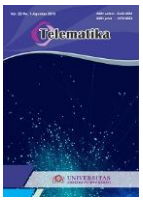


Terbit *online* pada laman web jurnal :
http://ejournal.amikompurwokerto.ac.id/index.php/telematika/

**Telematika**

Accreditated SINTA "2" Kemenristek/BRIN, No. 85/M/KPT/2020

# The Evaluation of Image Messages in MP3 Audio Steganography Using Modified Low-Bit Encoding

**Ilham Firman Ashari**

Teknik Informatika / Sub Jurusan Teknik Elektro, Informatika, dan Sistem Fisika
Institut Teknologi Sumatera
E-mail:firman.ashari@if.itera.ac.id

**ARTICLE INFO**

***History of the article:***
Receive October 7, 2020
Revised February 12, 2021
Received June 29, 2021
Available online August 31, 2021

**Keywords:**
Steganography
Cryptography
Modified Low Bit Encoding
Base64
Information Security

**Correspondence:**
Telepon: +62 721 8030188
E-mail:
firman.ashari@if.itera.ac.id

**ABSTRACT**

Information security is an important aspect of maintaining the confidentiality of information. One type of document kept secret is images (.jpg, .png, .gif, dan .bmp). MP3 audio files are popular audio files that can be used as a medium for steganography. The method implemented in this study uses Base64 in the image and the insertion method used is using Low-Bit Encoding (LBE). In this study, the parameters of the value of LBE will be evaluated. The purpose of the evaluation is to compare the LBE parameter parameters that are the most optimal in securing the message while taking into account the quality of steganographic files. The results obtained from the study are the proposed method that supports the imperceptibility aspect seen from the image histogram and audio frequency spectrum. The fidelity aspect is seen from the PSNR and SNR values, where The optimal value on the LBE + 2 parameter is by considering the capacity of the picture message that can be inserted and audio quality. PSNR LBE + 2 values range from 50-60 dB with SNR different about 0.01% from LBE + 1. The proposed method does not support the robustness aspect because it is not resistant to attacks by bit rate manipulation and channel mode. The test results on the recovery aspect are worth 100%, meaning that the image's quality and size before and after extraction will be the same. And finally, the test results on the payload aspect, there is an increase in message capacity with LBE + 2 around 12.5% of LBE + 1, and using LBE + 3 will increase the maximum size around 25% of LBE + 1 and about 14% of LBE + 2. The insertion and extraction time for LBE + 3 is slower compared to the others.

## INTRODUCTION

The development of information technology is growing rapidly, especially in the field of internet and multimedia, so that many provide convenience to the community (Choi et al., 2020). The positive impact of the development of information technology is to provide comfort in accessing information (Payam et al., 2017)(Choi et al., 2020). Along with the development of information technology, this can have positive and negative impacts (Payam et al., 2017).

A variety of modern techniques are widely used by people who are not responsible for being able to obtain information that is not their ownership, therefore need a method to secure information (Gurgaon, 2017). One technique that can be used to secure information is cryptography (Bansal et al., 2020)(Ashari, 2018). Another technique that can be used for information security, without giving rise to perception is steganography (Rehman et al., 2018). To improve information security, cryptography can be combined with steganography (Taha et al., 2019).

The security aspect of using cryptographic messages will guarantee the confidentiality, integrity, and availability of the message (Gaikwad et al., 2015). According to Douglas (2018), three criteria must be met in steganography, namely fidelity, where the quality of the media or the cover of the container is not much different from the original file after the message is inserted. Then robustness, the hidden data must be resistant (robust) to various manipulation attacks, generally, no steganography method is resistant to attacks, because the main purpose of steganography is to eliminate the perception that there is a secret message, and the last criterion is recovery, the hidden data must be able to Re-disclosed, the information disclosed must be in the original condition as before (Roy & Changder, 2016).

The methods in steganography can be divided into two, namely the spatial domain method and the frequency domain method (Singh et al., 2015). The spatial domain method maps and converts bytes or bits directly from the media cover with the message to be inserted. Steganographic methods are based on the spatial domain, that's LSB (Hussain et al., 2018). Another method of steganography is the frequency domain, such as DCT (Discrete Cosine Transform) and DWT (Domain Wavelet Transform) (Kadhim, 2017). The advantage of using spatial domain-based steganography methods is the value of PSNR (Peak Signal to Noise Ratio) and MSE (Mean Square Error) which is a standard for measuring and determining the quality or difference between files before the message is embedded and after the message is extracted. The result from the spatial method, better than the frequency domain in capacity, but in terms of frequency domain security is better (Sairam & Bhoopaty, 2018). The weakness of this spatial domain method can be covered by using a combination of steganography and cryptography (Bansal et al., 2020).

Audio files that are currently popular that are common and are widely distributed on the internet media are MP3 audio files, although there are several kinds of audio files such as WAV, MIDI, and FLAC (Jenkins & Laurier, 2013). According to (Denegri-knott, 2015), MP3 audio is becoming popular because it is a good audio compression in terms of size and sound quality, this is indicated by the increasing spread and use of MP3 audio files on internet media. Due to the popularity of MP3 audio, MP3 audio can be used as a means of securing information. In terms of size, MP3 are larger than image files, which is certainly comparable to messages that can be inserted into MP3s more than in image files. MP3 is not only a sound player file but is an option that can be used as an alternative to audio steganography (Dutta et al., 2019).

The steganography method used in the study is the Low-Bit Encoding (LBE) method. LBE is a technique by changing the less influential bits with bits of information that are inserted (Chowdhury et al., 2016) (Parthasarathi et al., 2017). The implementation of the LBE method is becoming more popular and simpler in the message insertion process if compared to other methods such as phase coding, echo hiding, and spread spectrum (Malviya et al., 2012)(Chakraborty, 2019). In its development, it is known that the use of the LBE method in audio steganography is vulnerable to statistical analysis and steganalysis (Rokhman & Maharanti, 2011). To improve security, before using the LBE method for image insertion, the data is encrypted first using Base64.

Some of the most recent studies on audio steganography have been conducted by (Mulyono et al., 2018), (Setiaji et al., 2015). Mulyono (2018) proposed the steganography method using WAV audio cover media by using the vigenere cipher encryption algorithm, the inserted message in the form of a text message. The results obtained by PSNR and MSE are good. Setiaji (2015) proposed an audio steganography method with an insertion scheme using MELSB, which is a variation of ELSB. The object in this study uses an audio file with WAV extension. By using the concept of bit position variation makes the presence of a message not easy to detect. PSNR obtained from research is of good value. Mengusulkan metode

steganografi pada audio WAV dengan menggunakan LSB dan enkripsi dengan vigenere cipher. Dimana pesan akan dienkripsi dengan vigenere, kemudian disisipkan pada bit lSB dari audio WAV.

The storage media used in this study are MP3 audio files, due to the popularity of MP3 audio files. Text files are very common for steganography, therefore in this study using messages with image formats (.jpg, png, BMP, and gif). To simplify and secure picture messages, and in converting bytes, storing or extracting bytes, the Base64 algorithm is used. The focus of this research will be to compare each parameter using the LBE encoding method. This LBE method can be modified. In this research, LBE degree 1 bit, LBE degree 2 bit, LBE degree 3 bit will be compared, the purpose is to find out what degree of modification is the most optimal for quality and capacity with good PSNR and MSE values.

## RESEARCH METHODS

In this study, audio steganography was implemented using the Java programming language with Netbeans 8.2 RC IDE, with testing using a core I7 9750H processor and internal storage of the Samsung 970 Evo Plus 1 TB NVME SSD. This study will be divided into two main stages, namely the stages of encoding and embedding as stages for image insertion and extraction and decoding stages as stages for image extraction.

### A. Encoding and Embedding Method

A general description of the work process scheme of the encoding and embedding method can be seen in Figure 1. In figure 1, the initial stage is the encoding process, the user enters two files namely an image file and an MP3 audio file as a cover, besides that the user needs to enter the LBE variable value that will be used for the embedding process. The inputted image will be validated to check the image format, if the entered format is other than the image (.jpg, .png, .bmp, and .gif), it will be rejected. After the validation, the image will be converted into byte stream form and encoded using base64 encoder to be a text encoder string. The encoder text string will then be converted into a byte stream format and the results will be temporarily stored in a variable. MP3 audio files that have been inputted will be converted into byte stream form and the embedding process will be performed using LBE, by taking the byte stream from MP3 audio and byte stream from the image. After that, the results will be saved in the form of byte streams and will be converted into images.

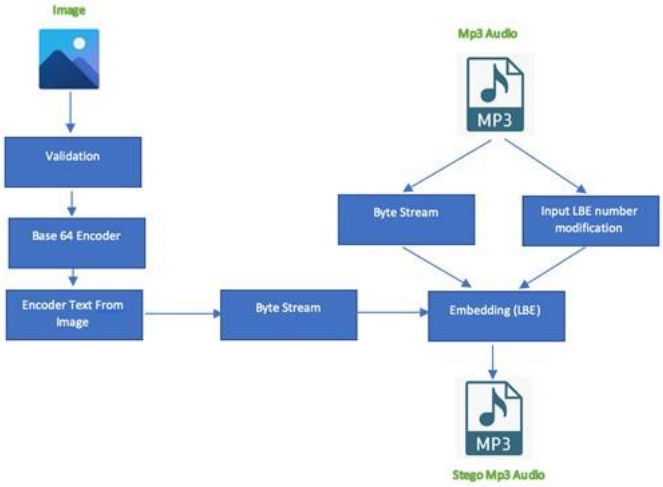



Figure 1. General description of encoding and embedding

## 1. Metode Base64 Encoding

The Base64 algorithm is an encoding algorithm that can be used for multimedia files. In the encoding and decoding process, a Base64 index table is needed which can be seen in Table 1, This table is used to represent ASCII strings to Base64 strings.

Table 1. Index Base64

| *I* | *V* | *I* | *V* | *I* | *V* | *I* | *V* | *I* | *V* |
|---|---|---|---|---|---|---|---|---|---|
| 0 | A | 14 | O | 28 | c | 42 | q | 56 | 5 |
| 1 | B | 15 | P | 29 | d | 43 | r | 57 | 6 |
| 2 | C | 16 | Q | 30 | e | 44 | s | 58 | 7 |
| 3 | D | 17 | R | 31 | f | 45 | t | 59 | 8 |
| 4 | E | 18 | S | 32 | g | 46 | u | 60 | 9 |
| 5 | F | 19 | T | 33 | h | 47 | v | 61 | 0 |
| 6 | G | 20 | U | 34 | i | 48 | w | 62 | + |
| 7 | H | 21 | V | 35 | j | 49 | x | 63 | - |
| 8 | I | 22 | W | 36 | k | 50 | y | | |
| 9 | J | 23 | X | 37 | l | 51 | z | | |
| 10 | K | 24 | Y | 38 | m | 52 | 1 | | |
| 11 | L | 25 | Z | 39 | n | 53 | 2 | | |
| 12 | M | 26 | a | 40 | o | 54 | 3 | | |
| 13 | N | 27 | b | 41 | p | 55 | 4 | | |

*Note : Index = I, Value = V

Suppose that the byte string is known from the file ilhamfirmanashari.jpg by using the encoding "ISO-8859-1", for example, the Base64 string is "**is**". Details of the conversion from the ASCII string to Base64 Encoded can be seen in Figure 2.

| Huruf | i | | | | | | | | s | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ASCII | 105 | | | | | | | | 115 | | | | | | | | | | | | | | | |
| BIT | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| index | 26 | | | | | | | | 23 | | | | | | | | 12 | | | | | | | |
| Base64 Encoded | a | | | | | | | | X | | | | | | | | M | | | | | | | |

Figure 2. Convert an ASCII String to Base64

Base64 Encoding Algorithm, which are:

1. Each character string “is” is converted into a decimal ASCII number
2. If a group of bits does not meet the 24-bit lengths, then padding is done by adding the character '=', or by adding 0 bits of 8 bits.
3. Decimal values will be converted into binary strings.
4. Binary strings will be grouped by 6 bits of data.
5. Convert 6 bits of data into decimal values.
6. Decimal values will be changed and converted based on Table 1 according to the index.

## 2. Embedding Method

The insertion process is the process of hiding images into MP3 audio media as a cover. This process will produce MP3 audio stego that has been inserted image. Before inserting a secret message into the MP3 audio storage media object, the first step that must be done is to understand the file structure of MP3 audio. In MP3 audio consists of several frames, in each frame consists of several headers and audio data. The internal structure of MP3 audio files and MP3 headers can be seen in Figure 3.

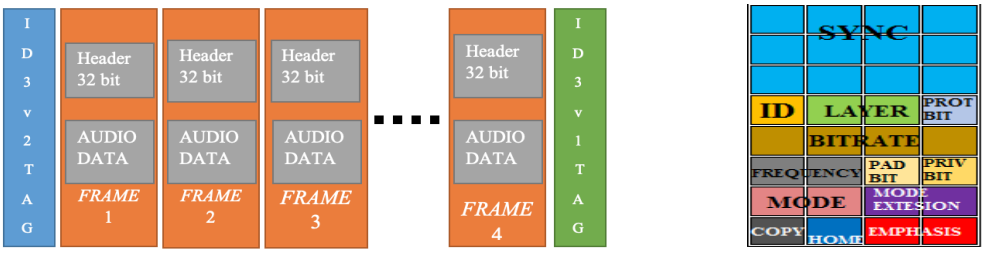



Figure 3. The internal structure of MP3 audio files and MP3 headers

In MP3 audio consists of several frames, in each frame consists of several headers and audio data. Message insertion is done in each frame of each header in the MP3 audio file. If it is known that the message to be inserted is in the form of ilhamfirmanashari.jpg image file, then the detailed steps of image insertion are performed, namely:

a. Change the ilhamfirmanashari.jpg file to the base64 string. The results can be seen in Figure 4.

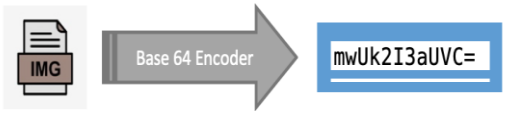



Figure 4. Convert the ilhamfirmanashari.jpg file to the Base64 string

b. Converts MP3 audio files to byte stream
c. Then proceed to the validation stage.
d. Check that MP3 audio has an id3v2 tag and an id3v1 tag. If there is an id3v2 tag, then check the size. if there is an id3v1 tag, then check the size.
e. If there is no id3v1 tag and id3v2 tag, then immediately read the MP3 frame header.
f. Count the number of MP3 frames, count the size of the MP3 frame, count the size of MARK. MARK is a marker word with the character "@ #!", is used as a sign that an MP3 file has a secret file in it.
g. Take measurements of the image capacity, using equation (1) as follows

a = ( size of MP3 frame – (32 bit or 4 *byte* (HEADER)
b = 2 *byte* CRC or VBR + check of size MODE CHANNEL
c = the number of frame MP3
d = if there is TAG, then (size of tag id3v2 + size of tag id3v1) + *mark size*)) / Shift Index

$$total = \frac{((a+b)*c)-d}{Shift\ Index} \qquad (1)$$

h. Enter the LBE parameter value, then save it to the shift index variable. If the parameter used is LBE + 1, then the Shift Index value used is 8. If the parameter used is LBE + 2, then the Shift Index value used is 7, and if LBE + 3 then the Shift Index value used is 6.

The details of the LBE + 1, LBE + 2, LBE + 3 algorithm are as follows

- LBE + 1 replaces each end bit with 1 message bit. For example, the image bit is 0111. Illustration of image insertion bits in the stego audio bit can be seen in Figure 5 and Figure 6.

10101111 00101001**0** 00110010 11100010

Figure 5. Bit audio cover LBE+1

10101110 00101001**1** 00110011 11100011

Figure 6. Bit audio stego (Bit yang terubah = 4 bit)

- LBE+2 replaces every last 2 bits with message bits. For example, the image bit is 0111. An illustration of the image insertion bit in the stego audio bit can be seen in Figure *7* and Figure *8*.

10101111 00101001**0** 00110**0**10 11100**0**10

Figure 7. Bit audio cover LBE+2

10101101 00101001**1** 00110**0**10 11100**0**10

Figure 8. Bit audio stego (Bit yang terubah = 2 bit)

- LBE+3 replaces every last 3 bits with message bits. Suppose the image bit is 0111, an illustration can be seen in Figure 9 and Figure 10.

10101111 00101001**0** 00110**0**10 11100**0**10

Figure 9. Bit audio cover LBE+3

10101011 00101011**0** 00110**0**10 11100**0**10

Figure 10. Bit audio stego (Bit yang terubah = 2 bit)

i. Insert each binary string from the image using the LBE insert algorithm based on the following pseudocode notation:

```
Tmp ← byte(((bytes[index_temp] and 0xFF) and 0xFE) or
((this.msg[msgIndex]  shiftright
this.toShift[(this.shiftIndex++) modulo this.toShift.length]
```

The results will be accommodated into the tmp variable in the converted byte stream form

j. Stream bytes will be converted to stego MP3 audio (audio that already contains secret images).

## B. Extraction and Decoding Method

An overview of extraction and decoding methods can be seen in Figure 11. Before the extraction process is performed, the user must first enter the MP3 Audio stego, which is an audio file that already contains a secret image. After that, the user must enter the LBE parameter number for the extraction process. Stego MP3 audio will be converted into a byte stream. Based on the number of LBE, the binary data from the image will be taken in the audio file by the shift of the inputted parameters, then the results will be converted into text decoder strings using Base64 Decoder. The text decoder string will be converted to a byte stream, then the byte stream will be converted and converted to an image and the last saved image file.

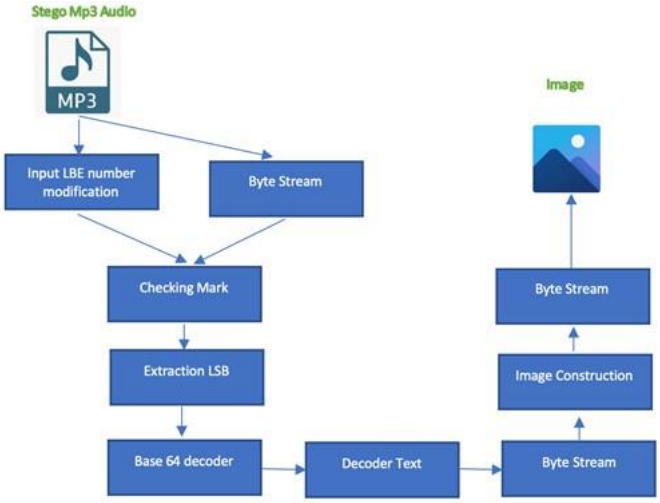



Figure 11. Overview of extraction and decoding

**1. Extraction Method**

The extraction process is the process of revealing the existence of a message or pulling a message that was previously inserted in an MP3 audio file. The flow of the image byte extraction process from MP3 audio, namely:

a. Change the Stego Audio MP3 into a byte stream, as shown in Figure 12.


Figure 12. Convert MP3 audio to byte stream

b. Inputs the appropriate LBE parameter values when inserting images
c. After that, mark validation is performed, if no byte stream mark is obtained, it is ensured that in the audio there is no secret image.
d. At the extracting method stage, the LBE bits will be retrieved based on the appropriate LBE value parameters. LBE extraction algorithm is based on the following pseudocode notation:

```
tmp_extract ←(byte) (tmp_extract or ((bytes[startOffset++]
shiftleft toShift[shiftIndex modulo toShift.length]) and
andByte[shiftIndex++ modulo toShift.length]));
```

If using the LBE + 1 parameter, when extraction the final 1 bit is taken as illustrated in Figure 13.

10101110 00101001 00110011 11100011

Figure 13. Position the bytes of the image on the audio cover with 4 bytes of stego audio (LBE+1)

The bits obtained from Figure 13, is **0111**. If using the LBE + 2 parameter, when extracting it will take the last 2 bits as illustrated in Figure 14.

10101101 00101001 00110010 11100010

Figure 14. Position the bytes of the image on the audio cover with 4 bytes of stego audio (LBE+2)

The bits obtained from Figure 14, is **0111** by looking at the character marker bits first. If using the LBE + 3 parameter, when extracting it will take the last 3 bits as illustrated in Figure 15.

10101111 00101001 00110010 11100010

Figure 15. Position the bytes of the image on the audio cover with 4 bytes of stego audio (LBE+3)

e. The extracted bits will be put back together and automatically sorted from the leftmost to the right bits to complete 8 bits, then converted to byte streams
f. Convert stream bytes to images and save them as an image file

**2. Base64 Decoding Method**

The decoding method with Base64 is the opposite of its encoding method. Suppose that the conversion from the image byte string after extracting from the audio file is known as "**aXM=**". Details of the conversion from Base64 Decoded to ASCII string can be seen in Figure 16.

| Base64 Decoded | a | | | | | | X | | | | | | M | | | | | | = | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| index | 26 | | | | | | 23 | | | | | | 12 | | | | | | 0 | | | | | |
| BIT | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ASCII | 105 | | | | | | 115 | | | | | | | | | | | | | | | | | |
| Huruf | i | | | | | | s | | | | | | | | | | | | | | | | | |

Figure 16. Convert Base64 String to ASCII

Base64 Decoding Algorithm

1. Change the base 64 string into decimal values based on the Base64 index table (Table 1)
2. Convert each decimal value into a binary string
3. Take 8 bits from each value in the binary string
4. Convert 8 bits into decimal values
5. Change the decimal value to ASCII string

## RESULTS AND DISCUSSION

The implementation of this research is presented as shown in Figure 17. Audio quality assessment can be done objectively or subjectively. Subjective assessment by playing and hearing audio directly or using an assessment using opinions with Mean Opinion Score (Rojali et al., 2017) and objective assessment by conducting tests that focus on aspects of imperceptibility, fidelity, recovery, robustness, and payload (Douglas et al., 2018). This study only uses an objective assessment (imperceptibility, fidelity, recovery, robustness, and payload).

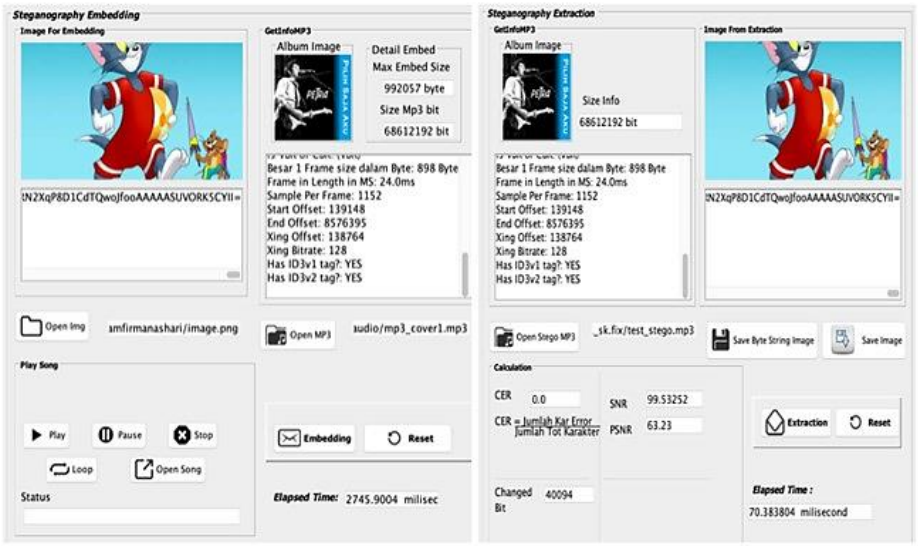



Figure 17. Display interface of embedding and extracting steganography applications

a. *Imperceptibility*, the image in the audio cannot be perceived and is not recognized as being in the audio file. The hearing senses cannot realize the difference in sound when an audio file has been inserted message. To measure the imperceptibility, the histogram and amplitude visualization of the original stego file and audio file are used. One example of the implementation of steganography by testing aspects of imperceptibility by using LBE + 1, LBE + 2, and LBE + 3 parameters is in Table 2 and Table 3 below.

Table 2. Testing the imperceptibility aspect with LBE+1, LBE+ 2, and LBE+3 before Embedding

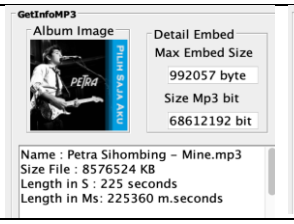
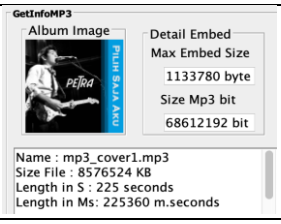
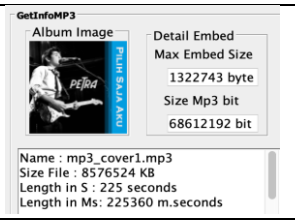
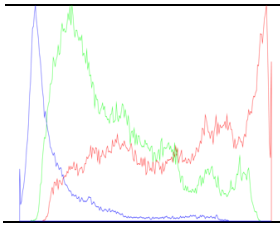
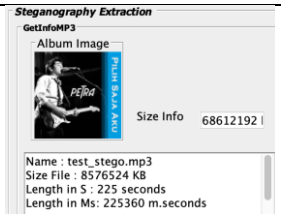
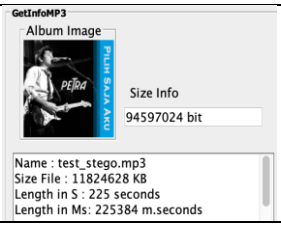
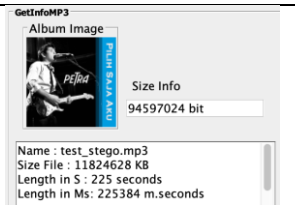
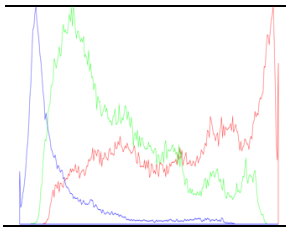
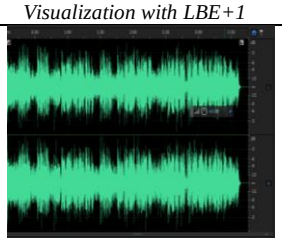
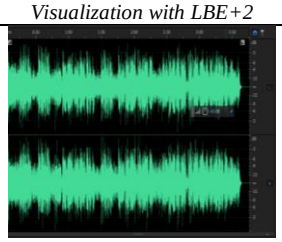
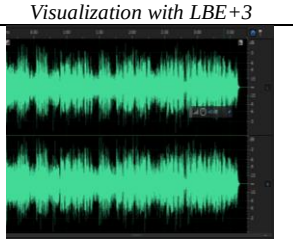
| *Image Before Embedding (file_embed.jpeg)* | *MP3 Information Before Embedding (file_1.mp3) with LBE + 1* | *MP3 Information Before Embedding (file_1.mp3) with LBE +2* | *MP3 Information Before Embedding (file_1.mp3) with LBE +3* |
|---|---|---|---|
|  |  |  |  |
| *Histogram (file_embed.jpeg)* | *Spectrum Frequency Visualization with LBE+1* | *Spectrum Frequency Visualization with LBE+2* | *Spectrum Frequency Visualization with LBE+3* |
|  |  |  |  |

Table 3. Testing the imperceptibility aspect with LBE+1, LBE+ 2, and LBE+3 after embedding

| *Image After Embedding (file_embed.jpeg)* | *MP3 Information After Embedding (file_1.mp3) with LBE + 1* | *MP3 Information After Embedding (file_1.mp3) with LBE +2* | *MP3 Information After Embedding (file_1.mp3) with LBE +3* |
|---|---|---|---|
|  |  |  |  |
| *Histogram (file_extract.jpeg)* | *Spectrum Frequency Visualization with LBE+1* | *Spectrum Frequency Visualization with LBE+2* | *Spectrum Frequency Visualization with LBE+3* |
|  |  |  |  |

From the tests conducted, it can be concluded that from the aspect of imperceptibility, the image file that is inserted and after extracting there is no significant difference because the quality and size of the image are the same, this is seen from the histogram on the image in three areas, namely red, green, and blue. From the audio cover file, the frequency spectrum also does not have a significant difference, and information from the MP3 audio before and after the message is inserted is no different.

b. *Fidelity*, the quality of stego file can be said to be good if the quality of the stego file not much different from the original file. To determine the quality of a stego audio file, it uses a measurement with the SNR parameter (Signal Noise Ratio) (Bansal et al., 2020) and PSNR (Datta et al., 2015)

$$SNR = \frac{NB-CB}{NB} \quad (2)$$

Note: NB is the total number of MP3 bits, CB is the modified bit due to LBE

$$PSNR = 10\ log_{10} \frac{\sum_{i=1}^{m} {x_1}^2}{\sum_{i=1}^{m} {x_1}^2 - \sum_{i=1}^{m} {x_0}^2} \quad (3)$$

Note : $x_n$ represents the original audio file or cover and $x_0$ represents the audio MP3 steganography. This test using test data (test_gambar1.jpg, test_gambar2.jpg, and test_gambar3.jpg), and the results will be presented in Table 4.

Table 4. The fidelity aspect test results use LBE + 1, LBE + 2, LBE + 3

| *No* | *Steganography File* | *Audio and Image Size (byte)* | *Embed Method* | *Image Size (byte)* | *Bit Changed Because LBE* | *SNR* | *PSNR* |
|---|---|---|---|---|---|---|---|
| 1 | MP3_cover1.mp3 | 8.576.524 | LBE+1 | 13.362 | 8882 | 99.89 % | 66.47 |
| 2 | MP3_cover2.mp3 | 9.249.508 | LBE+1 | 32.761 | 21738 | 99.81 % | 57.78 |
| 3 | MP3_cover3.mp3 | 11.824.628 | LBE+1 | 36.170 | 23994 | 99.74 % | 56.21 |
| 4 | MP3_cover1.mp3 | 8.576.524 | LBE+2 | 13.362 | 8857 | 99.89 % | 63.45 |
| 5 | MP3_cover2.mp3 | 9.249.508 | LBE+2 | 32.761 | 2179) | 99.81 % | 53.10 |
| 6 | MP3_cover3.mp3 | 11.824.628 | LBE+2 | 36.170 | 24065 | 99.73 % | 51.45 |
| 7 | MP3_cover1.mp3 | 8.576.524 | LBE+3 | 13.362 | 8877 | 99.89 % | 61.43 |
| 8 | MP3_cover2.mp3 | 9.249.508 | LBE+3 | 32.761 | 21705 | 99.81 % | 50.06 |
| 9 | MP3_cover3.mp3 | 11.824.628 | LBE+3 | 36.170 | 24115 | 99.73 % | 49.54 |

From the tests conducted to measure aspects of fidelity by comparing the three methods of embedding LBE + 1, LBE + 2, and LBE + 3, it can be concluded that the best embedding method is LBE + 1 seen from the PSNR and SNR values greater than the others. However, when the audio is listened to, LBE + 2 is arguably the most optimal method for SNR and PSNR when viewed from the number of messages inserted.

c. *Recovery*, ensure that the byte of the image string before being inserted is the same as the byte of the image string after extracting, meaning that the image file remains intact. To measure the recovery aspect, CER testing is used, the test results are shown in Table 5.

$$CER = \frac{NE}{SB} \quad (4)$$

Note: NE is the number of bytes of error image string, SB is the sum of all bytes of the image string

Table 5. Recovery aspects testing results

| *No* | *Cover File* | *Embed Method* | *Image Size Using Base64 Encoder (byte)* | *Embedding Status* | *Extracting Status* | *Image Size After Base64 Decoder (byte)* | *CER Value* |
|---|---|---|---|---|---|---|---|
| 1 | MP3_cover1.mp3 | LBE+1 | 13.362 | Success | Success | 13.362 | 0 |
| 2 | MP3_cover2.mp3 | LBE+1 | 32.761 | Success | Success | 32.761 | 0 |
| 3 | MP3_cover3.mp3 | LBE+1 | 36.170 | Success | Success | 36.170 | 0 |
| 4 | MP3_cover1.mp3 | LBE+2 | 13.362 | Success | Success | 13.362 | 0 |
| 5 | MP3_cover2.mp3 | LBE+2 | 32.761 | Success | Success | 32.761 | 0 |
| 6 | MP3_cover3.mp3 | LBE+2 | 36.170 | Success | Success | 36.170 | 0 |
| 7 | MP3_cover1.mp3 | LBE+3 | 13.362 | Success | Success | 13.362 | 0 |
| 8 | MP3_cover2.mp3 | LBE+3 | 32.761 | Success | Success | 32.761 | 0 |
| 9 | MP3_cover3.mp3 | LBE+3 | 36.170 | Success | Success | 36.170 | 0 |

It can be concluded from testing with data using three steganography MP3 files with LBE + 1, LBE + 2, and LBE + 3 embedding methods, then the CER value = 0 is obtained, meaning that there is no difference in the number of bytes of the inserted image string and the extracted image. From testing on the recovery aspect, the success rate is 100%. That is, this method supports the recovery aspect of data security.

d. *Robustness*, testing is done by manipulating MP3 signals (Bitrate and Channel Mode). This test uses MP3 Quality software. The results of testing the signal manipulation of the bit rate can be seen in Table 6, the results of testing of channel mode manipulation can be seen in Table 7.

Table 6. MP3 cover manipulation test results on bitrate

| *No* | *MP3 Cover* | *Initial Bit Rate (kbps)* | *Max Capacity Before Manipulation (Byte)* | *Final Bit Rate (kbps)* | *Max Capacity After Manipulation (Byte)* | *Extraction Status* |
|---|---|---|---|---|---|---|
| 1 | MP3_stego1.mp3 | 300 | 992057 | 250 | 779609 | Failed |
| 2 | MP3_stego2.mp3 | 224 | 1370228 | 310 | 1858609 | Failed |
| 3 | MP3_stego3.mp3 | 256 | 885361 | 190 | 631635 | Failed |

Table 7. MP3 cover manipulation test results on modus channel

| *No* | *MP3 Cover* | *Initial Channel Mode* | *Max Capacity Before Manipulation (Byte)* | *Final Channel Mode* | *Max Capacity After Manipulation (Byte)* | *Extraction Status* |
|---|---|---|---|---|---|---|
| 1 | MP3_stego1.mp3 | Joint Stereo | 992057 | Dual Channel | 992057 | Failed |
| 2 | MP3_stego2.mp3 | Joint Stereo | 1370228 | Mono | 1370228 | Failed |
| 3 | MP3_stego3.mp3 | Stereo | 885361 | Stereo | 885361 | Failed |

Based on testing the robustness aspect, it can be concluded that the stego audio file is not resistant to attacks by manipulation of the Bit Rate and Channel Mode, because picture messages cannot be extracted from the stego audio file.

e. *The payload* is related to message capabilities. In steganography, this aspect is quite important, because as much as possible the message must be able to be inserted without affecting the quality of the cover file. Testing is done by comparing parameters with LBE + 1, LBE + 2, and LBE + 3. Payload aspect test results can be seen in 8.

$$increased\ capacity = (LBE_n - LBE_{n-1}) * 100\% \quad (5)$$

Table 8. Payload aspects with LBE + 1, LBE + 2, LBE + 3

| *NO* | *Audio File* | *Image File* | *Audio Size (byte)* | *Image Size (byte)* | *Embedding Method* | *Maximum size of images that can be inserted (bye)* | *Embed Time (ms)* | *Extract Time (ms)* |
|---|---|---|---|---|---|---|---|---|
| **1** | MP3uji1.mp3 | uji_gambar1.jpg | 8.576.524 | 13.362 | LBE+1 | 992057 | 770.23 | 26.99 |
| **2** | MP3uji2.mp3 | uji_gambar2.jpg | 9.249.508 | 32.761 | LBE+1 | 1370228 | 1733.44 | 27.95 |
| **3** | MP3uji3.mp3 | uji_gambar3.jpg | 11.824.628 | 36.170 | LBE+1 | 885361 | 1711.04 | 39.05 |
| **4** | MP3uji1.mp3 | uji_gambar1.jpg | 8.576.524 | 13.362 | LBE+2 | 1133780 | 773.664 | 52.95 |
| **5** | MP3uji2.mp3 | uji_gambar2.jpg | 9.249.508 | 32.761 | LBE+2 | 1565975 | 1744.7853 | 54.21 |
| **6** | MP3uji3.mp3 | uji_gambar3.jpg | 11.824.628 | 36.170 | LBE+2 | 1011842 | 1723.2172 | 62.99 |
| **7** | MP3uji1.mp3 | uji_gambar1.jpg | 8.576.524 | 13.362 | LBE+3 | 1322743 | 843.4274 | 55.45 |
| **8** | MP3uji2.mp3 | uji_gambar2.jpg | 9.249.508 | 32.761 | LBE+3 | 1826971 | 1812.7758 | 59.76 |
| **9** | MP3uji3.mp3 | uji_gambar3.jpg | 11.824.628 | 36.170 | LBE+3 | 1180482 | 1834.5253 | 66.60 |

## CONCLUSIONS AND RECOMMENDATIONS

The test is carried out based on an objective assessment by paying attention to aspects of imperceptibility, recovery, fidelity, robustness, and payload. The results of imperceptibility testing by looking at the image histogram curve of the image before and after insertion with the same results and there is no difference, means that the image quality before and after insertion is the same. Audio files before and after insertion by looking at the comparison between the methods LBE + 1, LBE + 2, LBE + 3 do not look very significant differences. Based on testing on the fidelity aspect, we can see optimal PSNR and SNR values on LBE + 2, by looking at the capacity of the inserted picture message and image quality. PSNR LBE + 2 values range from 50-60 dB with SNR different about 0.01% from LBE + 1. The test results based on the aspect of robustness show that the proposed method is not resistant to attacks with changes in bit rate and channel mode. Testing of the recovery aspect gives the result that the proposed method already supports the 100% recovery aspect, meaning that the quality and size of the image (.jpg, png, BMP, and. gif). Before insertion and after extraction will be the same. And finally, the test results on the payload aspect, there is an increase in message capacity with LBE + 2 around 12.5% of LBE + 1, and using LBE +

3 will increase the maximum size around 25% of LBE + 1 and about 14% of LBE + 2. The insertion and extraction time for LBE + 3 is slower compared to the others. This research can be developed further by using a more secure encryption algorithm, for example using RC4 or AES. In addition, it can use several variations of other message objects such as audio and video.

## REFERENCE


Ashari, I. F. (2018). Graph Steganography Based On Multimedia Cover To Improve Security and Capacity. *2018 International Conference on Applied Information Technology and Innovation (ICAITI), April 2019*, 194–201. doi: 10.1109/ICAITI.2018.8686741

Bansal K., Agrawal A., and Bansal N.. (2020) "A Survey on Steganography using Least Significant bit (LSB) Embedding Approach," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), pp. 64-69, doi: 10.1109/ICOEI48184.2020.9142896.

Chakraborty, R. (2019). Audio Steganography- A Review. *International Journal of Trend in Research and Development, Volume 6(3), ISSN: 2394-9333*, *6*(July).

Choi, J. R., Straubhaar, J., Skouras, M., Park, S., Santillana, M., & Strover, S. (2020). Techno-capital: Theorizing media and information literacy through information technology capabilities. New Media & Society. https://doi.org/10.1177/1461444820925800

Chowdhury, R., Bhattacharyya, D., Bandyopadhyay, S. K., & Kim, T. H. (2016). A view on LSB based audio steganography. *International Journal of Securitdy and Its Applications*, *10*(2), 51–62. https://doi.org/10.14257/ijsia.2016.10.2.05

Datta, B., Pal, P., & Bandyopadhyay, S. K. (2015). Robust multi layer audio steganography. *12th IEEE International Conference Electronics, Energy, Environment, Communication, Computer, Control: (E3-C3), INDICON 2015*, 1–6. https://doi.org/10.1109/INDICON.2015.7443342

Denegri-knott, J. (2015). MP3. *Taylor and Francis Group*, *3866*(September), 397–401. https://doi.org/10.1080/10253866.2015.1048962

Douglas, M., Bailey, K., Leeney, M., & Curran, K. (2018). An overview of steganography techniques applied to the protection of biometric data. *Multimedia Tools and Applications*, *77*(13), 17333–17373. https://doi.org/10.1007/s11042-017-5308-3

Dutta, H., Das, R. K., Nandi, S., & Prasanna, S. R. M. (2019). An Overview of Digital Audio Steganography. *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, *0*(0), 1–19. https://doi.org/10.1080/02564602.2019.1699454

Gaikwad, T., Patil, M., Sagvekar, P. V., & Racha, P. D. (2015). Network Security : A Study Using Cryptography. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, *3*(II), 392–398.

Gurgaon, I. (2017). International Journal of Computer Science and Mobile Computing An Overview Of Network Security-Its Types And Techniques Nishu SethI Types of Security: Fig: Types of Security. *International Journal of Computer Science and Mobile Computing*, *6*(5), 235–237. www.ijcsmc.com

Hussain, M., Wahab, A. W. A., Idris, Y. I. Bin, Ho, A. T. S., & Jung, K. H. (2018). Image steganography in spatial domain: A survey. *Signal Processing: Image Communication*, *65*, 46–66. https://doi.org/10.1016/j.image.2018.03.012

Jasvir Singh, Gaganjot Kaur, & Manveer Kaur Garcha. (2015). Review of Spatial and Frequency Domain Steganographic Approaches. *International Journal of Engineering Research And*, *4*(06), 1122–1124. https://doi.org/10.17577/ijertv4is061082

Jenkins, B., & Laurier, W. (2013). MP3: THE MEANING OF A FORMAT. *Canadian Journal of Communication*, *30*(2), 1–2.

Kadhim, B. A. (2017). High Security Steganography Model Based on DWT, DCT and RSA. *Journal of Engineering and Applied Science*, *10*, 8875–8881.

Malviya, S., Saxena, M., & Khare, A. (2012). Audio Steganography by Different Methods. *International Journal of Emerging Technology and Advanced Engineering*, *2*(7), 371--375.

http://www.ijetae.com/files/Volume2Issue7/IJETAE_0712_64.pdf

Mulyono, I. U. W., Susanto, A., Anggraeny, T., & Sari, C. A. (2019). Encryption of Text Message on Audio Steganography Using Combination Vigenere Cipher and LSB (Least Significant Bit). *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, *4*(1), 63–74. https://doi.org/10.22219/kinetik.v4i1.701

Parthasarathi, M., Shreekala, T., Nadu, T., & Nadu, T. (2017). Secured Data Hiding in Audio Files Using Audio Steganography Algorithm. *International Journal of Pure and Applied Mathematics*, *116*(21), 619–627.

Payam H, Samira G and Masoud A. (2017). Impact of Information Technology on Lifestyle: A Literature Review and Classification Int. Journal of Virtual Communities and Social Networking (IJVCSN) 9 pp 1-23. doi: 10.1088/1757-899X/1073/1/012056

Rehman, A., Saba, T., Mahmood, T., Mehmood, Z., Shah, M., & Anjum, A. (2018). Data hiding technique in steganography for information security using number theory. *Journal of Information Science*, *45*(6), 1–12. https://doi.org/10.1177/0165551518816303

Rojali, Salman, A. G., & George, G. (2017). Website-based PNG image steganography using the modified Vigenere Cipher, least significant bit, and dictionary based compression methods. *AIP Conference Proceedings*, *1867*, 1–11. https://doi.org/10.1063/1.4994462

Rokhman, N., & Maharanti, J. (2011). Deteksi Steganografi Berbasis Least Significant Bit (LSB) Dengan Menggunakan Analisis Statistik. *IJCSS (Indonesia Journal of Computing and Cybernatics Systems)*, *5*(1), 57–62. https://doi.org/10.1111/een.12476

Roy, R., & Changder, S. (2016). Quality evaluation of image steganography techniques: A heuristics based approach. *International Journal of Security and Its Applications*, *10*(4), 179–196. https://doi.org/10.14257/ijsia.2016.10.4.18

Sagarmay Deb. (2014). Information Technology, Its Impact on Society and Its Future. *Advances in Computing*, *4*(1), 25–29. https://doi.org/10.5923/j.ac.20140401.07

Sairam, T. ., & Bhoopaty, B. (2018). Comparative Analysis of Spatial and Frequency Domain Based Data Hiding 1. *International Journal of Pure and Applied Mathematics*, *119*(15), 2353–2359.

Setiaji, H. C. F., Tjondronegoro, S., & Hidayat, B. (2015). Audio Steganography using Modified Enhanced Least Significant Bit in 802.11n. *Journal of Measurements, Electronics, Communications, and Systems*, *2*(3), 7465–7474. https://doi.org/10.25124/jmecs.v1i1.1479

Taha, M. S., Mohd Rahim, M. S., Lafta, S. A., Hashim, M. M., & Alzuabidi, H. M. (2019). Combination of Steganography and Cryptography: A short Survey. *IOP Conference Series: Materials Science and Engineering*, *518*(5), 1–13. https://doi.org/10.1088/1757-899X/518/5/052003