Combination of Rail Fence Cipher Algorithm and Least Significant Bit Technique to Secure the Image File

Dian Rachmawati1*, Mohammad Andri Budiman1*, and Akhmad Yusuf1*

1Departemen Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Sumatera Utara, Jl. Universitas No. 9-A, Medan 20155, Indonesia

*Email: dian.rachmawati@usu.ac.id, mandrib@usu.ac.id, akhmadyusufritonga@gmail.com

Abstract. Image is a visual representation of an object. The process of sending information, especially images, can be done anywhere and with any intermediary, but not all information is public. Sometimes someone needs privacy or security from data or information that will be sent or received. One way to secure image files is to use cryptography. Cryptography is the study of how to keep data or messages safe when sent, from the sender to the recipient, without experiencing interference from third parties. In this study cryptography used is Rail Fence Cipher that was combined with steganography, so that the secret message could not be seen in plain view. Steganography serves to hide secret messages in other messages, so the secret is a very hidden existence. There are several methods in Steganography, one of which is Least Significant Bit. The concept of Least Significant Bit works by replacing the last bit of RGB image, where the message is converted first into bits. Based on experiments conducted on the Rail Fence Cipher algorithm, the time of encryption and decryption needed when the resolution of the image gets larger is linearly proportional to the value. The image does not show significant changes before and after inserting a secret message.

1. Introduction
Image is the visual form of information of an object. The delivery and distribution of images can already use a variety of media, so the security of the image becomes lower. One way to secure image files is to use cryptography and steganography.

In data transmission such as transferring an object especially image, aspect of confidentiality, and integrity are highly needed [1]. Cryptography is the science and art of keeping messages secure. Cryptography, in general, is science and art to maintain the confidentiality of news [2]. Rail Fence Cipher is one of the encryption techniques to disguise the writing by changing the position of the character with a diagonal shape down and up. To solve it, we must see the level of the writing, because this cipher is usually systematic [3].

Steganography is a technique of hiding confidential data into a medium so that hidden data is difficult to recognize by the human senses. Steganography requires two priorities, namely the container and secret data that’s hidden. Digital steganography uses digital media as a container, for example, image, sound, text, and video. Confidential data that is hidden can also be in the form of image, audio, text, and video [4].

Least Significant Bit (LSB) is a method for inserting images or images into the cover image. Bits with the smallest weight of each pixel of an image are converted into bits of the message to be hidden [5].

To protect the data, both of cryptography and steganography can be done simultaneously. The science of cryptography use randomizes or disguise the data, whereas the science of steganography to hide data in a way [6].
2. Method

In this implementation, there are four main menus; they are encryption Rail Fence Cipher, Embedding LSB, Extraction LSB, and decryption Rail Fence Cipher. Rail Fence Cipher is one of transposition algorithms. A transposition cipher involves the rearranging of the letters in the plaintext to encrypt the message [7].

Steps for encryption Rail Fence Cipher [8]:

1. Input the image to be encrypted.
2. Take the RGB value of each pixel then sort it from the initial to the end pixel. In this test, I took 1x5 pixels of the image, as in Table 1:

Table 1. The RGB value of the insert image

| R= 169 | R= 169 | R= 169 | R= 169 | R= 169 |
|--------|--------|--------|--------|--------|
| G = 41 | G = 41 | G = 41 | G = 41 | G = 41 |
| B = 41 | B = 41 | B = 41 | B = 41 | B = 41 |

3. The RGB value order of each pixel is used as a plain image value.
4. Key input. Key = 5
5. Create a table that has columns along with plain images and as many lines as keys. Then arrange the zigzag plain image in the table.

Table 2. Plain image values in a zigzag form

| 169 | 41 |
|-----|----|
| 41  | 41 |
| 41  | 169|
| 41  | 41 |
| 169 | 41 |
| 41  | 169|

6. Read Table 2 from the first row to the last to get the cipher image. Cipher image = 169 41 41 169 41 169 41 41 169 41 41 41 41 169

Steps for embedding the secret image:

1. Input cipher image. Cipher image = 169 41 41 169 41 169 41 41 169 41 41 41 169 41 41 169 41 41 41 169
2. Change all RGB cipher image values in binary form. 10101001 00101001 00101001 00101001 00101001 10101001 00101001 10101001 00101001 00101001 10101001 00101001 00101001 10101001 00101001 00101001 00101001 00101001 00101001 00101001 10101001
3. Input container image or cover image. Take the RGB value from the cover image. In this test, I use 8x5 pixels as a container image.
4. Change each RGB value in the image cover into an 8-bit binary form.
5. Insert the value of the cipher image that has been converted into binary form to every last bit of the cover image.
6. In Table 3 the insertion process to convert plain image to stego image is done by replacing the last bits in the image of the container or the cover image with the RGB binary value of the cipher image.

Steps for extraction image:

1. In the message image extraction process, the system will take the last bits of the stego image that have been inserted in the previous process; the bits can be seen in Table 4.5.
2. Every 8 bits are arranged to get RGB to be used as a cipher image.
3. Cipher image is used to restore the image in the decryption process. 

Ciphertext image =

10101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001

Table 3. Embedding Process

| 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0101001 | 10111001 | 0101001 |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0100000 | 10111001 | 0101001 |
| 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 11111001 | 0100000 | 0100000 |
| 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 |
| 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0100000 | 10111001 | 0101001 |
| 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 11111001 | 0100000 | 0100000 |
| 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 |
| 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0100000 | 10111001 | 0101001 |
| 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 11111001 | 0100000 | 0100000 |
| 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 |
| 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0000000 | 0100000 | 10111001 | 0101001 |
| 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 1011001 | 11111001 | 0100000 | 0100000 |
| 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 | 0001001 |

Steps for decryption Rail Fence Cipher:

1. Input Cipher Image (in binary):

10101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001 00101001

Ciphertext Image (in decimal):

169 41 41 41 169 41 41 169 41 41 41 169 41 41 41 169

2. Key Input = 5

3. Create an empty table that has a cipher image length column and a key length row. Fill the table with an empty string. Then input an asterisk based on the position so that it forms a zigzag pattern.

4. Input the cipher image into a table that has a "*" starting from the first line to the last line, as illustrated in Table 4. Ciphertext image = 169 41 41 41 169 41 41 169 41 41 41 169 41 41 41 169

Table 4. Input Cipher Image

| 169 | 41 | 41 |
|-----|----|----|
| 41 | 169 | 41 |
| 169 | 41 | 41 |
| 41 | 169 | 41 |
5. Read the RGB value in the table to return the plain image in the decryption process. The way of reading is based on position by following the Zig-Zag pattern. This process can be seen in Table 5.

Table 5. How to Read the result of decryption

| RGB Value | RGB Value | RGB Value | RGB Value |
|-----------|-----------|-----------|-----------|
| 169       | 41        | 41        | 169       |
| 41        | 169       | 41        | 41        |
| 169       | 41        | 41        | 169       |

6. In Table 5, the plain image value is read zigzag according to the arrow.
Plain Image = 169 41 41 169 41 41 169 41 41 169 41 41 169 41 41

3. Results and Discussions
The experiments were performed on Android 5.1 Lollipop with an Octa-core processor and 2GB RAM. The Integrated Development Environment (IDE) used for coding is Android Studio, and the programming language used is Java. The results of the experiments of each set are presented in Tables 6, 7, 8, and 9 as follows.

Table 6. Test results of the Image Encrypted with Key Variations

| No. | Key Length | Plain Image Resolution | Time(Seconds) |
|-----|------------|-------------------------|---------------|
| 1   | 5          | 50 x 50                 | 0.892         |
| 2   | 10         | 50 x 50                 | 0.767         |
| 3   | 15         | 50 x 50                 | 0.729         |
| 4   | 20         | 50 x 50                 | 0.792         |
| 5   | 25         | 50 x 50                 | 0.767         |

In table 6, it can be seen that the time of the plain image encryption process using different keys is not always stable. The longer the key was used, it will not always take a long time to process or vice versa.

Table 7. Image testing results that are Encrypted with Plain Image Resolution Variations

| No. | Key Length | Plain Image Resolution | Time(Seconds) |
|-----|------------|-------------------------|---------------|
| 1   | 10         | 30 x 30                 | 0.185         |
| 2   | 10         | 40 x 40                 | 0.357         |
| 3   | 10         | 50 x 50                 | 0.767         |
| 4   | 10         | 60 x 60                 | 1.880         |
| 5   | 10         | 70 x 70                 | 2.851         |

In table 7, the graph of the results of encryption testing with variations in plain image resolution shows the more significant the image resolution, so the encryption process longer, or vice versa, if the image resolution smaller, the encryption process time be faster.
Table 8. Test results of the Image Decrypted with Key Variations

| No. | Key Length | Plain Image Resolution | Time(Seconds) |
|-----|------------|------------------------|---------------|
| 1   | 5          | 50 x 50                | 0.808         |
| 2   | 10         | 50 x 50                | 0.761         |
| 3   | 15         | 50 x 50                | 0.720         |
| 4   | 20         | 50 x 50                | 0.860         |
| 5   | 25         | 50 x 50                | 0.744         |

In table 8, it can be seen that the cipher image decryption process using different keys is not always stable. The longer the key is used, it is not always the processing time that takes a long time or vice versa.

Table 9. The results of testing the Image Decrypted with the Variation of Image Resolution

| No. | Key Length | Plain Image Resolution | Time(Seconds) |
|-----|------------|------------------------|---------------|
| 1   | 10         | 30 x 30                | 0.122         |
| 2   | 10         | 40 x 40                | 0.305         |
| 3   | 10         | 50 x 50                | 0.761         |
| 4   | 10         | 60 x 60                | 1.527         |
| 5   | 10         | 70 x 70                | 2.613         |

In table 9, the results of decryption testing with variations in cipher image resolution shows the more significant the image resolution, so the decryption process longer or vice versa, if the image resolution smaller, the decryption process time be faster.

4. Conclusions

Based on the experiments conducted in this study, it can be concluded that the process of running time of encryption and decryption rail fence algorithm is the time needed when the resolution of the image gets bigger is relatively stable. Both of cryptography and steganography can be done simultaneously. In this research, it uses rail fence as algorithm in cryptography and least significant bit as algorithm in steganography.

5. Acknowledgments

The authors gratefully acknowledge that the present research is supported by Fund Dissemination IPTEKS Research Results for Lecturers / Researchers Universitas Sumatera Utara.

References

[1] Dian Rachmawati et al 2018 IOP Conf. Ser.: Mater. Sci. Eng. 300 012040
[2] Schneier, B. 1996. Applied Cryptography: Protocols, Algorithms, and Source Code in C. 2nd Edition. John Wiley & Sons, Inc: New Jersey
[3] Navaneethan, Ramkesh. Advanced Rail Fence Cipher Algorithm. School of Information Technology and Engineering, VIT University, Vellore. 2016
[4] Hidayat, Wildan. 2010. Perlindungan Pesan Rahasia Pada Citra Digital Menggunakan Metode Least Significant Bitsteganografi. Medan. Universitas Sumatera Utara.
[5] F Prashanti .G*, Sandhya Rani.K, Deepthi.S. 2013. LSB and MSB Based Steganography for Embedding Modified DES Encrypted Text. International Journal of Advanced Research in Computer Science and Software Engineering 3 (8): 788 – 799.
[6] D. Rachmawati and M. A. Budiman 2017 New approach toward data hiding by using affine cipher and least significant bit algorithm 2017 4th International Conference on Computer
Applications and Information Processing Technology (CAIPT) Kuta Bali 2017 pp 1-6. doi: 10.1109/CAIPT.2017.8320737

[7] Stallings, William, Cryptography and Network Security, 3rd edition, Pearson Education, 2003.
[8] Talbert, Robert, The Cycle Structure and Order of the Rail Fence Cipher, Cryptologia, Taylor & Francis, 2006