Key generation method based on Genitor Genetic Algorithm model

Andrii Pryimak [1], Yurii Yaremchuk [1] and Nataliia Kunanets [2]

[1] *Vinnytsia National Technical University, Khmelnytsky highway 95, Vinnytsia, 21000, Ukraine*
[2] *Lviv Polytechnic National University, S. Bandery str., 12, Lviv, 79000, Ukraine*

Abstract

The study examines the current genetic algorithm-based key generation methods. Their models of work were studied, and the advantages and disadvantages are described, in particular slow key generation process because of the use of the classical model of genetic algorithm, as well as not always accurate determination of the statistical security of the final key. In this regard, a new method of generating 128-bit keys for symmetric cryptographic algorithms based on the Genitor model of the genetic algorithm has been proposed. To determine the fitness function, it is proposed to use five statistical tests. Statistical testing of the proposed method by NIST STS tests is presented, which showed that the obtained statistical security indicators are in the range of 0.9–1, which reflects the high statistical security of this method. In comparison to existing methods, the proposed showed better results almost on the whole range. The proposed method has a higher level of statistical security than analyzed methods because it showed higher results by 1-5% in twelve tests out of fifteen. Comparison of the speed of the methods also showed greater efficiency of the proposed method than existing analogues, as the key generation rate increased by an average of 0.536 - 0.918 ms, which is a significant improvement.

Keywords

Genetic algorithm, key generation, cryptography, security

1. Introduction

Due to the large amount of data that is transmitted daily through computer networks and stored in cloud environments, information security has become one of the most important aspects of networking. There are many approaches of protecting information on the web, however one of the most effective and popular is cryptography. Cryptography addresses the issue of providing information's confidentiality, safety and validity (secure transmission of data, exchange of information or its storage). The main goal of cryptography can be reached by ensuring confidentiality of information, primarily in order to protect it from unauthorized access. The essence of encryption is that to perform the reverse operation you need to know the secret code, otherwise obtaining the original data will be impossible [1].

Any cryptographic system's key, which is a unpredictable bits's stream applied by the cryptographic algorithm to accomplish the action of converting plain text to cipher and vice versa, is what makes the system secure. The main parameter of the security of the cryptographic key is the degree of entropy or randomness. A sequence can only be random when it can not be reproduced. This means that if you run the generator of truly random sequences twice at the same input, then its output will be always different. However, research conducted by SEC Consult [2] has shown that millions of devices around the world are at risk due to the fact that SSH keys and HTTPS credentials are duplicated. More than 9% of all

CMiGIN 2022: 2nd International Conference on Conflict Management in Global Information Networks, November 30, 2022, Kyiv, Ukraine
EMAIL: andrii.pryimak@live.com (A. Pryimak); yurevyar@gmail.com (Yu. Yaremchuk); nek.lviv@gmail.com (N. Kunanets)
ORCID: 0000-0001-9695-0462 (A. Pryimak); 0000-0002-6303-7703 (Yu. Yaremchuk); 0000-0003-3007-2462 (N. Kunanets)

HTTPS hosts on the Internet and 6% of all SSH hosts' private keys, which is a large part of the World Wide Web, were discovered in a data set of 580 keys [2].

Thus, solving the problem of the security of the keys for encryption and decryption of the information is very important.

2. Related works

The problem of security of cryptographic keys arises from the very process of their generation. Most systems are based on cryptographically secure pseudo-random number generators (CSPRNG). They must generate such sequences that no efficient algorithm can distinguish them from completely random sequences by the polynomial time. In other words, no statistical test will be able to distinguish the resulting sequence of pseudo-random numbers from a truly random sequence. However, it is known that such generators have a number of vulnerabilities [3]:

- improper implementation of algorithms and bugs in the code;
- contain intentionally built in weaknesses for an outside party to exploit;
- CSPRNGs can be lacking enough entropy to generate sufficiently random numbers.

Therefore, to make the key secure and as unpredictable as possible, this paper considers methods due to the concept of natural selection, which makes the genetic algorithm (GA) [4] a good applicant for the process to be followed to generate the key. Today, there are several methods of using GA to increase the security of the keys:

A. Z. Zakaria and co-authors investigated the possibility of using the classical GA model, in which next generations are formed from the children of the current generation, to generate the strongest key for symmetric cryptography algorithms DES (48-bits) and AES (128-bits).

The proposed method of key generation includes 10 steps, basic GA operations, fitness function, which is calculated based on values obtained from Gap test and Frequency test, as well as calculation of the Hamming distance to select the best key from all sets. The disadvantages of this method include the generation of a large number of sets with hundreds of chromosomes (due to the use of the classical model of GA), which leads to slow key generation (average 1.581 ms for 128 bits sequence in N-iterations), as well as possible inaccuracies in the calculation of fitness function, as only two statistical tests are used [5].

M. Alkharji, M. Al Hammoshi, C. Hu and H. Liu used classical model of GA, the initial set of 64 chromosomes, roulette-wheel as a selection function, uniform crossover, mutation rate - 0.003 and the maximum number of generations - 10,000 times, to generate a key with a length of 4096 bits for the asymmetric RSA cryptography algorithm. The final step in this method is to select the output key from the final set of chromosomes with the best value of fitness function.

Although the authors claim that their proposed method is quite simple in terms of mathematical calculations, and the result is a statistically secure key, but according to the provided recommendations of set size and number of iterations for finding the best key, you will need to spend 3.5760 ms, which is enough time-consuming process [6].

C. Chunka, R. S. Goswami and S. Banerjee in their research used the classical model of GA, two-point crossover and three-point mutation for improving randomness of the initial set of chromosomes set. The selection of the most secure key (128-bits), calculates by the definition of the fintess function.

The proposed approach's shortcoming is that the only Hamming distance's calculation is insufficient to verify the statistical security of the final key, so it can not be said that the generated key will be secure enough for further use in cryptographic algorithms [7].

S. Jawaid, A. Saiyeda and N. Suroor used the modified model of classical GA to find the most secure key. The fitness function was determined by using the Frequency Test and Gap Test. The proposed method consists of 7 steps, an initial set of 100 chromosomes and 100 iterations.

A distinctive feature of this method is that it generates 3 sets (except the initial set, which is replenished with new chromosomes, new sets are also generated after the process of crossover and mutation). In each of the three sets, one of the most secure keys was chosen due to the fitness function's value and then compared to each other. The key with the highest fitness function is the generated key for the crypto algorithm.

The authors tested their method only with a 48-bits keys for the DES cryptographic algorithm, so it is not clear how many iterations it will take to generate a 128-bits key, which is more secure, and how it will affect the execution time of the algorithm. Also, the use of two tests (Gap Test and Frequency Test) to find the fitness function may not be enough to claim that the generated key is secure enough [8].

M. Ragavan and K. Prabu considered the possibility of using a genetic algorithm to generate a 128-bit key for a symmetric AES crypto algorithm. The proposed method includes all the main operations of GA – set generation, crossover, mutation, calculation of the fitness function and the process of selection the best chromosomes.

It is proposed to use an initial set of 15 chromosomes, and for security testing of key and calculation of its fitness function - the value of entropy, but the authors did not specify how many iterations of the algorithm must be repeated to achieve the desired level of entropy. It is also worth mentioning that the average key generation time by this method is greater than the previous methods (2.5540 ms) [9].

The comparison of the currently recognized key generation techniques based on GA is shown in Table 1.

Table 1

Comparison of existing methods of key generation based on GA

Method name	Used model of GA	Length of generated key, (bits)	Number of iterations	Mean time of method processing, (ms)
Method [5]	Classical	128	N	1.581
Method [6]	Classical	4096	10000	3.5760
Method [7]	Classical	128	3	1.963
Method [8]	Classical modified	128	100	-
Method [9]	Classical	128	N	2.5540

Based on the analysis of currently known methods of generating cryptographic keys based on genetic algorithms, it can be concluded that the method presented in [5] and the method presented in [7] showed the best results in the number of iterations, execution time and confirmed cryptographic security of the output keys. However, among to all of their advantages, they also have several disadvantages, due to the use of the classical model of GA, there is a need to generate many sets, which in turn significantly slows down the proposed methods. In this regard, it is proposed to discover the Genitor model of the genetic algorithm, which, unlike the classical model of GA, overwrites the chromosome with the lowest fitness function and as a result does not generate a new set each time and there is no need to recalculate the fitness function of existing chromosomes. It may be also possible to use the difference in speed to more accurately calculate the fitness function based on more statistical tests.

3. Problem Statement

Conduct research on the Genitor model of the genetic algorithm, in particular, in order to strengthen the security of the generated keys, as well as exploration of the possibility of accelerating the process of their generation. Based on the study, propose a method of generating keys with a length of 128 bits. Test the proposed approach statistically, then compare the results to those of other methods already in use.

4. Proposed work

Using the basic operators of the genetic algorithm (crossover, mutation and selection), in this research was proposed a method of key generation that will be applied by symmetric algorithms to encrypt and decrypt information. As already mentioned, it will be useful to use the Genitor model of GA, because this model does not generate new sets, but only overwrites the chromosome with the lowest

value of the fitness function by the resulting chromosome from each iteration of the algorithm. In addition, this model of GA has the following features:

- fixed set size;
- fixed bit size of genes;
- chromosomes for crossover process are selected randomly;
- there are no restrictions on the type of crossover and mutation;
- as a result of chromosomes crossover, one offspring is obtained, which takes the place of the least adapted chromosome from the initial set.

Since for modern symmetric algorithms (AES, CAST5, IDEA, Blowfish, Twofish) the main characteristic of crypto security is the length of the key, encryption with keys of 128 bits and above is considered as a secure, because the decryption of information without a secret key requires years of work of the most powerful supercomputers. Therefore, there was decided to use 128-bits key length in the proposed method, which in turn includes stages:

1. to generate the initial set of chromosomes – 32 sequences with a length of 128 bits of each one.
2. to determine the fitness function of each chromosome, it is proposed to use five statistical tests – Runs Test, Frequency Test, Longest-Run-of-Ones in a Block Tests, Approximate Entropy Test and Cumulative Sums (Cusums) Test.
3. to select two completely random chromosomes and generation of two random points of crossover. Performing the process of double crossover and obtaining one offspring (Figure 1).

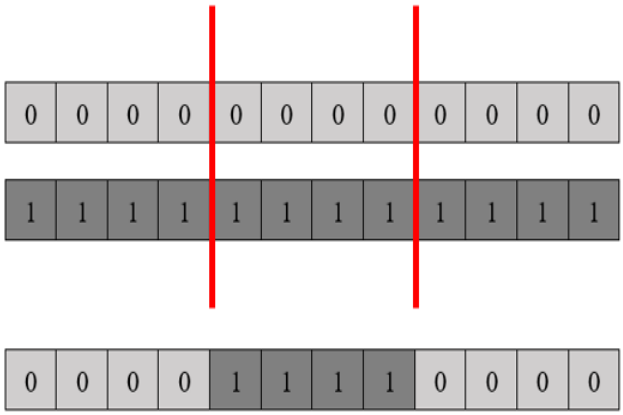

Figure 1: An example of performing the process of double crossover and obtaining one offspring

4. Generation of a random mutation point and, accordingly, mutation of the selected bit of chromosome (Figure 2).

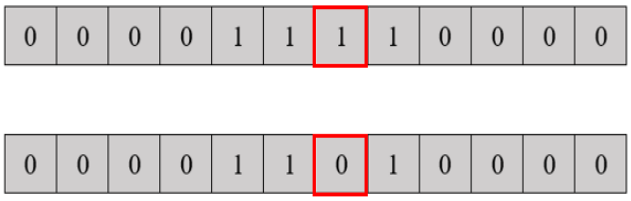

Figure 2: Example of mutation of the 7th bit of chromosome

5. Calculation of the fitness function's value of the resulting chromosome by five statistical tests:

$$F = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5, \quad (1)$$

where F – value of fitness function, λ – the $p - value$ of each of the five statistical tests.

6. If the chromosome has a value of the fitness function F greater than 3,75, then it is directly selected as the final key. If the value is less than 3,75, then it is necessary to overwrite the chromosome with the lowest value of the fitness function from the current set by this chromosome (if its value is greater) and perform the next iteration.

The flowchart of the proposed method of key generation is presented on Figure 3.

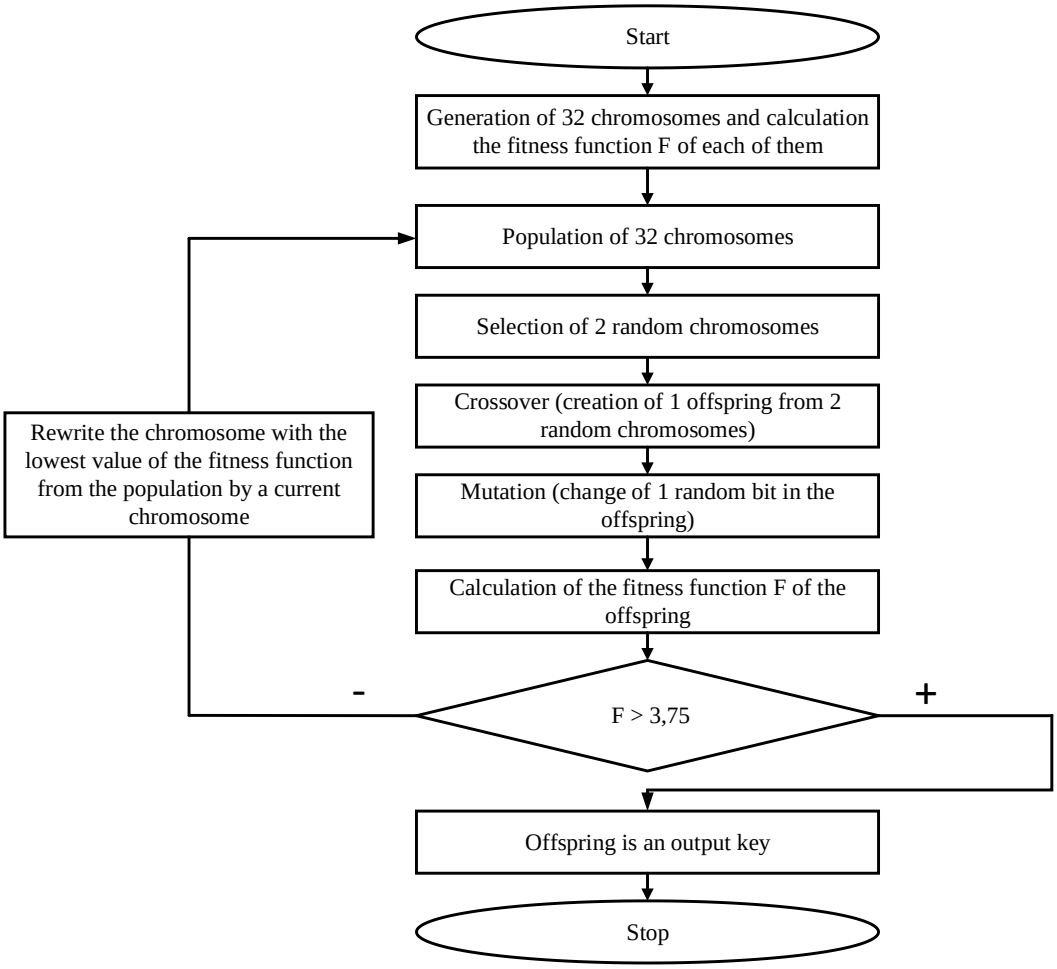

Figure 3: The flowchart of the proposed method

The experiments showed that on average, 120-180 iterations of the algorithm are required to generate a chromosome with the proper value of the fitness function. The resulting key can participate in the encryption and decryption of information by symmetric cryptographic algorithms.

In addition, the use of Genitor model of GA saves computing resources and time, as demonstrated in the next section of this paper. And testing of the resulting chromosomes with five statistical tests accurately estimates the randomness of the generated sequences.

5. Results and Discussion

5.1. Verification of statistical security of the key generation method

With the purpose of verification of the suggested method's produced keys' statistical security, a package of statistical tests NIST STS (National Institute of Standard and Technologies Statistical TestSuite) [10] was used. It includes composition of statistical test: Runs Test and Longest-Run-of-Ones in a Block Tests, Frequency Test and Frequency Test within a Block, Random Excursions Test and Random Excursions Variant Test, Binary Matrix Rank Test, Discrete Fourier Transform Test, Overlapping Template Matching Test, Non-overlapping Template Matching Test, Maurer's Universal Statistical Test, Serial Test, Approximate Entropy Test, Cumulative Sums Test, Linear Complexity Test. These tests serve to show how random binary sequences produced by either hardware or software random number generators are. They rely on numerous statistical characteristics that random sequences only possess [11].

The main parameters for passing the tests were chosen:

- key length – 128 bits;
- value of α – 0,01;
- number of tests – 100.

The results of passing all 15 tests using keys that were previously produced using the suggested approach are shown in Figure 4.

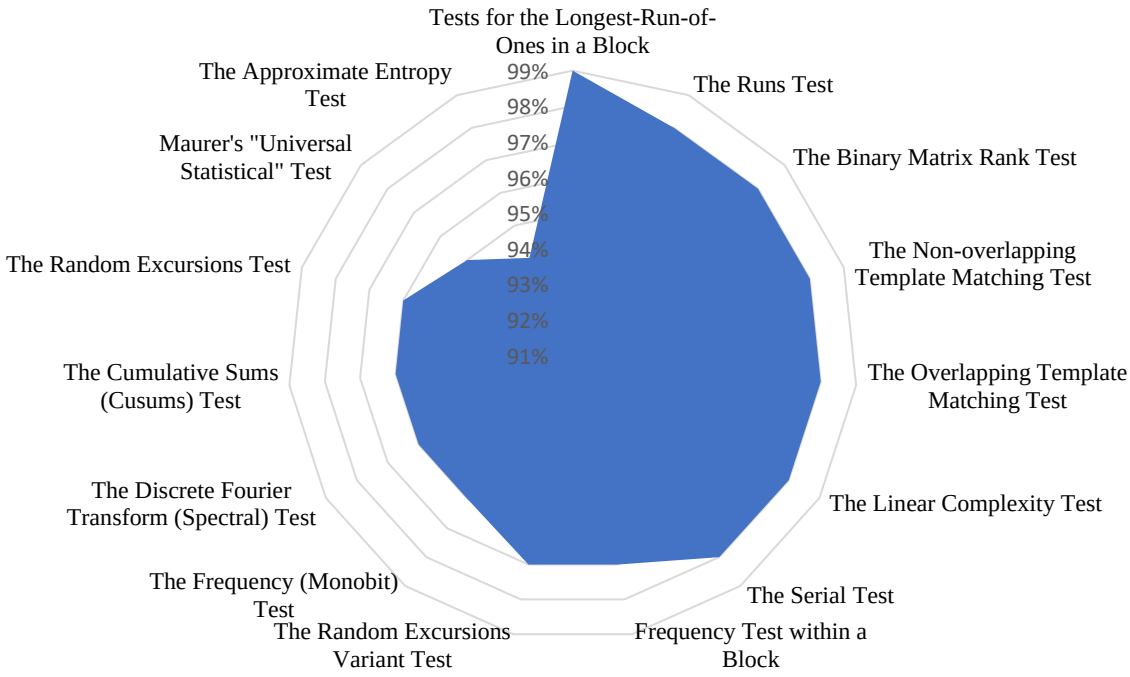

Figure 4: Results of passing of fifteen NIST STS tests by generated keys

Figure 4 illustrates that all tests showed high results, which in turn indicates a high statistical security of the keys generated. A statistical overview of the suggested strategy is presented in Figure 5.

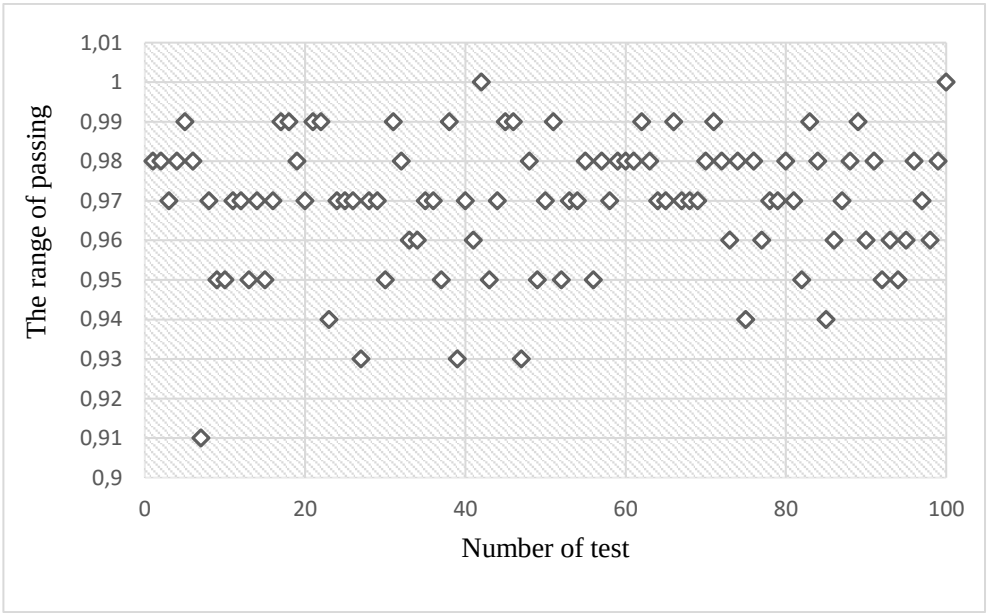

Figure 5: Statistical analysis of the suggested key generation method

Figure 5 shows that the test results of key generation method are inside of the range of 0.9–1, which shows the high statistical security of this method.

5.2. Comparison of the proposed and known key generation methods using GA's statistical security

With the purpose of comparing the statistical security of the proposed and known methods of GA's key generation, a package of statistical tests NIST STS [10] was used.

The main parameters for passing the tests were chosen:

- key length – 128 bits;
- value of α – 0,01;
- number of tests – 100.

Figure 6 shows the outcomes of keys that were previously generated by the suggested approach and the methods described in [5], [7] passing all 15 tests.

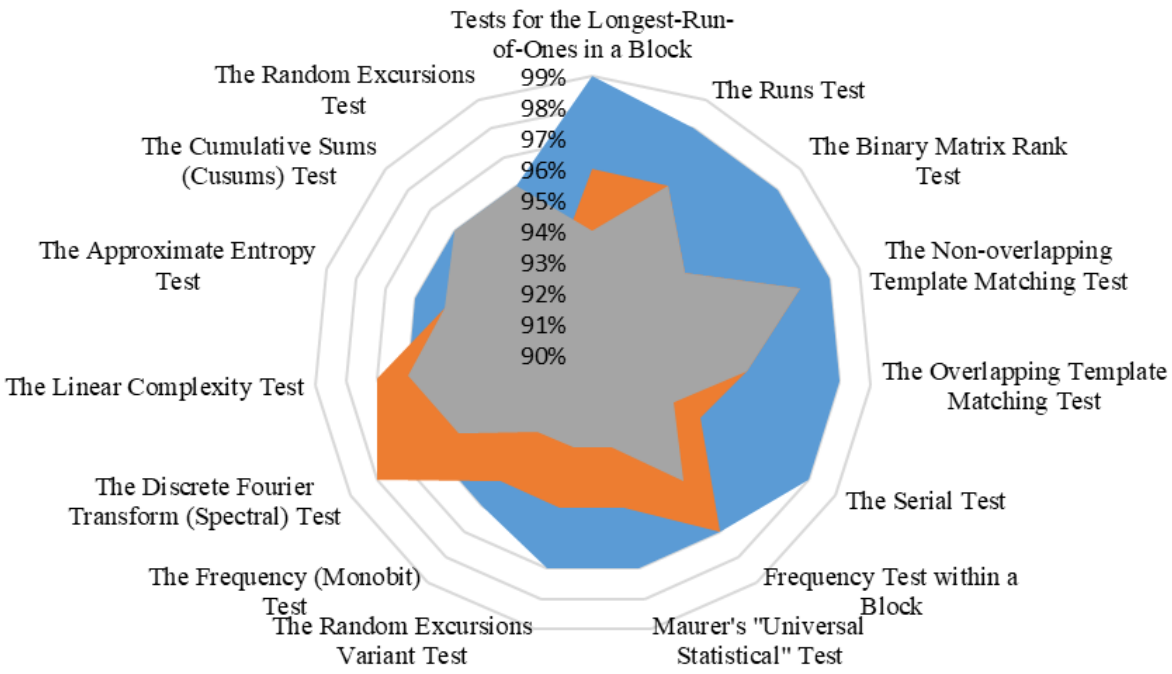

Figure 6: Comparison of the results of passing 15 NIST STS tests by the keys generated by the proposed method and methods [5, 7]

On the basis of the aforementioned statistical test results, it can be concluded that the suggested approach creates keys with a greater level of statistical security than the method described in [5], as it shown higher rates by 1-4% in twelve out of fifteen tests. Additionally, compared to other methods, the suggested strategy offered superior performance rates. Thus, comparing it with the method presented in [7], the proposed method has higher rates in twelve tests out of fifteen by 1-5%.

A comparison of the test results for each test from the NIST statistical package using the proposed and known methodologies [12–16] is shown in Figure 7.

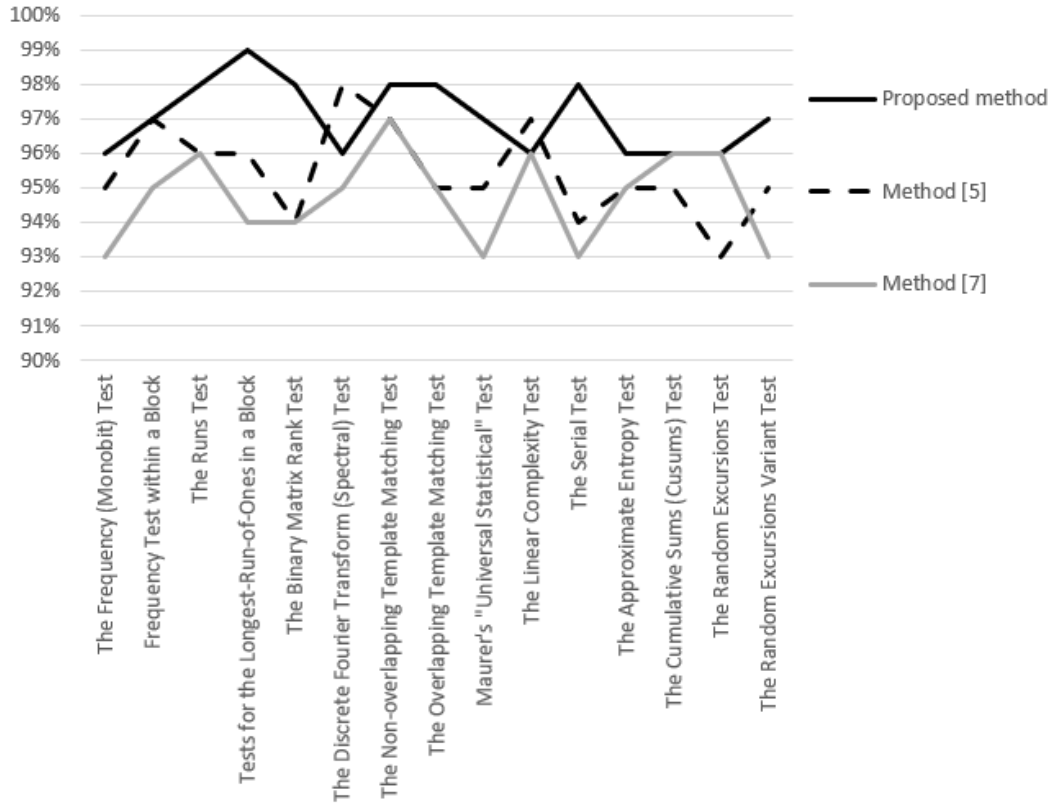

Figure 7: Graphical comparison of test results

As can be seen from the Figure 5, the proposed method showed better results in almost all of the tests, indicating its higher level of statistical security.

5.3. Comparison of the speed and effectiveness of the proposed and existing key generation methods according to GA

As noted earlier in this research, in addition to the statistical security of keys, it is also desirable to increase the speed of methods of generating keys based on GA. Thus, it is evident from Table 1 that the current approaches are based on the traditional GA paradigm, which necessitates the creation of a new set after each iteration in order to further produce a strong key. That is why in the proposed method we decided to use Genitor model GA, which greatly simplifies the process of optimizing the initial set. And despite the fact that 5 statistical tests are used to find the fitness function's value, the average key generation time for 100 test executions of the method is in the range of 0.835 - 1.256 ms, which is on average 0.536 and 0.918 ms faster than the results of the method presented in [5] and the method presented in [7], respectively. This difference can be considered as a significant improvement in performance. A comparison of the performance of the proposed method and known ones is shown in Table 2.

Table 2
Comparison of the proposed and known key generation methods based on GA

Method name	Used model of GA	Length of generated key, (bits)	Number of iterations	Mean time of method processing, (ms)
Proposed method	Genitor	128	N	1,045
Method [5]	Classical	128	N	1.581
Method [7]	Classical	128	3	1.963

Furthermore, after 100 experiments of key generation using the proposed method, it was determined that the average number of iterations is in the range of 120-180 times. Comparative characteristics of the speed of the methods are presented in Figure 8.

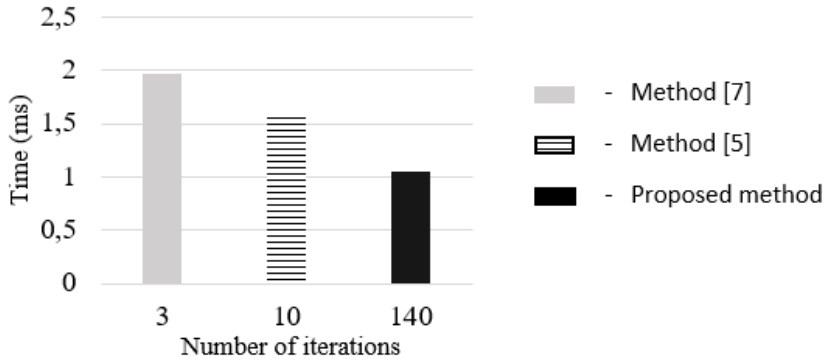

Figure 8: Graphical comparison of speed of methods

As can be seen from Figure 6, the proposed method showed better performance results despite a much larger number of iterations.

6. Conclusions

Existing methods of key generation based on a genetic algorithm were analyzed in this research. Their advantages and disadvantages were described, especially slow key generation process because of the use of the classical model of genetic algorithm, as well as not always accurate determination of the statistical security of the final key.

Thus, to increase the security of the keys used in symmetric cryptographic algorithms for encrypting and decrypting information, a study was conducted on the possibility of using Genitor model GA and five statistical tests to calculate the fitness function. Based on this study, a new method of generating keys with a length of 128 bits was proposed.

The NIST STS package of statistical tests was used to evaluate the outcomes of the proposed genetic algorithm-based key generation method. The passing test results do not surpass the range of 0.9-1, demonstrating the great statistical security of this procedure. In almost all of the tests, the new method outperformed the current ones. The proposed method has a higher level of statistical security than the method presented in [5], because it showed higher rates by 1-4% in twelve tests out of fifteen. Comparing it to the method presented in [7], the proposed method has higher rates in twelve tests out of fifteen by 1-5%.

Comparing the speed of the proposed method and existing analogues, it is clear that the proposed is on average 0.536 and 0.918 ms faster than the results of the method presented in [5] and the method presented in [7], respectively, which is a significant improvement in performance.

7. References

[1] A. Pryimak, Y. Yaremchuk, Increasing of Strength of Blowfish Cipher by Genetic Algorithm Weak Keys Optimisation, Legal, Regulatory and Metrological Support of Information Security System in Ukraine 35 (2018) 106–115.

[2] K. B. Sudeepa, G. Aithal, V. Rajinikanth, S. C. Satapathy, Genetic Algorithm Based Key Sequence Generation for Cipher System, Pattern Recognition Letters 133 (2020) 341-348.

[3] A. Li, Potential weaknesses in pseudorandom number generators, 2017, pp. 1-12.

[4] D. Whitley, A genetic algorithm tutorial, Stat Comput 4 (1994) 65–85. doi: https://doi.org/10.1007/BF00175354.

[5] A. Z. Zakaria, S. N. Ramli, Ch. Ch. Wen, et. al., Enhancing the Randomness of Symmetric Key using Genetic Algorithm, International Journal of Innovative Technology and Exploring Engineering (IJITEE) 8 (2019) 327-330.

[6] M. Alkharji, M. Al Hammoshi, C. Hu, H Liu, Genetic Algorithm based key Generation for Fully Homomorphic Encryption, in: Proceedings of 16th Annual Security Conference, Las Vegas, NV, 2017. URL: http://029e2c6.netsolhost.com/II-Proceedings/2017/IIVC2017_ALKHARJI_et_al.pdf.

[7] C. Chunka, R. S. Goswami, S. Banerjee, A Novel Approach to Generate Symmetric Key in Cryptography Using Genetic Algorithm (GA), in: Abraham, A., Dutta, P., Mandal, J., Bhattacharya, A., Dutta, S. (eds) Emerging Technologies in Data Mining and Information Security, Advances in Intelligent Systems and Computing, vol 755. Springer, Singapore. https://doi.org/10.1007/978-981-13-1951-8_64.

[8] S. Jawaid, A. Saiyeda, N. Suroor, Selection of Fittest Key Using Genetic Algorithm and Autocorrelation in Cryptography, Journal of Computer Sciences and Applications 3(2) (2015) 46-51. doi:10.12691/jcsa-3-2-5.

[9] M. Ragavan, K. Prabu, Dynamic Key generation for Cryptographic Process using Genetic Algorithm, International Journal of Computer Science and Information Security 17(4) (2019), 246-250.

[10] L. E. Bassham et al., A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST, Sep. 2010. URL: https://www.nist.gov/publications/statistical-test-suite-random-and-pseudorandom-number-generators-cryptographic.

[11] J. Haney, M. Theofanos, Y. Acar, S. S. Prettyman, Organizational views of NIST cryptographic standards and testing and validation programs', National Institute of Standards and Technology, Gaithersburg, MD, NIST IR 8241, 2018. URL: https://nvlpubs.nist.gov/nistpubs/ir/2018/NIST.IR.8241.pdf.

[12] H. Guo, C. Qin, EML-Based Vector Radio-Frequency Optical Signal Generation Adopting Frequency Doubling With Precoding, IEEE Photonics Journal 8(5) (2016). doi: 10.1109/JPHOT.2016.2604483.

[13] X. Li, J. Xiao, Y. Xu, J. Yu, QPSK Vector Signal Generation Based on Photonic Heterodyne Beating and Optical Carrier Suppression, IEEE Photonics Journal 7(5) (2015). doi: 10.1109/JPHOT.2015.2486685.

[14] M. Iavich, T. Kuchukhidze, S. Gnatyuk, A. Fesenko, Novel certification method for quantum random number generators, International Journal of Computer Network and Information Security 13(3) (2021) 28-38.

[15] S. Kumari, P. Chaudhary, C. M. Chen, M. K. Khan, Questioning Key Compromise Attack on Ostad-Sharif et al.'s Authentication and Session key Generation Scheme for Healthcare Applications, IEEE Access 7 (2019) 39717-39720. doi: 10.1109/ACCESS.2019.2905731.

[16] S. Gong, X. Tao, N. Li, H. Wang, Private Key and Group Key Generation Using Correlated Sources and Wiretap Broadcast Channel in Presence of One-Way Public Communication, IEEE Access 7 (2019) 126812-126830. doi: 10.1109/ACCESS.2019.2937799.