Stream-keys generation based on graph labeling for strengthening Vigenere encryption

Antonius Cahya Prihandoko[1], Dafik[2], Ika Hesti Agustin[3]
[1]Department of Informatics, Faculty of Computer Science, University of Jember, Jember, Indonesia
[2]Department Mathematics Education, Faculty of Teacher Training and Education, University of Jember, Jember, Indonesia
[3]Department of Mathematics, Faculty of Mathematics and Natural Science, University of Jember, Jember, Indonesia

Article Info

Article history:

Received May 4, 2021
Revised Dec 19, 2021
Accepted Jan 26, 2022

Keywords:

Cipher block chaining
Encryption algorithm
Graph labeling
Stream-keys
Vigenere cipher

ABSTRACT

This paper address the cryptographic keys management problem: how to generate the cryptographic keys and apply them to secure encryption. The purpose of this research was to study on utilizing graph labeling for generating stream-keys and implementing the keys for strengthening Vigenere encryption. To achieve this objective, the research was carried out in four stages: developing an algorithm for generating stream-keys, testing the randomness of the constructed keys, implementing the eligible keys in a modified Vigenere encryption and, finally, analyzing the security of the encryption. As the result, most of stream-keys produced by the algorithm are random, and the implementation of the stream keys to the modified Vigenere cipher achieve good security. The contributions of this research are utilizing graph labeling to generate stream-keys and providing different encryption keys for different blocks in a block based cipher with low storage capacity. The novel technical results yielded from this research are the algorithm of developing the source of the stream-keys based on graph labeling, the algorithm of constructing the initial block keys, and the protocol of a modified Vigenere encryption using stream-keys and operated in cipher block chaining mode.

Corresponding Author:

Antonius Cahya Prihandoko
Department of Information Technology, University of Jember
Kalimantan Street No. 37, Jember-68121, East Java, Indonesia
Email: antoniuscp.ilkom@unej.ac.id

1. INTRODUCTION

A critical characteristic of information is confidentiality. Information is said to be confidential when it is kept secret to unauthorized parties [1]. Confidentiality guarantees that only those who have rights for accessing information are able to do so. Cryptography is a well-known method to achieve information confidentiality. In this method, information is encrypted before being distributed over an insecure networks. With this capability, cryptography has a widely range of applications: securing content distribution systems [2], improving digital rights management systems [3]-[4], establishing speech encryption [5]-[6], tracing traitor in a content distribution system [7], solving security problem in mobile computing [8]-[9], and many more. The strength of cryptography protocols relies on the encryption-decryption keys management: how to protect the keys from disclose to unauthorized parties. Without a doubt, it is the biggest challenge for many cryptographic methods. Investigations on the keys management are unceasingly carried out and are concentrated to accomplish information confidentiality in accordance with the level of security required.

To address the keys management problem, many researchers focused on keys generation. This aspect is the most strongly part of encryption technique [10]. Many keys generation techniques have been proposed in previous publications. Some techniques were proposed to produce encryption keys for regular information exchange protocols by utilizing various aspects. Ramasamy *et al.* [11] generated the keys by integrating an enhanced logistic map (ELM) with the block scrambling encryption technique. Agarwal *et al.* [12] constructed a non-transitional cryptosystem key by utilizing public keys exchange protocol; at the end of the protocol, sender and receiver will be able to generate the same private key. Gayathri *et al.* [13] generated private key by combining the advanced encryption standard (AES) and elliptic curve cryptography (ECC). Moosavi *et al.* [14], [15] and Gonzales-Manzano *et al.* [16] made use the feature of electrocardiogram (ECG) to construct secret keys. The interpulse interval (IPI) feature of ECG inspired the proposed two approaches. The first approach combined a pseudo-random number and consecutive IPI sequences. The second approach integrated the advanced encryption standard (AES) algorithm and IPI sequence as the initial keys generator for the AES algorithm. The security of the generated keys is analyzed in terms of distinctiveness, randomness, and temporal variance. Another techniques generated encryption keys for specific purposes. Mahendran and Mani [17] generated the key using sequential advancement and permutated predetermined procedures to overcame the drawback of hill cipher in selecting the correct encryption key matrix. Al-Moliki *et al.* [18] introduced a key extraction protocol for optical orthogonal frequency division multiplexing (OFDM) schemes to secure the visible light communication network. Margelis *et al.* [19] generated secret key based on discrete cosine transform (DCT) to achieve confidentiality in the internet of things (IoT). Song *et al.* [20] proposed a privacy-preserving key generation scheme to protect data confidentiality as well as user's privacy in a cloud technology. They designed a new attribute-based encryption scheme that protects user's privacy during key releasing. The scheme separates the functionality of attribute auditing and key generating such that the key generation center will not identify user's attribute and the attribute auditing center is not able to detect the user's secret key. Briefly, many key generation techniques were ultimately aimed at achieving information confidentiality. Key generation itself, however, only contributes partly to information security. It must be collaborated with a secure encryption mechanism to achieve an optimum information confidentiality.

This paper address the cryptographic keys management problem by collaborating key generation and encryption modification. We propose a novel solution for the problem: generating encryption keys using graph labeling and applying the keys to a modified Vigenere encryption mechanism. This paper uses a specific terminology "stream-keys" to explain the encryption keys. A stream-key is the encryption key that is generated from an initial key using a particular stream function. In the simulation stage, the eligible keys were applied to a modified Vigenere cipher and were observed whether the keys can improve the security of the encryption mechanism. The Vigenere cipher was chosen in the simulation stage because of two reasons. First, this cipher is one of the most studied cryptographic schemes and its algorithm is very simple to encrypt the text message [21], [22]. Second, Vigenere cipher is a traditional cryptosystem that still has many applications. This cipher can be applied for protecting data security [23], [24], providing a secure communication [25], [26], securing data in the form of digital images [27], and creating a digital signature scheme [28]. Most of those applications, however, utilized Vigenere cipher in its original form, that is, information is split into blocks and all blocks are encrypted using the same key. This mechanism is relatively easy for an intruder to analyze the encryption key and break the system. We propose to solve this obstacle by applying the stream-keys which provide different encryption keys for different blocks. Furthermore, the encryption process is undertaken in cipher block chaining mode. This collaboration makes the adversary find difficulties to analyze and guess the keys. The novel contributions of this research are generating stream-keys using graph labeling and providing different encryption keys for different blocks in a block based cipher with low storage capacity. The novel technical results yielded from this research are the algorithm of developing the source of the stream-keys based on graph labeling, the algorithm of constructing the initial block keys, and the protocol of a modified Vigenere encryption using stream-keys and operated in cipher block chaining mode.

The rest of this paper is outlined as follows. Section 2 describes the model and algorithm for constructing stream-keys from graph labeling. Section 3 provides the results of the randomness test applied to the constructed keys, encryption algorithms in the modified Vigenere cipher, and security analysis of the cipher. Section 4 concludes with some highlight statements yielded from the research.

2. RESEARCH METHOD

The research was undertaken in four steps: i) developing a model of utilizing graph labeling for constructing stream-keys; ii) undertaking randomness tests to all generated keys; a random key is eligible to be used as the encryption key; iii) applying eligible keys to the modified Vigenere encryption. In this stage, different plaintext blocks were encrypted using different block keys. Furthermore, the encryption was undertaken in cipher block chaining (CBC) mode to confuse the encryption of a particular block with the one of the previous block; and iv) analyzing the security of the modified Vigenere encryption based on four attacks model (cipher-text only, known plaintext, chosen plaintext, and chosen cipher-text).

The model of constructing encryption keys based on graph labeling is depicted in Figure 1. First of all, graph labeling produces a set of labels; the labeling can be vertices labeling, edges labeling or total labeling. The second step is constructing a layered diagram of the labeling. The layered diagram provides a sequence of numbers that will be used as the source (s) of encryption keys. Using parameters initial digit (i) and length of block (b), an initial block key can be extracted from the source (s). The initial block key will be the input of the stream function to generate the stream-key.

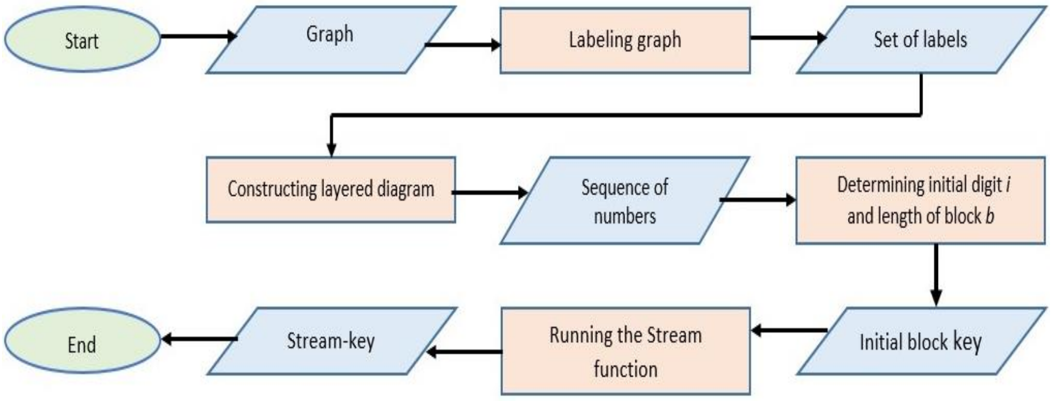
Start
Graph
Labeling graph
Set of labels
Constructing layered diagram
Sequence of numbers
Determining initial digit *i* and length of block *b*
Initial block key
Running the Stream function
Stream-key
End

Figure 1. A model of Stream-key generation from graph labeling

The material used in the experiment, in term of graph labeling, was a super H-antimagic total labeling (SHATL) implemented to a generalized shackle of graph. Suppose $V(G)$ and $U(G)$ is the set of vertices and edge, respectively, of a graph G. A bijective function f is called an (a,d)-H-antimagic total labeling of graph G if $f: V(G) \cup E(G) \rightarrow \{1, 2, \dots, |V(G)| + |E(G)|\}$ such that for all subgraphs of G isomorphic to H, the total H-weights $w(H) = \sum_{v \in V(H)} f(v) + \sum_{e \in E(H)} f(e)$ form an arithmetic sequence $\{a, a + d, a + 2d, \dots, a + (n - 1)d\}$ where a and d are positive integers and n is the number of all subgraphs of G isomorphic to H. Moreover, if $f(V(G) \rightarrow \{1,2, \dots |V(G)|\}$, then the ($a$, d)-H-antimagic total labeling f is called super.

A shackle of graph H, represented by $G = shack(H, v, n)$, is a graph G generated by non-trivial graphs $H_1, H_2, \dots, H_n$, such that for every $1 \leq s, t \leq n$, with $|s - t| \geq 2$, H_s and H_t have no common vertex, but for every $1 \leq i \leq n - 1$, H_i and H_{i-1} have precisely one common vertex v, called connecting vertex, and all $n - 1$ connecting vertices are different. A generalized shackle of graph, represented by $G = gshack(H, K \subset H, n)$, is the graph obtained from $G = shack(H, v, n)$ by replacing the connecting vertex with any subgraph $K \subset H$. The existence of super (a,d)-H antimagic total labeling of generalized shackle of graph was proved using an integer set partition technique as it was appeared in [29], [30], and [31]. This proof warrants that generating encryption keys using SHATL is possible. A sequence of numbers produced from the graph labeling is then used as the source of encryption keys. For simulation, assume the cipher is working on 26 English letters, the numbers sequence can be developed through the algorithm 1.

Algorithm 1. Developing the source of stream-keys
```
input: a graph
output: a source of stream-keys $s$
1. START
2. INPUT a graph
3. Define $f$ to label the graph elements
4. IF $f$ is bijective, THEN continue to 5, ELSE back to 3
5. Take a certain $d$ for super ($a,d$)-HATL
6. Let $z$ ← the number of vertices plus 26
7. Draw the layered diagram by ignoring all labels greater than $z$
8. Place all edge labels in sequence from left to right and start from the top to the
   bottom layer.
9. Name the sequence by $s$ and let $t$ ← length of $s$
10. END
```

The sequence *s* of labels is then used as the source of stream-key. In the modified Vigenere cipher, a plaintext is divided into some blocks and is then encrypted block by block. At the encryption process, an initial block key *k* can be generated from the sequence *s* through the algorithm 2.

Algorithm 2. Generating the initial block key
```
input: a numbers sequence s of length t
output: the initial block key k
1. START
2. INPUT b ← length of block
2. INPUT i, such that $1 \le i \le t - b$
3. Take $k = s_i, s_{i+1}, s_{i+2}, \dots, s_{i+b-1}$ as initial block key
4. END
```

The encryption keys for the subsequent blocks can be generated from *k* by a stream function 1.

$$k_{j+b} = g(k_j, k_{j+1}, k_{j+b-1}, s_{j+b \ mod\ 26}) \tag{1}$$

The stream function *g* is run simultaneously with the encryption process until all blocks keys are generated.

3. RESULTS AND DISCUSSION

The algorithm 1 produces a sequence of numbers *s* that be used as the source of stream-keys. To illustrate this algorithm, we use the example of SHATL in a generalized shackle of graph as it was provided in [32]. The labeling is illustrated in Figure 2. It shows that the vertex and edge labels start from 1 to 30 and 31 to 79, respectively. A layered diagram rooted at label 1 is drawn by ignoring the labels greater than 56 as shown in Figure 3. The sequence obtained from the diagram, in its equivalence modulo 26, is *s* = 5, 13, 22, 0, 2, 14, 24, 23, 25, 1, 3, 21, 6, 10, 4, 17, 20, 7, 11, 16, 19, 8, 12, 15, 18, 9. The sequence *s* was then used as the source of stream-keys.

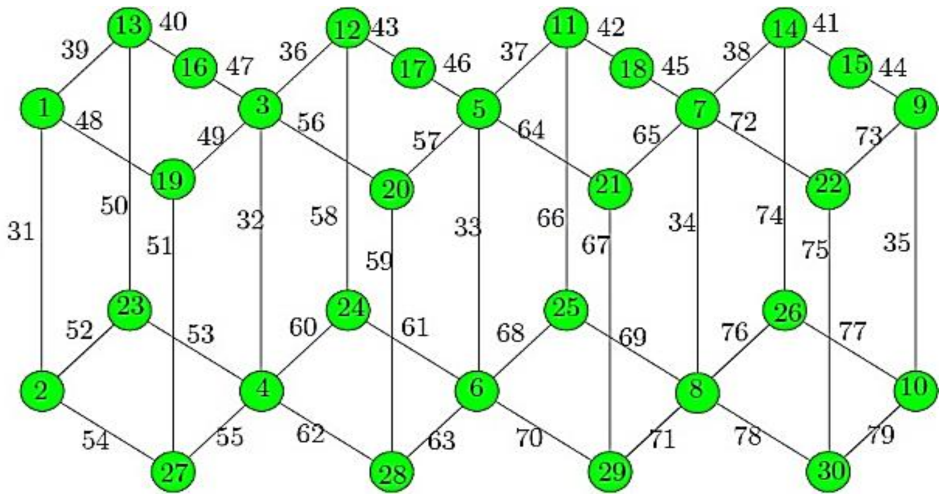

Figure 2. SHATL of a generalized shackle of graph *G* = *shack (H,K,4)*

The algorithm 2 generates an initial block key taken from *s*. For example, suppose *i* = 4, *b* = 7, and the stream function is defined as $k_{j+7} = k_j + k_{j+2} + s_{j+7 \ mod\ 26} \ mod\ 26$. The extracted initial block key is *k*=0, 2, 14, 24, 23, 25, 1 and, thus the stream-key is 0, 2, 14, 24, 23, 25, 1 | 13, 1, 14, 18, 4, 22, 6 | 18, 13, 25, 25, 0, 7, 1 | 3, 1, 17, 15, 6, 23, 24 | 20, 18, 11, 10, 1, 16, 17 | 8, 23, 18, 10, 22, 15, 8 | ...

Time complexity of the algorithm at processing an input graph until producing a stream-key is *O(n)*. This process can be divided into three steps. First of all is graph labeling. Growth time of the graph labeling algorithm execution is constant, so that its time complexity is *O(1)*. Secondly, the initial key determination stage using sequential search algorithm; the execution time of the algorithm runs linearly, so the complexity is *O(n)*. Finally, every input key is proceed by the same function, so that its time complexity is *O(1)*. Overall, time complexity of this process is *O(n)*. This process includes in a polynomial time algorithm with efficient performance.

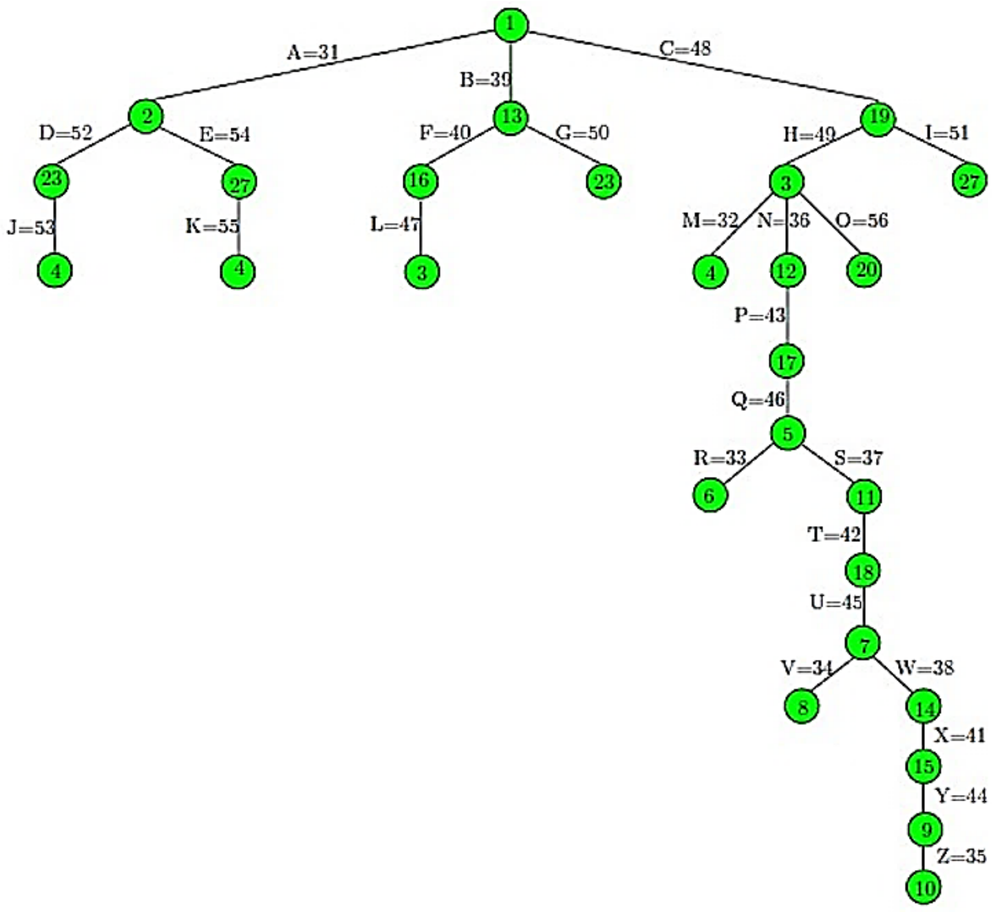

Figure 3. The layered diagram rooted at label 1

3.1. Randomness of the constructed keys

Randomness of the constructed stream-keys must be tested. A random key indicates that there is no specific pattern that can be utilized by any intruder for guessing the key. Utilizing a single sequence s, our stream-keys construction algorithm can generate multiple stream-keys. A randomness test [33] was applied to all generated sequences in the simulation. A randomness test requires a sequence with minimum length is 40. This requirement can always be fulfilled by the stream-keys construction algorithm. In our model, an infinite length stream-key can be generated from a single initial block key. In practice, a stream function can be run until the constructed stream-key has the same length as the plaintext.

To simulate the test we applied the MATLAB function, runstest, to all constructed stream-keys. The function returned a test decision for the null hypothesis: *the values in the stream-key come in random order*. Two possible results of the test are h=0 and h=1. The value h=0 means that the null hypothesis cannot be rejected at the 5% significance level, while h=1 means that the null hypothesis can be rejected. Recall previous example for i=4 and b=7, and let us construct the stream-keys using stream function in (2).

$$k_{j+b} = (k_{j+x} + k_{j+y} + s_{j+b \bmod 26}) \bmod 26 \tag{2}$$

Table 1 presents the randomness test results for stream-keys generated from a single initial key (i=4; b=7) by the stream function (2) with various values of x and y. The table shows that 97.3% tests return value of h = 0. These results indicate that the null hypothesis is accepted for almost all generated stream-keys. It means that most of these keys come in random order.

Experiment was continued for various parameter values. Table 2 presents the results of randomness test for stream-keys generated from various initial keys ($1 \leq i \leq 24$ and $3 \leq b \leq 13$) by the stream function (2) with x=0 and y=1. The values of $h=0$ for all columns are at least 88.8%, and even in some columns all values of h's are 0. Overall, this results indicate that 95.2% of constructed stream-keys are random.

The same experiments like one that was applied for the values x=0 and y=1 were also applied for the other values of x and y. On the other words, the experiments were continued using various stream functions. For $1 \le i \le 24$, Table 3 presents the percentages of random stream-keys at each combination of parameters (b, x, y). The number of stream-keys generated in the experiments was 1,024. The randomness test results show that 983 out of 1,024 stream-keys or 95.9% of the generated keys come in random order. These results indicate that most stream-keys constructed by our model are eligible to be used as the encryption keys.

Table 1. Randomness of generated keys for certain value of x and y

x	y					
	1	2	3	4	5	6
0	1	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

Table 2. Test results for encryption keys generated from various initial keys

i	b										
	3	4	5	6	7	8	9	10	11	12	13
1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	0	0	0	0
3	0	1	0	0	1	0	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	1	0	0	0	0
14	0	0	0	0	0	1	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	-
16	0	0	0	0	0	0	0	0	0	-	-
17	0	1	0	0	0	0	0	0	-	-	-
18	0	0	0	0	0	0	0	-	-	-	-
19	0	0	0	0	0	0	-	-	-	-	-
20	0	0	0	0	0	-	-	-	-	-	-
21	0	0	0	0	-	-	-	-	-	-	-
22	0	0	0	-	-	-	-	-	-	-	-
23	0	0	-	-	-	-	-	-	-	-	-
24	0	-	-	-	-	-	-	-	-	-	-

Table 3. Percentages of random encryption keys at each parameters

b	(x,y)					
	(0,1)	(1,2)	(2,3)	(3,4)	(4,5)	(5,6)
3	91.7%	95.8%	-	-	-	-
4	91.3%	95.7%	95.7%	-	-	-
5	100%	95.5%	90.9%	95.5%	-	-
6	100%	100%	95.2%	95.2%	95.2%	-
7	90%	100%	100%	95%	95%	95%
8	94.7%	94.7%	100%	100%	94.7%	94.7%
9	88.9%	94.4%	88.9%	100%	100%	94.4%
10	94.1%	88.2%	94.1%	94.1%	100%	100%
11	100%	93.8%	93.8%	100%	100%	100%
12	100%	100%	86.7%	100%	93.3%	100%
13	100%	100%	100%	92.9%	92.9%	100%
0's	199	201	175	157	135	116
Keys	209	209	185	162	140	119
% Random	95.2	96.2	94.6	96.9	96.4	97.5

3.2. Applying eligible keys to the modified Vigenere encryption

In this research, Vigenere cipher is modified in two terms. Firstly, instead of using a same block key, different blocks are encrypted using different keys by utilizing the eligible generated stream-keys.

However, the encryption system does not need a big storage capacity to save the keys. The system only needs to store the initial block key and the stream function. The keys used to encrypt subsequent blocks are constructed from the previous one by the stream function. Secondly, to make the modified Vigenere cipher stronger, the encryption process is undertaken in cipher block chaining (CBC) mode. By this mode, the encryption process of a block will be connected to the previous and the subsequent blocks encryption. Encryption process in the modified Vigenere cipher is illustrated in Figure 4.

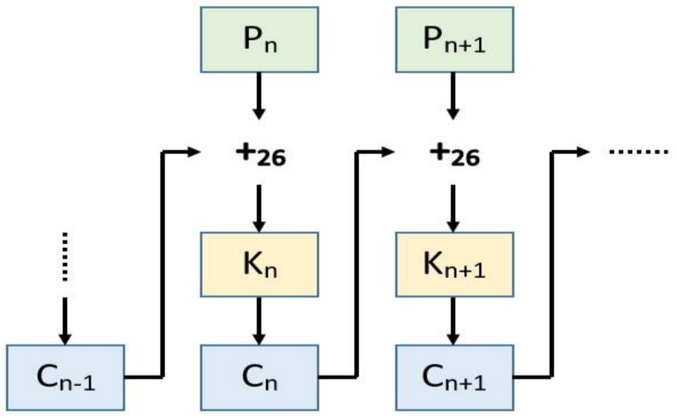

Figure 4. The CBC based encryption for the modified Vigenere cipher

Suppose a plaintext P of length h is divided into blocks of length b. For n = 1 until $\left\lceil\frac{h}{b}\right\rceil$, the ciphertext blocks C_n is computed using (3):

$$C_n = C_{n-1} + P_n + K_n \; mod \; 26 \tag{3}$$

where P_n, K_n, and C_n are the n-th block of plaintext, key sequence, and cipher-text, respectively. For n = 1, C_{n-1} is a null vector.

To illustrate the encryption algorithm in the modified Vigenere cipher, let us take an eligible stream-key with parameters taken from Table 2. According to this table, one of the stream-keys having the value h = 0 is the key that was generated from the initial block key with parameters i = 5 and b = 7 and was generated using stream function in (2) with x = 0 and y = 1. The generated stream-key was 2, 14, 24, 23, 25, 1, 3 | 15, 13, 24, 17, 6, 14, 22 | 19, 5, 22, 8, 10, 3, 23 | 10, 16, 22, 1, 18, 13, 3 | 0, 14, 11, 17, 2, 15, 4 | 17, 20, 8, 3, 21, 10, 15 | .....

Let us now use this stream-key to encrypt a plaintext "*all fight against corona*". By omitting the spaces, the plaintext is divided into three blocks of the length seven as follows: *allfigh* | *tagains* | *tcorona.* Each block is then encrypted using different key taken from the first three blocks of the stream-key: 2 14 24 23 25 1 3 | 15 13 24 17 6 14 22 | 19 5 22 8 10 3 23. By the algorithm, plaintext "allfightagainscorona" is encrypted to be cipher-text "CZJCHHKKMNTVIYWTXSTYV"

Time complexity of the modified Vigenere encryption algorithm at processing an input plaintext and resulting the cipher-text is *O(n log n)*. In this process, plaintext is divided into several blocks and each block is encrypted using a different key, then the encryption results from all blocks are combined to get the ciphertext. The linear logarithm algorithm belongs to the group of algorithms whose execution time grows in a polynomial time, so it is efficient.

3.3. Security analysis

To analyze the security of the modified Vigenere cipher, we simulated four main possible attack models: cipher-text only, known plaintext, chosen plaintext, and chosen cipher-text.

a) Cipher-text only: In the cipher-text only attack model, an intruder only knows the cipher-text. The intruder may make use of a brute-force scenario; applying all possible keys to decrypt the cipher-text for getting a meaningful plaintext. Suppose, plaintext of length h is divided into blocks of the length b. In the modified Vigenere encryption, different blocks are encrypted using different keys, so that there are 26^b possible keys for each block, or in total there exists $(26^b)^{\left\lceil\frac{h}{b}\right\rceil}$ possible keys. Moreover, in CBC mode, the keys for the 2nd to $\left\lceil \frac{h}{b} \right\rceil$-*th* blocks are confused by previous block encryption. This encryption mechanism is much stronger than the ordinary Vigenere encryption, where the intruder needs guess only 26^b possible keys at that case.

b) Know plaintext: The known plaintext attack model assumes that an intruder has information a part of the cipher-text and its corresponding plaintext. In the modified Vigenere cipher, however, knowing only several pairs of cipher-text-plaintext, or even all pairs in a block, is not adequate to reveal the whole blocks. This because the cryptosystem is a polyalphabetic cipher and different blocks are encrypted using different sequence of keys. In this attack model, again, the modified Vigenere cipher is also more secure that the ordinary one. In the ordinary Vigenere cipher, when all pairs of cipher-text and plaintext in a block are revealed then whole system will be breached.

c) Chosen cipher-text or plaintext: The chosen plaintext or chosen cipher-text attack models assumes that an intruder has a temporary access to the encryption or decryption machine. The intruder attempts to encrypt or decrypt a number of dummy plaintext or cipher-text and observes the results to derive the encryption or decryption keys. In our cryptosystem, even though utilizing the same source s, a new stream-key can be generated each time an encryption process is started by determining a new initial block key. Therefore, a temporary access to the encryption or decryption machine is not sufficient to break the system. In these attack models, an adversary needs much more efforts to break the modified Vigenere cipher compared to the ordinary one.

Overall in dealing with various attack models, modified Vigenere encryption is superior to regular Vigenere.

4. CONCLUSION

A novel model of stream-keys generation based on graph labeling has been developed in this research. Stream-keys produced by the model are mostly random, and thus eligible to be used as the encryption keys. The stream-keys make the block based cipher stronger: different blocks plaintext are encrypted by different block keys. Since only the initial block key and the stream function that need to be save, the block based cipher developed in this research requires less storage capacity. Furthermore, encryption mechanism is undertaken in the cipher block chain mode. This mode makes the encryption process of a plaintext block to be connected to the subsequent block encryption. This research ends up with a more secure block based cipher compared to the ordinary Vigenere cipher. All algorithms developed in this research: graph labeling based stream-keys generation algorithm and modified Vigenere encryption algorithm, are efficient according to their time complexities. Open problem: the future work of this research is studying whether the key generation model developed in this research can be used to create binary stream-keys that can be utilized to strengthen a modern block based cipher.

ACKNOWLEDGEMENTS

The authors wish to thank to all colleagues at the combinatorics, graph theory, and network topology (CGANT) research group, the University of Jember, Indonesia, for establishing this collaboration research.

REFERENCES

[1] M. E. Whitman and H. J. Mattord, *Principles of information security*. Boston: Course Technology, 2012.

[2] A. C. Prihandoko, H. Ghodosi, and B. Litow, "Secure and private content distribution in the DRM environment," *Information Systems International Conference*, pp. 659–664, 2013.

[3] A. C. Prihandoko, H. Ghodosi, and B. Litow, "White-box implementation to advantage DRM," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 7, no. 2, pp. 460–467, Apr. 2017, doi: 10.18517/ijaseit.7.2.1445.

[4] A. C. Prihandoko and H. Ghodosi, "Oblivious content distribution system to advantage digital rights management," in *2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT)*, Aug. 2017, vol. CAIPT 2017, pp. 1–5, doi: 10.1109/CAIPT.2017.8320733.

[5] O. M. Al-hazaimeh, "A new dynamic speech encryption algorithm based on lorenz chaotic map over internet protocol," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 5, pp. 4824–4834, Oct. 2020, doi: 10.11591/ijece.v10i5.pp4824-4834.

[6] Y. Alemami, M. A. Mohamed, S. Atiewi, and M. Mamat, "Speech encryption by multiple chaotic maps with fast Fourier transform," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 6, pp. 5658–5664, 2020, doi: 10.11591/ijece.v10i6.pp5658-5664.

[7] A. C. Prihandoko, H. Ghodosi, and B. Litow, "Deterring traitor using double encryption scheme," in *2013 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT)*, Dec. 2013, vol. COMNETSAT, pp. 100–104, doi: 10.1109/COMNETSAT.2013.6870869.

[8] A. S. Barote and A. O. Bang, "Security risks, limitations and operational problems in mobile computing," *International Journal of Advend Research in Computer and Electrical*, pp. 76–79, 2018.

[9] A. O. Bang and P. L. Ramteke, "Mobile computing application design and development issues," *International Journal of Scientific Engineering and Research*, vol. 2, no. 2, pp. 560–563, 2013.

[10] P. Dixit, A. K. Gupta, M. C. Trivedi, and V. K. Yudav, "Traditional and hybrid encryption techniques: a survey," in *Networking Communication and Data Knowledge Engineering, Lecture Notes on Data Engineering and Communication Technologies*, 2018, pp. 239–248, doi: 10.1007/978-981-10-4600-1_22.
[11] P. Ramasamy, V. Ranganathan, S. Kadry, R. Damasevicius, and T. Blazauskas, "An image encryption scheme based on block scrambling, modified zigzag transformation and key generation using enhanced logistic-tent map," *Entropy*, vol. 21, no. 656, pp. 1–17, 2019, doi: 10.3390/e21070656.
[12] S. Agarwal, "Symmetric key encryption using iterated fractal functions," *International Journal on Computer Network and Information Security*, vol. 4, pp. 1–9, 2017, doi: 10.5815/ijcnis.2017.04.01.
[13] P. Gayathri, S. Umar, G. Sridevi, N. Bashwanti, and R. Srikanth, "Hybrid cryptography for random-key generation based on ECC algorithm," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 3, pp. 1293–1298, 2017, doi: 10.11591/ijece.v7i3.pp1293-1298.
[14] S. R. Moonsavi, E. Nigussie, M. Levorato, S. Virtanen, and J. Isoaho, "Low-latecy approach for secure ECG feature based cryptographic key generation," *IEEE Access*, vol. 6, pp. 428–442, 2018, doi: 10.1109/ACCESS.2017.2766523.
[15] S. R. Moonsavi, E. Nigussie, S. Virtanen, and J. Isoaho, "Cryptographic key generation using ECG signal," in *14th IEEE Annual Consumer Communication & Networking Conference*, 2017, pp. 1024–1031, doi: 10.1109/CCNC.2017.7983280.
[16] L. Gonzales-Manzano, J. M. de Fuentes, P. Peris-Lopez, and C. Camara, "Encryption by heart (EbH)—using ECG for time-invariant symmetric key generation," *Future Generation Computer Systems*, vol. 77, pp. 136–148, 2017, doi: 10.1016/j.future.2017.07.018.
[17] R. Mahendran and K. Mani, "Generation of key matrix for hill cipher encryption using classical cipher," in *World Congress on Computing and Communication Technologies*, 2017, vol. WCCCT 2017, pp. 51–54, doi: 10.1109/WCCCT.2016.22.
[18] Y. M. Al-Moliki, M. T. Alresheedi, and Y. Al-Harthi, "Secret key generation protocol for optical OFDM system in indoor VLC networks," *IEEE Photonic Journal*, vol. 9, no. 2, 2017, doi: 10.1109/JPHOT.2017.2667400.
[19] G. Margelis, X. Fafoutis, G. Oikonomou, R. Piechocki, T. Tryfonas, and P. Thomas, "Efficient DCT-based secret key generation for the internet of things," *Ad Hoc Networks*, vol. 92, pp. 1–11, 2019, doi: 10.1016/j.adhoc.2018.08.014.
[20] Y. Song, H. Wang, X. Wei, and L. Wu, "Efficient-attribute-based encryption with privacy-preserving key generation and its application in industrial cloud," *Secure Communication Networks*, vol. 2019, pp. 1–9, 2019, doi: 10.1155/2019/3249726.
[21] S. D. Nasution, G. L. Ginting, M. Syahrizal, and R. Rahim, "Data security using vigenere cipher and goldbach codes algorithm," *International Journal of Enggineering Research and Technology*, vol. 6, no. 1, pp. 360–363, 2017.
[22] A. A. Soofi, L. Riaz, and U. Rasheed, "An enhanced vigenere cipher for data security," *International Journal of Scientific and Technology Research*, vol. 5, no. 3, pp. 141–145, 2016.
[23] A. D. Achmad, A. A. Dewi, M. R. Purwanto, P. T. Nguyen, and I. Sujono, "Implementation of vigenere cipher as cryptographic algorithm in securing text data transmission," *Journal of Critical Reviews*, vol. 7, no. 1, pp. 76–79, 2020, doi: 10.22159/jcr.07.01.15.
[24] R. Rahim *et al.*, "Combination vigenere cipher and one time pad for data security," *International Journal of Engineering & Technology*, vol. 7, no. 2.3, Mar. 2018, doi: 10.14419/ijet.v7i2.3.12624.
[25] D. Gautam, C. Agrawal, P. Sharma, M. Mehta, and P. Saini, "An enhanced cipher technique using vigenere and modified caesar cipher," in *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, May 2018, pp. 1–9, doi: 10.1109/ICOEI.2018.8553910.
[26] A. Saraswat, C. Khatri, Sudhakar, P. Thakral, and P. Biswas, "An extended hybridization of vigenere and caesar cipher techniques for secure communication," *Procedia Computer Science*, vol. 92, pp. 355–360, 2016, doi: 10.1016/j.procs.2016.07.390.
[27] Y. A. Gerhana, E. Insanudin, U. Syarifudin, and M. R. Zulmi, "Design of digital image application using vigenere cipher algorithm," in *2016 4th International Conference on Cyber and IT Service Management*, Apr. 2016, pp. 1–5, doi: 10.1109/CITSM.2016.7577571.
[28] R. Damara Ardy, O. R. Indriani, C. A. Sari, D. R. I. M. Setiadi, and E. H. Rachmawanto, "Digital image signature using triple protection cryptosystem (RSA, Vigenere, and MD5)," in *2017 International Conference on Smart Cities, Automation & Intelligent Computing Systems (ICON-SONICS)*, Nov. 2017, pp. 87–92, doi: 10.1109/ICON-SONICS.2017.8267827.
[29] M. Baca, L. Brankovic, M. Lascsakova, O. Phanalasy, and A. Semanicova–Fenovcıkova, "On d-Antimagic labelings of plane graphs," *Electronic Journal of Graph Theory Applications*, vol. 1, no. 1, pp. 28–39, 2013, doi: 10.5614/ejgta.2013.1.1.3.
[30] Dafik, A. K. Purnapraja, and R. Hidayat, "10.1016/j.procs.2015.12.082," *Procedia Computer Science*, vol. 74, pp. 93–99, 2015, doi: 10.1016/j.procs.2015.12.082.
[31] Dafik, Slamin, D. Tanna, A. Semaničová-Feňovčíková, and M. Bača, "Constructions of H-antimagic graphs using smaller edge-antimagic graphs," *Ars Combinatoria*, vol. 133, pp. 233–245, 2017.
[32] A. C. Prihandoko, D. Dafik, and I. H. Agustin, "Implementation of super H-antimagic total graph on establishing stream cipher," *Indonesian Journal of Combinatorics*, vol. 3, no. 1, pp. 14–23, Jun. 2019, doi: 10.19184/ijc.2019.3.1.2.
[33] P. M. Aleover, A. Guillamon, and M. C. Ruiz, "A new randomness test for bit sequences," *Informatica*, vol. 24, no. 3, pp. 339–356, 2013, doi: 10.15388/Informatica.2013.399.

BIOGRAPHIES OF AUTHORS

Antonius Cahya Prihandoko is a senior lecturer at the Information Technology Dept., Faculty of Computer Science, the University of Jember, Indonesia. He is also the coordinator of the Network and Security research group under the department of Information Technology. He received his Bachelor degree in Mathematics Education from the University of Jember in 1992; and his Master of Applied Science in Computer Science and PhD in Information Technology both from James Cook University, Australia, in 1999 and 2015, respectively. His research interests are in Cryptography and Coding Theory. He can be contacted at email: antoniuscp.ilkom@unej.ac.id.

Dafik is a professor in Combinatorics, Graph Theory and Mathematics Education. He is the coordinator of the Combinatorics, Graph And Network Topology research group under the University of Jember. He received his Bachelor degree in Mathematics Education from the University of Jember in 1992; his MSc in Mathematics from University of Manchester Institute of Science and Technology (UMIST) U.K. in 1998; and his PhD in Mathematics Combinatorics from Ballarat University, Australia in 2007. His research interests are in Graph Theory, Combinatorics, Mathematics Education and Applied Mathematics. He can be contacted at email: d.dafik@unej.ac.id.

Ika Hesti Agustin is a lecturer at the Mathematics Dept., Faculty of Mathematics and Natural Science, the University of Jember, Indonesia. She received her Bachelor degree in Mathematics from the University of Jember in 2006; and her Master of Science in Mathematics from ITS Surabaya in 2013. Her research interests are in Graph Theory, Graph Labeling and Applied Mathematics. He can be contacted at email: ikahesti.fmipa@unej.ac.id.