Date of publication MAR-29, 2023, date of current version OCT-19, 2022.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.22.1.2874

Symmetrical Cryptosystems based on Cellular Automata

SERHII OSTAPOV, BOHDAN DIAKONENKO, MAKSYM FYLYPIUK, KATERYNA HAZDIUK, LILIIA SHUMYLIAK, OLHA TARNOVETSKA
Yuriy Fedkovych Chernivtsi National University, Chernivtsi, 58012, Ukraine
(e-mail: s.ostapov@chnu.edu.ua, diakonenko.bohdan@chnu.edu.ua, mfylypiuk@gmail.com, k.gazdiuk@chnu.edu.ua, lshumylyak@gmail.com, savinskaolga@gmail.com)

Corresponding author: Serhii Ostapov (e-mail: s.ostapov@chnu.edu.ua).

ABSTRACT This paper deals with the development of two symmetric encryption algorithms on the basis of cellular automata: a block cipher, that is based on AES and uses three-dimensional cellular automata; a stream cipher, that exploits a hardware-software entropy generation (tracking of keystrokes and mouse pointer movement), as well as the developed hash function, based on "cryptographic sponge" architecture of SHA-3, modified by cellular automata transformations. The block cipher is designed in architecture of SP-network and uses the AES substitution block. Permutation layer and key generation is designed on the basis of cellular automata rules (rules "22", "105" and "150"). The optimal number of rounds to achieve maximum crypto resistance is determined. The stream cipher is designed on the basis of hardware-software entropy generation and uses the cryptographic hash-function in the SHA-3 architecture. Permutation function is developed on the basis of cellular automata rules (rules "30" and "146"). The procedures of shift and permutation of rows and columns is used for better permutation. A final permutation of state elements is used to improve the avalanche effect. The received results are analyzed and summarized; the conclusions and justifications about cipher parameters (like number of rounds, where needed) are made.

KEYWORDS cybersecurity; cellular automata; cipher; binary random sequence generator; hash function.

I. INTRODUCTION

NOWADAYS, the issue of cybersecurity receives a great attention from government structures as well as from public organizations. It is natural in the conditions of the aggressive Russian invasion of Ukraine and the increasing load of communication channels [1-3]. Cryptographic protection in this case plays a crucial role. Therefore, any scientific research and work, related to improvement of cryptographic protection, is considered relevant.

At the same time, the cellular automata (CA) [4-7] are widely used for construction of cryptographic primitives and other branches of simulation [8-12]. The first to denote such capability was Steven Wolfram [12]. Since then, there was developed a whole number of various cryptographic transformations: symmetric ciphers, hashing algorithms, etc. [4-7, 9]. Nevertheless, the issue of CA-based cryptosystem design is being rapidly developed, since the simplicity of architecture and possibility of multi-thread implementation allows improving statistical and cryptographic features of such systems.

This work is also focused on development of two CA-based symmetric ciphers: block and stream, which can be applied for protection of individual messages and files, as well as communication channels.

II. IMPLEMENTATION OF ENCRYPTION ALGORITHMS

A. CELLULAR AUTOMATA IN CRYPTOGRAPHIC TRANSFORMATIONS

Cellular automata have long been used for construction of cryptographic primitives [7-8, 13-21]. Frequently, the elementary one-dimensional cellular automata are used for generation of binary pseudo-random sequences, which have decent statistical characteristics [14-19]. There are known applications of CA for design of hashing algorithms [4, 5], block ciphers [7, 22-23] and stream encryption algorithms [7-8, 14-15, 17-19], etc. In our study we use elementary rules of one-dimensional CA, described in Table 1.

Table 1. Studied cell interaction rules of CA

№	Rule	Boolean form	Arithmetical form
1	"22"	$b'=a \oplus a \wedge b \wedge c \oplus b \oplus c$	$b'=((a+b+c+abc) mod2)$
2	"30"	$b'=a \oplus (b \vee c)$	$b'=((a+b+c+bc) mod2)$
3	"54"	$b'=(a \vee c) \oplus b$	$b'=((a+b+c+bc) mod2)$
4	"86"	$b'=(a \vee b) \oplus c$	$b'=((a+b+ab+c) mod2)$
5	"105"	$b'=\overline{(a \oplus b \oplus c)}$	$b'=((1+a+b+c) mod2)$
6	"135"	$b'=1 \vee a \oplus b \vee c$	$b'=((1+a+bc) mod2)$
7	"146"	$b' = (a \vee c) \wedge (a \vee b \vee c)$	$b' = (a + ab + c + bc + abc) mod2$
7	"149"	$b'=a \vee b \oplus c \vee 1$	$b'=((1+ab+c) mod2)$
8	"150"	$b'=a \oplus b \oplus c$	$b'=((a+b+c) mod2)$
9	"158"	$b'=a \oplus b \oplus c \vee b \wedge c$	$b'=((a+b+c+bc+abc) mod2)$

Rules "30", "86", "135" and "149" and their combinations are the most suitable for creation of binary pseudo-random sequence generators, others – for development of round transformations of hashing algorithms and block ciphers. It does not mean that other, not listed here transition rules (there are total of 256 rules, according to S. Wolfram), are unsuitable for cryptographic transformation; yet the authors have used exactly these rules for their work.

Next, we will cover the block cipher, based on three-dimensional CA that uses simple cell interaction rules.

B. BLOCK CIPHER BASED ON THREE-DIMENSIONAL CA

This cipher applies three-dimensional cellular automata for processing input block as well as key.

The main parameters of the developed cipher are:

block size – 64 bytes (512 bits);

key size – 64 bytes (512 bits);

number of rounds – up to 15;

one whole block is processed in one round.

The specified block and key length are optimal in our view for contemporary cryptographic needs. As for the number of rounds, the right number will be determined after completion of statistical tests.

The implementation of three-dimensional 64-byte CA is reduced to a selection of non-equivalent statistically-independent elementary interaction rules, that will be applied to nearest cell neighbors in three directions: X (rule "105"), Y (rule "22"), Z (rule "150"). In this case, the state of one cell is expressed in one byte, the cell interacts with six neighbors (two on each axis, X, Y, Z). Each pair of neighbors with the cell itself creates a separate elementary CA. New state of cell is received after applying all three rules to it, in such order: X – Y – Z. Bitwise operations on each byte are performed by built-in functions of programming language.

As a cipher architecture we chose the SP-network, similar to one used in the AES cipher. The flowchart of the one-block encryption and decryption sequences is shown in Fig. 1.

Let us describe the applied encryption and decryption procedures.

TransformDataCube. This operation is responsible for reversible transformation of the data block. The simple bitwise XOR is used (because it is reversible itself) with 8 closest cell neighbors on the four diagonals. Naturally, the reversed operation will be the same.

RotateCubeMatrices. The operation is similar to Rijndael's shiftRows, but for the three-dimensional data block. In this case, the rotation of four layers of 4x4x4 cube is done as following: the upper (zeroth) layer does not rotate, next one (first) rotates 90 degrees clockwise, second layer – 180 degrees, third – 270 degrees. This resembles the Rubik's cube rotation. During decryption of data block, the reversed operation invRotateCubeMatrices is used, where layers of cube rotate counter-clockwise.

SubBytes. This operation performs substitution, utilizing the S-Boxes of Rijndael, which proved the high degree of non-linearity and cryptographic robustness. The direct substitution is done by SubBytes, where the reversed one – by reversedSubBytes. Both S-Boxes can be found in FIPS-197, which describes the AES standard.

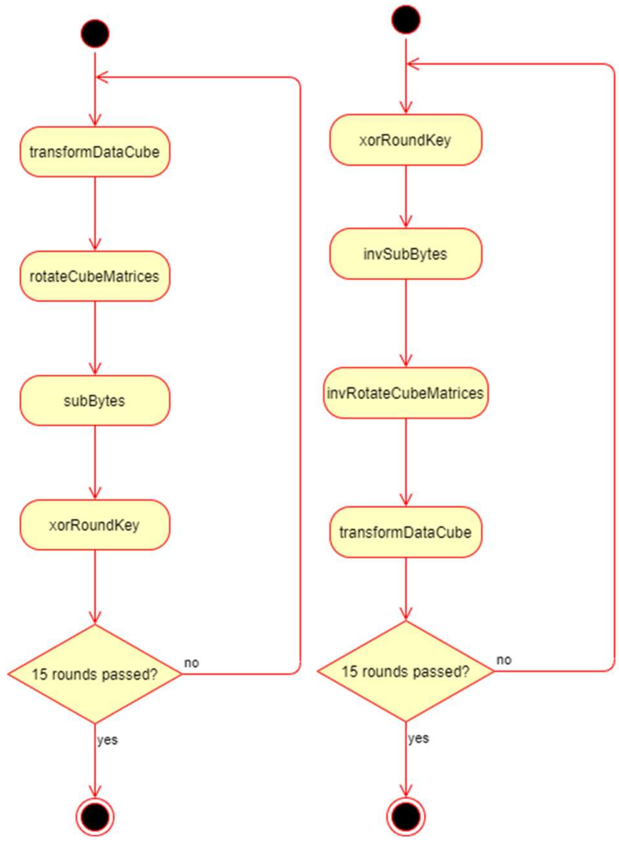

Figure 1. The flowchart of one input block encryption (left) and decryption (right) processes

XorRoundKey. An operation which mixes the round key block in the data block, uses plain XOR, the result is written to data block. For decryption the same operation is used.

Key schedule.

The key expansion operation does not need to be reversible, so we need to generate a certain amount of key material, needed for certain number of rounds. The key length of algorithm is 64 bytes (512 bits), so for NR number of rounds we need to generate 64xNR bytes of key material (512xNR in bits). Therefore, for 15-round cipher we need 960 bytes (7680 bits) of key material. All keys are created initially, before encryption or decryption, and are placed in a list from where each one will be picked for certain round.

For generation of round keys from initial one, a cellular automaton, described above, is used. In this case, the initial key will act as zeroth generation of CA, and then it is transformed for each round. This way, the requirement of synchronization for round keys on both ends of communication channel is satisfied.

Evaluation of the algorithm cryptographic strength.

The evaluation of cipher statistical characteristics has become almost the standard during the development of various encryption algorithms. For this the NIST statistical test suite [11] is employed. It consists of 188 different statistical tests, combined into 16 groups.

To perform the evaluation, firstly, the system must encrypt 12.5 MB file, which has 100 million bits. This file is encrypted with one key, but different number of rounds (from 1 to 15). The resulting sequence is submitted to NIST STS and after evaluation we can make conclusion about the suitable number of rounds as well as overall strength of the developed algorithm by NIST criteria.

It is reasonable to present the received results as a table (Table 2).

Table 2. Results of statistical tests

Percentage, % / Rounds, number of		100	99	98	97	96	95	<95	Average value
1	**Number of passed tests**	71	67	39	7	3	1	-	0.99027
2		73	58	39	8	10	-	-	0.98936
3		60	64	44	18	-	1	1	0.98846
4		65	68	31	16	3	4	1	0.98840
5		76	60	39	12	1	-	-	0.99053
6		72	56	40	14	5	-	1	0.98909
7		67	59	39	12	10	1	-	0.98840
8		71	72	40	5	-	-	-	0.99112
9		75	60	36	11	3	3	-	0.98979
10		77	66	29	13	2	1	-	0.99064
11		75	55	35	11	11	1	-	0.98899
12		69	60	40	13	4	2	-	0.98909
13		83	49	39	11	6	-	-	0.99021
14		86	50	38	13	1	-	-	0.99101
15		61	64	39	16	3	5	-	0.98793

We can also depict evaluation results as a chart (Fig. 2).

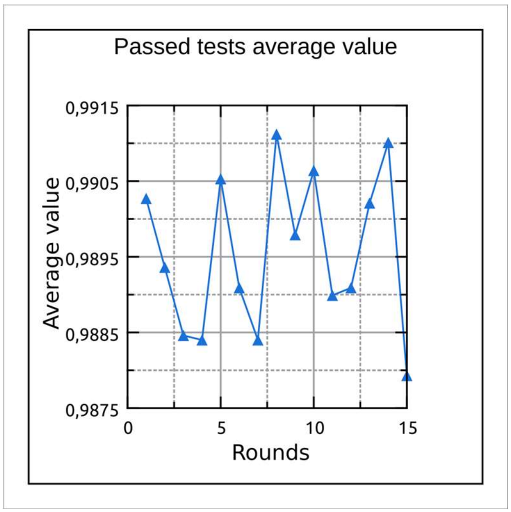

Figure 2. The chart of average probabilities of test passing for each number of rounds

As it is seen in Table 2 and Fig. 2, most statistical tests are passed with value higher than 0.95. Nevertheless, it is clearly that number of rounds affects the cipher differently, not to a great extent though. With number of rounds being increased, the average test value is highest at 8th and 14th rounds. At 15 rounds the average value is lowest, though by ~0.3%. Moreover, at 15 rounds our system fails in 5 tests. According to test results we can conclude, that the most optimal in terms of cryptographic strength, as well as speed is an 8-round variant of cipher, which has a maximum test value not lower than 0.97. Good results are observed also at 14 rounds.

Overall, as a result of our research, we can make conclusion, that the developed three-dimensional-CA-based block cipher has shown a sufficient level of cryptographic strength for use.

C. STREAM CIPHER BASED ON CELLULAR AUTOMATA

The alternate solution to block ciphers for encryption of communication channels can be the stream ciphers, based on cryptographically strong pseudo-random/random binary sequence generators. The sequences, produced by hardware, software of hardware-software generator, are XORed bitwise to the open message, resulting in encrypted text, properties of which are completely ensured by cryptographic strength of the generated sequence.

The computer hardware-produced random sequences are often used in implementation of a quality generator. In such cases it is better not to use the built-in microprocessor generators, but instead generate the hardware sequence based on user's actions: capture the time of keystrokes (or time between sequential keystrokes) or track the changes of mouse pointer movement direction, etc.

We developed applicable software, which gives an opportunity for user to generate such sequence and works in three modes with ability of their combination:

1) measurement of keystroke time (milliseconds);
2) measurement of time between keystrokes (milliseconds);
3) tracking when the mouse pointer changes its movement direction.

The hardware entropy, generated by these means, is sequentially hashed by cryptographic hash function based on cellular automata. As the result, we get a hardware-software generator of key gamut, which can be used to create a stream cipher.

For hash function implementation we used the architecture of "cryptographic sponge", proposed by the authors of the SHA-3 hash function [25], shown in Fig. 3.

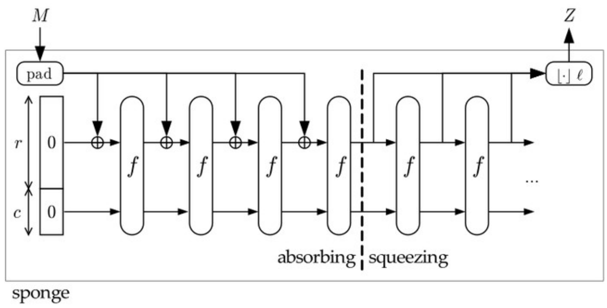

Figure 3. A scheme of "cryptographic sponge"

The workflow of such structure consists of two stages: "absorbing" – when input data is added blockwise to a mixing function, and "squeezing" – when the hashing result is supplied blockwise to the output of hashing function. This way, the required length of resulting hash digest is achieved. We used a mixing function of our own development; it is based on cellular automata. The operating array $r+c$, which is called “state” and has a size of 1600 bits, is filled with input vector M (generated by hardware part) and padded with required amount of zeros $M||0^k$, where $k=[r+c]-[M]$. Thus, we have 25 ULONG elements (64-bit unsigned integers).

The array (matrix $A[5x5]$) is broken into 5 intermediary blocks, computed by XOR operation like this:

$B[0]=A[0]\oplus A[5]\oplus A[10]\oplus A[15]\oplus A[20]$,

$B[1]=A[21]\oplus A[22]\oplus A[23]\oplus A[24]\oplus A[19]$,

$B[2]=A[14]\oplus A[9]\oplus A[4]\oplus A[3]\oplus A[2]$,

$B[3]=A[1]\oplus A[6]\oplus A[11]\oplus A[16]\oplus A[17]$,

$B[4]=A[18]\oplus A[13]\oplus A[8]\oplus A[7]\oplus A[12]$.

The elements of array B[i] are used to build additional elements $C[i]$ in this way: $C[i]=(B[j]\ll i)\oplus B[j+1]$, where i=0-14, $j=i$ mod 5.

After such computations we can fill the state matrix with new elements. The filling is done by columns: first – $A[0]$-$A[4]$, second – $B[0]$-$B[4]$; the remaining columns are filled sequentially with $C[i]$.

In the second stage of state matrix shuffling rules of the elementary cellular automata are involved. These are the rules “30” and “146”. They are chosen, because they show a chaotic and unpredictable behavior, which is crucial for hash function. Firstly, the rule “30” is used, then – rule “146”. Rules are applied spirally and clockwise: $A[i]=((A[i])_{R30})_{R146}$. Here, the indices $R30$ and $R146$ indicate the application of corresponding elementary CA rules to the array elements.

To improve the avalanche effect, the final permutation of state elements with bitwise circular shift is performed: $A[i]=\mathrm{Sh}_K(A[i+3])$, $\boldsymbol{K} \in \{\mathbf{7, 11, 13, 23, 29}\}$, Sh_K – circular bitwise shift to the left by K steps, i=0-24.

After finishing the “absorbing” stage, the “squeezing” stage is executed, resulting in a random stream of bits, which can be used for stream encryption.

III. RESULTS OF STATISTICAL TESTS

To evaluate the developed hash function (as well as the whole stream cipher) we used NIST STS. The conducted testing was just like the previous. The results of the statistical tests of our developed hash function were compared to SHA-3 (look Table 3).

Table 3. Results of hash function statistical evaluation

Test passing rate, %	SHA-3 (Keccak) based on cellular automata	Classic SHA-3 (Keccak)
100	79 (42.02%)	58 (31.01%)
99	64 (34.04%)	68 (36.36%)
98	27 (14.3%)	45 (24.06%)
97	15 (7.97%)	13 (6.95%)
96	3 (1.65%)	1 (0.53%)
95	0 (0%)	1 (0.53%)
<95	0 (0%)	1 (0.53%)

As we can see from the table, the statistical results of our developed hash function are at least not worse than those of a classic SHA-3.

For the avalanche effect evaluation the three variants of a famous pangram-sentence “The quick brown fox jumps over the lazy dog”, that contains all letters of the English alphabet:

1) The quick brown fox jumps over the lazy dog;

2) The quick brown fox jumps over the lazy dog. (full stop at the end of sentence);

3) The quick brown fox jumps over the lazy dog (starting with lowercase letter);

The resulting hash digests of our algorithm are given below:

1) 42063cf95300891402c6cb0913c788ab50bf8c2cd2f4dc1781fdebafcfb7679928c60d41609fd80165b7e63eebdc78ab7656a89af1e5e218a87cc3cf475be4d1;

2) 56b949ac21cf22a78c1d92bf01ef20937f8db80083a84cff1d0879b9b54d6fd9b787a0ab4539b8bfb0e57c002749bd5a8fdf37c2777b3cceece256a53df66a51;

3) ce8f6d5fc67cd52933ccba3836f9470b32a896a9545682a9850447df691ec39c59af4504547020e885d5ffef192eb1baa7af650ca7dd30413929c12f233f25b3.

As it is seen, when one symbol is added or when a letter changes its case, the hash value changes completely that indicates a good avalanche effect.

Therefore, the suggested methods can be successfully applied for generation of binary random sequence on a basis of hardware-software solutions.

In Fig. 4 the results diagram of statistical evaluation of the developed binary random stream generator based on cellular automata are given. We can observe that suggested method gives good statistical results.

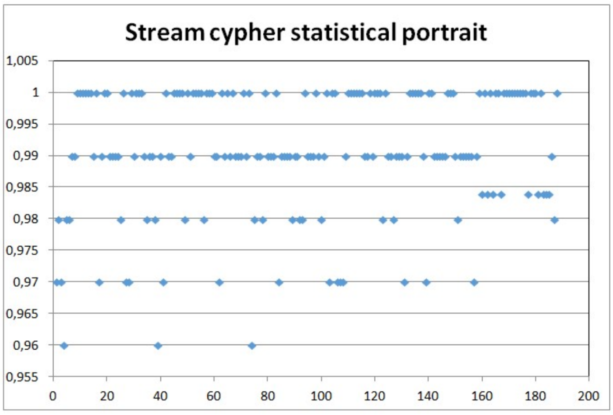

Figure 4. Results of statistical tests of the developed binary random sequence generator, based on CA; on the X-axis – the index of test, on Y-axis – the probability of passing

As we can see in Fig. 4, our developed stream cipher, based on CA, passes all NIST STS test, which reflects the relevance of applied transformations, including those that use cellular automata. Thus, we can recommend this encryption algorithm for protection of sensitive data during their transmission via telecommunication systems.

IV. CONCLUSIONS

The statistical characteristics of block and stream ciphers, based on cellular automata is developed and examined. It is shown that they can successfully be used for cryptographic applications as efficient means of incoming message obfuscation.

For development of the block cipher, a three-dimensional cellular automaton with rules “22”, “105” and “150” has been used. The substitution boxes from well-known AES cipher have been utilized. The own round function and key schedule operation is developed.

A hardware-software binary random sequence generator based on cellular automata is implemented.

As a hardware platform for generation of initial entropy, a keyboard and mouse of personal computer are used. For the following processing of the received random sequence, the own hash function with the architecture of “cryptographic sponge” is developed, as well as an own permutation function, where rules of the elementary CA “30” and “146” are applied.

As a result, a binary random sequence generator is obtained, which can successfully be used for stream encryption.

A statistical testing of the developed cryptographic algorithms was done by NIST statistical test suite. It showed the strong statistical characteristics of both ciphers.

The performed studies have also shown a good avalanche effect of the designed CA-based hash function.

To summarize, we can surely claim, that the application of cellular automata, both one-dimensional and multi-dimensional, allows engineers to construct simple yet efficient cryptographic structures that can be used to create high- quality means of protecting information confidentiality and integrity.

References

[1] Law of Ukraine “On the Basic Principles of Cybersecurity in Ukraine,” *The Official Bulletin of the Verkhovna Rada*, №45, Article 403, 2017. (in Ukrainian). [Online]. Available at: https://zakon.rada.gov.ua/laws/show/2163-19

[2] Law of Ukraine “On the modification of some laws of Ukraine about implementation of government policy in the field of active counteractions to the aggression in cyberspace,”. (in Ukrainian). [Online]. Available at: https://zakon.rada.gov.ua/laws/show/2470-20

[3] Presidential decree of Ukraine №37/2022 “On decision of National Security and Defense Council of Ukraine since 30th December 2021 “On Plan of Ukraine cybersecurity Strategy implementation,”. (in Ukrainian). [Online]. https://www.president.gov.ua/documents/372022-41289

[4] Y. Sbaytri, S. Lazaar, “A design of a new hash function based on cellular automata,” *Journal of Theoretical and Applied Information Technology*, vol. 99, issue 10, pp. 2280-2289, 2021. [Online]. Available at: http://www.jatit.org/volumes/Vol99No10/9Vol99No10.pdf

[5] A. E. Belfedhal, K. M. Faraoun, “Building secure and fast cryptographic hash functions using programmable cellular automata,” *Journal of Computing and Information Technology*, vol. 23, issue 4, pp. 317-328, 2015. https://doi.org/10.2498/cit.1002639

[6] B. Applebaum, Y.Ishai, E. Kushilevitz, Cryptography by Cellular Automata or How F ast Can Complexity Emerge in Nature?, [Online]. Available at: http://www.eng.tau.ac.il/~bennyap/pubs/CA.pdf

[7] S. Ostapov, O. Val, S. Yanushevsky, D. Chyzhevsky, *Cryptography on the Base of Cellular Automata*, In book: Internet in the Information Society. Publisher: Scientific Publishing University of Dabrowa Gornicza, Editors: Maciej Rostanski, Piotr Pikiewicz, Pawel Buchwald, 2015, pp. 71-86.

[8] R. Parvaz, Y. Khedmati, Y. Behroo, *Four-dimensional Hybrid Chaos System and its Application in Creating a Secure Image Transfer Environment by Cellular Automata*, arXiv:2110.15196 https://doi.org/10.48550/arXiv.2110.15196

[9] V. Zhikharevich, S. Ostapov, “Self-organization and evolution system simulation by the continuous nonsynchronizing cellular automata,” *International Journal of Computing*, vol. 8, issue 3, pp. 61-71, 2014. https://doi.org/10.47839/ijc.8.3.686

[10] M. Ghosh, R. Kumar, M. Saha and B. K. Sikdar, “Cellular Automata and its Applications,” Proceedings of the 2018 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS), 2018, pp. 52-56. https://doi.org/10.1109/I2CACIS.2018.8603689

[11] H. Fukś, “Four state deterministic cellular automaton rule emulating random diffusion,” In: Chopard, B., Bandini, S., Dennunzio, A., Arabi Haddad, M. (eds) Cellular Automata. ACRI 2022. Lecture Notes in Computer Science, vol. 13402, 2022. Springer, Cham. https://doi.org/10.1007/978-3-031-14926-9_13

[12] S. Wolfram, E. B. White, *A New Kind of Science*, Wolfram Media, 2002, 1197 p. ISBN_1-57955-008-8.

[13] A. Clarridge, K. Salomaa, *A Cryptosystem Based on the Composition of Reversible Cellular Automata*, Technical Report No. 2008-549, Berlin, Heidelberg, 2009, 314-325 pp. https://doi.org/10.1007/978-3-642-00982-2_27

[14] O. Val′, V. Zhikharevich, R. Ovchar, S. Ostapov, “Development and investigation of the key stream generators on the base of cellular automata,” *Radio Electronics, Computer Science, Control*, vol. 3, issue 34, pp. 58-63, 2015. https://doi.org/10.15588/1607-3274-2015-3-7

[15] A. Cicuttin, L. De Micco, M. L. Crespo, et al., “Looking for suitable rules for true random number generation with asynchronous cellular automata,” *Nonlinear Dynamics*, vol. 111, pp. 2711-2722, 2022. https://doi.org/10.1007/s11071-022-07957-8

[16] L. Li, Y. Luo, S. Qiu, et al., “Image encryption using chaotic map and cellular automata,” *Multimed Tools Appl*, vol. 81, pp. 40755–40773, 2022. https://doi.org/10.1007/s11042-022-12621-9

[17] M. Kutrib, A. Malcher, “One-dimensional pattern generation by cellular automata,” *Nat Comput*, vol. 21, pp. 361–375, 2022. https://doi.org/10.1007/s11047-022-09887-1

[18] J. Gravner X. Liu, “One-dimensional cellular automata with random rules: longest temporal period of a periodic solution,” *J. Probab.*, vol. 27, article no. 25, pp.1–23, 2022. https://doi.org/10.1214/22-EJP744

[19] K. Bhattacharjee, S. Das, “A search for good pseudo-random number generators: Survey and empirical studies,” *Computer Science Review*, vol. 45, 100471, 2022. https://doi.org/10.1016/j.cosrev.2022.100471

[20] C. Hanin, A. Sadak, F. Ziani, F. Omary, K. Achkoun, “SPF-CA-1.2: an enhanced version of cellular automata-based block cipher system,” *International Journal of Computer Mathematics*, vol. 6, no. 3, pp. 194-208, 2021. https://doi.org/10.1080/23799927.2021.1942991

[21] M. Kutrib and A. Malcher, “String generation by cellular automata,” *Complex Systems*, vol. 30, issue 2, pp. 111–132, 2021. https://doi.org/10.25088/ComplexSystems.30.2.111

[22] E. R. Lira, H. B. de Macêdo, D. A. Lima, et al., *A Reversible System based on Hybrid Toggle Radius-4 Cellular Automata and its Application as a Block Cipher*, arXiv:2106.04777, 2021. https://doi.org/10.48550/arXiv.2106.04777

[23] S. Maiti, D. R. Chowdhury, “Design of fault-resilient S-boxes for AES-like block ciphers,” *Cryptogr. Commun.*, vol. 13, pp. 71–100, 2021. https://doi.org/10.1007/s12095-020-00452-0

[24] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, NIST Computer Resource Center, 2010, [Online]. Available at: https://csrc.nist.gov/publications/detail/sp/800-22/rev-1a/final

[25] Federal Information Processing Standards Publication PUB 202, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, 2015. https://doi.org/10.6028/NIST.FIPS.202

***PROF. SERHII OSTAPOV**, Doctor of Physical and Mathematical Sciences. Head of the Computer Systems Software Department of Yuriy Fedkovych Chernivtsi National University.*

Field of scientific interests: information security, analysis of complex networks, software for scientific research, modeling based on cellular automata.

https://orcid.org/0000-0002-4139-4152

email: s.ostapov@chnu.edu.ua

BOHDAN DIAKONENKO, *bachelor of Software Engineering, master student of Computer System Software Department of Yuriy Fedkovych Chernivtsi National University.*

Field of scientific interests: information security, network analysis, cellular automata application.

E-Mail: diakonenko.bohdan@chnu.edu.ua

MAKSYM FYLYPIUK, *Master of Software Engineering. Software Development Engineer.*

Field of scientific interests: software-hardware binary generators on the base of cellular automata.

E-mail: mfylypiuk@gmail.com

KATERYNA HAZDIUK, *Doctor of Philosophy (PhD). Assistant of the Computer Systems Software Department of Yuriy Fedkovych Chernivtsi National University.*

Field of scientific interests: modeling of biological processes and systems by the method of mobile cellular automata.

https://orcid.org/0000-0002-7568-4422

email: k.gazdiuk@chnu.edu.ua

Liliia SHUMYLIAK, *Candidate of Technical Sciences. An Assistant of the Computer Systems Software Department of Yuriy Fedkovych Chernivtsi National University.*

Field of scientific interests: cellular automata modeling, artificial intelligence, cryptography.

https://orcid.org/0000-0002-6593-7334

email: l.shumylyak@chnu.edu.ua

OLHA TARNOVETSKA, *Candidate of Physical and Mathematical Sciences. An Assistant of the Computer Systems Software Department of Yuriy Fedkovych Chernivtsi National University.*

Field of scientific interests: asymptotic behavior of functions represented by functional series and integrals, theory of distribution of values of whole functions, crypto-graphy.

email: o.tarnovetska@chnu.edu.ua.