Jurnal Elektronik Ilmu Komputer Udayana
Volume 9 No. 3, February 2021

p-ISSN: 2301-5373
e-ISSN: 2654-5101

Web-based Image Steganography Application to Hide Secret Messages

Putu Isthu Canistya Chandra[a1], I Ketut Gede Suhartana[a2]

[a]Informatics Department, Udayana University
Jalan Raya Kampus Unud, Jimbaran, Bali, 80361, Indonesia
[1]isthucanistya16@gmail.com
[2]ikg.suhartana@unud.ac.id

Abstract

The Internet has become a very popular medium for communicating and exchanging information data. However, with the development of technology, information data theft is becoming more frequent so it is very necessary to maintain the security of information data. To maintain the security of information data, steganography can be used, namely hiding data or information to a medium. Least Significant Bit (LSB) is a method that can be used to perform the insertion. The Least Significant Bit (LSB) replaces the insignificant or smallest bit value so that when an insertion is done, the difference is not clear. So, this research aims to make a steganography application using the Least Significant Bit (LSB) method.

Keywords: *Applications, Steganography, Least Significant Bit (LSB), Base64, PHP (Hypertext Preprocessor)*

1. Introduction

Currently, the development of information and communication technology is very rapid, especially the internet, which is one of the most frequently used media in communicating and exchanging information data. However, various forms of crime can still be committed, such as stealing data or trying to find confidential information. So that information data security is one of the most important things until now. Several things must be considered when the message is sent by the sender so that the message reaches the recipient, namely confidentiality, integrity, authentication, availability, Access Control, and non-repudiation. These six aspects of computer security can protect messages sent by a sender to the receiver so that the message sent can reach its destination without missing a single bit of data [1].

To maintain the security of information data, we can use several methods, one of which is using steganography. The science of steganography is the study of hiding information data by inserting it into certain storage media such as images, videos, and so on. The science of steganography aims to disguise the existence of identity or secret message so that other people do not realize that something is contained in it.

Based on the problems described above, this study aims to build a website-based application that applies steganography by hiding secret messages into a jpg extension by utilizing Least Significant Bit (LSB) as a method of hiding secret messages. JPG extensions are chosen because they are often used to store images that will be used for web pages, multimedia, and other electronic publications [2]. So that there is no suspicion about the carrier media.

2. Research Methods

In this paper, to build a website-based steganography application, namely by using the PHP programming language (Hypertext Preprocessor) and utilizing the Least Significant Bit (LSB) method to hide secret messages. Least Significant Bit (LSB) is a technique of hiding information by converting the secret information into a sequence of bits in a byte (1 byte = 8 bits), then the

bits are put on the carrier by replacing every last bit on each byte [3]. Replacement on the last bit is done because it will only increase or decrease by 1 (one) byte value.

2.1. Application Design

The design of a steganography application that is built has two processes, namely the encryption process and the decryption process. In the encryption process, the user enters a key, secret message, and an image with a .jpg extension. The system will double encrypt the base64 of the secret message and key. After that, the encryption results will be inserted using the Least Significant Bit (LSB) method into the carrier image that has been entered by the user. The insertion result will be sent to the user for download.

While in the decryption process, the user enters the key and enters an image containing the secret message. The system will check the input image, if it is valid, it will be taken to the next process and vice versa. Furthermore, the system will perform the extraction using the same method as the insertion process. Then, the system will double base64 decryption to get the message and key. After getting the message and key, the system will check between the extracted and decryption key with the key that entered the user. If both keys are valid it will display a secret message that is carried by the image and if not then the system will ask the user to enter the key again. The UML Activity Diagram of this system can be seen in **Figure 1**.

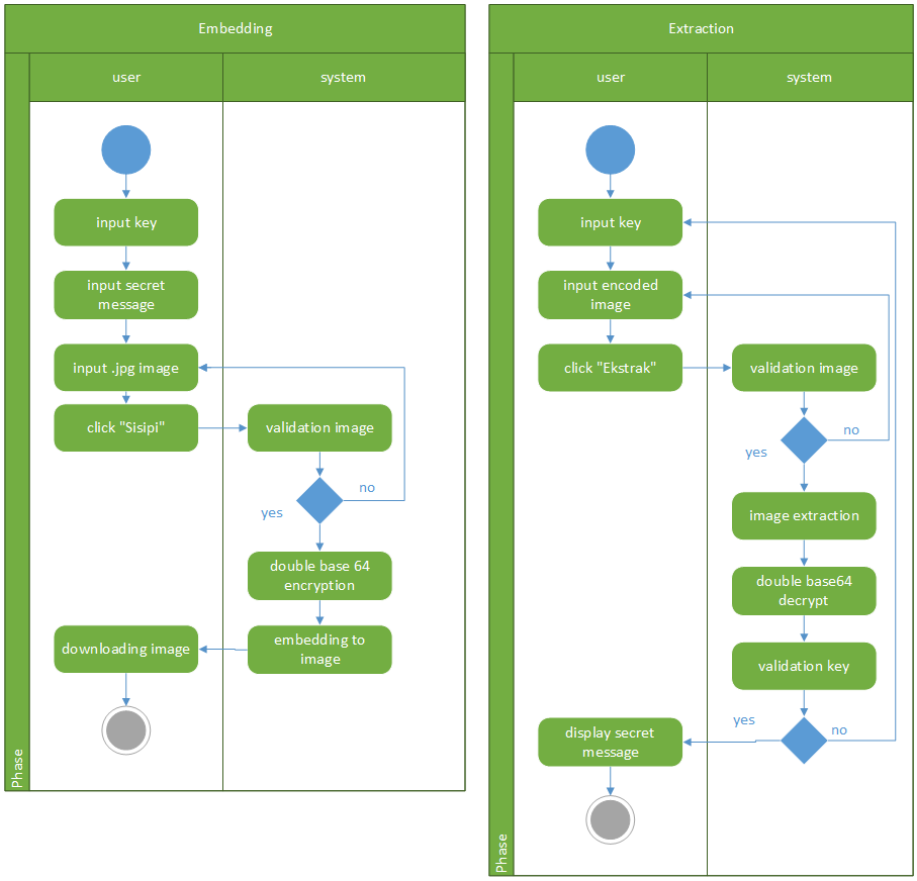

Figure 1.UML Activity Diagram

3. Result and Discussion

3.1. Application Interface

In the application display that has been created, there are two pages, namely an insertion page to carry out the insertion process and an extraction page to carry out the extraction process. There are three entries on the insertion page, namely, enter the key, enter the secret message, and enter an image that will be a container medium that has a .jpg extension. Whereas on the

p-ISSN: 2301-5373
e-ISSN: 2654-5101

extraction page there are only two entries, namely enter the key and enter the image you want to extract.

Figure 2. Insertion Interface

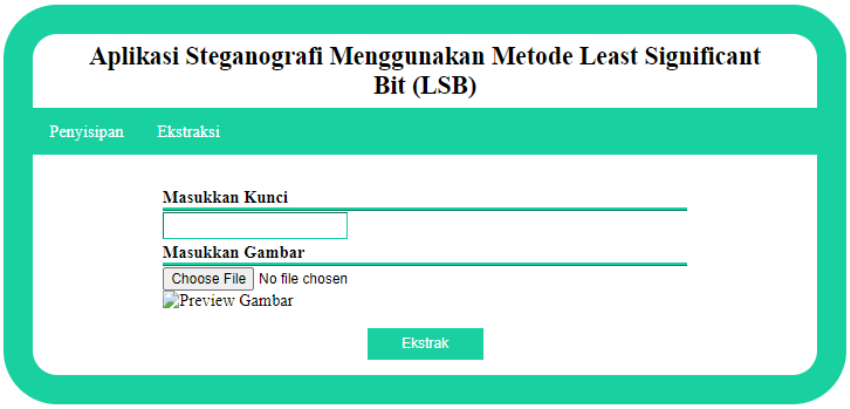

Figure 3. Extraction Interface

3.2. Testing Application

Application testing is carried out using black-box testing which is used to determine the functionality of the application that has been made.

a. Insertion

To test whether this application is running well, the author tries the application by entering a 6 character key, entering a secret message of 60 characters, and entering a jpg image which will be a medium container with a size of 64.1 KB (65,715 bytes).

Figure 4. Insertion Process

The results of the insertion process can produce images that have been inserted with a message with a size of 346 KB (355.062 bytes) and no difference is seen between pre-inserted and post-inserted images. So that the insertion process in this application is running well.

Figure 5. Image before insertion message.

Figure 6. Image after insertion message

b. Extraction
To test whether the application is running well in the extraction process, user enters the key and inserts the image that you want to extract.

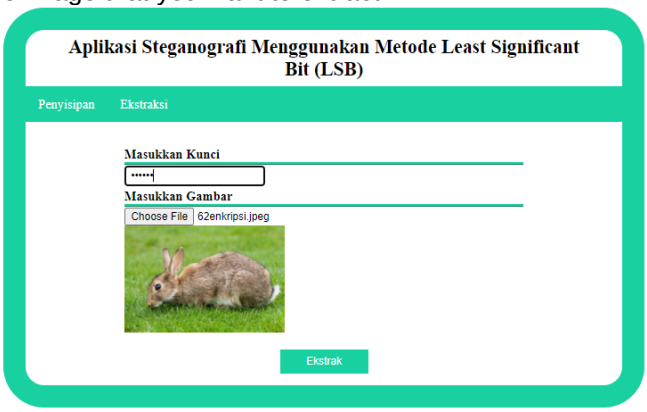

Figure 7. Extraction Process

Results of the extraction process can display the secret message contained in the image. So that the extraction process in this application is running well.

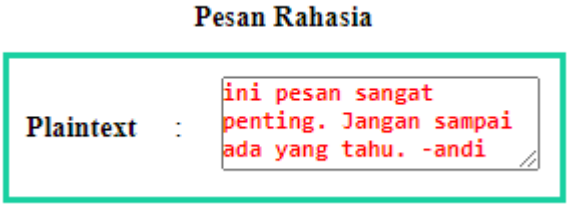

Figure 8. Extraction Results

4. Conclusion

The application of steganography using the Least Significant Bit (LSB) insertion method can hide confidential information data to the carrier media and the changes that occur are not visible. The test results show that there are differences in the size of the original image and the results of the image steganography. This change occurs because it depends on the length of the message that

p-ISSN: 2301-5373
e-ISSN: 2654-5101

is inserted. However, the resulting image steganography was unable to withstand the existing attacks. Therefore, further efforts are needed to improve the security of information data from several possible threats that will cause data loss such as distortion, cropping, and so on.

References

[1] N. Rismawati and M. F. Mulya, "Analisis dan Perancangan Simulasi Enkripsi dan Dekripsi pada Algoritma Steganografi untuk Penyisipan Pesan Text pada Image menggunakan Metode Least Significant Bit (LSB) Berbasis Cryptool2," *Faktor Exacta,* pp. 132-144, 2019.

[2] Darmayanti and A. H. K, "SISTEM STEGANOGRAFI PADA CITRA DIGITAL MENGGUNAKAN LEAST SIGNIFICANT BIT," *Prosiding Seminar Sains dan Teknologi FMIPA Unmul,* vol. I, 2016.

[3] D. Aditya, P. A. Panchadria and R. Setiyanto, "APPLICATION DEVELOPMENT METHOD STEGANOGRAPHY LEAST SIGNIFICANT BIT (LSB) WITH COMBINATION CRYPTOGRAPHIC ALGORITHM RC4 AND BASE64 BASED ON PHP," *Prosiding Seminar Nasional Multidisiplin Ilmu,* 2017.

This page is intentionally left blank