Implementation of Image Security using Elliptic Curve Cryptography RSA Algorithm and Least Significant Bit Algorithm

Handrizal1*, Herriyance1, J Hanayong1, M Zarlis1 and P Sihombing1

1 Department of Computer Science, Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Jl. University No. 9-A, Medan 20155, Indonesia

* handrizal@usu.ac.id

Abstract. Data exchange is a very important thing in today’s technological development. Two important factors must be considered in the process of data exchange, namely speed and security. Rapid technological development can result in insecure data because it can be stolen or manipulated by people who are not privileged. Thus, we need a way or method to be able to overcome these problems, such as cryptography and steganography. The Rivest-Shamir-Adleman or RSA algorithm is an asymmetric cryptographic algorithm that uses 2 prime numbers to generate a public key and a private key that is used in the encryption process for the data. The ECC concept is used to determine the seed or initial value in the generation of prime numbers in the RSA Algorithm by using point addition and multiplication in an elliptic curve. The Least Significant Bit Algorithm is a Steganography Algorithm that inserts bits from the message at the last bit in a cover, replacing the last bit. The parameter used is Running Time in steganography. The results of the research show that image files can be hidden in another image file and returned to normal with a combination of cryptography and steganography with an average Running Time for the insertion process of 3090.167 ms and the extraction process of 2917.167 ms.

1. Introduction

Communication is a very important thing. Information exchange can be done easily and quickly in various sizes and file types. With the advancement of information technology, it is not enough to only focus on the size of the information and the speed of exchange. The more important information is, the more secure it must be to prevent unauthorized parties from accessing said information. There are many ways to secure data, one of which is Cryptography and Steganography.

Images are the chosen media in this research paper, based on a variety of factors, such as total size, amount of information or data per size, and popularity of social media interaction in the form of image sharing. With sizes larger than texts generally, images also can store more information than texts and convey it faster. Coupled with the popularity of image sharing social media and applications, the common popularity of images makes hiding important images in other cover images very appealing.

Cryptography is the science or art of transforming data or information into other forms that can only be understood by the party intended to receive the data or information. In general, the process is divided into encryption and decryption processes. The data transformations are carried out using various algorithms. The public key algorithm refers to a cryptographic system that requires 2 different keys, one secret, and one public.
The Elliptic Curve Cryptography (ECC) is public-key cryptography developed by Neal Koblitz and Victor S. Miller in 1985. ECC uses points in an elliptic curve in the encryption and decryption process. Various studies on the ECC have concluded that the calculation of the Elliptic Curve Discrete Logarithmic Problem increases in difficulty exponentially with the size of the key used [1].

One of the algorithms used is the RSA algorithm, which is a public key cryptography algorithm invented by R. Rivest, A. Shamir, and L. Adleman. The RSA cryptosystem algorithm works on the speed of calculating the value of very long prime numbers and the difficulty in finding the factor of these large prime numbers [2].

Steganography is the study of inserting or hiding data or information behind other information. In this research, the media that will cover the information is an image. The purpose of steganography is to disguise a message so that it looks like it is not a secret message at all [3].

The RSA Algorithm and Least Significant Bit Algorithm method has been widely used to solve various problems such as, insert text to image [4], png image security [5], secure the image file [6], image hiding [7], hiding and data safety [8], secure asymmetric image [9]. In this research, the author proposes a combination of the aforementioned methods with the concept of ECC to strengthen the Key Generation Step of the RSA Algorithm by invoking the ECDLP on generated keys for the RSA Algorithm, by shuffling an ECC seed value using Point Doubling and generating the keys based on the result, thus strengthening the defense against brute-force cracking techniques on the keys.

From the description of the problem and the background above, the author wants to combine the Elliptic Curve Cryptography RSA cryptographic algorithm and the Least Significant Bit steganography algorithm in securing an image file that is inserted in a cover image, so that the information is more secure in the process of sending to the recipient.

2. Method
2.1. Rivest-Shamir-Adleman Algorithm
One of the reasons why the RSA algorithm was chosen is because it encrypts a message using a public key then using a private key to decrypt it, both from the recipient’s side, so there is no need to calculate a secret key as in symmetric cryptography, ensuring the confidentiality, integrity, authenticity, and non-reputability of the data [2].

2.1.1. Key Generation. The key generation process is carried out by the recipient. The steps for generating the RSA algorithm key are as follows:
   1. Generate two prime numbers, namely $p$ and $q$, with the condition $p \neq q$
   2. Calculate the value of $n = p \cdot q$
   3. Calculate $\phi (n) = (p - 1) \cdot (q - 1)$
   4. Choose the public key $e$ randomly, provided that it satisfies:
      • $1 < e < \phi (n)$
      • $\text{GCD} (e, \phi (n)) = 1$
   5. Generate secret key $d$, with the condition $d \cdot e \equiv 1 \pmod {\phi (n)}$

2.1.2. Encryption Process. After the recipient generates the public key, the recipient then sends the key to the sender for message encryption. The process for encrypting the message is as follows:
   1. Determine the message (plaintext) $m$ to be sent and convert it to a decimal number with ASCII encoding.
   2. Receive the recipient's public keys, namely $e$ and $n$.
   3. Calculate $c$ (ciphertext) with the formula:
      $$c = m^e \mod n$$

2.1.3. Decryption Process. The process of decrypting the message is done by the recipient so that the message received is ciphertext converted into its original form (plaintext) using a private key so that it can be read. The decryption process is as follows:
1. Receive c (ciphertext) from the sender of the message.
2. Get the message recipient's secret key, namely d, and n.
3. Decrypt c (ciphertext) to m (plaintext) with the formula:
   \[ m = c^d \mod n \]  

2.2. Elliptic Curve Cryptography

Elliptic Curve Cryptography or ECC is a cryptographic system that uses the characteristics of an elliptic curve that can help build a good cryptographic system [10]. An elliptic curve is a curve that satisfies the following equation:

\[ y^2 = x^3 + ax + b \]  

Many studies on ECC have concluded that the difficulty in solving the Elliptic Curve Discrete Logarithmic Problem increases exponentially with the key size used. This property makes ECC an excellent choice in the encryption and/or decryption process compared to other cryptographic techniques which are only linearly or sub-exponentially difficult. ECC as a public key cryptography system was developed by Neal Koblitz and Victor S. Miller in 1985 [1].

In this study, 2 values called the ECC seeds will be generated, one for each of the 2 prime numbers used in the RSA Algorithm Key Generation Step, and the Point Doubling operation will be applied repeatedly until a fixed amount of times unto the ECC seeds and the resulting value will be rounded up and plugged into the formula \( 6k \pm 1 \) to generate an odd number that will be checked for primality. If the value is proven to be a true prime number, the value will be used as the prime numbers \( p \) and \( q \) in the following RSA Algorithm Key Generation step.

2.3. Least Significant Bit Algorithm

The Least Significant Bit Algorithm is a common and simple approach to embedding confidential information into a cover image. The least significant bit or the eighth bit, in an image, are replaced by the bits of the secret message. When using a 24-bit image, one bit of the red, blue, and green color component can be used, because the colors are represented by a byte. Thus we can store 3 bits in each pixel [11].

3. Results and Discussion

3.1. Implementation of Encryption and Embedding using Elliptic Curve Cryptography RSA Algorithm and LSB Algorithm

By using previously generated keys according to the steps above, we can encrypt and embed every bit of the message image into the cover image with the process as follows.

Encryption and Embedding Process

Example Generated Public Key \((n, e) = (16817, 839)\)

\[ c \text{ (ciphertext)} = m^e \mod n \]

Pixel (0, 0)

\[
\begin{align*}
R &= 7, \quad c = 7^{839} \mod 16817 = 7398 = 00011100 \ 11100110 \\
G &= 180, \quad c = 180^{839} \mod 16817 = 9294 = 00100100 \ 01001110 \\
B &= 242, \quad c = 242^{839} \mod 16817 = 3876 = 00001111 \ 00100100 \\
\end{align*}
\]

Pixel (1, 0)
Every bit of the resulting encrypted values is then inserted to every last bit of the color values in each pixel of the cover image, resulting in a one-byte value of color per pixel from the message being inserted into 16 different one-byte values of color per pixel of the cover image.

3.2. Implementation of Extraction and Decryption using Elliptic Curve Cryptography RSA Algorithm and LSB Algorithm

The value obtained from the extraction of the last bit from every byte of the pixel color value is then recombined into a 16-bit value for every 1 value, which will be decrypted using the private key that has been generated from the previous Key Generation Process Testing.

Extraction and Decryption Process

Example Generated Private Key \((n, d) = (16817, 59)\)

\[ m \text{(message)} = c^d \mod n \]

Pixel (0, 0)

\begin{align*}
R &= 0001100 11100110 = 7398 \\
&= 7398^{59} \mod 16817 = 7
\end{align*}

\begin{align*}
G &= 00000001 01001101 = 9294 \\
&= 9294^{59} \mod 16817 = 180
\end{align*}

\begin{align*}
B &= 00000001 01010100 = 3876 \\
&= 3876^{59} \mod 16817 = 242
\end{align*}

Pixel (1, 0)

\begin{align*}
R &= 00000011 00001111 = 8382 \\
&= 8382^{59} \mod 16817 = 185
\end{align*}

\begin{align*}
G &= 00000010 01110101 = 16297 \\
&= 16297^{59} \mod 16817 = 35
\end{align*}

\begin{align*}
B &= 00000001 10101110 = 1873 \\
&= 1873^{59} \mod 16817 = 215
\end{align*}

3.3. Testing Results of Elliptic Curve Cryptography RSA Algorithm and LSB Algorithm

Using the example keys that have been calculated on the sections before, Table 1 shows the time needed for encrypting, embedding extraction, and decrypting using different sized Messages with the same Cover Image, as follows:

| Message size | Process time of encryption and embedding | Process time of extraction and decryption |
|--------------|-----------------------------------------|-----------------------------------------|
| 200 x 200 pixel | 2103 ms | 1857 ms |
Table 2 shows the processing time for encrypting, embedding, extraction, and decrypting using different sized cover images with the same message, as follows:

**Table 2. Process time of encrypting, embedding extraction, and decrypting using different sized cover images with the same message.**

| Cover image size | Process time of encryption and embedding | Process time of extraction and decryption |
|------------------|-----------------------------------------|-------------------------------------------|
| 1500 x 1500 pixel| 2992 ms                                 | 2793 ms                                   |
| 2000 x 2000 pixel| 3030 ms                                 | 2881 ms                                   |
| 2500 x 2500 pixel| 3268 ms                                 | 3031 ms                                   |

Based on the results shown in Table 1, the time needed to complete the encryption and embedding process increases proportionally with the size of the message. Figure 1. represents the increase in the form of a chart.
Based on the results shown in Table 2, the time needed to complete the extraction and decryption process increases proportionally with Cover Image’s size. Figure 2. Represents the increase in the form of a chart.

Figure 2. Process time using different sized cover images with the same message.

4. Conclusion
Based on the results of the research done and the testing and implementation of the system, the conclusions obtained are as follows:
1. In this study, the designed system can perform encryption, insertion, extraction, and decryption processes, which is concluded from the original message image and the resulting extracted image being the same, using the ECC-RSA algorithm and the LSB algorithm.
2. The running time test results of the system show that the encryption and insertion process takes 3090.167 ms and the extraction and decryption process takes 2917.167 ms.
3. Based on the graph of the Process Time test results on the encryption and insertion as well as extraction and decryption process, shows that time taken for each process is directly proportional to the image file size in the form of pixels, where the larger the pixel size of an image, the longer it will take to secure the image.
5. Acknowledgments
The authors gratefully thank all who have supported the completion of this research. And to the Universitas Sumatera Utara that has allowed me to publish this paper.

Reference
[1] Singh, L. D., and Singh, K. M., 2015. Image Encryption using Elliptic Curve Cryptography. National Institute of Technology, Manipur. India.
[2] Preetha, M., and Nithya, M., 2013. A Study and Performance Analysis of RSA Algorithm. International Journal of Computer Science and Mobile Computing, Vol. 2, Issue. 6, June 2013.
[3] Nisha, S., and Farik, M., 2017. RSA Public Key Cryptography Algorithm – A Review. International Journal of Scientific & Technology Research, Volume 6, Issue 07, July 2017.
[4] R R A Lubis, S M Hardi, M Zarlis, I Jaya and J T Tarigan, 2019. Analysis on Combination of Watermarking Algorithm: Modified Least Significant Bit Algorithm with Least Significant Bit+1. Journal of Physics: Conference Series 1235 012081
[5] S M Hardi, D Rachmawati, F Chairinnisa, I Jaya and J T Tarigan, 2019. Combination of myszkowski transposition algorithm and modified least significant bit (mlsb) green channel on png image security. Journal of Physics: Conference Series 1235 012080.
[6] Dian Rachmawati, Mohammad Andri Budiman and Akhmad Yusuf, 2020, Combination of Rail Fence Cipher Algorithm and Least Significant Bit Technique to Secure the Image File. Journal of Physics: Conference Series 851 012069
[7] D Rachmawati, A Amalia and J Surya, 2017, Combination of Huffman Coding Compression Algorithm and Least Significant Bit Method for Image Hiding. Journal of Physics: Conference Series 801 012059
[8] S M Hardi, M Masitha, M A Budiman and I Jaya, 2020, Hiding and Data Safety Techniques in Bmp Image with LSB and RPrime RSA Algorithm. Journal of Physics: Conference Series 1566 012084
[9] Qiaoyun Xu, Kehui Sun and Congxu Zhu, 2020, A visually secure asymmetric image encryption scheme based on RSA algorithm and hyperchaotic map, Journal of Physics: Conference Series 95 035223.
[10] Agrawal, H., and Badadapure, P. R., 2016. A Survey Paper On Elliptic Curve Cryptography. International Research Journal of Engineering and Technology (IRJET) Volume 03 Issue 04.
[11] Swain Gandharba, 2018. High Capacity Image Steganography Using Modified LSB Substitution and PVD against Pixel Difference Histogram Analysis. Security and Communication Networks, Volume 2018,