Enhancing the Randomness of Symmetric Key using Genetic Algorithm

Afiqah Zahirah Zakaria, Sofia Najwa Ramli, Chuah Chai Wen, Cik Feresa Mohd Foozy, P. Siva Shamala Palaniappan, Nur Fadzilah Othman

Abstract: ***The focus of network security is to provide the secure, effective and private communication between the sender and the receiver. To achieve the aim of high security of sending information, the improvement in cryptography is needed to make sure the protection of the information against unauthorized users. Symmetric-key cryptography satisfies the constraint of resources in computational complexity performances, but it offers weak security since it is not resilient against physical compromise. One of the way to overcome the issue is by providing a cryptographic key that is strong, hard to break and almost unpredictable by the intruder. As the advancement of technology in Artificial Intelligence (AI), Genetic Algorithm (GA) is implemented to generate the best-fit key in symmetric-key cryptography. Due to natural selection of GA process, the generated key is found to be the most random and non-repeating as possible. Moreover, the fitness test shows the average fitness value of a generated key increases when the key length increases.***

Index Terms: Best-fit Key, Genetic Algorithm, Randomness, Symmetric-key

I. INTRODUCTION

Technology advancement is changing by leaps. Internet and technology rules our lives. Solving problems as easy as by browsing and clicking the solution in the internet via smart devices and laptops. Unfortunately, valid users as well as others can access the personal and private information without proper and strong security measurement. This gives the opportunity to the unauthorized user to steal the data or information for misuse.

Today, platform such as online banking, e-commerce and online shopping critically apply cryptography to provide the confidentiality of the user's information [1]. There are two processes involved in cryptography, which are encryption and decryption. For both processes, they use keys, either private key or public key [2]. Symmetric key cryptography involves only one key for encryption and decryption process, while, asymmetric key requires two keys, a public key for encryption and a private key for decryption process [3].

Revised Manuscript Received on May 22, 2019.

Afiqah Zahirah Zakaria, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

Sofia Najwa Ramli, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

Chuah Chai Wen, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

Cik Feresa Mohd Foozy, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

P. Siva Shamala Palaniappan, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Johor, Malaysia

Nur Fadzilah Othman, Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Melaka, Malaysia

Key is a root of cryptography process. It is a required parameter to make the transmitted information undecipherable. It specifies the particular transformation of the given plaintext into a cipher and vice versa [4]. For this reason, the key represents a shared secret between two or more parties that can be used to maintain private information [3]. Thus, it shows that symmetric key has its own weaknesses where it needs a secure channel to exchange the secret key. Due to it is a sharing key, the sender has to ensure that the exchanged remains in secret way. Besides, it has a problem that the origin and authenticity of message cannot be guaranteed especially when there is a dispute. This is because the sender and receiver use the same key, so it quite difficult to verify the message came from a particular user [5].

Despite the fact of the disadvantages of symmetric key, it also has its own good side. The benefit of symmetric key is the type of encryption is easy to carry out. The senders just need to specify and share the secret key, then, the process of encryption and decryption messages can be begin. Other than that, the use of symmetric key prevents widespread message security compromise [6]. For every communication with every different party, a different secret key is used. If a key is compromised, only the messages between a particular pair of sender and receiver are affected. The others communications are still secure.

Therefore, the key particularly in symmetric cryptography needs to be random and unique so that it is hard to break by unauthorized user [3]. Consequently, the best-fit key is needed to ensure the technique to protect information are implemented effectively. Previously, researchers introduced several types of cryptography techniques such as Advanced Encryption Standard (AES) and Data Encryption Standard (DES) for symmetric key and Rivest, Shamir and Adleman (RSA) algorithm, Diffee-Hellman Key Exchange and Elgamal Cryptography for asymmetric key [4]. However, the growth of Artificial Intelligence (AI) inspires the researchers to propose several approaches in improving cryptography techniques. This study presents Genetic Algorithm as an approach to provide the best-fit key for encryption and decryption, and thus make the information harder to be deprived by any unauthorized users [3].

This paper is organized into five sections. Section 1 provides brief introduction on symmetric-key cryptography and highlights several security issues. Section II discusses related work on generating the key and Section III illustrates the process of GA to find the fittest key for symmetric cryptography.

Section IV presents the result and analysis of the result. Finally, Section V concludes the study.

II. RELATED WORKS

As the increase of technology in this world today, experts and researchers need to come out with the most suitable method to keep any information safe against the attackers. The key with the properties of optimal randomness and uniqueness is one of the efforts to enhance cryptography process [4]. Thus, a good source of randomness is crucial for a number of cryptographic operations especially to establish secret session keys while commitment schemes use randomness to hide committed value [7].

Genetic Algorithm or GA is a function that imitates the process of natural selection in AI. The combination of randomness and permutation makes the algorithm robust and hard to break [3]. In a research conducted by Conci,A. *et.al* (2015), they focus on AES cryptography in color image steganography. The process was divided into two methodologies, which are genetic algorithm and path relinking. It also involves a hybrid approach, Least Significant Bits (LSB) substitution technique. In this research, GA is used to enhance the quality of the resulting image [8].

On the other hand, GA is used to analyse the security of quantum key distribution (QKD) protocols. The maximum tolerated noise level of a QKD protocol is found by implementing GA to improve the tolerated bound. In this research, GA evolves candidate solution G, where the initial solution is constructed by setting each element of G to a number chosen randomly in the internal range [-2, 2]. By choosing the range, it seems to provide good results which therefore, it did not spend a lot of time experimenting with other choices. Then, crossover takes place to cross one point for each individual vector in the collection G and the mutation process takes place. The 50 populations are generated at the end of the experiment. Thus, it shows the technique can be used to analyse the security of complicated QKD protocols requiring the adversary to interact with the users [9].

Due to the exploitation of randomness involved in crossover and mutation process in GA, this paper proposes GA to find the best-fit symmetric key in cryptography field. The calculation of fitness function depends upon the coefficient of auto correlation and phi-coefficient may help the GA technique to decide the best key from the result produced.

III. GENERATING BEST-FIT CRYPTOGRAPHIC KEY USING GENETIC ALGORITHM

Figure 1 illustrates the flowchart of GA to generate best-fit symmetric key. Based on pseudo random number generator used to produce unique keys in various ciphers, GA starts with generating population known as chromosomes of computer generated random keys. The number of genes equals to the length of key used. In this study, the lengths of the key used are 48-bits and 128-bits size due to DES and AES techniques respectively. Thus, a random initial population of 100 chromosomes each having 48 genes and 128 genes are generated. The population size depends on the large number of possible solutions. After the population is generated, it undergoes the genetic operations called crossover and mutation that increase the total number of chromosomes [3]. The individuals are probabilistically selected to participate in the genetic operations based on their fitness.

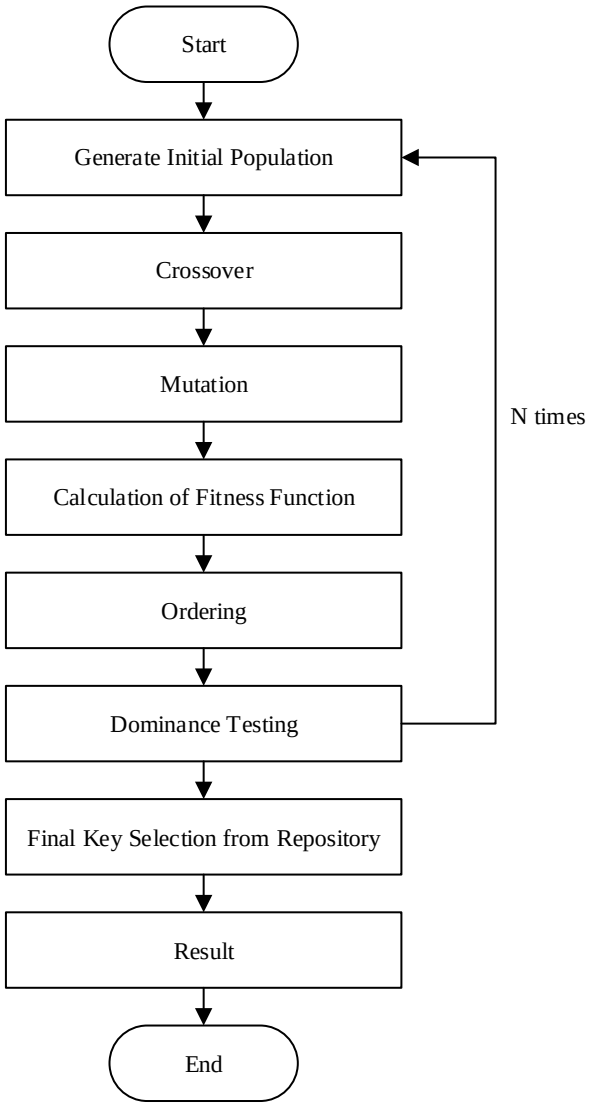

Fig. 1 The Flowchart of Genetic Algorithm

A. Crossover

The Crossover, known as the sexual genetic recombination which two randomly chosen individuals selected from populations undergo the mating process [3]. In mating process, the first individual bit is from the starting point until the crossover point while the bit of the second individual starts from crossover point until the end after of the chromosome. Figure 2 illustrates the example of the process. The example shows the point of crossover happened at crossover point = 25.

The crossover point is randomly generated for each mating process. As the crossover rate is fixed at 2.5, the number of crossover is calculated given by (1).

Thus, this produces 120 and 320 number of crossovers for 48 and 128 genes of chromosomes respectively. Now, the total population sizes for both different number of genes increase to 220 and 420 individually.

$$\text{Number of crossover} = \frac{\text{crossover rate} \times \text{key length} \times \text{number of populations}}{100} \quad (1)$$

Parent 1-
000010011011001011010111100011110000111001100001
Parent 2-
000001111001010110010000101011100111011111011001
Child - 00001001101100101101011110101110011101111101100 1

Fig. 2 The Example of Crossover Process from 48 Genes Chromosome

B. Mutation

Next, the mutation operation takes place. This process changes one or more genes in a chromosome from its previous state [10]. Figure 3 shows the mutation process that happened at mutation point = 45. Based on the previous example, the original chromosome comes from child as the output of crossover between Parent 1 and Parent 2. With mutation rate equals to 0.5, the number of mutation is calculated given by (2) which gives 52 and 268 number of mutations. Therefore, the new total population sizes now are increased to 272 and 692.

$$\text{Number of mutation} = \frac{\text{mutation rate} \times \text{key length} \times \text{number of populations}}{100} \quad (2)$$

Original chromosome-
000010011011001011010111101011100111011111011001
Mutated chromosome-
111100110100000111111100011011010100000011000001
Mutate 1 to 0 at point 45

Fig. 3 The Example of Mutation Process

C. Fitness Function

After mutation, fitness function needs to be calculated to test the suitability of the chromosome to go through the next process. The individual fitness of the best chromosome is increasing as the algorithm continues. The total fitness of the population is also increasing as a whole. This condition makes the size of the chromosome in binary form increases. Therefore, the key needs to be converted from binary into the decimal form.

Gap test and frequency test are performed on 10 selected populations using fitness function as in Table 1 and Table 2. Gap test measures the gap between the two repeating numbers. It is a test to compare each chromosome with the expected number of gaps. It counts the number of digits that appears between repetitions of a particular digit. A gap of length occurs between the recurrences of some digit [11].

Frequency test is used to calculate the randomness and test the number uniformity distribution. There are two different methods available, which are Kolmogorov-Smirnov test and Chi-Square test. The agreement between the distribution of a sample of generated random numbers and theoretical uniform distribution is measured by both tests. Both tests are based on the null hypothesis of no significant difference between the theoretical distribution and sample distribution [12]. Kolmogorov-Smirnov (K-S) Test is chosen because it is widely used as a goodness-of-fit test [13].

D. Ordering

The ordering step takes place after calculating the fitness function. Firstly, the chromosomes are arranged in a sorted order from the highest fitness value to the lowest value according to their fitness function. Based on Table 1, Figure 4 shows the example of fitness value for Population 1. Compared to other populations, the fitness value for Population 1 is the least fitness value.

Population 1-
111000111110111100001011111110111011111011011001

Fitness Value : 0.7026

Fig. 4 The Fitness Value

E. Dominance Testing

The dominance testing is then performed using the output of ordering step. In this process, the key with the highest fitness value is being paired with the rest of the keys. Next, hamming distance is calculated between pairs. The hamming distance is calculated by performing XOR of the two binary keys then calculating the number of 1's [4]. In Figure 5, Population 10, which is the topmost key with the highest fitness value are compared to Population 1 as the lowest fitness value and the hamming distance is calculated between them. The hamming distance between both populations is 17.

To end the process, the population with the maximum hamming distance is then chosen as the key that can dominate over others. The key is chosen as the final key.

Population 10-
110000000001010101110011010110110101001111110000

Population 1-
111000111110111100001011111110111011111011011001

Fig. 5 The Comparison between Populations

IV. RESULTS AND DISCUSSION

The results in Table 1 and Table 2 are based on two tests that applied in generated population. There are Gap Test and Kolmogorov-Smirnov Test. Based on both tests, the results are used in the fitness function to get the fitness value. The highest fitness value is selected as the most random key and be chosen as the key tested in the next steps.

Table. 1 The Test Results with 48 bits

No.	Gap Test λ_1	K-S Test λ_2	$\lambda = \lambda_1 + \lambda_2$	Fitness Value = $F = \frac{1}{\lambda = \lambda_1 + \lambda_2}$	Position
1	0.5474	0.3125	0.8599	0.7026	10
2	0.6203	0.3542	0.9745	0.7260	5
3	0.5691	0.3542	0.9233	0.7157	9
4	0.5909	0.3542	0.9451	0.7201	7
5	0.5691	0.3958	0.9649	0.7241	6
6	0.6960	0.4583	1.1543	0.7603	1
7	0.6126	0.4583	1.0709	0.7448	2
8	0.5688	0.3750	0.9438	0.7199	8
9	0.5909	0.4792	1.0701	0.7446	3
10	0.5470	0.4792	1.0262	0.7362	4

Table. 2 The Test Results with 128 bits

No.	Gap Test λ_1	K-S Test λ_2	$\lambda = \lambda_1 + \lambda_2$	Fitness Value = $F = \frac{1}{\lambda = \lambda_1 + \lambda_2}$	Position
1	0.6005	0.4141	1.0146	0.7339	7
2	0.5746	0.4063	0.9809	0.7273	10
3	0.5926	0.4141	1.0067	0.7324	8
4	0.5826	0.4219	1.0045	0.7319	9
5	0.6100	0.4219	1.0319	0.7373	6
6	0.6005	0.4766	1.0771	0.7459	2
7	0.6005	0.4531	1.0536	0.7415	4
8	0.5767	0.4688	1.0455	0.7399	5
9	0.5688	0.4922	1.0610	0.7429	3
10	0.6005	0.4922	1.0927	0.7489	1

Table 3 shows the result of the average fitness value and the computational complexity for Genetic Algorithm. The results states that the accuracy measured based on fitness value is increased if the key length increases. This may due to the greater of the key length makes the population generated through the process of GA most random and the probability of the repetition reduces. However, the computational time shows that the time taken for the algorithm to run increases due to the difference of the key length. In conclusion, 128 bits has higher fitness than 48 bits key size but it takes longer computational than 48 bits.

Table. 3 The Result of Computational Analysis for Genetic Algorithm

No.	Analysis of Result	Genetic Algorithm	
		48 bits	128 bits
1.	Accuracy based on average fitness value	0.7294	0.7382
2.	Elapsed time (sec)	1.386	1.593
3.	Self-time (sec)	1.333	1.581

V. CONCLUSION

Both 48 bits and 128 bits key sizes produce the results based on three parameters that are measured at the end of this study. The fitness test reveals that the longer key length is produced, the higher fitness value is determined. Nevertheless, elapsed time and self-time increases as the key length increases. This is because it needs more time to undergo several processes in GA which are complex and most random so that make almost impossible for the cryptanalysts to attack the data.

ACKNOWLEDGMENT

The authors would like to thank all the reviewers for their helpful comments. The authors would also like to appreciate the support of Universiti Tun Hussein Onn Malaysia for funding the study under Tier 1 Grant (H208).

REFERENCES

1. Kumari, S., "Recurrent sequences and cryptography," Master in Science dissertation, Department of Mathematics, National Institute of Technology Rourkela, India, 2013.
2. Dutta, S., Das, T., Jash, S., Patra, D. and Paul, P., "A Cryptography Algorithm Using the Operations of Genetic Algorithm & Pseudo Random Sequence Generating Functions," *International Journal*, vol. *3*(5), 2014.
3. Jhingran, R., Thada, V. and Dhaka, S., "A study on cryptography using genetic algorithm," *International Journal of Computer Applications*, vol. 118(20), 2015.
4. Jawaid, S., & Jamal, A., "Generating the Best Fit Key in Cryptography using Genetic Algorithm," *International Journal of Computer* Applications, vol. 98(20), pp. 33–39, 2014.
5. Kalaiselvi, K. and Kumar, A., "An empirical study on effect of variations in the population size and generations of genetic algorithms in cryptography," in *2017 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)*, pp. 1-5.
6. Hussain, H. N., "Implementation of Symmetric Encryption Algorithms," vol. 8(4), pp. 13–18, 2017.
7. Tipcevic, Mario & Kaya, Koc, Cetin. "True Random Number Generators," in *Open Problems in Mathematics and Computational Science*, 2014, pp. 1–45.
8. Conci, A., Brazil, A.L., Ferreira, S.B.L. and MacHenry, T., "AES cryptography in color image steganography by genetic algorithms," in *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, pp. 1-8.
9. Krawec, W.O., "A genetic algorithm to analyze the security of quantum cryptographic protocols," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2098-2105.
10. Goyat, S., "Cryptography Using Genetic Algorithms (GAs)," *Journal of Computer Engineering*, vol. 1(5), pp. 06–08, 2012.
11. Gap Test. (n.d.). Retrieved January 9, 2019, from https://www.eg.bucknell.edu/~xmeng/Course/CS6337/Note/master/node46.html
12. Frequency test. (n.d.). Retrieved January 9, 2019, from https://www.eg.bucknell.edu/~xmeng/Course/CS6337/Note/master/node43.html
13. Dong, X., "Small Improvement to the Kolmogorov-Smirnov Test," Master of Science dissertation, Department of Mathematics and Statistics, Georgia State University, Atlanta, GA, 2013.