International Journal of Communication Networks and Information Security (IJCNIS) Vol. 9, No. 3, December 2017

New Key Generation and Encryption Algorithms for Privacy Preservation in Mobile Ad Hoc Networks

Amiruddin[1,2], Anak Agung Putri Ratna[1], and Riri Fitri Sari[1]

[1]Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia, Indonesia
[2]Sekolah Tinggi Sandi Negara, Bogor, Jawa Barat, Indonesia

***Abstract*—**Mobile Ad Hoc Networks (MANETs) get widespread applications along with the evolving technologies. However, MANETs are at risk due to the shortage of security mechanisms. In this paper, we propose new algorithms for key generation and encryption for privacy preservation in MANETs. Our key generation algorithm modified Fibonacci sequence by adding scrambling factors to generate random key sequences with required length but incurred low computational overhead, whereas the encryption/decryption algorithm utilizes the One Time Pad (OTP) system by adding scrambling factors for data confidentiality which satisfies the randomness, diffusion, and confusion tests. Simulation of the proposed algorithms was conducted using Matlab and NS-2. Experiment results showed that the proposed algorithms produced random key sequences and Ciphertexts. Through several tests i.e. speed, correlation and autocorrelation, diffusion, and confusion tests, the simulation result showed the superiority of our algorithms over the other algorithms. For the proof of concept, our algorithms have been simulated in the network simulator, and the result showed that along with the increase of the number of nodes, the throughput of the network increased, while the delay is relatively constant around 6000 μs for 20 up to 70 nodes.

***Keywords*—** Encryption/Decryption, Fibonacci, Key Generation, MANET, One-Time Pad, Privacy Preservation, Scrambling factor

1 Introduction

Mobile Ad Hoc Networks (MANETs) are an emerging type of wireless networking, in which mobile nodes associate on an extemporaneous or ad hoc basis. MANETs are both self-forming and self-healing, enabling peer-level communications between mobile nodes without dependency on centralized resources or fixed infrastructure. The mobile nodes communicate with one another by wireless radio links without the use of any pre-established fixed communication network infrastructure or centralized administration, such as base stations or access points, and with no human intervention.

MANETs have been applied in wide area such as for military operations, disaster management, and others. However, MANETs are vulnerable due to the lack of security mechanisms. Thus, providing security feature for privacy preservation issues in MANETs is a very challenging task due to the characteristics of such a dynamic topology or nomadic environment. Data privacy preservation is one of the most challenging issues in MANETs specifically in transferring sensitive data through multi-hop routing. The unwanted revealing of data can cause a privacy breach and can be used in commencing an attack. Present research works realized the data privacy by using data transformation and data perturbation approach.

Recent studies related to the MANETs security apply cryptographic methods to support privacy preservation i.e. the data confidentiality, during data transformation [1], transmission [2, 3] as well as during the route discovery [4]. However, the existing methods have high complexity algorithms that cause a high computing overhead.

We focused on data transformation using an encryption method. One Time Pad is a cryptographic method that is simple yet proven mathematically with a very high security. It is unbreakable when it is applied correctly following the defined rules as mentioned in [5]. The disadvantage of OTP method is the need for a large storage for keys to support that the length of OTP key must be equal to the length of the plaintext to be encrypted. Researchers use different methods to extend the range of the keys as required without causing the increase of space requirement. For such a case, considering its simple operation, a modified type of Fibonacci sequence is used in our key generation algorithm.

In this work, the Fibonacci sequence is modified by adding scrambling factors to generate required random key sequences in a way of low computational overhead. In addition, we propose a new algorithm utilizing the modified OTP with scrambling factors for data encryption and decryption processes that supports the data confidentiality which satisfies the randomness, diffusion, and confusion tests. These algorithms are intended for use in MANETs.

Our main contributions in this work are 1) a new key generation algorithm that is based on the modified Fibonacci sequences with scrambling factors for generating long and random key sequences. 2) a new encryption/decryption algorithm utilizing modified OTP with scrambling factor for satisfying the randomness, diffusion and confusion tests.

The remainder of this paper is organized as follows. Section 2 briefly discusses the related works. Section 3 provides detailed description of the proposed method. Simulations and results and conclusion are given in Section 4 and 5, respectively.

2 Related Works

MANETs have become a key communication technology in tactical activities such as the movement of troops and equipment in military field, as well as rescue and evacuation during natural disasters. However, threats to MANETs are also increasing, e.g. the malicious nodes may attack the location information and routing messages. Therefore, such information/privacy preservation needs to be maintained and therefore researchers have proposed various ways such as trust management [6, 7], routing schemes [8], authentication, integrity and encryption [9].

In the literature, there are several works to address security issues c.q. privacy preservation on MANET. Ullah et al.[6] proposed a selection scheme of trust recommendations based on a factor of dissimilarity to protect the trust management system from dishonest recommendations. Wei et al.[10] proposed a trust management scheme based on uncertain reasoning to improve the security of MANET. However, trust-based security methods can't guarantee the protection of data / information transmission within the network.

Yang [11] proposed a key distribution scheme with efficient group communication based on identity-based broadcast encryption method. Nonetheless, this method is only limited to the key distribution and has not reached the stage of securing the data / information. Tselikis et al.[12] proposed a cooperative framework towards self-protection and self-organization. For key agreement with anonymity, that paper showed that various key agreement protocols can be applied to dynamically establish pair-wise or group keys. This method has also not reached the security of data/information transmission.

Khubalkar et al.[13] proposed a secure data transmission using Hash-based Message Authentication Code-Secure Hash Algorithm version 1 (HMAC-SHA1) for authentication, AES for confidentiality, and SHA1 for integrity of message. However, the use of some cryptographic methods for MANET networks can lead to heavy computing processes. Secure privacy preservation methods with light computing process are required.

An OTP-based cryptographic method is known as secure and lightweight process. OTP requires a random key, without repetition, and is the same length as Plaintext [14]. The encryption process happens by XOR-ing the Plaintext with the Key and results in a Ciphertext with the logic operation (1),

Ciphertext = XOR (Plaintext, Key) (1)

where XOR denotes the bitwise operations with exclusive-OR (XOR). Upon reception, the Ciphertext is XOR-ed with the Key to reveal the original Plaintext with the following formula,

Plaintext = XOR (Ciphertext, Key) (2)

OTP is theoretically unbreakable [5] if used properly according to the following four rules, i.e. the key length must be equal to the plaintext length; the key should be completely random; each key only used once; and the key is kept secret.

Boateng et al.[15] implemented an efficient encryption protocol based on OTP to reduce the overhead associated with the time and energy in wireless sensor networks. Jeyamala et al. [16] utilized the OTP for image encryption with chaotic approach. Srikantaswamy et al.[14] demonstrated that the OTP can be used as an efficient encryption scheme involving arithmetic and logic operations. Zheng et al. [17] implemented the OTP-based encryption for electrocardiogram (ECG) on Implantable Medical Devices (IMD). The OTP keys are generated independently by each IMD in accordance with the needs of long data to be encrypted hence the key distribution is unneeded. Mezaal et al. [18] proposed ecnryption algorithm based on OTP using logical operations, XOR and AND. However, all these OTP-based algorithm lack of the measurement of randomness, diffusion, and confusion properties. Therefore, our paper proposes a modified OTP-based privacy preservation method that holds randomness, diffusion, and confusion testing.

Recent literature showed that considering its unique characteristics, Fibonacci sequence is used widespread in the modern era as in the application of lighting system [19], in the prediction of fluid flow and heat transfer [20], automotive [21], as well as cryptography application [22, 23]. Fibonacci sequence is a sequence of integer numbers characterized that each element in it is the sum of the two preceding elements, except for the first two elements. The traditional Fibonacci sequence $F(i)$ is given by (3).

$$F(i)=\begin{cases} a & ;i=1 \\ b & ;i=2 \\ F(i-1)+F(i-2) & ;i\geq 3 \end{cases} \quad (3)$$

where a and b are initial and second number, and i is the i^{th} number of the Fibonacci sequence.

Boateng et al. [15] proposed an efficient encryption protocol using Fibonacci for wireless sensor network. Raphael et al. [24] proposed secure communication using the original Fibonacci sequences and Unicode symbol, where the Fibonacci sequences can be assumed as the key generator for that algorithm. These Fibonacci-based algorithms lack of the randomness measurement. Therefore, we propose modified OTP and Fibonacci-based privacy preservation methods that satisfies randomness, diffusion, and confusion testing.

Table 1 List of symbols

Symbol	Remark
$\boldsymbol{a, b}$	1st , 2nd element of Fibonacci Sequence, respectively
$\boldsymbol{d}$	Modulo number
$\boldsymbol{n}$	Maximum iteration or key length
$\boldsymbol{i}$	ith iteration
$\boldsymbol{U}_{\text{i}}$	ith element of Fibonacci sequence
$\boldsymbol{K}_{\text{i}}$	ith element of key sequence
$\boldsymbol{P}_{\text{i}}$	ith element of Plaintext
$\boldsymbol{C}_{\text{i}}$	ith element of Ciphertext
$\boldsymbol{mod}$	Modulus function
$\boldsymbol{XOR}$	exclusive-OR
$\boldsymbol{and}$	AND function
$+, -$	Addition, Substraction, respectively

3 Proposed Method

In this section, we described in detail the proposed algorithms, key generation and encryption/decryption algorithms.

3.1 List of Symbols

The symbols used in our algorithms are given on the Table 1.

3.2 Key Generation Function

For key generation in our algorithm, the Fibonacci sequence is modified to produce key sequence as long as required. The input parameters for the function are $\boldsymbol{a}, \boldsymbol{b}, n$ and $\boldsymbol{d}$ for the first value, the second value, the key length, and modulo number, respectively. Through the function, the first key, $\boldsymbol{K_1}$, is obtained by inputting $a * b - a$ and d into the sub-function $\boldsymbol{mod}$. $\boldsymbol{K_2}$ is obtained in similar way by inputting $a * b - b$ and d into the sub-function $\boldsymbol{mod}$. For $i = 3{:}\,n$, $\boldsymbol{K_i}$ is obtained by inputting the addition result of $\boldsymbol{K}_{(i-1)}+\boldsymbol{K}_{(i-2)}+3 * i$ along with $\boldsymbol{d}$ into sub-function $\boldsymbol{mod}$. Actually, the nature of Fibonacci sequence occurs on the sub-function ***addition*** marked by a $+$ symbol. However, such an addition leads to resulting in low randomness of key sequence and easy to guess. To make a better randomness of key sequences, we modified the original Fibonacci by adding the scrambling factor, $3 * i$.

Our proposed key generation process as previously described is then formulated to make a general equation as in (4), where $\boldsymbol{mod}$ is a modulo sub-function, $\boldsymbol{a}$ is the initial value of the Fibonacci sequence, $\boldsymbol{b}$ is the 2nd value of Fibonacci sequence, $\boldsymbol{d}$ is modulo number, n is the key length.

$$K_i = \begin{cases} \mathrm{mod}(a*b-a,d) & i=1 \\ \mathrm{mod}(a*b-b,d) & i=2 \\ \mathrm{mod}\big(K(i\text{-}1)+K(i\text{-}2)+3*i,\ \mathrm{d}\big) & 3\le i\le n \end{cases} \qquad (4)$$

Algorithm1: Key Generation

```
Input : initial values of a, b, n, d;
Output: Key sequence, K
K(1)← mod(a*b-a,d);
K(2)← mod(a*b-b,d);
For i ← 3 to n
  K(i)← mod(K(i-1)+K(i-2)+3*i,d);
```

The use of $* b - a$, $* b - b$, and $3 * i$ in the proposed key generation algorithm in (4) is intended as scrambling factors for randomness and long period in generated key sequences. Based on (4), we created key generation algorithm as in **Algorithm1**. The difference between the original Fibonacci and our key generation algorithm is given on Table 2.

Table 2 The difference between the original Fibonacci and our proposed key generation algorithm

Original Fibonacci	Proposed Key Generation Algorithm
$U_1 = a$	$U_1 = mod(a * b - a, d)$
$U_2 = b$	$U_2 = mod(a * b - b, d)$
$U_i = U_{i-1} + U_{i-2}$	$U_i = mod(U_{i-1} + U_{i-2} + 3 * i, d)$

3.3 Encryption Function

We propose encryption algorithm as in (5-7).

$$C_1 = mod\ (XOR(XOR\ (P_1, K_1), and\ (K_1, n), 256) \qquad (5)$$
$$C_2 = mod\ (XOR(K_2, P_2) + XOR\ (C_{i\text{-}1}, XOR\ (i, n), 256) \qquad (6)$$
$$C_i = mod\ (XOR(K_i, P_i) + XOR(C_{i\text{-}1}, XOR(i, n), 256);\ i > 1 \qquad (7)$$

If equation (5-7) is processed in one equation, it becomes:

$$C_i = \begin{cases} \mathrm{mod}(xor(xor(P_1, K_1), and(K_1, n), 256); & i=1 \\ \mathrm{mod}(xor(P_i, K_i) + xor(C_{i-1}, xor(i,n)), 256); & 2\le i\le n \end{cases} \qquad (8)$$

C_i is ith Ciphertext, P_i is ith Plaintext, K_i is ith Key, n is the key length, and i is counter. Pesudo-code of the encryption is given in **Algorithm2**.

Algorithm2: Encryption

```
Input: K, Msg
Output: Cipher
n ← length(Msg);
Cipher(1)← mod(bitxor(bitxor(Msg(1),
  Key(1)),bitand(Key(1),n)),256);
for i← 2 to n
Cipher(i)← mod(bitxor(Msg(i),Key(i))+
  bitxor(Cipher(i-1),bitxor(i,n)),256);
```

In our proposed encryption algorithm, the changes made on the original OTP encryption which differentiate our proposed algorithm are the triple XOR functions and the scrambling factors ($\boldsymbol{i}$ and $\boldsymbol{n}$), instead of using one single XOR function as in the original OTP. The use of triple XOR and scrambling factors is expected to make a great change on Ciphertext if one or some bit(s) of the key sequence or Plaintext is/are changed. Consequently, this will satisfy the diffusion and confusion properties. The difference between the original OTP and our modified OTP is given on Table 3.

Table 3 The difference between the original OTP and our modified OTP algorithm for encryption

Orig. OTP	Our Proposed Modified OTP
$C_i = xor(P_i, K_i)$	$C_i = \begin{cases} \mathrm{mod}(xor(xor(P_1, K_1), and(K_1, n), 256); & i=1 \\ \mathrm{mod}(xor(P_i, K_i) + xor(C_{i-1}, xor(i,n)), 256); & 2\le i\le n \end{cases}$

3.4 Decryption Function

The following is the operation of the decryption process occurs when i=1 to i=n.

$$P_1 = mod\ (xor\ (xor\ (C_1, K_1), and\ (K_1, n), 256) \qquad (9)$$
$$P_2 = mod\ (xor\ (C_2 - xor\ (C_1, xor\ (i, n)), K_2), 256) \qquad (10)$$
$$P_i = mod\ (xor\ (C_i - xor\ (C_{i\text{-}1}, xor\ (i, n)), K_i), 256) \qquad (11)$$

The decryption process is formulated in (12) and pseudo-coded as in **Algorithm3**.

$$P_i = \begin{cases} \mathrm{mod}(xor(xor(C_1, K_1), and(K_1, n), 256); & i=1 \\ \mathrm{mod}(xor(C_i - xor(C_{i-1}, xor(i,n), K_i), 256); & 2\le i\le n \end{cases}$$
(12)

It can be proven that the decryption algorithm of P_1 is reversible of the encryption algorithm of C_1 through the following processes (13-17) by substituting C_1 with (5).

$$P_1 = mod(xor(xor(C_1, K_1), and(K_1, n)), 256) \qquad (13)$$
$$P_1 = ((C_1\ xor\ K_1)\ xor\ (K_1\ and\ n))\ mod\ 256) \qquad (14)$$

$$P_1 = (((P_1\ xor\ K_1)\ xor\ (K_1\ and\ n)\ xor\ K_1\ xor\ (K_1\ and\ n))\ mod\ 256 \qquad (15)$$
$$P_1 = P_1\ xor\ K_1\ xor\ K_1\ xor\ (K_1 \text{ and } n)\ xor\ (K_1\ and\ n)\ mod\ 256 \qquad (16)$$
$$P_1 = P_1 \qquad (17)$$

Algorithm3: Decryption

```
Input: Key, Cipher
Output: Msg
n ← length(Cipher);
Msg(1)← mod(bitxor(bitand(Key(1),n),
  bitxor(Cipher(1),Key(1)),256);
for i← 2 to n
```

```
6   Msg(i)← mod(bitxor(Cipher(i)-
    bitxor(Cipher(i-1),bitxor(i,n),
    Key(i)),256);
```

It also can be proven that the decryption process of P_i is reversible of the encryption process of C_i, for $i = 2$ to n, through the following equations (18-22), by substituting Ci in (18) with (7).

$$P_i = mod(xor(C_i - (xor(C_{i-1}, xor(i,n)), K_i), 256) \quad (18)$$
$$P_i = (C_i - C_{i-1}\ xor\ (i\ xor\ n))\ xor\ K_i)\ mod\ 256 \quad (19)$$
$$P_i = (P_i\ xor\ K_i + (C_{i-1}\ xor\ (i\ xor\ n) - (C_{i-1}\ xor\ (i\ xor\ n))\ xor\ K_i)\ mod\ 256 \quad (20)$$
$$P_i = (P_i\ xor\ K_i + 0)\ xor\ K_i\ mod\ 256 \quad (21)$$
$$P_i = P_i \quad (22)$$

The difference between the original OTP and our proposed modified OTP algorithm for decryption is given on Table 4.

Table 4 Difference between the original and the proposed modified OTP algorithm for decryption

Original OTP	Our Proposed Modified OTP
$P_i = xor(C_i, K_i)$	$Pi = \begin{cases} mod(xor(xor(C_1, K_1), and(K_1, n), 256); & i=1 \\ mod(xor(C_i - xor(C_{i-1}, xor(i,n), K_i), 256); & 2 \le i \le n \end{cases}$

4 Simulation and Results

We have conducted the performance simulations of our algorithms using Matlab R2015a. For the proof of concept, we also have applied the proposed algorithms in network simulation using NS-2.35. The detailed parameters used in each experiment are given in the following sub sections.

4.1 Key Generation Simulation

We have simulated our proposed key generation algorithm by varying the initial conditions i.e. initial value (a), second value (b), and key size (n) and letting the modulo number (d) fixed as 256. We measured the key generation speed, the randomness and period of the key sequence, and also compared with an existing algorithm.

4.1.1 The speed of key generation

For measuring the key generation speed, we have generated several key sequences with different sizes with 1000 iterations and calculated the generation time by summing up all the generation time and dividing it with 1000.

In Figure 1 we can see the time needed for generating key sequences with different size, provided that the keys have the same initial value of Fibonacci function as previously mentioned. It is shown that the key generation run time increased along with the increase of the key size. However, the increase on key size is not linear to the increase on generation time since the ratio of the key size to the generation time relatively decreases along with the increase of the key size. The performance comparison between our proposed algorithm and an existing Fibonacci-based key generation algorithm, Raphael's algorithm [24], is given on Table 5. The performance of our proposed key generation algorithm looks slower than that of the other algorithm. However, this is understandable since our algorithm has more functions, which are intended for generating random key sequences.

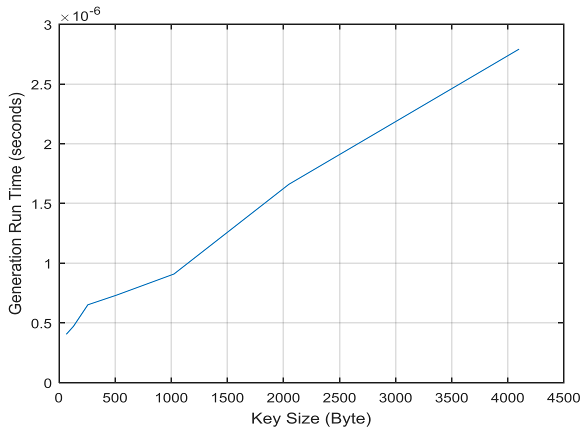

Figure 1 Key generation run time by 1000 experiments

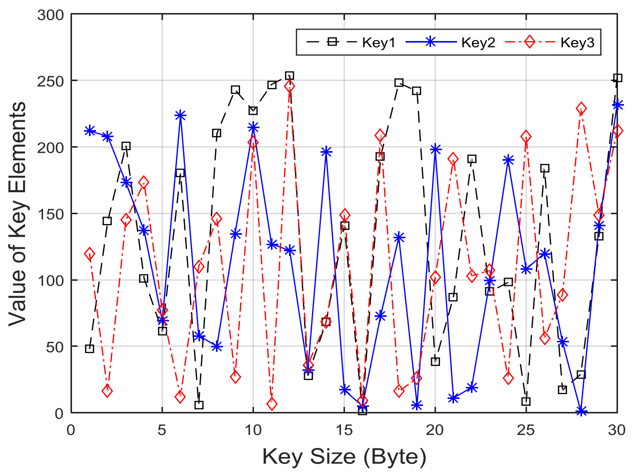

Figure 2 Plot of key element of three key sequences with value of a=208 (Key1), 108 (Key2), and 8(Key3) for size of 30 Bytes

To measure the randomness of the key sequences, we have simulated the key generation by varying only the initial value of a which are 208, 108, 8 for Key1, Key2, and Key3, respectively, as in Figure 2. It showed that all the key sequences produce different and random pattern. This pointed out that the key sequences are random.

Table 5 Key Generation Run Time

Key Size (KByte)	**Key Generation Run Time (seconds) for 1000 experiments**	
	Raphael's Algorithm [24] at a=1, b=32	**Our Algorithm at a=1, b=32**
25	$1.73x10^{-05}$	$1.76\ x10^{-05}$
50	$2.77\ x10^{-05}$	$2.98\ x10^{-05}$
75	$3.92\ x10^{-05}$	$4.15\ x10^{-05}$
100	$4.98\ x10^{-05}$	$5.24\ x10^{-05}$
125	$5.98\ x10^{-05}$	$6.43\ x10^{-05}$

4.1.2 The randomness of the key sequence

By varying only the value of ***b*** from 111 to 115, we obtained the plot of autocorrelation of the key sequences as in Figure 3. Autocorrelation function (ACF) is computed using Equation (23) which is described in [25], given a measurement of $Y1, Y2, \ldots, Yn$ at $X1, X2, \ldots, Xn$, with lag k.

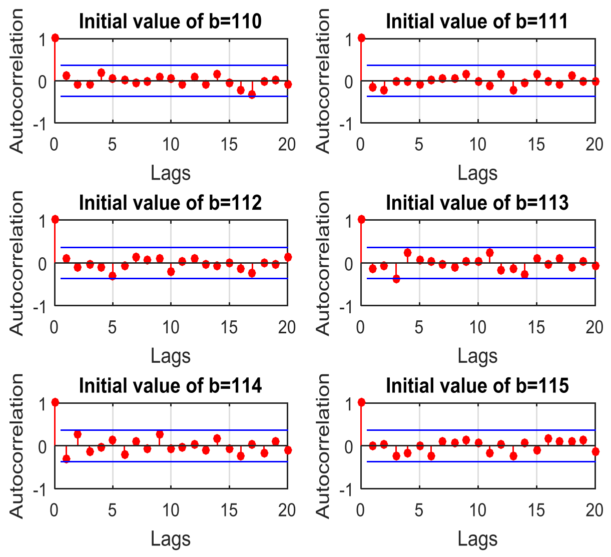

Figure 3 Plot of key sequence autocorrelation with different value of ***b***

Table 6 Correlation test between two key sequences

	Correlation Between Two Keys				
	K1, K2	**K2, K3**	**K3, K4**	**K4, K5**	**K5, K6**
Spearman Coeff	-0.1177	-0.0852	-0.0801	0.0781	-0.2224
Significant Test	0.5356	0.6533	0.6739	0.6816	0.2375
Result	Pass	Pass	Pass	Pass	Pass
	Coorelation Between Two Keys				
	K6,K7	**K7,K8**	**K8,K9**	**K9,K10**	**K10,K1**
Spearman Coeff	-0.0621	-0.1573	0.0567	-0.4185	0.0507
Significant Test	0.7445	0.4064	0.7659	0.0222	0.7901
Result	Pass	Pass	Pass	Pass	Pass

$$Rk = \frac{\sum_{i=1}^{N-k} (Yi - \overline{Y})(Yi + k - \overline{Y})}{\sum_{i=1}^{N} (Yi - \overline{Y})\hat{}2} \qquad (23)$$

As we can see on Figure 3, the autocorrelation value of all the key sequences with lag of 1 to 30, lie between the upper bound and lower bounds which indicates that the key sequences are uncorrelated or random. The autocorrelation for lag 0 is 1 which means the two compared key sequences are not random, since there is no shifting (zero shifting) of elements in the key sequences causing no difference between the two compared key sequences.
Using Spearman’s Correlation Test, by varying only the value of parameter ***b***, we have simulated key generation for 200 key sequences and produced correlation coefficient (Spearman Coefficient) and P-Value (Significant Test) as in Table 6 and Figure 4, using (24) and (25),

$$r_s = 1 - \left(\frac{6 \sum D^2}{n(n^2 - 1)} \right) \qquad (24)$$

$$t = r_s \sqrt{\frac{n-2}{1 - r_s{}^2}} \qquad (25)$$

where $rs,\ D, t, n$ are the Spearman correlation, different value, significant test, and number of element, respectively. On Figure 4, the experiment result for 200 key sequences by varying the parameter ***b*** produced P–Values which in average are always greater than the correlation coefficient, which indicate that the key sequences have no correlation or random.

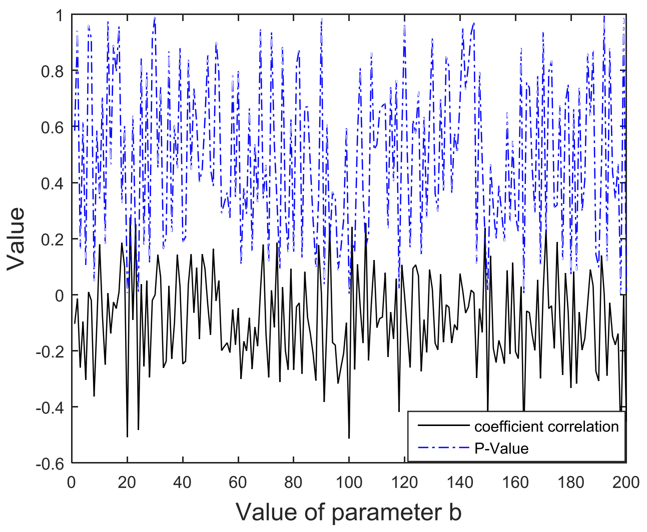

Figure 4 Correlation coefficient of key sequence by varying the value of b

Randomness comparisons between key sequence of our algorithm and the Raphael’s algorithm are depicted on Figure 5, Figure 6, and Table 7.

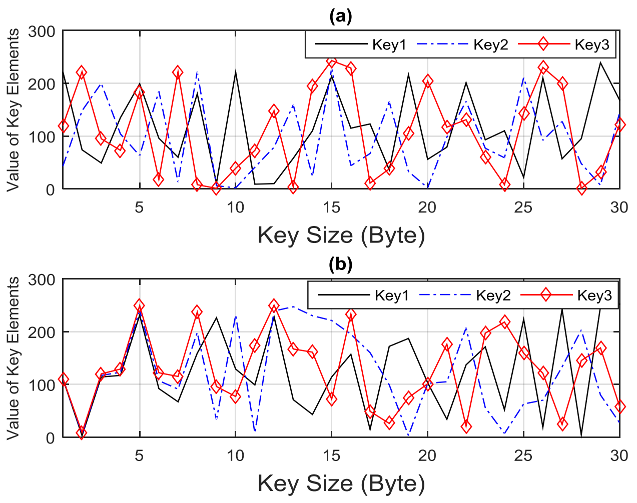

Figure 5 Comparison of key elements plot between three key sequences. (a) Proposed algorithm (b) Raphael's algorithm

On Figure 5,we can see that the plot of the key elements of three key sequences of our algorithm is higher than that of the comparing algorithm, where the comparing algorithm plotted relatively similar plot of the key elements.

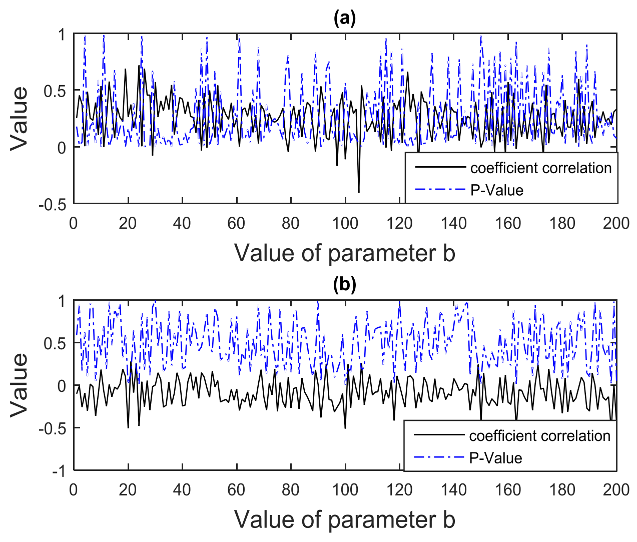

Figure 6 Comparison of coefficient correlation by varying value of b (a) Raphael's algorithm (b) Our algorithm

As shown in Figure 6, by varying only the parameter of ***b*,** the plot of coefficient correlation and P-value between our key generation algorithm and the comparing algorithm showed that our algorithm produced random key sequences which is indicated by the P-value that is in average greater than the correlation coefficient. In contrast, the comparing algorithm produced the competing P-value and correlation coefficient which means that the key sequence is correlated or not random. Furthermore, as shown in Table 7, comparing two adjacent key sequences using Spearman’s correlation test, our algorithm passed all the randomness test, while the other algorithm almost failed all. All these comparisons showed the strength of our algorithm over the other algorithm.

Table 7 Correlation comparison between two key sequences

Corrd Keys	Our Proposed Alg			Raphaels Alg		
	Rho	**Pval**	**Res.**	**Rho**	**Pval**	**Res.**
K1, K2	-0.118	0.5356	**Pass**	0.253	0.178	Fail
K2, K3	-0.085	0.6533	**Pass**	0.453	0.012	Fail
K3, K4	-0.080	0.6739	**Pass**	0.375	0.041	Fail
K4, K5	0.078	0.6816	**Pass**	0.005	0.980	**Pass**
K5, K6	-0.222	0.2375	**Pass**	0.484	0.007	Fail
K6, K7	-0.062	0.7445	**Pass**	0.252	0.180	Fail
K7, K8	-0.157	0.4064	**Pass**	0.234	0.214	Fail
K8, K9	0.057	0.7659	**Pass**	0.390	0.033	Fail
K9,K10	-0.419	0.0222	**Pass**	0.103	0.588	Pass
K10, K1	0.051	0.7901	**Pass**	0.535	0.002	Fail

4.1.3 The key period

Period of key sequence is the number of key elements without repeating all the key elements in the same order. The longer the period, the better the key sequence. Because our key generation algorithm uses four parameters ***a*, *b*, *d***, and ***n***, and there is a scrambling factor which depends on the iteration of the process, the key sequences generated by our algorithm have a long period which is equal to the parameter ***n*** (the key length). This property is called One Time Key or One Time Pad. Meanwhile, the algorithm used by Raphael et al. [24] also satisfy this. However, because it uses no modulo function, the generated key sequences have a regular ascending pattern that dissatisfies the randomness requirement, and requires a large memory for the sum of two numbers which are quite large.

4.2 Encryption Function Simulation

For encryption simulation, we provided, in advance, some plaintext files with various sizes of 64, 128, 256, 512, 1024, 2048, 4096 Bytes as the input along with the key files with respective sizes (e.g. 64 Bytes key file for 64 Bytes plaintext file) to conduct encryption simulation. We measured the encryption run time, the randomness of the generated Ciphertext, diffusion and confusion test, and also compared with the performance of existing OTP-based algorithm.

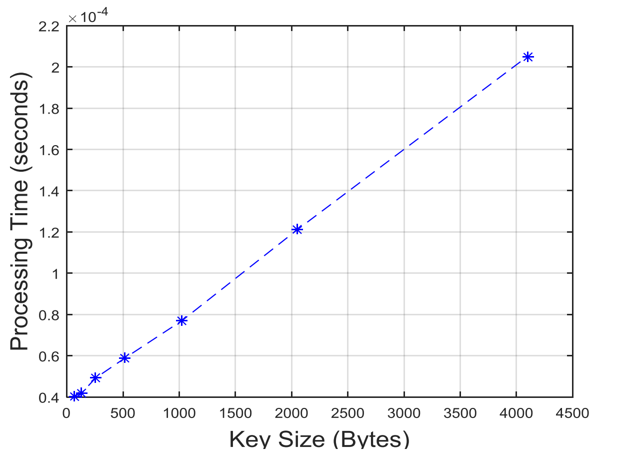

Figure 7 Encryption time for different file sizes

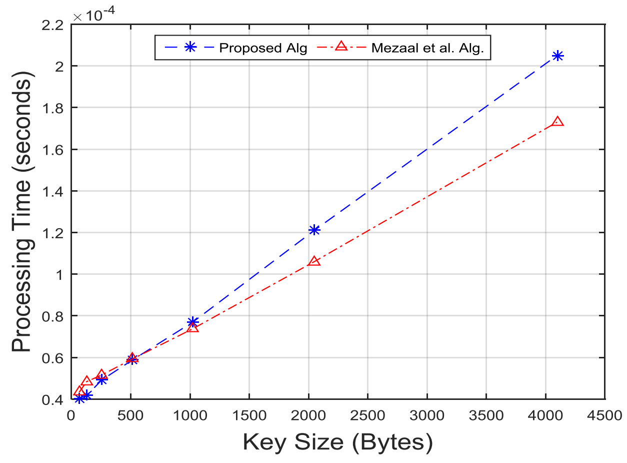

Figure 8 Comparison on encryption run time

4.2.1 The speed of encryption

On Figure 7 we can see the run time of encryption with different size of file. It is pointed that the increase of the file size to twice leads to the increase of the encryption time to less than twice, which means the greater the file size, the better the ratio of the encryption run time to the file size.

The encryption speed comparison between our algorithm and Mezaal’s algorithm [18] by varying the key size of 64 to 4096 Bytes as in Figure 8, showed that the encryption run time in average was similar. Both algorithms were in same speed for up to 1000 Bytes, while for more than it, Mezaal's algorithm was faster than our algorithm. However, this is understandable since our encryption algorithm uses more functions which are intended to satisfy the randomness, diffusion and confusion properties of the Ciphertexts.

4.2.2 *The randomness of Ciphertext*

The Ciphertext generated by our algorithm is random as shown on example plot of correlation coefficient and P-value of the Ciphertext for file size of 64 and 128 KB as shown in Figure 9. It is clear from the figure that the P-value of our algorithm is always greater than the correlation coefficient which means that the generated Ciphertext is uncorrelated or random.

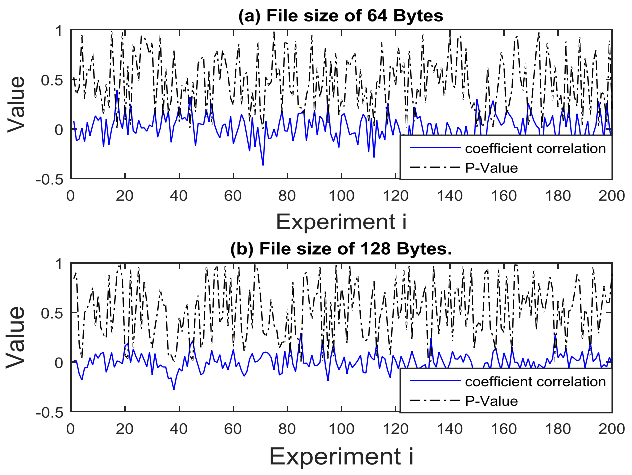

Figure 9 Coefficient correlation of example Ciphertexts for file size of 64 and 128 KB

Comparison of the randomness using Spearman's Correlation Test between Ciphertext of our algorithm and that of Mezaal's algorithm as in Figure 10 showed that our algorithm produce random Ciphertext since the P-value is always greater than the correlation coefficient. In contrast, Mezaal's algorithm showed that the P-value is almost always smaller than the correlation coefficient, which means that the Ciphertext is correlated or not random.

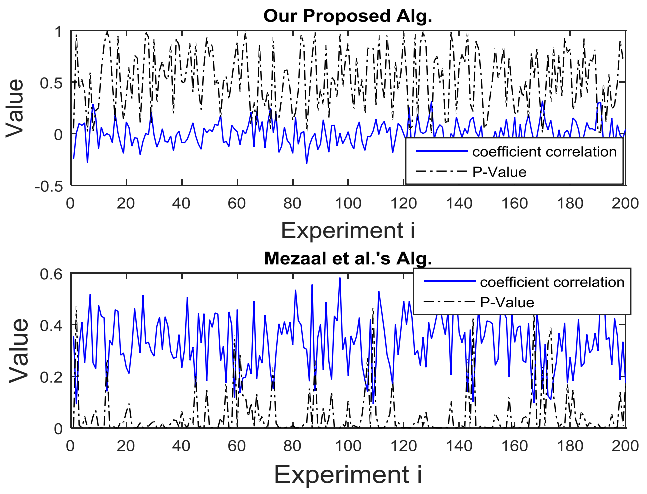

Figure 10 Comparison on Ciphertext coefficient correlation between our algorithm and Mezaal's algorithm

4.2.3 *Diffusion and confusion test*

A good encryption algorithm satisfy sensitivity measurement (the diffusion and confusion test). Difussion test measures the randomness of Ciphertext when one or more bits of the key is/are changed, whereas confusion test measures the randomness of Ciphertext when one or more bits of the Plaintext is/are changed. For these intentions, we have simulated the generation of four Ciphertexts with the size of 30 Bytes using four key sequences by differing/changing one up to three bits of the last three key sequences (Key1, Key2, Key 3 was changed 1, 2, and 3 bit, respectively) from the first key sequence (Key, changed 0 bit). On Figure 11 we can see an example plot of Ciphertext elements generated using each key sequences have different and random patterns. This means that the proposed encryption algorithm holds the diffusion test.

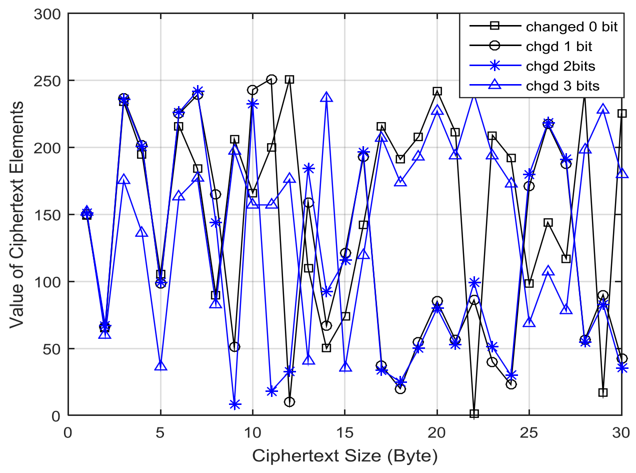

Figure 11 Plot of the Ciphertext elements if some bits of Key are changed**.**

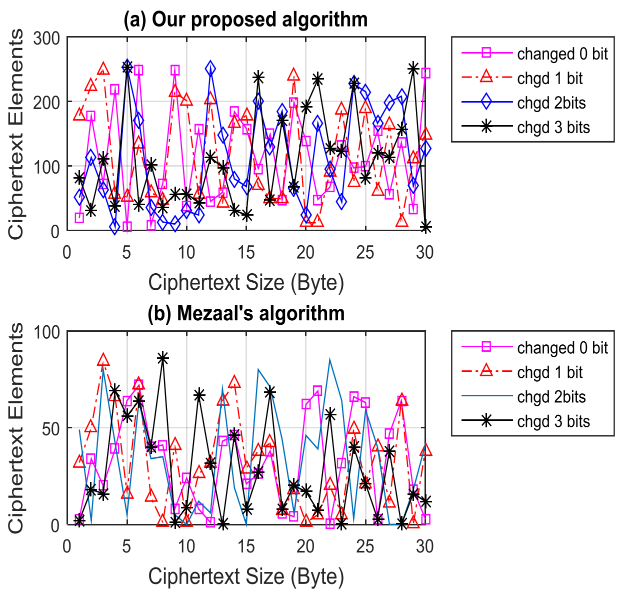

Figure 12 Comparison of diffusion on Ciphertext when changing 1-3 bits of Key Sequence

For comparing the diffusion test between the proposed algorithm and Mezaal's algorithms, we have simulated the generation of four Ciphertext with key size of 30 Bytes. Changing one or more bits on the key sequence using our algorithm produced different and random Ciphertext as in

Figure 12, while Mezaal's algorithm is less random, which is indicated that the key elements mostly locate below of the number 50. On confusion test comparison as shown on Figure 13, our algorithm is random since the Ciphertext elements plot of the three Ciphertexts have different and random pattern, whereas the Mezaal's algorithm is not random since the three Ciphertexts have similar key elements plot

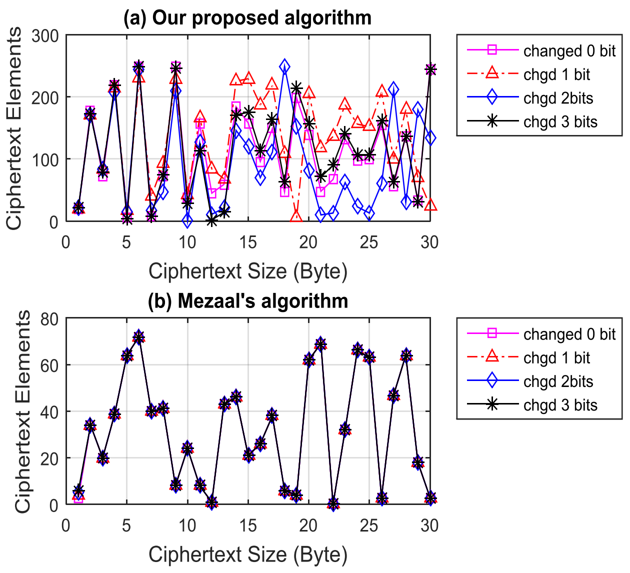

Figure 13 Comparison of confusion on Ciphertext when changing 1-3 bits of Plaintext

4.3 Decryption Function Simulation

For decryption simulation, we used the encrypted files obtained on the encryption simulation and key sequences that are generated using the same initial values as that used in key generation for the previous encryption simulation.

The processing time of encryption and decryption using the same keys is then compared and is given on Figure 14. It shows that the encryption time is relatively longer compared to the decryption time. This difference could be caused by the different function, where addition (+) is used in the encryption and substraction (−) in the decryption process.

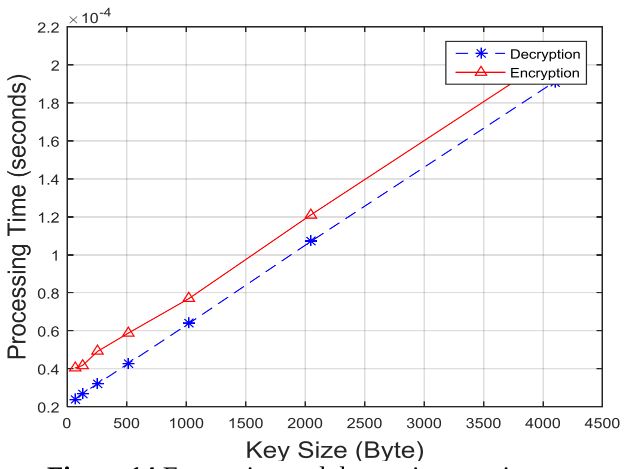

Figure 14 Encryption and decryption run time

The decryption speed comparison between our algorithm and Mezaal's algorithm is given on Figure 15. It is shown that the our algorithm is faster for key size up to 2400 Bytes. For key size of 2400 to 4096 Byte, our algorithm is slower than the comparing algorithm. However, this is reasonable since our algorithm uses more functions than the comparing algorithm.

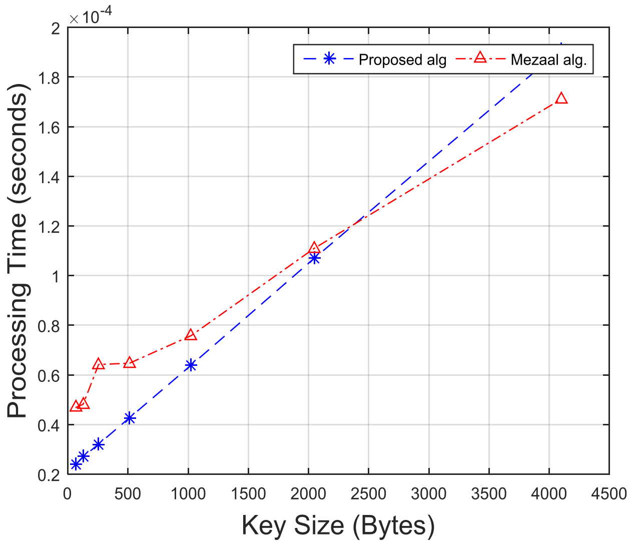

Figure 15 Comparison on decryption run time

4.4 Network simulation

For the proof of concept, we have conducted network simulations for applying and evaluating the proposed algorithms using NS-2.35, with parameters as in Table 8. We set the first half of the number of nodes as class 1 and the rest as class 2. For example, network animation for 70 nodes is given on Figure 16.

As the scenario, at a certain time after network initialization, each node on class 1 generated key sequence and used it for encrypting a message before sent it to a node on class 2 and vice versa. Upon receiving the encrypted message, each node will generate key sequence and use it for decrypting the encrypted message and announce the successful decryption, otherwise the integrity is not ensured.

Table 8 Network simulation parameters

Parameters	Values
Simulator	Ns-2.35
channel type	Wireless Channel
radio-propagation	Two Ray Ground
network interface type	Wireless Phy
MAC type	Mac/802.11
interface queue type	Drop Tail
link layer type	LL
antenna model	Omni Antenna
max packet in ifq	50
number of mobile nodes	10,20,30,40,50
routing protocol	AODV
X dimension of topography	1000
Y dimension of topography	700
time of simulation end	10 seconds

The network simulation for various number of nodes using our proposed algorithms on MANET resulted in throughput and delay as in Figure 17 and Figure 18, respectively, using formula and definition described in [26].

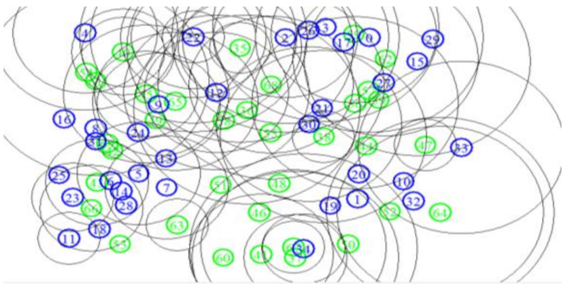

Figure 16 Screenshot of network animation for 70 nodes

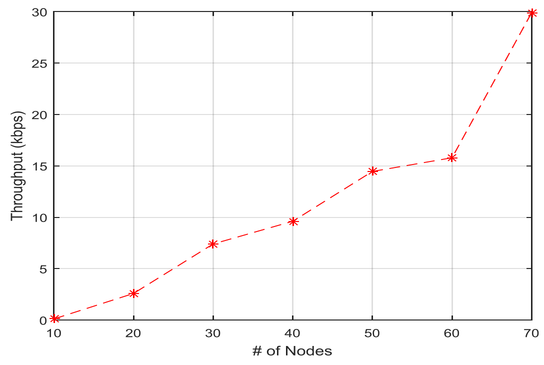

Figure 17 Network throughput for various number of nodes involved

It is shown that the throughput of the network increased along with the increase of the number of nodes involved on the network for 20 to 70 nodes. The reason of the increase of the throughput can be caused by the more number of nodes sending data. The network delay for 20 up to 70 nodes is relatively constant around 6000 µs, while for 10 nodes, the network delay is around 4000 µs. The increase of network delay for the increasing number of nodes is affected by the nature of AODV protocol that stores the route and link failure which causes long end-to-end latency in transmitting packets across the network.

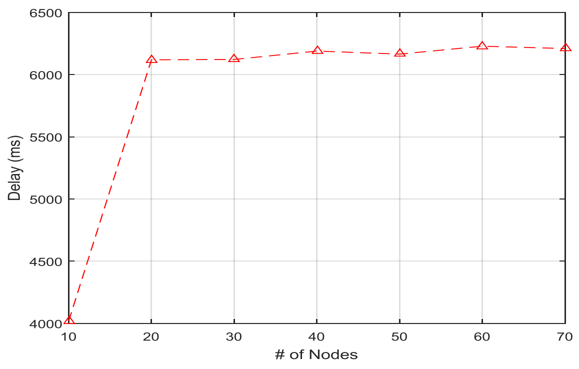

Figure 18 Network delay for various number of nodes involved

5 Conclusions

We have constructed new algorithms of key generation and encryption by utilizing a modified Fibonacci sequence and OTP method for privacy preservation on MANETs. We firstly designed the key generation function using modified Fibonacci and adding scrambling factors to generate key sequences which satisfy long period and randomness. Then, the encryption/decryption function was designed utilizing modified OTP and adding scrambling factors to satisfy the diffusion and confusion test of the Ciphertext. The key generation algorithm was then incorporated into the encryption/decryption function.
Experiment results showed that the processing time of key generation increased along with the increase of the key size. The randomness test result showed that the key sequences with various sizes produced autocorrelation values which are in the range of accepted value (between upper (0.2) and lower (-0.2) bound). The generated Ciphertexts also satisfy the randomness, diffusion, and confusion test. The comparison between our proposed algorithm and the existing algorithm showed the superiority of our algorithm. Furthermore, it is pointed out from the network simulation, along with the increase of the number of nodes, the throughput increased, while the delay is relatively constant around 6000 ms for 20 up to 70 nodes.

References

[1] B. S. Bhati and P. Venkataram, "Data privacy preserving scheme in MANETs," in 2014 World Congress on Internet Security (WorldCIS), London, UK, pp. 22-23, 2014.

[2] P. Vijayakumar, M. Azees, A. Kannan, and L. J. 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, pp. 1015-1028, 2016.

[3] S. Sivaranjani and S. Rajashree, "Secure data transfer in MANET using hybrid cryptosystem," in 2014 International Conference on Information Communication and Embedded Systems (ICICES), Chennai, India, pp. 1-5, 2014.

[4] W. Liu and M. Yu, "AASR: Authenticated Anonymous Secure Routing for MANETs in Adversarial Environment," IEEE Transaction on Vehicular Technology, vol. 63 No. 9, pp. 4585-4593, 2014.

[5] C. E. Shannon, "Communication theory of secrecy systems," Bell Syst. Tech, vol. 28 No. 4, pp. 656-715, 1949.

[6] Z. Ullah, M. H. Islam, A. A. Khan, and S. Sarwar, "Filtering Dishonest Trust Recommendations in Trust Management Systems in Mobile Ad Hoc Networks," International Journal of Communication Networks and Information Security (IJCNIS) vol. Vol. 8, No. 1, April 2016, pp. 18-30, 2016.

[7] N. Sirisala and C. S. Bindu, "Recommendations Based QoS Trust Aggregation and Routing in Mobile Adhoc Networks," International Journal of Communication Networks and Information Security (IJCNIS) vol. Vol. 8 No. 3, pp. 215-220, 2016.

[8] A. Sahnoun, A. Habbani, and J. E. Abbadi, "EEPR-OLSR: An Energy Efficient and Path Reliability Protocol for Proactive Mobile Ad-hoc Network Routing," International Journal of Communication Networks and Information

Security (IJCNIS) vol. Vol. 9, No. 1, p. 22-29, 2017.

[9] B. Muthusenthil and S. Murugavalli, "Privacy preservation and protection for cluster based geographic routing protocol in MANET," Wireless Networks, vol. 23 No. 1, pp. 79-87, 2017.

[10] Z. Wei, H. Tang, F. R. Yu, M. Wang, and P. Mason, "Security Enhancements for Mobile Ad Hoc Networks With Trust Management Using Uncertain Reasoning," IEEE Transaction on Vehicular Technology, vol. 63 no. 9, pp. 4647-4658, 2014.

[11] Y. Yang, "Broadcast encryption based non-interactive key distribution in MANETs," Journal of Computer and System Sciences, vol. 80 No. 3, pp. 533-545, 2014.

[12] C. Tselikis, S. Mitropoulos, and C. Douligeris, "A cooperative framework towards self-protection and self-organization in Mobile Ad Hoc Networks," in 2015 IEEE International Conference on Communications (ICC), London, UK, pp. 5922-5927, 2015.

[13] A. S. Khubalkar and D. L. R. Ragha, "Security Enabled DSR for Establishing Symmetric Key and Security in MANETs," in 10th International Conference on Wireless and Optical Communication Networks, Bhopal, India, pp. 1-5, 2013.

[14] S. G. Srikantaswamy and D. H. D. Phaneendra, "Enhanced One-Time Pad Cipher with More Arithmetic and Logical Operations with Flexible Key Generation Algorithm," International Journal of Network Security & Its Applications (IJNSA), vol. 3 no. 6, pp. 243-248, 2011.

[15] K. Boakye-Boateng, E. Kuada, and E. Antwi-Boasiako, "Efficient encryption protocol for wireless sensor networks using one-time pads," in 2016 18th Mediterranean Electrotechnical Conference (MELECON), Lemesos, Cyprus, pp. 1-6, 2016.

[16] C. Jeyamala, S. GopiGanesh, and G. S. Raman, "An image encryption scheme based on one time pads - A chaotic approach," in 2010 International Conference on Computing Communication and Networking Technologies (ICCCNT), , TamilNadu, India, pp. 1-6, 2010.

[17] G. Zheng, G. Fang, R. Shankaran, and M. A. Orgun, "Encryption for Implantable Medical Devices Using Modified One-Time Pads," IEEE Access, vol. 3, pp. 825-836, 2015.

[18] Y. S. Mezaal, D. A. Hammood, and M. H. Ali, "OTP encryption enhancement based on logical operations," in 2016 Sixth International Conference on Digital Information Processing and Communications (ICDIPC), Beirut, Lebanon, pp. 109-112, 2016.

[19] 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 2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Chiang Mai, Thailand, pp. 1-4, 2016.

[20] 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 2016 International Conference on Applied System Innovation (ICASI), Okinawa, Japan, pp. 1-4, 2016.

[21] T. Arafune, Y. Kobayashi, S. Shibuya, and H. Kobayashi, "Fibonacci sequence weighted SAR ADC algorithm and its DAC topology," in 2015 IEEE 11th International Conference on ASIC (ASICON), Chengdu, China, pp. 1-4, 2015.

[22] F. Wang, J. Ding, Z. Dai, and Y. Peng, "An Application of Mobile Phone Encryption Based on Fibonacci Structure of Chaos," in 2010 Second World Congress on Software Engineering (WCSE), Tianjin, China, pp. 97-100, 2010.

[23] S. Liu, G. Qi, and X. A. Wang, "Fast and Secure Elliptic Curve Scalar Multiplication Algorithm Based on a Kind of Deformed Fibonacci-Type Series," in 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), Krakow, Poland, pp. 398-402, 2015.

[24] A. J. Raphael and D. V. Sundaram, "Secured Communication through Fibonacci Numbers and Unicode Symbols," International Journal of Scientific & Engineering Research, vol. 3 no. 4, pp. 1-5, 2012.

[25] J. Zhang, A. Marshall, R. Woods, and T. Q. Duong, "Secure key generation from OFDM subcarriers' channel responses," in 2014 IEEE Globecom Workshops (GC Wkshps), Austin, TX, USA, pp. 1302-1307, 2014.

[26] P. Li, Y. Fang, and J. Li, "Throughput, Delay, and Mobility in Wireless Ad Hoc Networks," in INFOCOM, 2010 Proceedings IEEE, San Diego, CA, USA, pp. 1-9, 2010.