The Seybold Report

ISSN 1533-9211

DOI: 10.5281/zenodo.14598679

ASYNCHRONOUS METHOD FOR GENERATING A STREAM CIPHER BASED ON CELLULAR AUTOMATA

M.M. AL- HIARI

Faculty of Computer Science and Information Technology, Department of Cyber Security, Jerash University, Jerash, Jordan. E-mail: hiari5355@yahoo.com

Abstract

The paper discusses the methodology for constructing a self-synchronizing stream cipher based on the rules of elementary cellular automata. The objective of the research is to increase the reliability of stream message encryption, which is achieved by implementing a varying initial number of bits that form the key gamma. To solve this problem, elementary cellular automata of a given dimension were used, in which the initial settings were specified by the bits of the first N bits of the message. Based on the generated evolutions of elementary cellular automata, a key gamma was formed, which, using the XOR operation, generated a ciphergram in real time. To select the most suitable transition rule for cellular automata, experiments were carried out in which various rules were analyzed. As a result of the analysis of the results obtained, the most suitable transition rules for cellular automata, as well as their combinations, were selected. For more reliable stream encryption, it is proposed to periodically change the dimension of the cellular automaton, as well as the rule of its transitions. The proposed stream cipher can be easily implemented in both software and hardware.

Keywords: Self-Synchronous Stream Cipher, Elementary Cellular Automaton, Evolution, Wolfram's Rule.

INTRODUCTION

In modern conditions of human activity, high-quality communication is very important. The existence of modern information transmission systems and the Internet has entailed the creation of a parallel information environment, which is a communication web that has entangled the entire globe. In this communication environment, large amounts of data are constantly transmitted, which must not be subject to distortion. Also, the data often should not be accessible to many users of the data network. Therefore, many methods and means of security it have been developed and implemented [1 - 4].

Data in the transmission network can be protected by different methods [1 - 4]. Such security methods can be divided into: methods that secure against physical unauthorized access to means of information transmission [5] and methods that secure information that is transmitted in the public domain [6]. Methods of secureting information of the first group include modern fiber-optic data transmission systems [5], the use of various specialized devices and other physical methods that block access to the communication system. The second group of methods includes information security methods based on cryptography and steganography [7, 8].

Much attention is paid to methods of secureting transmitted information during its transmission in real time. In this case, it is better to use open access transmission networks, since closed access networks are much more expensive, and they also have their own specifics in the implementation of physical communications. Easier to use existing open communication tools.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14598679

However, the use of such networks requires reliable methods of encryption and transmission of information without delays and a significant increase in volumes.

Among the ciphers that are generated in real time, stream ciphers, which are based on bit encryption, stand out [10, 11]. For such ciphers, an important element is the generator that generates the key gamma. Typically, such a generator is a pseudo-random bit sequence generator (PRBSG) [10]. The key in such ciphers is the structure of the PRBSG and its initial settings [10]. In addition, the length of the repetition period of the pseudo-random bit sequence at the PRBSG output plays an important role. Ideally, this length should exceed the length of the secret message. There are two types of stream ciphers: synchronous [12, 13] and asynchronous [12 - 15] stream ciphers. Synchronous stream ciphers require tight synchronization on the sending and receiving sides. This circumstance often limits the use of synchronous stream ciphers and does not benefit their development. This limitation is especially true for users located over large distances. Asynchronous stream ciphers attract particular attention. The only limitations of such ciphers is the mandatory use of a certain number of the first bits of the message to set the initial settings of the PRBSG or its configuration. The paper discusses a method for generating an asynchronous (self-synchronous) stream cipher based on cellular automata.

Problem statement

For real-time message encryption, one of the main tasks is to represent the input bit sequence without containing statistical connections with the original message. The most effective is the use of stream encryption methods, which are based on representing the initial message as a sequence of bits and mixing these bits with bits generated by a key gamma generator. All existing methods are implemented based on shift registers, chaotic ciphers, and also use cellular automata for image encryption. Most of these methods require a large amount of computation and are used to encrypt objects represented by large amounts of data (for example, images). For encrypting rapidly changing information in real time, known methods do not always provide a high-quality stream cipher. Therefore, the task of generating a stream cipher in real time is still relevant.

The work solves the problem of developing a method for stream encryption of images with a variable number of first bits, which carry out initial settings based on cellular automata (CA) technologies. The problem posed is solved by generating evolutions of elementary cellular automata with different dimensions, the cell states of which are sources for the formation of encryption key bits.

Relative works

Stream ciphers are divided into synchronous and asynchronous. Both are symmetric ciphers. Asynchronous stream ciphers require a certain number of the first bits of the message to form and set the PRBSG. A large number of stream ciphers are based on linear feedback shift registers (LSFSR) [10, 16]. For such self-synchronizing stream ciphers, the number of bits in the shift register determines the number of initial setup bits. One of the disadvantages of such ciphers is the limited length of the repetition period of the pseudo-random bit sequence, which

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14598679

is determined by the length of the shift register. Also, often the output sequence in such PRBSGs is easily predictable. The initial filling of the LSFSR can be determined by solving a system of linear equations. To destroy linear complexity, a nonlinear combination of several LSFSRs is used [17]. For example, a Geff generator can be used [18, 19]. However, this generator is also not resistant to correlation attacks.

Currently, chaotic self-synchronizing stream ciphers are widely used [20, 21]. These ciphers are built on the basis of chaotic cards. These self-synchronous chaotic stream ciphers use a large number of computational operations and are most suitable for encrypting information that does not change rapidly (for example, images). However, for real-time encryption, encryption failures may occur.

There is also a self-synchronizing stream cipher called MOUSTIQUE [22]. This cipher is software-specific and has limitations in key size, input memory, and special encryption function, which limit its use.

Much research is devoted to the development of stream ciphers implemented on the basis of cellular automata [10, 23, 24]. These stream ciphers are mainly used to encrypt bit sequences in real time. Work [24] considers the use of elementary cellular automata for image encryption. Various combinations of rules are presented for a fixed dimension of the cellular automaton. This situation requires a combination of five evolution rules to obtain high quality encryption, which complicates the encryption process. The dimension of an elementary cellular automaton is determined by the dimension of the image and remains unchanged throughout the entire encryption process. The number of evolutionary steps is also determined by the dimension of the cellular automaton. Which requires using a combination of multiple rules for each bit layer of the image.

In terms of creating self-synchronizing stream ciphers with varying initial states of key gamma generators, elementary cellular automata were not used. In this direction, the presented work is relevant and has scientific novelty.

Asynchronous stream cipher with variable key length based on cellular automata

Both synchronous and asynchronous stream ciphers mostly require a pseudo-random bit sequence generator, which generates a key gamma to generate a stream ciphergram. If in synchronous stream ciphers the PRBSG begins to work from the first bit of the input message, then in asynchronous stream ciphers the key bit sequence begins to form after the formation of a certain number of the first bits of the input message. As a rule, these first bits carry out the initial settings of the already built-in PRBSG, the configuration of which is already set on the transmitting and receiving sides. This paper proposes an asynchronous stream cipher, which is implemented by encryption and decryption tools with a built-in key gamma generator implemented on an elementary cellular automaton (ECA). Such generators are considered in [25, 26]. Generators are implemented based on ECA evolutions according to the rules chosen for this. In classical ECA, evolution can be viewed based on 256 rules (from 0 to 255), which are described in detail in the works of Stephen Wolfram [25, 26]. Each next step of evolution is considered based on the states of the cells of the previous step of evolution. In this case, the

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14598679

state of one cell of each evolution step is formed based on the analysis of the states of three cells of the previous evolution step. The cell's own state and the states of two neighboring cells are analyzed. For convenience, ECAs are presented in expanded form. The outermost cells in each row are adjacent. Examples of ECA evolutions for rules: 30, 45, 51, 77, 90, 105, 107, 111, 150, 165, 184, 214 in Fig. 1 are shown.

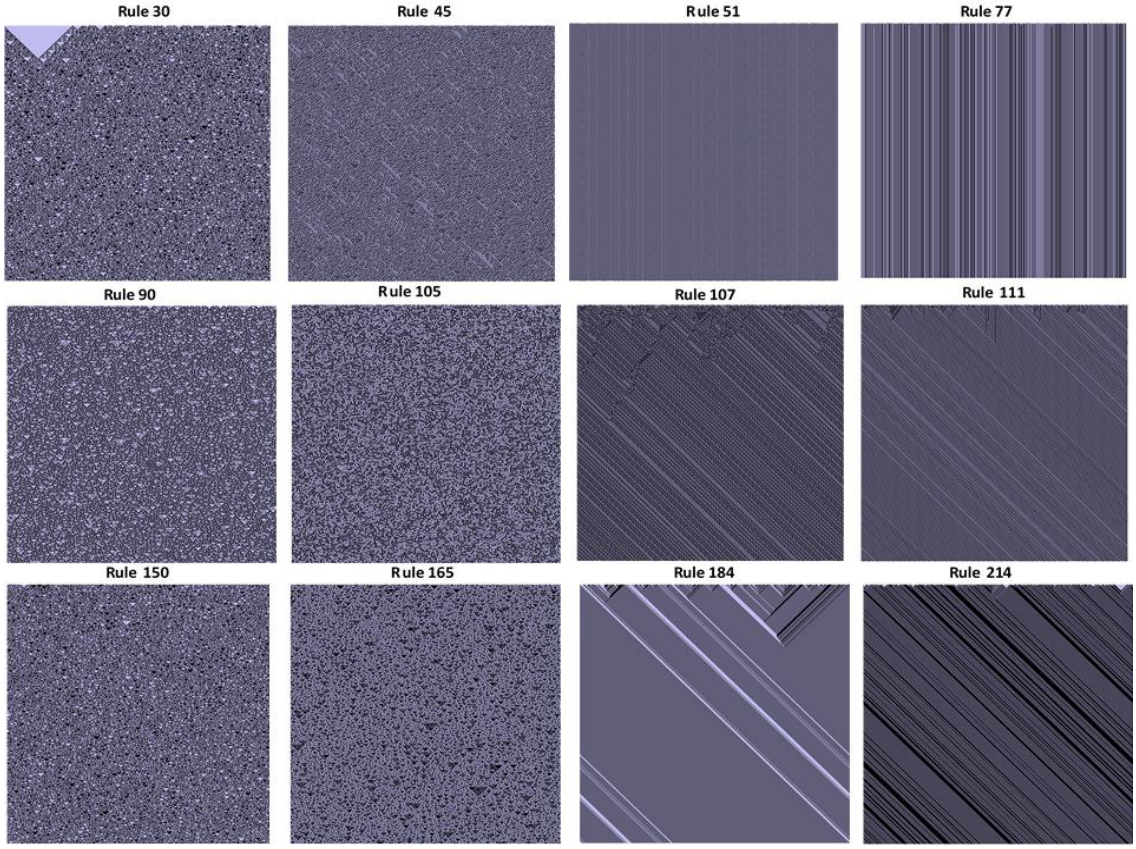

Fig 1: Examples of ECA evolutions for rules: 30, 45, 51, 77, 90, 105, 107, 111, 150, 165, 184, 214. 700 evolutionary steps are used

Using visual analysis, can select the appropriate ECA transition rule, which allows you to obtain a high-quality ciphergram that is resistant to various types of attacks. A visual analysis of the presented ECA evolutions shows that the most suitable rules are: 30, 45, 90, 105, 150 and 165. When implementing rule 30, initial settings with a large number of cells with logical "0" states were used. This significantly affected the first steps of evolution, which do not provide high quality encryption. Therefore, it is better to continue encryption through several evolution steps. The first specified bits of the message, starting from the (N+1) th bit, are encrypted with the bits obtained at the zero evolution step (N is the number of initial setup bits). During the arrival time of N bits, N evolution steps are formed. In this case, N can be a sufficiently large number and then a smaller number of evolutionary steps can be used. For example, after K evolutionary steps (K≤N), as shown in Fig. 2.

ISSN 1533-9211

DOI: 10.5281/zenodo.14598679

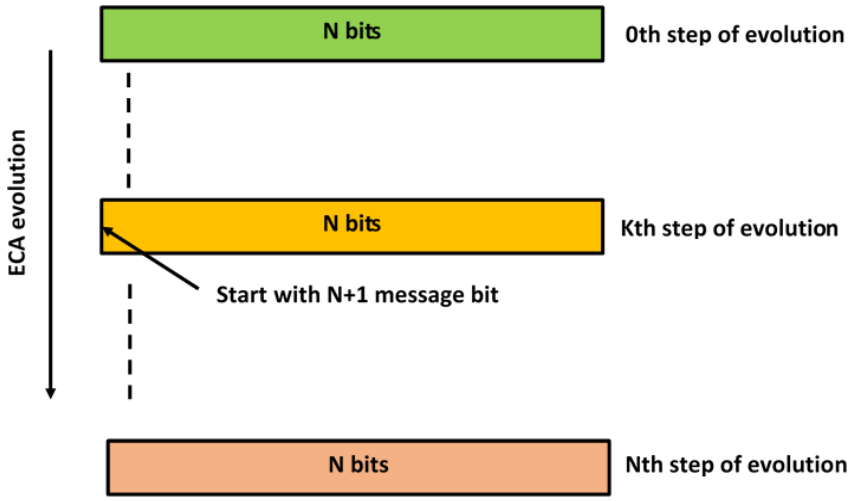

Fig 2: An example of forming an evolution in N steps, where N is the number of first bits of the message

In fact, during the formation of the first N bits of the message, N evolutionary steps of the ECA are formed according to the selected transition rule. Therefore, N+1 bits of the message and subsequent bits of the message can already be encrypted with bits generated at the Kth evolutionary step. In this case, it is difficult for the enemy to determine the algorithm and structure of the key gamma generator. Also, to increase the reliability of encryption, it is necessary to use a certain number of bits from each evolutionary step. For example, the key gamma can be formed by the first 10 bits obtained at the Kth evolutionary step, bits from 11 to 20 obtained at the (K+1) th evolutionary step, etc.

For clarity, let's imagine a message consisting of 250,000 bits in the form of a two-dimensional binary array of size 500×500 (Fig. 3). Where black represents logical "1s" and white represents logical "0s". The first bits of the message start at the left edge of the top row of the array, the 500th bit is the last right bit in the first row of the array, and the 501st bit of the message is the first left bit of the second row of the array, etc.

Fig 3: An example of an input message represented as a binary array of size 500×500

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14598679

For the convenience of encryption, the first 500 bits of the message are used, which form the initial states of the ECA cells. Bits are encrypted using the XOR operation. Examples of encryption of this binary array in Fig. 4 are presented. The number of evolutionary steps corresponds to 500, which is sufficient to encrypt this array with an ECA dimension of 500 cells.

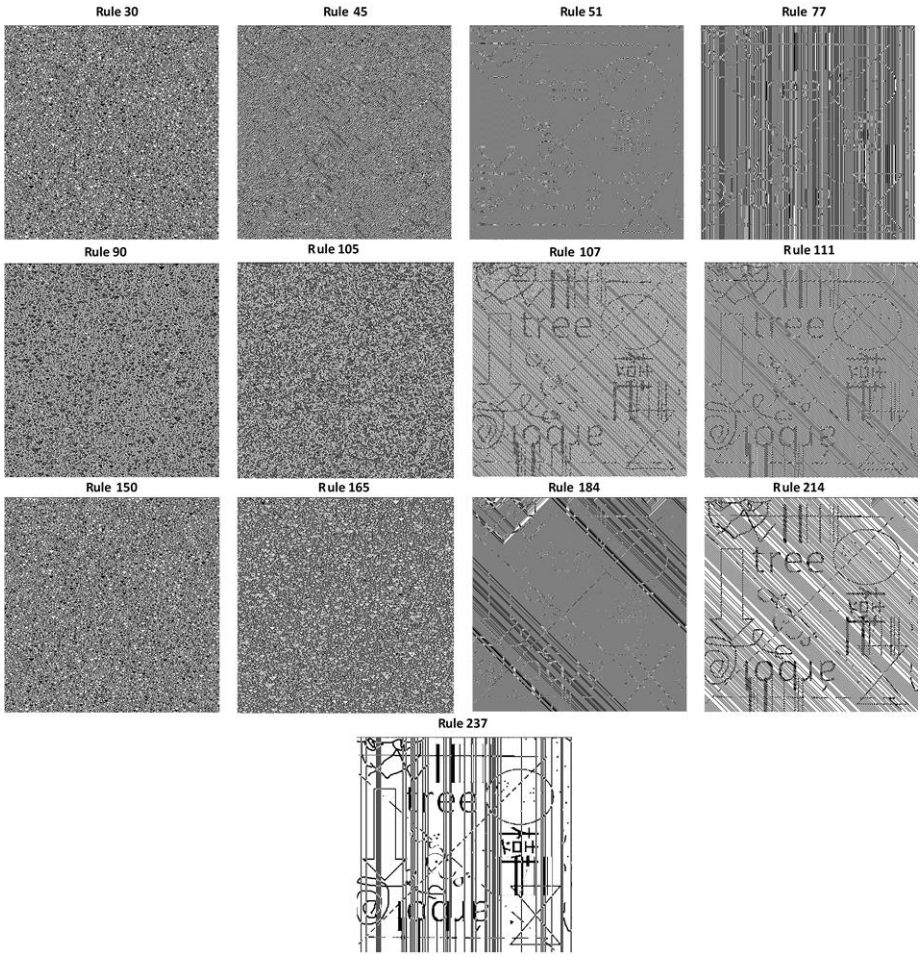

Fig 4: Examples of encryption of a binary array shown in Fig. 3 using ECA rules: 30, 45, 51, 77, 90, 105, 107, 111, 150, 165, 184, 214 and 237

Analysis of Figure 4 showed that rules 51, 77, 107, 111, 184, 214 and 237 do not provide high quality encryption, which indicates the impossibility of using these rules. Their use is possible in combination with other rules. If we use the same rules for the same array with fewer initial setup bits, we will get a completely different encryption picture. For example, if N=100 bits, then the result

ISSN 1533-9211

DOI: 10.5281/zenodo.14598679

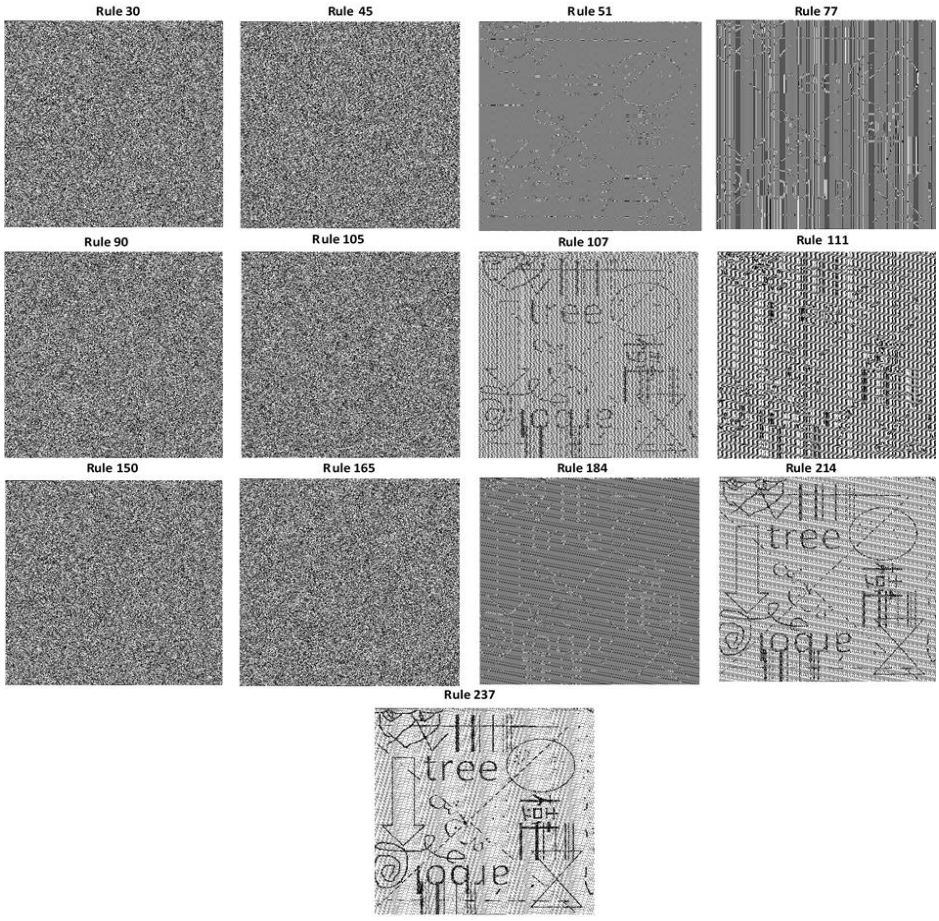

Fig 5: An example of encryption of a binary array shown in Fig. 3 at N=100

In Figure 5 are shows the results for ECA rules: 30, 45, 51, 77, 90, 105, 107, 111, 150, 165, 184, 214 and 237.The following rules turned out to be unreliable for encryption: 51, 77, 107, 184, 214 and 237. Rule 111 requires additional research using special tests. However, if there are doubts during visual analysis, then it is better not to use this rule.

This example shows different pictures of bit array encryption when using different ECA dimensions. In this case, it is necessary to vary the evolution by changing the ECA dimension at various subsequent evolutionary steps. This variation of dimensions at different stages of evolution allows us to obtain a stream cipher with high resistance to various types of attacks. For example, at the initial steps of evolution N setup bits were used, and at K+1 step N-Q bits can be used. After the (K+1) – th step of evolution, the structure of the stream cipher changes abruptly. Even if Q=1, the stream cipher differs significantly from the cipher obtained in the previous steps of the ECA evolution. A combination of rules that provide high quality encryption allows to increase the repetition period of the bit sequence, which makes the cipher more resistant to attacks. At the same time, it is better to apply different rules at different steps of the evolution of one first rule, which does not allow an adversary to break the entire cipher

ISSN 1533-9211

DOI: 10.5281/zenodo.14598679

if he somehow became aware of the initial part of the stream cipher, which is the least secure. The use of a combination of untrusted rules in an XOR operation with an input bitmap was also explored. The results of such a study in Fig. 6 are presented.

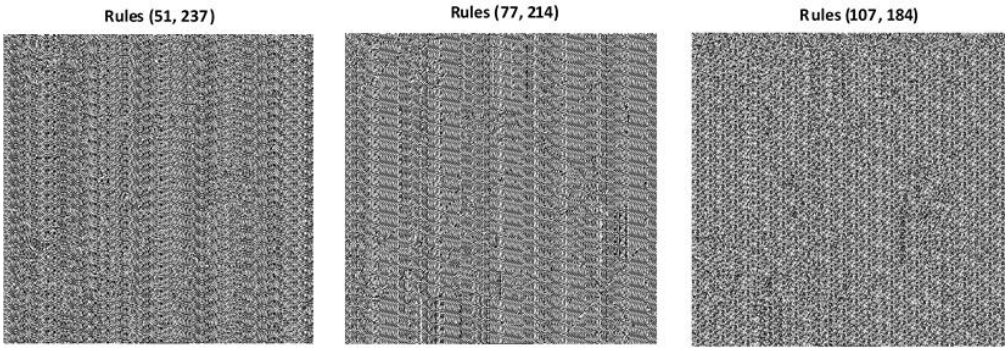

Fig 6: Examples of encryption of a binary array shown in Fig. 3, using combinations of ECA rules: (51, 237), (77, 214) and (107, 184)

The results presented in Fig. 7 does not visually allow to determine the input binary array. Such results require additional research based on special tests. Research is needed to find the best combinations of ECA rules. If you use more combinations of rules, i.e. complicate the overall evolution based on the XOR operation, then you can get good encryption results. At the same time, complicating evolution using rules that does not give high quality generally does not give the desired result. Some combinations of evolutions of such rules in Fig. 7 are presented.

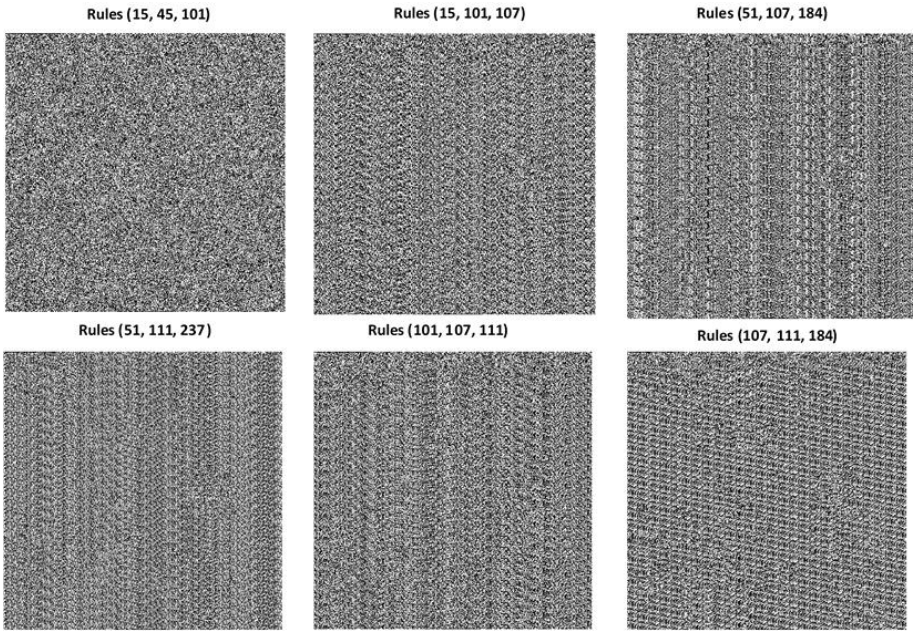

Fig 7: Examples of encrypting a binary array using complicated evolutions using a combination of ECA rules

ISSN 1533-9211

DOI: 10.5281/zenodo.14598679

In Fig. 7 shows combinations of ECA rules that do not provide high quality encryption. This is easily determined using visual analysis. However, in Figure 7 provides an example that shows a combination of rules 15, 45 and 101. Here, rules 15 and 101, both individually and together, do not show high quality encryption. Adding rule 45 even visually allows to verify the high quality of encryption. This indicates that having one desired rule (in this example, rule 45) leads to good encryption results. For a more detailed analysis of the proposed stream cipher, the method of constructing a number distribution diagram was used [10]. This method consists of dividing the bit sequence into groups of eight bits (1 byte) and converting each group into a decimal number. After this, a diagram of the distribution of numbers in the range from 0 to 255 is built. Each histogram position corresponding to one of the numbers determines how many times this number occurs in the generated bit sequence.

In the example presented in Fig. 3, 250000 bits are used, which corresponds to 31250 decimal numbers. For each ECA rule, a histogram of the distribution of numbers was built, as well as for a combination of rules. The number of initial bits was assumed to be 100 (N=100). Histograms of the distribution of numbers for rules 45, 90, 105, and 150 and combinations of rules (51, 237), (77, 214), (15, 45, 101) and (101, 107, 111) in Fig. 8 are presented.

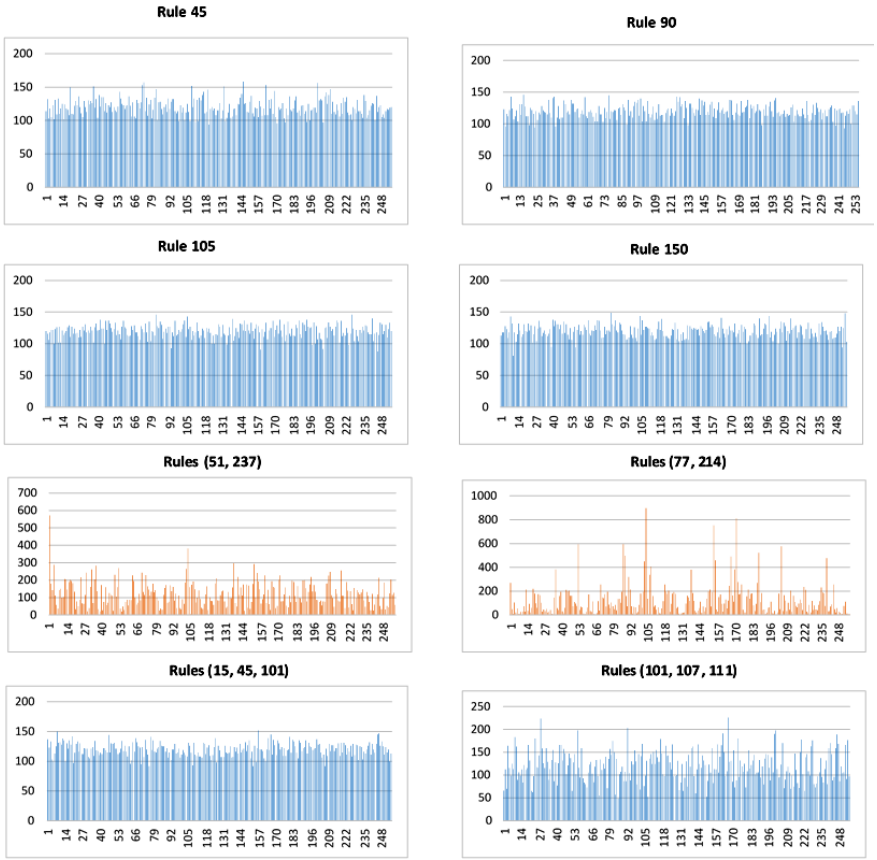

Fig 8: Distribution histograms for rules 45, 90, 105, and 150 and combinations of rules (51, 237), (77, 214), (15, 45, 101) and (101, 107, 111)

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14598679

As a result of the analysis of the presented histograms of the distribution of numbers, it was established that an even distribution of numbers is shown by the histograms constructed for the rules 45, 90, 105, 150 and (15, 45, 101). As can be seen from the rules 15, 51, 77, 101, 107, 111, 214, 237 and their combinations (51, 237), (77, 214), (15, 45, 101) and (101, 107, 111) numbers are not distributed evenly. Therefore, these rules cannot be used for encryption. If a combination of rules contains at least one rule that individually provides high quality encryption, then the distribution of numbers is uniform. The results of the analysis of the use of ECA rules for encryption are presented for messages consisting of a small number of bits. For a large number of bits, there is no need to use a large memory array to store the key gamma bits resulting from the evolution of ECA. It is enough to carry out a cycle of a small number of evolutionary steps. If it is necessary to repeat the evolution, it is sufficient to store the initial state of the ECA and the rules that implement the evolution. If a combination of evolutions is used, the key gamma generator becomes more complicated.

To improve the quality of encryption, the number of first bits of the message can vary. In this case, it is necessary to change the connections between the first and N-th bits of the installation sequence, which somewhat complicates the structure of the PRBSG. However, it is more difficult for an adversary to recognize the structure of the generator, even if he knows that certain ECA rules are used for encryption. The PRBSG itself is simple in both software and hardware implementation and does not require complex calculations when generating key bits.

CONCLUSION

The paper considers and investigates an asynchronous stream cipher implemented on an elementary cellular automaton. An elementary cellular automata is used as a key gamma generator, the dimension of which can vary and is determined by the number of the first bits of the message. An experimental analysis was carried out of the evolutions that are formed by various rules of elementary cellular automata, which provide high quality encryption. To confirm the selected rules of transitions of cellular automata, histograms of the distribution of numbers in the ciphergram were constructed. The use of elementary cellular automata made it possible to create a flexible encryption system due to the possibility of constantly changing the number of the first bits of the message, which implement the initial settings of the generator. The possibility of combining several transition rules is considered. This made it possible to establish that a combination of transition rules that do not provide high quality encryption also does not provide the required quality of encryption. The key gamma generator can be implemented in both software and hardware, and also has a simple structure. The work shows that it is better to start encryption not from the first steps of evolution, but after a certain number of steps, which makes it difficult for the enemy to determine the first installation bits of the message. Due to the possibility of constantly changing the dimension of a cellular automaton at different stages of evolution, the reliability of the proposed self-synchronized stream cipher increases, which practically ensures high resistance to various types of attacks.

In further research, the author plans to implement an asynchronous stream cipher based on two-dimensional cellular automata.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14598679

References

1) Y. G. Paithankar. Transmission Network Protection: Theory and Practice. Routledge; 1st edition (2017). 383 pages.

2) Wojciech Mazurczyk, Steffen Wendzel, Sebastian Zander, Amir Houmansadr, KrzysztofSzczypiorski. Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Counter measures. Wiley-IEEE Press (2016). 296 pages

3) Joseph Migga Kizza. Guideto Computer Network Security. Springer; 5th ed. 2020. 620 pages.

4) Charlie Kaufman, Radia Perlman, Mike Speciner, Ray Perlner. Network Security: Private Communication in a Public World. Addison-Wesley Professional; 3rd edition (2022). 544 pages

5) Stamatios Kartalopoulos. Next Generation Intelligent Optical Networks: From Access to Backbone. Springer (2022). 592 pages.

6) QuinnKiser. Computer Networking and Cybersecurity: A Guide to Understanding Communications Systems, Internet Connections, and Network Security Along with Protection from Hacking and Cyber Security Threats. (2020). 194 pages

7) Niels Ferguson, Bruce Schneier, Tadayoshi Kohno. Cryptography Engineering: Design Principles and Practical Applications. Wiley; 1st edition (2010). 384 pages.

8) Stefan Katzenbeisser. Information Hiding Techniques for Steganography and Digital Watermarking. Artech House. (2022). 334 pages.

9) Stepan Bilan, Andrii Demash. High performance encryption tools of visual information based on cellular automata. - Information Technology and Security. - 2016. - Vol. 4, № 1(6). - C. 62-75.

10) StepanBilan. Formation Methods, Models, and Hardware Implementation of Pseudorandom Number Generators: Emerging Research and Opportunities. - (2017). - IGI Global, USA. - P. 301.

11) StepanBilan, MykolaBilan, SergiiBilan. Research of the method of pseudo-random number generation based on asynchronous cellular automata with several active cells. - MATEC Web of Conferences, - Vol. 125, - 02018 (2017), - P. 1-6.

12) AndreasKlein. StreamCiphers. Springer; 2013th edition (April 8, 2013). 742 pages.

13) Gerardus Blokdyk. Streamcipher A Complete Guide. 5STARCooks (March 10, 2022). 306 pages.

14) B. Chenetal. Cryptanalysis of some self-synchronous chaotic stream ciphers and their improved schemes. International Journal of Bifurcation and Chaos. (2021): 1- 26.

15) Amir Daneshgar, Fahimeh Mohebbipoor. A Secure Self-Synchronized Stream Cipher. The Computer Journal 61(8, 2017): 1-28

16) Maria George and Peter Alfke. Linear Feedback Shift Registers in Virtex Devices. XAPP210 (v1.3) April 30, 2007. Application Note: Virtex Series and Virtex-II Series. https://docs.xilinx.com/v/u/en-US/xapp210

17) Bruce Schneier. Applied Cryptography: Protocols, Algorithms and Source Code in C 20th Anniversary Edition, Wiley; 20th Anniversary edition (March 30, 2015), 784 pages

18) Maiya Din, Ashok K. Bhateja, RamRatan. Cryptanalysis of Geffe Generator Using Genetic Algorithm. ProceedingsoftheThirdInternationalConferenceonSoftComputingforProblemSolving. 2014. Advances in Intelligent Systems and Computing bookseries (AISC, volume 259).

19) F. Handayania) and N. P. R. Adiati. Analysis of Geffe Generator LFSR Properties on the Application of Algebraic Attack. Proceedings of the 4th International Symposium on Current Progress in Mathematics and Sciences (ISCPMS2018): 1-9.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.14598679

20) Baoju Chen. SiminYu. David D-U Li. JinhuLü. Cryptanalysis of Some Self-Synchronous Chaotic Stream Ciphers and Their Improved Schemes. June 2021. International Journal of Bifurcation and Chaos 31(08):2150142

21) Chunlei Fan ID and Qun Ding. A Novel Image Encryption Scheme Based on Self-Synchronous Chaotic Stream Cipher and Wavelet Transform. Entropy 2018, 20, 445: 1-13.

22) Joan Daemen1 and Paris Kitsos. The Self-Synchronizing Stream Cipher Moustique. New stream cipher Designes. Springer. (2008). 21—223.

23) Niyat, A. Y., Moattar, M. H., &Torshiz, M. N. (2017). Color image encryption based on hybrid hyper-chaotic system and cel-lular automata. Optics & Lasers in Engineering,90(March),225–237.

24) Nashatal Bdour. Image encryption methodology based on cellular automata. Journal of Theoretical and Applied Information Technology 15th December 2022. Vol.100. No 23. 6998-7004.

25) Wolfram, S. (1986). Random Sequence Generation by Cellular Automata. Advances in Applied Mathematics, 7(2), 429–432. doi:10.1016/0196- 8858(86)90028-X

26) Wolfram, S. (2002). A new kind of science. Wolfram Media.