# DEVELOPMENT OF AN INTERACTIVE CRYPTOGRAPHING MACHINE BASED ON HIDING FILES IN FILES WITH A LIBRARY PROCESSING LIBRARY FOR JAVASCRIPT ON THE LOCALHOST PLATFORM

Hasby Fajrus Shodiq[1)], Desyderius Minggu[2)], Ananda Herdi Akbar[3)]
[1), 2), 3)]Prodi Teknik Telekomunikasi Militer. Politeknik Angkatan Darat
Jl.Raya Anggrek No.1 Junrejo, Batu, Indonesia
E - mail : [1)] hasbyshodiq777@gmail.com, [2)]desyderius07@gmail.com
[3)]aherdiakbar@gmail.com

## *PENGEMBANGAN MESIN SANDI BERBASIS INTERAKTIF MENYEMBUNYIKAN FILE DALAM FILE DENGAN LIBRARY PROCESSING FOR JAVASCRIPT DI PLATFORM LOCALHOST*

***Abstrak:*** *Perkembangan teknologi yang pesat mendorong peningkatan kebutuhan untuk pengamanan data yang cerdas dan sulit tebaca orang lain. Ada beberapa cara yang sangat menarik untuk dikaji yaitu teknik steganografi berbasis library Processing for JavaScript. Melalui berbagai serangkaian bercobaan yang mengimplementasikan teknik tersebut memfokuskan pada bagaimana file tersembunyi dapat disisipkan dan diekstraksi kembali tanpa mengurangi data dan mengganggu fungsi dari file tersembunyi tersebut maupun mengubah fungsi file penampungnya. Dengan berbagai rangakaian percobaan dan pengujian diperoleh gambaran bahwa percobaan ini mampu menjada konsistensi hasil dan bisa dikembangkan lebih lanjut, dengan harapan pengembangan sistem pengamanan data yang praktik, fleksibel dan aman dimasa depan.*
*Kata kunci: Steganografi, Processing for JavaScript, server lokal, Penyembunyian file, pengamanan data*

***Abstract:*** *Rapid technological developments are driving an increasing need for intelligent, unreadable data security. One particularly interesting approach is the Processing for JavaScript library-based steganography technique. A series of experiments implementing this technique focused on how hidden files can be embedded and re-extracted without compromising the data, disrupting the function of the hidden file, or altering the function of the host file. Through various experiments and tests, it was found that this experiment was able to maintain consistent results and could be further developed, with the hope of developing a practical, flexible, and secure data security system in the future.*
*Keywords: Steganography, Processing for JavaScript, local server, File hiding, data security*

## INTRODUCTION

In the 5.0 era, technological developments are accelerating alongside the advancement of technology. This means that this increasing development will increasingly simplify human work, both in terms of work and daily activities, and can also facilitate communication access in various fields. However, with technological advancements, there is a crucial aspect of communication security, especially in critical sectors such as government, banking, and the military. The potential vulnerability of news or letter content leaks could occur in the 5.0 era by irresponsible parties for misuse.

On July 16, 2025, a circular from the Central Kalimantan Regional Police's operational task force leaked, stating that the telegram containing an order to control illegal mining (PETI) with the number STR: 276/VI/OPS.1.3/2025 dated June 6, 2025, had been in effect. This leak caused a number of conventional gold miners to withdraw all heavy equipment and temporarily halt mining activities. This leak also resulted in the enforcement not running properly. Of course, this incident did not only happen once or twice, but it often occurs among law enforcement officers and those guarding the sovereignty of the Republic of Indonesia.

In addition to the police, the Indonesian National Armed Forces (TNI) also leaked a letter of assignment to Papua in 2021. This leak included the locations where the operation would take place, the names of the troops, the number of personnel, and the types of weaponry. This leak was widely circulated online, especially on Facebook and WhatsApp. This leak has had a significant impact on troop safety, potentially leaking handovers at the operation location. Certainly, the enemy will easily read and easily defeat the TNI if this leaked data is taken seriously.

Examples of leaked operational letters suggest that technological advancements are a double-edged sword, capable of facilitating human work and also detrimental to others. Security is essential for operational mail delivery for Indonesian authorities to ensure it operates according to command and control requirements. This incident also highlights the need for appropriate security measures to prevent mail from being leaked to third parties. In light of current technology, several information security techniques have been developed, such as cryptography, steganography, and others (Sani & Nujjaid, 2025).

From all these problems, a cipher machine was created to secure the news that will be sent so that it can reach the recipient safely and cannot be read by other parties. With this research, there is an update to the

cipher machine version by using a local server that uses the processing library for JavaScript so that it can process steganography on the website provided. From this research, the differences will also be sought before and after encoding or decoding, as well as the file size and also how about the results of the received file(Xu et al., 2023).

Steganography is a method that hides the original file in the file that will be overtaken, if cryptography scrambles the contents of the message so that it cannot be read by others, then steganography hides files in larger files, this steganography aims to maintain the security of the news from the contents of the letter so that it is safe and cannot be known by others who want to misuse the contents of the message. In the current era, steganography has developed greatly, especially with the existence of artificial intelligence, so that images that look ordinary but inside contain secret messages that can only be read by the recipient intended by the sender (Şahin et al., 2021).

A local server typically resides on a computer or device used as a server. This local server is very useful in supporting information systems and efficient data sharing in a limited environment. Local servers also create access to a closed network, allowing only devices connected to the network to access it. With numerous advantages such as increased data security, full system control, lower operational costs, and no third-party dependency, local servers are often used for application development before being implemented on public servers (Erik Taufik et al., 2021).

The Processing for JavaScript library is a programming language with a Java-based initial development stage, aimed at visual learning in programming, animation, and generative art. Processing for JavaScript was developed as a JavaScript library that adopts the Processing syntax paradigm and utilizes HTML5 Canvas technology as the primary medium for depicting two-dimensional and three-dimensional graphics.

## RESEARCH METHODS

The method used is the experimental or direct trial method. This method was chosen because it allows direct testing through the implementation of a program hypothesis that has been specifically designed using the processing library for JavaScript. This allows the system to be tested repeatedly to achieve the desired results.

This study uses two variables: independent and dependent. The independent variable is the file hiding algorithm in the program being run, while the dependent variable is the resulting file that has been overlaid with the hidden file. This

study should provide insight into the extent to which the system can hide files within other files without significantly altering the primary function of the overlaid file (Li et al., 2023).

Research Components

1. Processing for JavaScript is a JavaScript library that is commonly used as the main medium for developing graphical functions and data processing to hide files in data and visually.

2. The steganography algorithm can be said to be a basic technique for hiding files, this algorithm also functions to insert files into the files that will be overwritten.

3. The storage media file is the main file that will be used for the main display that people will see, so it is a place to store files that will be hidden.

4. The file to be hidden is the original file or a file whose contents or data are protected, therefore it is hidden in the media container file.

5. Testing devices in the form of laptops, PCs, computers or mobile phones that are equipped with browser software that can run JavaScript-based programs.

**System Planning**

The system design in this study aims to design a method for hiding files within other files using the Processing for JavaScript library. This system is composed of several important components that are interconnected, starting from the file input stage, the insertion process, to the data extraction process(Apau et al., 2024).

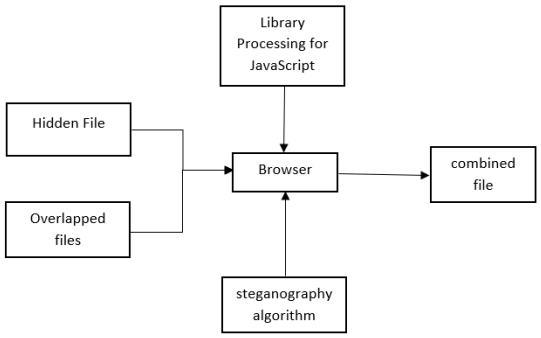


Figure 1. System block diagram

The first step involves selecting a storage medium, which is usually an image, music, video, or other document file, and the file to be hidden, such as text or a secret message. To ensure the process runs smoothly, it's recommended that the minimum size of the storage medium be five or more times larger than the file to be hidden. For example, if the file to be hidden is 100 KB, the recommended storage medium should be larger than 500 KB to prevent the file from failing to hide(Smith et al., 2024).

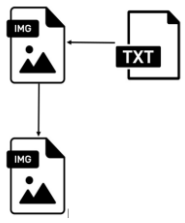


Figure 2. Application illustration

This study provides an illustration of the interaction between the user and the

Processing for JavaScript-based file hiding system. The main actor in the system is the user who has two main needs, first hiding files in a container file and retrieving files from the container file. Users can directly use the encode feature, which is to hide the files they want to hide into a passenger file, and to restore using the decode feature, which is to restore the files hidden in the passenger file so that they can be read and viewed with the naked eye again in their entirety(Subramanian et al., 2021).

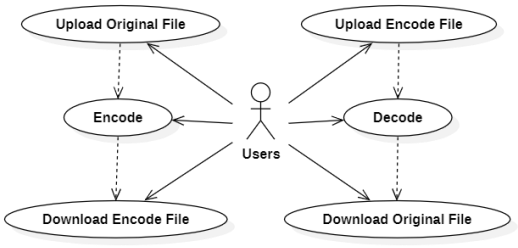


Figure 3. Use case diagram

The file embedding process, where this system implements a PROCESSING FOR JAVASCRIPT-based steganography algorithm, makes the hidden file invisible to the naked eye. Only the container file that hides the hidden file is visible. The key is to adhere to a predetermined file size to prevent damage to the hidden file.

## RESEARCH RESULT

There are several results that have been obtained from several experiments before being encoded and decoded in the steganography program in the Processing for JavaScript library.

1. The physical media file does not change.


Figure 4. Physical container file

2. Change the size of the container file

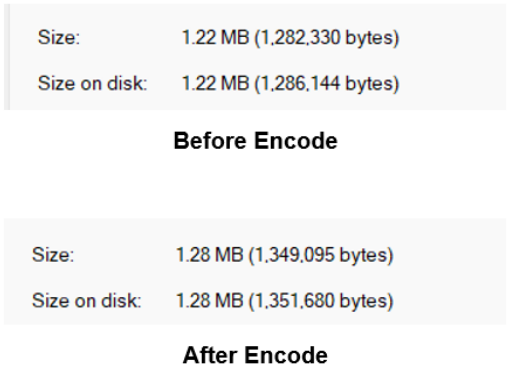


Figure 5. Container file size

3. Hidden Files Do Not Change When Before Encode and Finish Decode

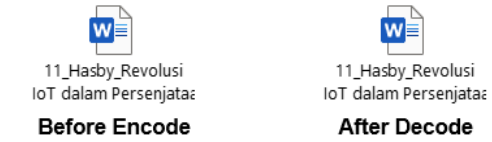


Figure 6. Passenger File

4. Encode and Decode test results

Table 1. Encode and Decode Testing

| Process | Number of trials | Successful trials | Success rate |
|---|---|---|---|
| Encode | 10 | 10 | 100% |
| Decode | 10 | 10 | 100% |

**DISCUSSION**

Based on the results of this study, it is stated that the container file (in the example using an image) does not show any crucial changes in its visual appearance or functionality after the hidden file is inserted. This proves that the Processing for JavaScript-based steganography method applied is successful without any visual or functional changes. The main goal of steganography is to hide a file so that it is not visible and the file being hosted is not suspected of having a hidden file inside it, so this system also indirectly increases the level of security higher in securing news. This fact also provides the concept that the success of hiding a secret message is not only inserting it into a file but ensuring that there is no indication of suspicion of the container file.

In this research experiment, there was also a difference in file size, from the original 1.22 MB container file to 1.28 MB after the hidden file was inserted. The difference in size change was 0.06 MB (approximately 60 KB), while the hidden file itself had a file size of 15 KB. This condition occurs because the encoding process not only adds the hidden file but also adds bits and structural information so that the hidden file can be inserted correctly so it is not damaged and can be extracted back to its original form.

The hidden file remained unchanged after both encoding and decoding, as evidenced by its unaltered external and internal contents. No damage or size was found in the hidden file. This experiment demonstrates that the JavaScript processing library-based steganography algorithm is indeed capable of maintaining system reliability for storing and recovering data accurately.

The results of the test experiments on encode and decode table 1 show that the process successfully achieved 100% success. Of the 10 trials, 10 were successful, indicating that the encode process was successful without damaging the passenger file and the decode process was successful without damaging the file inserted in the passenger file. With the results of the steganography experiment based on the processing library for JavaScript, it has a very high level of consistency and reliability in inserting hidden files and extracting them back to appear visible without damaging the bits of the file. This finding indicates the effectiveness of the processing library for JavaScript in the steganography process.

## CLOSURE

Referring to the experimental results of the research that has been conducted, it shows that the relationship between the variables of the container file size, the size of the hidden file, and the success of the encoding and decoding processes is related. A container file that is five times larger than the hidden file will not cause significant visual changes, even though there is a change in the size of the container file when it is finished encoding. From several experiments, it also has a 100% success rate, indicating that the steganography process based on the Processing for JavaScript library is indeed very effective and reliable, especially since the decode results also do not change the data structure of the bits of the hidden file when re-extracted. As a suggestion, future researchers can also vary the type of passenger file in terms of size and extension to be able to have a broader picture of the optimization limitations of the Processing for JavaScript-based steganography technique.

## BIBLIOGRAPHY

Apau, R., Asante, M., Twum, F., Hayfron-Acquah, J. Ben, & Peasah, K. O. (2024). Image steganography techniques for resisting statistical steganalysis attacks: A systematic literature review. *PLoS ONE*, *19*(9 September). https://doi.org/10.1371/journal.pone.0308807

Erik Taufik, M., Maariful Huda, M., & Siwi Candra, T. (2021). *IMPLEMENTASI MANAGEMENT SERVER PADA RADIO HYBRID CIGRA MENGGUNAKAN WEBRTC*.

Li, G., Li, S., Li, M., Zhang, X., & Qian, Z. (2023). *Steganography of Steganographic Networks*. www.aaai.org

Şahin, F., Çevik, T., & Takaoğlu, M. (2021). Review of the Literature on the Steganography Concept. *International Journal of Computer Applications*, *183*(2), 38–46. https://doi.org/10.5120/ijca2021921298

Sani, N. F. M., & Nujjaid, M. A. (2025). Efficient Text Data Hiding Technique Using Cryptography and Steganography. *Journal of Computer Science*, *21*(1), 43–51. https://doi.org/10.3844/jcssp.2025.43.51

Smith, V., Mendoza, M., & Ullah, I. (2024). Data Security Techniques Using Vigenere Cipher And Steganography Methods In Inserting Text Messages In Images. In *Journal of Information System and Technology Research journal homepage* (Vol. 3, Issue 3).

Subramanian, N., Elharrouss, O., Al-Maadeed, S., & Bouridane, A. (2021). Image Steganography: A Review of the Recent Advances. *IEEE Access*, *9*, 23409–23423. https://doi.org/10.1109/ACCESS.2021.3053998

Xu, Y., Mou, C., Hu, Y., Xie, J., & Zhang, J. (2023). *Robust Invertible Image Steganography*.