Secret Key Steganography: improve security level of LSB algorithm

Ahmed Imad Hammoodi Al-Jarah
Postgraduate in Science and Computing Engineering
National Autonomous University of Mexico (UNAM)
Mexico City, Mexico
ahmedal_jarrah@yahoo.com

Jorge Luis Ortega Arjona
Mathematics Departament, Science Faculty
National Autonomous University of Mexico (UNAM)
Mexico City, Mexico
jloa@ciencias.unam.mx

***Abstract*—Internet has revolutionized the way people share information, store data, communicate, learn, and do business. Internet has been used for information transmission; this has changed in the last decade. Today, governments, businesses and individuals are increasingly relying on-line applications; the need to secure data exchange over the Internet and store it on the cloud has gained more importance. Data security aims to keep information safe by using cryptography and/or steganography. Steganography is a suitable candidate for keeping information safe, hiding the existence of data itself inside another medium. The Least Significant Bit (LSB) algorithm is one of the most common techniques to hide secret messages in an image. The main problem with LSB is knowing whether such an image has a secret message inside, making it easy to retrieve it by collecting the least significant bit from the steg-image. This paper presents an attempt to improve the security level of the LSB algorithm by using a secret steganography key. The proposed algorithm $_{SK}$LSB is encoding the secret key from the cover image, encodes the secret message using the secret key, embedding it to the cover image. Even if attackers know about the steganography, they cannot know about the secret key. The $_{SK}$LSB algorithm helps to get a higher and better level of security, useful for keeping sensitive, relevant, and important information stored. The results regarding this paper are compared with LSB algorithm regarding its Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) values aiming to show this method is more secure and does not affect the image imperceptibility.**

Keywords—Image steganography, secret key, Improve LSB, Embedding data, secret data transmission.

I. INTRODUCTION

Data security has gained more importance in recent years, as a result of the increasing reliance on computers and networks in daily activities. The main focus has been to keep the data and information safe from the preying eyes of attackers through the utilization of data security techniques such as cryptography and steganography. Cryptography is a group of techniques used to convert information into a non-readable form through the use of an encryption key and an encryption algorithm [1][2]. Steganography, on the other hand, hides the message inside another medium (e.g., an image, audio file,…etc.) [3]. Cryptography is used to protect the content of messages; one of its main liabilities is that the encrypted messages can draw attention of possible attackers, by knowing that "data is sensitive", and could be compromised to get the original message. Another liability has to do with the secret key transmission: while exchanging the key between sender and receiver, the risk of having it intercepted means to have access to the information since it can be decrypted. Steganography is about concealing the data inside another medium [4]. A message hidden in a selfie or a personal picture, for example, would not attract attention, even though the same secret message can be embedded into these ones. Steganography is a type of security through obscurity.The components involved in steganography are listed below [5][6]:

1. Payload (secret message): Information that is to be concealed and transmitted.
2. Steganography medium (carrier): The medium in which the information is hidden.
3. Steg-Key: An optional component that involves additional secret data needed for both embedding and extracting processes. It must be known by the sender and the recipient.
4. Steg-object: the final result after the steganography process be done.

For example, if you want to send the secret message to someone: "This backpack is so big it can break my back for carrying it," and want to get it past my editor, the message would be the payload. If you send this message embedded in your selfie, the picture would be the carrier (medium) file.

Image steganography is one of the most famous and widely used techniques, because images have a higher capacity for information redundancy, that allows the possibility of embedding large amount of data safely. Hiding information in an image is both simple and effective. The most important method of image steganography is Least Significant Bit (LSB). This is based on substitution of the least significant bit of the concealment, which is commonly used and widespread because it is very easy to be applied. The main advantage of this technique is its high capacity for concealment (known as transparency). It can hide the information with little effect on the cover image [7]. One disadvantage is that if the attacker knows about the steganography, it is easy to retrieve the secret message. The proposed algorithm $_{SK}$LSB solves this issue by using a secret steganography key. Using this secret key, even if an attacker could retrieve the LSB bits, they would not be able to get the secret message because the secret key is unknown. The proposed algorithm achieves a higher and better security level even compared with the normal LSB algorithm.

II. Steganography types

Figure 1 shows the three basic types of steganography. This classification depends on the principles of encryption [3].

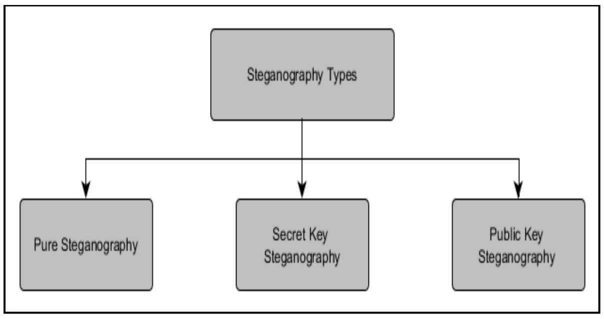


Figure 1. Steganography types

A. *Pure Steganography*

Pure steganography does not have a steganography key. The pure steganography system is one of the most practical concealment systems. This technique uses a group of media that are both meaningful and popular; it makes communicating parties able to exchange data without raising doubts. Both parties have the concealment and retrieval functions. The embedding process can be described as the following [3]:

$$E : C \times M \to C$$

where C: cover, M: message

The retrieving process of the message from the cover is described as the following [3]:

$$D : C \to M$$

B. *Secret Key Steganography*

Secret key steganography is similar to a symmetric encryption, where the sender chooses a medium and ensures the secret message in it using a secret key. It is supposed that the key used in the process of embedding is known to the receiver, so the receiver can reverse the process and retrieve the message. In absence of the key, an intercepting party cannot retrieve the embedded message. The concealment process is described as follows [3]:

$$E_K : C \times M \times K \to C$$

The retrieving process of the message from the cover is presented as follows [3]:

$$D_K : C \times K \to M$$

C. *Public Key Steganography*

Public key steganography is similar to an asymmetric encryption, which does not need to exchange the secret key. It uses two keys: a private key and a public key. Public key is stored in a public database and it is used in the embedding process to include the secret message into the cover, while the private key is used in the process of retrieving the secret message; this means that the key used in the process of embedding is not the same key used in the process of retrieval [3].

III. Related work

Applying the LSB algorithm in images means that the least significant bits for some or all the data in the image are replaced with one bit of the secret message. The color image is composed of a set of pixels. Each pixel represents the three color values: red, green, and blue. In LSB techniques, each pixel can hide three bits of the secret message [8]. For example, if the image dimension is 800*600 pixels, the image file can hide 800*600*3 = 1,440,000 bits or 180,000 bytes of secret message. The following example explains the embedding process:

	Red	**Green**	**Blue**
Pixel 1	0100111	11101001	11001000
Pixel 2	0100111	11001000	11101001
Pixel 3	1001000	00100111	11101001

To embed the number 198, (represented in binary as 11000110) into the previous pixel set, the result would be:

	Red	**Green**	**Blue**
Pixel 1	010011**1**	1110100**1**	1100100**0**
Pixel 2	010011**0**	1100100**0**	1110100**1**
Pixel 3	100100**1**	0010011**0**	11101001

Ressi Dwitias's work [9] shows the possibility of using one or two LSBs in a greyscale image, and the difference between them. Dwitia's work offers a discussion of insertion comparisons if carried out with 1-bit and 2-bit. Both images' results are tested for practicability.

Secret key is tested by Pallavi Kanojia [10] image steganography. The work makes use of the color image RGB to hide the secret message. The image is divided into red, green and blue matrixes. Here, an XOR operation is performed between one bit from the secret key and a bit from the red matrix; depending on the result, the secret message embeds it into the green or blue matrix. For example, if red matrix LSB of the first pixel is 0 and the secret key value is 1, the XOR result is 1. This is stored then in the green matrix. If the result is 0, the secret message is stored in the blue matrix. Such a distribution of the secret key information protects the message from non-authorized users.

Jagan Raj Jayapandiyan [11] proposes an enhanced Least Significant Bit (eLSB) improving the cover image's quality while comparing it to a LSB algorithm used normally in steganography. Raj's work focuses on spatial domain and an encoding division into two phases. In the first phase, metadata is created and embeds the header information in the first few bytes of the cover image; the second phase relies on processing the secret message and storing it into the cover image through analyzing the character sequence of the secret message. The main idea is occupying less space given to the secret message in the cover image; this makes steg-image quality better than with the existing LSB algorithms.

Ghazanfar Farooq Siddiqui proposes an improved image steganography method [12], with its main characteristic to be data hiding capacity of steg-images, as well as making it more imperceptible. This comes from the need of protecting some records and extremely confidential information, such as medical data. The Image Region Decomposition method (IRD) inserts in a patient's medical images secret information with a higher level of imperceptibility. The algorithm splits the grayscale magnetic resonance images (MRI) into three segments: high (H), medium (M), and low (L) intensity. Each one is made from k-number of pixels, and each one of them operates the block of n-least significant bits (LSBs), where $1 \leq n \leq 3$. The sill value t1 and t2 divides the image into the three regions or segments explained above. These segments may vary of size from image to image. The pixel bits up to three least significant bits are exploited in the low intensity region (L). In this segment, secret patient data is integrated into the third LSB with adjustments of first and second LSB. In the second segment (medium intensity region) (M), the second LSB is modified with the first LSB adjustments. In the high intensity region (H), the third segment, only the first LSB is used for data embedding. The three gray levels are used together for the incorporation of secret message. A secret key is calculated to select the pixel index value in a random order before embedding and extracting procedures. A real number range (2 to 9) is used to calculate the value of the secret key. An attempt to intercept the steg-image media LSBs would not be able to destroy or obtain the secret message and could also decrease the visual quality, making it perceptible to the human eye.

In Seyedeh Haleh Seyed Dizaji's work [13], a novel method is presented to hide an image into a cover image. This algorithm is based on the LSB substitution using Graphic Processor Unit (GPU). It hides an image into the cover image, the size of the secret image is quadrant of the cover image. This serves as a cipher key that distorts and encrypts the secret image using XOR operand. The pixels of the secret image are 8 bits (four bit-pair) each one. Each of this bit-pairs is distorted using XOR operand with two LSB bits of the host image. The results are placed in the location of two LSB bits of the cover image. That makes the first part of the cover image to hide two LSB bits of the secret image pixels, while the second part of the cover image hides third and fourth bits. This is done in a similar way for other bits of the secret image pixels.

IV. Proposed Algorithm

The proposed algorithm (SKLSB) makes use of a secret key steganography to improve the security level. It generates the secret key from the pixel color of the cover image, without needing to send the secret key to the receiver, hance, preventing the risk of hacking the secret key while transmitting over the network; the algorithm depends on generating the secret key for both entities, sender and receiver. The SKLSB algorithm uses color image of 24 bits pixel size; each pixel represents the three-color values: red, green, and blue. The steps are shown below:

A. Embedding Algorithm:

```
1) Read the cover-image (C) and the secret message (M)
2) Convert the secret message (M) to binary and store it
   in binary message (Bm)
       Bm = binary (M)
3) Check the image embedding capacity
   Capacity =( image height X image width X 3) // 8
   If capacity < secret message size (Bm)
       Then go to step 11
4) Take color bytes (R,G,B) from the pixel of the cover-
   image, convert them to binary form and save them in
   variables
       R = binary (Red Byte[pixel(C)])
       G = binary (Green Byte[pixel(C)])
       B = binary (Blue Byte[pixel(C)])
5) Generate the secret key bit for every color byte using
   the following formula:
       Sk = ( $\sum_{i=1}^{7} color[i]$ ) mode 2
       Color = R, G or B and i = bit index
6) Make XOR operation between the secret key (Sk) and
   a bit of the secret message (Mb). Store it in Message
   after decode (Mk)
       Mk = Sk XOR Mb
7) Insert the Mk in the LSB of the color byte Cb[7]
       Cb[0] = Mk
8) Repeat steps 4-6 three times for the three color bytes.
9) Repeat the steps 3-7 until the secret message finishes.
10) Return steg-image (S)
11) End
```

B. Extracting Algorithm:

```
1) Read the Steg-image (S)
2) Take color bytes (R,G,B) from the pixel of the cover-
   image, convert them to binary form and save them in
   variables
     R = binary (Red Byte[pixel(C)])
     G = binary (Green Byte[pixel(C)])
     B = binary (Blue Byte[pixel(C)])
```

3) *Generate the secret key bit for every color byte using the following formula:*

$$Sk = \left(\sum_{i=1}^{7} \mathrm{color}[i] \right) \ mode\ 2$$

Color = R, G or B and i = bit index

4) *Retrieve the LSB from every color byte to have encrypted message*

Mk = color[0]

5) *Make XOR operation between the secret key and a bit of the encrypted message, both from the same color*

Mb = Sk XOR Mk

M += Mb

6) *Repeat steps 3-5 three times for the three color bytes.*
7) *Repeat the steps 2-6 until all the secret message be retrieved.*
8) *Return the secret message (M)*
9) *End*

Example:

Hide the secret message (101) bits inside the pixel[115,218,147].

Embedding process

1. *Encode the message using the secret key*

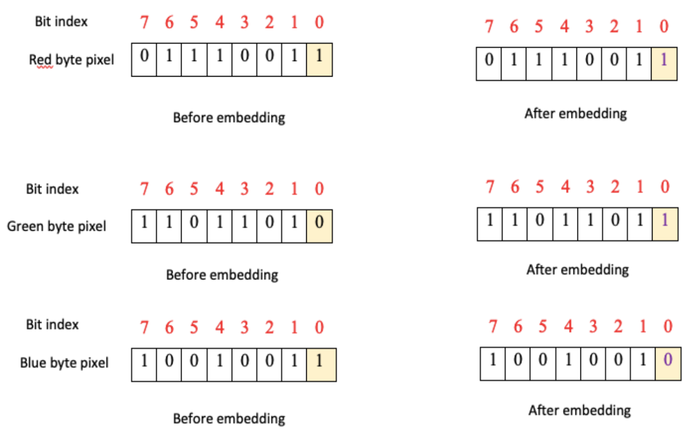
Input:
Message = (101)
pixel = [[0 1 1 1 0 0 1 1]
[1 1 0 1 1 0 1 0]
[1 0 0 1 0 0 1 1]]

Solution:
Bm = (101)
R = [0 1 1 1 0 0 1 1]
G = [1 1 0 1 1 0 1 0]
B = [1 0 0 1 0 0 1 1]

Generate the secret key
Sk_r = (0+1+1+1+0+0+1) mod 2 = 0
Sk_g = (1+1+0+1+1+0+1) mod 2 = 1
Sk_b = (1+0+0+1+0+0+1) mod 2 = 1

Mk = Sk XOR Mb
Mk[0] = 0 XOR 1 = 1
Mk[1] = 1 XOR 0 = 1
Mk[2] = 1 XOR 1 = 0
Message after XOR operation = (110)

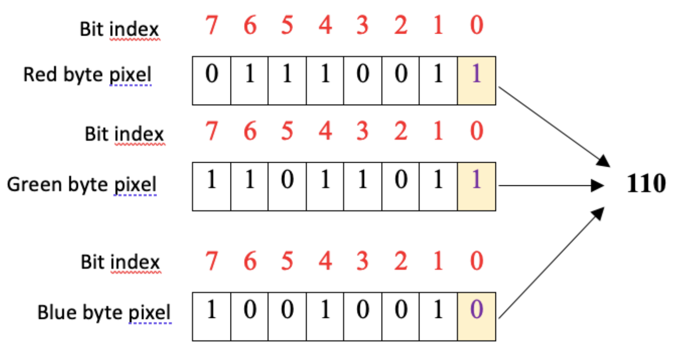
2. *Insert the message to the cover-image*



Extracting process

1. *Retrieve the message bits from steg-image*

pixel = [[0 1 1 1 0 0 1 1]
[1 1 0 1 1 0 1 1]
[1 0 0 1 0 0 1 0]]



2. *Decode the message using the secret key*

Solution:
R = [0 1 1 1 0 0 1 1]
G = [1 1 0 1 1 0 1 1]
B = [1 0 0 1 0 0 1 0]

Generate the secret key
Sk_r = (0+1+1+1+0+0+1) mod 2 = 0
Sk_g = (1+1+0+1+1+0+1) mod 2 = 1
Sk_b = (1+0+0+1+0+0+1) mod 2 = 1
Mk = Sk XOR Mb
Mk[0] = 0 XOR 1 = 1
Mk[1] = 1 XOR 1 = 0
Mk[2] = 1 XOR 0 = 1
Secret Message = (101)

V. EXPERIMENTAL RESULTS AND COMPARISONS

In this section, comparisons and experimental results for the evaluation of the SKLSB algorithm are presented. The algorithm has been implemented in Python 3.8. The hardware platform used for this experiment is a desktop computer with a 1.8 GHz Dual-Core Intel i5-7 CPU, 8 GB RAM and MacOS Catalina 64-bit operating system.

A series of standard 512x512 color cover images (24-bit images) are used to evaluate image quality between the original/cover image and steg-image. MSE (Mean Square Error) and PSNR (Peak Signal Noise Ratio) are two evaluation methods for measuring the image quality between both images. Figure 2 shows six images used in the experiment, before and after the steganography process.

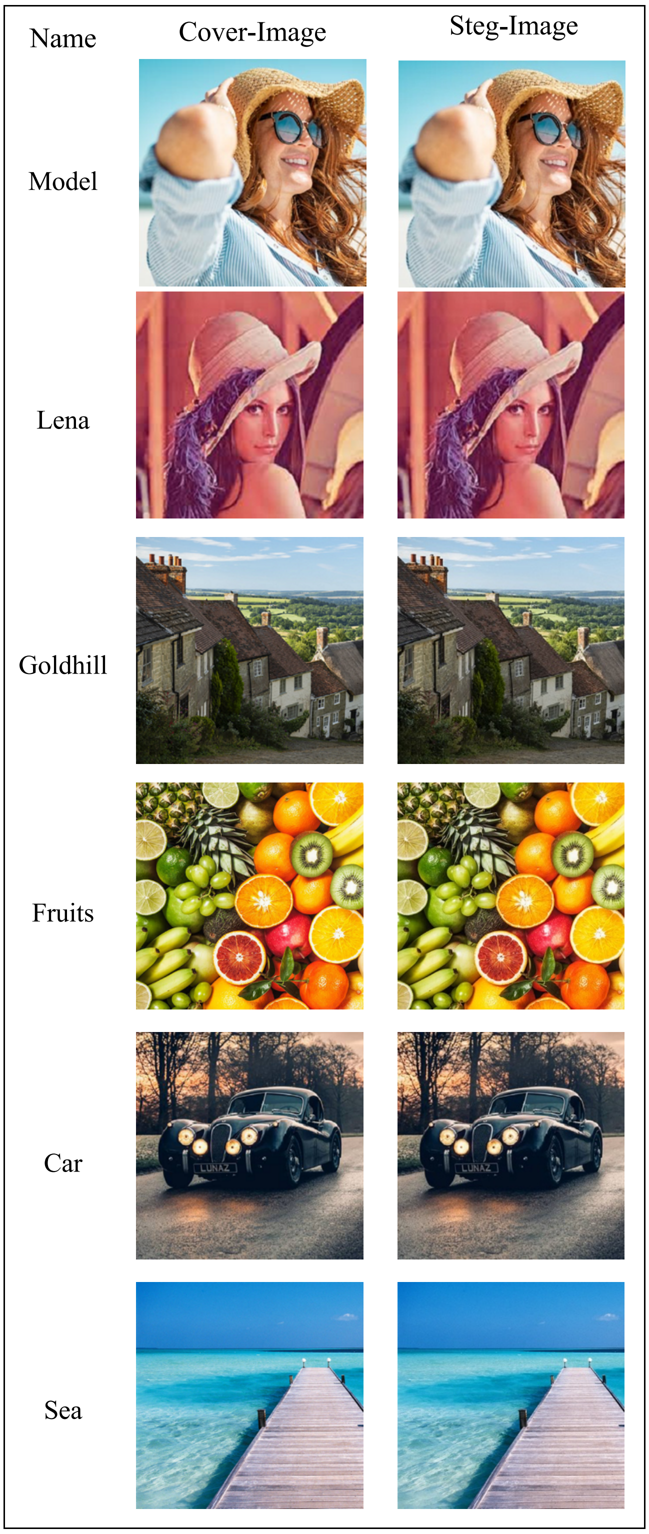


Figure 2. Cover images and steg-images 512 x512

Observing the steg-images in figure 2, we can assume that human eye is not able of distinguish between a normal (cover) image and a steg-image already manipulated to hide information. Nevertheless, it is necessary to check the image quality. One of the evaluation metrics used to measure steg-image quality is PSNR value. Higher values inside this metric mean that the steg-image quality is better. Generally, a value higher than 30 dB is considered visually indistinguishable [14]. To calculate PSNR value, the MSE must be previously calculated. Formulas showing the calculation of MSE and PSNR values are given in (1) and (2).

$$\text{MSE} = \frac{1}{MxN} \sum_{i=0}^{M} \sum_{j=0}^{N} \left(Cover(i,j) - steg(i,j)\right)^2 \ldots 1$$

$$\text{PSNR} = 10 \text{ x } \log_{10} \frac{255^2}{MSE} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2$$

It is important to note that M and N values represent the size of the images. After obtaining MSE from both images, PSNR value must be calculated to make a comparison.

Table I shows the difference between the LSB algorithm and the proposed algorithm (SKLSB) depending on the MSE (Mean Square Error) and PSNR (Peak Signal Noise Ratio) when using the LSB algorithm and the proposed SKLSB algorithm for different cover images and message sizes.

TABLE I. MSE AND PSNR COMPARISON BETWEEN LSB ALGORITHM AND THE PROPOSED SKLSB ALGORITHM (DIFFERENT COVER IMAGES AND MESSAGE SIZES)

Image	**Message size in bytes**	**LSB**		**SKLSB**	
		MSE	**PSNR**	**MES**	**PSNR**
Model	20	0.0001	87.7424	0.0001	87.7424
Lena	40	0.00019	85.2405	0.0001	85.4142
Goldhill	80	0.0004	82.0359	0.0004	81.8629
Fruits	160	0.0008	79.0528	0.0007	79.2914
Car	320	0.0016	76.0493	0.0017	75.7520
Sea	1024	0.0051	71.0289	0.0052	70.9331

Analyzing the results in Table I, it is clear that LSB algorithm and the proposed SKLSB algorithm almost have the same MSE and PSNR results when using different cover images and message sizes. Another experiment has been performed for comparison using the same image and changing the secret message size. Table II shows the results.

TABLE II. MSE AND PSNR COMPARISON BETWEEN LSB ALGORITHM AND THE PROPOSED SKLSB ALGORITHM (SAME COVER IMAGE AND DIFFERENT MESSAGE SIZES)

Image	**Message size in bytes**	**LSB**		**SKLSB**	
		MSE	**PSNR**	**MES**	**PSNR**
Model	20	0.0001	87.7424	0.0001	87.7424
Model	40	0.0002	84.9125	0.0002	84.6322
Model	80	0.0004	82.0905	0.0004	81.9819
Model	160	0.0007	79.1009	0.0008	79.0802
Model	320	0.0015	76.1672	0.0016	76.0768
Model	1024	0.0051	71.0000	0.0051	71.0375
Model	12288 Máximum capacity	0.0624	60.1770	0.0625	60.1666

The results in Table II show that LSB algorithm and the proposed SKLSB algorithm have almost the same MSE and PSNR results when using the same cover image and different message sizes. A third and last experiment has been carried out using the different images and the same secret message size; Table III shows the result.

TABLE III. MSE AND PSNR COMPARISON BETWEEN LSB ALGORITHM AND THE PROPOSED SKLSB ALGORITHM (DIFFERENT COVER IMAGES AND SAME MESSAGE SIZE)

Image	Message size in bytes	LSB		SKLSB	
		MSE	PSNR	MES	PSNR
Model	1024	0.0051	71.0000	0.0051	71.0375
Lena	1024	0.0052	70.9638	0.0052	71.0203
Goldhill	1024	0.0052	70.9258	0.0052	70.9616
Fruits	1024	0.0051	71.0117	0.0051	71.0075
Car	1024	0.0052	70.9310	0.0052	70.9184
Sea	1024	0.0051	71.0289	0.0052	70.9331

This last experiment shows almost the same result as the previous experiments.

VI. CONCLUSION

From the experimental results of Section V, it is evident that the proposed SKLSB algorithm achieves a higher and better level of security using the secret key, while providing almost the same MSE and PSNR values. The quality of the cover image presents almost any change compared to the traditional LSB algorithm in steganography secret message embedding process. The proposed SKLSB algorithm uses the image pixel to generate the secret key, solving the problem of transmitting the key; this is due to this algorithm doesn't exchange the secret key between the sender and receiver. Also, adding the secret key does not affect the visual imperceptibility, embedding-capacity and undetectability. After applying this SKLSB algorithm into sensitive data, an improvement in the security level is clear. Therefore, the algorithm makes it suitable for its usage in the management of important information. An advantage of this algorithm is that it can be used and stored in the cloud, making the information more secure and efficient to be delivered. Future work in the proposed algorithm could be parallel for improving its processing time.

REFERENCES

[1] Vinay Kumar Pant, Joyti Parlash and Amit Asthana, "Three Step Data Security Model for Cloud Computing based on RSA and Steganography Techniques", International conference on green computing and Internet of things 2015 IEEE.

[2] Craig P. Bauer, "Secret History: The Story of Cryptology", Second Edition, , published by Chapman and Hall/CRC, 2021,ISBN: 1138061239, p. xix, 4-30.

[3] Stefan Katzenbeisser, "Information Hiding Techniques for Steganography and Digital Watermarking", published by Artech House Print on Demand in the United States of America, December 1999, p. 2-4,16-25.

[4] R Praveen Kumar and V Hemanth and MShareef "Securing Information Using Sterganoraphy" 2013 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2013]

[5] Shilpa Pund-Dange, "Steganography: A Survey", Part of the Advances in Intelligent Systems and Computing book series (AISC, volume 729) ,published by Springer Nature Singapore Pte Ltd. 2018,p.327-333.

[6] Sheelu , Babita Ahuja, "An Overview of Steganography", IOSR Journal of Computer Engineering(IOSR-JCE), e-ISSN: 2278-0661, p-ISSN: 2278-8727 Volume 11, Issue 1 (May. - Jun. 2013), PP 15-19.

[7] Kazunori Uruma, Katsumi Konishi,Tomohiro Takahashi and Toshihiro Furukawa, " A Novel Approach to Image Steganography Based on the Image Colorization" published by IEEE Sydney, NSW, Australia, 2019.

[8] K. Sathish Shet, A. R. Aswath, M. C. Hanumantharaju and Xiao-Zhi Gao, " Design and development of new reconfigurable architectures for LSB/multi-bit image steganography system", Springer Science and Business Media New York 2016, DOI 10.1007/s11042-016-3736-0.

[9] Ressi Dwitias Sari and Andysah Putera Utama Siahaan, " Least Significant Bit Comparison between 1-bit and 2-bit Insertion", International journal for innovative research in multidisciplinary field, 2018, ISSN: 2455-0620.

[10] Pallavi Kanojia and Vijay Choudhary, "LSB Based Image Steganography With The Aid of Secret Key and Enhance its Capacity via Reducing Bit String Length", published by IEEE, 2019, ISBN: 978-1-7281-0167-5.

[11] Jagan Raj Jayapandiyan, Kavita C and Sakthivel K, " Enhanced Least Significant Bit Replacement Algorithm in spatial domain of Steganography using character sequence optimization ", published by IEEE Access, 2020, Electronic ISSN: 2169-3536.

[12] Ghazanfar Farooq Siddiqui, Muhammad Zafar Iqbal, Khalid Saleem, Zafar Saeed, Adeel Ahmed, Ibrahim A. Hameed and Muhammad Fahad Khan, "A Dynamic Three-Bit Image Steganography Algorithm for Medical and e-Healthcare Systems", published by IEEE Access, Volume: 8 , 02 October 2020 Electronic ISSN: 2169-3536.

[13] Seyedeh Haleh Seyed Dizaji, Mina Zolfy Lighvan, and Ali Sadeghi, "Hardware-Based Parallelism Scheme for Image Steganography Speed up " , International Conference on Innovative Computing and Communications, Advances in Intelligent Systems and Computing, Springer Nature Singapore Pte Ltd. 2021.

[14] Serdar SOLAK, "High embedding capacity data hiding technique based on EMSD and LSB substitution algorithm", DOI 10.1109/ACCESS.2020.3023197, published by IEEE Access.