OPEN ACCESS
Indo-JC
Indonesia Journal of Computing
ISSN 2460-9056
socj.telkomuniversity.ac.id/indojc

Ind. Journal on Computing
Vol. 5, Issue. 3, December 2020. pp. 1-14
doi:10.34818/indojc.2021.5.3.493

Grid-based Image Encryption using Code-Based Cryptography

Dian Anggoro Putro Bhagaskoro [#1], Ari Moesriami Barmawi [*2]

 # School of Computing, Telkom University
Bandung, Indonesia

[1] putrobhagaskoro@gmail.com

[*] *School of Computing, Telkom University*
Bandung, Indonesia

[2] mbarmawi@melsa.net.id

Abstract

Recently, image is frequently sent or exchanged electronically, such that image security is important. One of the methods for securing images is RSA (Rivest–Shamir–Adleman). However, RSA needs more time for securing an image. For overcoming the RSA problem, McEliece Cryptosystem is introduced to grid-based image encryption. The image is divided into blocks and each block is divided into grids, then finally McEliece Cryptosystem is applied to the pixels in the grids. The experiment is focused on computational time for encrypting and decrypting images. Based on the experiment's result, the computational of encryption time can be reduced about 20-30%, thus it was proven that the execution time of the proposed method is less than the previous one, while maintaining the security.

Keywords: McEliece Cryptosystem, RSA, Image Encryption, Image Decryption, Grid

Abstrak

Dewasa ini, citra sering dikirim atau dipertukarkan secara elektronik, sehingga keamanan citra menjadi penting. Salah satu metode untuk mengamankan citra yaitu dengan mengenkripsi citra menggunakan RSA (Rivest–Shamir–Adleman). Walaupun demikian, RSA membutuhkan lebih banyak waktu untuk melakukan enkripsi dan dekripsi citra. Untuk mengatasi masalah RSA, McEliece *Cryptosystem* diusulkan untuk enkripsi dan dekripsi citra berbasis *grid*. Pada metode ini, citra dibagi menjadi beberapa blok dan setiap blok dibagi menjadi beberapa *grid*, lalu akhirnya McEliece *Cryptosystem* diterapkan untuk mengenkripsi piksel-piksel di dalam *grid* tersebut. Percobaan difokuskan pada waktu komputasi untuk enkripsi dan dekripsi gambar. Berdasarkan hasil percobaan, waktu komputasional untuk enkripsi berkurang sekitar 20-30%, sehingga waktu eksekusi menggunakan metode McEliece lebih kecil bila dibandingkan menggunakan metode sebelumnya, dengan tetap menjaga keamanan.

Kata Kunci: McEliece Cryptosystem, RSA, Enkripsi Gambar, Dekripsi Gambar, Grid

I. Introduction

Nowadays, images are often used to exchange information via the internet. For securing images, encryption is needed such that it cannot be stolen by those who do not participate in data communication. A method used to secure images is grid-based image encryption based on RSA (Rivest–Shamir–Adleman) [1,2]. RSA cryptosystem was introduced in 1978 by R. Rivest, A. Shamir, and L. Adleman [1] by utilizing discrete logarithm problem for securing data. Since RSA requires high computational time for encryption and decryption, then it is very inefficient for encrypting images. For overcoming the problem, McEliece

Received on January, 2020. Accepted on December, 2020

cryptosystem is used for securing image. Based on the experiment's result in 60 images, the average computational time of the encryption can be reduced about 20-30 %, while maintaining the security.

II. Literature Review

This section discusses about RSA cryptosystem and Grid-based Image Encryption using RSA.

A. RSA Cryptosystem

In 1978, R. Rivest, A. Shamir, and L. Adleman developed public key cryptography, whose security lies in the discrete logarithm problem [5]. The idea of RSA is based on the fact that a multiplication of two large prime numbers is difficult to factorize. The public key is a number modulo of two large prime numbers multiplication, while the private key is the inverse of the public key. Therefore, since encryption strength depends on the key size, then when the public key increases, the strength of encryption increases exponentially. For generating RSA keys, the following method is performed.

a. Choosing two different large prime numbers p and q.
b. Calculating $n = pq$. After n is obtained, the value will be used for encryption and decryption.
c. Calculating $\phi(n) = \phi(p).\phi(q) = (p-1)(q-1)$, where ϕ is the Euler function.
d. Choosing integer d, such that $1 < d < \phi(n)$ and $gcd(d, \phi(n)) = 1$, where d is used as the private key of a user for decrypting messages.
e. Calculating $e \equiv d^{-1} \ (mod\ \phi(n))$, where e is a multiplicative inverse of $d \ (mod\ \phi(n))$, such that $d.e \equiv 1 \ (mod\ \phi(n))$. In this case, e is used as the public key of the user.

After generating the keys, the encryption and decryption process can be performed. Encryption is done by calculating

$$\boldsymbol{C \equiv E(M) \equiv M^e \ (mod\ n)} \tag{1}$$

where $\boldsymbol{C}$ is ciphertext, $\boldsymbol{E}(\boldsymbol{M})$ is an encryption function, e is the public key, and **M** is the original message.

After $\boldsymbol{C}$ is received, the recipient decrypts it using the private key d that has been obtained from the key generation. Decryption is done by calculating

$$\boldsymbol{M \equiv D(C) \equiv C^d \ (mod\ n)} \tag{2}$$

where M is the original message, $\boldsymbol{D}(\boldsymbol{C})$ is the decryption function, and d is the private key of the recipient.

B. Grid-based Image Encryption using RSA

In this section, the image encryption using RSA is shown based on [2]. Grid-based image encryption using RSA is illustrated in Figure 1.

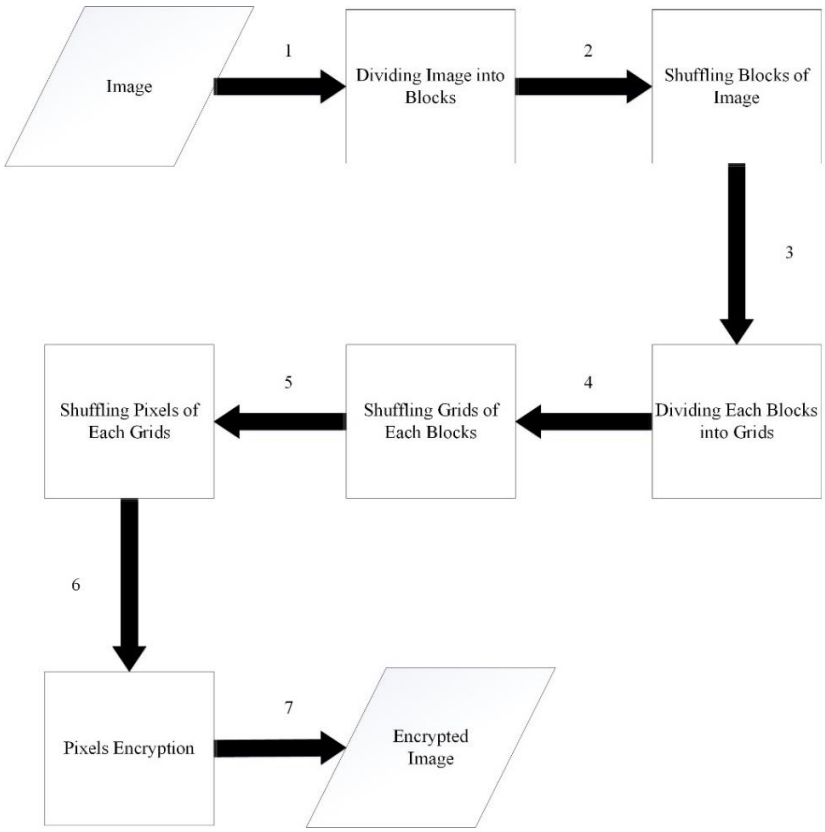

Fig. 1. Grid-based Image Encryption using RSA

Before the encryption process begins, the image is divided into blocks as shown in Figure 2. Then, blocks are swapped using Algorithm 1. Each block has same number of rows and columns. Number of blocks in the image is calculated using Equation (3).

$$frame = \sum_{l=1}^{l=N} \sum_{m=1}^{m=N} B_{lm} \tag{3}$$

where $frame$ = number of blocks, l = block row index, m = block column index, B_{lm} = a block in row l and column m, N = number of block row index and column index.

After the image is divided into blocks, each block is divided into several grids as shown in Figure 2 and the position of all the grids in each block are swapped using Algorithm 1. Number of grids in a block is calculated using Equation (4).

$$B_{ij} = \sum_{n=1}^{n=M} \sum_{r=1}^{r=M} G_{nr} \tag{4}$$

where B_{ij} = number of grids of block with row's index i and column's index j, n = grid row index, r = grid column index, G_{nr} = a grid in row n and column r of the block, M = number of grid's rows and columns.

After the block is divided into several grids, each pixel in each grid are swapped using Algorithm 1. Number of pixels in a grid is calculated using Equation (5).

$$G_{ij} = \sum_{x=1}^{x=K} \sum_{y=1}^{y=K} p_{xy} \quad (5)$$

where G_{ij} = number of pixels of grid with row's index i and column's index j, x = pixel row index, y = pixel column index, p_{xy} = pixel in row x and column y of the grid, K = number of pixel's rows and columns.

After the pixels are swapped, each pixel is encrypted using RSA cryptosystem based on Algorithm 1. The image division is given in Figure 2.

Algorithm 1 Grid-based Image Encryption using RSA

Input: Original Image, m, n, Public Key e.

1: // Divide original image into a number of blocks B with size of m x m;
2: // Indexing blocks with block row index and block column index, corresponding with Equation (3);
3: // In every block, divide block into a number of grids G with size of n x n;
4: // Indexing grids with grid row index and grid column index in every block, corresponding with Equation (4);
5: // Indexing pixels with pixel row index and pixel column index in every grid, corresponding with Equation (5);
6: for each block do
7: swapBlock $B_{00} \rightarrow B_{JJ}$; $B_{01} \rightarrow B_{J-1J}$; and etc.;
for every block B_{pq}, where $p = 0..J$ and $q = 0..J$;
8: for each grid do
9: swapGrid $G_{00} \rightarrow G_{KK}$; $G_{01} \rightarrow G_{K-1K}$; and etc.;
for every grid G_{ab}, where $a = 0..K$ dan $b = 0..K$;
10: for each pixel do
11: swapPixel $P_{00} \rightarrow P_{LL}$; $P_{01} \rightarrow P_{L-1L}$; and etc.;
for every P_{xy}, where $x = 0..L$ and $y = 0..L$;
end for
end for
end for
12: $P_{xy} \rightarrow P'_{xy}$; // Encrypt every pixel using RSA cryptosystem.
13: Group all P_{xy}.
14: Group all G_{ab}.
15: Group all B_{pq}.

Output: Encrypted Image

Note:
P_{xy} = original image pixel, P'_{xy} = encrypted image pixel, J = block size, K = grid size, and L = pixel's index in a grid.

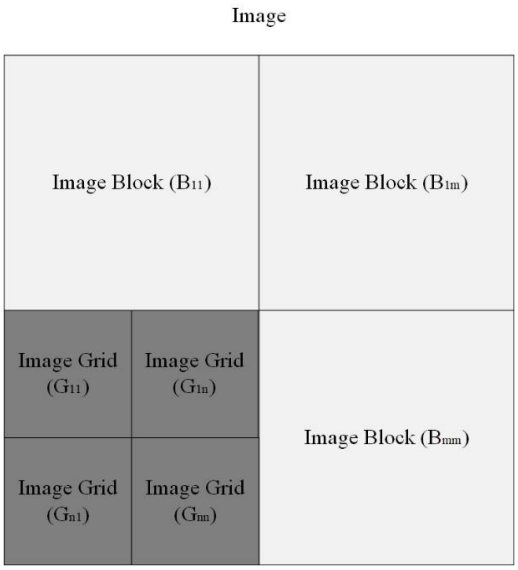

Fig. 2. Image Division into Blocks and Grids

C. Grid-based Image Decryption using RSA

Decryption of image using RSA is the opposite of the encryption process and is illustrated in Figure 3.

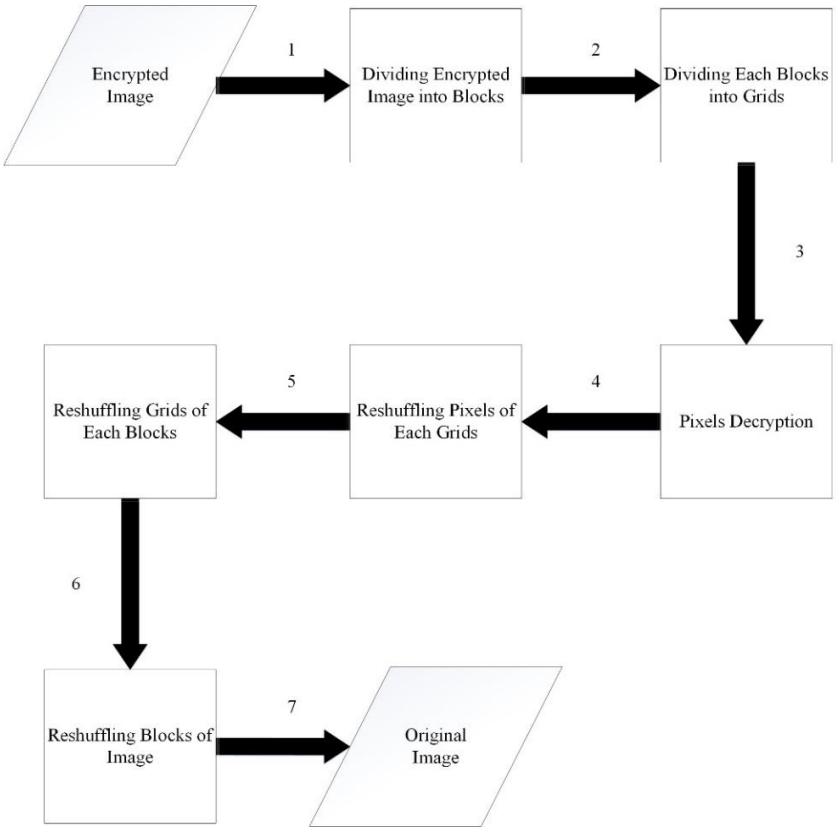

Fig. 3. Grid-based Image Decryption using RSA

In the decryption process, each pixel is decrypted into the original pixel. After each pixel is decrypted, each pixel on each grid is returned to its original order and depicted in Equation (5). Then, each grid in each block is returned to the original order and depicted in Equation (4). Finally, each block in the image is returned to its original order and grouped into original image and depicted in Equation (3). The algorithm used for grouping the image into the original image is shown in Algorithm 2.

Algorithm 2 Grid-based Image Decryption using RSA

Input: Encrypted Image, Private Key d.

1: $P'_{xy} \rightarrow P_{xy}$*; // Decrypt every pixel using RSA cryptosystem.*
2: *// Divide original image into a number of blocks B with size of m x m;*
3: *// Indexing blocks with block row index and block column index, corresponding with Equation (3);*
4: *// In every block, divide block into a number of grids G with size of n x n;*
5: *// Indexing grids with grid row index and grid column index in every block, corresponding with Equation (4);*
6: *// Indexing pixels with pixel row index and pixel column index in every grid, corresponding with Equation (5);*
7: *for each pixel in a grid do*
8: *swapPixel* $P_{LL} \rightarrow P_{00}; P_{L-1L} \rightarrow P_{01}$*; and etc.;*
for every P_{xy}*, where* $x = 0..L$ *and* $y = 0..L$*;*
9: *for each grid in a block do*
10: *swapGrid* $G_{KK} \rightarrow G_{00}; G_{K-1K} \rightarrow G_{01}$*; and etc.;*
for every grid G_{ab}*, where* $a = 0..K$ *dan* $b = 0..K$*;*
11: *for each block do*
12: *swapBlock* $B_{JJ} \rightarrow B_{00}; B_{J-1J} \rightarrow B_{01}$*; and etc.;*
for every block B_{pq}*, where* $p = 0..J$ *and* $q = 0..J$*;*
end for
end for
end for
13: *Group all* P_{xy}*.*
14: *Group all* G_{ab}*.*
15: *Group all* B_{pq}*.*

Output: Decrypted Image

Note:
P'_{xy} *= encrypted image pixel and* P_{xy} *= decrypted image pixel,* J *= block size,* K *= grid size, and* L *= pixel's index in a grid.*

III. Research Method

This section discussed about the method for encrypting and decrypting grid-based images using McEliece Cryptosystem, a public key cryptographic system whose security was based on error correction code [4,6,8]. McEliece Cryptosystem is introduced because it is resistant against quantum attack and it has lower time complexity than RSA. Error correction code is used to correct the errors of message received in the receiver, such that he/she receives the original messages. Since McEliece uses Hamming code for correcting the errors, then additional bits are added to the message before transmission. These additional bits is further used for obtaining the original message.

McEliece cryptosystem is based on selecting specific codes that have an efficient decoding algorithm such that the decryption process becomes faster. In addition for having an efficient decoding algorithm, Hamming code can also corrects all one-weight errors added to the code such that the key determination process is faster.

McEilece cryptosystem maintains confidentiality in three ways, hiding the private key, multiplying the matrix generator for hamming code with an arbitrary matrix that has an inverse and permutation matrix, and adding an error vector to the message to be sent.

Before encrypting and decrypting messages, key generation is performed to determine public and private keys. To generate public and private key in the McEliece cryptosystem, the following steps need to be carried out.

a. Choosing a binary linear code C (n, k) with fast decoding algorithm that able to correct t errors and generate a k x n matrix generator G for code C.
b. Choosing a random k x k non-singular binary matrix S.
c. Choosing a random n x n permutation matrix P, using $randperm()$ function in Matlab.
d. Calculating the matrix $\hat{G}$ with size of k x n, by calculating $\hat{G} = SGP$. In this case, $(\hat{G}, t)$ is the public key and (S, G, P) is the private key.

Modeling explanation is divided into two parts, because the encryption and decryption process has different paths.

A. Image Encryption

Image encryption consists of several steps, which are dividing images into several blocks, shuffling all blocks, dividing each block into several grids, shuffling all grids on each block, and encrypting pixels in each grid. The overview of encryption process is described in Figure 4.

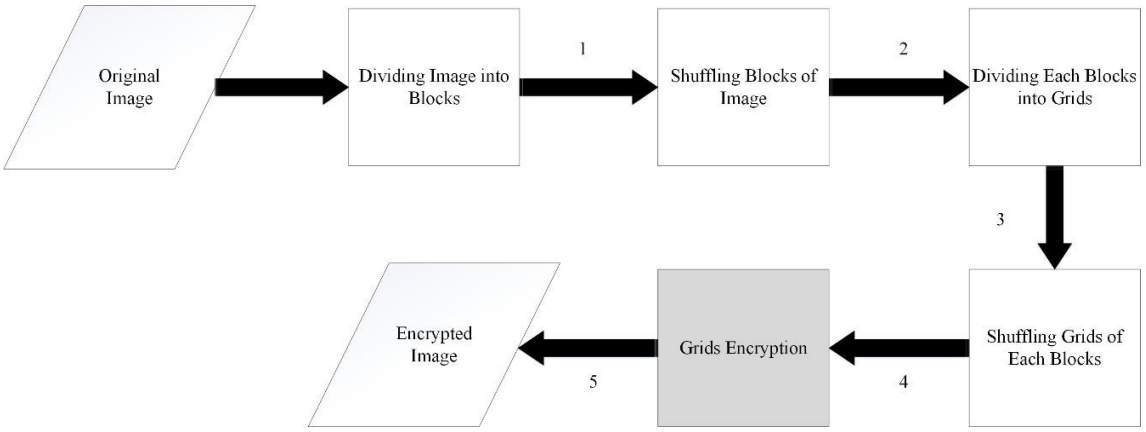

Fig. 4. Proposed Encryption Process

1) Dividing Image into Blocks:

Before the encryption process begins, the image has to be divided into blocks. Number of blocks in the image is calculated using Equation (3).

2) Shuffling Blocks of the Image:

After image is divided into blocks, each block are swapped using Algorithm 3.

3) Dividing Each Blocks into Grids:

Each blocks of image is divided into several grids. Number of grids in a block is calculated using Equation (4).

4) Shuffling Grids of Each Blocks:

Each grid in a block are swapped using Algorithm 3.

5) Pixel Encryption:

After all grids in each block are shuffled, every pixel of each grids is encrypted using McEliece cryptosystem [7]. The encryption process is represented by $G_{nr} \rightarrow G'_{nr}$, where G_{nr} are pixels in grid with row's index n and column's index r before encrypted, while G'_{nr} are pixels in grid with row's index n and column's index r after encrypted. Encryption is done as follows:

a. Encoding the pixel p in G_{nr} into a binary row of length k. Encoding algorithm is described in Algorithm 4.
b. Calculating the vector $c' = p\hat{G}$, where $\hat{G}$ is the public key and $\hat{G} = S * G * P$.
c. Generating a random n-bit vector e consists of t of "1" which can correct errors up to t [8], where t is called as the weight.
d. Calculating the secret message $c = c' + e$.
e. Converting c into decimal number b.
f. Calculating $enc = b \ mod \ 256$, where enc is remainder for encrypted pixel with range of $0 - 255$.
g. Calculating q with div function to find the quotient, such that $q = \lfloor enc/256 \rfloor$.
h. Keeping q for decrypting the encrypted pixel.

Example of the encryption result is shown in Figure 5.

Fig. 5. Encryption Example

Algorithm 3 Grid-based Image Encryption using McEliece

Input: Original Image, l, m, n, r

1: *// Indexing blocks with block row index and block column index, corresponding with Equation (3);*
2: *for* $i = 1 \dots l$
    $j = l - i + 1$;
3: *for* $k = 1 \dots m$
        $swap(B_{ik}, B_{jk})$;
    endfor
endfor
4: *// Indexing grids with grid row index and grid column index in every block, corresponding with Equation (4);*
5: *for* $a = 1 \dots l$
6: *for* $b = 1 \dots m$
7: *for* $i = 1 \dots n$
            $j = n - i + 1$;
8: *for* $k = 1 \dots r$
                $swap(B_{ab}.G_{ik}, B_{ab}.G_{jk})$;
            endfor
        endfor
    endfor
endfor

9: *for* $x = 1 \dots w$
10: *for* $y = 1 \dots z$
$p_{xy} \rightarrow p'_{xy}$; *// Encrypt pixel using McEliece cryptosystem.*
endfor
endfor
11: *Group all* G_{nr}.
12: *Group all* B_{lm}.

Output: Encrypted Image

Note:
p_{xy} = *original image pixels,* p'_{xy} = *encrypted image pixels, and* w x z = *image size.*

Algorithm 4 Pixel Encryption Algorithm using McEliece Cryptosystem

Input: Image Pixel p, Public Key (S, G, P), t

1: $binaryPixel \leftarrow decimalToBinaryFunction(p)$; *//converting image pixel into binary digit and encoding process.*
2: $e = errorVectorGenerator(t)$; *// generatine error vector with weight of t*
3: $publicKey = S * G * P$;
4: $cipher \leftarrow binaryPixel * publicKey$; *// encryption process*
5: $c \leftarrow cipher \oplus e$; *// encrypted pixel*
6: $b \leftarrow binaryToDecimalFunction(c)$;
7: $enc \leftarrow b \bmod 256$;
8: $q \leftarrow floor\,(b/256)$;

Output: Encrypted Pixel enc, q.

B. Image Decryption

Image decryption is the opposite of the encryption process and consists several steps, which are dividing the encrypted image into blocks, dividing each block into grids, decrypting all grids, reshuffling the grid order into the initial grid, merging all grids into several blocks, reshuffling the order of blocks into the initial block, and merging all blocks into original images. The overview of the method for decryption is described in Figure 5.

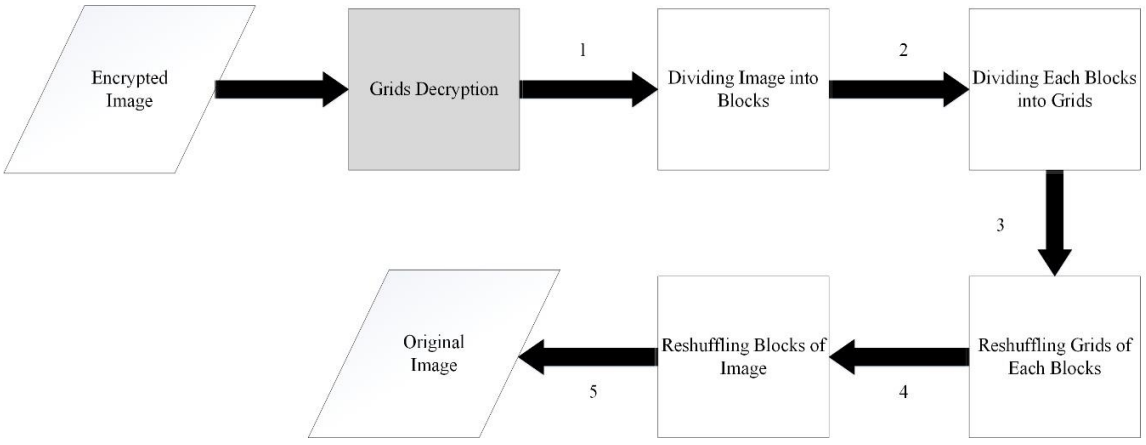

Fig. 6. Proposed Decryption Process

1) Grids Decryption:

In the decryption process, every pixel of each grid is decrypted into the original grid using McEliece Cryptosystem [7] as represented by $G'_{nr} \rightarrow G_{nr}$, where G'_{nr} is an encrypted pixel in grid with row's index n and column's index r, while G_{nr} is a decrypted pixel in grid with row's index n and column's index r. Decryption is done as follows:

a. Calculating P^{-1}, the inverse of P, where P is random permutation matrix.
b. Calculating $b = q * 256 + enc$.
c. Convert b into binary digit c.
d. Calculating $\hat{c} = c(P^{-1})$.
e. Using hamming decoding algorithm of code C for the decoding process $\hat{c}$ into $\hat{p}$, such that the message is returned into 1 x k matrix. Decoding algorithm is described in Algorithm 5.
f. Calculating $p = \hat{p}S^{-1}$ for each pixel.
g. Convert p into decimal number for determining decrypted pixel.

Algorithm 5 Pixel Decryption Algorithm using McEliece Cryptosystem

Input: Encrypted Pixel enc, q, Private key (S, G, P)

1: $b \leftarrow q * 256 + enc$;
2: $c \leftarrow decimalToBinaryFunction(b)$;
3: $\hat{c} \leftarrow cP^{-1}$;
4: $\hat{e} \leftarrow eP^{-1}$; *// the inverse of e, where e is error vector.*
5: $SXG \leftarrow \hat{c} \oplus \hat{e}$; *// $SXG = S * G * P * P^{-1}$*
// Because G was of the form $[I_k\ A]$, $\hat{p}$ is just the first k positions of SXG, where I_k is k x k matrix identity and A is $(n-k)\ x\ k$ random matrix, and no multiplication is needed.
6: $\hat{p} \leftarrow decodingFunction(SXG)$;
7: $p \leftarrow \hat{p} * S^{-1}$;
8: $p \leftarrow binaryToDecimalFunction(p)$;

Output: Decrypted Pixel p

2) Dividing Image into Blocks:

After every pixel in a each grid is decrypted, image is divided into blocks using Algorithm 6.

3) Dividing Blocks into Grids:

After an image is decrypted, every blocks of the image is divided into grids using Algorithm 6.

4) Reshuffling Grids of a Block:

All grids in each block are returned to the original order using Algorithm 6 and number of grids in a block is calculated using Equation (4).

5) Recovering Blocks of Image:

After reshuffling grids of every block, each blocks of image is returned to its original order until the original image is recovered. This step is conducted using Algorithm 6 and number of blocks is calculated using Equation (3).

Algorithm 6 Grid-based Image Decryption using McEliece

Input: Encrypted Image, l, m, n, r

1: *for* $x = 1 \ldots w$
2: *for* $y = 1 \ldots z$
$p'_{xy} \rightarrow p_{xy}$; *// Decrypt pixel using McEliece cryptosystem.*
endfor

endfor
3: *// Indexing blocks with block row index and block column index, corresponding with Equation (3);*
4: *// Indexing grids with grid row index and grid column index in every block, corresponding with Equation (4);*
5: *for* $a = 1 \dots l$
6: *for* $b = 1 \dots m$
7: *for* $i = 1 \dots n$
8: $j = n - i + 1$;
9: *for* $k = 1 \dots r$
$swap(B_{ab}.G_{ik}, B_{ab}.G_{jk})$;
endfor
endfor
endfor
endfor
10: *for* $i = 1 \dots l$
$j = l - i + 1$;
11: *for* $k = 1 \dots m$
$swap(B_{ik}, B_{jk})$;
endfor
endfor
12: *Group all* G_{nr}.
13: *Group all* B_{lm}.

Output: Decrypted Image

Note:
p_{xy} = *decrypted image pixels,* p'_{xy} = *encrypted image pixels, and* w x z = *image size.*

IV. RESULTS AND DISCUSSION

This section describes the results of experiments, including the analysis of the result. The analysis of the result consists of the security evaluation and computational time analysis.

A. Security Evaluation

Security evaluation is conducting by comparing the entropy between images encrypted with RSA and McEliece method. For evaluating the security, the following process are conducted.

a. Selecting 60 images that have image resolution of 256 x 256 pixels from [3]. Tested images consists of 20 images with bell-shaped (center-skewed) color intensity histogram, 20 images with left-skewed histogram, and 20 images with right-skewed histogram.
b. Applying the encryption and decryption using RSA and McEliece cryptosystem on all images.
c. Calculating the entropy of the encrypted images using RSA and McEliece cryptosystem.

For analyzing the experiment result, the entropy has to be calculated and further evaluation. Entropy is a statistical measure of randomness that can be used to characterize the texture of an input image [11]. The entropy of an image is calculated using Equation (6) and (7).

$$E = -\sum_{i=1}^{n} P_i \,.\log_2 P_i \tag{6}$$

$$P_i = P_r(X = x_i) \tag{7}$$

The higher the entropy value, the more random the distribution of pixels in the image. The results is shown in table 1.

TABLE I
ENTROPY COMPARISON BETWEEN RSA AND MCELIECE ENCRYPTED IMAGE

Test	Bell-shaped Histogram Images		Left-skewed Histogram Images		Right-skewed Histogram Images	
	Entropy of RSA-encrypted Image	Entropy of McEliece-encrypted Image	Entropy of RSA-encrypted Image	Entropy of McEliece-encrypted Image	Entropy of RSA-encrypted Image	Entropy of McEliece-encrypted Image
1	7.662186214	7.658244767	7.391185792	7.391640658	7.708000145	7.680355628
2	7.644267871	7.653226402	7.574471175	7.550625513	7.695223968	7.660607348
3	7.655853245	7.643524623	7.51389138	7.510952413	7.665841545	7.662210642
4	7.653692536	7.651857585	7.676174491	7.675037229	7.666666119	7.662516779
5	7.64970474	7.668616861	6.450444627	6.478110811	7.718910159	7.654911161
6	7.63732843	7.654377948	7.218846992	7.161630753	7.662688068	7.654022643
7	7.646772769	7.666690144	6.333839203	6.366715538	7.687958715	7.664287308
8	7.651460795	7.660194243	7.59951911	7.622172789	7.647835365	7.650975879
9	7.621813343	7.625915734	7.633588735	7.657610162	7.677583035	7.657442572
10	7.62932175	7.665971526	7.670633626	7.659481858	7.679105952	7.662945223
11	7.677932705	7.672656368	7.636250228	7.637214958	7.660508586	7.655283774
12	7.665354671	7.668904745	7.657258904	7.696886483	7.58449224	7.520573591
13	7.646022941	7.661391468	6.823411081	6.857829193	7.704038972	7.67243764
14	7.673953804	7.659593749	7.625178761	7.645264799	7.663228989	7.54067421
15	7.664590749	7.686986811	7.606028772	7.638253014	6.306964438	6.315947201
16	7.698651484	7.697089956	7.645055218	7.654588444	7.616980431	7.625854866
17	7.664448706	7.644124894	7.592971865	7.48835994	7.57927555	7.581028108
18	7.644853903	7.649234959	7.578990266	7.58618716	7.654849136	7.598180161
19	7.654123337	7.659405942	7.140353136	7.220990578	7.30599161	7.044975051
20	7.582879188	7.567228483	7.634588344	7.64234748	7.624063655	7.587909919

Based on Table 1, the image entropy value using McEliece is overall similar with using RSA, such that the pixel randomness level of RSA-encrypted images is similar to the McEliece-encrypted images. However, for right-skewed histogram image, average of pixel randomness using the proposed method is less than RSA-encrypted images. This condition is occurred because RSA encryption uses exponential function and the pixel value of right-skewed histogram image large, such that the distribution range of RSA-based encrypted pixel value is broader than McEliece-based encrypted pixel value distribution range. In this case, the randomness of RSA-based encrypted pixel is higher than McEliece-based encrypted pixel value.

B. Execution Time Analysis

This section discusses the execution time evaluation when using RSA and McEliece Cryptosystem. For analyzing execution time, the following experiments are conducted.

a. Selecting 60 images that have image with resolution of 256 x 256 pixels from [3]. Images consist of 20 images with bell-shaped histogram, 20 images with left-skewed histogram, and 20 images with right-skewed histogram.
b. Applying the encryption and decryption using RSA cryptosystem and McEliece cryptosystem on all images.
c. Calculating the execution time for encryption and decryption process.
d. Comparing the execution time when using RSA cryptosystem and McEliece cryptosystem.

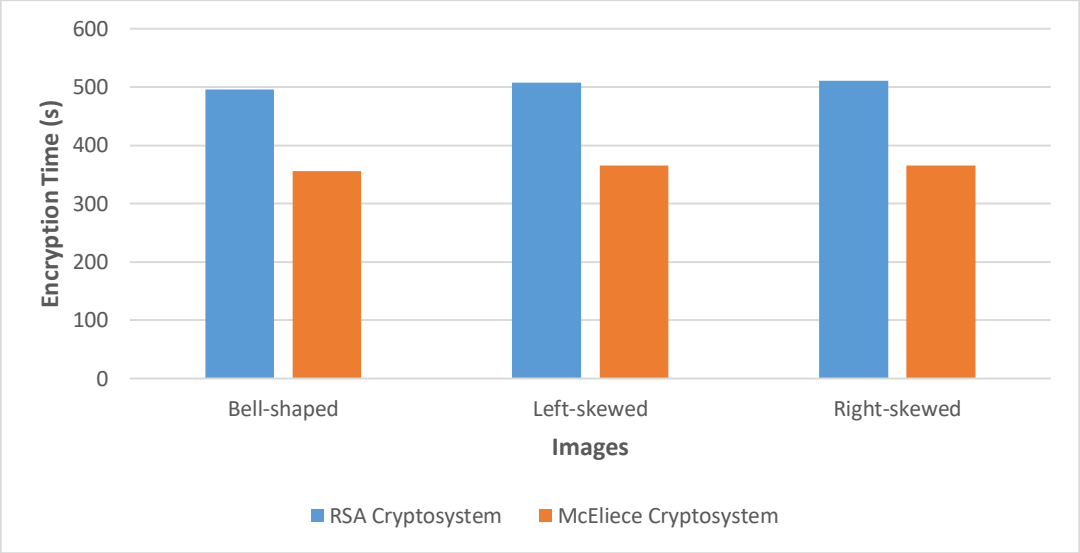

Fig. 7. Average Time of Encryption Process

Based on Figure 7, it can be concluded that the time required for encrypting image when using RSA cryptosystem is longer than the encryption process using the McEliece cryptosystem. This condition is occurred, because RSA encryption requires exponential multiplication and modulo operation for calculating the value of encrypted pixel, while McEliece only requires matrix multiplication, where matrix multiplication needs less time complexity.

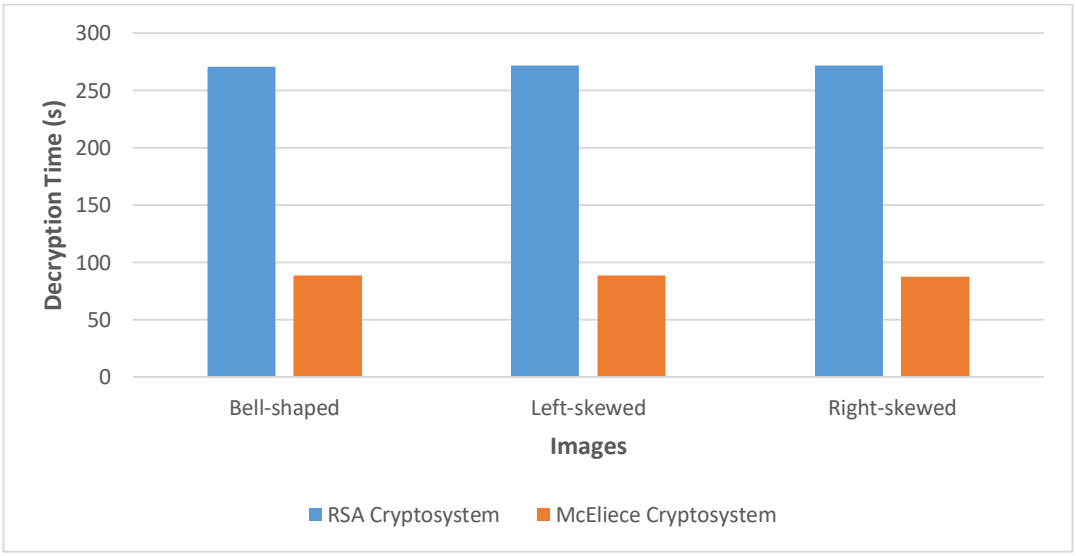

Fig. 8. Average Time of Decryption Process

Based on Figure 8, it can be concluded that the time required for decrypting image when using RSA cryptosystem requires a longer time than the decryption process with the McEliece cryptosystem. This condition is occurred, because RSA decryption requires exponential multiplication and modulo operation for calculating the value of original pixel, while McEliece only requires matrix multiplication, where matrix multiplication needs less time complexity.

V. Conclusion

Based on the results of the analysis, it can be concluded that grid-based image encryption using RSA Cryptosystem requires more time to perform encryption and decryption than McEliece Cryptosystem, while the security is maintained. It is proven since the randomness pixels encrypted by RSA and McEliece are overall similar, which means that the security of grid-based image encryption using RSA and McEliece cryptosystem are similar. However, the randomness pixels encrypted by RSA is less than using McEliece cryptosystem for bell-shaped and left-skewed histogram image while for right-skewed histogram image, the randomness level when using RSA is greater than using McEliece.

Acknowledgement

This research was supported by School of Computing Telkom University. We thank our colleagues from Telkom University who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

References

[1] El-Deen, A., El-Badawy, E. and Gobran, S., 2014. Digital image encryption based on RSA algorithm. J. Electron. Commun. Eng, 9(1), pp.69-73.
[2] Singh, B.K. and Gupta, S.K., 2015. Grid-based Image Encryption using RSA. International Journal of Computer Applications, 115(1).
[3] Chaladze, G. Kalatozishvili L. 2017. Linnaeus 5 Dataset for Machine Learning.
[4] Dambra, A., Gaborit, P., Roussellet, M., Schrek, J. and Tafforeau, N., 2014. Improved Secure Implementation of McEliece Signature Schemes on Embedded Devices. IACR Cryptology ePrint Archive, 2014, p.163.
[5] Rivest, R.L., Shamir, A. and Adleman, L., 1978. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2), pp.120-126.
[6] Mceliece, R.J., 1978. A public-key cryptosystem based on algebraic. Coding Thv, 4244, pp.114-116
[7] Zhao, G., Yang, X., Zhou, B. and Wei, W., 2010, July. RSA-based digital image encryption algorithm in wireless sensor networks. In 2010 2nd International Conference on Signal Processing Systems (Vol. 2, pp. V2-640). IEEE.
[8] Stinson, D.R. and Paterson, M., 2018. Cryptography: Theory and Practice. CRC Press.
[9] Wu, Y., Zhou, Y., Saveriades, G., Agaian, S., Noonan, J.P. and Natarajan, P., 2013. Local Shannon entropy measure with statistical tests for image randomness. Information Sciences, 222, pp.323-342.
[10] Katz, J., Menezes, A.J., Van Oorschot, P.C. and Vanstone, S.A., 1996. Handbook of applied cryptography. CRC press.
[11] Gonzalez, R. C., R. E. Woods, and S. L. Eddins. Digital Image Processing Using MATLAB. New Jersey, Prentice Hall, 2003, Chapter 11.