Improved LSB-Based Image Steganography Integrated With Public Key Encryption

Yoong-Shem Hong2, a) and Shahbe Mat-Desa1, 2, b)

*1Centre for Image and Vision Computing, CoE for Artificial Intelligence, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Malaysia*

*2Faculty of Computing and Informatics, Multimedia University, Persiaran Multimedia, 63100 Cyberjaya, Malaysia*

*b) Corresponding author: shahbe@mmu.edu.my*

*a) 1201103427@student.mmu.edu.my*

**Abstract.** The focus of this paper is to discuss the approach of hiding information using LSB steganography and meanwhile boost the security of the system with Public Key Encryption (PKE). Many people prefer LSB because it is both simple and efficient but is still vulnerable to cyber attacks. When we use RSA encryption, only the intended recipients can access the message, providing additional security to the system. The method is carried out using Python for its cryptographic features. Tests are done to check that pictures are of good quality throughout the embedding process, without affecting their overall appearance. The same algorithm works for different sized images and aspects. To check how stable and effective the system is, a wide range of messages were sent through it. Experiments have revealed the ability to keep image quality high while also transporting secret data inside the picture. This research points out that merging steganography with encryption can increase the safety of digital communication.

# INTRODUCTION

Because of digital communication is advancing rapidly, more people are requiring security when sending data. Traditional ways of protecting data through cryptography attract attention which makes them vulnerable to attacks. Steganography is a better choice than cryptography for anyone who needs to keep their information hidden. A frequent way of hiding information is the Least Significant Bit (LSB) approach which inserts secret data into the least important parts of an image’s pixel values. Although LSB is effective, it can be broken by using statistical or visual approaches because its pattern of hiding data is easy to spot.

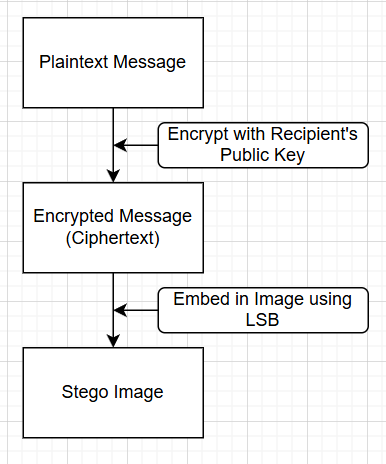
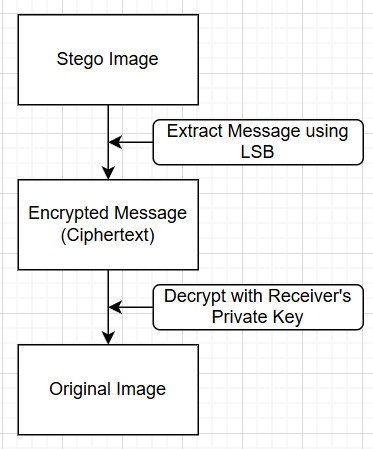
As a solution, this paper suggests using LSB steganography together with Public Key Encryption (PKE). For the message to be embedded in the image safely, Rivest, Sharmir, Adleman (RSA) is used which to makes it possible for only people with the private key to retrieve the information after decryption. Using this approach, our data is well protected and no one can detect the presence of hidden messages. Here, we are presenting this research with the following aims:

1. To identify the issues that are connected to text-in-image steganography, especially the ones that appear images cannot carry a lot of information, they’re vulnerable to steganalysis and there is no encryption used while sending them. This helps researchers identify chances to improve their work in the same area.
2. To develop a steganography system by adding RSA encryption to the LSB method for hiding information in images. Users can pick an image, add a secret message by using the recipient’s public key and safely put the encrypted message into the image.

## RELATED WORK

Many people have looked into steganography as a safe method of communication. There are several techniques, including methods from the spatial domain such as LSB substitution and techniques from the transform domain, such as Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT). LSB is still recommended because it is simple and does not greatly affect the quality of the image, but is still possible to find out about it using statistical analysis and image processing attacks.

Various researchers have suggested using hybrid techniques which merge steganography with encryption. Moumen and Sissaoui [1] developed a process that links LSB steganography with RSA encryption which gives better security against attacks by hackers and still preserves the high quality of images. Hiary et al. [2] introduced a mixed steganography system that uses LSB matching and replacement which makes the messages embedded in images more secure. The basic working structure of RSA encryption, illustrating both the sender and receiver processes is shown in Figure 1.

(a) (b)

**FIGURE 1**.Basic working structure of RSA encryption: (a) Sender side and (b) Receiver side

Innovations today are designed to increase dependability, strength and the amount of payload that can be carried. Rahman et al. [3] suggested adding Magic Matrix, Multi-Level Encryption Algorithm (MLEA), and pixel-flipping techniques to the LSB substitution method. The new approach helped reduce errors in embedding and gave better results in terms of transparency, resistance to tampering and efficiency. They stressed that it is important to choose the right image formats and optimize messages to ensure the message is hidden and still strong.

With the help of deep learning, steganography has become more complex. Kweon et al. [4] introduced a method of steganography that involves using private keys with deep multi-image images. In their model, several secret images are hidden in a single cover image and only the proper private key can unlock each image. It deals with the major security issue of secret data being secretly extracted on a wide scale and improves access control.

Other researchers like Alanzy et al. [5] and Olewi et al. [6] carried out studies to combine hybrid encryption, Morse code encryption and LSB steganography for more secure digital communication. Meanwhile, Zhang et al. [7], introduced a steganographic system using Elliptic Curve Cryptography which guarantees the hiding of the data and provides strong protection against malicious attacks on the steganographic ciphertext.

Moreover, Aslam et al [8], by reviewing different studies, stated that using LSB together with RSA or AES encryption makes the system more secure. Because of this strategy, if the hidden message is found by steganalysis, the real content is still safe and can only be viewed with the correct decryption key.

These studies highlight that using cryptography along with LSB steganography increases the safety of communication. However, there are difficulties in ensuring a high capacity, low visibility and strong resistance to steganalysis. Further studies should investigate how to embed data using adaptive methods and look into other cryptographic algorithms to boost the security of steganography.

## METHODOLOGY

The method for developing this safe text image steganography application that combines with public key encryption is described in detail in this project. There are five main stages of this development:

***Stage 1: System Design.*** The application is made to securely embed encrypted messages in JPEG or PNG files by combining RSA encryption with Least Significant Bit steganography. Important components include:

1. User interface: Allows users to upload JPEG or PNG pictures, input the message they wanted, view their own keys, and perform encoding or decoding process.
2. Steganography Engine: In charge of employing LSB techniques to embed and extract messages.
3. Cryptographic Module: It uses RSA to encrypt the user input message before embedding and decrypt messages after extraction.

***Stage 2: Key Generation and Management***. Once a new user has created a new account, the system will automatically generate a unique pair of RSA keys, both private and public keys for each user:

1. Public Key: Utilized by the senders to encrypt messages and send it to their recipient.
2. Private Key: Kept secret by the recipient and use to decrypt messages intended for them.

***Stage 3: Encryption and Embedding***. When the user uploads an image and enters a message, the system first will encrypt the input message using the recipient’s public key with RSA encryption. Then the encrypted data is divided into Base64-encoded chunks to keep it compatible with the LSB embedding process. The full encrypted payload is proceeded to embed into the least significant bits of the image pixel values. After successfully embedded, the steganography image containing the hidden and encrypted message is saved and displayed to the user. The flowchart for this encryption and embedding process is illustrated in Figure 2(a).

***Stage 4: Message Extraction and Decryption.*** In order to retrieve a hidden message, the user has to upload the encoded image and enter their own private key. The system will extract the embedded data using the LSB decoding function, pulling the hidden message from the image’s pixel values. The encrypted chunks are then decoded from Base64 and decrypted with the user’s private RSA key. Once decrypted successfully, the original plaintext input message is shown. The flowchart for extraction and decryption is presented in Figure 2(b).

A diagram of a public key

AI-generated content may be incorrect. A diagram of a computer process

AI-generated content may be incorrect.

1. (b)

**FIGURE** **2**. Flowcharts of (a) Encryption and embedding, and (b) Extraction and decryption

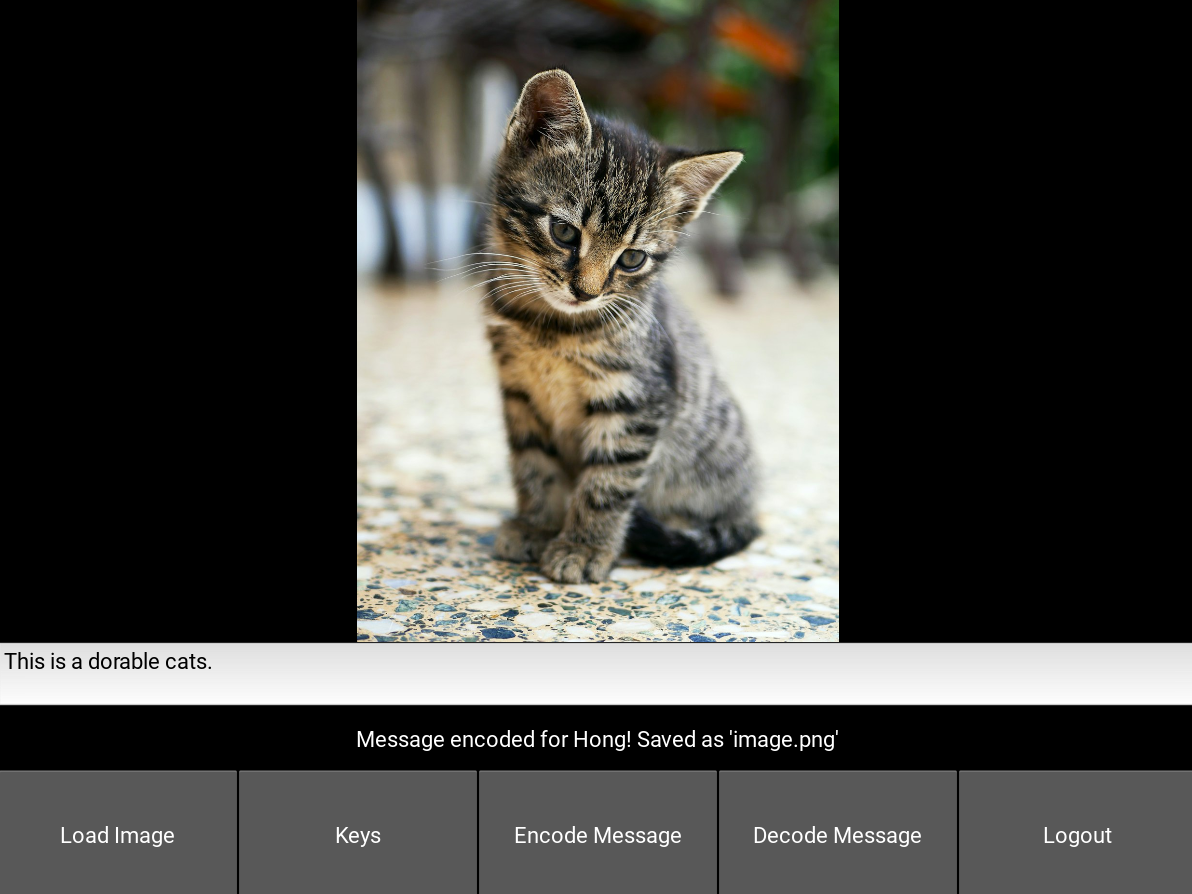
***Stage 5: Testing and Evaluation***. The application was tested under a variety of conditions to ensure it is both robust and reliable. This includes verifying the encryption and decryption processes by using the wrong public and private keys to confirm that unauthorized access is prevented. The application is also evaluated to make sure the embedding process preserves image quality without noticeably changing the image’s appearance. Additionally, performance tests are also carried out on different image resolutions and message lengths to assess how well the application performs and remains stable across different types of scenarios.

The PC based steganography application was tested in different types of scenarios to assess its functionality, performance, and security. Screenshots of the application are shown in Figure 3 (a) and (b). The application allows users to embed encrypted text messages into PNG or JPEG images using the LSB technique combined with RSA public key encryption. Functionality tests were carried out and it confirmed that messages were properly encrypted with the recipient’s public key and successfully embedded within the images. During the decoding process, only the correct private key could decrypt the message, making sure the access was limited to the intended recipient.

A screenshot of a login screen

AI-generated content may be incorrect.

(a)



(b)

**FIGURE** 3. Steganography application: (a) Login screen, and (b) Screenshot of the main menu

The measurement of the quality between the before and after image can be defined by Peak Signal to Noise Ratio (PSNR) and Mean Square Error (MSE). The PSNR and MSE formulas are given in Equations (1) and (2), respectively.

(1)

(2)

To assess the visual quality and imperceptibility of the stego images, PSNR and MSE were measured after embedding encrypted messages using LSB steganography.

# RESULTS AND discussion

To assess the visual integrity of images after embedding, comparisons were made between the original and stego-images. Both visual inspection and file size analysis revealed a few changes. Table 1 provides the image size changes. For JPEG, the original file size of 203 KB increased significantly to 1.69 MB after embedding, demonstrating a dramatic expansion. This can be attributed to JPEG’s lossy compression algorithm, which becomes less efficient when pixel values are altered at the bit level. Since the encrypted payload introduces high-entropy, non-compressible data, the JPEG file fails to maintain its original compactness. In contrast, the PNG image, which uses lossless compression, showed a more moderate increase from 38.6 KB to 85.3 KB after message embedding. This indicates that PNG images are more suitable for steganographic applications, as they preserve the integrity of modified pixel data without ballooning disproportionately in size. The results emphasize the importance of choosing the right image format depending on the balance between size efficiency and message capacity. The JPEG and PNG before and after embedding texts are shown in Figure 4 and Figure 5, respectively.

A kitten sitting on the ground

AI-generated content may be incorrect. A kitten sitting on the ground

AI-generated content may be incorrect.

(a) (b)

**FIGURE 4**. (a) JPEG before and (b) JPEG after embedding texts

A cat sitting on the floor

AI-generated content may be incorrect. A cat sitting looking up

AI-generated content may be incorrect.

(a) (b)

**FIGURE 5**. (a) PNG before and (b) PNG after embedding texts

**TABLE** **1.** Image size changes

|  |  |  |
| --- | --- | --- |
| **Image Format** | **Before** | **After** |
| JPEG | 203 KB | 1.69 MB |
| PNG | 38.6 KB | 85.3 KB |
|  |  |  |

The MSE and PSNR values are shown in Table 2. For the JPEG image, the resulting MSE was 0.000349 with a PSNR of 82.69 dB. This extremely high PSNR indicates excellent image quality preservation, with virtually no noticeable distortion despite the embedded data. The minimal MSE further confirms that the modifications to pixel values were negligible. Although JPEG is a lossy format and typically not preferred for pixel-sensitive operations like steganography, this result demonstrates that the method can still be effective when carefully implemented. In contrast, the PNG stego image processed using a lossless compression format yielded an MSE of 0.00462 and a PSNR of 71.48 dB. While slightly lower than JPEG in this test, the result is still highly favorable, as PSNR values above 40 dB are generally considered imperceptible to human vision. The higher MSE compared to JPEG may be attributed to differences in compression behavior and original image complexity. Nevertheless, the PNG output confirms that the embedding process remains visually undetectable and suitable for secure message hiding using RSA-encrypted payloads.

**TABLE** **2**. MSE and PSNR values

|  |  |  |
| --- | --- | --- |
| **Image Format** | **MSE** | **PSNR** |
| JPEG | 0.000349 | 82.69 dB |
| PNG | 0.004620 | 71.48 dB |

The encryption accuracy was also validated through multiple tests. Messages encrypted with RSA could only be decrypted using the corresponding private key. When incorrect keys were used, the system will generate an error message, demonstrating the strength of the encryption. This is to ensure that message confidentiality is always maintained, as unauthorized users cannot easily access the original plaintext message even if they manage to extract the hidden data from the image.

# CONCLUSION

Overall, combining RSA public-key encryption with LSB steganography is able to improve the overall security of hidden data transmission. Experimental results have shown the effectiveness of this approach in preserving both image quality and message confidentiality, even after embedding encrypted messages. Tests were conducted on different types of image resolutions and formats provided a few important information. PNG images, which use lossless compression, were better for steganographic embedding exhibiting only moderate increases in file size and negligible visual distortion. On the other hand, JPEG images showed significant file size inflation due to their lossy compression reacting poorly to the encrypted payload, despite maintaining a high PSNR value and low MSE values. The system also proved that able to handle different kinds of message lengths with consistent performance. Even with longer messages, image quality remained high, with PSNR values showing over 70 dB. The encryption method was equally robust, allowing only one correct private key to decrypt embedded messages. This is to ensure that even if hidden data is extracted, unauthorized users cannot access the original content at all. In conclusion, this dual-layer security model offers strong protection, adaptability to different types of image and message sizes, and minimal impact on visual quality, making it an effective solution for secure digital communication.

# References

1. A. Moumen, and H. Sissaoui, “Images Encryption Method using Steganographic LSB Method, AES and RSA algorithm,” Nonlinear Engineering **6**(1), (2017).
2. H. Hiary, K. Eddin, M. S., and A. Al-Dhamari, “A Hybrid Steganography System based on LSB Matching and Replacement,” Ijacsa **7**(9), (2016).
3. S. Rahman, J. Uddin, H. Hussain, S. Shah, A. Salam, F. Amin, I. De La Torre Díez, D.L.R. Vargas, and J.C.M. Espinosa, “A novel and efficient digital image steganography technique using least significant bit substitution,” Sci Rep **15**(1), 107 (2025).
4. H. Kweon, J. Park, S. Woo, and D. Cho, “Deep Multi-Image Steganography with Private Keys,” Electronics **10**(16), 1906 (2021).
5. M. Alanzy, R. Alomrani, B. Alqarni, and S. Almutairi, “Image Steganography Using LSB and Hybrid Encryption Algorithms,” Applied Sciences **13**(21), 11771 (2023).
6. H.I. Olewi, M.M. Msallam, S.K. Salim, and H.A.H. Al-Behadili, “Enhanced Security Through Integrated Morse Code Encryption and LSB Steganography in Digital Communications,” TS **41**(1), 519–524 (2024).
7. X. Zhang, K. Chen, J. Ding, Y. Yang, W. Zhang, and N. Yu, “Provably Secure Public-Key Steganography Based on Elliptic Curve Cryptography,” IEEE Trans.Inform.Forensic Secur. **19**, 3148–3163 (2024).
8. M.A. Aslam, M. Rashid, F. Azam, M. Abbas, Y. Rasheed, S.S. Alotaibi, and M.W. Anwar, “Image Steganography using Least Significant Bit (LSB) - A Systematic Literature Review,” in *2022 2nd International Conference on Computing and Information Technology (ICCIT)*, (IEEE, Tabuk, Saudi Arabia, 2022), pp. 32–38.