talenta PUBLISHER

DATA SCIENCE
Journal of Computing and Applied Informatics

Data Security Using Multi-bit LSB and Modified Vernam Cipher

G T Simbolon[1], Opim Salim Sitompul[2], E B Nababan[3]

[1]*Graduate School of Computer Science*
[2,3]*Department of Information Technology, Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan, Indonesia*

Abstract. Data security is one of the most important aspects of today's information era. Some methods are used to secure important data from hackers. The LSB is a steganographic algorithm that is often used to store data in the last bit. In order to improve the security, we combine steganography with cryptography enables. In this research LSB is modified using the multi-bit LSB model. Modifications are made to the bits of each character, the rotation by a certain amount can randomize the plaintext content before cryptographic algorithm, Vernam is performed. The bit on LSB can be inserted data as much as 1, 2, 3 or 4 - bit information. The calculation results of MSE and PSNR values indicate that the use of 1-bit LSB is superior to that of 2-, 3-, or 4-bit LSB.

Keyword: Cryptography, LSB, Steganographic, Vernam Encryption

Abstrak. *Keamanan data adalah salah satu aspek terpenting dari era informasi saat ini. Beberapa metode digunakan untuk mengamankan data penting dari peretas. LSB adalah algoritma steganografi yang sering digunakan untuk menyimpan data dalam bit terakhir. Untuk meningkatkan keamanan, kami menggabungkan steganografi dengan memungkinkan kriptografi. Dalam penelitian ini LSB dimodifikasi menggunakan model LSB multi-bit. Modifikasi dilakukan pada bit masing-masing karakter, rotasi dengan jumlah tertentu dapat mengacak konten plaintext sebelum algoritma kriptografi, Vernam dilakukan. Bit pada LSB dapat dimasukkan data sebanyak 1, 2, 3 atau 4 - bit informasi. Hasil perhitungan nilai MSE dan PSNR menunjukkan bahwa penggunaan LSB 1-bit lebih unggul daripada LSB 2-, 3-, atau 4-bit.*

Kata Kunci: *Cryptography, LSB, Steganographic, Vernam Encryption*

Received 24 May 2019 | Revised 12 July 2019 | Accepted 12 July 2019

1. Introduction

According to the Cisco Visual Networking Index (VNI), the total IP traffic that occurred in 2016 is expected to reach 91.3 exabytes per month. Given this amount of traffic, data security is becoming increasingly important, particularly for sensitive data, such as company data and state security information [1]. Digital information is indirectly transmitted through a data network via

*Corresponding author at:Department of Information Technology, Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan, Indonesia

E-mail address: ernabrn@usu.ac.id

ISSN: 2580-6769 | e-ISSN: 2580-829X | DOI: 10.32734/jocai.v3.i2-1048
Journal Homepage: https://jocai.usu.ac.id

a small electric current that is used as a link to analog signals [2]. Data often comprise important information that must be properly secured to prevent theft by certain parties [3]–[5].

Steganography and cryptography are two methods of securing data. Steganography conceals data, whereas cryptography secures data by encoding the plaintext. Acts of crime are becoming more advanced, and consequently, these methods often fail. Various types of data theft are performed to solve well-encrypted ciphertexts; hence, the power of steganography and cryptography must be increased through various methods.

The least significant bit (LSB) steganography technique functions by replacing the rightmost bit of each pixel according to the message bit. The vulnerability of LSB is the storage of plaintext bits in a row, which allows intruders to easily extract bits of confidential information. Herein, three color elements are alternately used from the red, green, and blue layers. The aim of this study is to improve data security by combining steganography and cryptography techniques.

The Vernam algorithm encrypts data by performing exclusive-OR (XOR) operations on each plaintext character [6], [7] in a given set of data. This algorithm is modified by rotating bits. After obtaining the cryptographic results, the ciphertext is hidden via the multi-bit LSB steganography technique on 24-bit imagery using several storage bit models. Steganography encrypts each bit of information into RGB color elements [9]–[12], and multi-bits can be used to increase security by creating variations of the LSB technique [12]–[15]. Each sample is tested for the mean square error (MSE) and peak signal-to-noise ratio (PSNR) values from stego-images, which are used to determine the most superior image with the smallest error rate; this superior image is used as the storage image. In this manner, the above described hybrid technique can be used to improve data security.

2. Cryptographic and Steganography Algorithms

2.1 Cryptographic Algorithm: Vernam Algorithm

The Vernam cipher is an algorithm based on the principle that each character in the plaintext is encrypted using the XOR process against a certain generated key [16], [17]. The provided key is generated repeatedly or extended to many plaintext lengths. In the Vernam cipher algorithm, a symmetric key type signifies that the same key is used for both encryption and decryption. The Vernam cipher’s vulnerability is the use of XOR for plaintext encryption and decryption because it is very simple. The following is an example of message encryption performed using the Vernam algorithm.

Encryption process:

```
Plaintext: “algorithm”
Row of bits:
```

```
01100001 01101100 01100111 01101111 01110010 01101001 01110100 01101000 011011
01

Key:
10111001 01110100 00111010 00010101 01101010 11010100 10101010 10001110 101010
10
--------------------------------------------------------------------------XOR

Ciphertext:
11011000 00011000 01011101 01111010 00011000 10111101 11011110 11100110 110001
11
```

Decryption process:

```
Ciphertext:
11011000 00011000 01011101 01111010 00011000 10111101 11011110 11100110 110001
11

Key:
10111001 01110100 00111010 00010101 01101010 11010100 10101010 10001110 101010
10
-------------------------------------------------------------------------XOR

Row of bits:
01100001 01101100 01100111 01101111 01110010 01101001 01110100 01101000 011011
01

Plaintext: “algorithm”
```

2.2 Steganography Least Significant Bit (LSB)

Steganography, which is a Greek term that means “closed writing,” is the technique of communicating by hiding the existence of messages. Steganography plays an important role in information security, and a steganography system comprises three elements: the cover image, secret message, and stego-image. Digital images are represented using X and Y coordinates that contain three color elements in each pixel. Typically, a gray image uses 8 bits, whereas a color image uses 24 bits to describe the RGB color model.

Several techniques exist for hiding information in the cover image. Spatial domain techniques manipulate pixel bit values to embed confidential information, and message bits are directly encoded into the pixel bits of the cover image. Thus, spatial domain techniques can be easily implemented [18].

LSB is the position of a bit in a binary integer that provides a unit value to determine whether a number is even or odd. Bit numbering is performed from 0 to 7 to represent the intensity value of a color. The LSB is the smallest or rightmost bit. The LSBs can quickly change if the amount changes slightly. For example, if a 1-bit LSB changes (00000000) to (00000001), the result is a 1-bit jump, and if a 2-bit LSB changes (0000000) to (00000011), the result is a 3-bit jump. If a 3-

bit LSB changes (00000000) to (00000111), the result is a 7-bit jump, whereas if a 4-bit LSB changes (00000000) to (00001111), the result is a 15-bit jump.

2.3 MSE and PSNR Measurement

Fidelity is a security element in the practice of hiding secret messages. The measurement of fidelity steganography can be performed by calculating the values of the MSE and PSNR. The PSNR is usually measured in decibels (dB). To determine the PSNR, first, the average value of the square of the MSE error must be determined. If the MSE value is small, a large PSNR value is produced, and vice versa. Images with a PSNR value ≥40 dB are considered high-quality images [18].

The MSE value can be calculated using the following formula:

$$\boldsymbol{MSE = \sum_{i=1}^{M} \sum_{j=1}^{N} (f(i,j) - g(i,j))^2 \,/\, M * N}, \qquad (1)$$

Where:

MSE = MSE value from image

M = Length of image (in pixels)

N = width of image (in pixels)

f (i, j) = Value of the pixel coordinates of the image before the message is inserted

.After calculating the MSE value, the PSNR value can be calculated as follows

$$\boldsymbol{PSNR = 10 * Log_{10} \left(\frac{C_{max}^2}{MSE}\right)}, \qquad (2)$$

Where:

PSNR = PSNR value of the image, and

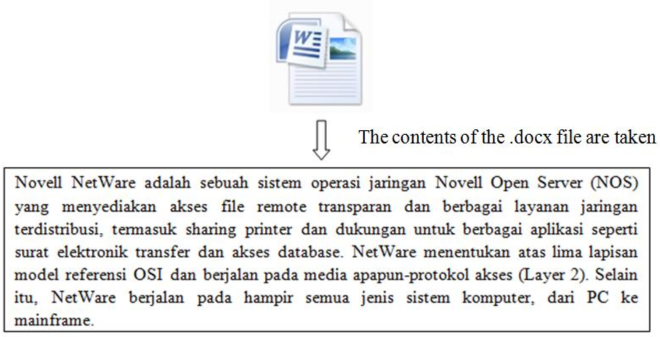
C_{max} = the maximum pixel value.

For RGB color images with values in each pixel, the MSE value is calculated for each color, and then all such values are summed and divided by 3 [19].

3. Methodology

In this section, after analyzing the two abovementioned data security techniques— the modified Vernam cipher cryptography and LSB multi-bit steganography techniques—the stages of system design are discussed as follows.

3.1 Data Input



The contents of the .docx file are taken

Novell NetWare adalah sebuah sistem operasi jaringan Novell Open Server (NOS) yang menyediakan akses file remote transparan dan berbagai layanan jaringan terdistribusi, termasuk sharing printer dan dukungan untuk berbagai aplikasi seperti surat elektronik transfer dan akses database. NetWare menentukan atas lima lapisan model referensi OSI dan berjalan pada media apapun-protokol akses (Layer 2). Selain itu, NetWare berjalan pada hampir semua jenis sistem komputer, dari PC ke mainframe.

Figure 1. Retrieval of *.docx* Content

Input in the form of a.docx file is limited to characters that can be changed in ASCII code, however the system cannot encrypt tables, images, or symbols. As illustrated in Fig. 1,.docx content refers to the contents of a.docx file in the form of words. The key input is also limited to a–z and A–Z characters.

3.2 Vernam Chiper Modification

This section discusses the modification of the Vernam cipher algorithm through bit rotation. Before the encryption process is completed, bit rotates the character and key bit positions to the left and right, respectively, and then performs the XOR operation. One character contains 8 bits, and bit rotation is performed from the 1st bit to the 8th bit (this can be adjusted as needed). Because one character contains 8 bits, if the rotation is performed more than eight times, the bit position repeats; i.e., 9-bit rotation is the same as 1-bit rotation, whereas 0-bit rotation is the same as 8-bit rotation.

During decryption, the first step is the XOR operation between the ciphertext and the key, which is performed via right-bit rotation. The XOR results are then rotated to the right for the message to return to its original sequence (plaintext).

1) Process of modifying the Vernam algorithm for encryption

Plaintext:

P0	P1	P2	P3	P4

Bit plaintext:

B07	B06	B05	B04	B03	B02	B01	B00	→	P0
B17	B16	B15	B14	B13	B12	B11	B10	→	P1
B27	B26	B25	B24	B23	B22	B21	B20	→	P2
B37	B36	B35	B34	B33	B32	B31	B30	→	P3
B47	B46	B45	B44	B43	B42	B41	B40	→	P4

1-bit rotation to left:

B06	B05	B04	B03	B02	B01	B00	B07	→	P0
B16	B15	B14	B13	B12	B11	B10	B17	→	P1
B26	B25	B24	B23	B22	B21	B20	B27	→	P2
B36	B35	B34	B33	B32	B31	B30	B37	→	P3
B46	B45	B44	B43	B42	B41	B40	B47	→	P4

Key:

K0	K1	K2	K3	K4

Bit key:

K07	K06	K05	K04	K03	K02	K01	K00	→	K0
K17	K16	K15	K14	K13	K12	K11	K10	→	K1
K27	K26	K25	K24	K23	K22	K21	K20	→	K2
K37	K36	K35	K34	K33	K32	K31	K30	→	K3
K47	K46	K45	K44	K43	K42	K41	K40	→	K4

1-bit rotation to the right:

K00	K07	K06	K05	K04	K03	K02	K01	→	K0
K10	K17	K16	K15	K14	K13	K12	K11	→	K1
K20	K27	K26	K25	K24	K23	K22	K21	→	K2
K30	K37	K36	K35	K34	K33	K32	K31	→	K3
K40	K47	K46	K45	K44	K43	K42	K41	→	K4

Ciphertext / XOR Operation:

B06⊕K00	B05⊕K07	B04⊕K06	B03⊕K05	B02⊕K04	B01⊕K03	B00⊕K02	B07⊕K01	→	C0
B16⊕K10	B15⊕K17	B14⊕K16	B13⊕K15	B12⊕K14	B11⊕K13	B10⊕K12	B17⊕K11	→	C1
B26⊕K20	B25⊕K27	B24⊕K26	B23⊕K25	B22⊕K24	B22⊕K23	B20⊕K22	B27⊕K21	→	C2

B36⊕K30	B35⊕K37	B34⊕K36	B33⊕K35	B32⊕K34	B31⊕K33	B30⊕K32	B37⊕K31	→	C3
B46⊕K40	B45⊕K47	B44⊕K46	B43⊕K45	B42⊕K44	B41⊕K43	B40⊕K42	B47⊕K41	→	C4

2) Process of modifying the Vernam algorithm for decryption

Ciphertext:

C0	C1	C2	C3	C4

Bit ciphertext:

C07	C06	C05	C04	C03	C02	C01	C00	→	C0
C17	C16	C15	C14	C13	C12	C11	C10	→	C1
C27	C26	C25	C24	C23	C22	C21	C20	→	C2
C37	C36	C35	C34	C33	C32	C31	C30	→	C3
C47	C46	C45	C44	C43	C42	C41	C40	→	C4

Key:

K0	K1	K2	K3	K4

Bit Key:

K07	K06	K05	K04	K03	K02	K01	K00	→	K0
K17	K16	K15	K14	K13	K12	K11	K10	→	K1
K27	K26	K25	K24	K23	K22	K21	K20	→	K2
K37	K36	K35	K34	K33	K32	K31	K30	→	K3
K47	K46	K45	K44	K43	K42	K41	K40	→	K4

1-bit rotation to Right:

K00	K07	K06	K05	K04	K03	K02	K01	→	K0
K10	K17	K16	K15	K14	K13	K12	K11	→	K1
K20	K27	K26	K25	K24	K23	K22	K21	→	K2
K30	K37	K36	K35	K34	K33	K32	K31	→	K3
K40	K47	K46	K45	K44	K43	K42	K41	→	K4

Plaintext / XOR Operation:

C07⊕K00	C06⊕K07	C05⊕K06	C04⊕K05	C03⊕K04	C02⊕K03	C01⊕K02	C00⊕K01	→	P0
C17⊕K10	C16⊕K17	C15⊕K16	C14⊕K15	C13⊕K14	C12⊕K13	C11⊕K12	C10⊕K11	→	P1
C27⊕K20	C26⊕K27	C25⊕K26	C24⊕K25	C23⊕K24	C22⊕K23	C21⊕K22	C20⊕K21	→	P2
C37⊕K30	C36⊕K37	C35⊕K36	C34⊕K35	C33⊕K34	C32⊕K33	C31⊕K32	C30⊕K31	→	P3
C47⊕K40	C46⊕K47	C45⊕K46	C44⊕K45	C43⊕K44	C42⊕K43	C41⊕K42	C40⊕K41	→	P4

XOR result bits:

B07	B06	B05	B04	B03	B02	B01	B00	→	P0
B17	B16	B15	B14	B13	B12	B11	B10	→	P1
B27	B26	B25	B24	B23	B22	B21	B20	→	P2
B37	B36	B35	B34	B33	B32	B31	B30	→	P3
B47	B46	B45	B44	B43	B42	B41	B40	→	P4

Plaintext 1-bit Rotation to Right:

B00	B07	B06	B05	B04	B03	B02	B01	→	P0
B10	B17	B16	B15	B14	B13	B12	B11	→	P1
B20	B27	B26	B25	B24	B23	B22	B21	→	P2
B30	B37	B36	B35	B34	B33	B32	B31	→	P3
B40	B47	B46	B45	B44	B43	B42	B41	→	P4

3.3 Multi-bit LSB

Multi-bit LSB is performed using several bit insertion models. The steps performed in the multi-bit LSB process for inserting messages, in the form of three elements of color (RGB), in specified bits within each pixel are as follows.

Inserting the character as 1-bit LSB storage:

R	7	6	5	4	3	2	1	0
G	7	6	5	4	3	2	1	0
B	7	6	5	4	3	2	1	0
R	7	6	5	4	3	2	1	0
G	7	6	5	4	3	2	1	0
B	7	6	5	4	3	2	1	0
R	7	6	5	4	3	2	1	0
G	7	6	5	4	3	2	1	0

Inserting the character as 2-bits LSB storage:

R	7	6	5	4	3	2	1	0
G	7	6	5	4	3	2	1	0
B	7	6	5	4	3	2	1	0
R	7	6	5	4	3	2	1	0

Inserting the character as 3-bits LSB storage:

R	7	6	5	4	3	2	1	0
G	7	6	5	4	3	2	1	0
B	7	6	5	4	3	2	1	0

Inserting the character as 4-bits LSB storage:

R	7	6	5	4	3	2	1	0
G	7	6	5	4	4	2	1	0

The four patterns presented above store up to one character (i.e., 8 bits). The more are the number of bits used, the fewer are the pixels used for information storage: 1-bit insertion uses less than 3 pixels; 2-bit insertion uses less than 2 pixels; 3-bit insertion uses exactly 1 pixel; and 4-bit insertion uses less than 1 pixel. In addition, unused color elements can be used to save the next character.

3.4 Calculation of MSE and PSNR

Herein, a 24-bit color image was used for the experiment. The MSE and PSNR values were calculated to determine the extent to which an image changes post message insertion. Storage from 1-bit up to 4-bit LSBs was performed, and the MSE and PSNR values were calculated for each stego-image to determine which image is superior and how many bits should be used to store information or messages. The following formula was used to calculate the MSE and PSNR values. Using (2.2) and (2.3)

Example:

Table 1. RGB Pixels

Red			Green			Blue		
R1	...	Rn	G1	...	Gn	B1	...	Bn

$$MSE = \frac{1}{M*N}\left(\frac{(R'_1 - R_1)^2 + \cdots + (R'_n - R_n)^2 + (G'_1 - G_1)^2 + \cdots + (G'_n - G_n)^2 + (B'_1 - B_1)^2 + \cdots + (B'_n - B_n)^2}{3}\right)$$

$$PSNR = 10\ \log_{10}\left(\frac{C_{max}^2}{MSE}\right)$$

4. Result and Discussion

An example of an image used herein is one containing 4 × 4 pixels, which can accommodate up to 48 bits. If converted to characters, the image can accommodate up to 6 characters for the use of 1 bit. The following are the steps for 1-bit plaintext rotation and the 1-bit key.

Plaintext : Secret

83	101	99	114	101	116
01010011	01100101	01100011	01110010	01100101	01110100

Rotation : 1-bit to the left

10100110	11001010	11000110	11100100	11001010	11101000
166	202	198	228	202	232

Key : World

87	111	114	108	100
01010111	01101111	01110010	01101100	01100100

Rotation : 1-bit to the right

10101011	10110111	00111001	00110110	00110010
171	183	57	54	50

Ciphertext : -}ÿÊ°C

13	125	255	210	248	67

The results of the ciphertext are stored in an image using several bit storage models, such as that illustrated below. In the test conducted herein, a 4 × 4-pixel image was used as a cover image; this could accommodate 4 × 4 × 3 bits = 48 bits (i.e., six characters) using 1-bit LSB.

RGB bit for the 4 × 4 pixel cover image:

Red	Green	Blue

65	58	187	190	87	176	149	241	116	249	167	229
222	171	203	63	223	74	221	68	150	134	191	182
246	101	32	89	205	204	143	53	128	84	243	163
192	151	36	11	125	130	167	60	226	101	153	23

RGB binary for a 4 x 4 pixel cover image:

Red	Green	Blue
01000001 00111010 10111011 10111110	01010111 10110000 10010101 01111001	01110100 11111001 10100111 11100101
11011110 10101011 11001011 00111111	11011111 01001010 11011101 01000100	10010110 10000110 10111111 10110110
11110110 01100101 00100000 01011001	11001101 11001100 10001111 00110101	10000000 01010100 11110011 10100011
11000000 10010111 00100100 00001011	01111101 10000010 10100111 00111100	11100010 01100101 10011001 00010111

Ciphertext bit to be inserted: -}ÿÊ°C

00001101	01111101	11111111	11010010	11111000	01000011

Stego-image results obtained using 1-bit LSB:

Red				Green				Blue			
64	58	**186**	**191**	**86**	**177**	149	241	116	249	**166**	229
223	171	203	63	223	**75**	221	**69**	150	**135**	191	**183**
247	101	**33**	89	205	204	**142**	53	128	84	243	163
193	**150**	36	**10**	**124**	130	**166**	**61**	226	101	**152**	23

$$\text{MSE} = \frac{1}{16}(7)$$
$$= 0{,}4375$$
$$\text{PSNR} = 10 * Log10\left(\frac{249^2}{0{,}4375}\right)$$
$$= 51{,}5142$$

Stego-image results obtained using 2-bit LSB:

Red				Green				Blue			
64	**57**	187	**191**	**84**	**177**	149	**243**	**119**	**251**	167	**231**
223	**170**	**202**	**60**	**221**	**75**	**220**	68	**148**	**135**	**189**	**183**
246	101	32	89	205	204	143	53	128	84	243	163
192	151	36	11	125	130	167	60	226	101	153	23

$$\text{MSE} = \frac{1}{16}(10{,}67)$$

$$= 0{,}6667$$

$$\text{PSNR} = 10 * Log10\left(\frac{249^2}{0{,}6667}\right)$$

$$= 49{,}6846$$

Stego-image results obtained using 3-bit LSB:

Red				Green				Blue			
64	**63**	**191**	**188**	**83**	**182**	**151**	**246**	**113**	**255**	**166**	**231**
220	171	203	63	**218**	74	221	68	**144**	134	191	182
246	101	32	89	205	204	143	53	128	84	243	163
192	151	36	11	125	130	167	60	226	101	153	23

$$\text{MSE} = \frac{1}{16}(18)$$

$$= 1{,}125$$

$$\text{PSNR} = 10 * Log10\left(\frac{249^2}{1{,}125}\right)$$

$$= 47{,}4142$$

Stego-image results obtained using 4-bit LSB:

Red				Green				Blue			
64	**61**	**189**	**184**	**93**	**191**	**146**	**244**	**119**	**255**	**175**	**227**
222	171	203	63	223	74	221	68	150	134	191	182
246	101	32	89	205	204	143	53	128	84	243	163
192	151	36	11	125	130	167	60	226	101	153	23

$$\text{MSE} = \frac{1}{16}(19{,}333)$$

$$= 1{,}2083$$

$$\text{PSNR} = 10 * Log10\left(\frac{249^2}{1{,}2083}\right)$$

$$= 47{,}1022$$

Bold and underlined pixel bits are pixel bits that have been inserted into the message. In 1-bit LSB storage more image pixels are used because each RGB color element can only store 1-bit.

For storing 4-bit LSB fewer pixels are used because each RGB color element can store 4-bit ciphertex.

5. Conclusion

The results of this study indicate that the level of data security can be increased by combining steganography methods with cryptography algorithms. The resolution of the cover image and character length of the encrypted message considerably affect the MSE and PSNR values. The calculation results of MSE and PSNR values indicate that the use of 1-bit LSB is superior to that of 2-, 3-, or 4-bit LSB. However, according to the results, the use of 4-bit LSB is feasible because the PSNR value for 4-bit LSB is above 40 db. The smaller percentage of pixel usage in 4-bit LSB storage provides satisfactory MSE and PSNR values. The application of the multi-bit LSB method in steganographic activities is thus advantageous, in the sense that each pixel can hold more message bits compared with the conventional LSB method.

REFERENCES

[1] Cisco Visual Networking Index. Forecast and Methodology. *cisco.com*, http://www.cisco.com/c/en/us/solutions/collateral/ service-provider/ip-ngn-ip-next-generation-network/white_ paper_c11-481360.html. [Accessed: 30-Mar-2018]

[2] S. Aryza, M. Irwanto, Z. Lubis, A. P. U. Siahaan, R. Rahim, and M. Furqan, “A Novelty Design of Minimization of Electrical Losses in A Vector Controlled Induction Machine Drive,” in *IOP Conference Series: Materials Science and Engineering*, , vol. 300, no. 1. 2018

[3] R. Rahim *et al.*, “Combination Base64 Algorithm and EOF Technique for Steganography,” *J. Phys. Conf. Ser.*, vol. 1007, no. 1, pp. 1–5, 2018.

[4] R. Meiyanti, A. Subandi, N. Fuqara, M. A. Budiman, and A. P. U. Siahaan, “The Recognition of Female Voice Based on Voice Registers in Singing Techniques in Real-Time using Hankel Transform Method and Macdonald Function,” *J. Phys. Conf. Ser.*, vol. 978, no. 1, pp. 1–6, 2018.

[5] A. P. U. Siahaan and R. Rahim, “Dynamic Key Matrix of Hill Cipher Using Genetic Algorithm,” *Int. J. Secur. Its Appl.*, vol. 10, no. 8, pp. 173–180, Aug. 2016.

[6] A. P. U. Siahaan, “Securing Short Message ServiceUsing Vernam Cipher in Android Operating System,” *IOSR*, Apr. 2016.

[7] W. Fitriani, R. Rahim, B. Oktaviana, and A. P. U. Siahaan, “Vernam Encypted Text in End of File Hiding Steganography Technique,” *Int. J. Recent Trends Eng. Res.*, vol. 3, no. 7, pp. 214–219, Jul. 2017.

[8] A. P. U. Siahaan, “Noise-Like Region Security Improvisation in BPCS Steganography.”

[9] A. P. U. Siahaan, “High Complexity Bit-Plane Security Enchancement in BPCS Steganography,” *Int. J. Comput. Appl.*, vol. 148, no. 3, pp. 17–22, 2016.

[10] H. Nurdiyanto and R. Rahim, “Enhanced pixel value differencing steganography with

government standard algorithm," in *2017 3rd International Conference on Science in Information Technology (ICSITech),* , pp. 366–371. 2017

[11] A. P. U. Siahaan, "RC4 Technique in Visual Cryptography RGB Image Encryption."

[12] B. Datta, P. K. Pal, and S. K. Bandyopadhyay, "Multi-bit Data Hiding in Randomly Chosen LSB Layers of an Audio," in *2016 International Conference on Information Technology (ICIT),* , pp. 283–287. 2016

[13] M. Kaur and M. Juneja, "A new LSB embedding for 24-bit pixel using multi-layered bitwise XOR," in *2016 International Conference on Inventive Computation Technologies (ICICT),* , pp. 1–5. 2016

[14] R.-J. Chen, Y.-C. Chen, Jui-Linlai, and S.-J. Horng, "Data Hiding Using Flexible Multi-bit MER," in *2013 International Symposium on Biometrics and Security Technologies*, , pp. 24–31. 2013

[15] S. Goel, S. Gupta, and N. Kaushik, "Image Steganography – Least Significant Bit with Multiple Progressions,", pp. 105–112. 2015

[16] "The Vernam Cipher," *Crypto Museum*. [Online]. Available: http://www.cryptomuseum.com/crypto/vernam.htm.

[17] A. P. U. Siahaan, "Vernam Conjugated Manipulation of Bit-Plane Complexity Segmentation."

[18] C. . B. S., P. K., and R. D. K., "Least Significant Bit Algorithm for Image Steganography," *Int. J. Adv. Comput. Technol.*, vol. 3, no. 4, pp. 34–38, 2014.

[19] "Bit Numbering," *Wikipedia*. [Online]. Available: https://en.wikipedia.org/wiki/Bit_numbering. [Accessed: 15-Jun-2018].