Construction and Analysis of Key Generation Algorithms Based on Modified Fibonacci and Scrambling Factors for Privacy Preservation

Amiruddin Amiruddin[1,2], Anak Agung Putri Ratna[1], and Riri Fitri Sari[1]
(Corresponding author: Amiruddin Amiruddin)
Department of Electrical Engineering, Universitas Indonesia[1]
Jl. Margonda Raya, Pondok Cina, Beji, Kota Depok, Jawa Barat 16424, Indonesia
Sekolah Tinggi Sandi Negara, Bogor, Jawa Barat, Indonesia[2]
(Email: amir@stsn-nci.ac.id)
(Received Sept. 24, 2017; revised and accepted Apr. 12, 2018)

Abstract

Cryptographic key is the most important factor for supporting encryption of confidential data before it is transmitted in a communication network. A good cryptographic key has properties of random sequence and long period. For these purposes, a randomness capable and lightweight computing algorithm is required. The randomness capability and computation time of such an algorithm can be measured by using randomness test and algorithmic complexity analysis, respectively. In this paper, two models of key generation algorithm using the modified Fibonacci and scrambling factor were constructed. Such modification and scrambling factor are intended to support the randomness capability and low algorithmic complexity. The proposed key generation algorithms have been simulated and analyzed. The key generation algorithm Model 2 (called hereinafter "Scrambled Fibonacci-based") is better than Model 1 in term of randomness, despite both having similar linear algorithmic complexity, denoted by $\mathcal{O}(n)$.

Keywords: Cryptography; Key Generation; Randomness; Scrambled Fibonacci; Scrambling Factor

1 Introduction

Wireless communication system and its services have become an important component of modern life and society. An example of such a wireless network is the Internet of Things (IoT) that grows rapidly, nowadays, to support human beings need on information. However, due to the nature of the Radio Frequency (RF) spectrum used as shared transmission medium, wireless communications are essentially vulnerable and prone to interception [5]. The next generation of wireless communication systems should support applications with very low communication latency, availability, high reliability and security [27]. To protect from interception and to ensure the data confidentiality, many wireless systems use cryptographic systems with secret keys that are only available to the legitimate senders and recipients.

Various methods or approaches have been proposed to generate long and random encryption keys [36]. Each method or approach has advantages and disadvantages and cannot be applied to all different kinds of applications. Therefore, a key generation function should be tailored and adjusted to the characteristics of the applications that will use it. One important consideration in designing a key generation algorithm is its algorithmic complexity [24]. For applications in low-capacity devices for IoT, low complexity algorithms are required. Unfortunately, the existing key generation algorithms lack the measurement of their complexity.

Fibonacci sequence [10] which is a very famous series function in the field of mathematics can be used to generate encryption keys. It is a sequence of numbers where a number is found by adding up the two preceding numbers. Beginning with 0 and 1, the sequence goes as 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, and so forth. Written as a rule, the expression is $x_i = x_{i-1} + x_{i-2}$. Its lightweight operation and ability to save computing time are of the reasons for using it in the key generation function. Several cryptographic methods used Fibonacci sequence or its behavior for encryption application [9,12]. Applying Fibonacci is suitable for common areas that do not involve data privacy. However, for confidential data-based applications, it is necessary to make improvement on the Fibonacci function. Moreover, the operation used in Fibonacci produces a regular pattern (ascending or descending) that can be used as an entrance point to analyze the resulted key sequence. Therefore, it is necessary to modify the Fibonacci operation so that the resulted pattern becomes random and hardens the efforts to analyze it.

In this paper, a key generation algorithm based on the

modified Fibonacci and scrambling factor is constructed and analysed. The key generation function will be applied to the Internet of Things network with constrained devices that have limited storage, low computing power and energy. The contribution of this work is a construction and analysis of key generation algorithm based on Fibonacci and scrambling factor which satisfies the requirement for random and long period of key sequences.

The remaining of the paper is organized as follows. Section 2 gives a brief overview of the previous related works regarding cryptographic key generation methods. Section 3 describes the key generation definition and its performance measurement. The proposed method is described in Section 4. Section 5 discusses the simulation and the result and Section 6 closes the paper with the conclusion.

2 Related Works

As technology grows, research on cryptographic key management [17, 20, 22, 25, 32] continues to be done in various aspects including key generation [14, 18, 28, 29, 34, 35, 37], agreement [3, 6, 7, 13, 15, 33] or exchange [4], distribution [8], assignment [19], authentication [16], and update. However, we focus on efforts for key generation to be used on symmetric cryptosystems. Verma *et al.* [31] proposed a method for generating cryptographic key using biometrics with the help of fingerprint pattern. The algorithm generates key by extracting minutiae points and core point and the final key is obtained from the fingerprint image. Turakulovich *et al.* Turakulovich *et al.* [30] discussed comparative factors of the key generation techniques include randomness, key space, key space of biometric, entropy, measured entropy of biometric, convenience and cost, secure saving, update. However, they mostly discussed the factor for biometric-based key generation technique which is different from our proposed method.

Hossain *et al.* [11] proposed a One Time Key (OTK) generation technique based on User ID (UID) and password. The key generation method involves a server-side process to check for possible collisions between a new UID and a given UID to the previous client. This process takes a long time so it is not applicable for devices with limited storage and energy. Torre *et al.* [29] studied the practical performance of an enhanced channel-based key generation system with a very short roundtrip delay, allowing reciprocal channel assessment with increased accuracy. The reciprocal channel measurements performed by body-worn sensor nodes is used to extract encryption keys. Although the performance is slightly increased due to the shorter round-trip delay, further apparent non-reciprocity in the channel measurements can probably be attributed to inaccuracy of the received signal strength indication in the transceiver chip.

Tavangaran *et al.* [27] studied the secret key generation protocol for a compound Discrete Memoryless Multiple Sources (DMMS) with one-way communication in presence of an eavesdropper. The key generation protocol uses a two phase approach to achieve secret key. In the first step, the sender estimates his state and sends this along with other information which is obtained from his observation to the recipient. In the second step, the recipient uses this information including the estimated state of the sender to generate the secret key. However, the protocol has not been reported whether it has been implemented or not.

Al-Moliki *et al.* [2] enhanced the confidentiality of Visible Light Communication (VLC) networks by suggesting a new key generation protocol for optical Orthogonal Frequency Division Multiplexing (OFDM) schemes in an indoor environment. The keys are extracted from the bipolar OFDM samples produced from optical OFDM schemes. This approach which emphasizes the source of key generation differs from our proposed approach which emphasizes the process of the key generation.

Karimian *et al.* [14] proposed a novel approach of key generation that extracts keys from real-valued ECG features. However, this approach is only suitable for ECG-based applications although it can also be modified for other field applications.

In this research, we proposed new key generation algorithm based on modified Fibonacci and scrambling factor to support long periodicity and randomness of the generated key sequence. The research position of our proposed key generation algorithm among other algorithms is summarized in Table 1.

3 Key Generation and Performance Measurement

3.1 Key Generation

In the field of cryptography, key is the most urgent parameter for data encryption or decryption. By definition, a key is a sequence of a random string of bits created explicitly for scrambling and unscrambling data. Instances of cryptographic processes demanding the usage of keys include, inter alia, the transformation of plain text data into cipher text data (encryption) and vice versa (decryption), the computation and verification of a digital signature, the computation and verification of an authentication code from data, the computation of a shared secret that is used to obtain keying material, and the derivation of additional keying material from a key-derivation key.

There are two types of key, *i.e.* symmetric and asymmetric key. Symmetric key is a key used in a symmetric-key cryptographic algorithm which requires that the key must be kept secret. Asymmetric key is a key used with a public-key algorithm. In asymmetric key cryptography, there are two corresponding keys, *i.e.* private and public keys. A private key is a cryptographic key used with a public-key algorithm that must be kept secret and is uniquely associated with an entity that is authorized to use it. Public key is a key used with a public-key algorithm that may be made public and is associated with a

Table 1: Research position of key generation

Author	Method	Source	Implementation
Verma *et al.*, 2016	extraction	Biometric with finger-print pattern	Simulated on Matlab
Hossain *et al.*, 2016	generation	UID and password	Simulated on mobile phone
Torre *et al.*, 2017	extraction	Reciprocal channel measurements	Not reported
Tanvangaran *et al.*, 2017	generation	Compound: information, estimated state	Not yet
Al-Moliki *et al.*, 2017	extraction	Bipolar OFDM samples	Simulated on Monte Carlo
Karimian *et al.*, 2017	extraction	ECG value	-
Amiruddin *et al.*, 2017	generation	User input	Simulated on Matlab

private key and an entity that is authorized to use that private key.

Key generation is the process of generating keys for cryptographic purpose such as encryption [1,21]. A cryptographic key can be generated through a function in software or hardware with input parameters that can be obtained from various sources, *e.g.* channels used by each pair of users [28], phase fluctuations in fiber links, and values entered by the user.

3.2 Measurement

To measure the performance of the proposed key generation algorithm, several tests were used, *i.e.* key generation speed, key randomness, key periodicity, and algorithmic complexity analysis. The key generation speed was measured by recording the start time and finish time of the key generation process and then subtracting the start time from the finish time. The key randomness was measured by using autocorrelation function, while the key periodicity was manually analyzed. All of these kinds of key performance measurement were also used in [26]. The analysis of algorithmic complexity was measured by following the description and calculation example presented in [24].

4 Proposed Methods

We have constructed two models of new key generation algorithm based on the modified Fibonacci, described in detail as follows.

4.1 Model 1

In Model 1, modification made on Fibonacci series is the application of modulo number (annotated with c). In this model, the first key, K_1, is obtained by the result of $a\%c$. K_2 is obtained by the result of $b\%c$. K_i is obtained by the result of the addition of $(K_{i-1} + K_{i-2})\%c$. By applying a modulus number in this model, the ascending or descending pattern of the generated key sequences is reduced in periodicity. The key generation function of Model 1 is given in pseudocode form in Algorithm 1.

Algorithm 1 Key Generation (Model 1)

```
Begin
Initialize the parameters: a, b, n, c
Derive the 1st element of the key
K(1) ← mod(a, c)
Derive the 2nd element of the key
K(2) ← mod(b, c)
Derive the 3rd to n-th element of the key
for i = 3 to n do
    K(i) ← mod(K(i − 1) + K(i − 2), c)
end for
Output the key sequence, K
End
```

The simulation result showed that this model produces key sequences that have a better randomness level than those produced by the original Fibonacci. However, the randomness of the key sequences is not yet satisfied the randomness test. Therefore, we constructed another model of key generation algorithm, Model 2, by adding a scramble factor to the previous model, Model 1. This addition of scrambling factor was expected to make the generated key sequences more random to satisfy the randomness test.

4.2 Model 2

In Model 2, Fibonacci is modified to generate random and long period key sequences. Input parameters for the function are a, b, n and d as the first number, second number, key length, and modulus number, respectively. Through this model, the first key,K_1, is generated from the formula $(a*b-a)\%d$. K_2 is generated similarly from $(a*b-b)\%d$. For $i = 3 : n$, K_i is obtained from the addition of the two previous key, $K_{i-1} + K_{i-2}$, and a scrambling factor, $3 * i$ in formula of $K_{i-1} + K_{i-2} + 3 * i\%d$. Actually, the original Fibonacci is presented in the addition marked with a + symbol. However, such an addition can cause the resulted key sequences to have a low randomness and easy to be guessed (as described in Model 1). Therefore, in Model 2, a scrambling factor using a multiplication, $3 * i$, is added to improve the modified Fibonacci in generating random key sequences. The scrambling factor with the use of multiplication (*) involving an ever-changing

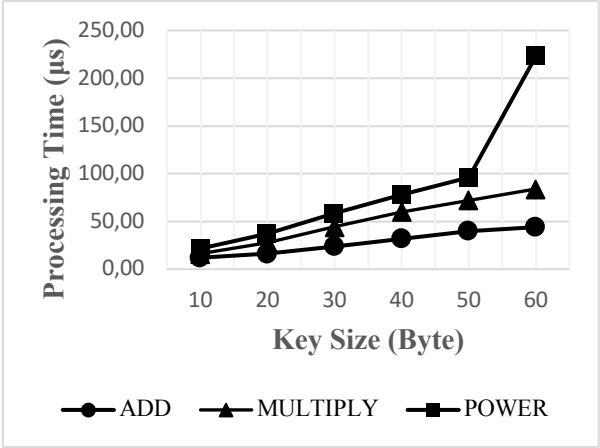

Figure 1: Processing time comparison of several operations

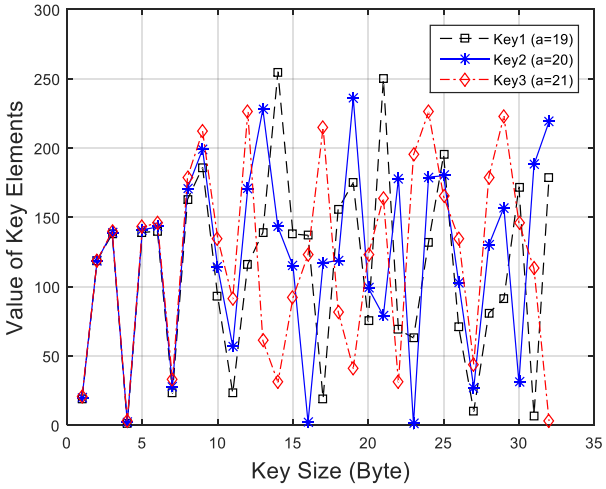

Figure 2: Key element plot of three key sequences with different initial value of parameter a of proposed algorithm Model 1

number, i, produces a non-patterned or random number.

The scrambling factor with the use of power $(^{})$ operation can also generate random numbers, but has a high computational overhead. The scrambling factor of the addition $(+)$ and subtraction $(-)$ produces a number that is ascending or descending pattern, while the division $(:)$ can cause infinite numbers if the divisor is a zero. In Figure 1 we show the experiment result of processing time comparison among the addition, multiplication, and power operations. Based on this result and the previous explanation, we use the multiplication in our scrambling factor in modifying the Fibonacci sequence.

The proposed generation algorithm of Model 2 is given in pseudocode form in Algorithm 2.

Algorithm 2 Key Generation (Model 2)

```
Begin
Initialize the parameters: a, b, n, c
Derive the 1st element of the key
K(1) ← mod(a * b − a, c)
Derive the 2nd element of the key
K(2) ← mod(a * b − b, c)
Derive the 3rd to n-th element of the key
for i = 3 to n do
    K(i) ← mod(K(i − 1) + K(i − 2) + 3 * i, c)
end for
Output the key sequence, K
End
```

The use of $*b-a$, $*b-b$, and $3*i$ in Algorithm 2 is intended to satisfy the randomness test and long periodicity of the generated key sequences as the requirements of good key as stated by Shannon [34].

The difference of operations among the original Fibonacci, modified Fibonacci Model 1, and Model 2 is presented in Table 2.

5 Results and Discussion

5.1 Model 1

The performance of the proposed key generation algorithms Model 1 and Model 2 were evaluated by simulation using Matlab software. The key generation algorithm of Model 1 does not meet the randomness test as in Figure 1, indicated by similar pattern of the three generated key sequences. By varying the value of parameter a with 19 (Key1), 20 (Key2), and 21(Key3), the three key sequences for up to 10 Bytes length have similar plot of key elements and this means that the key sequences are not random. Due to the unsatisfactory result of randomness of proposed Model 1 algorithm, we then proposed Model 2 algorithm.

5.2 Model 2

The key generation simulation used the Algorithm 2 by varying the parameters *i.e.* the first number (a), second number (b), and the key length (key size) (n) and let the modulus number (c) be remains in 256, as the maximum value of alphabet characters. In this simulation, we measured the key generation speed and randomness level, and compared with other algorithm.

5.3 Key Generation Speed

For measuring the speed of key generation, 1000 (one thousand) key sequences were generated. Each key sequence was then recorded at the start and the end time to get the processing time (finish time - start time), then looked for the average processing time by summing up all processing time and then divided by 1000. We can see on Figure 2 the time it took to generate key sequences of

Table 2: Different operations among fibonacci-based algorithms

Original Fibonacci	Modified Fibonacci Model 1	Modified Fibonacci Model 2
$U_1 = a$	$U_1 =$ mod (a, c)	$U_1 =$ mod $(a * b - a, d)$
$U_2 = b$	$U_2 =$ mod (b, c)	$U_2 =$ mod $(a * b - b, d)$
$U_i = U_{i-1} + U_{i-2}$	$U_i =$ mod $(U_{i-1} + U_{i-2}, c)$	$U_i =$ mod $(U_{i-1} + U_{i-2} + 3 * i, d)$

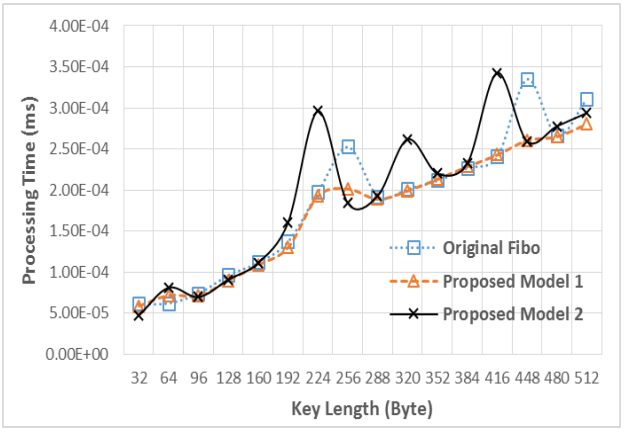

Figure 3: Comparison of key generation processing time (ms) for varying key length (byte)

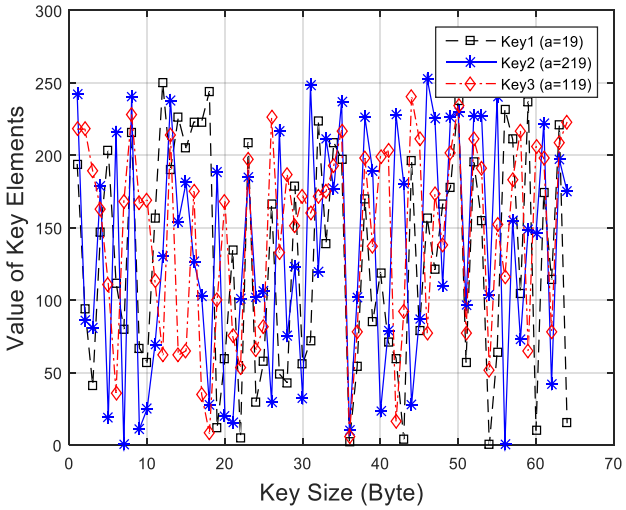

Figure 4: Plot of key elements of three key sequences of 64 Bytes with different value of parameter a: 19 (Key1), 219 (Key2), and 119 (Key3) of the proposed algorithm Model 2

varying key length sizes with the same initial parameter.

It showed that the key generation time increases along with the increase of the key length. However, the proposed Model 1 algorithm has the fastest processing time among all, while the proposed Model 2 has the lowest processing time in certain time but still has a relatively same processing time to the original Fibonacci algorithm. However, this can be explained as a consequence of using more functions in the proposed algorithm Model 2 than the number of functions in the Model 1 and the original Fibonacci. The use of more functions is intended to support the randomness of the generated key sequences. The increase in key length size is not linear with increasing generation time, since the ratio between the two is relatively decreasing as the key length increases.

5.4 Randomness of Key Sequence

Randomness tests are involved in a measuremnet to analyze the distribution of a set of data to see if it is uncorrelated or random. To test the randomness of the generated key sequences, we have simulated key generation by varying the value of variable a, *i.e.* 19, 219, 119 for Key 1, Key 2, and Key 3, as given in Figure 3. It appears that all the key sequences have different and irregular patterns indicating that all of the key sequences are uncorrelated or random.

By varying only the parameter values b from 119 to 124, we generated six key sequences and obtained their plot of key autocorrelation values as shown in Figure 4. The autocorrelation function (ACF), Rk, is calculated using the Equation (1) described in [52], when given a measurement of the variables $Y_1, Y_2, ..., Y_n$ on $X_1, X_2, ..., X_n$, by shifting (lag) of k.

$$R_k = \frac{\sum_{i=1}^{N-k} \left(Y_i - \overline{Y}\right)\left(Y_{i+k} - \overline{Y}\right)}{\sum_{i=1}^{N} \left(Y_i - \overline{Y}\right)^2} \quad (1)$$

As shown in Figure 4, the autocorrelation value of all 64-Byte key sequences with the defined lags of 1 to 20 is in the range between the upper boundary (0.2) and the lower boundary (-0.2) indicating that all of the key sequences are uncorrelated or random. The autocorrelation value for lags of 0 is 1 which means that the two compared key sequences are not random, since there is no key element shift (lag = 0) so there is no difference between the two key sequences. By using Pearson's correlation test, and varying the parameter of b, we have simulated the key generation for 200 key sequences and yielded the correlation coefficient between two key sequences as in Table 3.

The Pearson correlation is calculated using Equation (2),

$$rP = \frac{\sum_{i=1}^{N-k} \left(X_i - \overline{X}\right)\left(Y_{i+k} - \overline{Y}\right)}{\sum_{i=1}^{N} \left(X_i - \overline{X}\right)^2 \left(Y_i - \overline{Y}\right)^2} \quad (2)$$

Table 3: Correlation test (Pearson coefficient and significant test) between two key sequences generated by the proposed Model 2 algorithm

	Correlation between 2 key sequences				
	K_1, K_2	K_2, K_3	K_3, K_4	K_4, K_5	K_5, K_6
Prson Coef	-0.117	-0.085	-0.080	0.078	-0.222
Sig. test	0.535	0.653	0.673	0.681	0.237
Result	Pass	Pass	Pass	Pass	Pass
	Correlation between 2 key sequences				
	K_6, K_7	K_7, K_8	K_8, K_9	K_9, K_{10}	K_{10}, K_1
Prson Coef	-0.062	-0.157	0.056	-0.418	-0.050
Sig. test	0.744	0.406	0.765	0.022	0.790
Result	Pass	Pass	Pass	Pass	Pass

where rP, X, Y, $\overline{(X)}$, $\overline{Y}$, i, n are Pearson correlation, value of variable x, value of variable y, mean value of variable x, mean value of variable y, nth iteration, and number of elements.

A comparison of the randomness of several key sequences between the proposed algorithm and the Raphael algorithm is given in Figure 5, Figure 6, and Table 4. In Figure 5, it can be seen that the plot of key elements between the three key sets generated by the proposed algorithm is more random than that generated by Raphael's algorithm which yields relatively similar plots of key elements.

As shown in Figure 6, by varying only the value of parameter b, the correlation coefficient plot and the P-value of the key sequence generated by the proposed algorithm indicate a random sequence of keys because the value of P-value in average is greater than the correlation coefficient. While in contrast, Raphael's algorithm produces a key sequence that has P-value and a competing correlation coefficient, indicating that the key sequence has a correlation or not random. Further, as shown in Table 4, the comparison of two adjacent key sequences using the Spearman correlation test, indicating that the proposed algorithm passed all randomness tests, while the Raphael algorithm failed on all random tests. All of the above comparisons show the advantages of proposed Model 2 key generation algorithm over Raphael's algorithm.

Table 4: Correlation comparison of two adjacent key sequences of the proposed algorithm

Adj.Key	**Proposed algorithm**			**Raphael's algorithm**		
	Rho	**Pval**	**Test**	**Rho**	**Pval**	**Test**
K_1, K_2	-0.118	0.5356	**Pass**	0.253	0.178	Fail
K_2, K_3	-0.085	0.6533	**Pass**	0.453	0.012	Fail
K_3, K_4	-0.080	0.6739	**Pass**	0.375	0.041	Fail
K_4, K_5	0.078	0.6816	**Pass**	0.005	0.980	**Pass**
K_5, K_6	-0.222	0.2375	**Pass**	0.484	0.007	Fail
K_6, K_7	-0.062	0.7445	**Pass**	0.252	0.180	Fail
K_7, K_8	-0.157	0.4064	**Pass**	0.234	0.214	Fail
K_8, K_9	0.057	0.7659	**Pass**	0.390	0.033	Fail
K_9, K_{10}	-0.419	0.0222	**Pass**	0.103	0.588	**Pass**
K_{10}, K_1	0.051	0.7901	**Pass**	0.002	0.002	Fail

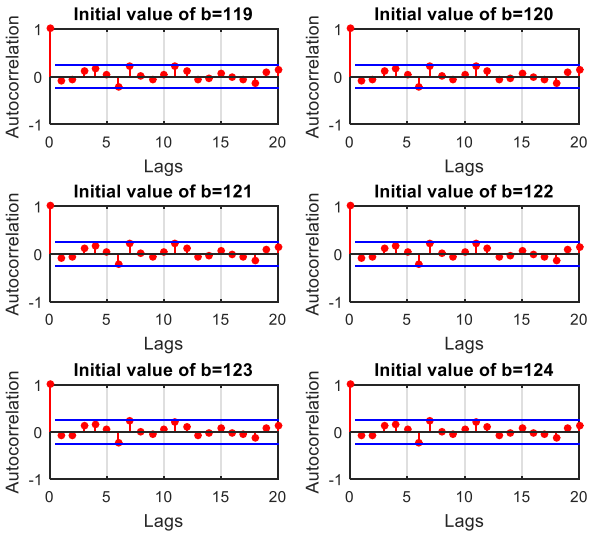

Figure 5: Plot of autocorrelation value of 64 Byte key sequences with different value of parameter b generated by the proposed Model 2 algorithm

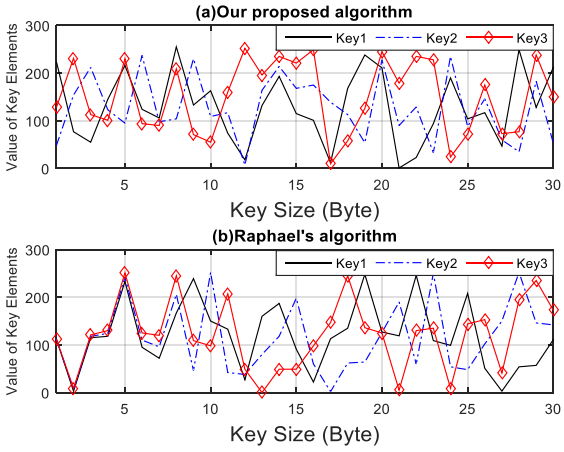

Figure 6: Comparison of key element plot among three key sequences generated by (a) proposed algorithm (b) Raphael's algorithm

5.5 The Periodicity of Key Sequences

Suppose a given key sequence A is 1 3 4 6 5 8 1 3 4 6 5 8 1 3 4 6 5 8 ... Then the period of key sequence A is 6 (the number of elements from 1 3 4 6 5 8 before experiencing the exact same loop). The longer the period, the better a key sequence. Since the proposed algorithm uses four parameters a, b, d, and n, and there is a scrambling factor that depends on the process iteration, the key sequences generated by the proposed algorithm has a long period equal to n (the key length to be generated, determined by the user). This character is called One Time Key or One Time Pad. Meanwhile, the algorithm used by Raphael *et al.* [23] also satisfies a long period, but since it does not use the modulus function, the resulting key sequence has a regular pattern of ascending or descending. So it does not meet the randomness test and it requires a larger memory to sum the number that will enlarge as the key length increases.

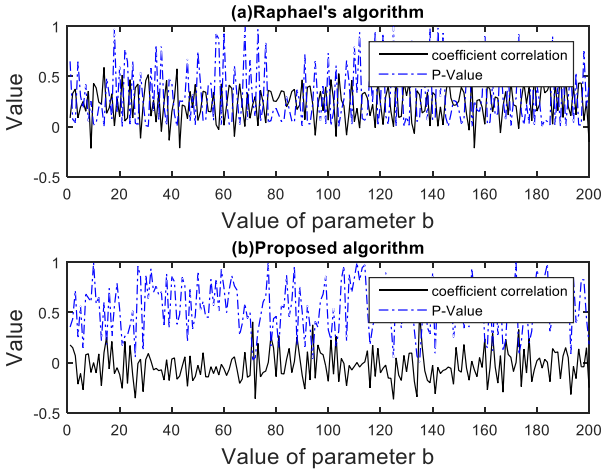

Figure 7: Comparison of correlation coefficient with different value of parameter b

5.6 Complexity of Algorithm

In the Key Generation algorithm Model 1 (see Algorithm 1), there are 10 Line of Codes (LOCs). However, LOCs beginning with a double forward slash (//) are only descriptions and not executed, and hence these parts are ignored and not counted in the analysis of the algorithmic complexity. LOC 3, 5, 7-9 are the processing part of the algorithm that will be calculated on its complexity. In LOC 3, there are 2 instructions *i.e.* mod (a, d) and assignment (=) of the variable K(1). In LOC 5, similar to LOC 3, there are 2 instructions, *i.e.* mod (b, c) and assignment (=) of the variable $K(2)$. LOC 7-9 is an incremental loop as many as $(n + 1 - 3)$ or $(n - 2)$ times with variable i as the counter. The assignment instruction $(i = 3)$ is executed before the loop.

Inside the loop, there are 1 comparison instruction $(i < n+1)$ and 1 incremental counter $(i++)$. In addition, there are also an assignment operation $(K(i) =$ mod $(K(i-1)+K(i-2), c))$ involving 1 sum operation $(+)$, 1 modulus operation (mod) and 1 assignment (=) of the variable $K(i)$. Thus, in LOC 7-9 there are 5 instructions repeated $(n - 2)$ times and 1 non-repeated instruction. Thus, in detail, the number of instructions in LOC 3 = 2, LOC 5 = 2, and LOC 7-9 = $5(n - 2) + 1$. Thus the complexity of the Algorithm 1 is $5(n - 2) + 1 + 2 + 2$ or $5(n - 2) + 5$ or $5n - 5$ or $\mathcal{O}(n)$.

In the Key Generation algorithm Model 2 (see Algorithm 2), there are 10 LOCs. LOCs beginning with double forward slash are ignored and not counted in the analysis of the algorithmic complexity. LOC 3, 5, 7-9 is the processing part of the algorithm that will be calculated on its complexity. In LOC 3, there are 4 instructions *i.e.* multiplication (*), subtraction (-), modulus (mod) and assignment (=) of the variable K(1). In LOC 5, similar to LOC 3, there are 4 instructions, *i.e.* multiplication (*), subtraction (-), modulus (mod) and assignment (=) of the variable K(2). LOC 7-9 is an incremental loop as many as $(n + 1 - 3)$ or $(n - 2)$ times with the variable i as the counter. The assignment instruction $(i = 3)$ is executed before the loop.

Inside the loop, there are 1 comparison instruction $(i < n + 1)$ and 1 incremental counter $(i++)$. In addition, there are also an assignment operation of $(K(i) =$ mod $(K(i-1) + K(i-2) + 3 * i, d))$ involving 2 addition operations $(+)$, 1 multiplication operation (*), 1 modulus operation (mod) and 1 assignment operation (=) of the variable $K(i)$. Thus, in LOC 5-7, there are 7 instructions repeated (n-2) times and 1 non-repeated instruction. Thus, in detail, the number of instructions in $LOC3 = 4$, $LOC5 = 4$, and LOC 7-9 = $7(n - 2) + 1$. Thus the complexity of the Model 2 algorithm is $7(n - 2) + 1 + 4 + 4$ or $7(n - 2) + 9$ or $7n - 5$ or $\mathcal{O}(n)$.

From the above analysis, it appears that both models of the proposed key generation algorithms have the same algorithmic complexity that is linear complexity expressed by $\mathcal{O}(n)$. However, only the Model 2 satisfies the randomness test.

6 Conclusions

We have utilized the Fibonacci sequence in constructing two proposed models of key generation algorithm, Model 1 (without scrambling factor) and Model 2 (with scrambling factor). The modification by adding a scrambling factor is intended to generate key sequences that satisfy randomness tests and long periodicity which have not been used in measuring the existing key algorithms found in recent literature. Simulation results of the two models indicate that the key generation time increases along with the increase of the key length, but the ratio between key generation time and key length relatively decreases. The randomness test results indicate that the key sequences generated by Model 1 do not meet the randomness test despite having relatively fast computation time.

Model 2 produces key sequences with accepted autocorrelation value (accepted random value) since it is always in the range between the upper boundary (0.2) and the lower boundary (-0.2). The result of algorithmic complexity analysis showed that both of key generation algorithms have a similar linear algorithmic complexity, expressed by $\mathcal{O}(n)$. However, considering all the performance measurement used in the present work, the proposed key generation algorithm Model 2 (Called hereinafter scrambled fibonacci) is the best compared to the proposed Model 1 and original Fibonacci-based Raphael algorithm. The future work of this research would be to implement the proposed Scrambled Fibonacci-based key algorithm in an encryption/decryption application in constrained devices to support security and privacy preservation in the Internet of Things.

References

[1] D. S. AbdElminaam, "Improving the security of cloud computing by building new hybrid cryptography algorithms," *International Journal of Electronics and Information Engineering*, vol. 8, no. 1, pp. 40–48, 2018.

[2] Y. M. Al-Moliki, M. T. Alresheedi, and Y. Al-Harthi, "Secret key generation protocol for optical OFDM systems in indoor VLC networks," *IEEE Photonics Journal*, vol. 9, no. 2, pp. 1–15, 2017.

[3] S. Arasteh, S. F. Aghili, and H. Mala, "A new lightweight authentication and key agreement protocol for internet of things," in *13th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC'16)*, pp. 52–59, 2016.

[4] M. Azrour, Y. Farhaoui, and M. Ouanan, "A new secure authentication and key exchange protocol for session initiation protocol using smart card," *International Journal of Network Security*, vol. 19, no. 6, pp. 870–879, 2017.

[5] S. Baksi, J. Snoap, and D. C. Popescu, "Secret key generation using one-bit quantized channel state information," in *IEEE Wireless Communications and Networking Conference (WCNC'17)*, pp. 1–6, 2017.

[6] M. Bayat, M. Aref, "An attribute based key agreement protocol resilient to KCI attack," *International Journal of Electronics and Information Engineering*, vol. 2, no. 1, pp. 10–20, 2015.

[7] T. Y. Chang, W. P. Yang, M. S. Hwang, "Simple authenticated key agreement and protected password change protocol", *Computers & Mathematics with Applications*, vol. 49, pp. 703–714, 2005.

[8] S. F. Chiou, M. S. Hwang, and S. K. Chong, "A simple and secure key agreement protocol to integrate a key distribution procedure into the DSS," *International Journal of Advancements in Computing Technology (IJACT'12)*, vol. 4, no. 19, pp. 529–535, 2012.

[9] K. Eguchi, K. Abe, M. Fujimoto, D. Yan, and I. Oota, "The development of a negative single-input/multi-output driver using a fibonacci-like converter," in *13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON'16)*, pp. 1–4, 2016.

[10] S. M. Farooq and S. H. S. Basha, "A study on fibonacci series generation algorithms," in *3rd International Conference on Advanced Computing and Communication Systems (ICACCS'16)*, pp. 1–5, 2016.

[11] S. Hossain, A. Goh, C. H. Sin, and L. K. Win, "Generation of one-time keys for single line authentication," in *14th Annual Conference on Privacy, Security and Trust (PST'16)*, pp. 686–689, 2016.

[12] C. H. Hsu, H. S. Dang, and T. A. T. Nguyen, "The application of fibonacci sequence and taguchi method for investigating the design parameters on spiral micro-channel," in *International Conference on Applied System Innovation (ICASI'16)*, pp. 1–4, 2016.

[13] Q. Jiang, S. Zeadally, J. Ma, and D. He, "Lightweight three-factor authentication and key agreement protocol for internet-integrated wireless sensor networks," *IEEE Access*, vol. 5, pp. 3376–3392, 2017.

[14] N. Karimian, Z. Guo, M. Tehranipoor, and D. Forte, "Highly reliable key generation from electrocardiogram," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 6, pp. 1400–1411, 2017.

[15] M. Lavanya and V. Natarajan, "Lightweight key agreement protocol for iot based on IKEv2," *Computers & Electrical Engineering*, vol. 64, pp. 580-594, 2017.

[16] C. C. Lee, M. S. Hwang, L. H. Li, "A new key authentication scheme based on discrete logarithms", *Applied Mathematics and Computation*, vol. 139, no. 2, pp. 343-349, July 2003.

[17] A. Lei, H. Cruickshank, Y. Cao, P. Asuquo, C. P. A. Ogah, and Z. Sun, "Blockchain-based dynamic key management for heterogeneous intelligent transportation systems," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1832–1843, 2017.

[18] C. T. Li, M. S. Hwang and Y. P. Chu, "An efficient sensor-to-sensor authenticated path-key establishment scheme for secure communications in wireless sensor networks", *International Journal of Innovative Computing, Information and Control*, vol. 5, no. 8, pp. 2107-2124, Aug. 2009.

[19] I. C. Lin, M. S. Hwang, C. C. Chang, "A new key assignment scheme for enforcing complicated access control policies in hierarchy", *Future Generation Computer Systems*, vol. 19, no. 4, pp. 457–462, May 2003.

[20] I. C. Lin, H. H. Ou, M. S. Hwang, "Efficient access control and key management schemes for mobile agents", *Computer Standards & Interfaces*, vol. 26, no. 5, pp. 423–433, 2004.

[21] L. Liu and Z. Cao, "Analysis of two confidentiality-preserving image search schemes based on additive homomorphic encryption," *International Journal of Electronics and Information Engineering*, vol. 5, no. 1, pp. 1–5, 2016.

[22] Z. Mahmood, H. Ning, and A. Ghafoor, "Lightweight two-level session key management for end user authentication in internet of things," in *IEEE International Conference on Internet of Things and IEEE Green Computing and Communications and IEEE Cyber, Physical and Social Computing and IEEE Smart Data*, pp. 323–327, 2016.

[23] A. J. Raphael and Dr. V. Sundaram, "Secured communication through fibonacci numbers and unicode symbols," *International Journal of Scientific and Engineering Research*, vol. 3 no. 4, no. 4, pp. 1–5, 2012.

[24] L. Rosyidi and R. F. Sari, "A practical approach for complexity analysis of autonomic internet of things protocol algorithm," in *19th International Symposium on Wireless Personal Multimedia Communications (WPMC'16)*, pp. 256–261, 2016.

[25] T. H. Sun and M. S. Hwang, "A hierarchical data access and key management in cloud computing," *ICIC Express Letters*, vol. 6, no. 2, pp. 569–574, 2012.

[26] Y. Suryanto, Suryadi, and K. Ramli, "A secure and robust image encryption based on chaotic permutation multiple circular shrinking and expanding," *Journal of Information Hiding and Multimedia Signal Processing-Ubiquitous International*, vol. 7 no. 4, pp. 697–713, 2016.

[27] N. Tavangaran, H. Boche, and R. F. Schaefer, "Secret-key generation using compound sources and one-way public communication," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 227–241, 2017.

[28] C. D. T. Thai, J. Lee, and T. Q. S. Quek, "Physical-layer secret key generation with colluding untrusted relays," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 1517–1530, 2016.

[29] P. Van Torre, Q. Van den Brande, J. Verhaevert, J. Vanfleteren, and H. Rogier, "Key generation based on fast reciprocal channel estimation for body-worn sensor nodes," in *11th European Conference on Antennas and Propagation (EUCAP'17)*, pp. 293–297, 2017.

[30] K. Z. Turakulovich and Y. B. Karamatovich, "Comparative factors of key generation techniques," in *International Conference on Information Science and Communications Technologies (ICISCT'16)*, pp. 1–3, 2016.

[31] I. Verma and S. Jain, "Biometric based key-generation system for multimedia data security," in *3rd International Conference on Computing for Sustainable Global Development (INDIACom'16)*, pp. 864–869, 2016.

[32] P. Vijayakumar, M. Azees, A. Kannan, and L. Jegatha Deborah, "Dual authentication and key management techniques for secure data transmission in vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, no. 4, pp. 1015–1028, 2016.

[33] F. Wu, L. Xu, S. Kumari, X. Li, J. Shen, K. K. R. Choo, M. Wazid, and A. K. Das, "An efficient authentication and key agreement scheme for multi-gateway wireless sensor networks in iot deployment," *Journal of Network and Computer Applications*, vol. 89, no. Supplement C, pp. 72–85, 2017.

[34] S. Xiao, Y. Guo, K. Huang, and L. Jin, "High-rate secret key generation aided by multiple relays for internet of things," *Electronics Letters*, vol. 53, no. 17, pp. 1198–1200, 2017.

[35] H. Zhang, Y. Liang, L. Lai, and S. Shamai Shitz, "Multi-key generation over a cellular model with a helper," *IEEE Transactions on Information Theory*, vol. 63, no. 6, pp. 3804–3822, 2017.

[36] J. Zhang, T. Q. Duong, A. Marshall, and R. Woods, "Key generation from wireless channels: A review," *IEEE Access*, vol. 4, pp. 614–626, 2016.

[37] J. Zhang, B. He, T. Q. Duong, and R. Woods, "On the key generation from correlated wireless channels," *IEEE Communications Letters*, vol. 21, no. 4, pp. 961–964, 2017.

Biography

Amiruddin Amiruddin, a lecturer at Sekolah Tinggi Sandi Negara (STSN), Indonesia. He received Bachelor?s degree in Informatics from Universitas Budi Luhur. He received his Master?s in Information Technology from the Faculty of Computer Science, Universitas Indonesia (UI). He just received a Ph.D degree from the Electrical Engineering Department, Faculty of Engineering, Universitas Indonesia (UI).

Anak Agung Putri Ratna, a Senior Lecturer of Computer Engineering at Electrical Engineering Department, Faculty of Engineering, Universitas Indonesia (UI). She graduated with a BSc in Electrical Engineering from UI, a Master in Engineering from the Waseda University Japan, and a Ph.D in Electrical Engineering from UI.

Riri Fitri Sari, a Professor of Computer Engineering at the Electrical Engineering Department, Faculty of Engineering, Universitas Indonesia (UI). She graduated with a BSc in Electrical Engineering from UI, a Master in Human Resources Management from the Atmajaya University Jakarta and an MSc in Software Systems and Parallel Processing from the Department of Computer Science, University of Sheffield, UK, funded by British Council Chevening Award, and a Ph.D in Computer Networks from the School of Computing, University of Leeds, UK. Her current main teaching and research area includes Computer Network, Internet of Things (IoT), Cloud Computing, Vehicle Ad Hoc Networks, and ICT implementation.