The IJICS (International Journal of Informatics and Computer Science)
Vol 4 No 2, September 2020

ISSN 2548-8384 (online)
ISSN 2548-8449 (print)
Page 50 – 56
DOI 10.30865/ijics.v4i2.2120

Design of Steganographic Applications in A Processed Image using Algorithm Dynamic Markov Compression

Akmal Dirgantara Lubis, Garuda Ginting, Fadlina

Department of Computer Science, Universitas Budi Darma, North Sumatra, Indonesia
Email: akmaldir@gmail.com

Abstract−Confidential text data is an important matter that needs to be protected and kept confidential. Secret text data is a treasure where many people who want to try to find out first find out its contents. Therefore it is not uncommon for crimes to appear intentionally committed by irresponsible people. With the increasing number of people who commit crimes who deliberately steal confidential data and damage confidential text data so that it can harm certain parties. There have been several attempts to deal with the issue of security of confidential data sent over the internet, including using cryptographic and steganographic techniques.

Keywords: Steganography, Compression

1. INTRODUCTION

Confidential text data is an important matter that needs to be protected and kept confidential. Secret text data is a treasure where many people who want to try to find out first find out its contents. Therefore it is not uncommon for crimes to appear intentionally committed by irresponsible people. With the increasing number of people who commit crimes who deliberately steal confidential data and damage confidential text data so that it can harm certain parties. There have been several attempts to deal with the issue of security of confidential data sent over the internet, including using cryptographic and steganographic techniques. Cryptography is the study of how to maintain the confidentiality of data, keeping data or messages safe when sent, from sender to recipient without experiencing interference from third parties. (Ahmad Ramadoni Sitorus, Volume: iv, Number: 3, October 2014). While steganography (steganography) is the science and art of hiding secret messages in messages so that the existence of the secret message cannot be known [1].

Steganography is one of the techniques used in securing information, namely by hiding information into digital media with certain methods. So that no visual difference between the original file and the file that has been inserted information (stegoimage). So it is not known by people who can solve stegoimage without knowing the existing key. Digital data that can be a place or media data that will be hidden in steganography are images / images, audio and video [2].

2. THEORY

2.1 Steganography

Steganography (Steganography) is the science and art of hiding secret messages in other messages so that the existence of the secret message cannot be known. Steganography comes from the Greek language that is steganos which means hidden writing. Steganography is very contrast with cryptography. If cryptography conceals the meaning of the message while the existence of the message persists, then steganography covers the existence of the message. Steganography can be seen as a continuation of cryptography in practice secret messages are first encrypted, then ciphertext is hidden in other media so that third parties are not aware of its existence. The hidden message can be extracted back exactly the same as the original.

2.2 Compression

The compression process is the process of reducing the size of a data to produce a digital representation that is dense or incompressible but still can be represented by the quantity of information contained in the data. In imagery, video and audio, compression leads to minimization of the number of bit rates for digital representation. In some literature, the term compression is often also called source coding, data compression, bandwidth compression and signal compression.

2.3 Huffman Algorithm

Huffman algorithm is an image compression algorithm that uses a statistical approach. The sequence of steps to encode this algorithm is as follows.
1. Sort garyscale values based on frequency of occurrence.

ISSN 2548-8384 (online)
ISSN 2548-8449 (print)
Page 50 – 56
DOI 10.30865/ijics.v4i2.2120

2. Combine the two trees that have the smallest occurrence frequency and re-order.
3. Repeat step 2 until there is one binary tree remaining.
4. Label the binary tree with the left side of the tree labeled 0 and the right side of the tree labeled 1.
5. Trace the binary tree from the roots of the leaves. Barisa the side labels of the root of to the leaves are the huffman code.

3. RESULT AND DISCUSSION

The system analysis phase is the decomposition of a whole information system into its component parts with the aim of identifying and evaluating problems, opportunities, constraints and expected needs so that improvements can be proposed.

3.1 Embedded Process / Message Insertion

The workings of the Last Sihgnificant Bit (LSB) method in steganography are as follows:
1. Convert the image to be inserted into binary form
2. Convert the value of the degree of gray image level into binary numbers in the form of a matrix.
3. Take the bits of each byte of the image to be inserted into the binary blocks of the image as the container.
4. The inserted image bits will be placed at the end of the binary image by replacing the binary of the image in accordance with the bits of the inserted image
5. The image that has been inserted is called a stego image

The embedded message stage is the stage of inserting a message into a media container with the aim of hiding the message so that it is not seen or known by others who are not entitled to know it.

3.2 Message Extraction / Disclosure Process

Then, after inserting with the Last Sihgnificant Bit (LSB) method, the extraction / disclosure of the stego image will be performed to retrieve the image / message that has been inserted. The workings of the Last Sihgnificant Bit method in the process of disclosing the message are as follows.
1. Convert the stego image value to binary numbers in the form of a matrix.
2. Then match each stego image block with the inserted binary image
3. If appropriate, the image that has been inserted will be retrieved and can be proven.

3.3 Application of Dynamic Markov

Here's how the encoding and decoding of the Dynamic Markov compression algorithm works with the characters from the existing image matrix as follows:

$$\begin{pmatrix} 63 & 97 & 87 \\ 112 & 127 & 101 \\ 55 & 85 & 87 \\ 70 & 63 & 43 \\ 125 & 69 & 14 \end{pmatrix}$$

Figure 1. Image Matrix 3x5

Table 1. Calculation of compression results

No	Karakter	Dictionary	Kode bit
1	63	Ø 63	111111
2	97	Ø 97	1000011
3	87	Ø 87	1110101
4	112	Ø 112	0000111
5	127	Ø127	1111111
6	101	Ø101	1010011
7	55	Ø55	111011
8	85	Ø85	1010101
9	8770	39	11,0001001

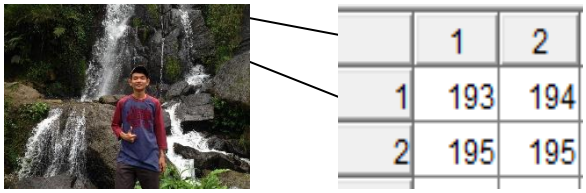
The IJICS (International Journal of Informatics and Computer Science)
Vol 4 No 2, September 2020
ISSN 2548-8384 (online)
ISSN 2548-8449 (print)
Page 50 – 56
DOI 10.30865/ijics.v4i2.2120

No	Karakter	Dictionary	Kode bit
10	6343	143	1,110101
11	125	Ø125	1011111
12	69	Ø69	1100011
13	14	Ø14	0111

Based on the binary code of each character, all files can be changed to: 111111 1000011 1110101 0000111 1111111 1010011 111011 1010101 11 0001001 1 110101 1011111 1100011 0111. Because the numbers 0 and 1 represent 1 bits, so the data bits above consist of 30 bits in other words the image size after compression using Dynamic Markov is 89 bits.

3.4 Application of the Least Significant Bit (LSB) Method

Suppose the message you want to insert is a 2 x 2 image = 4 pixels.

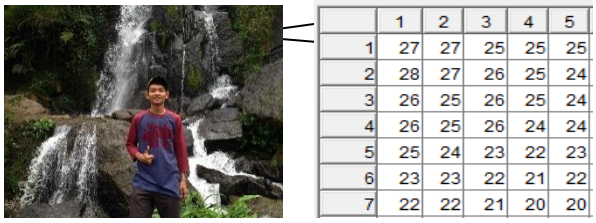


	1	2
1	193	194
2	195	195

Figure 2. Picture Becomes a Message

The binary value of the image to be inserted into the image is as follows:
193 = 11000001
194 = 11000010
195 = 11000011
195 = 11000011

The binary value above will be inserted into the image which will later be converted into a binary value. For example. Taken color images as the insertion media secret message, as follows.



	1	2	3	4	5
1	27	27	25	25	25
2	28	27	26	25	24
3	26	25	26	25	24
4	26	25	26	24	24
5	25	24	23	22	23
6	23	23	22	21	22
7	22	22	21	20	20

Figure 3. Image Pixel Value

Table 2. Binary Values of Figs

00011011	00011011	00011001	00011001	00011001
00011100	00011011	00011010	00011001	00011000
00011010	00011001	00011010	00011001	00011000
00011010	00011001	00011010	00011000	00011000
00011001	00011001	00010111	00010110	00010111
00010111	00010111	00010110	00010101	00010110
00010110	00010110	00010101	00010100	00010100

3.5 Embedded / Message Insertion

For the first steganographic process is the first pixel value of "193" whose binary value is "11000001". The last digit of the image bit will be replaced by the first bit value of the secret message bit.

Table 3. Before and After Inserted

Before Inserted			After Inserted	
27	00011011	inserted 1	0001101<u>1</u>	Unchanged
27	00011011	inserted 1	0001101<u>1</u>	Unchanged
25	00011001	inserted 0	0001100<u>0</u>	24
25	00011001	inserted 0	0001100<u>0</u>	24
25	00011001	inserted 0	0001100<u>0</u>	24

The IJICS (International Journal of Informatics and Computer Science)
Vol 4 No 2, September 2020
ISSN 2548-8384 (online)
ISSN 2548-8449 (print)
Page 50 – 56
DOI 10.30865/ijics.v4i2.2120

28	00011100	inserted 0	00011100	Unchanged
27	00011011	inserted 0	00011010	26
26	00011010	inserted 1	00011011	27

Furthermore, the steganography process of the second pixel value is "194" whose binary value is "11000010" The last digit of the image bit will be replaced by the first bit value of the secret message bit.

Table 4. Before and After Inserted

Before Inserted			After Inserted	
25	00011001	inserted 1	00011001	Unchanged
24	00011000	inserted 1	00011001	25
26	00011010	inserted 0	00011010	Unchanged
25	00011001	inserted 0	00011000	24
26	00011010	inserted 0	00011010	Unchanged
25	00011001	inserted 0	00011000	24
24	00011000	inserted 1	00011001	25
26	00011010	inserted 0	00011010	Unchanged

Furthermore, the steganography process of the third pixel value is "195" whose binary value is "11000011" The last digit of the image bit will be replaced by the first bit value of the secret message bit.

Table 5. Before and After Inserted

Before Inserted			After Inserted	
25	00011001	inserted 1	00011001	Unchanged
26	00011100	inserted 1	00011101	29
24	00011000	inserted 0	00011000	Unchanged
24	00011000	inserted 0	00011000	Unchanged
25	00011001	inserted 0	00011000	24
24	00011000	inserted 0	00011000	24
22	00010111	inserted 1	00010111	Unchanged
22	00010110	inserted 1	00010111	23

Furthermore, the steganography process of the fourth pixel value is "195" whose binary value is "11000011". The last digit of the image bit will be replaced by the first bit value of the secret message bit.

Table 6. Before and After Inserted

Before Inserted			After Inserted	
23	00010111	inserted 1	00010111	Unchanged
23	00010111	inserted 1	00010111	Unchanged
23	00010111	inserted 0	00010110	22
22	00010110	inserted 0	00010110	Unchanged
21	00010100	inserted 0	00010100	Unchanged
22	00010110	inserted 0	00010110	Unchanged
22	00010110	inserted 1	00010111	23
22	00010110	inserted 1	00010111	23

The results of the pixel values after steganography process are as follows:

Table 7. Steganographic Processes

27	27	24	24	24
28	26	27	25	25
26	24	26	24	25
26	25	29	24	24
24	24	23	23	23
22	23	22	21	22
23	23	21	20	20

3.6 Stego Image Extraction / Disclosure

After doing the embedded process, it is necessary to disclose the secret message so that it can be read by the recipient or user of the message, as follows:

00011011 27
00011011 27

ISSN 2548-8384 (online)
ISSN 2548-8449 (print)
Page 50 – 56
DOI 10.30865/ijics.v4i2.2120

00011000 24
00011000 24
00011000 24
00011100 28
00011010 26
00011011 27

It is clear from the last number in the binary block that the inserted message is 11000001, if it is converted to decimal number 193, the first pixel value of the image. Then for the disclosure of the second pixel is

00011001 25
00011001 25
00011010 26
00011000 24
00011010 26
00011000 24
00011001 25
00011010 26

The visible binary is 11000010. If converted to decimal number 194, the second pixel value. Then for the disclosure of the third pixel is

00011001 25
00011101 29
00011000 24
00011000 24
00011000 24
00010111 24
00010111 23
00011000 23

The visible binary is 11000110. If you change it to decimal number 195, it's the third pixel value. Then for the disclosure of the fourth pixel is

00010111 23
00010111 23
00010110 22
00010111 22
00010100 21
00010110 22
00010111 23
00010110 23

The visible binary is 11000010. If converted to decimal number 195, the fourth pixel value.

4. IMPLEMENTATION

Implementation is a step that is used to operate the application that is designed. In this case, it is explained how to run the application. The application processing program is a processing unit consisting of procedures and data implementation. Software is a system that is used for processing data or certain applications. Software used can be categorized, namely:

1. The operating system is to control all activities on the computer using the Windows 7 Ultimate operating system
2. Computers running the program are required to install Visual Basic 2008 and the like.

In the main view there are several menus, namely, the process menu, the about menu and the exit menu and in the process menu there are insert and compression menus, compression and extract menus.

Figure 4. Main course

The IJICS (International Journal of Informatics and Computer Science)
Vol 4 No 2, September 2020
ISSN 2548-8384 (online)
ISSN 2548-8449 (print)
Page 50 – 56
DOI 10.30865/ijics.v4i2.2120

Information:
a. The file menu contains the grayscale menu and exits
b. The lsb menu contains a menu insert message and extract message.
c. The huffman menu contains an image compression and image decompression menu.
d. The help menu contains the about me menu

This display will display the process from color image to grayscale image

Figure 5. Grayscale

Information:
a. Open image file button to display color images.
b. Grayscale button to display grayscale images.
c. Save button to save the image.
d. Exit button to exit the display.

This display will display the process of inserting messages into a grayscale image.

Figure 6. Insert message

Information:
a. Open image file button to open grayscale image.
b. Insert and save button to save the ordered grayscale image.
c. Exit button to exit the display.

In this view the process of decompressing the image will be displayed. As shown in the following image.

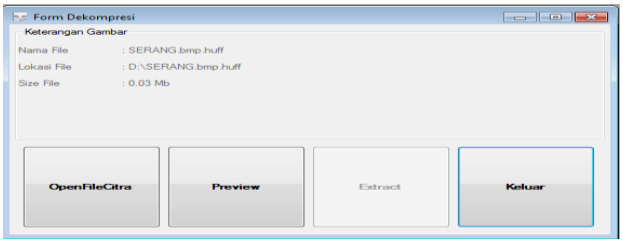


Figure 7. Caption

Information:
a. Image file open button to display image captions
b. View button to see image size.
c. Decompress button to see the decompress process.
d. Exit button to exit the display.

ISSN 2548-8384 (online)
ISSN 2548-8449 (print)
Page 50 – 56
DOI 10.30865/ijics.v4i2.2120

5. CONCLUSION

Based on the description and discussion that the author has done, some conclusions can be drawn relating to the software that has been designed, namely:

1. The process of the lsb method, converts the text to be hidden into binary form, then takes the bits from each byte of text to be hidden into binary blocks of the image. The hidden bits of text will be placed at the end of the binary image by replacing the binary of the image according to the bits of the inserted text.
2. Process of the Dynamic Marcov Compression Algorithm, calculates the number of types of characters and the number contained in a file, then arranges each character type in the least order to the most number, then makes a binary tree based on the character sequence from the smallest to the largest number , and member code for each character. Replacing existing data with bit codes based on binary trees and storing the number of bits for the largest bit code, the type of character ordered from the largest to the smallest exit frequency along with data that has been turned into bit codes as data compression results.
3. The design is done using the use case design tool for hiding messages in the image, and activity diagrams describe the processes running on the system, then designed using Microsoft Visual Basic 2008 with the main menu form, form insert and compression, form decompression and extract.

REFERENCES

[1] Adi Nugroho. 2009. Rekayasa Perangkat Lunak Menggunakan UML Dan. Java. ANDI, Yogyakarta
[2] Budi SutedJo, Dharma Oetomo. 2005 Perencanaan Dan Pembangunan Sistem Informasi Yogyakarta: Penerbit Andi.
[3] Dr. Jogiyanto H.M., M.B.A. 2003. Sistem Informasi Berbasis Komputer :Konsep Dasar dan Komponen. Edisi 2.Yogyakarta : BPFE Yogyakarta
[4] Ariyanto, Nina Setyaningsih, dan Edward Tanu Jaya, Pembuatan Aplikasi Profesional dengan Visual Basic.NET Jakarta: Salemba Infotek, 2009
[5] Munir, Rinaldi (2004). Diktat Kuliah IF5054 Kriptografi: Steganografi dan Watermarking. Institut Teknologi Bandung
[6] Nugroho, Adi, Rekayasa Perangkat Lunak Berorientasi Objek Dengan Metode USDP, 2010