domain, transform domain, masking and filtering and distortion. Each technique has been realized via numerous algorithms. For example, the idea of the spatial domain was materialized by Least Significant Bit (LSB), Edges Based data Embedding (EBE) (Islam *et al*., 2014), Pixel Value Differencing (PVD) (Shen and Huang, 2015) and Random Pixel Embedding (RPE) (Tiwari *et al*., 2014). LSB is the most famous algorithm amongst all the spatial domain techniques because of its least effect on the quality of a cover image. However, this technique is lacking in message protection; a simple attack can easily be used to retrieve the hidden message.

In this study, we have proposed a new solution to address the security issues of the Least Significant Bit (LSB) algorithm by adding an extra layer of security to the secret message. Even better, this goes without affecting the carrier image quality. The whole idea is to replace the existing sequential pixel selection with random pixel selection that is controlled by the solution points on the elliptic curve equation. Moreover, further confusion is achieved via embedding noise bits to the unused pixels. Under this solution, the secret message remains protected even after the stego-image has been compromised because attackers still need to figure out the correct pixels that hold every bit of secret message as well as its arrangement for successful message reconstruction.

This paper is organized as follows: In section 2, we present an explanation and the main conception of the LSB embedding algorithm. In section 3, the previous work is presented with the recent solution, improvement on the LSB algorithm security and the main drawbacks of each solution. In section 4, we describe the EC equation and the way of generating the elliptic curve with the EC equation parameters. In the next few sections, the proposed algorithm is evaluated and explained in detail the brute-force attack analysis and results of performance measurements. We finally conclude our work in section 9.

LSB Algorithm

There are two techniques used when implementing LSB algorithms, which are LSB Replacement (LSBR) and LSB Matching (LSBM). The LSBR algorithm is a well-known technique, which replaces the LSB plane of the cover-image with the secret bit stream (Neeta *et al*., 2006). This technique transform a secret message into binary form and sequentially overwrites one bit at a time; the LSBR of the selected cover image pixels bytes (Cole, 2003). In the spatial domain, the LSBR is one of the most useful techniques (Khosravi *et al*., 2012). The foremost advantages of the LSBR algorithm are ease of implementation, simplicity to understand and very low effects on the cover-image with high payload (Li *et al*., 2009). Due to these factors, many algorithms have been developed based on this concept (Mohamed *et al*., 2011). However, the secret message embedding is imbalanced when using this algorithm, therefore, it is easy to detect the messages existence by the traditional detection methods (Ker, 2005). Another LSB technique is LSBM (also known as ±1 embedding), which employs a minor modification to LSB replacement. The secret bit and LSB of the cover-image pixel are matched prior embedding a secret bit. If it is identical, do nothing; otherwise, +1 or -1 is randomly added to the corresponding pixel value. This technique is used to avoid asymmetry in LSBR. However, this modification increases the complexity of the LSBR and causes a high-frequency noise and more effects on image quality. With both techniques, the availability of secret messages can be detected by a perceptual and statistical characteristics analysis of cover-image, such as an image histogram (Paul and Preneel, 2004). To overcome these issues, researchers can either minimize the effect of the algorithm on the image quality such that the message remains unnoticeable to the attackers, or maximize the difficulties to extract the message if its existence was discovered.

Our proposed solution works based on LSBR algorithm, which has the best embedding quality in a spatial domain. Therefore, we chose the existing LSBR algorithm to benchmark with the proposed solution. We try to improve the security of the LSBR algorithm without affecting the high quality of embedding process. In addition, the message will be protected against brute-force attack and statistical analysis.

Related Works

Recently, many works have been proposed to improve the security of the LSB algorithm. Jung *et al*. (2008) have combined the LSB algorithm with a Multi-Pixel Differencing (MPD) to enhance the security of the LSB embedding. In their proposed solution, the estimated value of pixel smoothness is calculated as a summation of pixels from four different pixel blocks. If

the smoothness value is small, the LSB algorithm is used for embedding the secret message, otherwise the MPD algorithm is used instead. This algorithm is simple to implement, however, it is limited to image characteristics because it depends on pixels value when choosing the embedding technique. Moreover, the smoothness value needs to be calculated for each block in both stegano and retrieving processes and that may impose some degradations in the performance.

To improve the security of the LSB algorithm, Raja and Chowdary (2005) have combined three techniques which are the LSB algorithm, RAW image compression and the Discrete Cosine Transform (DCT). At first, the secret message is embedded into the cover image using the LSB algorithm. Then, the image is transformed from the spatial domain to the frequency domain using DCT technique. The resulting image is compressed using quantization and run-length encoding algorithms. The proposed solution extends a high security to the embedded message, but this at the expense of the complexity and speed of the algorithm. Moreover, the technique is limited to raw images (original image without any compression) in performing the steganography operation.

Viswanatham and Manikonda (2010) have proposed the use of random pixel selection in the LSB algorithm. Initially, the image region is selected, then random numbers are generated to select pixels from the operated region. Furthermore, random numbers are added to the pixels as a password. Instead of the high security of the proposed solution, there is no perceptual transparency (visual effects) considered in it. As a result, the stego image can easily be identified for holding secret message and image analysis can be performed to track the changed pixel's bit.

In order to improve the robustness of the LSB steganography, Khalaf and Sulaiman (2011) have proposed a new technique based on LSB matching. In the proposed technique, the secret message is encrypted using RSA algorithm and converted into a bit stream and divided into segments. The cover image is also divided into the same number of segments. Each segment from encrypted message was compared with cover image segments to find the best matching segment to embed the encrypted data segment in it. The proposed scheme provided two layers of security which are encryption layer and the random sequence of segment embedding. Nonetheless, it has disadvantages in that when using RSA encryption because it requires a long key to achieving a high security. In addition, needing to exchange the number of segments used in hiding process between sender and receiver, helping unauthorized access to get the message size and segments order. As a result, the encryption layer is the only protection layer in the proposed scheme. Besides, segments matching will degrade the performance of the data embedding process and require specific criteria and technique for matching.

Recently, Akhtar *et al*. (2014) have proposed a new technique to improve the security of the LSB algorithm as well as to maintain low effects on image quality. The technique inverts the least significant bit of the selected pixels using a bit-inversion technique in order to minimize the embedding impact on the pixel's value, prior applying the LSB embedding algorithm. The secret message is guarded using a password before embedding into the image and transmitting to the recipient. This technique improves the quality of stego-image and as a result it delivers a better Peak Signal-to-Noise's (PSNR) value. Additionally, the proposed solution has randomized the pixel selection to improve the robustness of the LSB embedding as an alternative to sequential embedding by using the Rivest-Cipher 4 (RC4) algorithm. However, the RC4 algorithm is known for some security issues and therefore not recommended for secure applications (Paul and Preneel, 2004). It suffers from a probability issue concerning the first two output bytes which can be exploited by the attackers, which could lead to compromising the secret key. In addition, the vulnerability issue is widening if non-randomized keys are used, the beginning of the output keystream are not discarded, or the same keystream is re-used (Robles and Choi, 2009). Furthermore, recent studies show that by incorporating the RC4 algorithm into security protocols such as WEP resulted in vulnerable protocols (Sasikumar *et al*., 2010; Kadry and Smaili, 2010).

Based on LSB array, Swain and Lenka (2015) proposed a new steganographic technique for improving the security of LSB algorithm. The technique defines four LSB array: LSB0, LSB1, LSB2 and LSB3. The LSB array is selected based on the message size. For longer message, the LSB3 array can be used whereas LSB0 is used for smaller message. The secret message words mapped based on the chosen array for maximum matching. The length of each secret message word and the indication for starting embedding of it in the image are encrypted by RSA algorithm. However, the proposed solution has a limitation in the security due to a couple of points. The RSA encryption is not efficient because it requires very long key size to obtain a high security and the performance is slower than other modern encryption algorithms. Additionally, using an indication of message size, when using a specific LSB array, may help the intruder recognize the secret message.

Elliptic Curve Equation

In 1985, Neal Koblitz and Victor Miller independently introduced a new security approach called Elliptic Curve Cryptography (ECC) (Mohamed, 2014) as an alternative to other public-key systems such as RSA and DSA. Although the algorithm comes with some demonstrable advantages, it does not enter the wide

usage until 2004. The security of this cryptosystem relies on Elliptic Curve Discrete Logarithm Problem (ECDLP) and, until now, this problem cannot be solved by any sub-exponential algorithm. The simplified elliptic curve equation is given by:

$$y^2 \equiv x^3 + ax + b \pmod{p} \quad (1)$$

where, p is a prime number greater than 3 and Interestingly enough, this cryptosystem appears to address the problem of the key length in RSA. As such, an elliptic curve cryptosystem requires smaller parameters compared to the one used in RSA. Therefore, ECC is said to be more efficient than RSA.

The set of solutions (x,y) on elliptic curve operated by a point addition operation from a group structure that satisfies the requirements for a public key cryptosystem. Computing the number of points on the curve is very important and it is related to the hardness of ECDLP, therefore, there are several algorithms that have been developed such as Schoof algorithms (Schoof, 1985) or its modified version called SEA (Schoof-Elkies-Atkin) with better efficiency and security (Ku *et al*., 2014).

The number of solutions on the curve is largely influenced by the parameters a, b and p. Changing any of these values lead to entire changes in the elliptic curve solutions. The generation of the points requires us first to choose a point called base point, (x_0,y_0) which can be any point on the curve. Other points can be generated using the formula $Q = kG$ (Hankerson *et al*., 2010) where k is a scalar. Choosing different G results in a set of different sequence of points generation. However, all sets are said to be equal.

As for the security, based on certain mathematical intractability problem (Lauter, 2004), it is infeasible to determine the EC parameters (a, b and p) by knowing the EC points. The bigger the EC parameters results in more solutions, therefore, the time needed for brute-force attack gets much longer.

Proposed Algorithm

In the proposed algorithm, we exploit the capability of the elliptic curve equation in randomizing locations for point generation as a way to improve the security of existing LSB algorithm. Randomized points are mapped to cover image pixels that are responsible for hiding the secret message. By random, we mean the selection is controlled by the parameters a, b and p of EC equation. Equation 1 shows that the size of x and y of any point (x, y) cannot exceed p. Whenever necessary, variable p can be set to the required value, as an example, to the horizontal and vertical pixel counts of an image. This p determines the boundary for pixel selections on the cover image. Using this technique, EC parameters a, b, p and a base point G are considered as a key and the recipient must use this in order to retrieve the embedded secret message. As such, the communication parties are required to exchange these parameters beforehand.

Algorithm 1: Proposed Algorithm (Message Hiding : Sender Part)

Input: Bytes array of scanned input message *D*, EC Parameters (*a; b* and *p*),
Selected generator point (*G*) and *cImage*.
Tlist: array of point (x,y).
Generate EC points using EC parameters and EC equation.
Save EC points in *Tlist*
Convert input message bytes into stream of binary form *binaryF*.
Initial (bin) value with 1
Loop *x* = 1 to width of(*cImage*)
Loop *y* = 1 to hight of(*cImage*)
  IF (*x; y*) in *Tlist* **Then**
    value = *binaryF*[*bin*]
    rImage(*x; y*) = *cImage*(*x; y*) and 254
    rImage(*x; y*) = *cImage*(*x; y*) || *value*
  End If
  bin = *bin* + 1
End Loop
Output: *rImage* which is the image after embedding message in it.

Algorithm 1 is responsible for embedding the secret message into a carrier image. Initially, the sender generates the EC solution using the agreed EC parameters. These points, each represented by x and y coordinates, are used to address the pixels on carrier image. Using LSB algorithm, all bits of the secret message will be hidden into the selected pixel's least significant bit one after the other.

Figure 1 shows a set of integer solutions on an EC equation, exhibiting highly randomness characteristics of points location. The size of the figure is determined by the maximum values of x and y coordinates. A points itself is defined by its (x, y) coordinates which are responsible for carrying a bit of secret message. In this example, the square figure is made the size of the carrier image, therefore, there is a one-to-one mapping between a point and a pixel.

The sequence of points generation depends solely on the value of base point G. Consequently, using different base point leads to generating a totally different order of points. Given a base point $G(x_0, y_0)$, the rest of the points can all be generated iteratively as $Q_0 = G$,$Q_1 = 2_G$, ...,$Q_N = nG$. The generated sequence is well dispersed over the square figure.

Algorithm 2: Proposed Algorithm (Adding Random Noise Bits)

Input: Image with embedded message in it *rImage* and amount of noise (*maxNoise*)
T_{list}: array of EC points
Initialize noise counter *nCounter* = 0
While *nCounter* <= *maxNoise*
Random (x,y) where *x* <= *width*(*rImage*) and *y* <= *height*(*rImage*)
  If (*x; y*) not in *Tlist* **then**
    value = random(bit) | for noise message.
    rImage(*x; y*) = *cImage*(*x; y*) and 254
    rImage(*x; y*) = *cImage*(*x; y*) || *value*
    nCounter = *nCounter* + 1
  End If
End While
Output: *rImage* which is the image after hiding noise in it.

Figure 2 shows such behavior for two EC equations with the same *a*, *b* and *p* but different *G* point. Each point is represented by a vertex (+) and the generation sequence is shown by an edge that connects an earlier point (*i*-1)*G* to a current point *iG*. Point generation is very chaotic and the point location of (*x*, *y*) depends on the base point. We notice that the vertices remain at the same location in both images but with difference edges connections.

Using the same pixel locations over and over again can be vulnerable and by analyzing a set of images the attacker can easily get to the secret locations without needing to know the EC parameters. Due to this, we further proposed random (noise) bits to be embedded into the unused least significant bits. In Algorithm 2, which represents noise embedding process, the sender randomly generates (*x*, *y*) locations for the noise bits and defined the amount of noise that to be added. If the (*x*, *y*) value is overlapping with any EC point, it will be discarded. Otherwise, it will be used to hold a random bit value. These additional bits cause further confusion to the attackers and thus image analysis is made more difficult. The more noise is added to the image, the more disturbing is appended to the attacker's analysis. However, this will not affect the receiver's process of retrieving the secret message at the receiver side because the whole processes are entirely controlled by the EC parameters.

In brute-forcing the secret message, the attacker needs to know not only the points that hold the secret but also the correct ordering of the points and this is a delicate issue if the number of possible orders is gigantic. Mathematically, we can calculate the number of possibilities of orderly selecting points using permutation function as follows:

$$P(n,r) \equiv \frac{n!}{(n-r)!} \tag{2}$$

where, *n* is the total number of pixels in the carrier image and *r* is the length of secret message in bits such that $0 \leq r \leq n$. Using an example in Fig. 1, we have 250 integer points and by assuming that all are occupied by message bits, the number of possible non-repeating arrangements is $P(250, 250) = 250!/(250\text{-}250)! = 3.232*10^{492}$. The result shows that the number of possible arrangements for only 250 solutions is enormous and time needed to brute-force attack can be extremely involved.

Algorithm 3: Proposed Algorithm (Data Hiding: Receiver Part)

Input: Bytes array of scanned input image *cImage*, EC Parameters (*a; b* and *p*),
Selected generator point (*G*) and.
Generate EC points using EC parameters and EC equation.
Save EC points in *Tlist*
Initial binary output stream (binaryS) with 0.
  For *i* = 1 to width of(*cImage*)
    binaryS = *binaryS* + *LSB*(*rImage*(*Tlist*[*i*]))
  End For
Split binary stream (binaryS) into block of 8 bits
Convert each block to it ASCII code form and save it as output text *TextMessage*.
Output: *TextMessage* which is contain the original message from sender.

Nevertheless, the large number of possible arrangements is not the only obstacle in stealing the secret. Let say the attacker has found the number of bits to be 8 (this gives us 40320 possible arrangements) and it has been extracted from their locations as 1-1-1-1-0-0-0-0. The secret has 4 bits of '1' and '0' each. The only thing left is to arrange correctly. Some arrangements result to a set of acceptable characters and deciding on the correct one can be confusing. In this example, we will get 28 acceptable characters as shown in Table 1. Each character can be a part of the original secret message and the attacker will have difficulties to determine the correct one.

Algorithm 3 represents a list of steps executed on the receiver's part to retrieving the secret message from the stego-image. Using the secretly exchanged EC parameters, the receiver generates the list of solutions that satisfies the EC equation. As a consequence, the algorithm deals directly with the pixels that hold the secret and thus save the processing effort. The additional benefit of the new technique is that if the image has been compromised, both sender and receiver only need to agree on a new set of EC parameters without needing to choose a new image.

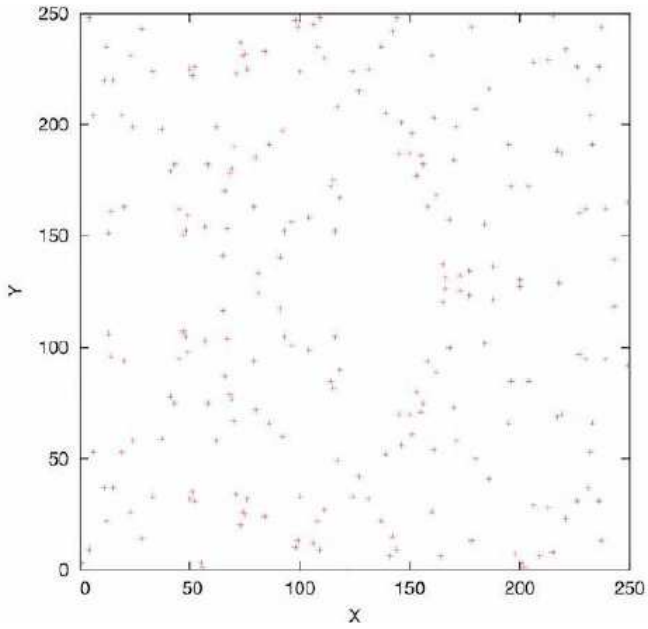

Fig. 1. Integer solution on EC equation $y^2 \equiv x^3 + 3x + 5 (mod\ 257)$

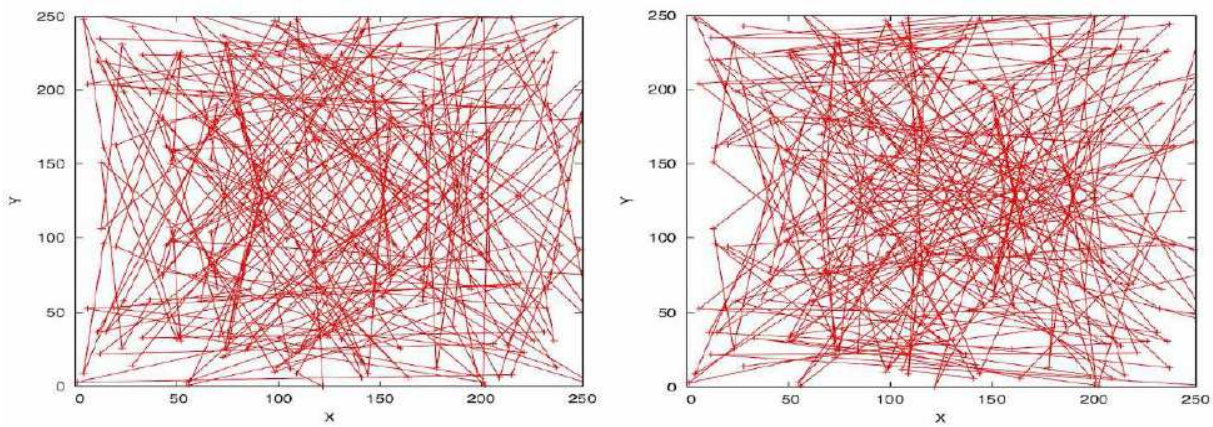

Fig. 2. Connected EC points of EC equation $y^2 \equiv x^3 + 3x + 5 (mod\ 257)$ with two different (G) point (a) G(28,14) (b) G(56,256)

Table 1. The possible acceptable characters

Character	Binary code	Character	Binary code
‘	00100111	U	01010101
+	00101011	V	01010110
.	00101110	Y	01011001
3	00110011	Z	01011010
5	00110101	\	01011100
6	00110110	c	01100011
9	00111001	e	01100101
:	00111010	f	01100110
!	00111100	i	01101001
G	01000111	j	01101010
K	01001011	l	01101100
M	01001101	q	01110001
N	01001110	r	01110010
S	01010011	x	01111000

Two elements determine the amount of secret message that can be hidden in a cover image. First, the size of the cover image delimits the number of pixels on and hence the size of the secret message. Second, the values of EC parameters a, b and p. These values determine not only the number of points, but also the boundaries for x and y coordinates of the points. Noteworthy, the values of a and b is inversely related to the number of points.

Security Analysis against Brute-Force Attack

Assume carrier image can be of the same picture but different resolution such as 28×28, 216×216 or 232×232.

In reality, the number of pixels should be larger than the number of message bits, therefore a different image resolution is selected according to meet the size of the secret message. Let's have a scenario where an image of n pixels is used to hide a secret message of s bits as well as the noise of e bits such that $s + e = m \leq n$. For simplicity, let $m = n$.

(A) Normally, the original image is available to the attackers as well as its stego-image can easily be obtained via some kind of eavesdropping on the communications link.

(B) By having both images, the attacker task is to identify which pixel bits have been used to carry the secret message. This is confusing since some (if not all) least significant bits somehow have been operated either by message bits or noise bits. The attacker doesn't really know how many bits have been the secret message, neither the noise bits. The challenge is to determine which bits are responsible for carrying the secret and if found, which arrangement yield the correct secret message. For that, the attacker needs to try every number of the bit(s) starting from 1 before (possibly) succeeding at s.

(C) Let say, the attacker decides $r \leq s$ bits were used, then he/she needs to select which r bits out of n bits that have been used to carry the secret message. There are $C(n, r)$ possible combinations. For each choice, the attacker needs to arrange the bits into correct order so that the original secret message can be retrieved. This again results in $r!$ orderings. The whole processes amount to $P(n, r)$ permutations. Moreover, since each pixel can hold the value of the secret message of either 0 or 1, the entire search space is equal to $2^{P(n, r)}$.

(D) The attacker need to repeat step (C) for different r such that $1 \leq r \leq s$ until he/she found the secret message. Assume that s can go up to n (the number of message bits equals the number of pixels), we have the total works required to brute-force this algorithm equals $\sum_1^n P(n,r)$.

Assume that we are at $r = 64$, if we have a color image of $n = 28*28$ bits and a secret message of $s = 64$ bits, the number of possible orderly selection of 26 bits out of 216 is given by $P(2^{16}, 2^6) = 65536!/(65536-64)! = 1.74151522*10^{308}$ and the respective brute-force attacks require $2^{1.74151522}*10^{308}$ guesses. A result is an enormous number of possibilities that the attacker needs to go through before getting to the secret message. From this mathematical calculation, we can expect significant improvement from the proposed technique against the brute force attacks. The analysis shows that the pixels location can not be extracted by the attacker because it depends on many variables which are a, b and p. In addition, the noise bits that have been added to the unused pixels make the attempts to predict the EC parameters much more difficult. The proposed algorithm combines the LSB algorithm of steganography with the EC randomized point generation to present a random distribution of pixels location for steganographic algorithms.

Image Quality

A Python with Java application has been used to implement both the standard LSB and the proposed algorithm, as well as image statistics calculator. In the experiment, five hundred different EC parameters with a fixed message size of 768 bits have been used for the testing. As for the comparison, we use standard LSB algorithm as our benchmark. The Peak-Signal-to-Noise-Ratio (PSNR) has been used as a parameter for evaluating the performance of the proposed algorithm from the perspective of image quality degradation. The PSNR represents the amount of changes is respect to the algorithm used during message hiding. The higher PSNR value means the lesser the effects on the image quality. The PSNR can be calculated by the following equation:

$$PSNR = \frac{MAX^2}{MSE} \tag{3}$$

where, MAX is the maximum possible number of pixels and the MSE is the Mean Squared Error. The MSE value is calculated with the following equation:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} I(i,j) - K(i,j)^2 \tag{4}$$

where, m and n are the image dimensions and I and K are the image array values. Another performance measurement that can be used to show the quality of the embedding process is the mean value. This value describes the average brightness or pixel values of the image. The higher mean value means a higher brightness. In data embedding, this value describes the amount of effects on the image quality and brightness, therefore, the image should not have significant changes in the mean value (Jain, 2012). Mean value could be calculated by the following formula:

$$Mean = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} I(i,j) \tag{5}$$

where, m and n are the image dimensions and I is the image array values. In statistics, the Standard Deviation (SD) is a measure of an amount of variation in a set of data. With images, the underlying brightness probability distribution is estimated by the SD value, additionally, it characterized the noise in the image.

Adding more noise in the image causes decreasing the SD value (Petrou and Petrou, 2010). Hence, the embedding technique should produce an image with a closer SD value to the original image SD. The SD value is calculated by the following formula:

$$\sigma = \sqrt{\frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left(x_{i,j} - \bar{x}\right)^2}{(n*m)-1}} \quad (6)$$

where, σ is the standard deviation, x value of each pixel, $\bar{x}$ mean value of the image and (n*m) is the image size.

Results and Discussion

In the experiment, image histogram is used to evaluate the quality of stego-image in respect to the cover image. We use 263*263 color image as a cover image and the EC parameters a, b and p are set to 2, 3 and 263 respectively. Using these parameters, we generated 269 integer solutions and with each point consumes a pixel of bit-depth 3 bytes, we can hide in total 789 bits of the secret message. For our testing purpose, the amount of secret message that has been used in the experiment is 789 bits. For better security, we also include 8192 bits of noise to the stego-image.

In general, the results show that the effect of the new technique on the image quality is approximately equal to that of standard LSB. However, the security level is relatively higher compared to the standard LSB which can easily be brute-forced by attackers. In this new technique, the security is hardened in that, by brute-forcing, attackers will come to a point where many choices of intelligible solutions appear and therefore deciding which one is the right one is made difficult. The idea of this technique is to create an algorithm that mimics near unconditionally secure system by adding further confusion to the stego-image via the use of noise bits into the cover image.

From Fig. 3, the first bits stream, labeled as 'Before Hiding' represents a block of LSBs taken from a sequence of pixels in an image. Of all those LSBs, the grayed bits are those selected by the EC points as locations for hiding the secret message. In this example, we have selected 8 different locations for concealing the letter 'A'.

The second bits stream labeled as 'After Hiding' shows a result after hiding the letter 'A'. Here we can also see some yellowish bits, of which have been selected for concealing the random noise, in this case, 4 bits. Suppose an attacker obtains the stream 'After Hiding' and starts brute forcing attack on LSB of pixels, the result of the attack yields a significant number of characters and if we use many input characters, that will give the attacker a countless number of words.

Figure 4 shows a cover image and its histogram with three bands (Red, Green and Blue). After hiding data with about 1200 characters (secret message and noise) into the image, we have a resulting histogram as depicted in Fig. 5. We observe that the histogram for stego-image is very similar to that of the original image.

Table 2 shows an average PSNR for the three most popular test images for steganography algorithms namely Lenna, Baboon, Cameraman, Clover, Flower and Bud. The average PSNR for the proposed algorithm is very close to and sometimes better than that of standard LSB.

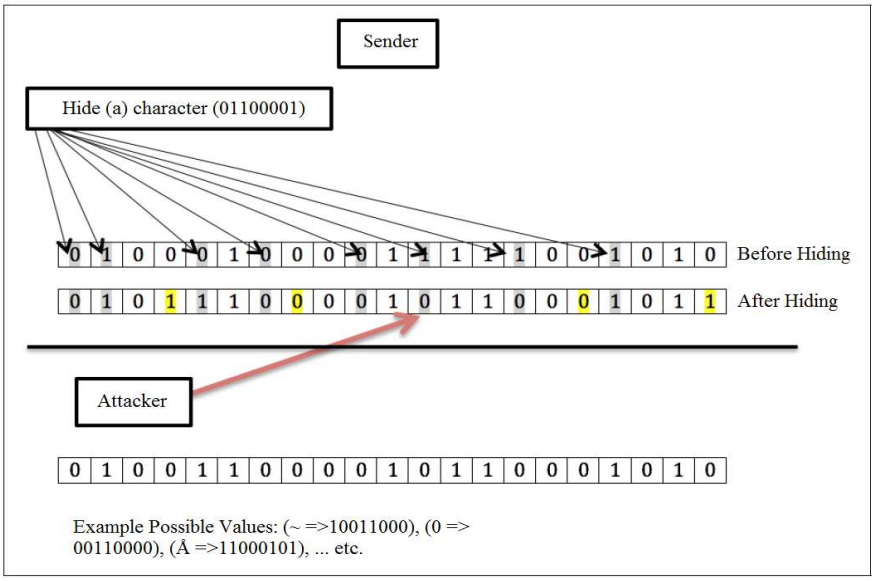

Fig. 3. Brute force attack example

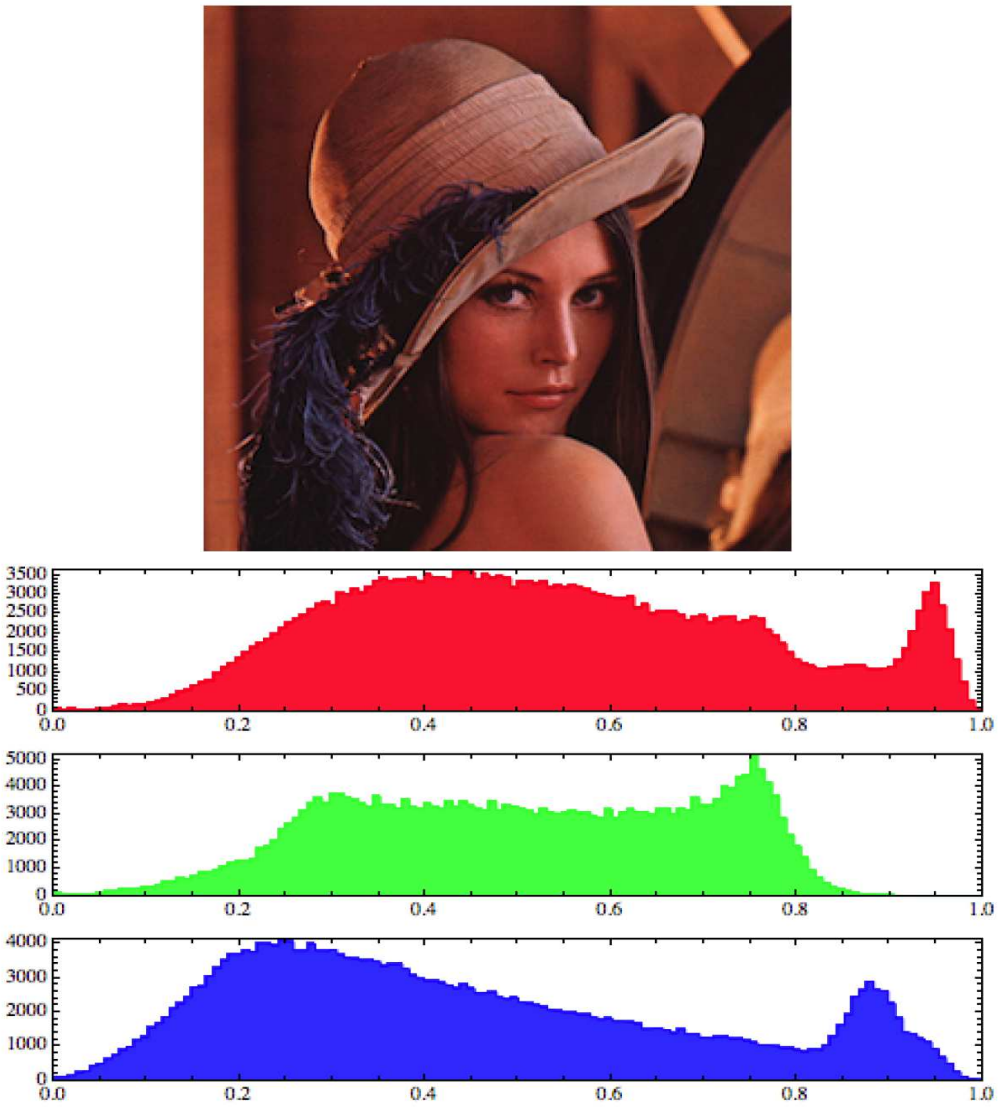

Fig. 4. Cover image and its histogram

Table 2. PSNR test results

Image name	LSBR	Proposed algorithm
Lenna	66.51 dB	66.78 dB
Baboon	66.45 dB	66.19 dB
Camera-man	66.64 dB	66.61 dB
Clover	65.84 dB	65.45 dB
Flower	64.44 dB	64.57 dB
Bud	65.75 dB	65.72 dB

Table 3. Mean test results

Image name	Original image	LSBR	Proposed algorithm
Lenna	180.21	180.205	180.216
Baboon	131.70	131.683	131.694
Camera-man	118.72	118.695	118.734
Clover	173.97	173.872	173.862
Flower	138.85	138.832	138.825
Bud	144.99	144.951	144.967

Table 4. SD test results

Image name	Original image	LSBR	Proposed algorithm
Lenna	49.110	49.160	49.106
Baboon	60.380	60.355	60.364
Camera-man	62.300	62.291	62.294
Clover	71.814	71.820	71.825
Flower	75.364	75.358	75.352
Bud	82.925	82.921	82.922

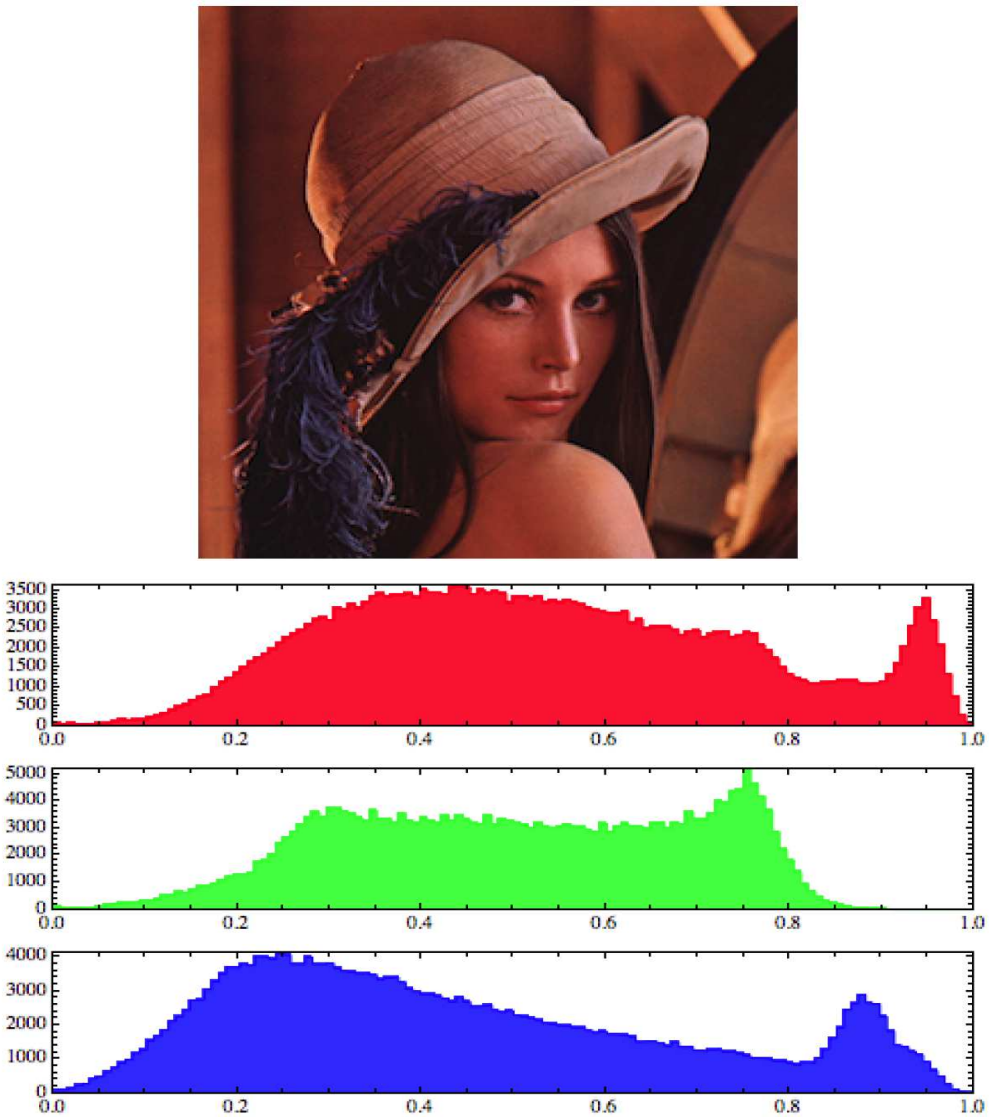

Fig. 5. Stego image and its histogram

Table 3 and 4 show the mean and SD values of the images before and after implementing the embedding process using standard LSB (LSBR) and the proposed algorithm. There is no significant difference in the SD and mean values between the original image and stego-image after applying the proposed solution. On the other side, the proposed solution produces an image with same quality that we can get with standard LSB (LSBR). From all of that, we infer that the proposed solution has minor effects on image quality which cannot be noticed when applying statistical analysis. This further the challenges of steganalysis and intruder when aiming to expose the presence of the hidden message. Based on these experiments, we should agree on the ability of this algorithm in minimizing the distortion effect on the cover image.

Conclusion

In this study, we have proposed a new steganography technique based on the infamous LSB algorithm. This variant is designed to withstand brute-force attacks. The idea is to embed the secret message into pixels that are randomly selected. Selection is based on points on EC whereas randomization is controlled by parameters (key) of the EC equation. Moreover, further confusion is achieved via the use of noise bits embedded into some unused pixels. The new technique is able to prevent the attacker from predicting the pixels that were involved in the message embedding process. Furthermore, if the image, the EC parameters or the base point (G) are compromised, the user can simply change the EC parameters without needing to change the cover image.

Acknowledgement

The authors are grateful to the anonymous referees who have contributed to improving the quality of this paper. This research was supported by the Malaysian Ministry of Higher Education [Grant No: FRGS/1/2015/ICT03/UNISZA/02/1(RR141)].

Author's Contributions

The authors contributed equally to the writing of this paper and they read and approved the final manuscript.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues are involved.

References

Akhtar, N., S. Khan and P. Johri, 2014. An improved inverted LSB image steganography. Proceedings of the International Conference on Issues and Challenges in Intelligent Computing Techniques, Feb. 7-8, IEEE Xplore Press, pp: 749-755. DOI: 10.1109/ICICICT.2014.6781374

Cole, E., 2003. Hiding in Plain Sight: Steganography and the Art of Covert Communication. 1st Edn., John Wiley and Sons, Inc., New York, ISBN-10: 0471444499, pp: 335.

Deshmukh, B. and T. Pattewar, 2014. A novel approach for edge adaptive steganography on LSB insertion technique. Proceedings of the International Conference on Information Communication and Embedded Systems, Feb. 27-28, IEEE Xplore Press, pp: 1-5. DOI: 10.1109/ICICES.2014.7033807

Hankerson, D., A.J. Menezes and S. Vanstone, 2010. Guide to Elliptic Curve Cryptography. 1st Edn., Springer, ISBN-10: 1441929290, pp: 312.

Islam, S., M.R. Modi and P. Gupta, 2014. Edge-based image steganography. EURASIP J. Inform. Security, 2014: 1-14. DOI: 10.1186/1687-417X-2014-8

Jain, R., 2012. High capacity data hiding using LSB steganography and encryption. Int. J. Eng. Sci. Technol., 4: 57-68.

Johnson, N.F. and S. Jajodia, 1998. Exploring steganography: Seeing the unseen. Computer, 31: 26-34. DOI: 10.1109/MC.1998.4655281

Jung, K.H., K.J. Ha and K.Y. Yoo, 2008. Image data hiding method based on multi-pixel differencing and LSB substitution methods. Proceedings of the International Conference on Convergence and Hybrid Information Technology, Aug. 28-30, IEEE Xplore Press, pp: 355-358. DOI: 10.1109/ICHIT.2008.279

Kadry, S. and M. Smaili, 2010. An improvement of RC_4 cipher using vigenère cipher. Int. J. Computat. Intell. Informat. Security, 1: 83-92.

Ker, A., 2005. Improved detection of LSB steganography in grayscale images. Proceedings of the 6th international conference on Information Hiding, May 23-25, Springer Berlin Heidelberg, Toronto, Canada, pp: 97-115. DOI: 10.1007/978-3-540-30114-1_8

Khalaf, E.T. and N. Sulaiman, 2011. A robust data hiding technique based on LSB matching. World Acad. Sci. Eng. Technol., 58: 117-121.

Khosravi, M., S. Soleymanpour-Moghaddam and M. Mahyabadi, 2012. Improved pair-wise LSB matching steganography with a new evaluating system. Proceedings of the 6th International Symposium on Telecommunications, Nov. 6-8, IEEE Xplore Press, pp: 982-986. DOI: 10.1109/ISTEL.2012.6483129

Khosravi, M.J. and A.R. Naghsh-Nilchi, 2014. A novel joint secret image sharing and robust steganography method using wavelet. Multimedia Syst., 20: 215-226. DOI: 10.1007/s00530-013-0341-1

Ku, J., Z. Cai and X. Yang, 2014. Hybrid differential evolutionary algorithms for koblitz elliptic curves generating. Proceedings of the International Conference on Mechatronics Control and Electronic Engineering, (MCE' 14), pp: 714-717.

Lauter, K., 2004. The advantages of elliptic curve cryptography for wireless security. IEEE Wireless Commun., 11: 62-67. DOI: 10.1109/MWC.2004.1269719

Li, X., B. Yang, D. Cheng and T. Zeng, 2009. A generalization of LSB matching. IEEE Signal Process. Lett., 16: 69-72. DOI: 10.1109/LSP.2008.2008947

Luo, W., F. Huang and J. Huang, 2010. Edge adaptive image steganography based on LSB matching revisited. IEEE Trans. Inform. Forens. Security, 5: 201-214. DOI: 10.1109/TIFS.2010.2041812

Maji, A.K., R.K. Pal and S. Roy, 2014. A novel steganographic scheme using sudoku. Proceedings of the International Conference on Electrical Information and Communication Technology, Feb. 13-15, IEEE Xplore Press, pp: 1-6. DOI: 10.1109/EICT.2014.6777849

Mohamed, M., 2014. A survey on elliptic curve cryptography. Applied Math. Sci., 8: 7665-7691. DOI: 10.12988/ams.2014.49752

Mohamed, M., F. Al-Afari and M.A. Bamatraf, 2011. Data hiding by LSB substitution using genetic optimal key-permutation. Int. Arab J. E-Technol., 2: 11-17.

Neeta, D., K. Snehal and D. Jacobs, 2006. Implementation of LSB steganography and its evaluation for various bits. Proceedings of the 1st International Conference on Digital Information Management, Dec. 6-6, IEEE Xplore Press, pp: 173-178. DOI: 10.1109/ICDIM.2007.369349

Paul, S. and B. Preneel, 2004. A new weakness in the RC_4 keystream generator and an approach to improve the security of the cipher. Proceedings of the 11th International Workshop on Fast Software Encryption, Feb. 5-7, Springer Berlin Heidelberg, Delhi, India, pp: 245-259. DOI: 10.1007/978-3-540-25937-4_16

Petrou, M. and C. Petrou, 2010. Image Processing: The Fundamentals. 2nd Edn., John Wiley and Sons, ISBN-10: 047074586X, pp: 818.

Pfitzmann, B., 1996. Information hiding terminology-results of an informal plenary meeting and additional proposals. Proceedings of the 1st International Workshop on Information Hiding, May 30-Jun. 1, Springer-Verlag, U.K., pp: 347-350. DOI: 10.1007/3-540-61996-8_52

Raja, K. and C. Chowdary, 2005. A secure image steganography using LSB, DCT and compression techniques on raw images. 3rd International Conference on Intelligent Sensing and Information Processing, Dec. 14-17, IEEE Xplore Press, pp: 170-176. DOI: 10.1109/ICISIP.2005.1619431

Raphael, A.J. and V. Sundaram, 2011. Cryptography and steganography: A survey. Int. J. Comput. Technol. Applic., 2: 626-630.

Robles, R. and M.K. Choi, 2009. Symmetric-key encryption for wireless internet SCADA. Proceedings of the International Conference on Security Technology, Dec. 10-12, Springer Berlin Heidelberg, Jeju Island, Korea, pp: 289-297. DOI: 10.1007/978-3-642-10847-1_36

Sasikumar, P., C. Vivek and P. Jayakrishnan, 2010. Key-management systems in vehicular ad-hoc networks. Int. J. Comput. Applic., 10: 23-28.

Schoof, R., 1985. Elliptic curves over finite fields and the computation of square roots mod p. Math. Comput., 44: 483-494.

Shen, S.Y. and L.H. Huang, 2015. A data hiding scheme using pixel value differencing and improving exploiting modification directions. Comput. Security, 48: 131-141. DOI: 10.1016/j.cose.2014.07.008

Swain, G. and S.K. Lenka, 2015. A novel steganography technique by mapping words with LSB array. Int. J. Signal Imag. Syst. Eng., 8: 115-122.

Tiwari, A., S.R. Yadav and N. Mittal, 2014. A review on different image steganography techniques. Int. J. Eng. Innovative Technol., 3: 121-124.

Viswanatham, V.M. and J. Manikonda, 2010. A novel technique for embedding data in spatial domain. Int. J. Comput. Sci. Eng., 2: 233-236.