Combination of myszkowski transposition algorithm and modified least significant bit (mlsb) green channel on png image security

S M Hardi*, D Rachmawati, F Chairinnisa, I Jaya, J T Tarigan*

Faculty of Computer Science and Information Technology, University of Sumatera Utara, Jl. Universitas No. 9-A, Medan 20155, Indonesia.

Email: vani.hardi@usu.ac.id

Abstract. On sending images via the internet, such as email, social media, and other are very vulnerable to technological crimes such as tapping, or modification. Without security guarantees, others can easily get messages sent over the internet. In this research, combined cryptographic algorithms and steganography. Cryptographic methods can ensure the security of data information by converting it into random codes, however making the data can’t be read and understood by others. One of the cryptographic methods used by the author is the Myszkowski Transposition Algorithm. But the use of such encryption method does not always guarantee data security because the data is messy with other words can be regarded as unusual and suspicious. This algorithm also has the disadvantage that as complicated as any we do transposition or permutation on the characters in plaintext, we just do a randomization of the order of plaintext but we don’t change it. So it must be combined with steganography technique that is using Modified Least Significant Bit (MLSB), so the message is not visible inserted visible. Therefore, if the information is on the wrong side will cause losses. The result of this study indicate that the Myszkowski Transposition algorithm and Green Channel Modified Least Significant Bit (MLSB) on PNG image coverage, can restore the image as before. So it can be concluded that the Myszkowski Transposition algorithm and Modified Least Significant Bit (MLSB) is capable of securing the message. Therefore, it can be concluded that the bigger the image size will be the longer the process.

1. Introduction

These days, sending images by internet network, for example email, social media, etc. is vulnerable to technological crimes such as interruptions, tapping, or modification. Without security guarantees, other people can easily get messages sent by internet network. This concern makes the development of information hiding techniques to be transmitted. If the information is on the wrong side, it will cause a loss.

The cryptographic method can guarantee the security of information data by changing it into random codes so that the data cannot be read and understood by others. One of the cryptographic methods used by the author is Myszkowski Transposition Algorithm. But the use of the encryption method does not always guarantee data security because the data is disheveled in other words can be regarded as unusual and suspicious. This algorithm has advantage of having a high level of complexity with a simple key. But this algorithm also has disadvantage that no matter how complicated transposition or permutate characters in plaintext, its only randomize the order from plaintext but don't change it [9]. So it must be combined with steganography using MLSB, where the excess MLSB is not visible and is easy to implement [2].
In the insertion of the image will be done on the green channel. Because of the nature of reflection from the surface of the eye, red channels from the image sometimes experience excessive saturation (oversaturated), especially in the central region and optic nerve. While the blue channel can experience saturation that is too low (undersaturated) and there is a lot of noise. Therefore, a green channel is used to process a good image because only this channel saturation is in the right composition and the green channel is better than other gray levels [4].

To avoid these problems, steganography is born for data hiding, which refers to how much the third party is unable to detect hidden information [3]. Steganography is the science or art of hiding messages. The message is hidden in order to be unknown to others. Those who know are themselves and others they want. Steganography discusses ways to disguise and hide messages [7]. One method of hiding messages is to use Modified Least Significant Bit (MLSB) or a modification method of the LSB (Least Significant Bit) algorithm.

2. Method
In this research the author will insert a secret message that has good image quality so that it is not visible in plain view, using the combination of Myszkowski and MLSB Transposition on the PNG Image Security File on the Green Channel.

2.1 Myszkowski Transposition Algorithm
The Myszkowski Transposition Algorithm is a variation of the columnar transposition proposed by Emile Victor Theodore Myszkowski in 1902. This variant requires a keyword that has a repetitive character. In a typical columnar transposition, the appearance of the same character is treated as the next character in alphabetical order, for example the FILZA keyword will have a sequence [2 3 4 5 1]. In the myszkowski transposition, the appearance of the same character is given the same sequence number so that the FILZA keyword has a sequence [2 3 4 5 1].

2.2 Modified Least Significant Bit Green Channel
Steps for embedding the secret image:
1. Input the cover image and file cipher.
2. Previously the cipher file whose contents in decimal were first converted to binary, where binary here is a binary value from the previous decimal value.
3. Embed the value of the secret binary into G (green channel) of the cover image’s pixel. A pixel can contain just 1 bit of the secret binary array’s value.
4. Do step 3 until all the secret binary’s value have been embedded.

Steps for extraction the secret image:
1. Input the stego image and the same key during the encryption process.
2. Get MLSB’s value from the stego image’s pixel that is chosen from green channel, to get the binary value of cipher file.

Example:

![Figure 1. RGB Cover Image](image)

Encryption process:
Key of encryption is FILZA [2 3 4 5 1].
Figure 2. Image 2x1 pixel

Figure 2. shows that the image above is an image that will be encrypted using the Myszkowski Transposition algorithm. Then the first stage of the encryption process can be seen in Table 1. below:

| 2   | 3   | 4   | 5   | 1   |
|-----|-----|-----|-----|-----|
| 255 | 113 | 255 | 137 | 236 |

So that the results of image encryption will be seen in Figure 3. below:

Figure 3. Encrypted Image

Embedding process:

Insertion using the Green Channel MLSB, insertion will use the result data from encryption in the form of a cipher file as a cover. Message in the form of images below:

(a)                 (b)

Figure 4. The image to be inserted (a) is decimal (b) binary value

So, get an image from the embedding process like the image below:
Table 4. First Extraction Process

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 0 | R: 120 G: 0100111 B: 60 | R: 227 G: 1010101 B: 149 | R: 111 G: 0000111 B: 60 | R: 90 G: 0101010 B: 90 | R: 77 G: 1101001 B: 310 | R: 231 G: 0001111 B: 60 | R: 255 G: 0000000 B: 255 | R: 4 |
| 1 | R: 0 G: 0100011 B: 160 | R: 255 G: 0110011 B: 34 | R: 0 G: 0000001 B: 255 | R: 124 G: 0111111 B: 49 | R: 0 G: 0011111 B: 89 | R: 99 G: 0010110 B: 120 | R: 139 G: 0111101 B: 239 | R: 197 G: 1111111 B: 31 |
| 2 | R: 255 G: 1010101 B: 149 | R: 0 G: 0000001 B: 255 | R: 190 G: 0110001 B: 190 | R: 90 G: 0101010 B: 90 | R: 77 G: 1101001 B: 310 | R: 231 G: 0001111 B: 60 | R: 255 G: 0000000 B: 255 | R: 4 |
| 3 | R: 255 G: 0000000 B: 0 | R: 0 G: 0110100 B: 240 | R: 0 G: 0101000 B: 240 | R: 255 G: 1100001 B: 240 | R: 242 G: 1010001 B: 240 | R: 153 G: 1101110 B: 255 | R: 245 G: 0000001 B: 111 | R: 216 G: 1101001 B: 180 |
| 4 | R: 254 G: 1110111 B: 158 | R: 246 G: 1110001 B: 229 | R: 148 G: 1110001 B: 136 | R: 0 G: 0000001 B: 255 | R: 153 G: 1101110 B: 255 | R: 255 G: 0000000 B: 255 | R: 255 G: 0000000 B: 255 | R: 0 |
| 5 | R: 253 G: 1110111 B: 97 | R: 0 G: 0110100 B: 240 | R: 254 G: 0100000 B: 96 | R: 255 G: 1010001 B: 255 | R: 255 G: 1010001 B: 255 | R: 201 G: 0000001 B: 201 | R: 90 G: 0101000 B: 255 | R: 160 G: 1101001 B: 99 |

Extraction process:

Table 5. Second Extraction Process

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 0 | R: 120 G: 0100111 B: 60 | R: 227 G: 1010101 B: 149 | R: 111 G: 0000111 B: 60 | R: 90 G: 0101010 B: 90 | R: 77 G: 1101001 B: 310 | R: 231 G: 0001111 B: 60 | R: 255 G: 0000000 B: 255 | R: 4 |
| 1 | R: 0 G: 0100011 B: 160 | R: 255 G: 0110011 B: 34 | R: 0 G: 0000001 B: 255 | R: 124 G: 0111111 B: 49 | R: 0 G: 0011111 B: 89 | R: 99 G: 0010110 B: 120 | R: 139 G: 0111101 B: 239 | R: 197 G: 1111111 B: 31 |
| 2 | R: 255 G: 1010101 B: 149 | R: 0 G: 0000001 B: 255 | R: 190 G: 0110001 B: 190 | R: 90 G: 0101010 B: 90 | R: 77 G: 1101001 B: 310 | R: 231 G: 0001111 B: 60 | R: 255 G: 0000000 B: 255 | R: 4 |
| 3 | R: 255 G: 0000000 B: 0 | R: 0 G: 0110100 B: 240 | R: 0 G: 0101000 B: 240 | R: 255 G: 1100001 B: 240 | R: 242 G: 1010001 B: 240 | R: 153 G: 1101110 B: 255 | R: 245 G: 0000001 B: 111 | R: 216 G: 1101001 B: 180 |
| 4 | R: 254 G: 1110111 B: 158 | R: 246 G: 1110001 B: 229 | R: 148 G: 1110001 B: 136 | R: 0 G: 0000001 B: 255 | R: 153 G: 1101110 B: 255 | R: 255 G: 0000000 B: 255 | R: 255 G: 0000000 B: 255 | R: 0 |
| 5 | R: 253 G: 1110111 B: 97 | R: 0 G: 0110100 B: 240 | R: 254 G: 0100000 B: 96 | R: 255 G: 1010001 B: 255 | R: 255 G: 1010001 B: 255 | R: 201 G: 0000001 B: 201 | R: 90 G: 0101000 B: 255 | R: 160 G: 1101001 B: 99 |

Decryption process:
Table 6. Stage Decryption Process

|   | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 0 | 255 | 255 | 255 | 137 | 236 |
| 1 | 132 | 0 | 0 | 0 | 0 |

R: 255  G: 113  B: 255

Figure 7. Result image of the Decryption Process

In Figure 7, it can be seen that the previously encrypted image can be restored to its original value.

3. Results and Discussions

The experiments were performed on Windows 7. Notebook with Intel Celeron Processor, 32-bit architecture, and 4096MB RAM. The Integrated Development Environment (IDE) used for coding is SharpDevelop 4.3 and the programming language used is C#.

Table 7. Running time of encryption – embedding process

| Encryption and Embedding | Large Size Image (Pixels) | Processing Time (ms) |
|--------------------------|----------------------------|----------------------|
| Encryption Image         | 15x26 pixels               | 3 ms                 |
| Embedding Image          | 200x150 pixels             | 399.02 ms            |

Table 7 shows the results of testing the process time of encryption and embedding image messages. So that from the results of the test, the time for message encryption is 3 ms. And the results of testing the average insertion time is obtained for 399.02 ms. Where in the table above explains that, the larger the pixel size of the image, the longer the testing process.

Table 8. Running time of extraction – decryption process

| Extraction and Decryption | Large Size Image (Pixels) | Processing Time (ms) |
|---------------------------|----------------------------|----------------------|
| Extraction Image          | 200x150 pixels             | 3 ms                 |
| Decryption Image          | 15x26 pixels               | 8 ms                 |

In Figure 8, it appears that the difference in difference is only a little, so that it is not visible in plain or vague. In addition not all pixel elements that change pixel values.

(a) Cover Image  
(b) Stego Image

Figure 8. Comparing Cover Image (a) and Stego Image (b)
Table 8 shows the results of the extraction process time testing and image message decryption. So that from test results the time of extraction process is obtained for 3 ms. And on the average test the decryption process is obtained for 4 ms. In the table above explains that, in the process of extraction and decryption the results of the average processing time is less than 1 second. So as not to cause suspicion that there is a message in the image.

4. Conclusions
The conclusion of this research, based on the results of MSE and PSNR tests, it can be concluded that the changes that occur in the cover image are relatively invisible, so that using of myszkowski transposition algorithm and MLSB green channel is secure and simple. System is able to process encryption, embedding, extraction, and decryption. So that it can be concluded that the original image and process image are same. And it can be concluded that the larger of image size, the longer process will be.

5. Acknowledgments
The authors gratefully thank to all who have supported in the completion of this research. And to the Universitas Sumatera Utara that has given me the opportunity to publish this paper

References
[1] Alqahtani, J., Alghazzawi D., & Cheng, L. 2016. Steganalysis Algorithm For PNG Images Based On Fuzzy Logic Tehnique. International Journal of Network Security & Its Application (IUNSA). Vol 8, No 2, November 2016.
[2] Cheddad, A., J. Condell, K. Curran, P. Mc Kevitt. 2010. Digital Image Steganography: Survey and Analysis of Current Methods. Elsevier. Northern Ireland, UK, Signal Processing 90, 2010, pp.727-752.
[3] Farid, N., Bowo N., & Yuniarsi, R. 2016. Implementasi Metode Steganografi Least Significant Bit dengan Algoritma Hill Cipher pada Citra Bitmap. Jurusan Teknik Informatika. Universitas Dian Nuswantoro.
[4] Kolar, R. and Harabis, V., 2009. Automatic Rigid Registration and Analysis of Colour Fundus Image in Patient with Diabetic Retinopathy, IFMBE proceedings, vol 25/11, 251-24.
[5] McAndrew A. 2004. An Introduction to Digital Image Processing with Matlab. Notes for SCM2511 Image Processing 1. School of Computer Science and Mathematics Victoria University of Technology.
[6] Munir, R. 2006. Kriptografi. Penerbit Informatika Bandung. Bandung.
[7] Niswati, Z. 2012. Steganografi Berbasis Least Significant Bit (LSB) untuk Menyisipkan Gambar ke dalam Citra Digital. Faktor Exacta. Vol 5, No 2. ISSN: 1979-276X.
[8] Nurmaesah, N., Tutik L. & Arni R. M. 2017. Aplikasi Stegografi Untuk Menyisipkan Pesan Dalam Media Image. ISSN: 2339-1103.
[9] Pohan, R. Y. 2014. Studi dan Perbandingan Berbagai Macam Algoritma Cipher Transposisi. Jurusan Teknik Informatika ITB, Bandung.