Secure Information Hiding Based on Random Similar Bit Mapping

Abdul Alif Zakaria and Norli Anida Abdullah

***Abstract*—The goal of cryptography is to maintain the secrecy of information while steganography aims to hide the information. A hybrid steganography and cryptography method was introduced to increase the security of data transmission. Random Similar Bit Mapping (RSBM) was proposed to hide a secret message without modifying the image and generates a Position File (PF) which stores the positions of a hidden message. PF is encrypted using Advanced Encryption Standard (AES) algorithm before being sent to the receiver. Two security measures were proposed to estimate the message location in PF based on Correct Position Finding (CPF) for message detection against a brute force attack. This paper compared related works using the proposed security measures to evaluate its security. From the CPF probability analysis, RSBM produced the lowest CPF probability results, while recording the highest CPF time complexity results in CPF time complexity analysis. In conclusion, RSBM ensured high data security which can be implemented in any information hiding application.**

***Index Terms*—Cryptography, information hiding, probability analysis, steganography, time complexity analysis.**

I. Introduction

The objective of security is to protect against those who may harm intentionally or unintentionally [1]. Security can be seen in many organizations but this research prioritizes communication and information security. Communication security protects technology, media, and content. Meanwhile, information security protects the confidentiality, integrity, and availability.

Steganography defines as covered writing that aims to keep the existence of a hidden message from others [2]. Statistical method can detect the message existence [3]. Steganography is compared to cryptography due to their similarity in securing message but in different ways. Cryptography keeps the secrecy of the message but not its existence [4]. A known ciphertext paves the way for an attacker to apply cryptanalysis to gain the message [5].

We have identified a research gap from the literature. Steganography introduced image modifications that could arouse suspicion about the existed message. If attackers aware of the existence, steganography is considered broken [6]. The research motivation is to avoid introducing suspicion as occurred in existing methods.

Cryptography and steganography have been used together in communication [7], [8]. Usually, the message is encrypted before being hidden in an image. This paper introduced a method that hides a message before encrypting it. Random Similar Bit Mapping (RSBM) was designed to hide a message without modifying the image and uses AES to encrypt a generated Position File (PF).

A way to break the security of steganography is by obtaining the key using brute force [9]. This paper compared the security of the methods by deploying Correct Position Finding (CPF) probability analysis and CPF time complexity analysis against brute force attack.

This paper is divided into five sections. Section II describes related works of steganography and cryptography implementation in digital images. Section III discusses the proposed method named RSBM. Section IV presents the experimental setup and results. Finally, Section V discusses the conclusion of the research work.

II. Related Works

This paper focused on the combination of steganography and cryptography methods that do not modify the image and generates a *PF*. *PF* increases its security because it requires an extra credential to gain the message. Due to limited literature over the methods, we narrow the research scope to the following methods.

Static Parsing Steganography (SPS) [10] is based on parsing the image instead of modifying it. Divide-and-conquer strategy is the fundamental of SPS that finds the longest common substring from the message and image files. The process is repeated until all of the message bits matched to bits of the image. The message locations found in the image are stored in *PF* and encrypted.

Multiple Cover Objects (MCO) [11] hides the message using the Least Significant Bit algorithm (LSB) [12], [13]. MCO able to hide the message in more than one image. MCO determines the message bits that matched the image bits and stores the location in *PF*, then encrypts it.

Closest Pixel-pair Mapping (CPPM) [14] finds the closest pixel value of the message in the image. CPPM creates a reference hash table (RHT) that contains the pixel positions and differences. RHT maps the message pixel position to the image. Each pixel value position that was stored in RHT is entered in Memorization Lookup Table (MLT) that operates

Manuscript received November 15, 2019; revised March 2, 2020. This work was supported by CyberSecurity Malaysia.

A. A. Zakaria is with the Department of Cryptography Development, CyberSecurity Malaysia, 63000 Selangor, Malaysia (e-mail: alif@cybersecurity.my).

N. A. Abdullah is with the Centre for Foundation Studies in Sciences, University of Malaya, 50603 Kuala Lumpur, Malaysia (e-mail: norlie@um.edu.my).

doi: 10.18178/ijmlc.2020.10.4.974

like *PF* is then encrypted.

III. Proposed Work

Steganography using reference data that are processed together with its image to extract the message is discussed in this section. Fig. 1 displays the reference data which is called *PF* works as an additional key apart from the stego key. This paper proposed a method to structure *PF* in which the message is mapped with the image without introducing distortion which is called the RSBM.

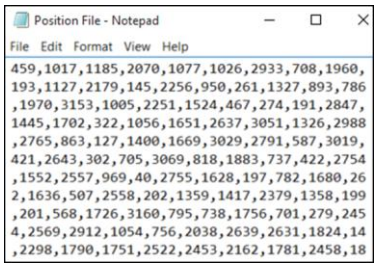

Fig. 1. Position file.

A. Method Architecture

Fig. 2 displays components needed which are the stego key, secret message, and image file to hide a message. The key and image are shared with the receiver but the message is kept secret. Firstly, the message is hidden in an image then generated a *PF*. The stego key randomized the message distribution. Lastly, AES generated a ciphertext by encrypting the *PF* with the stego key.

Three files required by the receiver are the stego key, image, and ciphertext. However, the sender does not need to send the image to the receiver if both parties agreed to use a publicly available image. The receiver must have the same image used by the sender to hide the message.

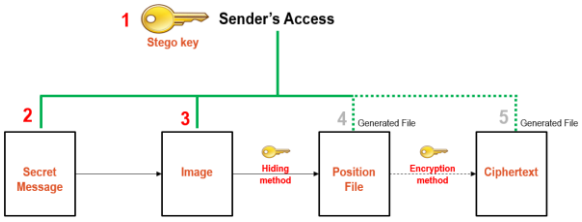

Fig. 2. Hiding process.

The extraction process requires the stego key, ciphertext, and image files. By using the AES and key, the ciphertext is decrypted to access the *PF*. If an incorrect key is used, the receiver is not able to use the *PF* in the next step. Secondly, the *PF* and key are used to identify the message locations in the image. Fig. 3 displays the structure of the message extraction.

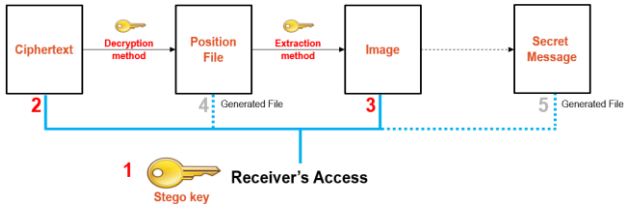

Fig. 3. Extraction process.

B. Method Architecture

The algorithm consists of seven steps which are scan, separate, store, generate, permute, index, and permute position. Fig. 4-Fig. 7 displays the hiding algorithm steps. Firstly, all bits contained in the image and intended message are scanned and recorded as shown in Fig. 4.

1) Scan:

a) Total bit "1"s and "0"s in the image are recorded.

Bit ? ?= ?and Bit ? ?= ?

b) Total bit "1"s and "0"s in the message are recorded.

Bit ? ?= 0 and Bit ? ?= 0?

MSB	Image						LSB
1	1	0	0	1	0	0	0
0	0	1	1	1	0	1	1

(bit "0" = 8 & bit "1" = 8)

MSB	Secret Message						LSB
1	1	1	0	0	0	1	0
1	0	1	0	0	1	0	0
0	1	0	1	0	1	1	0
1	0	0	1	1	1	0	1
0	1	1	0	1	0	1	0

(bit "0" = 20 & bit "1" = 20)

Fig. 4. Scan process.

Bit "1"s and "0"s of the image and message files are separated and the bit positions are stored as in Fig. 5.

2) Separate:

a) Bits "0" in the image and secret message are separated from bits "1" of both files.

b) Bits "1" in the image and secret message are separated from bits "0" of both files.

3) Store:

a) Each position of the bits "0" in image and secret message are stored.

Bits "0" positions in image:

$3,4,6,7,8,9,10,14$.

Bits "0" positions in secret message:

$4,5,6,8,10,12,13,15,16,17,19,21,24,26,27,31,33,36,38,40$.

b) Each position of the bits "1" in image and secret message are stored.

Bits "1" positions in image:

$1,2,5,11,12,13,15,16$.

Bits "1" positions in secret message:

$1,2,3,7,9,11,14,18,20,22,23,25,28,29,30,32,34,35,37,39$.

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Bit "0"	Position																			
Image	3	4	6	7	8	9	10	14												
Secret Message	4	5	6	8	10	12	13	15	16	17	19	21	24	26	27	31	33	36	38	40

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Bit "1"	Position																			
Image	1	2	5	11	12	13	15	16												
Secret Message	1	2	3	7	9	11	14	18	20	22	23	25	28	29	30	32	34	35	37	39

Fig. 5. Separate and store.

The position of each bit "1"s and "0"s is permutated using a Pseudorandom Number Generator (PRNG) and stego key as the seed. The new positions are stored in *PF* as shown in Fig. 6.

4) Generate:

a) n_0 numbers are generated. (n_0 = no. of bits "0" in Secret Message).

b) n_1 numbers are generated. (n_1 = no. of bits "1" in Secret Message).

5) Permute:

a) The position of each bit "0"s are permutated and the new positions are stored.

b) The position of each bit "1"s are permutated and the new positions are stored.

Message bits are stored in similar bits (bit "1" to "1" or bit "0" to "0") of the image. These new positions of the message b is called Indexed Position as shown in Fig. 7.

6) Index: New positions of bit "0"s and "1"s are indexed accordingly and stored. To protect the Indexed Position, stego key is used during the extracting process. The final permutation values are called the *PF* as shown in Fig. 8.

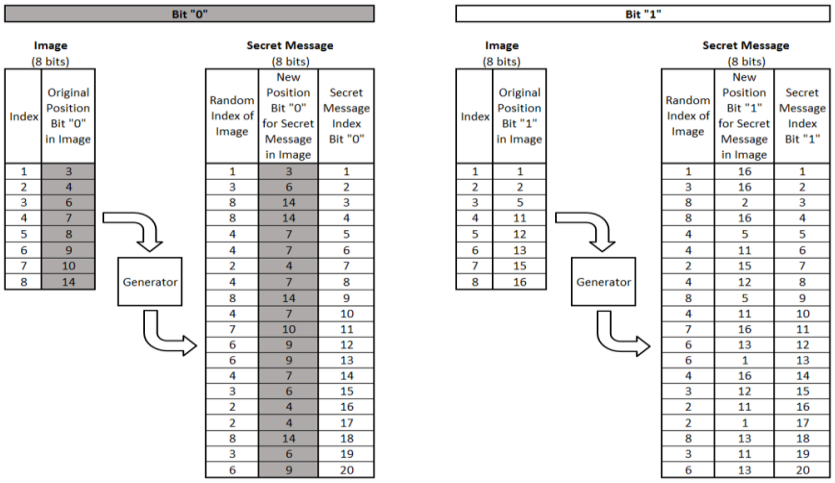

Fig. 6. Generate and permute.

Secret Message	1	1	1	0	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0	1
Secret Message Index Bit "0"				1	2	3		4		5		6	7		8	9	10		11	
Secret Message Index Bit "1"	1	2	3				4		5		6			7				8		9
New Position of Secret Message in Image	16	16	2	3	6	14	16	14	5	7	11	7	4	15	7	14	7	12	10	5

Secret Message	0	1	1	0	1	0	0	1	1	1	0	1	0	1	1	0	1	0	1	0
Secret Message Index Bit "0"	12			13		14	15				16		17			18		19		20
Secret Message Index Bit "1"		10	11		12			13	14	15		16		17	18		19		20	
New Position of Secret Message in Image	9	11	16	9	13	7	6	1	16	12	4	11	4	1	13	14	11	6	13	9

Fig. 7. Indexed position.

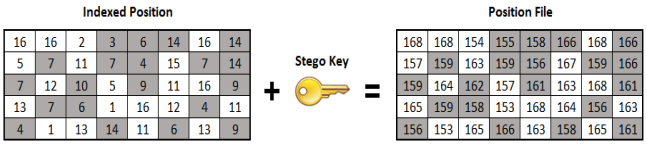

Fig. 8. Position file creation (hiding process).

7) Permute Position:

a) Value of stego key is computed using ASCII value.

e.g.: Stego key = *Alif 2s*

$$A=65, l=108, i=105, f=102, 2=50, s=115$$

b) The value is summed up using the formula below.

$$Sum=\lfloor(65*1)+(108*2)+(105*3)+(102*4)+$$

$$(50*5)+(115*6)\rfloor \bmod(256)$$

$$=152$$

c) *PF* values are assigned as $y_1, y_2, \ldots, y_n$. (n = no. of bits in Secret Message)

d) Indexed Position values are assigned as $x_1, x_2, \ldots, x_n$.

e) The Indexed Position values are added with the *Sum* to obtain the *PF* values.

$$(y_1, y_2, \ldots, y_n)=[(x_1, x_2, \ldots, x_n)+Sum]?$$

$$=\lfloor(16,16,2,3,6,\ldots)+152\rfloor$$

$$=[168,168,154,155,158,\ldots]$$

The generated *PF* stores the locations of the message in the image. Stego key, image, and *PF* are required in the process of secret message extraction. These three files required to be sent to the receiver using a secure channel to protect it from an unintended receiver.

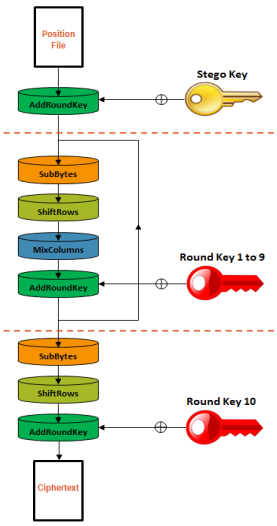

Fig. 9. AES encryption.

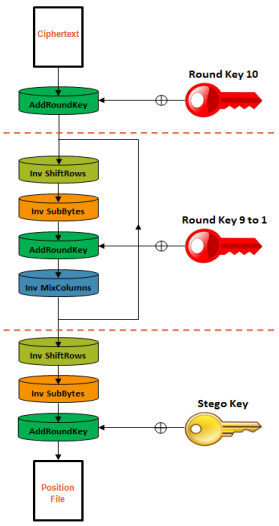

Fig. 10. AES decryption.

C. Encryption and Decryption Algorithm

AES is used for encryption and decryption. It operates in ten rounds which are divided into nine main rounds and a final round [15]. Fig. 9 shows the processes in AES which are Sub Bytes, Shift Rows, Mix Columns, and Add Round Key. The final output is called the ciphertext. To decrypt the ciphertext, the AES operates in reverse order of the encryption process as shown in Fig. 10.

D. Extraction Algorithm

The secret message extraction process requires the receiver to have access to the stego image and *PF*. *PF* is a

dummy file that aims to confuse the unintended receiver. The exact hidden message positions are stored in the Indexed Position. Fig. 11 shows the stego key is used to authenticate the receiver to obtain the Indexed Position.

1) Permute Position:

a) Value of stego key is computed using ASCII value
e.g.: Stego key = *Alif* 2*s*

$$A=65, l=108, i=105, f=102, 2=50, s=115$$

b) The value is summed up using formula below.

$$Sum = \lfloor (65*1)+(108*2)+(105*3)+(102*4)+ (50*5)+(115*6) \rfloor \bmod (256) = 152$$

c) Indexed Position values are assigned as $x_1, x_2, \ldots, x_n$.

d) *PF* values are assigned as $y_1, y_2, \ldots, y_n$.

e) The PF values are subtracted with the *Sum* to obtain the Indexed Position.

$$(x_1, x_2, \ldots, x_n) = \lfloor (y_1, y_2, \ldots, y_n) - Sum \rfloor ?$$
$$= \lfloor (168,168,154,155,158,\ldots) - 152 \rfloor ?$$
$$= [16,16,2,3,6,\ldots]$$

Fig. 11. Indexed position creation (extraction process).

Both Indexed Position and image files are required during this process. The Indexed Position identified the message that is stored in the image.

2) List: Each of the bits that are stored in the image according to the Indexed Position is listed as shown in Fig. 12.

3) Extract: Secret Message has been completely extracted from the image as shown in Fig. 13.

Secret Message Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Indexed Position	16	16	2	3	6	14	16	14	5	7	11	7	4	15	7	14	7	12	10	5
Secret Message	1	1	1	0	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0	1

Secret Message Index	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Indexed Position	9	11	16	9	13	7	6	1	16	12	4	11	4	1	13	14	11	6	13	9
Secret Message	0	1	1	0	1	0	0	1	1	1	0	1	0	1	1	0	1	0	1	0

Fig. 12. List.

MSB	Secret Message						LSB
1	1	1	0	0	0	1	0
1	0	1	0	0	1	0	0
0	1	0	1	0	1	1	0
1	0	0	1	1	1	0	1
0	1	1	0	1	0	1	0

Fig. 13. Extract.

IV. Experimental Results

There are different types of security evaluations for cryptography and steganography. The randomness test using NIST Statistical Test [16] is common in cryptography. As this research implemented the standard AES, randomness testing is not necessary. This is because the research contribution lies in the proposed steganography method. Steganography evaluates the visual imperceptibility of an image by calculating the mean square error (MSE) and peak signal-to-noise ratio (PSNR) [17]. However, these tests cannot be used in the proposed method since there is no image modification.

This paper proposed two steganography security measures based on Correct Position Finding (CPF) for message detection against a brute force attack. The following section discusses the CPF probability analysis and CPF time complexity analysis. The experiments were carried out on the proposed method and three other existing methods which are SPS [10], MCO [11], and CPPM [14] as discussed in the previous section.

A. Test Samples

The proposed method was implemented in C++ and tested on the Baboon (5 KB) and Airplane (11 KB) as shown in Fig. 14 and Fig. 15. Different sizes of image samples were used to observe the effect of hiding different sizes of messages. 18 messages were generated by a PRNG.

Fig. 14. Baboon.

Fig. 15. Airplane.

B. Correct Position Finding (CPF) Probability Analysis Samples

The experiment's objective is to measure the security of the method using CPF probability analysis. This security measure is proposed to measure the prevention step to secure the hidden message. It measures the difficulty of an attacker's point of view should they tried to execute a brute force attack on the method.

Security can be evaluated by computing CPF probability of the message in the image. A result that closes to zero indicates the impossibility of finding the hidden message. A secure method should have a lower CPF probability result than random guessing. Table I shows the results indicator for this experiment to analyze the tested steganography methods.

TABLE I: Experimental Results Indicator

Low CPF Probability	High CPF Probability
• More image space utilization (possible hiding positions)	• Less image space utilization (possible hiding positions)
• Requires more effort to find the hidden secret message	• Requires less effort to find the hidden secret message
• Lower chance of finding the hidden secret message	• Greater chance of finding the hidden secret message
• A good indicator of security	• A bad indicator of security

There are several steps to calculate the CPF probability result of the hiding method. Firstly, the total number of possible embedding positions in the image which is denoted by *n* is identified. Meanwhile, the total number of secret elements is denoted by *m*. Possible embedding positions is

denoted by p_j given $j = 1, 2, \ldots, n$. The set of all possible embedding positions in the image is known as the sample space of the experiment and is denoted by *S*. The sample space *S* is shown in the following equation.

$$S = \{p_1, p_2, p_3, p_4, p_5, \ldots p_n\} \quad (1)$$

Secondly, the event or the subject matter that requires the calculation of CPF probability which is denoted by *E* is defined. *E* is the event of finding the hidden message in the image. According to this experiment, the event of finding the correct i^{th} element of the message is denoted by e_{ij} given $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. The event *E* is described in (2).

$$E = \{e_{1j}, e_{2j}, e_{3j}, e_{4j}, e_{5j}, \ldots e_{mj}\} \quad (2)$$

Since the length of the element in e_{ij} and p_j are equal for all *i* and *j*, the CPF probability of i^{th} element of the message is denoted by as shown in (3).

$$P(e_{ij}) = P(p_j) \quad (3)$$
$$= \frac{\text{Number of correct } i^{\text{th}} \text{ element of secret message}}{\text{Total number of possible embedding position in image}}$$
$$= \frac{1}{n}$$

The CPF probability of the hidden message bits in the image is denoted by $P(E)$. The equation of CPF probability $P(E)$ is shown in (4).

$$P(E) = P(e_{1j} \cap e_{2j} \cap e_{3j} \cap e_{4j} \cap e_{5j} \cap \ldots \cap e_{mj}) \quad (4)$$
$$= P(e_{1j}) \times P(e_{2j}) \times P(e_{3j}) \times P(e_{4j}) \times P(e_{5j}) \times \ldots P(e_{mj})$$

If the CPF probability of one hidden message element in the image is equally the same for all of the elements in the event *E* as shown in (5), then the CPF probability equation of event *E* is shown in (6).

$$P(e_{1j}) = P(e_{2j}) = P(e_{3j}) = P(e_{4j}) = P(e_{5j}) = \ldots = P(e_{mj}) \quad (5)$$

$$P(E) = \left[P(e_{ij})\right]^m \quad (6)$$
$$= \left(\frac{1}{n}\right)^m$$

From this equation, the more the message is hidden, the lower the result obtained. The difficulty of the attacker from getting the correct message becomes higher if the result is low. The experiment was implemented to estimate the number of possible message hiding positions in the image. Low CPF probability result represented a high number of possible hiding positions and increased the variant of message distribution that would be slowing down a brute force attack by the attacker.

C. Results of Correct Position Finding (CPF) Probability Analysis

This section analyzes the CPF probability of secret message in the image against hiding capacity from two experiments that were carried out. Table II shows the results of the proposed and existing methods using Baboon image as the cover image. Meanwhile, Table III shows the experimental results using the Airplane image.

Table II shows that the proposed method has the lowest CPF probability result as compared to others. At the lowest hiding capacity, the proposed method produced 7.2×10^{-37} CPF probability result. As the hiding capacity increased, the results become lower for all of the tested methods. The CPF probability result of the proposed method approached zero when 72 bits of the message were mapped. If the result approaches zero, it indicates impossibility [18] of finding the message. Fig. 16 shows the proposed method outperformed the existing methods.

TABLE II: CPF PROBABILITY VS NO. OF MAPPED SECRET BITS (BABOON)

	Method	CPPM [14]	SPS [10]	MCO [11]	Proposed Method	
No. of Mapped Secret Bits	8	2.4×10^{-4}	$3.0 \times 10\text{-}5$	1.2×10^{-29}	7.2×10^{-37}	CPF Probability
	16	5.9×10^{-8}	3.0×10^{-5}	1.4×10^{-58}	5.1×10^{-73}	
	24	1.4×10^{-11}	9.2×10^{-10}	1.7×10^{-87}	3.7×10^{-109}	
	32	3.5×10^{-15}	9.2×10^{-10}	2.1×10^{-116}	2.6×10^{-145}	
	40	8.4×10^{-19}	2.8×10^{-14}	2.5×10^{-145}	1.9×10^{-181}	
	48	2.0×10^{-22}	2.8×10^{-14}	3.0×10^{-174}	1.4×10^{-217}	
	56	5.0×10^{-26}	8.5×10^{-19}	3.6×10^{-203}	9.7×10^{-254}	
	64	1.2×10^{-29}	8.5×10^{-19}	4.4×10^{-232}	7.0×10^{-290}	
	72	2.9×10^{-33}	2.6×10^{-23}	5.2×10^{-261}	≈ 0	
	80	7.1×10^{-37}	2.6×10^{-23}	6.3×10^{-290}	≈ 0	
	88	1.7×10^{-40}	7.8×10^{-28}	≈ 0	≈ 0	
	96	4.2×10^{-44}	7.8×10^{-28}	≈ 0	≈ 0	
	104	1.0×10^{-47}	2.4×10^{-32}	≈ 0	≈ 0	
	112	2.5×10^{-51}	7.2×10^{-37}	≈ 0	≈ 0	
	120	6.0×10^{-55}	7.2×10^{-37}	≈ 0	≈ 0	
	128	1.4×10^{-58}	2.2×10^{-41}	≈ 0	≈ 0	
	136	3.5×10^{-62}	2.2×10^{-41}	≈ 0	≈ 0	
	144	8.5×10^{-66}	6.6×10^{-46}	≈ 0	≈ 0	

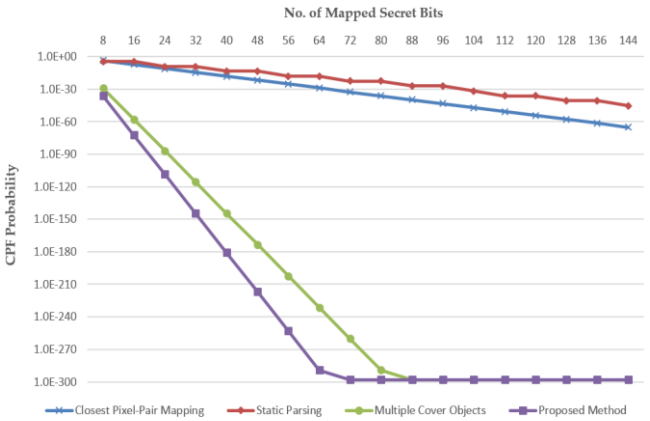

Fig. 16. CPF probability vs No. of mapped secret bits (baboon).

The second experiment was done on the Airplane image. As the image size and capacity of the message increased, the CPF probability approached zero more rapidly as shown in Table III. It takes 72 bits of message for the method to produce a result that approaches zero. Fig. 17 shows larger image sizes increased the attacker's difficulty to find the message. The proposed method achieved the lowest result, followed by MCO. CPPM and SPS produced a close result to each other.

From the two experiments, the proposed method produced the lowest CPF probability results. The method has a higher number of possible hiding positions that implemented high image space utilization in hiding the message. The high number of possible hiding positions increased the variant of message distribution which complicates attackers from getting access to the message.

TABLE III: CPF PROBABILITY VS NO. OF MAPPED SECRET BITS (AIRPLANE)

	Method	CPPM [14]	SPS [10]	MCO [11]	Proposed Method	
	8	6.3×10^{-5}	4.9×10^{-5}	8.8×10^{-33}	5.4×10^{-40}	
	16	6.2×10^{-9}	9.7×10^{-10}	5.5×10^{-65}	9.8×10^{-80}	
	24	7.5×10^{-13}	9.7×10^{-10}	7.6×10^{-98}	4.7×10^{-120}	
	32	7.5×10^{-17}	9.7×10^{-10}	9.1×10^{-130}	5.3×10^{-159}	
	40	6.6×10^{-21}	2.8×10^{-15}	3.3×10^{-163}	1.4×10^{-199}	
	48	2.0×10^{-25}	2.8×10^{-15}	3.7×10^{-195}	2.2×10^{-239}	
	56	5.0×10^{-29}	9.3×10^{-20}	2.5×10^{-228}	6.4×10^{-278}	
	64	3.2×10^{-33}	9.3×10^{-20}	2.4×10^{-260}	1.8×10^{-298}	
No. of	72	2.4×10^{-37}	6.9×10^{-25}	7.8×10^{-298}	≈ 0	CPF
Mapped	80	7.2×10^{-41}	6.9×10^{-25}	≈ 0	≈ 0	Probability
Secret	88	4.6×10^{-45}	1.4×10^{-30}	≈ 0	≈ 0	
Bits	96	6.8×10^{-49}	1.4×10^{-30}	≈ 0	≈ 0	
	104	7.7×10^{-53}	6.0×10^{-35}	≈ 0	≈ 0	
	112	5.7×10^{-57}	6.0×10^{-35}	≈ 0	≈ 0	
	120	5.8×10^{-61}	1.3×10^{-40}	≈ 0	≈ 0	
	128	5.1×10^{-65}	1.3×10^{-40}	≈ 0	≈ 0	
	136	8.5×10^{-70}	8.7×10^{-45}	≈ 0	≈ 0	
	144	3.0×10^{-74}	8.7×10^{-45}	≈ 0	≈ 0	

TABLE IV: EXPERIMENTAL RESULTS INDICATOR

Low CPF Probability	High CPF Probability
• More random distribution of hidden secret messages in the image. • It requires more time to find a hidden secret message. • A good indicator of security.	• Less random distribution of hidden secret messages in the image. • It requires less time to find a hidden secret message. • A bad indicator of security.

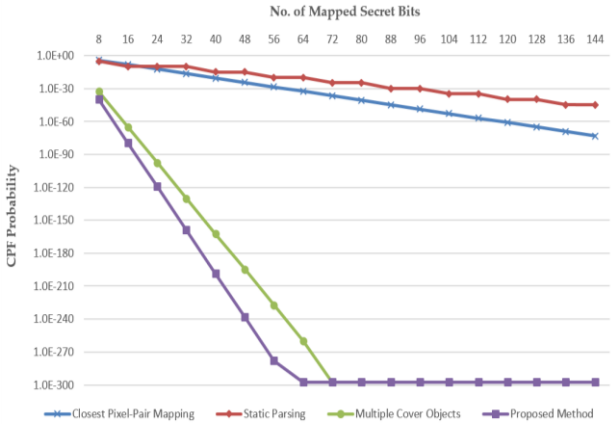

Fig. 17. CPF probability vs No. of mapped secret bits (airplane).

D. Correct Position Finding (CPF) Time Complexity Analysis

The experiment aims to measure the security by computing the CPF time complexity of a hidden message against a brute force attack. High CPF time complexity indicated that the method is secure. Meanwhile, low CPF time complexity may easily expose the location of the messages. Table IV displays the experimental results indicator that can be used to analyze each method.

The event of finding the correct i^{th} element of the secret message is denoted by e_{ij} given $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$ where the total number of message elements is denoted by m while the total number of possible embedding positions is denoted by n. E is the event of finding the hidden message in the image as described in (7). CPF time complexity for an event E to be performed is denoted by $T(E)$ is shown in (8).

$$E = \{e_{1j}, e_{2j}, e_{3j}, e_{4j}, e_{5j}, \ldots e_{mj}\} \tag{7}$$

$$T(E) = T(e_{1j}) + T(e_{2j}) + T(e_{3j}) + T(e_{4j}) + T(e_{5j}) + \ldots + T(e_{mj}) = \sum_{i=1}^{m} T(e_{ij}) \tag{8}$$

The unit used in this experiment for CPF time complexity measurement is in second. Different message and image files were used to show the effect of multiple hiding capacities. This testing method measures the randomness property of message distribution in the image. Longer time taken result indicates more random message distribution. Random message distribution is an important aspect of information hiding that would complicate attackers from finding the hidden message.

E. Results of Correct Position Finding (CPF) Time Complexity Analysis

The CPF time complexity of hidden messages in the image against hiding capacity was conducted. Table V shows the Baboon image has been used as the test sample to compare the security of methods. Table VI shows that the second test has been using the Airplane image to perform the test to compare the security of these methods.

Table V shows the proposed method recorded the highest CPF time complexity of the message in the image. At the lowest hiding capacity, the method recorded the highest CPF time complexity with 0.003 seconds. As the hiding capacity increased, all methods consumed higher CPF time complexity. The method recorded the highest CPF time complexity with 0.456 seconds at the highest hiding

capacity.

TABLE V: CPF TIME COMPLEXITY VS NO. OF MAPPED SECRET BITS (BABOON)

	Method	CPPM [14]	SPS [10]	MCO [11]	Proposed Method	
No. of Mapped Secret Bits	8	0.001	0.001	0.002	0.003	CPF Time Complexity (second)
	16	0.002	0.002	0.010	0.015	
	24	0.003	0.003	0.047	0.066	
	32	0.004	0.005	0.060	0.095	
	40	0.016	0.009	0.119	0.122	
	48	0.020	0.011	0.143	0.151	
	56	0.023	0.014	0.164	0.171	
	64	0.025	0.015	0.192	0.193	
	72	0.028	0.017	0.218	0.226	
	80	0.030	0.018	0.246	0.250	
	88	0.035	0.020	0.271	0.287	
	96	0.038	0.023	0.299	0.303	
	104	0.042	0.025	0.319	0.328	
	112	0.045	0.026	0.332	0.358	
	120	0.048	0.028	0.365	0.372	
	128	0.051	0.029	0.382	0.395	
	136	0.056	0.030	0.424	0.429	
	144	0.058	0.032	0.441	0.456	

The Airplane image was used as the sample in the second experiment. Referring to Table VI, if the image size and capacity of the message increased, CPF time complexity should be higher. The proposed method recorded the highest CPF time complexity of 0.487 seconds to map 144 bits of message. Increasing the size of the image or the capacity of the message exponentially increased the attacker's CPF time complexity to perform a brute force attack [9]. The experiment produced expected results using different sizes of images. The proposed method recorded the highest CPF time complexity followed by MCO. CPPM and SPS produced a close result to each other as in the previous experiment.

From the results of the experiments, the proposed method consumed the highest CPF time complexity. It concluded that the method applied a more random message distribution. The higher CPF time complexity, the longer time the message can be protected from the attacker. Randomization property showed that the proposed method has a high-security level in securing information transmission for better communication.

TABLE VI: CPF TIME COMPLEXITY VS NO. OF MAPPED SECRET BITS (AIRPLANE)

	Method	CPPM [14]	SPS [10]	MCO [11]	Proposed Method	
No. of Mapped Secret Bits	8	0.002	0.002	0.005	0.010	CPF Time Complexity (second)
	16	0.003	0.003	0.015	0.029	
	24	0.004	0.004	0.078	0.081	
	32	0.005	0.005	0.108	0.126	
	40	0.019	0.010	0.121	0.151	
	48	0.022	0.015	0.145	0.172	
	56	0.026	0.017	0.185	0.191	
	64	0.036	0.018	0.207	0.212	
	72	0.037	0.022	0.232	0.253	
	80	0.038	0.023	0.257	0.271	
	88	0.040	0.026	0.296	0.314	
	96	0.041	0.028	0.317	0.320	
	104	0.046	0.029	0.320	0.347	
	112	0.048	0.032	0.338	0.374	
	120	0.050	0.033	0.368	0.399	
	128	0.056	0.037	0.393	0.420	
	136	0.059	0.038	0.444	0.456	
	144	0.064	0.040	0.452	0.487	

V. CONCLUSION

A new information hiding method was presented called RSBM. RSBM hides the message bit into a similar bit of the image in a random manner. The locations of the hidden message are stored in a generated *PF*. This approach does not modify the content of the image at all. The *PF* is encrypted using the AES algorithm to protect its content from an unintended receiver.

RSBM introduced a significant advantage compared to the previous approach. Firstly, RSBM can map a message with high possible hiding positions without modifying the image that would increase the attacker's complexity in obtaining the message. The method's randomization property in distributing the message would slow down the attacker's time in deploying a brute force attack [9]. Secondly, the proposed security measures named CPF probability analysis and CPF time complexity analysis able to distinguish the security of steganography methods. The security measures are suitable to be implemented in any steganography methods which add additional evaluation metrics in their research.

From the two experiments, the RSBM produced the lowest CPF probability and recorded the highest CPF time complexity results. It proved that the method achieved high security by implementing hybrid steganography and cryptography methods. The combination adds another security layer compared to the implementation of steganography or cryptography alone.

In the future, we will attempt to reduce the size of *PF* so that it would not utilize high memory capacity when hiding a large size of a message. Secondly, an encryption algorithm should be developed. Some attacks have been done on the AES algorithm in recent years that need to be aware [19, 20]. By developing a new algorithm, attackers may take a lot of time to identify and execute attacks.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

All authors discussed the contents of the manuscript and contributed to its preparation. A. A. Z. designed and implemented the proposed scheme. N. A. A. provided the experimental setup.

ACKNOWLEDGEMENT

The authors wish to thank CyberSecurity Malaysia for funding the paper presentation and publication. This work was supported in part by the University of Malaya, Malaysia.

REFERENCES

[1] M. E. Whitman and H. J. Mattord, *Principles of Information Security*, 4th ed., USA: Cengage Learning, 2011.

[2] R. Balaji and G. Naveen, "Secure data transmission using video steganography," in *Proc. International Conference on Electro/Information Technology*, 2011, pp. 1-5.

[3] N. Provos, "Defending against statistical steganalysis," in *Proc. 10th USENIX Security Symposium*, 2001, pp. 323-336.

[4] W. C. Kuo and L. C. Wuu, "Multi-bit data hiding scheme for compressing secret messages," *Applied Sciences*, vol. 5, no. 4, pp. 1033-1049, 2015.
[5] J. Fridrich, M. Goljan, D. Soukal, and T. Holotyak, "Forensic steganalysis: Determining the stego key in spatial domain steganography," in *Proc. Electronic Imaging; International Society for Optics and Photonics*, 2005, pp. 631-642.
[6] O. C. Abikoye, K. S. Adewole, and A. J. Oladipupo, "Efficient data hiding system using cryptography and steganography," *International Journal of Applied Information Systems*, vol. 4, no. 11, pp. 6-11, 2012.
[7] P. P. Aung and T. M. Naing, "A novel secure combination technique of steganography and cryptography," *International Journal of Information Technology, Modeling and Computing*, vol. 2, no. 1. pp. 55-62, 2014.
[8] A. A. Nair and D. Job, "A secure dual encryption scheme combined with steganography," *International Journal of Engineering Trends and Technology*, vol. 13, no. 5, pp. 218-225, 2014.
[9] A. Al-Ataby and F. Al-Naima, "A modified high capacity image steganography technique based on wavelet transform," *The International Arab Journal of Information Technology*, vol. 7, no. 4, pp. 358-364, 2010.
[10] H. Farhat, K. Challita, and J. Zalaket, "Static parsing steganography," in *Proc. International Conference on Digital Information and Communication Technology and Its Applications*, 2011, pp. 485-492.
[11] K. Challita and H. Farhat, "Combining steganography and cryptography: New directions," *International Journal of New Computer Architectures and their Applications*, vol. 1, no. 1, pp. 199-208, 2011.
[12] M. Hussain and M. Hussain, "A survey of image steganography techniques," *International Journal of Advanced Science and Technology*, vol. 54, pp. 113-124, May 2013.
[13] P. Y. Pawar and S. H. Gawande, "M-commerce security using random LSB steganography and cryptography," *International Journal of Machine Learning and Computing*, vol. 2, no. 4, pp. 427-430, 2012.
[14] A. Ahmed, N. Agarwal, and S. Banerjee, "Image steganography by closest pixel-pair mapping," in *Proc. IEEE International Conference on Advances in Computing, Communications and Informatics*, 2014, pp. 1971-1975.
[15] J. Daemen and V. Rijmen, *AES Proposal: Rijndael*, 1999.
[16] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," *Booz-Allen and Hamilton Inc Mclean Va*, 2001.
[17] M. Hussain, W. A. Abdul, N. Javed, and K. H. Jung, "Hybrid data hiding scheme using right-most digit replacement and adaptive least significant bit for digital images," *Symmetry*, vol. 8, no. 6. pp. 41, 2016.
[18] P. Manimegalai, K. S. Gomathi, D. Ponniselvi, and M. Santha, "The image steganography and steganalysis based on peak-shaped technique for MP3 audio and video," *International Journal of Computer Science and Mobile Computing*, vol. 3, no. 1, pp. 300-308, 2014.
[19] M. Karpovsky, K. J. Kulikowski, and A. Taubin, "Robust protection against fault-injection attacks on smart cards implementing the advanced encryption standard," in *Proc. International Conference on Dependable Systems and Networks*, 2004, pp. 93-101.
[20] A. Moradi, M. T. M. Shalmani, and M. Salmasizadeh, "A generalized method of differential fault attack against AES cryptosystem," in *Proc. International Workshop on Cryptographic Hardware and Embedded Systems*, 2006, pp. 91-100.

Abdul Alif Zakaria is an information security analyst in the Department of Cryptography Development at the CyberSecurity Malaysia. He received his bachelor's degree in Statistics in 2008 and a master's degree in information security in 2018 from University of Malaya. His research interests are blockchain, cryptography, and steganography.

Norli Anida Abdullah is a senior lecturer in the Mathematics Division, Center for Foundation Studies in Science at the University of Malaya (UM). She went through her A-level in the UM, and received all her BSc, MSc, and PhD from UM. Her primary research interest is in the field of applied statistics. Specifically, she is interested in medical data, outliers, circular data, biomedical imaging, and information hiding.