RESEARCH ARTICLE OPEN ACCESS

Digital Watermarking on Images Using the Modified Least Significant Bit (MLSB) with Random Message Spread Using the Linear Congruential Generator (LCG) - Case Study: Integrated ICT Laboratory Budi Luhur University

Agung Sulistyanto[1], Abdullah ‘Alim[2], Hairil Fiqri Sulaiman[3]
1(Faculty of Information Technology, Budi Luhur Univesity, Jakarta
Email: agung.sulistyan@gmail.com)
2 (Faculty of Information Technology, Budi Luhur Univesity, Jakarta
Email: alimm.abdullah@gmail.com)
3 (Faculty of Information Technology, Budi Luhur Univesity, Jakarta
Email: hairilfiqri@gmail.com)

--************************----------------------------------

Abstract:

With the development of computer and internet devices that are increasingly fast making data and digital information more widely used. But some data may not be distributed or modified freely (without permission), because it contains the intellectual property rights of the creator. This violation is common in internet media but not all internet users are aware of it. So, digital content to be published must have rules in the use and protection that are widely recognized, so that the intellectual property rights of the creator are maintained. The way to protect intellectual property rights over digital works is to license content. In this final project, a Creative Commons License is used. Digital watermarking is a technique used to enter information that shows ownership of data and digital information. In this final project, the watermark mark will be attached to the digital image as a companion to the commonly used digital content. License messages are inserted using the Modified Least Significant Bit (MLSB) method as the insertion and compression of message bits, so that fewer messages are inserted. Then the method of generating Pseudo Random Number Generator (PRNG) with Linear Congruential Generator (LCG) to insert messages randomly.

Keywords —**Copyright, Digital Watermarking, Least Significant Bit (LSB), Modified Least Significant Bit (MLSB), Linear Congruential Generator (LCG).**

--************************----------------------------------

I. INTRODUCTION

With the development of computer and internet devices that are increasingly fast making data and digital information more widely used. It can be noted that the internet facilitates the wide and open exchange of data. But some data may not be distributed or modified freely (without permission), because it contains the intellectual property rights of the creator. This violation is common in internet media but not all internet users are aware of it.

So, a digital data that will be published must have rules in the use and protection that are widely recognized, so that the intellectual property rights of the creator are maintained. The way to protect intellectual property rights over a digital work is by licensing the content.

License information needs to be entered in other media, in order to survive various activities in an effort to eliminate it. One digital media that can be used as a message container is digital images. Digital imagery was chosen because it is widely used as a complement to text-based content. So with the use of digital images, the license information entered can withstand various attacks in an effort to eliminate it.

This research is limited to digital images that will be used are BITMAP and PNG formats (not transparent background). And messages that are stored are limited to characters A to Z, a to z and 1 to 0.

II. THEORETICAL BASIS

A. *Digital Watermarking*

Watermarking is a way to insert a *watermark* or the process of adding code permanently into a digital image that wants to be protected by not damaging the original image and is resistant to attack (Munir, 2006).

B. *LeastSignificantBit*

Least Significant Bit (LSB) is part of a binary data row that has the least meaningful or smallest value. The LSB bit is located on the far right of the binary row, because the value of 1 LSB bit in the binary sequence only represents the value of 1 decimal, then this bit is considered meaningless. So if there is a change in the LSB bit value then there will be no significant change.

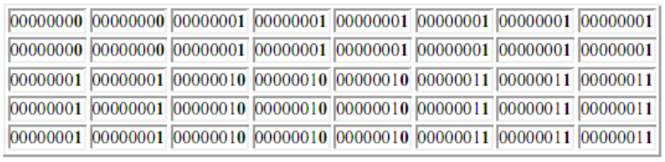
Fig 1 Bananas.jpg

Suppose the data in the form of "secret" text will be inserted into the image. If represented in *binaries* the word "secret" becomes,

TABLE 1
ASCII CONVENSION “secret” TEXT

Character	*ASCII value (decimal)*	*Hexadecimal*	*Binary*
s	115	73	01110011
e	101	65	01100101
c	99	63	01100011
r	114	72	01110010
e	99	63	01100011
t	116	74	01110100

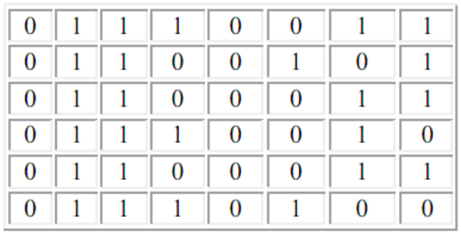
Suppose the binary value of the image bananas.jpg as follows.



00000000	00000000	00000001	00000001	00000001	00000001	00000001	00000001
00000000	00000000	00000001	00000001	00000001	00000001	00000001	00000001
00000001	00000001	00000010	00000010	00000010	00000011	00000011	00000011
00000001	00000001	00000010	00000010	00000010	00000011	00000011	00000011
00000001	00000001	00000010	00000010	00000010	00000011	00000011	00000011

Fig 2 Bananas.jpg binary value

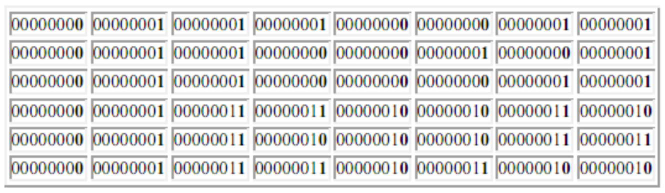
So in its application the bit value in the word "secret" will be inserted or replace the 8th bit value in binary bananas.jpg.



0	1	1	1	0	0	1	1
0	1	1	0	0	1	0	1
0	1	1	0	0	0	1	1
0	1	1	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0

Fig 3 The binary value of the "secret" text

Final result (image-stego),



00000000	00000001	00000001	00000001	00000000	00000000	00000001	00000001
00000000	00000001	00000001	00000000	00000000	00000001	00000000	00000001
00000000	00000001	00000001	00000000	00000000	00000000	00000001	00000001
00000000	00000001	00000011	00000011	00000010	00000010	00000011	00000010
00000000	00000001	00000011	00000010	00000010	00000010	00000011	00000011
00000000	00000001	00000011	00000011	00000010	00000011	00000010	00000010

Fig 4 Bananas.jpg binary value after the message is inserted

C. *Modified Least Significant Bit (MLSB)*

Modified Least Significant Bit (MLSB) or modification of the LSB algorithm is used to encode an identity into the original image. MLSB uses the manipulation of various insertion bits before encoding the message (Zaher, 2011). Modify messages with the MLSB algorithm where the message bits that should be 1 character have a value of 8 ASCII bits will be modified to 5 bits (31 decimal places). In this algorithm the characters and numbers are represented in 5 bits before being inserted into the original image with the LSB technique. Changing the ASCII value to 5 bits is carried out with the following processes:

a. The process of changing the insertion data with ASCII code. For example the message to be inserted the sentence "STEGO with 05 bits" which if converted into ASCII binaries requires memory of 18 x 8 bits = 144 bits. In the MLSB algorithm the message above is changed to ASCII (hex): $53_{(16)}$, $54_{(16)}$, $45_{(16)}$, $47_{(16)}$, $4F_{(16)}$, $20_{(16)}$, $77_{(16)}$, 69 (16), $74_{(16)}$, $68_{(16)}$, $20_{(16)}$, $30_{(16)}$, $35_{(16)}$, $20_{(16)}$, $62_{(16)}$, $69_{(16)}$, $74_{(16)}$, $73_{(16)}$. Then do the normalization process with the *Control Symbol* as in the table.
b. Read the insertion mark (ASCII) until the space mark $20_{(16)}$, that is $53_{(16)}$, $54_{(16)}$, $45_{(16)}$, $47_{(16)}$, $4F_{(16)}$.
c. Because it is a line of uppercase characters, all values are XOR with the value of dozens of ASCII of the hex value of the smallest character in capital letters. The smallest character value is 'A' ie $41_{(16)}$ so $53_{(16)} \oplus 40_{(16)} = 13_{(16)}$, $54_{(16)} \oplus 40_{(16)} = 14_{(16)}$, $45_{(16)} \oplus 40_{(16)} = 05_{(16)}$, $47_{(16)} \oplus 40_{(16)} = 07_{(16)}$, $4F_{(16)} \oplus 40_{(16)} = 0F_{(16)}$.
d. So we get the first group insertion data are $IC_{(16)}$, $13_{(16)}$, $14_{(16)}$, $05_{(16)}$, $07_{(16)}$, $0F_{(16)}$, $1D_{(16)}$ where $IC_{(16)}$ is Control Symbol for uppercase and $ID_{(16)}$ are Control Symbol values that are used to replace the space value $20_{(16)}$.
e. The second group insertion data are $77_{(16)}$, $69_{(16)}$, $74_{(16)}$, $68_{(16)}$ reduced by the scores of the smallest dozens of characters 'a' $61_{(16)}$, to $77_{(16)} \oplus 60_{(16)} = 17_{(16)}$, $69_{(16)} \oplus 60_{(16)} = 09_{(16)}$, $74_{(16)} \oplus 60_{(16)} = 14_{(16)}$, $68_{(16)} \oplus 60_{(16)} = 08_{(16)}$.
f. The second group data will be combined with the first group data into $IC_{(16)}$, $13_{(16)}$, $14_{(16)}$, $05_{(16)}$, $07_{(16)}$, $0F_{(16)}$, $ID_{(16)}$, $IB_{(16)}$, $17_{(16)}$, $09_{(16)}$, $14_{(16)}$, $08_{(16)}$.
g. Third group data are $30_{(16)}$, $35_{(16)}$ reduced by the scores of the smallest group of numbers '0' ie $30_{(16)}$ to $30_{(16)} \oplus 30_{(16)} = 00_{(16)}$, $35_{(16)} \oplus 30_{(16)} = 05_{(16)}$.
h. Third group data will be merged with the previous group data and added with Control Symbol $ID_{(16)}$ for spaces and $IE_{(16)}$ for numbers to be $ID_{(16)}$, $IE_{(16)}$, $00_{(16)}$, $05_{(16)}$.
i. Data for the fourth group are $62_{(16)}$, $69_{(16)}$, $74_{(16)}$, $73_{(16)}$ reduced by the scores of the lowest scores of 'a' $61_{(16)}$ to $62_{(16)} \oplus 60_{(16)} = 02_{(16)}$, $69_{(16)} \oplus 60_{(16)} = 09_{(16)}$, $74_{(16)} \oplus 60_{(16)} = 14_{(16)}$, $73_{(16)} \oplus 60_{(16)} = 13_{(16)}$.
j. The fourth group data will be combined with the previous group data and added with Control Symbol $ID_{(16)}$ for spaces and $IB_{(16)}$ for lowercase letters and $IF_{(16)}$ for the end of the data to be $ID_{(16)}$, $IB_{(16)}$, 02 (16), $09_{(16)}$, $14_{(16)}$, $13_{(16)}$, $IF_{(16)}$.

So the messages to be inserted are $IC_{(16)}$, $13_{(16)}$, $14_{(16)}$, $05_{(16)}$, $07_{(16)}$, $0F_{(16)}$, $ID_{(16)}$, $IB_{(16)}$, $17_{(16)}$, $09_{(16)}$, $14_{(16)}$, $08_{(16)}$, $ID_{(16)}$, $IE_{(16)}$, $00_{(16)}$, $05_{(16)}$, $ID_{(16)}$, $IB_{(16)}$, $02_{(16)}$, $09_{(16)}$, $14_{(16)}$, $13_{(16)}$, $IF_{(16)}$. This message only needs 23 x 5 bits = 115 bits which if converted to binary data becomes 11100, 10011, 10100, 00101, 00111, 01111, 11101, 11011, 10111, 01001, 10100, 01000, 11101, 11110, 00101, 00101, 11101, 11011, 00010, 01001, 10100, 10011, 11111. After being converted to a maximum value of 5 bits, the message bits will be inserted using the Least Significant Bit (LSB) technique. To get the original message (decoded), XOR is performed again on the message bytes with the lowest dozens value corresponding to the group of numbers and removes the Control Symbol value.

D. Random Number Generation

Random numbers cannot be predicted. Random numbers generated by mathematical formulas are pseudo random numbers, because the generation of numbers can be repeated again. This generation of random numbers is called a *Pseudo Random Number Generator* (PRNG). PRNG produces a sequence of values that each value depends on the value previously generated. However, the output of the PRNG is not completely random because in the loop it can find the same value. *Linear Congruential Generator* (LCG) is one of the oldest and most popular pseudo random number algorithms. This algorithm was invented by D. H. Lehmer in 1951. Random numbers on LCG are obtained through the following equation:

$$X_{n+1} = (aX_n + c) \mod m$$

Fig 5 Linear Congruential Generator (LCG)

Determination of initial values X_0 and constants (a, b and m) will determine the quality of random numbers generated. The largest LCG period is modulus m, even in most cases the period is less than the module. Because of its random nature and its value can be resurrected, it can be used as an address to insert messages randomly and as a key in re-extracting messages that have been inserted in steganographic techniques.

E. Creative Commons License

The Open Content Principle is based on the ideas of the *Free and Open Source Software* (FOSS) movement. The Open Source Approach began in the software market in the 1990s. Basically it stems from the great success of GNU-Linux and its license, the GNU *General Public License* (GPL). The main originator of the Open Content movement was Lawrence Lessig, in 2001, he joined Hal Abelson and Eric Eldred and founded the Creative Commons (CC) initiative to promote digital ownership (music, film, pictures) together.

CC is the most extensive open content licensing model. The popularity and wide use of this license means CC is considered a standard for open content licenses. CC provides a set of ones consists of six licenses as shown in Figure 3. Each license contains one or several elements from a total of four basic elements ("licensing features") which are illustrated by pictograms as follows.

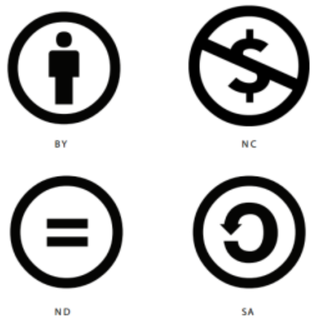


Fig 6 Pictogram of CC licensing features

a. CC BY (Attribution)
   BY is an abbreviation for attribution is an obligation to include the name of the creator and other parties that must be named.
b. CC NC (Non-Commercial)
   NC is a commercial use not permitted by license.
c. CC ND (No Derivatives)
   ND is only a verbatim copy of a work that can be shared.
d. CC SA (Share Alike)
   SA is a similar sharing where a derivative work can only be disseminated with the same license as the licensed work.

III. SYSTEM DESIGN AND APPLICATION

In designing this application there are two main functions, the message insertion process (*embed*) and the message extraction process (*extract*). In Figure 3.1 the flow of the message insertion process is embed and in Figure 3.2 the flow of the message extraction process is explained.

A. Message Insertion

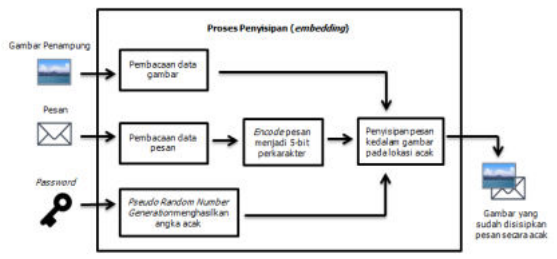

Proses Penyisipan (embedding)
Gambar Penampung
Pesan
Password
Pembacaan data gambar
Pembacaan data pesan
Encode pesan menjadi 5-bit perkarakter
Penyisipan pesan kedalam gambar pada lokasi acak
Pseudo Random Number Generation menghasilkan angka acak
Gambar yang sudah disisipkan pesan secara acak

Fig 7 Insertion Process Flow

In the process of inserting a message (*embedding*) there are stages which are sequentially explained as follows:

a. Read the Input File

At this stage, there are three entries, a message container image, a message to be inserted, and a password that will be used as a key in making random numbers. To select the image to be used as a message container, in the embed section there is a "Pilih" button that will display File Chooser to select the desired image as a container. The image that has been selected as a container will be duplicated. The duplicate files will be sampled by byte based on the RGB value of each pixel, this byte data will be processed for inserting the message. In the message and password the data byte is obtained by converting the characters entered with the ASCII value.

b. Encode Message

At this stage the message bytes that have been obtained will be encoded with *Modified Least Significant Bit* (MLSB). In the MLSB method the ASCII byte value for a character that initially has a maximum value of 8 bits (255 decimal) is encoded so that it only has a maximum value of 5 bits (31 decimal places). This coding is generated based on the Symbol Control table that was explained in Chapter II. After being encoded, this byte will be inserted into the container image byte.

c. Make Random Numbers

At this stage the password byte that has been obtained will be used to generate a number of random numbers using the *Pseudo Random Number Generator* (PRNG) *Linear Congruential Generator* (LCG) method that was explained in Chapter II. The ASCII value of each character will be looped and become one of the parameters in forming a random value. This random value will be used as the address for inserting the message byte into the image byte.

d. Insert Message

This stage is the core of the embed process, which is inserting a message. After encoding the message bits into 5 bits and forming random numbers, the next step is to insert the message bytes into random image bytes. Based on random numbers formed using the *Least Significant Bit* (LSB) method that was explained in Chapter II. In the process of inserting 1 byte the message will be represented as 8 bits. Because the message value has been previously encoded to be 5 bits (31 decimal), the 6th, 7th and 8th bits do not need to be inserted because the bit value is definitely 0.

e. Constructing an Image

Once inserted, at this stage the sample byte image is reconstructed back into a digital image that already has a secret message in it (*stego-image*).

B. *Message Extraction*

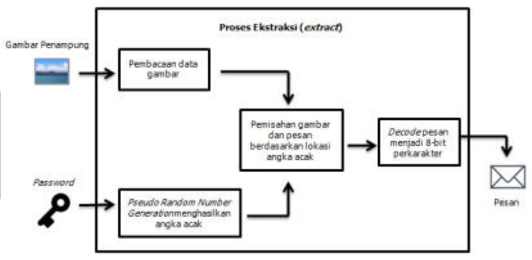

Proses Ekstraksi (extract)
Gambar Penampung
Pembacaan data gambar
Pemisahan gambar dan pesan berdasarkan lokasi angka acak
Decode pesan menjadi 8-bit perkarakter
Password
Pseudo Random Number Generation menghasilkan angka acak
Pesan

Fig 8 Extraction process flow

a. Read the Input File

At this stage, there are two inputs, an image that already contains a message and a password that will be used as a key in determining the location where the message bits are inserted. To select the image to be extracted, in the extract section there is a "Pilih" button that will display the File Chooser to select the image to extract the message in. The selected image will be taken by sample byte based on the RGB value of each pixel, this byte data will be processed for message extraction. The password is obtained by byte data by converting the characters entered with the ASCII value.

b. Make Random Numbers

At this stage, the password byte that has been obtained will be used to generate a number of random numbers using the *Pseudo Random Number Generator* (PRNG) *Linear Congruential Generator* (LCG) method that was explained in Chapter II. The ASCII value of each character will be looped and become one of the parameters in forming a random value. The password used during the embed and extract process must be the same to get the same random value. This random value will be used as an address to get the value of the message bits in the image.

c. Splitting the Message Data on the Image

At this stage the message bits in the image are obtained by taking 1 *Least Significant Bit* (LSB) from the image bytes sequentially based on the location determined by the random value. Every 5 bits of image obtained, are reconstructed back to an 8 bit byte value. This process is repeated until all messages are obtained.

d. Decode the Message

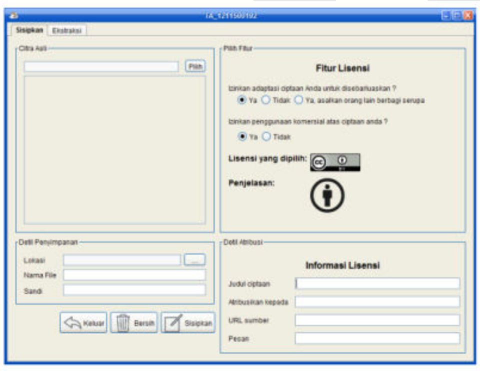
At this stage the message byte value for each character has been obtained, but is still encoded for a maximum of 5 bits (31 decimal places). As explained in Chapter II, at this stage the message byte is decoded to return the ASCII value for each character using the *Modified Least Significant Bit* (MLSB) method.

e. Constructing the Message

After the ASCII byte value of the message is obtained, the message character is written to the file with the extension .txt and saved according to the given file name.

IV. RESULT AND DISCUSSION

Display the main page which also displays the form insert. The "Pilih" button is used to select the image to be used as a license message container. In the Select Features section is enabled for choosing the license feature that is used there is also an explanation on each selected license combination. In the Attribution Detail section, it is used to fill in detailed information of the license used. The Insertion section is used to complete the output location and the output image name and password as the insertion key. To start the message insertion process you can use the "Sisipkan" button, to clean input with the "Bersih" button and to exit the application you can use the "Keluar" button.



Izinkan adaptasi ciptaan Anda untuk disebarluaskan ?
[✓ Ya] Tidak Ya, asalkan orang lain berbagi serupa

Izinkan penggunaan komersial atas ciptaan anda ?
[✓ Ya] Tidak

Fig 9 Display Embed Form

In the extraction form, it functions to issue a license message that has been previously inserted in the container image. The "Pilih" button is used to select the image to extract the message in it. In the extraction section there is a location and file name for storing outputs and a password that is used as a key in the extraction process. After that to start the message extraction process you can use the "Ekstrak" button, to clean the input with the "Bersih" button and to exit the application can use the "Keluar" button.

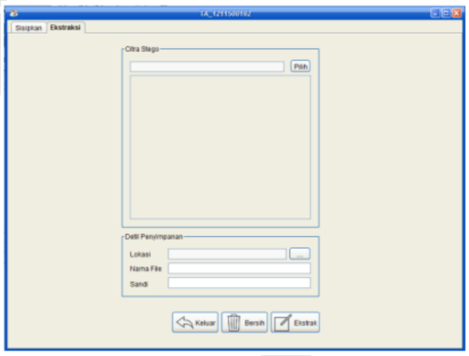


Fig 10 Display Extraction Form

A. Advantages

The advantages of this digital watermarking application are as follows:

a. Using a randomization algorithm so messages that are inserted are more difficult to detect.
b. Using a randomization algorithm so the stego image is more similar to the original image.
c. The process of insertion and extraction is relatively fast because the application is desktop-based so that the source of the image is more quickly accessed by the application.
d. Easy to use because of a simple application design.

B. Disadvantages

The disadvantages of this digital watermarking application are as follows:

a. Desktop-based application, so to use it the user needs to install the application first.
b. Have not been able to insert watermarks on many images directly.
c. Changed image (cropping, resizing, etc.) can damage the message that is inserted.

V. CONCLUSIONS

In accordance with this discussion about digital watermarking, the conclusions that can be drawn by the author are as follows:

a. By using a password to determine the random position of the insertion, the security of the message inserted in the image is maintained safely.
b. The *Modified Least Significant Bit* (MLSB) method used has been running well, it can be noted in the test table in Chapter 4 that the message bit size that is inserted becomes smaller than it should be.
c. It can be noted in crocodile.bmp insertion, the more varied the characters that are inserted, the more character Control Symbols are used to make the size of the inserted message larger than the original size.
d. An attempt was made (not attached) to insert an image containing a license message (image-stego) into other document media (PDF). In the results of the experiment the message was extracted successfully, but it should be noted in separating the image-stego with the document (PDF), so as not to change the format, size, color value of the image-stego that was previously inserted.
e. In the insertion of PNG images with transparent background, the background changes to black after the insertion process.
f. PNG image insertion in the ARGB color format can reduce image size because the image is converted to the RGB color format before the message is inserted.
g. In the insertion of PNG images with a transparent background, the background changes to black after the insertion process.

REFERENCES

[1] Hidayat, Erwin Yudi, & Hastuti, Khafiizh 2013, Analisa Steganografi Metode Least Significant Bit (LSB) Dengan Penyisipan Sekuensial Dan Acak Secara Kuantitatif Dan Visual, Semarang, Jurnal Techno.com Vol.12, No. 3.

[2] Imam, Khairul 2013, Penyembunyian Pesan Rahasia Pada Citra Digital Dengan Teknik Steganografi Menggunakan Metode Least Significant Bit (LSB), Jakarta, Universitas BUDI LUHUR.

[3] Kadir, Abdul, & Susanto, Adhi 2013, Teori Dan Aplikasi Pengolahan Citra, Yogyakarta, Andi Publisher.

[4] Keutzer, Till 2014, Konten Terbuka – Pedoman Praktis Penggunaan Lisensi Creative Commons, ISBN: 978-602-72890-1-7, Jakarta, Wikimedia Indonesia.

[5] Lubis, Rifki Respati Ashari 2015, Analisis Kombinasi Algoritma Watermarking Modified Least Significant Bit (MLSB) Dengan Least Significant Bit+1, Sumatera, Universitas Sumatera Utara.

[6] Munir, Rinaldi 2004, Diktat Kuliah IF5054 Kriptografi : Steganografi dan Watermarking, Bandung, Institut Teknologi Bandung.

[7] Sakti, Dolly Virgian Shaka Yudha, Agani, Nazori, Hardjianto, Mardi 2015, Pengaman Sistem Menggunakan One Time Password Dengan Pembangkitan Password Hash SHA- 256 Dan Pseudo Random Number Generator (PRNG) Linear Congruential Generator (LCG) Di Perangkat Berbasis Android, Jakarta, Universitas BUDI LUHUR.

[8] *Sulistyanto, Agung 2015, Materi Pelatihan : Aplikasi Steganografi Dengan Metode LSB dan Enkripsi Pesan Dengan Pembangkitan Bilangan Acak, Jakarta, Laboratorium ICT Terpadu Universitas BUDI LUHUR.*

[9] Vebrina, Yus Gias 2014, Makalah : Spread Spectrum Steganography, Bandung, Institut Teknologi Bandung.

[10] Zaher, Mazen Abu 2011, Modified Least Significant Bit (MLSB), Kanada, Jurnal CCSENET Vol. 4, No. 1.