Regular Article

Saule Nyssanbayeva, Nursulu Kapalova*, Armiyanbek Haumen and Olzhas Suleimenov

The LBC-3 lightweight encryption algorithm

https://doi.org/10.1515/eng-2022-0372
received May 13, 2022; accepted August 31, 2022

Abstract: This article presents a developed lightweight data encryption algorithm called LBC-3. Its essential difference from the known algorithms is the R1 function and the procedure for generating round keys. The main characteristics of this lightweight algorithm and all the transformations used in the encryption and decryption processes are described. The process of generating the round keys of the algorithm is also considered. The results of the study of the cryptographic properties of the algorithm using the "avalanche effect" and statistical tests are presented. The avalanche property was tested for each round with each bit of the source text changing. Based on the work carried out, it was found that the proposed encryption algorithm is effective in providing a good avalanche effect, and the binary sequence obtained after encryption is close to random. The research revealed good cryptographic properties of this algorithm.

Keywords: encryption algorithm, lightweight algorithm, cryptographic transformations, avalanche effect, cryptographic strength

1 Introduction

The main directions of the development of cryptography are largely associated with the development of communications, information technology, and computing hardware. It is the progress in these areas that has made possible the widespread use of compact devices with low computing power that have access to the Internet and implement the concept of the "Internet of Things." The Internet of Things is a wireless self-configuring network between objects such as household appliances, vehicles, various sensors, and detectors, as well as RFID (Radio Frequency IDentification) tags. Severe restrictions on the internal computing resources of such devices make it difficult or impossible to use classical cryptographic algorithms. This led to the emergence of a new section of cryptography – lightweight cryptography – aimed at creating strong cryptographic algorithms and protocols with acceptable strength in the context of limited resources [1,2].

Along with the effective implementation of known block cipher algorithms, work on the creation of new block ciphers focused on optimal implementation at the microprogram or hardware level for specialized applications is topical. Most of the research work (R&D) on the development of lightweight cryptoalgorithms has been carried out in the field of block cipher algorithms. Recently, many lightweight solutions have been proposed [3].

The National Institute of Standards and Technology has initiated a public standardization process to select one or more data-associated authentication encryption and hashing schemes suitable for resource-constrained devices.

In February 2019, 57 candidates were submitted to NIST for consideration. Of those, 56 were accepted as candidates in the first round in April 2019. Four months later, NIST selected 32 candidates for a second round. In March 2021, NIST announced 10 finalists who will advance to the final round of the selection process.

As a result, the following algorithms made it to the final: ASCON, Elephant, GIFT-COFB, Grain-128AEAD, ISAP, PHOTON-Beetle, Romulus, SPARKLE, TinyJAMBU, and Xoodyak.

In Kazakhstan, in the field of information security, foreign hardware and software are mainly used, which are transparent to their developers. Therefore, the development of domestic cryptographic information protection facilities (CIPFs), including home-grown lightweight encryption algorithms based on previously developed encryption algorithms, is currently a pressing issue.

* **Corresponding author: Nursulu Kapalova,** Information security laboratory, Institute Information and Computational Technologies, Almaty, Kazakhstan, e-mail: kapalova@ipic.kz
Saule Nyssanbayeva: Information security laboratory, Institute Information and Computational Technologies, Almaty, Kazakhstan, e-mail: sultasha1@mail.ru
Armiyanbek Haumen: Information security laboratory, Institute Information and Computational Technologies, Almaty, Kazakhstan, e-mail: haumen.armanbek@gmail.com
Olzhas Suleimenov: Information security laboratory, Institute Information and Computational Technologies, Almaty, Kazakhstan, e-mail: suleimenov97@gmail.com

The Laboratory of Information Security of the Institute of Information and Computational Technologies (IICT) of the Committee of Science (CS) of the Ministry of Education and Science (MES) of the Republic of Kazakhstan (RK) conducts research scientific work on the development and analysis of new encryption systems, electronic digital signature, generation of cryptographic keys, and authentication, as well as the creation of software CIPFs based on them. This article presents the results of creating a domestic lightweight encryption algorithm and studying its cryptographic properties.

2 Description of the algorithm

The developed LBC-3 algorithm is structurally different from existing lightweight algorithms such as PRESENT, although the architecture is similar. The proposed LBC-3 algorithm has its own substitution table (S-Box), and the R1 function has been added to improve diffusion, due to which high "avalanche effect" rates are achieved. It should also be noted that, unlike PRESENT, in LBC-3, round keys are generated in five rounds using the R2 function.

The LBC-3 algorithm is a symmetric block algorithm and has the following main characteristics:

- Block length is 64 bits,
- Master key length is 80 bits,
- Round key length is 64 bits, and
- Number of rounds is 20.

The structure of the algorithm is shown in Figure 1.

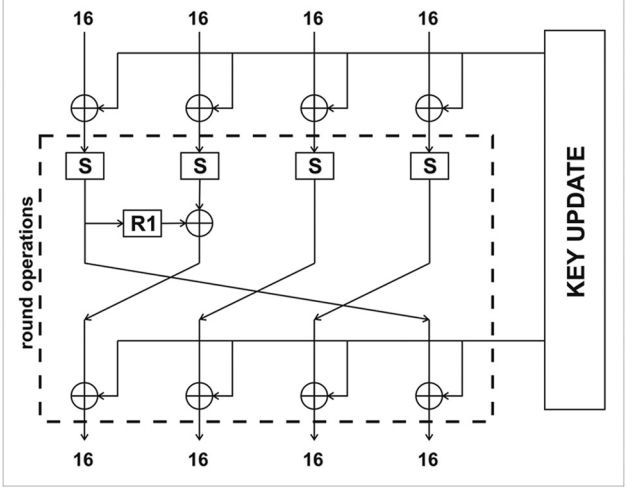

Figure 1: General scheme of the LBC-3 algorithm.

The encryption process consists of several rounds, each of which includes the following transformations:

- S transformation,
- R1 transformation,
- L transformation, and
- X transformation.

2.1 S Transformation

Every modern encryption algorithm uses a non-linear transformation in the form of a substitution table. It has been proven to be a strong cryptographic primitive against linear and differential cryptanalysis [4]. Lightweight block algorithms for non-linear transformations also use tabular substitutions – S-boxes. However, due to the requirement for lightweight algorithms, S-boxes should not take up a large amount of memory (ROM and RAM). To store a table that replaces 8-bit data fragments, 256 bytes are needed, and to store a 4-bit S-box, only 16 bytes are required [5]. Hence, 4-bit S-boxes are mainly used in lightweight cryptography. These include algorithms such as PRESENT and SERPENT. For the non-linear transformation of the LBC-3 algorithm, the 4-bit form of the substitution table was chosen. This S-box was constructed using a pseudo-random sequence generator (Table 1) and used to create a lightweight algorithm.

The non-linear transformation of the LBC-3 algorithm uses operations on half-bytes (4 bits). For each half-byte, a non-linear bijective substitution is applied, given by a one-dimensional array of 16 elements (4-bit S-box).

A 16-bit subblock is fed to the input of the transformation S and is divided into four groups of 4 bits. Each half-byte of the input block is an index of the value in the substitution table (Table 1). Thus, at the output of the transformation S, a set of half-bytes will be obtained, located at the corresponding indices in the given substitution table:

$$S(a) = S(a_0)||S(a_1)||S(a_2)||S(a_3), \quad S : V_4 \to V_4,$$

where a is 16 input bits and a_i are half-bytes of the subblock.

Table 1: S-Box

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A	2	7	E	C	6	F	1	0	B	5	3	9	4	8	D

2.2 R1 transformation

The linear transformation of the LBC-3 algorithm is a cyclic shift of bits and sub-blocks to a certain position. The input block of bits is divided into four subblocks of 16 bits each. The linear transformation at the subblock level is performed using the R1 function, which is applied only to the first subblock (Figure 1). The result of the R1 function is added to the second block modulo 2 (XOR operation). The function R1 has the following form:

$$\mathrm{R1}(a) = a \oplus (a \ll 7) \oplus (a \ll 10), \quad a \in F_2^{16},$$

where "$\ll$" is the operator of circular shifting the bits of the subblock to the left.

2.3 L Transformation (cyclic shift of subblocks)

This linear transformation is performed on the whole block. Here, the subblocks are rotated to the left by one position; i.e., the second subblock moves to the first position, the third subblock replaces the second block, the fourth subblock replaces the third one, and the first subblock moves to the fourth position (Figure 1).

2.4 X Transformation (adding round keys)

After rotating the subblocks, round keys modulo 2 are added. The round key, consisting of 64 bits, is divided into four subkeys of 16 bits each and added to the corresponding data subblocks.

2.5 Encryption process

Encryption by the LBC-3 algorithm begins with "whitening" the source text; that is, "zero keys" are added to the source text before the round transformation. The initial 64 bits of the master key are taken as "zero keys." Next, round transformations are performed.

If we designate the round encryption process as E, we get:

$$E(A) = [X(L(S(a_0)||R1(S(a_0)) \oplus S(a_1)||S(a_2)||S(a_3)))]_{rc} = B,\ rc = 1,\overline{20},$$

where $A = \{a_0, a_1, a_2, a_3\}$ is the 64-bit source text, $\{a_i\}$ are 16-bit subblocks. B is the 64-bit ciphertext, rc is the number of rounds, and $\oplus$ means modulo 2 addition.

2.6 Decryption process

The decryption process is performed in reverse order. Here, instead of the transformations S and L, the inverse transformations S^{-1} and L^{-1} are used, and the transformations R1 and X remain the same as in the encryption process (Table 2):

$$D(B) = [S^{-1}(L^{-1}(X(b_i)))]_{rc} = A, \quad rc = 1,\overline{20},$$

where $\{b_i\}$ are ciphertext subblocks. Function R1 is executed only for subblock $\{b_1\}$. That is, $\{b_2\} = \mathrm{R1}(b_1) \oplus b_2$.

2.7 Round key generation

The LBC-3 algorithm has a master key of 80 bits. From this sequence of bits, the round keys of the algorithm are generated. The key generation process consists of several transformations. The scheme for generating round keys is shown in Figure 2.

Table 2: Reverse S-box

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	7	1	B	D	A	5	2	E	C	0	9	4	F	3	6

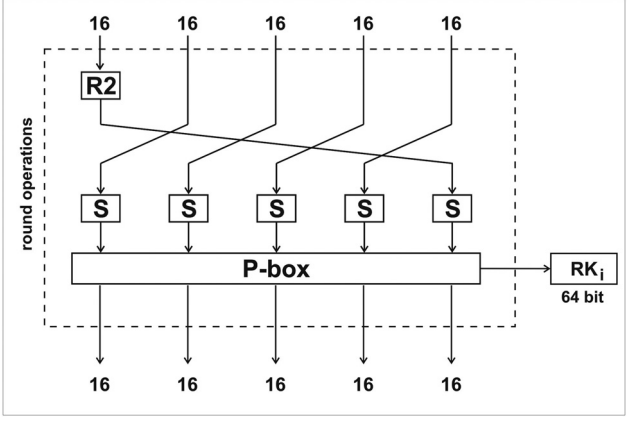

Figure 2: Scheme for generating round keys of the LBC-3 algorithm.

The master key of 80 bits is divided into five subblocks of 16 bits each. The key generation process itself has internal transformation rounds. The R2 function is applied to the first subblock and has the following form:

$$R2(a) = a \oplus (a \ll 6) \oplus (a \ll 13), \quad a \in F_2^{16},$$

where "≪" is the subblock left-shift bit operator. Then, the sub-blocks are rotated to the left by one position (Figure 2). Each subblock is then replaced via a 4-bit S-box with new values. Here, the S-box from the main encryption algorithm is used (Table 1). The resulting new bit values are combined into one sequence, forming 80 bits. These bits are grouped by 4 bits, forming 20 groups. These groups of bits are permutated according to Table 3 (P-Box).

In the table, row A is the initial position of 4-bit groups, row B is the new position of these bits in the sequence. Thus, the earlier transformations (function R2, rotation, S-box, P-box) are repeated 5 times (rounds). After five rounds of 80 bits, the initial 64 bits (on the left side of the sequence) are taken and used as round encryption (decryption) keys. The 80 bits obtained from the previous transformations will be used as the basis for the next round of keys. This process continues until all round encryption keys are obtained.

3 Study of the cryptographic properties of the algorithm

First, such a cryptographic property of the LBC-3 algorithm as the "avalanche effect" was investigated. To check for a good avalanche effect in a particular algorithm, an avalanche criterion is used. A cryptographic algorithm satisfies the avalanche criterion if a change in one bit of the input sequence changes, on average, half of the output bits.

Using the developed computer program that implements the avalanche criterion, the LBC-3 algorithm was investigated for the presence of the avalanche effect.

To characterize the degree of the avalanche effect in the transformation, an avalanche parameter is determined and used – the numerical value of the deviation of the probability of changing bits in the output sequence when bits in the input sequence change from the required probability value equal to 0.5.

For the avalanche criterion, the value of the avalanche parameter is determined by the formula

$$\varepsilon_i = |2k_i - 1|,$$

where i is the number of the bit to be changed at the input and k_i is the probability of changing half of the bits in the output sequence when the ith bit in the input sequence is changed.

The values of the specified avalanche parameter are in the range from 0 to 1 inclusive. In this case, the smaller the value of the avalanche parameter, the stronger the avalanche effect in the transformation [6].

The results of the study are shown in Tables 4–6.

The following table shows the number of output bits changed in each round, given that the first bit of the source text is changed.

During the study of the "avalanche effect" property, all values of the avalanche parameter were calculated when changing each bit of the source text for each round. The calculation results are shown in Table 5.

According to the results of the study, the best indicator of the avalanche effect is achieved after the sixth round of encryption (Figure 3). Based on this result, it can be assumed that 20-round encryption will provide high security of the ciphertext.

Test examples of the avalanche effect of the algorithm are listed in Table 6.

4 Statistical analysis of the algorithm

For statistical analysis, 20 files of various types and sizes were selected, containing from 20 to 1,000 kb of information. Data on the selected files are presented in Table 7. Testing was carried out using the software package "Automated system for the selection of statistical tests by D. Knuth and graphic tests," which implements a set of statistical tests.

After converting each file using five different keys, the corresponding encrypted files were obtained. The

Table 3: P-Box

A	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
B	5	12	18	16	10	13	20	2	15	4	17	7	19	3	6	9	1	8	11	14

Table 4: Results of the study of the avalanche effect

Round number	Number of bits changed	Avalanche effect	Avalanche parameter value $\varepsilon_i \approx$	Round number	Number of bits changed	Avalanche effect	Avalanche parameter value $\varepsilon_i \approx$
1	10	0.1563	0.6874	11	32	0.5	0
2	19	0.2969	0.4062	12	34	0.5313	0.0626
3	30	0.4688	0.0624	13	26	0.4063	0.1874
4	29	0.4531	0.0938	14	35	0.5469	0.0938
5	34	0.5313	0.0626	15	36	0.5625	0.125
6	29	0.4531	0.0938	16	30	0.4688	0.0624
7	40	0.625	0.25	17	35	0.5469	0.0938
8	33	0.5156	0.0312	18	31	0.4844	0.0312
9	34	0.5313	0.0626	19	34	0.5313	0.0626
10	34	0.5313	0.0626	20	33	0.5156	0.0312

Table 5: Results of the study of the avalanche effect by rounds

Rounds	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Average value of the avalanche parameter	0.8837	0.7671	0.5805	0.3877	0.2031	0.1479	0.0952	0.0849	0.0947	0.0991
Rounds	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Average value of the avalanche parameter	0.1147	0.0879	0.0928	0.0937	0.103	0.0879	0.0845	0.0869	0.0962	0.083

Table 6: Results of the avalanche effect

Source text	Ciphertext	Avalanche effect
11 11 11 11 11 11 11 11	E3 0D DC 82 CB 19 35 B0	0.4375 (28)
11 11 11 11 11 11 11 10	EC CF 52 D1 D8 0A 5B F2	
11 22 33 44 55 66 77 88	0B 16 94 70 8A E4 0D 56	0.4063 (26)
11 22 33 44 55 66 77 87	49 44 89 00 42 61 DB 67	
00 00 00 00 00 00 00 00	07 77 C8 DD 8F FD 55 A6	0.4844 (31)
10 00 00 00 00 00 00 00	B2 96 C8 6F 78 93 FD E5	

resulting 100 files were subjected to graphical and evaluation statistical tests. In graphical tests, the statistical properties of ciphertexts are displayed in the form of graphical dependencies, and in evaluation tests, the statistical properties are determined by numerical characteristics. As a result, according to the relevant data, a conclusion is made about the success of the passed test. Figures 4 and 5 show data on the number of files that successfully passed the graphical and evaluation tests.

In practice, the following approaches to testing lightweight encryption algorithms in software implementation are used:

1. Microcontroller Benchmarking by NIST LWC team (https://github.com/usnistgov/Lightweight-Cryptography-Benchmarking).
2. AVR/ARM Microcontroller Benchmarking by Rhys Weatherley (https://rweather.github.io/lightweight-crypto).
3. AVR/ARM/RISC-V Microcontroller Benchmarking by Sebastian Renner, Enrico Pozzobon, and Jürgen Mottok (https://lwc.las3.de).
4. RISC-V Benchmarking by Fabio Campos, Lars Jellema, Mauk Lemmen, Lars Müller, Daan Sprenkels, and Benoit Viguier (https://github.com/AsmOptC-RiscV/Assembly-Optimized-C-RiscV).

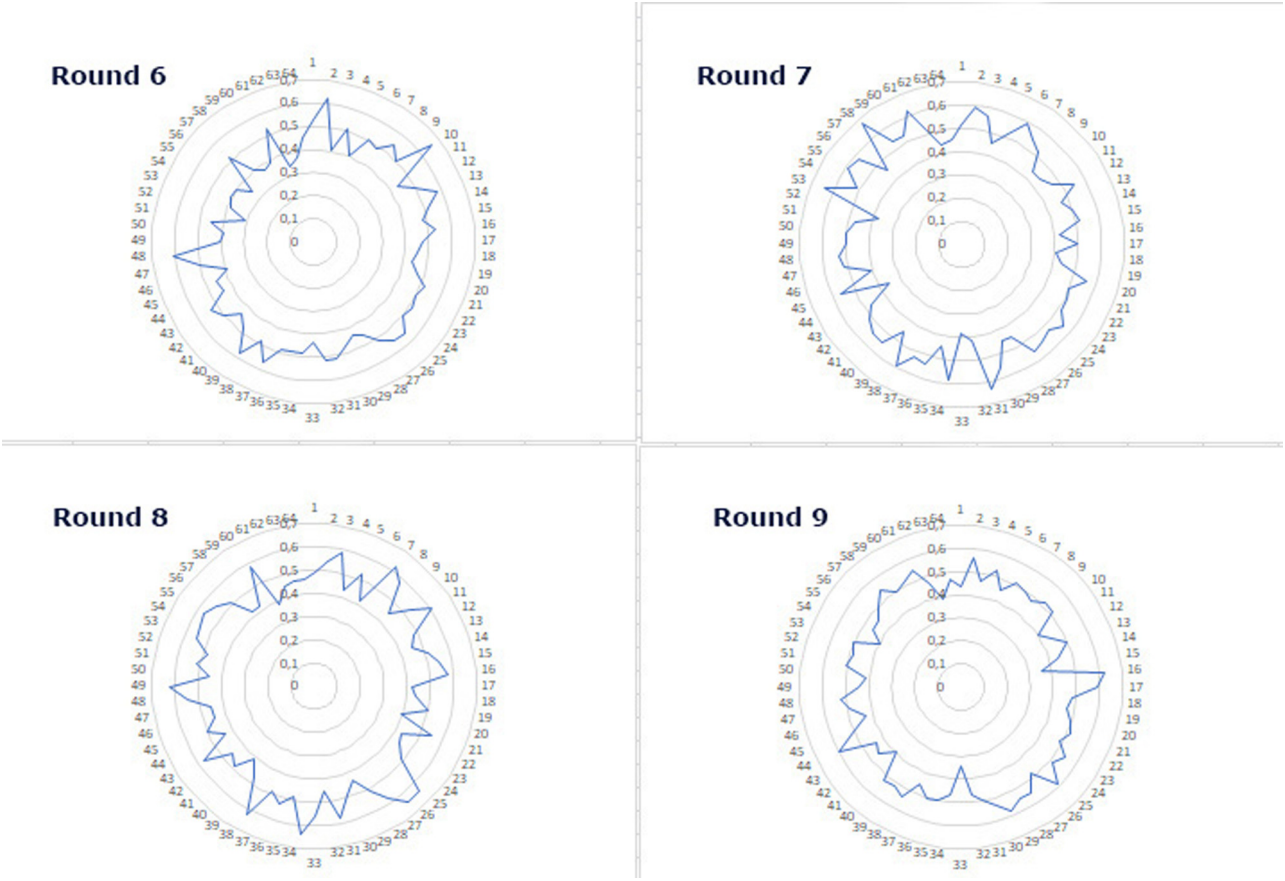

Figure 3: Histogram of avalanche parameter values in rounds 6–9.

Table 7: Plaintext files used in testing the LBC-3 algorithm

File number	File type	File format
1–5	*.docx	Microsoft Word document
6–10	*.xls	Microsoft Excel document
11–15	*.pptx	Microsoft PowerPoint document
16–20	*.pdf	Cross-platform open format
21–25	*.rar	Archived RAR document
26–30	*.zip	Archived ZIP document
31–35	*.jpg	Graphic document in raster format
36–40	*.png	Graphic document in raster format
41–45	*.txt	Text file
46–50	*.gif	Graphic document in raster format
51–55	*.html	Web document
56–60	*.cat	System file for merging files
61–65	*.mp4	Audio file
66–70	*.wmz	Vector Image Media file
71–75	*.dll	Dynamic Link Library file
76–80	*.log	Event log file
80–85	*.lex	Adobe Linguistic Library Data File
86–90	*.djvu	Graphic and text format document
91–95	*.xml	Web document
96–100	*.mp3	Audio file

5. eBACS (ECRYPT Benchmarking of Cryptographic Systems): General-purpose Processor (Intel, AMD, ARM Cortex-A, Qualcomm) Benchmarking (http://bench.cr.yp.to).

Of all the approaches presented earlier, for testing the developed and other lightweight encryption algorithms, the second option was chosen, i.e., AVR/ARM Microcontroller Benchmarking by Rhys Weatherley.

The developed lightweight encryption algorithm was tested on the ArduinoUnoR3 board.

Main features of ArduinoUnoR3:

1. Microcontroller – ATmega328;
2. Clock frequency – 16 MHz;
3. Operating voltage – 5 V; and
4. Flash-memory – 32 MB.

The free software Arduino IDE version 2.0.0-rc3 was used as a software shell for compiling and uploading the source code of the developed encryption algorithm to the ArduinoUnoR3 board.

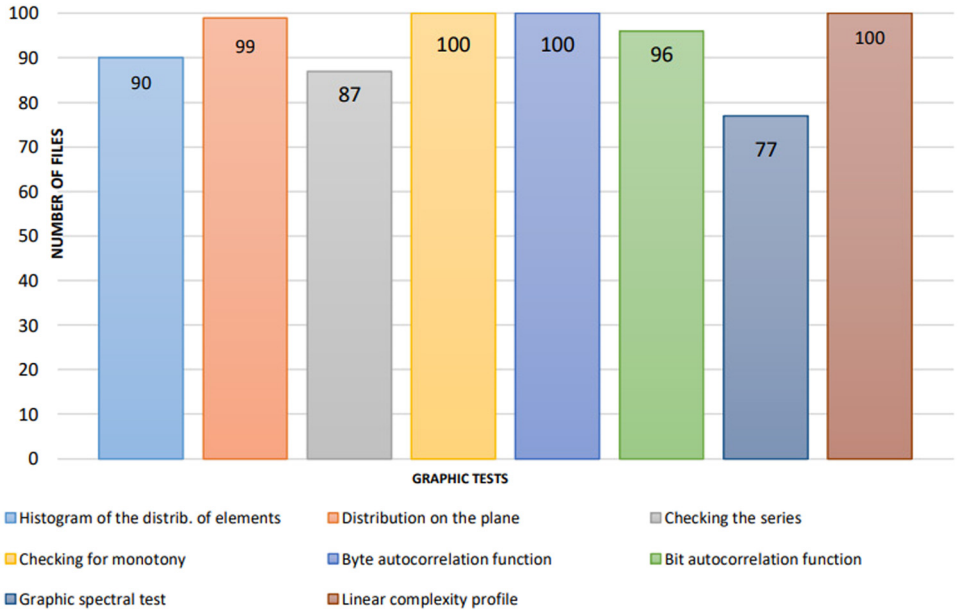

Figure 4: Graphical test results.

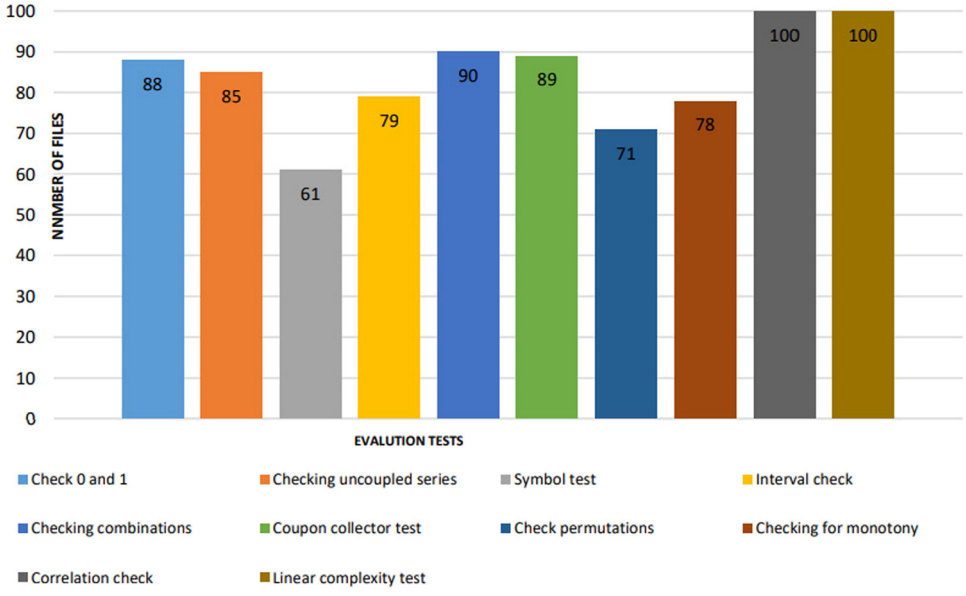

Figure 5: Evaluation test results.

ISL_LBC was developed in the high-level C++ programming language, which allowed us to slightly modify the source code for use in the ArduinoIDE. All necessary source codes and sketches for the study were taken from the corresponding website [7].

Theoretical and experimental tests have shown that the algorithm fully complies with the basic cryptographic requirements. The study of the cryptographic strength of the encryption algorithm will be continued in subsequent works.

5 Conclusion

Based on the tests and studies carried out, it has been established that the proposed LBC-3 encryption algorithm provides a good avalanche effect. According to the results of randomness testing using statistical tests, it was found that the binary sequence obtained after encryption is close to random.

Further work is to investigate other cryptographic properties of this algorithm using various cryptanalysis

methods. This lightweight cipher will be implemented in software on different platforms to analyze and evaluate its reliability. Recommendations and calculations for its hardware implementation will also be prepared.

Acknowledgment: The research work was carried out within the framework of the project AP09259570 "Development and study of a domestic lightweight encryption algorithm with limited resources" at the IICT of the RK MES CS.

Conflict of interest: Authors state no conflict of interest.

References

[1] Zhukov AE. Lightweight cryptography. Part 1. Cybersecurity issues, NPO Echelon, Moskow. Vol. 1, 2015. p. 26–43 (in Russian).
[2] Zhukov AE. Lightweight cryptography. Part 2. Cybersecurity issues, NPO Echelon, Moskow. Vol. 2, 2015. p. 2–10 (in Russian).
[3] Kapalova NA, Haumen A, Suleimenov OT. Lightweight cryptographic information protection systems. Proceedings of the International Scientific-Practical Conference Actual Problems of Information Security in Kazakhstan. Almaty; 2021. p. 48–53 (in Russian).
[4] Kazlauskas K, Kazlauskas J. Key-dependent S-box generation in AES block cipher system. Informatica. 2009;20:23–34.
[5] Preneel B. Perspectives on lightweight cryptography. Inscript: Shanghai, China; October 2010. p. 20–4.
[6] Kapalova NA, Haumen A, Sakan K. Diffusion properties of linear transformations. Proceedings of the Scientific Conference RK MES IICT Modern Problems of Informatics and Computing Technologies. Almaty; 2020. p. 189–93 (in Russian).
[7] https://github.com/rweather/arduinolibs.