

Research Paper

The Evaluation LSB Steganography on Image File Using 3DES and MD5 Key

*Ilham Firman Ashari*¹, *Eko Dwi Nugroho*¹, *Dodi Devrian Andrianto*¹, *M. Asyroful Nur Maulana Yusuf*^d,
*Makruf Alkarkhi*¹

¹Program Studi Teknik Informatika, Fakultas Teknologi Industri, Institut Teknologi Sumatera, Terusan Ryacudu Street, Lampung Selatan, 35365, Indonesia

ARTICLE INFORMATION

Received: June 14th, 2022
 Revised: March 14th, 2024
 Available online: March 31st, 2024

KEYWORDS

Steganography, LSB, 3DES, MD5,
 Cryptography

CORRESPONDENCE

Phone: +62 (0751) 12345678
 E-mail: firman.ashari@if.itera.ac.id

A B S T R A C T

Information security is paramount for individuals, companies, and governments. Threats to data confidentiality are increasingly complex, demanding strong protection. Therefore, cryptography and steganography play pivotal roles. This study proposes the utilization of LSB (Least Significant Bit) steganography on image files employing the 3DES (Triple Data Encryption Standard) algorithm. The aim is to facilitate secure transmission and reception of data, including confidential messages, within digital information media. The research methodology involves implementing 3DES + LSB using Image Citra and innovating 3DES + MD5 Hash in .txt files. The results and discussions described include, among others, Pseudocode, Cryptographic Testing, and Steganography Testing. Based on the results of program analysis and testing, it can be concluded that the more messages that are inserted in the image, the more pixel differences there are in the stego image. The more colors in the image to which the message will be inserted, the more pixel differences in the stego image will be. The images that stego objects can present are only images with .png and .jpeg extensions. Testing from the fidelity aspect, the average PSNR obtained is 66,365, meaning that the stego image quality is very good. Testing from the recovery aspect, from 4 tested stego images, showed that messages can be extracted again. Testing of the robustness spec using two attack techniques, namely rotation, and robustness, shows that the message cannot be extracted from the image. Testers of the computation time, from testing 1-1000 characters, show the average time required for computation is about 0.798 seconds.

INTRODUCTION

In the era of increasingly advanced technology, as it is today, the use of digital information media to communicate, receive, and send important data has become a necessity [1]. Confidentiality and data security are essential in digital information media or information systems today; taking data or information to other parties can harm those with data or information [2]. Confidential data or information is precious for the owner of the data or information, coupled with a high-speed internet network and not guaranteed security [3][4][5]. So, efforts are needed to deal with these problems, and one of these efforts is to apply cryptographic techniques [6].

Cryptography is a study that studies techniques or methods to process confidential data or information, aiming to secure confidential data or information by carrying out the encryption and decryption process on data or information to be secured [7]. Encryption is a process in which data or information is converted into another form that is difficult to understand. At the same time, decryption is a process where data or information converted into another form is returned to its original or original form [8].

<https://doi.org/10.25077/jitce.8.01.8-18.2024>

Decryption is the process of returning a ciphertext message to plaintext or the original message [9]. One of the data or information that is kept secret is the image, while the technique for securing the image is Steganography [10][11]. To improve message security, cryptography and steganography techniques can be combined [12][13][14].

Steganography is a technique found in cryptographic studies that has been used to hide information in a medium such as images, video, and sound [15]. An example of the application of steganography techniques is if there is an image that is inserted information, the information will not be visible in the image that has been inserted, but if the image is extracted using special software steganography, then the inserted information will be visible [16].

A method contained in image media steganography is called Least Significant Bit (LSB). The Least Significant Bit (LSB) method is a method used to insert information that goes through the process of replacing the 8th, 16th, and 24th bits in a binary image of an image file with a binary image of confidential information to be secured [17].

[Attribution-NonCommercial 4.0 International](https://creativecommons.org/licenses/by-nc/4.0/). Some rights reserved

Triple Data Encryption Standard (3DES) is a development of the previous encryption system, namely the Data Encryption Standard (DES); the encryption system on 3DES has three layers and is certainly more secure than DES. At the time the DES algorithm was created, a 56-bit key size was considered sufficient [18]. However, with the development of technology, the ability to compute attacks also increases, so 3DES is made by providing a simple method and increasing the size of the DES key to prevent computing attacks by not using a new block cipher or block cipher [19].

When compared with other block cipher algorithms, namely DES, Blowfish, and Twofish, the 3DES algorithm is better in terms of security. In this study, we propose the implementation of LSB (Least Significant Bit) steganography on image files using the 3DES (Triple Data Encryption Standard) algorithm, with the hope of helping digital information media users send and receive data or information and secret messages safely.

METHOD

In this study, the implementation of LSB in image files uses the 3DES algorithm using the Python Google Collab programming language, with testing using an Apple M1 Pro processor and 512 Gb SSD NVMe internal storage supported by 16Gb RAM with the Mac OS Monterey operating system. This research is divided into two stages, namely the encoding and embedding stages, which are the image insertion and extraction stages, and the decoding stages, which are the image extraction stages.

Encoding and Embedding Methods

An overview of the working process scheme of the encoding method can be seen in the Figure 1.

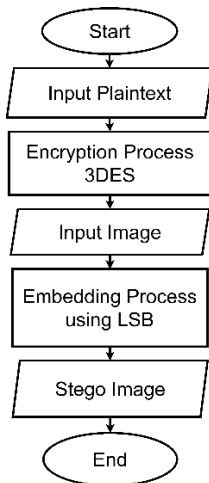


Figure 1. Encryption and Embedding process flow

In the early stages of encoding, the user is asked to input plaintext, which will be encrypted and inserted in the image; the 3DES encryption process is carried out by adding a predefined key consisting of 24 letters. After that, the program will display the results of the 3DES encryption and the inputted plaintext. The user is prompted to input an image that will be used as a medium for inserting messages. The LSB encryption process involves inserting 3DES encryption results from the ciphertext. The user will input the name of the stego file with the desired extension,

and the program will perform and generate the process of the stego image that has been inserted text from the encryption result. 3DES. The image of the insertion of a text message that has been encrypted by 3DES is called a stego image.

Furthermore, regarding the innovation of Triple Data Encryption Standard (3DES) cryptography by adding the MD5 Hash function to the .txt file. The flowchart of 3DES + MD5 Encryption in a .txt file is shown in Figure 2.

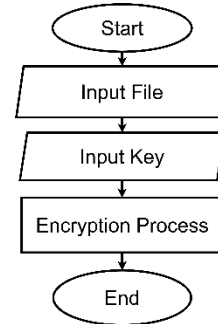


Figure 2. Encryption Process Flow using 3DES and MD5

At this stage, the researcher innovates on the implementation of 3DES cryptography by adding MD5 to the .txt file. The user is asked to input the .txt file using the directory path; the user inputs a key or key, and the program will perform 3DES + MD5 encryption, and the contents of the .txt file will changed to the result of the 3DES+MD5 encryption process. The Triple Data Encryption Standard (3DES) encryption process can be seen in Figure 3.

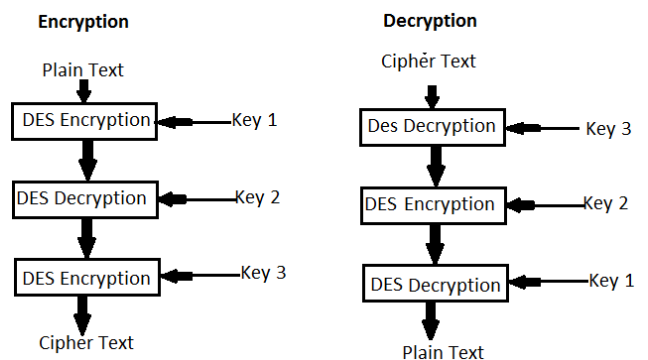


Figure 3. Overview of Encryption With 3DES

The entered plaintext will be substituted in the 64-bit DES process. Then enter the second stage where enter 2DES with the same 32-bit key. These two parts are included in 16 DES cycles where one DES cycle is a Feistel network model. Three repetitions are essential to prevent meet-in-the-middle-attack. In simple form 3DES (EEE mode) can be written as follows:

$$C = Ek_3(Ek_2(Ek_1(P)))$$

$$P = Dk_1(Dk_2(Dk_3(C)))$$

The first version of 3DES uses 2 keys (EDE mode):

$$C = Ek_1(Ek_2(Ek_1(P)))$$

$$D = Dk_1(Dk_2(Dk_1(C)))$$

The second version of 3DES uses 3 keys (EDE mode):

$$C = Ek_3(Dk_2(Ek_1(P)))$$

$$D = Dk_1(Ek_2(Dk_3(C)))$$

Description of Variables:

- C = Ciphertext
- P = Plaintext
- EK₁ = Encryption 1
- EK₂ = Encryption 2
- EK₃ = Encryption 3
- DK₁ = Decryption 1
- DK₂ = Decryption 2
- DK₃ = Decryption 3

Base64 Encoding Method

The Base64 algorithm functions to convert the bytes of encryption or decryption of binary data or text into ASCII characters. The Base64 encoding consists of 26 uppercase letters, 26 lowercase letters, 10 numbers, + and / for newlines. In the encoding and decoding process, the Base64 index table is treated as seen in the Table 1.

Table 1. Table 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		

Suppose a byte string is to be inputted in an image file using 3DES encoding with the sentence "IF". The detail conversion from ASCII string to Base64 Encode

Base64 encoding algorithm, which is as follows:

1. Each character string is converted to a decimal ASCII number.
2. If the bit group does not consist of 24-bits, then the padding is done by adding 0-bit characters.
3. The decimal value is converted to a binary string.
4. Binary strings will be grouped based on 6-bit data.
5. Convert 6-bit data to decimal values.
6. The decimal value will be changed and converted based on the Base64 table.

Letter	I	F	
ASCII	73	70	
BIT	0 1 0 0 1 0 0 1	0 1 0 0 0 1 1 0	0 0 0 0 0 0 0 0
Index	18	20	24
Base64	S	U	Y =

Figure 4. Encoding ASCII to Base64

Insertion Method

The embedding process is the process of hiding messages in the image file from the 3DES encryption using the Least Significant Bit (LSB) method. This method is widely used because it is very simple and easy to implement, and the media that is often used for implementing steganography is images. The advantage of the LSB method is that it can input messages in images that are not much different from the original image. The illustration of the insertion and encoding image can be seen in Figure 5.

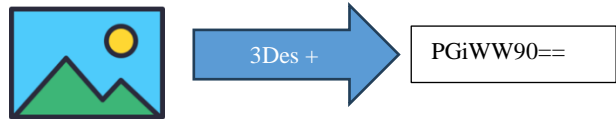
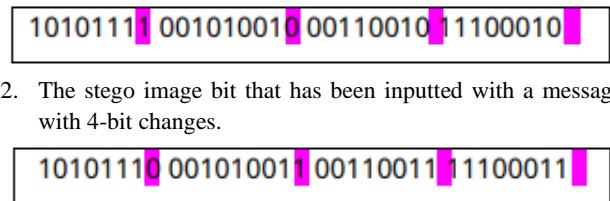


Figure 5. Overview of Insertion and Encoding

The LSB process for inputting messages to the image where the illustration of the message insertion bit in the image bit:

1. The process of determining the last 1 bit in the RGB image.
2. The stego image bit that has been inputted with a message with 4-bit changes.



Extraction and Decoding Method

An overview of the working process scheme of the extraction and decoding method can be seen in Figure 6. At the data processing stage on the stego image, the user is asked to input the name of the stego image with an extension. The extraction process on the stego image uses the Least Significant Bit (LSB) method to view the message on the stego image; the program displays random text messages on the stego image, and the text message decryption process random using Triple Data Encryption Standard (3DES), and the program will display plaintext results.

Furthermore, after the 3DES + MD5 encryption process is complete, the following is a flowchart or flowchart on 3DES + MD5 Decryption in the .txt file. The illustration of the decryption process can be seen in Figure 7.

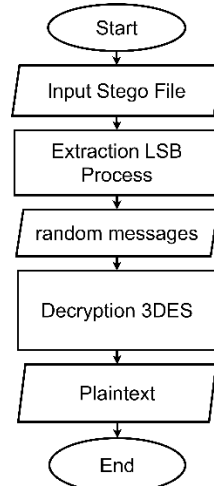


Figure 6. Overview of Extraction and Decryption

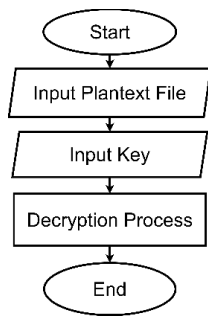


Figure 7. Decryption Process Flow Using 3DES and MD5

At this stage, the process of decrypting the .txt file is performed using the 3DES+MD5 cryptographic algorithm. The user is asked to input the .txt file with the directory path, and the user is asked to input a key or key. After that, it goes through a branching process where if key A is the same as key B, then the process 3DES+MD5 decryption will be carried out, and if key A is not the same as key B, then it will not go through the 3DES+MD5 decryption process. After the 3DES+MD5 decryption process has been carried out, the .txt file will again contain the original text message.

Extraction Method

The extraction process is the process of revealing the existence of the message in the stego image. The flow of the message extraction process from the image can be seen in figure 8.

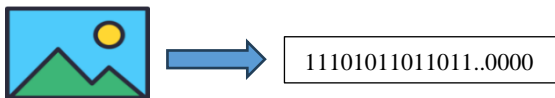
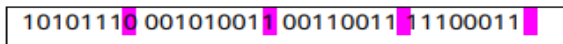


Figure 8. Convert image to binary

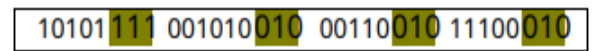
1. LSB extraction in the image file is taken based on the appropriate value parameter, if the last 1 bit is taken, it is as shown below:



2. The LSB extraction in the image file is taken from the last 2-bit, so it looks like the image below:



3. LSB extraction in the image file is taken based on the last 3 bits, as shown below:



Base64 Decoding Method

The Base64 decoding method is the opposite of the Base64 encoding method. For example, converting from a byte string to Base64 decode into an ASCII string, it can be seen as in the illustration below (Figure 9).

Base64	S	U	Y	=
Index	18	20	24	0
BIT	0 1 0 0 1 0	0 1 0 1 0 0	0 1 1 0 0 0	0 0 0 0 0 0
ASCII	73	70		
Letter	I	F		

Figure 9. Decoding Base64 to ASCII

Base64 decoding algorithm, as follows:

1. The program converts Base64 strings to decimal values based on the Base64 index table
2. Converts each decimal value to a binary string
3. The 8 bits of each value become a binary string and are converted to a decimal value
4. Converts decimal values to ASCII strings

RESULTS AND DISCUSSION

In this research, two innovations were carried out using the Python programming language. The first innovation is to implement the 3DES + LSB cryptographic algorithm on image files, and the second innovation implements the 3DES + MD5 cryptographic algorithm on .txt files.

The first innovation regarding the implementation of cryptography and steganography on 3DES + LSB using the key = This is 24 byte key!!, and the main function from encryption and decryption can be seen in Figure 10.

```

Procedure GenerateMD5Key(key):
  HashedKey = CalculateMD5Hash(key)
  Return HashedKey

Procedure EncryptUsing3DESWithHash(message, key):
  HashedKey = GenerateMD5Key(key)
  Initialize 3DES encryption algorithm with HashedKey
  Cipher = Initialize 3DES cipher in ECB mode
  EncryptedMessage = Encrypt(message) using Cipher
  Return EncryptedMessage

Procedure DecryptUsing3DESWithHash(encryptedMessage, key):
  HashedKey = GenerateMD5Key(key)
  Initialize 3DES decryption algorithm with HashedKey
  Cipher = Initialize 3DES cipher in ECB mode
  DecryptedMessage = Decrypt(encryptedMessage) using Cipher
  Return DecryptedMessage

Main:
  kunci_3des = b'This is 24 byte key!!' # 24-byte key for 3DES encryption
  Message = "Hello, world!" # Message to be encrypted
  Print "Original Message:", Message

  EncryptedMessage = EncryptUsing3DESWithHash(Message, kunci_3des)
  Print "Encrypted Message:", EncryptedMessage

  DecryptedMessage = DecryptUsing3DESWithHash(EncryptedMessage, kunci_3des)
  Print "Decrypted Message:", DecryptedMessage
  
```

Figure 10. Pseudocode for Main Function from Encryption and Decryption

```

function embed_message_into_image(image_path, message):
img = open_image(image_path)
pixels = get_pixel_data(img)
message_bin = convert_message_to_binary(message)
index = 0
for pixel in pixels:
    red, green, blue = split_pixel(pixel)

    if index < length(message_bin):
        red = alter_least_significant_bit(red, message_bin[index])
        index += 1
    if index < length(message_bin):
        green = alter_least_significant_bit(green, message_bin[index])
        index += 1
    if index < length(message_bin):
        blue = alter_least_significant_bit(blue, message_bin[index])
        index += 1

    update_pixel(img, (index // 3 % img.width, index // 3 // img.width), (red, green, blue))

save_image(img, 'encoded_image.png')

```

Figure 11. Pseudocode for LSB Encoding Function

```

function extract_message_from_image(image_path):
img = open_image(image_path)
pixels = get_pixel_data(img)

extracted_message = ""
index = 0
while True:
    red, green, blue = split_pixel(pixels[index])

    extracted_message += str(red & 1)
    extracted_message += str(green & 1)
    extracted_message += str(blue & 1)

    index += 1

    if length(extracted_message) % 8 == 0 and extracted_message ends with '00000000':
        break

message = ""
for i from 0 to length(extracted_message) by 8:
    message += char_from_binary(extracted_message[i:i + 8])

return message

```

Figures 12. Pseudocode for LSB Decoding Function

The encryption process in steganography uses the LSB (Least Significant Bit) as in the pseudocode, which can be seen in Figure 11.

The decryption process in steganography uses the LSB (Least Significant Bit) algorithm by paying attention to the RGB (Red,

Green, Blue) pixels with $n = 3$ and RGBA (Red, Green, Blue, Alpha) with $n = 4$. as in the pseudocode can be seen in Figure 12.

The second innovation regarding the implementation of cryptography with the 3DES + MD5 algorithm, which uses the DES3 and MD5 keywords in Python programming as the pseudocode, can be seen in Figure 13.

```

function main():
message = "pesan"
key = "kunci"

encrypted_message = encrypt_3des_with_hash(message.encode(), key)
input_image = "gambar_input.png"

embed_message_into_image(input_image, encrypted_message.decode())

hidden_message = extract_message_from_image("encoded_image.png")
decrypted_message = decrypt_3des_with_hash(hidden_message.encode(), key)

print("Pesan Asli:", message)
print("Pesan Setelah Proses Dekripsi:", decrypted_message.decode())

main()

```

Figures 13. Variables used for encryption with 3DES and MD5

Cryptographic Testing

In this program, the researcher tested the data by comparing between 3DES without using the MD5 Hash function and the 3DES program using the MD5 Hash function. The comparison is carried out with the scenario of entering three different messages and using the same key because the key for the 3DES process without the MD5 function cannot be entered by the user. By making this comparison, the researcher obtained the data from the test. The data used in this research can be seen in Table 2. The results of the comparison of the 3 DES and, 3DES + MD5 algorithms can be seen in Table 3.

Steganography Testing

In this program, researchers tested the data by analyzing three scenarios. In the first scenario, the researcher makes a comparison

using the same object and different messages. Then, the second scenario performs a comparison using different objects and the same message. The latter performs comparisons using different objects with different messages as well.

From the three test results that have been carried out, there is a change in the pixels between the image before the word is inserted and after it is inserted. However, at a glance, the two images do not look different. To find out how many different pixels there are, the researcher uses a website to compare images. The website is safeimagekit.com. By making this comparison, the researcher obtained the data from the test. These data can be seen in Table 5. The results of the data, evidenced by the placement of different pixels, for more details, see the Table 6.

Table 2. Text Message, Character Length and Key

Text Message	The Length of Characters	Key
prodi teknik informatika	24	abcdefghijklmnopqrstuvw
matakuliah kriptografi	22	abcdefghijklmnopqrstuvw
ngoding mantaf	14	abcdefghijklmnopqrstuvw

Table 3. Results of Comparison of DES and 3DES+MD5 Algorithms

The result of 3DES	The Length of Characters	The Result of 3DES + MD5	The Length of Characters
2OABMdgMjM0w/VfvkFbDbid23povmjQ8	32	j}hW~G_	24
FPcVeGgneWibhaNBP3ZSESU7iYY4iQ==	32	wq]hB~G	22
Xu+Sz0sXh+znLwIkyhQ=	20	t8 2R	14

Table 4. Computation time with 3DES + MD5

Computation Time (3DES)	The Length of Characters 3DES	Computation Time (3DES + MD5)	The Length of Characters 3DES + MD5
5s	32	7s	24
6s	32	7s	22
4s	20	7s	14

Table 5. Experiment Scenario Comparison of Pixel Differences Message 1 and Message 2

Scenario	Pixel Difference		pixel difference	Message	
	Process 1	Process 2		Message 1	Message 2
Same Object Different Message	845	846	1	matakuliah kriptografi	prodi teknik informatika
Different Object Same Message	844	95	749	kampus itera	kampus itera
Different Objects Different Messages	844	97	747	ngoding sehat	ngoding mantaf

Table 6. Perception Image Parameter Testing Experiment




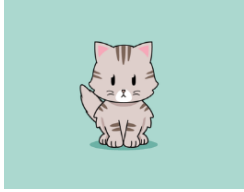


Parameter	Location Difference
Same Object, Different Message, Process 1	
Same Object Message Different Process 2	
Object Different Message Same Process 1	
Objects Different Message Same Process 2	




Table 6 (continued). Perception Image Parameter Testing Experiment

Parameter	Location Difference
Different Object Message Different Process 1	
Different Object Message Different Process 2	

To determine how much capacity to insert words in an image, the researcher conducted a comparison test of several images using images with a resolution of 10px * 10px, 20px * 20px, and 30px * 30px. The results of comparing the resolution to the difference in the number of pixels can be seen in Table 7.

From the Table 7, it is known that the larger the resolution of the image, the more character capacity that can be inserted into the steganographic image. In this manner, the pixel difference will also be even greater. Researchers conducted this test in order to be able to find pixel differences clearly when using HD images; the differences in the location of the pixels are not clearly visible when viewed without assistance.

Table 7. Image comparison test of Resolution

Resolution	Size	Max Char	Number of Pixel Differences	Image Results from Steganography
10px * 10px	10.9 KB	7 Char	26	
20px * 20px	11.5 KB	45 Char	123	
30px * 30px	12 KB	107 Char	285	

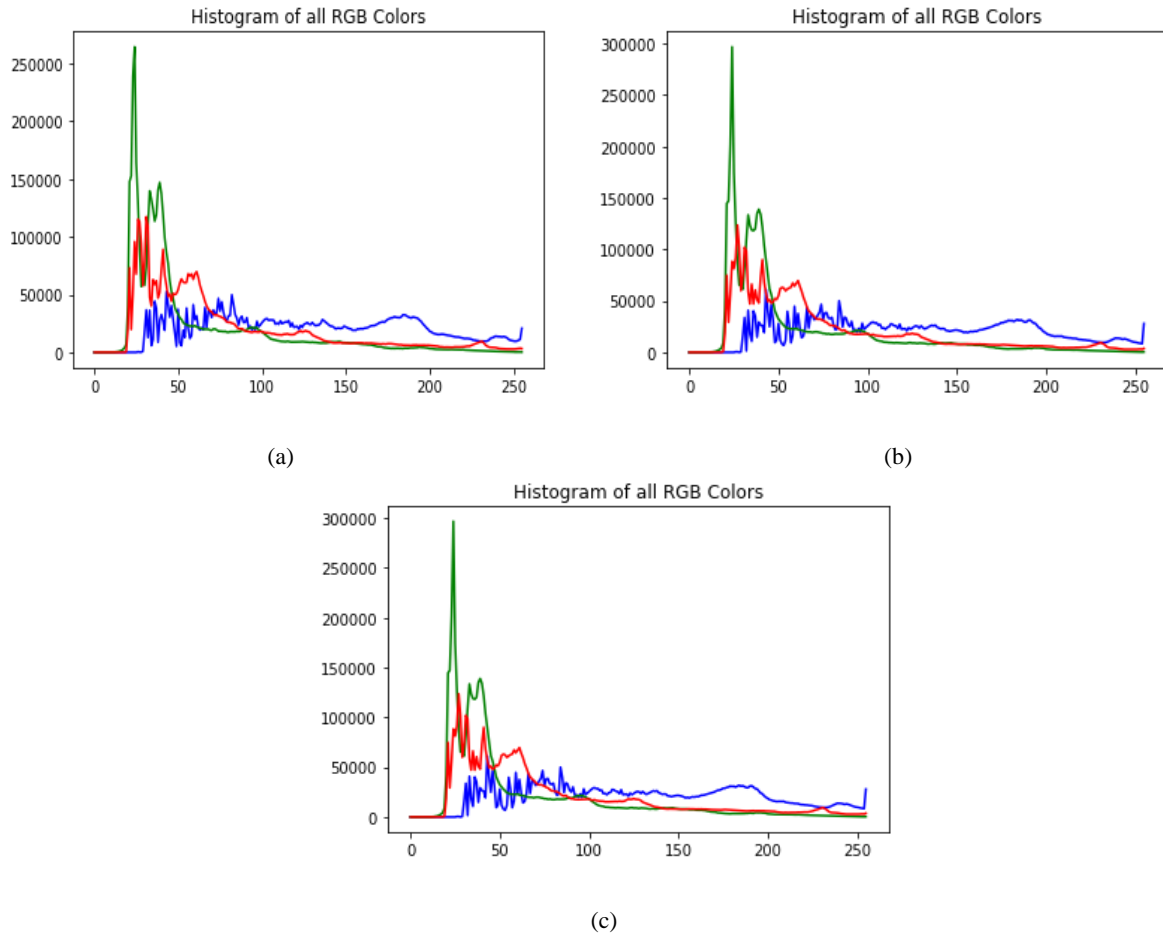


Figure 17. Histogram of: (a) Original image RGB; (b) image after plaintext is inserted; (c) Image after pasting with LSB and 3

Imperceptibility Test Using Histogram

In this program, researchers conducted a histogram test to determine the difference in frequency between the original image, the image inserted by plaintext, and the image inserted with 3DES. The original image before inserting the message can be seen in Figure 17 (a). The image after Plaintext is pasted can be seen in Figure 16 (b). The image after Inserting the 3DES and LSB messages can be seen in Figure 17(c).

Based on the results of the three histogram calculations, the researchers analyzed the disparities observed. In comparing the original image with the image embedded with text, discrepancies in the green histogram are evident, as illustrated in the graph above. In the two images containing embedded data, such as plaintext and 3DES-encrypted images, the histogram exhibits variations in red, blue, and green channels. This discrepancy arises because the embedded text alters the image's pixel values, resulting in deviations from the original image.

Computing Time Test

Computational time testing is done by measuring the time required for computing from the process to outputting the output. The results of the processing can be seen in the Table 8.

From the test results using six samples of the number of characters, it was found that the average time required for 3DES was 0.117 seconds, while the average time required for 3DES + LSB with MD5 Key was 0.798 seconds.

Table 8. The Comparison of Computation Time 3DES and 3DES + LSB

Length of Message	3DES	3DES + LSB with MD5 Key
1 character	0.021 seconds	0.382 seconds
10 characters	0.024 seconds	0.542 seconds
50 characters	0.036 seconds	0.675 seconds
100 characters	0.055 seconds	0.749 seconds
500 characters	0.086 seconds	0.991 seconds
1000 characters	0.124 seconds	1.453 seconds
Average	0.117 seconds	0.798 seconds

Table 9. Fidelity and Recovery Test Results

Stego Image and Size	MSE	Size Message (Characters)	PSNR	Recovery
Image1.png (40 Kb)	0.0001020	100	67.630	✓
Image2.png (25 Kb)	0.0002021	200	75.626	✓
Image3.png (38 Kb)	0.0002125	400	55.421	✓
Image4.png (56 Kb)	0.0000352	1000	63.865	✓
Median	0.0001298		66.635	-

Table 10. Rotation and resizing test results

Stego Image	Rotation			Resizing			
	90°	180°	270°	10%	30%	50%	80%
Image1.png	X	X	X	X	X	X	X
Image2.png	X	X	X	X	X	X	X
Image3.png	X	X	X	X	X	X	X
Image4.png	X	X	X	X	X	X	X

Fidelity, Recovery, and Robustness Testing

The fidelity aspect of the test results was assessed utilizing the Peak Signal-to-Noise Ratio (PSNR). The calculation formulas can be seen in equation (1) to (3).

$$PSNR = 10 \log_{10} \frac{C_{MAX}^2}{MSE} \tag{1}$$

Where MSE states the mean squared error which is defined as:

$$MSE = \frac{1}{MN} \sum_{X=1}^M \sum_{Y=1}^N (S_{XY} - C_{XY})^2 \tag{2}$$

S_{xy} is a representation of the bits in the stego image file and C_{xy} is a representation of the bits in the cover image. C_{max}^2 is a representation of the maximum value of pixels in the image, which is 255 and the minimum is 1. M and N is the size of Image.

$$C_{max}^2 \leq \begin{cases} 1 \\ 255 \end{cases} \tag{3}$$

The test results presented in Table 9 showed that as the number of message characters increased, the PSNR value decreased. Additionally, the results depicted in Table 10 indicate that the stego image is not resistant to rotation and resizing attacks. Among the four images tested, none of them succeeded in extracting the message.

CONCLUSIONS

Based on the results of program analysis and testing, several conclusions can be drawn. Firstly, it is observed that as the

number of messages inserted into the image increases, there is a corresponding increase in pixel differences within the stego image. Similarly, the inclusion of more colors in the image intended for message insertion results in higher pixel differences in the stego image. It is noted that stego objects can only present images with .png and .jpeg extensions. Furthermore, the combination of 3DES and LSB methods is effective in maintaining message security without altering the image's state or shape. This is supported by the fidelity testing, which yielded an average PSNR of 66.365, indicating a very good stego image quality. Additionally, the recovery testing demonstrated successful message extraction from the tested stego images.

However, when subjected to robustness testing using rotation and robustness attack techniques, it was found that the message extraction from the image was unsuccessful, highlighting the robustness of the method against these attacks. Moreover, computation time testing revealed that the average time required for computation ranged around 0.798 seconds for testing 1-1000 characters.

The researcher provides several valuable recommendations for future studies concerning the 3DES (Triple Data Encryption Standard) cryptographic algorithm. Firstly, they suggest exploring the versatility of 3DES in conjunction with LSB (Least Significant Bit) innovations, proposing that this combination could extend support to stego objects with file extensions beyond the conventional .png and .jpeg formats. Secondly, the researcher highlights the potential benefits of integrating 3DES with MD5 methods, specifically for encrypting plaintext in ASCII (base64

encoded) format. Moreover, they emphasize the importance of diversifying the keys utilized in the 3DES cryptographic algorithm to enhance security measures. Lastly, the researcher advises conducting tests with a more extensive range of parameters to evaluate the algorithm's performance and capabilities comprehensively. These recommendations aim to guide and inspire future researchers in advancing the utilization and understanding of the 3DES cryptographic algorithm in various applications and scenarios.

REFERENCES

- [1] J. Simamarta *et al*, *Sistem Keamanan Data*. kitamenulis.id, 2022.
- [2] A. Hidayat and A. Faizin, "Perbandingan Kriptografi Menggunakan Algoritma Data Encryption Standart (Des) Dan Algoritma Rivest Shamir Adleman (Rsa) Untuk Keamanan Data," *JASIEK (Jurnal Apl. Sains, Informasi, Elektron. dan Komputer)*, vol. 1, no. 2, pp. 143–148, 2019, doi: 10.26905/jasiek.v1i2.3451.
- [3] I. F. Ashari, M. Alfarizi, M. N. K., and M. A. H., "Vulnerability Analysis and Proven On The neonime. co Website Using OWASP ZAP 4 and XSpear," *J. Teknol. Komput. dan Sist. Inf.*, vol. 5, no. 2, pp. 75–81, 2022.
- [4] I. F. Ashari, L. Rizta Anugrah P, N. Andintya W, and S. T. Denira, "Analisis Celah Keamanan Dan Mitigasi Website E-Learning Itera Menggunakan Owasp Zed Attack Proxy (Zap) Vulnerability and Mitigation Analysis of the Itera E-Learning Website Using Owasp Zed Attack Proxy (Zap)," *Din. Rekayasa*, vol. 19, no. 1, pp. 29–35, 2023.
- [5] I. F. Ashari, V. Oktariana, R. G. Sadewo, and S. Damanhuri, "Analysis of Cross Site Request Forgery (CSRF) Attacks on West Lampung Regency Websites Using OWASP ZAP Tools," *J. Sisfokom (Sistem Inf. dan Komputer)*, vol. 11, no. 2, pp. 276–281, 2022.
- [6] P. B. T. Kumbara and M. A. I. Pakereng, "Perancangan Teknik Kriptografi Block Cipher Berbasis Pola Permainan Tradisional Rangkum Alu," *J. Tek. Inform. dan Sist. Inf.*, vol. 5, no. 2, pp. 189–200, 2019, doi: 10.28932/jutisi.v5i2.1714.
- [7] S. I. Febriani, S. Juanita, and M. Hardjianto, "Implementasi Kriptografi Teks pada SMS Menggunakan Algoritme Multiple Encryption dengan Metode RSA dan 3DES," *J. Telemat. Inst. Teknol. Harapan Bangsa, Bandung*, vol. 15, no. 2, pp. 77–84, 2020.
- [8] I. Gunawan, "Modifikasi Keamanan File dengan Algoritma Hill Cipher Untuk Mengantisipasi Dari Serangan Brute Force," *TECHSI - J. Tek. Inform.*, vol. 11, no. 2, p. 237, 2019, doi: 10.29103/techsi.v11i2.1272.
- [9] P. H. Rantellingi, E. Saputra, J. T. Informatika, U. Papua, and P. Korespondensi, "Algoritma Kriptografi Triple Des Dan Steganografi Lsb Sebagai Triple Des Cryptography Algorithm and Lsb Steganography As," vol. 7, no. 4, 2020, doi: 10.25126/jtiik.202071838.
- [10] T. R. Kuncoro and R. Aditama, "Analisis Kombinasi Algoritma Kriptografi Rsa Dan Algoritma Steganografi Least Significant Bit (Lsb) Dalam Pengamanan Pesan Digital," *Statmat J. Stat. Dan Mat.*, vol. 1, no. 2, pp. 60–82, 2019, doi: 10.32493/sm.v1i2.2947.
- [11] I. F. Ashari, A. W. Bhagaskara, J. M. Cakrawarty, and P. R. Winata, "Image Steganography Analysis Using GOST Algorithm and PRNG Based on LSB," *Techno.Com*, vol. 21, no. 3, pp. 700–713, 2022, doi: 10.33633/tc.v21i3.6331.
- [12] I. F. Ashari, "The Evaluation of Image Messages in MP3 Audio Steganography Using Modified Low-Bit Encoding," *Telematika*, vol. 15, 2021.
- [13] I. F. Ashari, "Graph Steganography Based On Multimedia Cover To Improve Security and Capacity," in *2018 International Conference on Applied Information Technology and Innovation (ICAITI)*, 2018, no. April 2019, pp. 194–201.
- [14] I. F. Ashari, "The Evaluation of Audio Steganography to Embed Image Files Using Encryption and Snappy Compression," *Indones. J. Comput. Sci.*, vol. 11, no. 2, pp. 318–336, 2022.
- [15] B. J. Simbolon, "Steganografi Penyisipan Pesan Pada File Citra Dengan Menggunakan Metode LSB (Least Significant Bit)," *J. Nas. Komputasi dan Teknol. Inf.*, vol. 4, no. 1, pp. 1–6, 2021, doi: 10.32672/jnkti.v4i1.2656.
- [16] Bakir and Hozairi, "Implementasi Metode Least Significant Bit (LSB) Dengan Enkripsi Cipher Caesar Pada Steganografi Menggunakan Image Processing," *JUSTINDO (Jurnal Sist. Teknol. Inf. Indones.)*, vol. 3, pp. 75–81, 2018.
- [17] E. Nirmala, "Penerapan Steganografi File Gambar Menggunakan Metode Least Significant Bit (LSB) dan Algoritma Kriptografi Advanced Encryption Standard (AES) Berbasis Android," *J. Inform. Univ. Pamulang*, vol. 5, no. 1, p. 36, 2020, doi: 10.32493/informatika.v5i1.4646.
- [18] A. M. Hariman and R. Puspasari, "Penerapan Aplikasi Pengamanan Data/File Dengan Metode Enkripsi Dan Dekripsi Algoritma 3Des Dalam Jaringan Lokal Area," *Semin. Nas. Teknol. Inf. dan Multimed. 2017 - STMIK AMIKOM Yogyakarta*, no. 2014, pp. 265–273.
- [19] Z. Basim and P. Painem, "Implementasi Kriptografi Algoritma RC4 Dan 3DES dan Steganografi Dengan Algoritma EOF Untuk Keamanan Data Berbasis Desktop Pada SMK As-Su'udiyah," *Skanika*, vol. 3, no. 4, pp. 45–52, 2020.

AUTHORS BIOGRAPHY



Ilham Firman Ashari

Ilham Firman Ashari is a lecturer from the Informatics Engineering at Institut Teknologi Sumatera.



Eko Dwi Nugroho

Eko Dwi Nugroho is a lecturer from the Informatics Engineering at Institut Teknologi Sumatera.



Dodi Devrian Andrianto

Dodi Devrian Andrianto is informatics engineering student at Institut Teknologi Sumatera



Makruf Alkarkhi

Makruf Alkarkhi is informatics engineering student at Institut Teknologi Sumatera



M. Asyroful Nur Maulana Yusuf

M. Asyroful Nur Maulana Yusuf is informatics engineering student at Institut Teknologi Sumatera