



---

11<sup>th</sup> Workshop on  
Current Trends in Cryptology  
(CTCrypt 2022)



June 6-9, 2022, Novosibirsk, Russia

---

Pre-proceedings

**CTCrypt 2022 is organized by**

- Academy of Cryptography of the Russian Federation
- Steklov Mathematical Institute of Russian Academy of Science
- Technical Committee for Standardization «Cryptography and security mechanisms» (TC 026)

## Steering Committee

### Co-chairs

- Aleksandr Shoitov – Academy of Cryptography of the Russian Federation,  
Russia
- Vladimir Sachkov – Academy of Cryptography of the Russian Federation,  
Russia
- Igor Kachalin – TC 026, Russia

### Steering Committee Members

- Andrey Zubkov – Steklov Mathematical Institute of RAS, Russia
- Dmitry Matyukhin – TC 026, Russia  
Federal Educational and Methodical Association in
- Andrey Pichkur – System of Higher Education on Information Security,  
Russia

## Program Committee

### Co-chairs

- Alexander Lapshin – Academy of Cryptography of the Russian Federation, Russia
- Dmitry Matyukhin – TC 026, Russia
- Andrey Zubkov – Steklov Mathematical Institute of RAS, Russia

### Program Committee Members

- Sergey Aleshnikov – Immanuel Kant Baltic Federal University, Russia
- Alexey Alexandrov – Vladimir State University named after Alexander and Nikolay Stoletovs, Russia
- Sergey Checheta – Federal Educational and Methodical Association in System of Higher Education on Information Security, Russia
- Ivan Chizhov – Lomonosov Moscow State University, Russia
- Vladimir Fomichev – "Security Code", LLC, Russia
- Yury Kharin – Research Institute for Applied Problems of Mathematics and Informatics, Belarus
- Grigory Marshalko – TC 026, Russia
- Eduard Primenko – Lomonosov Moscow State University, Russia
- Boris Ryabko – Federal Research Center for Information and Computational Technologies; and Novosibirsk State University, Russia
- Vasily Shishkin – "NPK Kryptonite", JSC, Russia
- Stanislav Smyshlyaev – "Crypto-Pro", LLC, Russia
- Alexey Tarasov – Federal Educational and Methodical Association in System of Higher Education on Information Security, Russia
- Natalia Tokareva – Sobolev Institute of Mathematics SB RAS, Russia
- Andrey Trishin – "Certification Research Center", LLC, Russia
- Alexey Urivskiy – "InfoTeCS", JSC, Russia
- Amr Youssef – Concordia University, Canada
- Andrey Zyazin – Russian Technological University (MIREA), Russia

## INVITED TALKS

# The International Olympiad in Cryptography “Non-Stop University CRYPTO”

Anastasiya Gorodilova<sup>1</sup> and Natalia Tokareva<sup>1,2</sup>

<sup>1</sup>Sobolev Institute of Mathematics, Novosibirsk, Russia

<sup>2</sup>Novosibirsk State University, Novosibirsk, Russia

gorodilova@math.nsc.ru, tokareva@math.nsc.ru

## Abstract

We would like to attract your attention to the annual international event in theoretical cryptography that was invented and organized in Russia. Non-Stop University CRYPTO is the International Olympiad in Cryptography that was held for the eight times in 2021. Its aim is to involve young researchers in solving curious and tough scientific problems of modern cryptography. In this paper, particular problems and their solutions of the Olympiad'2021 are discussed. They are relevant to ciphers, quantum circuits, historical ciphers, electronic voting, masking, implementation on a chip, etc. We consider open problems stated during the Olympiad history and discuss new results obtained. The 2021 year open problems concerned quantum error correction and s-Boolean sharing of a function are discussed in details.

**Keywords:** cryptography, ciphers, masking, quantum error correction, electronic voting, s-Boolean sharing, orthogonal arrays, Olympiad, NSUCRYPTO.

## 1 Introduction

Non-Stop University CRYPTO (NSUCRYPTO) is the unique international cryptographic Olympiad in the world [18]. It contains scientific mathematical problems for professionals, school and university students. Its aim is to involve young researchers in solving curious and tough scientific problems of modern cryptography. From the very beginning, the concept of the Olympiad was not to focus on solving olympic tasks but on including unsolved research problems at the intersection of mathematics and cryptography. Everybody can participate the Olympiad as far as it holds via the Internet.

The history of Non-Stop University CRYPTO started in 2014. We were inspired by an experience of the Russian Olympiad in Mathematics and Cryptography for school-students and decided to organize an International event with real scientific content for students and professionals. Since then eight

Olympiads were held and more than 3000 students and specialists from 68 countries took part in it. Program committee consists of 31 members from cryptographic groups all over the world. Between them are creators of several modern technologies and ciphers, like AES, Chaskey, etc. Main organizers are Cryptographic centre (Novosibirsk), Mathematical Center in Akademgorodok, Novosibirsk State University, Sobolev Institute of Mathematics, KU Leuven, Belarusian State University, Tomsk State University and Kovalevskaya North-West Centre of Mathematical Research.

The popularity of the Olympiad is growing. Thus, in the beginning of 2022 there is a significant number of publications about it. For instance, the Press Service of Saudi Arabia publishes the results of the NSUCRYPTO Olympiad on its website, and the well-known Al Jazeera media congratulates the bronze medalists from its pages. The official websites of the universities of London, Bombay, the Boyai Institute, the Vietnam Center for Scientific Research and other organizations introduce the winners from their countries. NSUCRYPTO is included into the list of International Olympiads, the number of prize-winning students at which affects the university’s entry into the RAEX-100 rating of the best universities in Russia.

In 2021, the Olympiad was dedicated to the 100th anniversary of the Cryptographic Service of Russian Federation. There were 746 participants from 33 countries. 19 problems were proposed to participants and 4 of them included open problems. The list of open problems for all the years is available on the web-cite of the Olympiad and it is still possible to solve them and receive special prizes from the Program committee.

According to the results of each Olympiad, scientific articles are published with an analysis of the solutions proposed to the participants, including unsolved ones, see [1, 2, 11, 12, 13, 14, 17].

In this work we would like to share several problems of the Olympiad 2021 and invite you to cooperate.

## 2 An overview of open problems

A specialty of the Olympiad is that unsolved problems at the intersection of mathematics and cryptography are formulated to the participants along with problems with known solutions. All the open problems stated during the Olympiad history as well as their current status can be found at the Olympiad website [19]. There are 26 open problems in this list.

We would like to say that the variety and difficulty of the problems are wide. In fact, we suggest problems that are of great interest to cryptography

over which many mathematicians are struggling in search of a solution. For example, these problems include “APN permutation” (2014), “Big Fermat numbers” (2016), “Boolean hidden shift and quantum computings” (2017), “Disjunct Matrices” (2018), and others. For instance, the problem “8-bit S-box” (2019) were inspired by [8].

Despite the fact that really hard problems can be found in the list of the Olympiad problems, participants are not afraid to take on such tasks. Indeed, some of the problems we suggested can be solved or partially solved even during the Olympiad. For example, three problems were solved completely: “Algebraic immunity” (2015), “Sylvester matrices” (2018), “Miller — Rabin revisited” (2020). Also, during the Olympiad partial solutions for four problems were suggested. These problems are “Curl27” (2019), “Bases” (2020), “Quantum error correction” (2021) and “s-Boolean sharing” (2021). The last two problems we discuss in details in sections 3.5 and 3.6.

Furthermore, what is important for us that some researchers are working on finding solutions after the Olympiad was over. In [16], a complete solution was found for the problem “Orthogonal arrays” (2018). The authors have shown that no orthogonal arrays  $OA(16\lambda, 11, 2, 4)$  exist with  $\lambda = 6$  and 7. Another problem, “A secret sharing” (2014) was partially solved in [9, 10], where particular cases were considered and was recursively solved in [3].

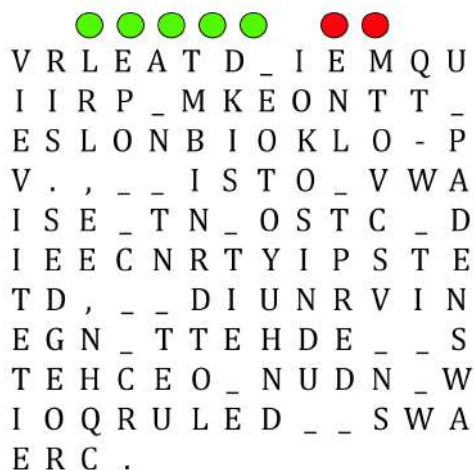
### 3 Particular problems of NSUCRYPTO’2021

We give here some problems from the Olympiad 2021 with solutions.

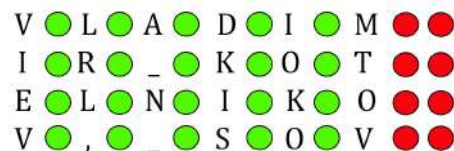
#### 3.1 Problem “Have a look and read!”

**Formulation.**

Read a secrete message.



**Solution.** This is a classical permutation cipher. The circles above the text are hints how to read a message.



The rest letters can be read the same way. The whole message is

*Vladimir Kotelnikov, Soviet scientist, invented the unique secret equipment SOBOL-P. It was not decrypted during the Second World War.*



### 3.2 Problem “A present for you!”

**Formulation.** Alice wants to implement the lightweight block cipher PRESENT on a chip. She starts with the bit permutation that is illustrated in Fig. 1 (we refer to [5] for the exact bit movement). Clearly, many lines are intersecting, and this would cause a short circuit if the lines were metal wires. Is it possible to avoid this problem by using several “layers,” i.e., parallel planes? That is to draw the lines without intersections on each layer. We assume that

- the work area is a rectangle bounded by the lines where input and output bits are placed and the lines of the outermost connections  $P(0) = 0$  and  $P(63) = 63$ ;
- input and output bits are ordered; connections are represented by arbitrary curves;
- color of a line indicates the number of its layer, a line can change color several times;
- the point where a line changes color indicates a connection from one layer to another.

- Q1** What is the minimum number of layers required for implementing in this way the PRESENT bit permutation?
- Q2** Find a systematic approach how to draw a valid solution for the minimum number of layers found in **Q1** and present the drawing!

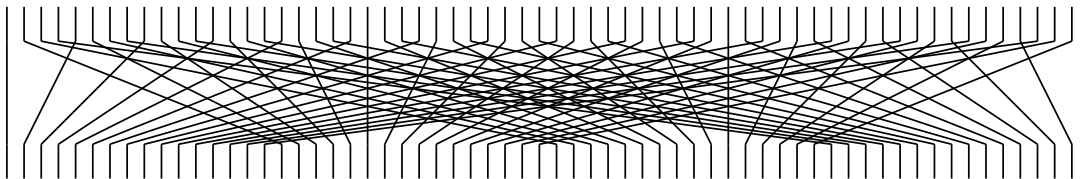
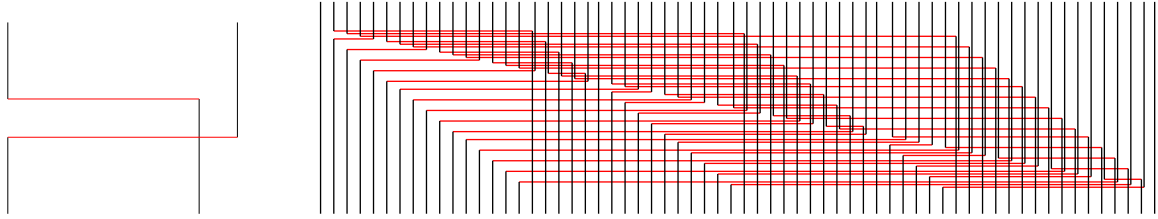


Figure 1: Illustration of the bit permutation used in PRESENT.

**Solution.** A first attempt to solve this problem, would be to try and connect some inputs and outputs. However, it will not take long to get stuck without a systematic approach.

An observation is that lines with several different angles create a problem, as it becomes difficult to predict where they might intersect with other lines. A way to overcome this is to work with only horizontal and vertical lines. The vertical lines can be in one color, and the horizontal lines in another color. This approach gives us an idea to use two layers. Let us show how to draw a scheme. All lines of the same color are parallel, however some lines might overlap. To see how to address this, consider the simple case of swapping two inputs, as shown in Fig. 2 (a). As the drawing shows, overlapping lines can be avoided by moving the second input slightly to the right. This is just done to make the drawing a bit easier; note that it does not affect the validity of the solution as the order of the inputs is preserved.

This method can be extended to an arbitrary number of inputs. A full solution for the PRESENT bit permutation given in Fig. 2 (b).



(a) Swapping two lines. (b) Illustration of the PRESENT permutation using two layers.

Figure 2: Illustrations to the solution of the problem “A present for you!”

### 3.3 Problem “Nonlinear hiding”

**Formulation.** Nicole is learning about secret sharing. She created a binary vector  $y \in \mathbb{F}_2^{6560}$  and splitted it into 20 shares  $x_i \in \mathbb{F}_2^{6560}$  (here  $\oplus$  denotes the bit-wise XOR):

$$y = x_1 \oplus x_2 \oplus \dots \oplus x_{20}.$$

Then, she created 20 more random vectors  $x_{21}, \dots, x_{40}$  and shuffled them together with the shares  $x_1, \dots, x_{20}$ . Formally, she chose a secret permutation  $\sigma$  of  $\{1, \dots, 40\}$  and computed

$$z_1 = x_{\sigma(1)}, \quad z_2 = x_{\sigma(2)}, \quad \dots \quad z_{40} = x_{\sigma(40)},$$

where each vector  $z_i \in \mathbb{F}_2^{6560}$ . Finally, she splitted each  $z_i$  into 5-bit blocks, and applied a secret bijective mapping  $\rho : \mathbb{F}_2^5 \rightarrow \mathcal{S}$ , where

$$\mathcal{S} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, y\}$$

(this strange alphabet has y instead of v).

Formally, she computed  $Z_i \in \mathcal{S}^{1312}$ ,  $1 \leq i \leq 40$  such that

$$Z_i = (\rho(z_{i,1\dots 5}), \rho(z_{i,6\dots 10}), \dots, \rho(z_{i,6556\dots 6560})).$$

After Nicole came back from school, she forgot all the details! She only has written all the  $Z_i$  and she also remembers the first 6432 bits of  $y$  (128 more are missing). The attachment [20] contains the 6432-bit prefix of  $y$  on the first line and  $Z_1, \dots, Z_{40} \in \mathcal{S}^{1312}$  on the following lines, one per line.

Help Nicole to recover full  $y$ !

**Solution.** This problem is inspired by the setting of generic white-box attacks [4]. Consider an obfuscated program, where a secret function is protected by a linear masking scheme (secret sharing), and the shares are scattered among fully random values. In addition, each value is protected by a

fixed random S-box (so called *encoding*). The goal of an adversary is to recover the full secret function from a partial knowledge of it on a few inputs, just by observing all the described values.

In the Olympiad’s problem, each row  $Z_i$  corresponds to a chosen share or a random value, and each column corresponds to a distinct “execution” (i.e., a recording of values on a distinct input of the program).

This problem can be solved by formulating the problem as a quadratic system of equations over  $\mathbb{F}_2$  and solving it through linearization. More precisely, introduce 40 variables  $t_i \in \mathbb{F}_2$ , one per each row  $i, 1 \leq i \leq 40$ , describing whether the  $i$ -th row is a secret share. In addition, introduce 32 variables  $m_c \in \mathbb{F}_2$ , one per each  $c \in \mathcal{S}$ , describing the first bit of  $\rho^{-1}(c)$ . Then, each known 5-bit chunk  $y_{5j+1\dots 5j+5}$  of  $y$  (more precisely, its first bit) gives a quadratic equation

$$\text{equation } j, 1 \leq j \leq 1286 : \quad \bigoplus_{1 \leq i \leq 40} t_i \cdot m_{Z_{i,j}} = y_{5j+1}.$$

This system can be linearized. More precisely, introduce a new variable  $w_{i,j} = t_i \cdot m_{Z_{i,j}} \in \mathbb{F}_2$  per each monomial  $t_i \cdot m_{Z_{i,j}}$ . There are  $40 \times 32 = 1280$  variables and  $6432/5 \geq 1286$  equations. After solving this linear system, we can see which rows  $Z_i$  correspond to the shares of  $y$  and a mapping defining first coordinate of  $\rho^{-1}$  (up to a constant), allowing to recover every 5-th bit of the missing part. Repeating this procedure for 4 other positions allows to fully recover the value (note that the values of  $t_i$  would already be recovered).

Also, there was a hidden text in the random beginning prefix of  $y$  dedicated to the 100th anniversary of the Cryptographic Service of Russian Federation:

*2021 marks the centenary of the cryptographic service in Russia!  
On May 5, 1921, the 8th special department was created. Its tasks included the study of theoretical problems of cryptography and the development of new ciphers, the organization of cipher communication, cryptanalysis, radio monitoring and radio interception, etc.*

### 3.4 Problem “Shuffle ballots”

**Formulation.** In electronic voting,  $n$  voters take part. Each of them is assigned a unique identifier that is a number from the set  $\{0, 1, \dots, n - 1\}$ . Shuffling of ballots during elections is implemented through the encryption of identifiers. When encrypting, the following conditions must hold:

1. The encryption result is again an integer from  $\{0, 1, \dots, n - 1\}$ .

2. The encryption process must involve the block cipher AES with a fixed key  $K$ .
3. The number of requests to  $\text{AES}_K$  must be the same for each identifier.
4. In order to manage security assurances, it should be possible to customize the number of requests to  $\text{AES}_K$ .

Suggest a way how to organize the required encryption process of identifiers for  $n = 5818342$  and  $n = 5818343$ . In other words, propose a method for organizing a bijective mapping from  $\{0, 1, \dots, n-1\}$  to itself that satisfies conditions described above.

**Solution. Case 1.** The number  $n = 5818342$  is composite. It is factored as a product of numbers close to each other, namely  $n_1 = 2594$  and  $n_2 = 2243$ . Hence, an identifier  $x \in \{0, 1, \dots, n-1\}$  can be uniquely represented as  $x = x_1 n_2 + x_2$ , where  $x_1 \in \{0, 1, \dots, n_1-1\}$  and  $x_2 \in \{0, 1, \dots, n_2-1\}$ .

We can encrypt identifiers by applying several rounds of the form:

$$(x_1, x_2) \leftarrow (y_1, (x_2 + \text{AES}_K(y_1 + \beta)) \bmod n_2), \quad y_1 = (x_1 + \text{AES}_K(x_2 + \alpha)) \bmod n_1.$$

Here  $\alpha, \beta$  are round constants. We process numbers with  $\text{AES}_K$  encoding them in 128-bit blocks before encryption and decoding back after.

The proposed construction follows the UNF (Unbalanced Number Feistel) scheme [15]. When  $n_1 \approx n_2$  (that is our case), at least 3 rounds should be used to ensure security. Generally speaking, security guarantees are strengthened with increasing the number of rounds.

**Case 2.** The number  $n = 5818343$  is prime. So, the UNF scheme cannot be directly applied. Nevertheless, we can reduce the problem to the UNF encryption for a composite modulus  $n' = n-1$  that was considered in Case 1 above. We act as follows:

1. A number  $a$  is chosen at random from the set  $\{0, 1, \dots, n-1\}$ .
2. Suppose we need to encrypt  $x \in \{0, 1, \dots, n-1\}$ . If  $x \neq a$ , then we determine

$$x' = \begin{cases} x, & x < a; \\ x-1, & x > a. \end{cases}$$

The number  $x'$  belongs to the set  $\{0, 1, \dots, n'-1\}$ . We encrypt  $x'$  using the UNF scheme with  $d$  rounds.

3. If  $x = a$ , then we assign to  $x$  the ciphertext  $n' = n-1$ . Additionally, to satisfy Requirement 3 for a constant number of requests to AES, we perform  $d$  dummy AES encryptions. Note that Requirement 3 is a countermeasure against timing attacks.

We would like to briefly present ideas proposed by the participants.

**The first idea.** The prime  $n$  is incremented rather than decremented. Using UNF, we construct a bijection  $E_K$  on  $\{0, 1, \dots, n\}$ . Then we encrypt  $x \neq n$  with  $E_K$  and get  $y \neq n$ . What should we do if  $E_K(x) = n$ ? There are 3 possibilities:

1. Precalculate  $x_0 = E_K^{-1}(n)$ . If  $x = x_0$ , then return  $E_K(n)$ . If  $x \neq x_0$ , then return  $E_K(x)$ .
2. Precalculate  $y_0 = E_K(n)$ . If  $y = E_K(x)$  is equal to  $n$ , then return  $y_0$ . Otherwise, return  $y$ .
3. Without precalculations. Calculate  $y = E_K(x)$  and  $z = E_K(y)$ . If  $y = n$ , then return  $z$ . Otherwise, return  $y$ .

**The second idea.** The encryption can be given by a permutation polynomial over the integer ring modulo  $n$ . For example,

$$f_K(x) = (\dots((x + k_1)^e + k_2)^e + \dots + k_{r-1})^e + k_r \pmod n.$$

Here  $k_1, k_2, \dots, k_r$  are round keys which are built using  $\text{AES}_K$  (for instance,  $k_i = \text{AES}_K(i) \pmod n$ ) and  $e$  is coprime with  $\varphi(n)$ . We are dealing with the composition of permutations  $x \mapsto x^e \pmod n$  and  $x \mapsto (x + 1) \pmod n$  which is itself a permutation.

### 3.5 Problem “Quantum error correction”

**Formulation.** The procedure of error correction is required for quantum computing due to intrinsic errors in quantum gates. One of approaches to quantum error correction is to encode quantum information in three-qubit states, i. e.  $\alpha_0 |0\rangle + \alpha_1 |1\rangle \rightarrow \alpha_0 |000\rangle + \alpha_1 |111\rangle$ .

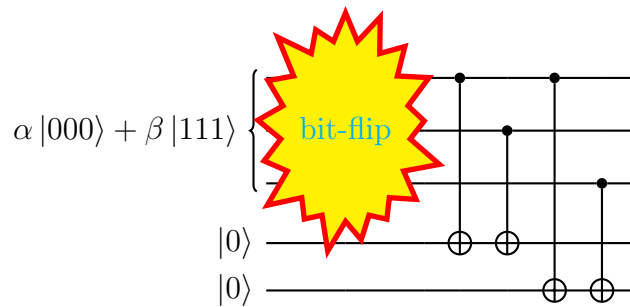
Below are problems for a special prize!

- Q1** Design a circuit which implements such encoding.
- Q2** Design a circuit which restores the initial state of the three-qubit system, if a single bit-flip error  $|0\rangle \leftrightarrow |1\rangle$  occurs in one of three qubits. Hint: use two additional qubits and three-qubit Toffoli gates.
- Q3** What will happen, if the quantum gates used for error correction are imperfect? What will be the threshold for gate fidelity, when the error correction will stop working?

**Solution. Q1.** The encoding can be described by the following circuit:

$$\left. \begin{array}{l} \alpha |0\rangle + \beta |1\rangle \\ |0\rangle \\ |0\rangle \end{array} \right\} \alpha |000\rangle + \beta |111\rangle$$

**Q2.** Let us firstly describe the authors’ solution. To find the bit-flip in each qubit, we introduce two ancillary qubits and entangle them with our three data qubits via CNOT gates:

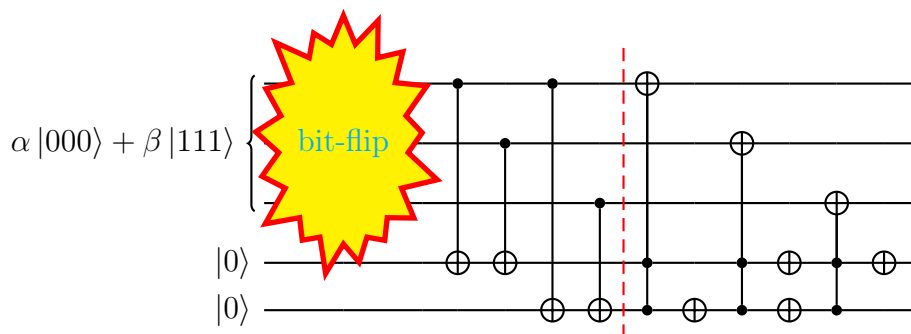


Without bit-flips in the data qubits, both ancillary qubits will stay in the state  $|00\rangle$ , because the states of data qubits are identical. It means that, depending on the initial state of the first qubit, the Pauli-X gate will be either never applied to the ancillary qubits, or applied twice.

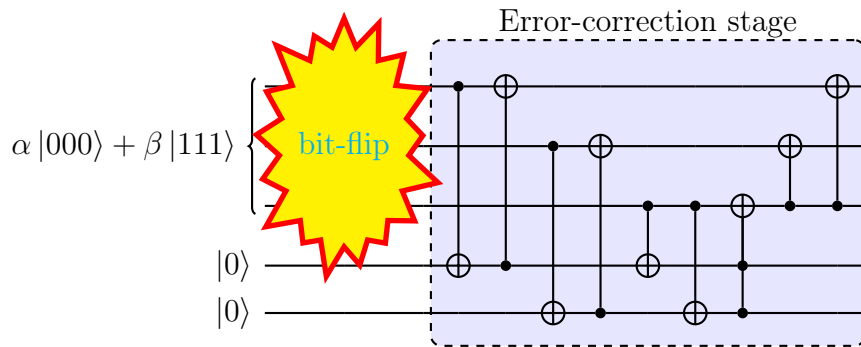
If there is a bit-flip in any of data qubits, the Pauli-X gate will be applied once or three times to one of the ancillary qubits. This will indicate the error in the particular data qubit:

- state  $|00\rangle$  means “no error”;
- state  $|11\rangle$  means “error in the 1st qubit”;
- state  $|10\rangle$  means “error in the 2nd qubit”;
- state  $|01\rangle$  means “error in the 3d qubit”.

Now it is possible to restore the initial state by applying Toffoli gates. For example, a Toffoli gate with two ancillary qubits used as control ones and first data qubit used as target ones will flip its state if the ancillary qubits are in state and leave it unchanged in any other case (no error in the first qubit). Similarly, the flips in other qubits can be restored. The final circuit is



During the Olympiad, twelve teams made progress in solving the problem and suggested good and correct schemes. We would like to mention the best one proposed by the team of Viet-Sang Nguyen, Nhat Linh Le Tan, Nhat Huyen Tran Ngoc (France, Paris). Taking into account discussions on **Q3** in the solution of this team, we mark this problem as “*partially solved*”. In their circuit, only one Toffoli gate is used:



**Q3.** Several participants proposed interesting ideas on this problem. In some of them, the minimum fidelities for a success probability were considered independently for every type of gates, i. e. Pauli-X, CNOT and Toffoli gate, and corresponding diagrams were shown. In another, it was assumed that the probability of imperfect operation of each gate is the same, then the threshold when error correction stops working was estimated.

There was an approach under assumption that the error-box makes a single bit-flip error and the error-correction box makes a mistake, both with some fixed probabilities, and the probability that the error-box makes multiple bit-flip errors is neglectable. It was obtained that the error-correction stops working when the probability of its proper is larger than  $1/2$ .

### 3.6 Problem “ $s$ -Boolean sharing”

**Formulation.** In cryptography, a field known as side-channel analysis uses extra information such as the power consumption of an implementation to break a cryptographic primitive. In order to defend against these attacks, one does not need to change the primitive but only the way the primitive is implemented. A popular countermeasure is called “sharing” where the computation of the primitive is split in multiple parts (this notion was firstly suggested in [6, 7]). Each part seemingly operates on random data such that an adversary has to observe all parts of the computation in order to gain sense of the secret information that was processed.

An  $s$ -Boolean sharing of a variable  $x \in \mathbb{F}_2$  is a vector  $(x_1, x_2, \dots, x_s) \in \mathbb{F}_2^s$  such that  $x = \bigoplus_{i=1}^s x_i$ . A vectorial Boolean function  $G : \mathbb{F}_2^{sn} \rightarrow \mathbb{F}_2^{sm}$  is

an  $s$ -Boolean sharing of a function  $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$  if for all  $x \in \mathbb{F}_2^n$  and  $(x_1, \dots, x_s) \in \mathbb{F}_2^{sn}$ ,  $x_i \in \mathbb{F}_2^n$ , such that  $\bigoplus_{i=1}^s x_i = x$ ,

$$\bigoplus_{i=1}^s G_i(x_1, \dots, x_s) = F(x).$$

Here,  $G = (G_1, \dots, G_s)$ ,  $G_i : \mathbb{F}_2^{sn} \rightarrow \mathbb{F}_2^m$  and “ $\oplus$ ” denotes the bit-wise XOR.

**Q1** Write an algorithm which takes in a vectorial Boolean function and an integer  $s$  and returns true/false on whether the function is a  $s$ -Boolean sharing of another function. In case the result is true, the algorithm also returns the function whose sharing is the algorithm’s input.

**Q2 Problem for a special prize!** Propose a theoretical solution to the problem of checking whether the function is a  $s$ -Boolean sharing of another function.

**Example.** If you give the Boolean function  $G : \mathbb{F}_2^6 \rightarrow \mathbb{F}_2^3$  such that

$$\begin{aligned} G_1(a, b, c, d, e, f) &= ad \oplus ae \oplus bd \\ G_2(a, b, c, d, e, f) &= be \oplus bf \oplus ce \\ G_3(a, b, c, d, e, f) &= cf \oplus cd \oplus af \end{aligned}$$

the algorithm should return true when  $s = 3$  together with the function  $F : \mathbb{F}_2^2 \rightarrow \mathbb{F}_2$  such that  $F(x, y) = xy$ , where  $x = a \oplus b \oplus c$  and  $y = d \oplus e \oplus f$ .

**Solution to Q1.** We will give a general approach. Consider a function  $G : \mathbb{F}_2^{sn} \rightarrow \mathbb{F}_2^{sm}$  of variables  $x_1, \dots, x_{sn}$ , we check whether it is an  $s$ -Boolean sharing of some function  $F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$ . Take an arbitrary permutation of the  $sn$  input bits  $\pi$ , there are a total of  $sn!$  of such permutations (we note that one can reduce this number as some permutations would lead to the same sharing). Denote  $\pi(x_1, \dots, x_{sn}) = (y_1, \dots, y_{sn})$  and  $z_i = (y_{(i-1)*n+1}, \dots, y_{i*n})$  for  $i \in \{1, \dots, s\}$ . We want to verify whether

$$\bigoplus_{i=1}^s G_i(z_1, \dots, z_s) = F\left(\bigoplus_{i=1}^s z_i\right),$$

for all  $(z_1, \dots, z_s) \in \mathbb{F}_2^{sn}$ . This is easily done via a brute force approach of going through all  $(z_1, \dots, z_s) \in \mathbb{F}_2^{sn}$  (this requires  $2^{sn}$  evaluations) and verifying the above equation. In case the equation does not hold, we go to the next permutation  $\pi$ . Otherwise, we stop searching and return true. The algorithm would require around  $sn! \cdot 2^{sn}$  steps.



**Ideas on Q2.** The most interesting idea found by the participants considers the algebraic normal form of the shared function. Let us consider an ordered case where it is known which inputs would form the shares of the function. Let  $F$  be an arbitrary Boolean function. In case  $F$  is the unshared function of some  $G$ , then

$$\bigoplus_{i=1}^s G_i(x_1, \dots, x_s) = F\left(\bigoplus_{i=1}^s x_i\right),$$

Notice that for each monomial  $x^1 \cdot \dots \cdot x^\ell$  in  $F$ , we get the shared monomial  $(\bigoplus_i x_i^1) \cdot \dots \cdot (\bigoplus_i x_i^\ell)$ . We then verify for each monomial in  $G$  whether the other shares of that monomial are also present. If so, we remove  $(\bigoplus_i x_i^1) \cdot \dots \cdot (\bigoplus_i x_i^\ell)$  and repeat until no more monomial are present in  $G$ .

The best solution found was given by the team of university students Gongyu Shi, Ruoyi Kong, Haoxiang Jin (China, Shanghai) and awarded a special prize for “partially solving” the problem.

**Acknowledgments.** The work was carried out within the framework of the state contract of the Sobolev Institute of Mathematics (project no. FWNF-2022-0018).

## References

- [1] Agievich S., Gorodilova A., Idrisova V., Kolomeec N., Shushuev G., Tokareva N., “Mathematical problems of the second international student’s Olympiad in cryptography”, *Cryptologia*, **41:6** (2017), 534–565.
- [2] Agievich S., Gorodilova A., Kolomeec N., Nikova S., Preneel B., Rijmen V., Shushuev G., Tokareva N., Vitkup V., “Problems, solutions and experience of the first international student’s Olympiad in cryptography”, *Prikladnaya Diskretnaya Matematika*, **3** (2015), 41–62.
- [3] Ayat S. M., Ghahramani M., “A recursive algorithm for solving "a secret sharing" problem”, *Cryptologia*, **43:6** (2019), 497–503.
- [4] Biryukov A., Udovenko A., *Attacks and Countermeasures for White-box Designs*, Cryptology ePrint Archive: Report 2018/049, <https://eprint.iacr.org/2018/049.pdf>.
- [5] Bogdanov A., Knudsen L. R., Leander G., Paar C., Poschmann A., Robshaw M. J. B., Seurin Y., Vikkelsoe C., “PRESENT: An Ultra-Lightweight Block Cipher”, *LNCS*, CHES 2007, **4727**, 2007, 450–466.
- [6] Chari S., Jutla C. S., Rao J. R., Rohatgi P., “Towards sound approaches to counteract power-analysis attacks”, *LNCS*, CRYPTO 1999, **1666**, 1999, 398–412.
- [7] Goubin L., Patarin J., “DES and Differential Power Analysis The "Duplication" Method”, *LNCS*, CHES 1999, **1717**, 1999, 158–172.
- [8] Fomin D. B., “New Classes of 8-bit Permutations Based on a Butterfly Structure”, *Matematicheskie Voprosy Kriptografii*, **10:2** (2019), 169–180.
- [9] Geut K., Kirienko K., Sadkov P., Taskin R., Titov S., “On explicit constructions for solving the problem “A secret sharing””, *Prikladnaya Diskretnaya Matematika. Prilozhenie*, **10** (2017), 68–70, In Russian.
- [10] Geut K. L., Titov S. S., “On the blocking of two-dimensional affine varieties”, *Prikladnaya Diskretnaya Matematika. Prilozhenie*, **12** (2019), 7–10, In Russian.

- [11] Gorodilova A., Agievich S., Carlet C., Gorkunov E., Idrisova V., Kolomeec N., Kutsenko A., Nikova S., Oblaukhov A., Picek S., Preneel B., Rijmen V., Tokareva N., “Problems and solutions of the Fourth International Students’ Olympiad in Cryptography (NSUCRYPTO)”, *Cryptologia*, **43**:2 (2019), 138–174.
- [12] Gorodilova A., Agievich S., Carlet C., Hou X., Idrisova V., Kolomeec N., Kutsenko A., Mariot L., Oblaukhov A., Picek S., Preneel B., Rosie R., Tokareva N., “The Fifth International Students’ Olympiad in Cryptography — NSUCRYPTO: problems and their solutions”, *Cryptologia*, **44**:3 (2020), 223–256.
- [13] Gorodilova A. A., Tokareva N. N., Agievich S. V., Carlet C., Idrisova V. A., Kalgin K. V., Kolegov D. N., Kutsenko A. V., Mouha N., Pudovkina M. A., Udovenko A. N., “The Seventh International Olympiad in Cryptography: problems and solutions”, *Siberian Electronic Mathematical Reports*, **18**:2 (2021), A4–A29.
- [14] Gorodilova A., Tokareva N., Agievich S., Carlet C., Gorkunov E., Idrisova V., Kolomeec N., Kutsenko A., Lebedev R., Nikova S., Oblaukhov A., Pankratova I., Pudovkina M., Rijmen V., Udovenko A., “On the Sixth International Olympiad in Cryptography NSUCRYPTO”, *Journal of Applied and Industrial Mathematics*, **14**:4 (2020), 623–647.
- [15] Hoang V. T., Rogaway P., *On generalized Feistel networks*, Cryptology ePrint Archive: Report 2010/301, <https://eprint.iacr.org/2010/301>.
- [16] Kiss R., Nagy G. P., “On the nonexistence of certain orthogonal arrays of strength four”, *Prikladnaya Diskretnaya Matematika*, **52** (2021), 65–68.
- [17] Tokareva N., Gorodilova A., Agievich S., Idrisova V., Kolomeec N., Kutsenko A., Oblaukhov A., Shushuev G., “Mathematical methods in solutions of the problems from the Third International Students’ Olympiad in Cryptography”, *Prikladnaya Diskretnaya Matematika*, **40** (2018), 34–58.
- [18] <https://nsucrypto.nsu.ru/>.
- [19] <https://nsucrypto.nsu.ru/unsolved-problems/>.
- [20] <https://nsucrypto.nsu.ru/media/MediaFile/data-sharing.txt>.

# Cryptographic center (Novosibirsk): creation, research, perspectives

Natalia Tokareva<sup>1,2</sup>

<sup>1</sup>Sobolev Institute of Mathematics, Novosibirsk, Russia

<sup>2</sup>Novosibirsk State University, Novosibirsk, Russia

tokareva@math.nsc.ru

## Abstract

We would like to present our Cryptographic Center created in 2011 in Novosibirsk on the base of Sobolev Institute of Mathematics and Novosibirsk State University. We have a various scientific activity in discrete mathematics and cryptography. We study cryptographic Boolean functions (bent, APN, algebraic immune functions, etc.), cipher design, distinct aspects of cryptanalysis, blockchain technologies, post-quantum cryptosystems, SAT-solvers for cryptography. Important part of our activity is related to education: crypto courses in Novosibirsk State University, master programmes "Cryptography" and "Quantum technologies and cryptography". We are main organizers of the International Olympiad in Cryptography Non-Stop University CRYPTO (since 2014), Summer school in Cryptography and Information security (since 2019), International conference on cryptography SIBECRYPT together with Tomsk State University (since 2021). Perspectives and details of group organization are also in focus of our attention.

**Keywords:** Cryptographic Center, scientific activity.

# On the preliminary national standard "Information technology. Cryptographic data security. Terms and definitions"

Alexander Cheryomushkin, Alexander Gorin, and Andrey Zubkov

Academy of Cryptography of the Russian Federation, Moscow, Russia  
avc238@mail.ru

## Abstract

The purpose of the report is a general description of the development and content of the draft terminology standard in the field of cryptographic protection of information. The general design and structure of the standard will be characterized, its central concepts will be highlighted, and the features of the wording of the definitions of some terms will be noted. The emergence of the standard is caused by the need for widespread, uniform, proper and correct use of cryptographic concepts. The terms given in it are harmonized with international standards. The existing differences in the construction of English-language and Russian-language thermal systems in the field of cryptographic information protection are also emphasized.

**Keywords:** terminology standard, cryptographic terms and definitions.

## Contents

### SYMMETRIC CRYPTOGRAPHY

#### DESIGN

**sMGM: parameterizable AEAD-mode** 23

*Liliya Akhmetzyanova, Evgeny Alekseev, Alexandra Babueva, Andrey Bozhko and Stanislav Smyshlyayev*

**Keyed Streebog is a secure PRF and MAC** 41

*Vitaly Kiryukhin*

**Effective algorithm to compute guaranteed number of activations in XS-circuits and it's application to the cipher design** 66

*Denis Parfenov, Aleksandr Bakharev, Aleksandr Kutsenko, Aleksandr Belov, and Natalia Atutova*

**On some algebraic aspects of heuristic algorithms for constructing cryptographically strong S-Boxes** 87

*Alejandro Freyre Echevarría, Oliver Coy Puente, and Reynier A. de la Cruz Jiménez*

#### ANALYSIS

**Related-key attacks on the compression function of Streebog** 122

*Vitaly Kiryukhin*

**On differential characteristics modulo  $2^n$  of the composition of bitwise exclusive-or and a bit rotation** 142

*Nikolay Kolomeec, Ivan Sutormin, Denis Bykov, Matvey Panferov, and Tatiana Bonich*

### PROBABILISTIC ASPECTS

**Entropically secure cipher for messages generated by Markov chains with unknown statistics** 159

*Boris Ryabko*

**Two Lempel-Ziv goodness-of-fit criteria for nonequiprobable  
random binary sequences** 169

*Vasiliy Kruglov*

**Experimental study of NIST Statistical Test Suite ability to  
detect long repetitions in binary sequences** 184

*Andrei Zubkov and Aleksandr Serov*

## ALGEBRAIC ASPECTS

**On the question of nonlinearity of vectorial functions over finite  
fields** 193

*Vladimir Ryabov*

**On subfunctions of self-dual bent functions** 210

*Aleksandr Kutsenko*

**Proper families of functions and their applications** 240

*Alexei Galatenko, Valentin Nosov, Anton Pankratiev, and Kirill  
Tsaregorodtsev*

## PUBLIC-KEY CRYPTOGRAPHY

**On the (im)possibility of ElGamal blind signatures** 256

*Liliya Akhmetzyanova, Evgeny Alekseev, Alexandra Babueva, and  
Stanislav Smyshlyaev*

**Solving some cryptanalytic problems for lattice-based cryp-  
tosystems with quantum annealing method** 272

*Ivan Lysakov*

SYMMETRIC CRYPTOGRAPHY  
DESIGN

# sMGM: parameterizable AEAD-mode

Liliya Akhmetzyanova, Evgeny Alekseev, Alexandra Babueva,  
Andrey Bozhko and Stanislav Smyshlyaev

CryptoPro LLC, Moscow, Russia  
{lah, alekseev, babueva, bozhko, svv}@cryptopro.ru

## Abstract

The paper introduces a new AEAD-mode called sMGM (strong Multilinear Galois Mode). The proposed construction can be treated as an extension of the Russian standardized MGM mode and its modification MGM2 mode presented at the CTCTrypt’21 conference. The distinctive feature of the new mode is that it provides an interface allowing to choose specific security properties required for a certain application case. Namely, the mode has additional parameters allowing to switch on/off misuse-resistance or re-keying mechanisms.

The sMGM mode consists of two main «building blocks» that are a CTR-style gamma generation function with incorporated re-keying and a multilinear function that lies in the core of the original MGM mode. Different ways of using these functions leads to achieving different sets of security properties. Such an approach to constructing parameterizable AEAD-mode allows to reduce the code size that can be crucial for constrained devices.

We provide security bounds for the proposed mode. We focus on proving misuse-resistance of the sMGM mode, since the standard security properties were already analyzed during development of the original MGM and MGM2 modes.

**Keywords:** MGM, MGM2, AEAD mode, security notion, security bounds, nonce-misuse, misuse-resistant, SIV, re-keying

## 1 Introduction

In this paper we study nonce-based Authenticated Encryption with Associated Data (AEAD) schemes, which aim to provide both integrity and confidentiality of data. The widespread use of AEAD schemes motivates the study of its non-standard security properties, such as misuse-resistance [14], leakage resilience [4] and others [3, 8]. In our work we focus on misuse-resistance and “defence in depth”.

Commonly nonce-based AEAD schemes are analyzed in a setting where each new message is encrypted with a previously unused nonce (actually, nonce is a “number used only once”), and corresponding ciphertext has to



be indistinguishable from a random string. However, in some high-level applications nonce uniqueness requirement is hard to fulfill. For example, a nonce can be reused in FDE (Full Disk Encryption) schemes [9], in case of processes parallelization [5], or as a result of tamper attacks [14]. Hence, the need for misuse-resistant schemes arises. Misuse-resistance is formalized with the MRAE security notion [14], where ciphertext of each unique message (encrypted with even non-unique nonce) has to be indistinguishable from a random string.

There are several ways to construct a misuse-resistant mode. The first one is wide-PRP constructions with an AEZ mode [13] as an example. Another approach is a SIV (synthetic IV) construction combining arbitrary encryption and tag generation mechanisms in a certain way. The most vivid example of a SIV-based mode is GCM-SIV mode [12]. Both these approaches do not provide high efficiency and have a lack of exploitation properties that can be a deal for constrained devices. As a result, the crypto libraries should support various modes and its consumers should be competent enough to select the most efficient mode providing desired security properties. From that our aim is to construct a single mode that provides a user-friendly interface allowing consumers just to select the desired security properties, and then the mode would be automatically configured to the optimal way of data processing.

Additionally we are focusing on increasing the key lifetime which is a critical issue for most applications. This can be achieved by incorporating an internal re-keying technique from [2]. The internal re-keying approach modifies the base mode of operation in such a way that each message is processed starting from the same key, which is changed using the certain key update technique during processing of the current message. The string consisting of all input cipher blocks processed under the same key is called a section and the key is called a section key. We notice that the internal re-keying also allows to achieve better security against side channels attacks.

Inspired by ideas used to design the **MGM** [11, 15] and **MGM2** modes [1] and following the aim outlined in the previous paragraphs, we develop a new AEAD mode **sMGM** (strong **MGM**). By adjusting certain parameters this mode allows to 1) switch on/off misuse-resistance, which is achieved by applying the SIV construction, and 2) increase the key lifetime using internal re-keying. We design **sMGM** in such a way that it can be implemented as a single mode and its code size is almost the same as for the conventional modes.

Moreover, **sMGM** is built with a provable security in mind and we provide strict proofs for our security claims in Sections 5.2 and 5.3. In Section 2

we analyze the obtained bounds for several use cases and discuss **sMGM** design. We notice, that the presented proofs contain a new hybrid PRP/PRF switching technique for schemes with re-keying and a new security proof for CTR scheme with re-keying and a random IV in IND-CPA\$ [6] model.

## 2 Our contribution

In this section we discuss a new AEAD mode **sMGM**. The encryption and decryption algorithms of **sMGM** as well as their domain and range sets are formally defined in Section 4. The **sMGM** mode is parameterized by the following values:

$$\mathbf{sMGM} \left[ \begin{array}{cccc} E, & r, & l_0, l, & siv \\ \text{a block cipher} & \text{a nonce length} & \text{re-keying sections lengths} & \text{misuse-resistance on/off} \end{array} \right]$$

The first and foremost property of **sMGM** is an optional resistance to nonce misuse, which is achieved by applying SIV-like design [14]. Nonce misuse resistance can be switch on by setting a flag *siv* to 1. Further for simplicity, we will write **sMGMs**, when we need to address **sMGM** with *siv* = 1. In order to support both options and reduce the code size we define two “building blocks”, which are CTR-KM and MultTag functions. First one is a CTR-style gamma generation function with incorporated re-keying as in [2]. The second one is a multilinear function used for tag generation, which lies in the core of the original **MGM** mode [11]. These blocks are used in the Encrypt-then-Mac way, if *siv* = 0, and in Mac-then-Encrypt way (where tag is used as IV during encryption) if *siv* = 1. The approach is schematically depicted on Figure 1.

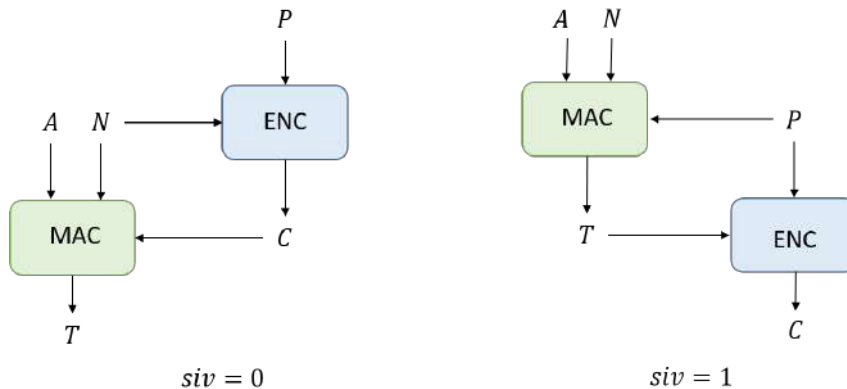


Figure 1: SIV approach

Moreover, **sMGM** is incorporated with a parameterizable internal re-keying. The major difference between the re-keying in **sMGM** and other re-

keying instantiations lies in a presence of a separate parameter  $l_0 \geq 0$  for the size of the initial section. The size of subsequent sections is defined by a parameter  $l > 0$ . The  $l_0$  parameter was introduced to control the main key lifetime, since in the **sMGM** mode the main key is used for processing data more frequently than the subsequent keys. For example, it can be set to 0 and the main key will be used only for generation of subsequent section keys. We notice, that by setting  $l_0$  to maximum data length, the re-keying can be switch off completely.

As a result, **sMGMs**, especially when combined with re-keying, provides a high security level in MRAE model (see Theorem 3) even if a single nonce is used in all queries. Moreover, **sMGMs** with re-keying has beyond birthday bounds in MRAE-int model (see Theorem 2). In this paper we focus on the security of the misuse resistant version of **sMGM**, since misuse resistance wasn't previously provided by **MGM**-like schemes. Security of another **sMGM** instance (with *siv* flag equal to 0) is somewhat similar to those of **MGM2** with re-keying and can be obtained by combining **MGM2** security proof from [1] and hybrid technique form **GCM-ACPKM** proof [2]. We also notice, that the integrity of non-SIV version of **sMGM** still holds in a nonce misuse setting.

Now we consider two instances of misuse resistant **sMGM** – with and without re-keying. We consider  $E_K$  to be a random permutation with  $n = 128$  and  $k = 256$ . The section sizes for the re-keyed instance are  $l_0 = 0$  and  $l = 2^6$ . In the Table 1 we provide security bounds for these two cases with a growing number  $q$  of encryption queries and a *single nonce* value used in all queries. The number of forgery attempts is fixed and equal to 1, the length  $m_P$  of plaintexts is bounded by  $2^{10}$  blocks or  $2^{14}$  bytes (which is the maximum size of TLS 1.3 records) and there is no additional data in all queries. In the table we denote by  $\delta_{priv}$  upper bounds for success probabilities of attack on privacy in MRAE model and by  $\delta_{int}$  of forgery in MRAE-int model.

$q$	non re-keyed sMGM		re-keyed sMGM	
	$\delta_{int}$	$\delta_{priv}$	$\delta_{int}$	$\delta_{priv}$
$2^{32}$	$2^{-62}$	$2^{-43}$	$2^{-62}$	$2^{-49}$
$2^{40}$	$2^{-46}$	$2^{-27}$	$2^{-46}$	$2^{-33}$
$2^{48}$	$2^{-30}$	$2^{-11}$	$2^{-30}$	$2^{-17}$
$2^{56}$	1	1	$2^{-14}$	1

Table 1: sMGMs security bounds

### 3 Definitions

Let  $|a|$  be the bit length of the string  $a \in \{0, 1\}^*$ . For a bit string  $a$  we denote by  $|a|_n = \lceil |a|/n \rceil$  the length of the string  $a$  in  $n$ -bit blocks. By  $\{0, 1\}^{\leq s}$  we denote the set of bit strings which length is less or equal to  $s$ .

For a string  $a \in \{0, 1\}^*$  and a positive integer  $l \leq |a|$  let  $\text{msb}_\ell(a)$  be the string, consisting of the leftmost  $l$  bits of  $a$ . For nonnegative integers  $l$  and  $i$  let  $\text{str}_l(i)$  be  $l$ -bit representation of  $i$  with the least significant bit on the right, let  $\text{int}(M)$  be an integer  $i$  such that  $\text{str}_\ell(i) = M$ . For bit strings  $a \in \{0, 1\}^n$  and  $b \in \{0, 1\}^n$  we denote by  $a \otimes b$  a string which is the result of their multiplication in  $GF(2^n)$  (here strings encode polynomials in the standard way). If the value  $s$  is chosen from a set  $S$  uniformly at random, then we denote  $s \stackrel{\mathcal{U}}{\leftarrow} S$ . We define a function  $\text{Set11}: \{0, 1\}^n \rightarrow \{0, 1\}^n$ ,  $\text{Set11}(x) = x$  or  $(110 \dots 0)$ .

For any set  $S$ , define  $\text{Perm}(S)$  as the set of all bijective mappings on  $S$  (permutations on  $S$ ), and  $\text{Func}(S)$  as the set of all mappings from  $S$  to  $S$ . A block cipher  $E$  with a block size  $n$  and a key size  $k$  is the permutation family  $(E_K \in \text{Perm}(\{0, 1\}^n) \mid K \in \{0, 1\}^k)$ , where  $K$  is a key.

### 4 sMGM mode

In this section we define a new AEAD mode – **sMGM**. The parameters of  $\text{sMGM}[E, r, l_0, l, \text{sig}]$  are defined in Section 2. For the nonce length the following limits should be observed:  $0 \leq r \leq n - 2 - \lceil \log_2(2 \lceil k/n \rceil) \rceil$ . The CTR-KM and MultTag functions are defined in Figure 2.

CTR-KM( $K, N, IV, f, len$ )	KM( $K, N$ )
$K_0 \leftarrow K$ $t \leftarrow \max(0, \lceil (len - l_0)/l \rceil)$ $st \leftarrow 1, end \leftarrow l_0$ <b>for</b> $j = 0 \dots t$ <b>do</b> : <b>for</b> $i = st \dots end$ <b>do</b> : $X_i \leftarrow E_{K_j}(0 \  f \  \text{str}_{n-2}(IV + i - 1))$ <b>if</b> $j \neq t$ : $K_{j+1} \leftarrow \text{KM}(K_j, N)$ $st \leftarrow end + 1$ $end \leftarrow \min(end + l, len)$ <b>return</b> $X_1, \dots, X_{len}$	$s \leftarrow \lceil k/n \rceil$ <b>for</b> $i = 1 \dots s$ <b>do</b> : $K^i \leftarrow E_K(10 \  \text{str}_{n-2}(N + i - 1))$ <b>return</b> $\text{msb}_k(K^1 \  K^2 \  \dots \  K^s)$
	MultTag( $K, \{H_1, \dots, H_{len}\}, M, len$ )
	$M_1 \  \dots \  M_{len} \leftarrow M$ $\tau \leftarrow \text{Set11} \left( \bigoplus_{i=1}^{len} M_i \otimes H_i \right)$ $T \leftarrow E_K(\tau)$ <b>return</b> $T$

Figure 2: CTR-KM and MultTag functions

The key, plaintext, associated data, ciphertext and tag sets for  $\text{sMGM}[E, r, l_0, l, siv]$  are defined as follows:  $\mathbf{K} = \{0, 1\}^k$ ,  $\mathbf{N} = \{0, 1\}^r$ ,  $\mathbf{A} = \mathbf{P} = \mathbf{C} = \{0, 1\}^{\leq n(2^{n-2}-2)}$ ,  $\mathbf{T} = \{0, 1\}^n$ . Moreover, the following condition should be satisfied:  $0 < |A| + |P| \leq n(2^{n-2} - 2)$ . The key generation function  $\text{sMGM.Gen}()$  is defined as  $K \xleftarrow{\mathcal{U}} \{0, 1\}^k$ , encryption and decryption algorithms are defined in Figures 3a and 3b respectively.

## 5 Security analysis

In this section we provide security analysis of misuse resistant  $\text{sMGM}$  instance (i.e.  $\text{sMGM}[E, r, l_0, l, 1]$ ) in the corresponding models. There are separate results for integrity formalized by MRAE-int model, and chosen ciphertexts confidentiality formalized by MRAE model.

We will denote by  $\text{Adv}_{\text{AEAD}}^{\text{MRAE-int}}(\mathcal{A})$  and  $\text{Adv}_{\text{AEAD}}^{\text{MRAE}}(\mathcal{A})$  the advantage of an adversary  $\mathcal{A}$  succeeding in breaking the properties of the AEAD mode in MRAE-int and MRAE models respectively. The advantage in the MRAE-int model is the probability that an adversary, which may repeat nonces in its queries, is able to forge a ciphertext that will be accepted as valid. The advantage in the MRAE model is the increase in the probability that an adversary, which may repeat nonces in its queries, is able to successfully distinguish an AEAD ciphertext from the output of an ideal cipher. In the MRAE model the adversary also has access to the *Decrypt* oracle, which in

sMGM[ $E, r, l_0, l, \mathbf{0}$ ].Enc( $K, N, A, P$ )	sMGM[ $E, r, l_0, l, \mathbf{1}$ ].Enc( $K, N, A, P$ )
$h \leftarrow  A _n, q \leftarrow  P _n, len \leftarrow h + q + 2, s \leftarrow \lceil k/n \rceil$ $N \leftarrow \text{int}(N \  0^{n-r-2})$	$h \leftarrow  A _n, q \leftarrow  P _n, len \leftarrow h + q + 2, s \leftarrow \lceil k/n \rceil$ $N \leftarrow \text{int}(N \  0^{n-r-2})$
.....Encryption.....	.....Padding.....
$\{\Gamma_1, \dots, \Gamma_q\} \leftarrow \text{CTR-KM}(K, N + s, N, 1, q)$ $C \leftarrow P \oplus \text{msb}_{ P }(\Gamma_1 \parallel \dots \parallel \Gamma_q)$	$a \leftarrow n A _n -  A , p \leftarrow n P _n -  P $ $M \leftarrow A \  0^a \  P \  0^p \  \text{str}_n( A ) \  \text{str}_n( P )$
.....Padding.....	.....Tag generation.....
$a \leftarrow n A _n -  A , c \leftarrow n C _n -  C $ $M \leftarrow A \  0^a \  C \  0^c \  \text{str}_n( A ) \  \text{str}_n( C )$	$\{H_1, \dots, H_{len}\} \leftarrow \text{CTR-KM}(K, N, N, 0, len)$ $T \leftarrow \text{MultTag}(K, \{H_1, \dots, H_{len}\}, M, len)$
.....Tag generation.....	.....Encryption.....
$\{H_1, \dots, H_{len}\} \leftarrow \text{CTR-KM}(K, N, N, 0, len)$ $T \leftarrow \text{MultTag}(K, \{H_1, \dots, H_{len}\}, M, len)$ <b>return</b> ( $C, T$ )	$IV \leftarrow \text{int}(\text{msb}_{n-2}(T))$ $\{\Gamma_1, \dots, \Gamma_q\} \leftarrow \text{CTR-KM}(K, N + s, IV, 1, q)$ $C \leftarrow P \oplus \text{msb}_{ P }(\Gamma_1 \parallel \dots \parallel \Gamma_q)$ <b>return</b> ( $C, T$ )

(a) sMGM.Enc algorithm

sMGM[ $E, r, l_0, l, \mathbf{0}$ ].Dec( $K, N, A, C, T$ )	sMGM[ $E, r, l_0, l, \mathbf{1}$ ].Dec( $K, N, A, C, T$ )
$h \leftarrow  A _n, q \leftarrow  C _n, len \leftarrow h + q + 2, s \leftarrow \lceil k/n \rceil$ $N \leftarrow \text{int}(N \  0^{n-r-2})$	$h \leftarrow  A _n, q \leftarrow  C _n, len \leftarrow h + q + 2, s \leftarrow \lceil k/n \rceil$ $N \leftarrow \text{int}(N \  0^{n-r-2})$
.....Padding.....	.....Decryption.....
$a \leftarrow n A _n -  A , c \leftarrow n C _n -  C $ $M \leftarrow A \  0^a \  C \  0^c \  \text{str}_n( A ) \  \text{str}_n( C )$	$IV \leftarrow \text{int}(\text{msb}_{n-2}(T))$ $\{\Gamma_1, \dots, \Gamma_q\} \leftarrow \text{CTR-KM}(K, N + s, IV, 1, q)$ $P \leftarrow C \oplus \text{msb}_{ C }(\Gamma_1 \parallel \dots \parallel \Gamma_q)$
.....Tag verification.....	.....Padding.....
$\{H_1, \dots, H_{len}\} \leftarrow \text{CTR-KM}(K, N, N, 0, len)$ $T' \leftarrow \text{MultTag}(K, \{H_1, \dots, H_{len}\}, M, len)$ <b>if</b> $T' \neq T$ : <b>return</b> $\perp$	$a \leftarrow n A _n -  A , p \leftarrow n P _n -  P $ $M \leftarrow A \  0^a \  P \  0^p \  \text{str}_n( A ) \  \text{str}_n( P )$
.....Decryption.....	.....Tag verification.....
$\{\Gamma_1, \dots, \Gamma_q\} \leftarrow \text{CTR-KM}(K, N + s, N, 1, q)$ $P \leftarrow C \oplus \text{msb}_{ C }(\Gamma_1 \parallel \dots \parallel \Gamma_q)$ <b>return</b> $P$	$\{H_1, \dots, H_{len}\} \leftarrow \text{CTR-KM}(K, N, N, 0, len)$ $T' \leftarrow \text{MultTag}(K, \{H_1, \dots, H_{len}\}, M, len)$ <b>if</b> $T' \neq T$ : <b>return</b> $\perp$ <b>return</b> $P$

(b) sMGM.Dec algorithm

Figure 3: sMGM mode

ideal world always return an error. These two models are formally defined in Appendix A.

Standard security notion for block ciphers are PRP-CPA («PseudoRandom Permutation under Chosen Plaintext Attack») and PRF («PseudoRandom Function») [6]. We will denote by  $\text{Adv}_E^{\text{PRP}}(\mathcal{A})$  and  $\text{Adv}_E^{\text{PRF}}(\mathcal{A})$  the advantage of an adversary  $\mathcal{A}$  succeeding in distinguishing  $E_K$  from a random permutation and a random function respectively.

## 5.1 Auxiliary results

In this section we introduce some auxiliary results, which will be used throughout subsequent proofs. We begin with Bernstein’s result for switching between random permutation and random function.

**Theorem 1** (Theorem 2.3 [7]). *For any distinguisher  $\mathcal{D}^f$  with an oracle  $f: \{0,1\}^n \rightarrow \{0,1\}^n$ , making at most  $q$  queries, the following inequality holds:*

$$\Pr[\mathcal{D}^\pi \rightarrow 1] \leq \Pr[\mathcal{D}^\rho \rightarrow 1] \cdot \left(1 - \frac{q-1}{2^n}\right)^{-q/2},$$

where  $\pi \stackrel{\mathcal{U}}{\leftarrow} \text{Perm}(n)$  and  $\rho \stackrel{\mathcal{U}}{\leftarrow} \text{Func}(n)$ .

Hereafter we will denote an expression  $\left(1 - \frac{q-1}{2^n}\right)^{-q/2}$  by  $B_q$ . The next statement will allow us to switch between a single random function and a set of independent random functions, when applying them to a number of non-overlapping subsets.

**Statement 1.** *For any finite set  $A$ , any integer  $k \leq |A|$ , any subsets  $A_1, \dots, A_k \subseteq A$ , such that  $A = A_1 \sqcup \dots \sqcup A_k$ ,  $A_i \cap A_j = \emptyset$  for  $i \neq j$ , and any distinguisher  $\mathcal{D}^f$  with an oracle  $f: A \rightarrow A$ , the following equality holds:*

$$\Pr[\mathcal{D}^\rho \rightarrow 1] = \Pr[\mathcal{D}^{\hat{\rho}} \rightarrow 1],$$

where  $\rho \stackrel{\mathcal{U}}{\leftarrow} \text{Func}(A)$  and  $\hat{\rho} = \{\rho_1, \dots, \rho_k\}$ ,  $\rho_i \stackrel{\mathcal{U}}{\leftarrow} \text{Func}(A)$ ,  $\hat{\rho}(a) = \rho_i(a)$  for  $a \in A_i$ .

## 5.2 MRAE integrity of sMGMs

**Theorem 2.** *For any MRAE-int-adversary  $\mathcal{A}$  for sMGMs, making at most  $q_E$  queries to the Encrypt oracle and at most  $q_D$  queries to the Decrypt oracle, where the block-length of associated data in each query is at most  $m_A$ , the block-length of plaintexts and ciphertexts in each query is at most*

$m_P$  and the number of distinct nonce values in all queries is at most  $q_N$ , there exist PRP-adversaries  $\mathcal{C}$  and  $\mathcal{C}_0$  for block cipher  $E$ , such that

$$\begin{aligned} \text{Adv}_{\text{sMGM}[E,r,l_0,l,1]}^{\text{MRAE-int}}(\mathcal{A}) \leq & \left( \frac{q(q-1)}{2^{n-1}} + \frac{q_D}{2^n} + q_N t_I \text{Adv}_E^{\text{PRP}}(\mathcal{C}) \right) \cdot B_{l+s}^{q_N t_I} \\ & \cdot B_{q(2l_0+2s+1)} + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_0), \end{aligned}$$

where  $q = q_E + q_D$ ,  $s = \lceil k/n \rceil$  and  $t_I = \lceil (m_A + m_P + 2 - l_0)/l \rceil$ . Adversary  $\mathcal{C}$  makes at most  $l + s$  queries to its oracle and  $\mathcal{C}_0$  makes at most  $q(2l_0 + 2s + 1)$  queries.

*Proof.* For processing the messages sMGMs uses the same block cipher with distinct key values: master key  $K$  and section keys  $K_i$  that depend on nonce values. We will consider block cipher with each distinct key as separate block cipher. Our foremost goal in the first part of the proof is to replace all block ciphers in sMGMs with random functions. This will allow us to apply Corollary 1 from [1] and obtain the bound. Recall, that we write sMGMs instead of  $\text{sMGM}[E, r, l_0, l, 1]$  for simplicity.

Now we proceed with the first step of the proof. At this step we replace block cipher with a master key  $K$  by random permutation  $\pi_0$ . Note that block cipher  $E_K$  is used for initial section processing, first re-keying mechanism and tag generation. We write  $\text{sMGMs}[E_K]$  to specify used block cipher. Let us consider experiments  $\text{Exp}_{\text{sMGMs}[E_K]}^{\text{MRAE-int}}$  and  $\text{Exp}_{\text{sMGMs}[\pi_0]}^{\text{MRAE-int}}$ . In a straightforward manner we construct such an adversary  $\mathcal{C}_0$ , that

$$\Pr \left[ \text{Exp}_{\text{sMGMs}[E_K]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \leq \Pr \left[ \text{Exp}_{\text{sMGMs}[\pi_0]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_0).$$

The adversary  $\mathcal{C}_0$  uses the adversary  $\mathcal{A}$  as a black box. It intercepts the queries of the adversary  $\mathcal{A}$  and process them by itself using its own oracle instead of calling  $E_K$  or  $\pi_0$ . Therefore, to simulate  $q$  queries  $\mathcal{C}_0$  makes at most  $q(2l_0 + 2s + 1)$  calls to its oracle, where  $2l_0$  term defines the number of processed blocks in the initial section during encryption and tag generation steps,  $2s$  term defines the number of processed blocks in the re-keying mechanism and  $+1$  arises from a call in a Tag generation process. The adversary  $\mathcal{C}_0$  outputs the same bit as the adversary  $\mathcal{A}$ .

The next step is to replace the random permutation  $\pi_0$  with a random function  $\rho_0$ . Applying Bernstein's result (Theorem 1), we have

$$\Pr \left[ \text{Exp}_{\text{sMGMs}[\pi_0]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \leq \Pr \left[ \text{Exp}_{\text{sMGMs}[\rho_0]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \cdot B_{q(2l_0+2s+1)}.$$

Since all inputs to this random function in the cases of 1) computing values  $H_j$  for initial section and computing first intermediate key in the tag



generation part; 2) computing values  $\Gamma_j$  for initial section and computing first intermediate key in the encryption part; 3) computing the tag are different (because of fixed bits in inputs), using one random function is indistinguishable from using three independent random functions  $\rho_I, \rho_C, \rho_t$  for these three cases due to Statement 1.

From this, our aim is to replace every block cipher in the tag generation part of sMGMs with a corresponding random function. We denote the keys appearing within the re-keying during processing the  $i$ -th,  $1 \leq i \leq q_N$ , adversarial query with a new nonce by  $K_{(i-1)t_I+1}, K_{(i-1)t_I+2}, \dots, K_{i \cdot t_I}$ , where  $t_I$  defines the maximum number of sections. Keys  $K_{(i-1)t_I+1}$ , are generated using random function  $\rho_I$  and, since  $\rho_I$  inputs are separated with fixed bits for  $H_j$  generation and for the re-keying processing, they can be considered random for every new nonce value (follows from Statement 1). Other keys  $K_{j+1}$  are generated as  $\text{KM}(K_j, N)$ . In a case, when a key is chosen randomly, we will write it with calligraphic font –  $\mathcal{K}_j$ . We will also write sMGMs $[\rho_I, E_{\mathcal{K}_1}, E_{\mathcal{K}_2}, \dots, E_{\mathcal{K}_{i \cdot t_I+1}}, E_{\mathcal{K}_{i \cdot t_I+2}}, \dots, E_{\mathcal{K}_{q_N \cdot t_I}}]$  to specify the block ciphers used in each integrity re-keying section in order of appearance (throughout all queries).

Now let us consider experiments  $\mathbf{Exp}_{\text{sMGMs}[\rho_I, \rho_1, \dots, \rho_{i-1}, E_{\mathcal{K}_i}, E_{\mathcal{K}_{i+1}}, \dots]}^{\text{MRAE-int}}$  and  $\mathbf{Exp}_{\text{sMGMs}[\rho_I, \rho_1, \dots, \rho_{i-1}, \pi_i, E_{\mathcal{K}_{i+1}}, \dots]}^{\text{MRAE-int}}$ . In a straightforward manner we construct such an adversary  $\mathcal{C}_i$ , that

$$\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_{i-1}, E_{\mathcal{K}_i}, \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \leq \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_{i-1}, \pi_i, \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_i).$$

The adversary  $\mathcal{C}_i$  uses the adversary  $\mathcal{A}$  as a black box. It intercepts the queries of the adversary  $\mathcal{A}$  and process them by itself using its own oracle instead of calling  $E_{\mathcal{K}_i}$  or  $\pi_i$ . Therefore,  $\mathcal{C}_i$  makes at most  $l + s$  calls to its oracle. It outputs the same bit as  $\mathcal{A}$ .

Next, we replace the random permutation with a random function, applying Bernstein's result (Theorem 1):

$$\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_{i-1}, \pi_i, E_{\mathcal{K}_{i+1}}, \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \leq \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_{i-1}, \rho_i, E_{\mathcal{K}_{i+1}}, \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \cdot B_{l+s},$$

where  $\rho_i$  is used both for  $H_j$  and  $K_{i+1}$  generation. However, since  $\rho_i$  inputs are separated with fixed bits for these two cases, we can claim, that the key  $K_{i+1}$  is generated randomly and independently from  $H_j$  (follows from Statement 1). Thus,

$$\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_i, E_{\mathcal{K}_{i+1}}, \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] = \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_i, E_{\mathcal{K}_{i+1}}, \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right].$$

Bringing all together, we have the following inequality:

$$\begin{aligned} \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_{i-1}, E_{\mathcal{K}_i}, E_{\mathcal{K}_{i+1}} \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] &\leq \\ &\leq \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\dots, \rho_{i-1}, \rho_i, E_{\mathcal{K}_{i+1}} \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \cdot B_{l+s} + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_i). \end{aligned}$$

Note that in the case, when  $\mathcal{K}_i$  is the key of last section, the same transition can be applied with small differences in justifications. The randomness of the next key (first intermediate key in the next query processing) is achieved earlier, since it is generated by  $\rho_I$  function.

Hence, starting from the experiment  $\mathbf{Exp}_{\text{sMGMs}[\rho_I, E_{\mathcal{K}_1}, \dots]}^{\text{MRAE-int}}$  and subsequently applying the described transition  $q_N \cdot t_I$  times, we obtain

$$\begin{aligned} &\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_I, E_{\mathcal{K}_1}, \dots]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \leq \\ &\leq \left( \dots \left( \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_I, \rho_1, \dots, \rho_{q_N \cdot t_I}]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \cdot B_{l+s} + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_{q_N \cdot t_I}) \right) B_{l+s} + \right. \\ &\quad \left. + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_{q_N \cdot t_I - 1}) B_{l+s} + \dots + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_2) B_{l+s} + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_1) = \right. \\ &= \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_I, \rho_1, \dots, \rho_{q_N \cdot t_I}]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \cdot B_{l+s}^{q_N t_I} + \sum_{i=1}^{q_N \cdot t_I} \text{Adv}_E^{\text{PRP}}(\mathcal{C}_i) \cdot B_{l+s}^{i-1}. \quad (1) \end{aligned}$$

It is easy to see, that in the experiment  $\mathbf{Exp}_{\text{sMGMs}[\rho_I, \rho_1, \dots, \rho_{q_N \cdot t_I}]}^{\text{MRAE-int}}$  intermediate keys for tag generation process are produced, but not used — random functions, used to produce coefficients  $H_j$ , are selected independently from them. From here, we can consider an experiment, where intermediate keys are not generated. Moreover, since the inputs to the functions  $\rho_I, \dots, \rho_{q_N \cdot t_I}$  do not intersect (for repeating nonces we just reuse previously computed coefficients  $H_j$ ), due to Statement 1, we can unite them under a single random function  $\rho_h$ . Hence,

$$\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_I, \rho_1, \dots, \rho_{q_N \cdot t_I}]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] = \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_h]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right]$$

The next step is to proceed only with the tag generation part of  $\text{sMGMs}[\rho_h]$ . For this let us introduce an auxiliary MAC construction  $\text{sMGM-MAC}$ .

$\text{sMGM-MAC.Gen}()$ <hr style="border: 0.5px solid black;"/> $\rho_t, \rho_h \xleftarrow{\mathcal{U}} \text{Func}(n)$ $K \leftarrow (\rho_t, \rho_h)$ $\text{return } K$	$\text{sMGM-MAC.Tag}(K, N, M)$ <hr style="border: 0.5px solid black;"/> $\tau \leftarrow \text{PreTag}(\rho_h, N, M)$ $T \leftarrow \rho_t(\tau)$ $\text{return } T$
$\text{PreTag}(\rho_h, N, M)$ <hr style="border: 0.5px solid black;"/> $l \leftarrow  M _n$ $\text{for } i = 1 \dots \ell \text{ do:}$ $H_i \leftarrow \rho_h(00 \parallel \text{str}_{n-2}(N + i - 1))$ $\tau \leftarrow \text{Set11}_r \left( \bigoplus_{i=1}^l (M_i \otimes H_i) \right)$ $\text{return } \tau$	$\text{sMGM-MAC.Verify}(K, N, M, T)$ <hr style="border: 0.5px solid black;"/> $\tau \leftarrow \text{PreTag}(\rho_h, N, M)$ $T' \leftarrow \rho_t(\tau)$ $\text{if } T' \neq T: \text{ return false}$ $\text{return true}$

Figure 4: The sMGM-MAC scheme

We claim that there exists an UF-CMA-adversary  $\mathcal{D}$ , making at most  $q_E$  queries to the *Tag* oracle and at most  $q_D$  queries to the *Verify* oracle, such that

$$\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_h]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1 \right] \leq \Pr \left[ \mathbf{Exp}_{\text{sMGM-MAC}}^{\text{UF-CMA}}(\mathcal{D}) \rightarrow 1 \right].$$

Indeed, let us construct the adversary  $\mathcal{D}$ , that uses the adversary  $\mathcal{A}$  as a black box. The adversary  $\mathcal{D}$  intercepts the queries of the adversary  $\mathcal{A}$  and process them by itself using its own oracles. For encryption/decryption  $\mathcal{D}$  implements lazy sampling for  $\rho_C$ . For tag generation/tag verification the adversary  $\mathcal{D}$  implements the padding procedure and sends the appropriate queries to its oracles.

If  $\mathcal{A}$  makes a non-trivial valid query  $(N, A, C, T)$  to the *Decrypt* oracle, then the adversary  $\mathcal{D}$  decrypts  $C$  using  $\rho_C$  to obtain a plaintext  $P$  and then makes corresponding non-trivial query  $(N, M = A \parallel 0^a \parallel P \parallel 0^c \parallel \text{len}_A \parallel \text{len}_P, T)$  to the *Verify* oracle. Hence, if the adversary  $\mathcal{A}$  forges, then the adversary  $\mathcal{D}$  also forges in  $\mathbf{Exp}_{\text{sMGM-MAC}}^{\text{UF-CMA}}$ .

Finally, we can apply Corollary 1 from [1] to obtain a bound for  $\Pr \left[ \mathbf{Exp}_{\text{sMGM-MAC}}^{\text{UF-CMA}}(\mathcal{D}) \rightarrow 1 \right]$ :

$$\Pr \left[ \mathbf{Exp}_{\text{sMGM-MAC}}^{\text{UF-CMA}}(\mathcal{D}) \rightarrow 1 \right] \leq \frac{q(q-1)}{2^{n-1}} + \frac{q_D}{2^n}.$$

Summarizing all the obtained bounds, we have

$$\begin{aligned}
 & \Pr [\mathbf{Exp}_{\text{sMGMs}[E_K]}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1] \leq \\
 & \leq \left( \left( \frac{q(q-1)}{2^{n-1}} + \frac{q_D}{2^n} \right) \cdot B_{l+s}^{q_N t_I} + \sum_{i=1}^{q_N \cdot t_I} \text{Adv}_E^{\text{PRP}}(\mathcal{C}_i) \cdot B_{l+s}^{i-1} \right) \cdot B_{q(2l_0+2s+1)} + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_0) \leq \\
 & \leq \left( \frac{q(q-1)}{2^{n-1}} + \frac{q_D}{2^n} + \sum_{i=1}^{q_N \cdot t_I} \text{Adv}_E^{\text{PRP}}(\mathcal{C}_i) \right) \cdot B_{l+s}^{q_N t_I} \cdot B_{q(2l_0+2s+1)} + \text{Adv}_E^{\text{PRP}}(\mathcal{C}_0). \quad (2)
 \end{aligned}$$

Denoting by  $\mathcal{C}$  the adversary with the the biggest advantage among  $\mathcal{C}_i$ , we obtain the statement of the Theorem.  $\square$

### 5.3 MRAE security of sMGMs

**Theorem 3.** *For any MRAE-adversary  $\mathcal{A}$  for sMGMs, making at most  $q_E$  queries to the Encrypt oracle and at most  $q_D$  queries to the Decrypt oracle, where the block-length of associated data in each query is at most  $m_A$ , the block-length of plaintexts and ciphertexts in each query is at most  $m_P$ , the number of distinct nonce values in all queries is at most  $q_N$  and the number of queries with the same nonce is at most  $q_R$ , there exist PRP-adversaries  $\mathcal{B}_0, \mathcal{B}_I$  and  $\mathcal{B}_C$  for block cipher  $E$ , such that*

$$\begin{aligned}
 \text{Adv}_{\text{sMGM}[E,r,l_0,l,1]}^{\text{MRAE}}(\mathcal{A}) & \leq \frac{q^2}{2^{n-1}} + \frac{q^2 \max(l, l_0)}{2^{n-2}} + \frac{q_D}{2^n} + \frac{q^2(2l_0 + 2s + 1)^2}{2^{n+1}} + \\
 & + q_N t_I \frac{(l + s)^2}{2^{n+1}} + q_N t_C \frac{(q_R l + s)^2}{2^{n+1}} + \text{Adv}_E^{\text{PRP}}(\mathcal{B}_0) + q_N t_I \text{Adv}_E^{\text{PRP}}(\mathcal{B}_I) + \\
 & + q_N t_C \text{Adv}_E^{\text{PRP}}(\mathcal{B}_C),
 \end{aligned}$$

where  $q = q_E + q_D$ ,  $s = \lceil k/n \rceil$ ,  $t_I = \lceil (m_A + m_P + 2 - l_0)/l \rceil$  and  $t_C = \lceil (m_P - l_0)/l \rceil$ . Adversary  $\mathcal{B}_0$  makes at most  $q(2l_0 + 2s + 1)$  queries to its oracle,  $\mathcal{B}_I$  makes at most  $l + s$  queries and  $\mathcal{B}_C$  makes at most  $q_R l + s$  queries.

*Proof.* We start with replacing all block ciphers with random functions. This will allow us to use the MRAE security theorem for SIV constructions from [14] to bound the security of sMGMs by PRF security of sMGM-MAC and IND-CPA\$ security of CTR-KM (with random IV and independent random functions used for processing each section).

First of all, we replace  $E_K$  with a random function  $\rho_0$ . As in the previous proof, we firstly replace it with a random permutation, building a PRP adversary  $\mathcal{B}_0$ . After that we use PRP/PRF Switching Lemma to replace random permutation with a random function. It is easy to see, that there are at

most  $q(2l_0 + 2s + 1)$  calls to  $E_K$ , hence, we have

$$\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[E_K, E]}^{\text{MRAE}-0}(\mathcal{A}) \rightarrow 1 \right] \leq \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_0, E]}^{\text{MRAE}-0}(\mathcal{A}) \rightarrow 1 \right] + \frac{q(2l_0 + 2s + 1)(q(2l_0 + 2s + 1) - 1)}{2^{n+1}} + \text{Adv}_E^{\text{PRP}}(\mathcal{B}_0).$$

At the next step we replace all other block ciphers with random functions as in the previous proof. However, we can't use Bernstein's lemma to switch from pseudorandom permutation to pseudorandom function, thus we have to apply PRP/PRF Switching Lemma [10]. There are at most  $q_N \cdot t_I$  keys in the tag generation part and  $q_N \cdot t_C$  keys in the encryption part. We construct adversaries  $\mathcal{B}_i^I$  and  $\mathcal{B}_i^C$  for each block cipher used in the tag generation and encryption parts respectively. For each block cipher in the tag generation part an adversary  $\mathcal{B}_i^I$  makes at most  $l + s$  queries (for processing the section and re-keying). For each block cipher in the encryption part an adversary  $\mathcal{B}_i^C$  makes at most  $q_R l + s$  queries (we multiply by  $q_R$  since block cipher inputs for  $\Gamma_j$  generation are distinct even if the same nonce is used). We denote a mode with independent random functions by  $\text{sMGMs}[\rho_0, \hat{\rho}]$ . At this point we have

$$\Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_0, E]}^{\text{MRAE}-0}(\mathcal{A}) \rightarrow 1 \right] \leq \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_0, \hat{\rho}]}^{\text{MRAE}-0}(\mathcal{A}) \rightarrow 1 \right] + q_N t_I \frac{(l + s)(l + s - 1)}{2^{n+1}} + q_N t_C \frac{(q_R l + s)(q_R l + s - 1)}{2^{n+1}} + \sum_{i=1}^{q_N \cdot t_I} \text{Adv}_E^{\text{PRP}}(\mathcal{B}_i^I) + \sum_{i=1}^{q_N \cdot t_C} \text{Adv}_E^{\text{PRP}}(\mathcal{B}_i^C).$$

In sequel we denote by  $\mathcal{B}_I$  ( $\mathcal{B}_C$ ) an adversary with the biggest advantage among  $\mathcal{B}_i^I$  ( $\mathcal{B}_i^C$  resp.).

We will denote by  $\text{CTR-KM}[\hat{\rho}_C]$  a CTR-KM (see Figure 2) construction, in which for each unique nonce in each re-keying section an independent random function is used to produce  $\Gamma_i$  (in queries with a repeating nonce the same sequence of independent functions is used). Encryption and decryption algorithms for  $\text{CTR-KM}[\hat{\rho}_C]$  are defined naturally.

Since inputs to random functions in the tag generation and encryption parts of  $\text{sMGMs}$  do not intersect, due to Statement 1, we claim, that these two parts are independent from each other. Finally we apply Theorem 1 [14]. There exist adversaries  $\mathcal{D}$  and  $\mathcal{C}$  such that

$$\begin{aligned} & \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_0, \hat{\rho}]}^{\text{MRAE}-0}(\mathcal{A}) \rightarrow 1 \right] - \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[E_K, E]}^{\text{MRAE}-1}(\mathcal{A}) \rightarrow 1 \right] = \\ & = \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_0, \hat{\rho}]}^{\text{MRAE}-0}(\mathcal{A}) \rightarrow 1 \right] - \Pr \left[ \mathbf{Exp}_{\text{sMGMs}[\rho_0, \hat{\rho}]}^{\text{MRAE}-1}(\mathcal{A}) \rightarrow 1 \right] = \\ & = \text{Adv}_{\text{sMGMs}[\rho_0, \hat{\rho}]}^{\text{MRAE}}(\mathcal{A}) \leq \text{Adv}_{\text{sMGM-MAC}[\rho_t, \rho_h]}^{\text{PRF}}(\mathcal{D}) + \text{Adv}_{\text{CTR-KM}[\hat{\rho}_C]}^{\text{IND-CPA\$}}(\mathcal{C}) + \frac{q_D}{2^n}. \end{aligned}$$

The only thing left is to derive a bound for  $\text{Adv}_{\text{CTR-KM}[\hat{\rho}_C]}^{\text{IND-CPAS}}(\mathcal{C})$ . The idea is similar to the classical proof of IND-CPAS security of CTR from [6]. In that proof the bad case happens if counters in two queries overlap. In our case, since each section is processed with its own independent random function, the bad case happens if for two queries counters in the same section overlap. We denote that event by  $\text{Bad}$  and an event, that counters overlap in queries  $j_1$  and  $j_2$ , by  $\text{Bad}_{j_1 j_2}$ .

We notice, that if in queries  $j_1$  and  $j_2$  counters overlap in the  $i$ -th section, then the following inequality holds

$$\begin{aligned} IV_{j_1} + k'(i) - l'(i) + 1 &\leq IV_{j_2} + k'(i) \leq IV_{j_1} + k'(i) + l'(i) - 1 \Leftrightarrow \\ IV_{j_1} - l'(i) + 1 &\leq IV_{j_2} \leq IV_{j_1} + l'(i) - 1, \end{aligned}$$

where  $l'(i)$  is a length of the  $i$ -th section (equal to  $l_0$  if  $i = 0$  and to  $l$  otherwise) and  $k'(i)$  is the counter offset in the beginning of the  $i$ -th section (equal to 0 if  $i = 0$  and to  $l_0 + l(i - 1)$  otherwise). Hence, for the probability of the event  $\text{Bad}_{j_1 j_2}$  we have

$$\Pr[\text{Bad}_{j_1 j_2}] = \Pr\left[IV_{j_1}, IV_{j_2} \stackrel{\mathcal{U}}{\leftarrow} \{0, 1\}^{n-2} : \exists i : IV_{j_1} - l'(i) + 1 \leq IV_{j_2} \leq IV_{j_1} + l'(i) - 1\right].$$

Since for every  $0 \leq i \leq t_C$  it is true, that  $l'(i) \leq \max(l_0, l)$ , we can bound the probability in the following way

$$\begin{aligned} &\Pr\left[IV_{j_1}, IV_{j_2} \stackrel{\mathcal{U}}{\leftarrow} \{0, 1\}^{n-2} : \exists i : IV_{j_1} - l'(i) + 1 \leq IV_{j_2} \leq IV_{j_1} + l'(i) - 1\right] \leq \\ &\leq \Pr\left[IV_{j_1}, IV_{j_2} \stackrel{\mathcal{U}}{\leftarrow} \{0, 1\}^{n-2} : IV_{j_1} - \max(l_0, l) + 1 \leq IV_{j_2} \leq IV_{j_1} + \max(l_0, l) - 1\right] = \\ &= \frac{2 \max(l_0, l) - 1}{2^{n-2}}. \end{aligned}$$

From that we obtain a bound for the event  $\Pr[\text{Bad}]$  (and, therefore for the adversarial advantage), going through all possible pairs of queries:

$$\begin{aligned} \text{Adv}_{\text{CTR-KM}[\rho]}^{\text{IND-CPAS}}(\mathcal{C}) &\leq \Pr[\text{Bad}] \leq \sum_{1 \leq j_1 < j_2 \leq q} \Pr[\text{Bad}_{j_1 j_2}] \leq \\ &\leq \frac{q(q-1)}{2} \cdot \frac{2 \max(l_0, l) - 1}{2^{n-2}} \leq \frac{q^2 \max(l_0, l)}{2^{n-2}}. \end{aligned}$$

Finally, using Lemma 1 from [1] to obtain a bound for  $\text{Adv}_{\text{sMGM-MAC}[\rho_t, \rho_h]}^{\text{PRF}}(\mathcal{D})$  and connecting everything together, we have the required bound.  $\square$

## 6 Open problems

In the future work we are going to develop the proposed parameterizable AEAD conception by adding new security features provided by the mode with respect to exploitation properties. Such properties as leakage resilience, RUP-security, Key-dependent messages security are to be considered in particular. We believe that the designated goal can be achieved in sMGM without significant difficulties by combining the building blocks of the mode in an appropriate way.

## References

- [1] Liliya Akhmetzyanova, Evgeny Alekseev, Alexandra Babueva, Andrey Bozhko, Stanislav Smyshlyaev, (2022) *Misuse-resistant MGM2 mode*, International Journal of Open Information Technologies, Vol 10, No 1.
- [2] Akhmetzyanova L., Alekseev E., Smyshlyaev S., Oshkin I. (2020) *On Internal Re-keying*. In: van der Merwe T., Mitchell C., Mehrnezhad M. (eds) Security Standardisation Research. SSR 2020. Lecture Notes in Computer Science, vol 12529. Springer, Cham. [https://doi.org/10.1007/978-3-030-64357-7\\_2](https://doi.org/10.1007/978-3-030-64357-7_2)
- [3] Andreeva E., Bogdanov A., Luykx A., Mennink B., Mouha N., Yasuda K. (2014) *How to Securely Release Unverified Plaintext in Authenticated Encryption*. In: Sarkar P., Iwata T. (eds) Advances in Cryptology – ASIACRYPT 2014. ASIACRYPT 2014. Lecture Notes in Computer Science, vol 8873. Springer, Berlin, Heidelberg. [https://doi.org/10.1007/978-3-662-45611-8\\_6](https://doi.org/10.1007/978-3-662-45611-8_6)
- [4] Davide Bellizia and Olivier Bronchain and Gaëtan Cassiers and Vincent Grosso and Chun Guo and Charles Momin and Olivier Pereira and Thomas Peters and François-Xavier Standaert, *Mode-Level vs. Implementation-Level Physical Security in Symmetric Cryptography: A Practical Guide Through the Leakage-Resistance Jungle*, Cryptology ePrint Archive, Report 2020/211, 2020, <https://eprint.iacr.org/2020/211>
- [5] Brandstetter L., Fischlin M., Schröder R.L., Yonli M. (2020) *On the Memory Fault Resilience of TLS 1.3*. In: van der Merwe T., Mitchell C., Mehrnezhad M. (eds) Security Standardisation Research. SSR 2020. Lecture Notes in Computer Science, vol 12529. Springer, Cham. [https://doi.org/10.1007/978-3-030-64357-7\\_1](https://doi.org/10.1007/978-3-030-64357-7_1)
- [6] Bellare M., Rogaway P. *Introduction to modern cryptography* //Ucsd Cse. – 2005. – T. 207. – C. 207.
- [7] Bernstein, D.J.: *Stronger Security Bounds for Permutations* (2005), <http://cr.yp.to/papers.html> (accessed on May 31, 2012)
- [8] John Black, Phillip Rogaway, and Thomas Shrimpton. 2002. *Encryption-Scheme Security in the Presence of Key-Dependent Messages*. In Revised Papers from the 9th Annual International Workshop on Selected Areas in Cryptography (SAC '02). Springer-Verlag, Berlin, Heidelberg, 62–75.
- [9] Chakraborty, D., López, C.M. & Sarkar, P. *Disk encryption: do we need to preserve length?*. J Cryptogr Eng 8, 49–69 (2018). <https://doi.org/10.1007/s13389-016-0147-0>
- [10] D. Chang and M. Nandi, *A Short Proof of the PRP/PRF Switching Lemma* // IACR ePrint Archive, 2008, Report 2008/078, <https://eprint.iacr.org/2008/078>.
- [11] Federal Agency on Technical Regulating and Metrology, *Information technology. Cryptographic data security. Authenticated encryption block cipher operation modes*, R 1323565.1.026-2019, 2019.
- [12] Gueron S., Lindell Y. *GCM-SIV: full nonce misuse-resistant authenticated encryption at under one cycle per byte* //Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. – 2015. – C. 109-119.

- [13] Hoang V.T., Krovetz T., Rogaway P. (2015) *Robust Authenticated-Encryption AEZ and the Problem That It Solves*. In: Oswald E., Fischlin M. (eds) *Advances in Cryptology – EUROCRYPT 2015*. EUROCRYPT 2015. Lecture Notes in Computer Science, vol 9056. Springer, Berlin, Heidelberg. [https://doi.org/10.1007/978-3-662-46800-5\\_2](https://doi.org/10.1007/978-3-662-46800-5_2)
- [14] Rogaway P., Shrimpton T. *A provable-security treatment of the key-wrap problem* // Annual International Conference on the Theory and Applications of Cryptographic Techniques. – Springer, Berlin, Heidelberg, 2006. – C. 373-390.
- [15] Smyshlyaev, S., Nozdrunov, V., Shishkin, V., and E. Smyshlyaeva *Multilinear Galois Mode (MGM)* // 2019, <<https://tools.ietf.org/html/draft-smyshlyaev-mgm-17>>

## A Security models

This section introduces models for an adversary that may repeat nonces in its queries. We begin with the strongest model, which formalizes both integrity and confidentiality properties – MRAE («Misuse-Resistant Authenticated Encryption - integrity»), firstly introduced in [14].

**Definition 1.** *For an AEAD-scheme  $\Pi$  the advantage of a MRAE-adversary  $\mathcal{A}$  is defined as follows:*

$$\text{Adv}_{\Pi}^{\text{MRAE}}(\mathcal{A}) = \Pr [\mathbf{Exp}_{\Pi}^{\text{MRAE}-1}(\mathcal{A}) \rightarrow 1] - \Pr [\mathbf{Exp}_{\Pi}^{\text{MRAE}-0}(\mathcal{A}) \rightarrow 1],$$

where experiments  $\mathbf{Exp}_{\Pi}^{\text{MRAE-int}}$  are defined below:

$\mathbf{Exp}_{\Pi}^{\text{MRAE}-b}(\mathcal{A})$	<i>Oracle Encrypt<sup>b</sup>(N, A, P)</i>	<i>Oracle Decrypt<sup>b</sup>(N, A, C, T)</i>
$K \xleftarrow{\$} \Pi.\text{Gen}()$	<b>if</b> $(N, A, P, \cdot, \cdot) \in \text{sent}$ :	<b>if</b> $(N, A, \cdot, C, T) \in \text{sent}$ :
$\text{sent} \leftarrow \emptyset$	<b>return</b> $\perp$	<b>return</b> $\perp$
$b' \xleftarrow{\$} \mathcal{A}^{\text{Encrypt}^b, \text{Decrypt}^b}()$	<b>if</b> $b = 1$ :	<b>if</b> $b = 1$ :
<b>return</b> $b'$	$(C, T) \leftarrow \Pi.\text{Enc}(K, N, A, P)$	<b>return</b> $\Pi.\text{Dec}(K, N, A, C, T)$
	<b>else</b> :	<b>else</b> :
	$C \parallel T \xleftarrow{\mathcal{U}} \{0, 1\}^{ P +s}$	<b>return</b> $\perp$
	$\text{sent} \leftarrow \text{sent} \cup \{(N, A, P, C, T)\}$	
	<b>return</b> $(C, T)$	

We also separately define a model formalizing the integrity property of AEAD schemes in nonce misuse setting – MRAE-int.

**Definition 2** (MRAE-int). *For an AEAD-scheme  $\Pi$  the advantage of a MRAE-int-adversary  $\mathcal{A}$  is defined as follows:*

$$\text{Adv}_{\Pi}^{\text{MRAE-int}}(\mathcal{A}) = \Pr [\mathbf{Exp}_{\Pi}^{\text{MRAE-int}}(\mathcal{A}) \rightarrow 1],$$

where experiment  $\mathbf{Exp}_{\Pi}^{\text{MRAE-int}}$  is defined below:



$\mathbf{Exp}_{\Pi}^{\text{MRAE-int}}(\mathcal{A})$	<i>Oracle Encrypt</i> ( $N, A, P$ )	<i>Oracle Decrypt</i> ( $N, A, C, T$ )
$K \xleftarrow{\$} \Pi.\text{Gen}()$	$(C, T) \leftarrow \Pi.\text{Enc}(K, N, A, P)$	$P \leftarrow \Pi.\text{Dec}(K, N, A, C, T)$
$\text{sent} \leftarrow \emptyset$	$\text{sent} \leftarrow \text{sent} \cup \{(N, A, C, T)\}$	<b>if</b> $(P \neq \perp) \wedge ((N, A, C, T) \notin \text{sent})$ :
$\text{win} \leftarrow \text{false}$	<b>return</b> $(C, T)$	$\text{win} \leftarrow \text{true}$
$\mathcal{A}^{\text{Encrypt, Decrypt}}()$		<b>return</b> $P$
<b>return</b> $\text{win}$		

# Keyed Streebog is a secure PRF and MAC

Vitaly Kiryukhin

LLC «SFB Lab», JSC «InfoTeCS», Moscow, Russia  
vitaly.kiryukhin@sfblaboratory.ru

## Abstract

One of the most popular ways to turn a keyless hash function into a keyed one is the HMAC algorithm. This approach is too expensive in some cases due to double hashing. Excessive overhead can sometimes be avoided by using certain features of the hash function itself. The paper presents a simple and safe way to create a keyed cryptoalgorithm (conventionally called «Streebog-K») from hash function Streebog  $H(M)$ . Let  $K$  be a secret key, then  $KH(K, M) = H(K||M)$  is a secure pseudorandom function (PRF) and, therefore, a good message authentication code (MAC). The proof is obtained by reduction of the security of the presented construction to the resistance of the underlying compression function to the related key attacks (PRF-RKA). The security bounds of Streebog-K are essentially the same as those of HMAC-Streebog, but the computing speed doubles when short messages are used.

**Keywords:** Streebog, Streebog-K, PRF, MAC, HMAC, provable security

## 1 Introduction

The HMAC algorithm was proposed in 1996 [10] as an efficient way to construct a keyed transformation (and, most importantly, a secure message authentication code) from a keyless hash function  $H(M)$

$$\text{HMAC}(K, M) = H((\overline{K} \oplus opad)||H(\overline{K} \oplus ipad||M)),$$

where  $\overline{K}$  is obtained by padding the secret key  $K$  with zero bits, *opad* and *ipad* are different nonzero constants.

The security proof [10] explicitly assumes the use of a «plain» Merkle-Damgård [7, 8] cascade as an underlying hash function  $H(M)$ : the message  $M$  is padded and splitted into  $b$ -bit blocks; the compression function  $g$  is iteratively applied to the previous  $n$ -bit state and  $b$ -bit block; the initial state  $IV$  is the predefined constant; the last state is the result of hashing. The result largely depends, among other things, on the weak collision resistance (WCR) of  $H$  in the «secret initial state» setting.

Collision resistance is broken in practice for several widely used hash functions, such as MD5 and SHA-1. In 2006, an updated proof was presented

in [14], showing that the **HMAC-MD5** and **HMAC-SHA-1** nevertheless remain secure. The reduction shows that **HMAC** is a secure pseudorandom function (PRF) if  $\mathbf{g}$  is a secure PRF (in the secret key and also in some restricted related-key settings). The proof was obtained via a non-uniform reduction (with «non-constructible» adversaries), leading to the insignificance of this result in practice [13].

In [13], along with a critique of the results [14], an alternative proof was also presented (the definition of PRF is slightly different). More precise bounds with the same initial requirements [14] and without the use of a «non-uniform computation model» were also obtained in the works [15, 17].

Russian hash function **Streebog** [1] can also be used in the **HMAC** [3, 6]. **Streebog** uses a modified Merkle-Damgård approach. Its compression function is based on a 12-rounds AES-like block cipher in Miyaguchi-Preneel mode. The internal state and the message block consist of  $n = 512$  bits. The output length of hash function can be either 512 or 256-bit.

The most important differences from the «plain» cascade are the following:

- before processing the  $i$ -th block, the state is summed modulo 2 with the number of already hashed bits;
- the last call of the compression function is used to «mix» the checksum (modulo  $2^n$ ) of all message blocks.

It is important to note that the differences between **Streebog** and the Merkle-Damgård scheme do not allow direct use of the results [10, 14, 13, 15, 17] for **HMAC-Streebog**. The proof of the latter's security was, among other things, given in [16]. However, the reduction descends not to the properties of the compression function  $\mathbf{g}$ , but to the properties of the hash function **H** itself.

Unfortunately, **HMAC-Streebog** has a significant overhead when working with short messages. We have at least 8 (resp. 9) calls of  $\mathbf{g}$  for **HMAC-Streebog-256** (resp. 512). However, the design of **Streebog** implicitly generates a more efficient solution.

The aforementioned features allow us to prove that «**Streebog-K**» («Keyed **Streebog**»)  $\mathbf{KH}(K, M) = \mathbf{H}(\overline{K}||M)$  is a secure pseudorandom function (PRF) under some plausible assumptions about the compression function  $\mathbf{g}$ . Thus, processing a short message requires 4 computations of  $\mathbf{g}$ : padded key, padded message, bit length, checksum. It is also easy to see that the proposed cryptalgorithm *does not require any changes* in **Streebog** itself. Other methods of involving  $K$ , such as secret-IV [11], along with simplifying the finalization and a number of small changes can provide a more computa-

tionally efficient and no less secure solution. Unfortunately, all this requires edits in the formal description of the hash function and in many existing implementations. Therefore, we consider «**Streebog-K**» that is devoid of these disadvantages.

The security of **Streebog** against the length-extension attack (i.e. the particular case of PRF-security) is explicitly claimed in [5]. However, as far as we know, there are no publicly available formal proofs.

The analysis of **Streebog-K** was carried out in the paradigm of provable security [19, 18]. We start from high-level description of **Streebog** (section 3) and its equivalent representation [22] carefully considered in the proof. Next, in section 4 we present and discuss «hard-to-solve» problems: the indistinguishability of  $\mathbf{g}$  from family of random functions under related key attacks (PRF-RKA) in two various settings. In the main part of the paper (section 5) we reduce the PRF properties of **Streebog-K** to the PRF-RKA properties of  $\mathbf{g}$ . Roughly speaking, if there is an effective attack against **Streebog-K**, then there is an attack against  $\mathbf{g}$ . The reduction gives us the upper bound on the probability of the adversary's success (for example, the forgery or the key recovery). The bound functionally depends on the capabilities of the adversary (amount of the computation resources, the number of adaptively chosen input-output pairs).

Two «beyond security bound» attacks against **Streebog-K** were also briefly considered (section 6). The first is the simple forgery attack, the second one is the key recovery attack, almost identical to the same against **HMAC-Streebog** [23].

Similar proofs can be also relatively easily obtained for **HMAC-Streebog** [3] and **S3G** [4]. The corresponding results are briefly discussed in section 7. The security bounds are almost the same in all cases, but **Streebog-K** requires a weaker notion of PRF-RKA-security from  $\mathbf{g}$ .

The good security bounds in the PRF setting allow you to use **Streebog-K** as a secure MAC and key derivation function.

## 2 Notations and definitions

We use the following notations throughout the paper:

$n = 512$  – block size in bits;  $k \leq 512$  – key size in bits;  $\oplus$  – bitwise XOR operation;  $\boxplus$ ,  $\boxminus$  – addition and subtraction modulo  $2^n = 2^{512}$ ;

$\parallel$  – concatenation of binary strings;

$V^*$  – the set of all binary strings of a finite length;

$V^n$  – the set of all  $n$ -bit strings with naturally defined operations  $\langle\langle\oplus\rangle\rangle$  and  $\langle\langle\boxplus\rangle\rangle$ ;

$V^{\leq L}$  – the set of binary strings of length no more than  $L$  bits;

$(V^n)^{\leq l}$  – the set of binary strings of length no more than  $l \cdot n$  bits, the length of each string is a multiple of  $n$ ;

$\text{bin}(x)$  –  $n$ -bit representation of the integer  $x$ ;

$\text{sum}_{\boxplus}(M) = m_1 \boxplus m_2 \boxplus \dots \boxplus m_l$  – the checksum (modulo  $2^n$ ) of blocks from  $l$ -block message  $M = m_1 || m_2 || \dots || m_l$ ;

$\text{sum}'_{\boxplus}(M) = m_1 \boxplus \dots \boxplus m_{l-1}$  – the checksum of all blocks from the message  $M = m_1 || m_2 || \dots || m_l$ , except for the last block;

$\text{Func}(\mathbf{X}, \mathbf{Y})$  – the set of all mappings from the set  $\mathbf{X}$  to the set  $\mathbf{Y}$ ;

$X \stackrel{\text{R}}{\leftarrow} \mathbf{X}$  – uniform and random selection of element  $X$  from the set  $\mathbf{X}$ .

The adversary is modeled by an interactive probabilistic algorithm that has access to other algorithms (oracles). We denote by  $\text{Adv}_{\mathbf{F}}^{TM}(\mathcal{A})$  a quantitative characterization (advantage) of the capabilities of the adversary  $\mathcal{A}$  in realizing a certain threat, defined by the model  $TM$ , for the cryptographic scheme  $\mathbf{F}$ . The resources of  $\mathcal{A}$  are measured in terms of time and query complexities. The time complexity  $t$  includes the description size of  $\mathcal{A}$  in some computation model. The query complexity  $q$  is measured in the number of adaptively chosen input/output pairs. If  $\mathbf{F}$  has a variable input length, the maximum length  $l_{\max}$  of the query (in  $n$ -bit blocks) is also characteristic of the adversary's resources. Without loss of generality, we assume that  $\mathcal{A}$  always uses exactly  $q$  unique queries (no redundancy and repetitions). The result of computations  $\mathcal{A}$  after interacting with oracles  $\mathcal{O}_1, \mathcal{O}_2, \dots, \mathcal{O}_w, w \in \mathbb{N}$  is some value  $x$  (usually binary),  $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \dots, \mathcal{O}_w} \Rightarrow x$ .

The maximum of the advantage among all resource constrained adversaries is denoted by

$$\text{Adv}_{\text{Alg}}^{TM}(t, q, l_{\max}) = \max_{\mathcal{A}(t', q', l') : t' \leq t, q' \leq q, l' \leq l_{\max}} \text{Adv}_{\text{Alg}}^{TM}(\mathcal{A}).$$

The cryptoalgorithm  $\text{Alg}$  is called secure in the threat model  $TM$  with respect to adversaries limited by resources  $(t, q, l_{\max})$  if  $\text{Adv}_{\mathbf{F}}^{TM}(t, q, l_{\max}) < \varepsilon$ , where  $\varepsilon$  is some small value determined by the requirements for the strength of the cryptosystem.

To demonstrate the practical significance of the obtained results, we sometimes substitute heuristic estimates based on assumptions into derived security bounds. The resulting informal estimates are denoted by symbol  $\langle\langle \lesssim \rangle\rangle$  meaning «less or equal if the assumptions are true».

**Definition.** The advantage of  $\mathcal{A}$  in the model  $PRF$  ( $PRF\text{-}CMA$  – indistinguishability from a random function under chosen message attack)

for the keyed cryptalgorithm  $\mathbf{F} : \mathbf{K} \times \mathbf{X} \rightarrow \mathbf{Y}$  is

$$\text{Adv}_{\mathbf{F}}^{\text{PRF}}(\mathcal{A}) = \Pr\left(K \stackrel{\text{R}}{\leftarrow} \mathbf{K}; \mathcal{A}^{\mathbf{F}_K(\cdot)} \Rightarrow 1\right) - \Pr\left(\mathbf{F} \stackrel{\text{R}}{\leftarrow} \text{Func}(\mathbf{X}, \mathbf{Y}); \mathcal{A}^{\mathbf{F}(\cdot)} \Rightarrow 1\right),$$

where  $\mathbf{K}$ ,  $\mathbf{X}$ ,  $\mathbf{Y}$  are spaces of the keys, messages, and outputs respectively.

As the example, for a PRF with a fixed input length, we have  $(\mathbf{K}, \mathbf{X}, \mathbf{Y}) = (V^n, V^n, V^n)$ . For **Streebog-K**,  $(\mathbf{K}, \mathbf{X}, \mathbf{Y}) = (V^k, V^{\leq L}, V^n)$ .

### 3 Streebog and Streebog-K

Streebog hashes the message  $M \in V^*$  as follows. The text is padded with bit string  $10\dots 0$ . At least one bit is always added, even if the message bit length  $L$  is already divisible by  $n$ . The string  $M' = M||10\dots 0$  is divided into  $l$  blocks of  $n = 512$  bits  $m_1||m_2||\dots||m_l$ . The compression function is sequentially applied to the previous state, the block and the counter

$$h_{i+1} = \mathbf{g}(h_i, m_{i+1}, \mathbf{i}), \quad i = 0, \dots, l-1, \quad h_0 = IV \in V^n,$$

where  $IV$  is a predefined constant which is different in both versions of the hash function, the counter  $\mathbf{i} = \text{bin}(i \cdot n) \in V^n$  is the number of already hashed bits.

Two more transformations are performed at the finalizing stage: the bit length  $L$  and the checksum  $\Sigma = \text{sum}_{\boxplus}(M')$  are «mixed» with the state

$$h_{l+1} = \mathbf{g}(h_l, L, \mathbf{0}), \quad H = \mathbf{g}(h_{l+1}, \Sigma, \mathbf{0}).$$

If 256-bit hash function is used, the output  $H$  truncated to 256 bit.

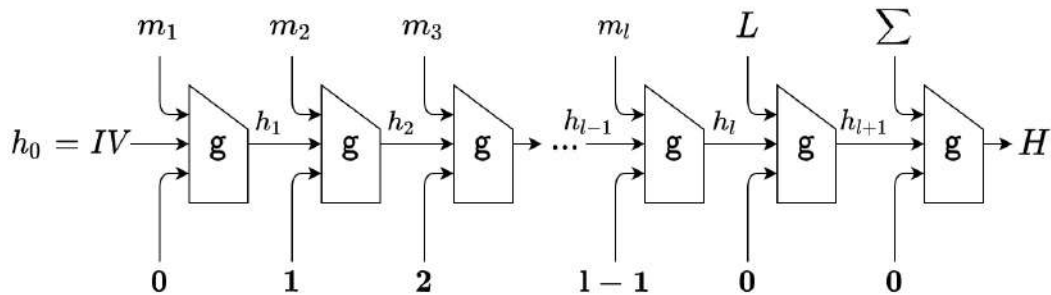


Figure 1: Keyless hash function Streebog-512.

The compression function is based on a 12-rounds AES-like block cipher  $\mathbf{E}$  in Miyaguchi-Preneel mode

$$\mathbf{g}(h_i, m_{i+1}, \mathbf{i}) = \mathbf{E}(h_i \oplus \mathbf{i}, m_{i+1}) \oplus h_i \oplus m_{i+1} = h_{i+1}.$$

In [22], the equivalent representation was proposed (see the detailed figure in the Appendix)

$$\begin{aligned}
 h_{i+1} &= \underbrace{\mathbf{E}(h_i \oplus \mathbf{i}, m_{i+1}) \oplus (h_i \oplus \mathbf{i}) \oplus m_{i+1} \oplus \mathbf{i}}_{\mathbf{g}'(h_i \oplus \mathbf{i}, m_{i+1})}, \\
 h_{i+1} &= \mathbf{g}'(h_i \oplus \mathbf{i}, m_{i+1}) \oplus \mathbf{i}, \\
 h_{i+2} &= \mathbf{g}'(\mathbf{g}'(h_i \oplus \mathbf{i}, m_{i+1}) \oplus \underbrace{\mathbf{i} \oplus (\mathbf{i} \boxplus \mathbf{1})}_{\Delta_i}, m_{i+2}) \oplus (\mathbf{i} \boxplus \mathbf{1}).
 \end{aligned}$$

Adjacent counters are summed with each other. However, the last counter appears differently  $h_l = \mathbf{g}'(h_{l-1} \oplus (\mathbf{1} \boxplus \mathbf{1}), m_l) \oplus \underbrace{(\mathbf{1} \boxplus \mathbf{1})}_{\tilde{\Delta}_{l-1}}$ .

Hence,  $\mathbf{g}(h_i, m_{i+1}, \mathbf{i})$  is replaced by

$$\begin{aligned}
 \mathbf{g}(h_i, m_{i+1}) &= \mathbf{E}(h_i, m_{i+1}) \oplus h_i \oplus m_{i+1} \oplus \Delta_i, \quad i = 0, \dots, l-2, \\
 \mathbf{g}(h_i, m_{i+1}) &= \mathbf{E}(h_i, m_{i+1}) \oplus h_i \oplus m_{i+1} \oplus \tilde{\Delta}_i, \quad i = l-1,
 \end{aligned}$$

and the sequence of unique counters  $\mathbf{i}$  is replaced by a «quasi-periodic» one  $\Delta_i = \mathbf{i} \oplus (\mathbf{i} \boxplus \mathbf{1})$ , for example,

$$\begin{aligned}
 \Delta_0, \Delta_1, \dots, \tilde{\Delta}_{15} &= \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{7}, \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{15}, \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{7}, \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{15}, \\
 \Delta_0, \Delta_1, \dots, \Delta_{15}, \tilde{\Delta}_{16} &= \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{7}, \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{15}, \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{7}, \mathbf{1}, \mathbf{3}, \mathbf{1}, \mathbf{31}, \mathbf{16}.
 \end{aligned}$$

Also, it is important that  $\Delta_i \neq \tilde{\Delta}_i \forall i = 0, \dots, 2^n - 1$ .

The keyed cryptoalgorithm **Streebog-K** defined as (fig. 2)

$$\mathbf{KH}(K, M) = \mathbf{H}(\overline{K} || M) = \mathbf{H}(K || \underbrace{0 \dots 0}_{n-k} || M), \quad K \in V^k, \overline{K} \in V^n,$$

where  $256 \leq k \leq 512 = n$  and  $K$  is padded with zero bits if necessary (as in [3]). **Streebog-256** and **Streebog-512** can be used as  $\mathbf{H}$  without significant differences in properties. Note that due to the key's prepending, the last value  $\tilde{\Delta}_l = \mathbf{1}$  has the index  $l$ , and not  $l-1$ . Further in the text, the compression function means  $\mathbf{g}(h, m) = \mathbf{E}(h, m) \oplus h \oplus m$ .

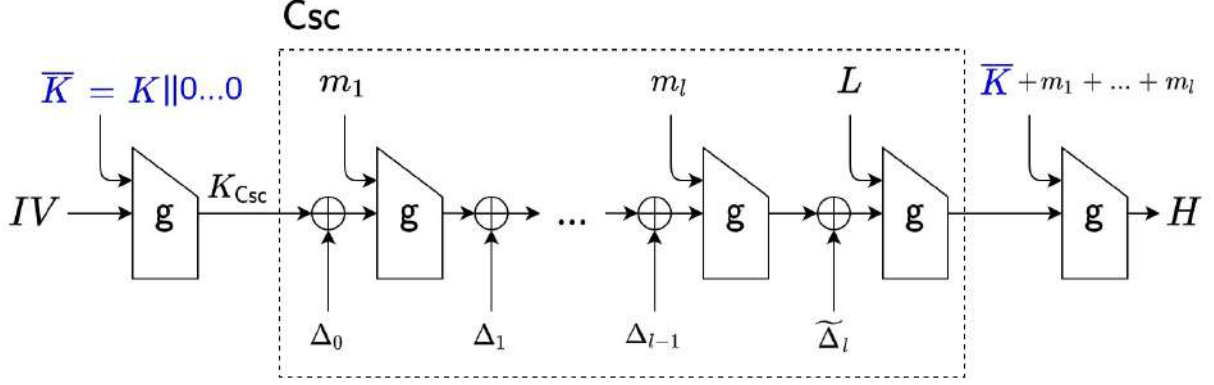


Figure 2: Streebog-K with the equivalent representation [22].

## 4 Related key attack settings

The security proof presented in the next section shows that if the adversary can break PRF-security of **Streebog-K**, then one of the following two problems is also easy to solve. However, we expect these problems to be hard –  $g$  successfully resists attacks using related keys in (at least) two settings. Therefore, the security of **Streebog-K** is also difficult to break.

**Problem 1.** *PRF-RKA $_{\oplus}$* -security of  $g_K^{\triangleright}(\cdot) = g(K, \cdot)$  in sense

$$\begin{aligned} \text{Adv}_{g^{\triangleright}}^{\text{PRF-RKA}_{\oplus}}(\mathcal{A}) = & \Pr \left( K \stackrel{R}{\leftarrow} V^n, \mathcal{A}^{g(K, \cdot), g(K \oplus \phi, \cdot)} \Rightarrow 1 \right) - \\ & - \Pr \left( f, f' \stackrel{R}{\leftarrow} \text{Func}(V^n, V^n), \mathcal{A}^{f(\cdot), f'(\cdot)} \Rightarrow 1 \right) \end{aligned}$$

– the pair of the compression functions (with the key  $K$  and with the related key  $K \oplus \phi$ ) is indistinguishable from the pair of random functions. The value of  $\phi \neq 0$  is chosen once by the adversary before the sequence of queries. The query consists of the block  $m$  and the binary flag «key  $K$ »/«key  $K \oplus \phi$ ».

In the most favorable case, there are only two distinguishing methods: brute-force attack against two keys and birthday-paradox

$$\text{Adv}_{g^{\triangleright}}^{\text{PRF-RKA}_{\oplus}}(t, q) \lesssim \frac{2 \cdot t}{2^n} + \frac{q^2}{2^{n+1}}.$$

**Problem 2.** *PRF-RKA $_{\boxplus}$* -security of  $g_K^{\nabla} = g(\cdot, K)$ . The relation between the keys is modular addition

$$\begin{aligned} \text{Adv}_{g^{\nabla}}^{\text{PRF-RKA}_{\boxplus}}(\mathcal{A}) = & \Pr \left( K \stackrel{R}{\leftarrow} V^k; \mathcal{A}^{g(\cdot, \bar{K} \boxplus \cdot)} \Rightarrow 1 \right) - \\ & - \Pr \left( K \stackrel{R}{\leftarrow} V^k; f_{\bar{K} \boxplus \sigma} \stackrel{R}{\leftarrow} \text{Func}(V^n, V^n), \forall \sigma \in V^n; \mathcal{A}^{f_{\bar{K} \boxplus \sigma}(\cdot)} \Rightarrow 1 \right). \end{aligned}$$

The query consists of the block  $m$  and the value  $\sigma$ . The response is  $g_{\bar{K} \boxplus \sigma}^{\nabla}(m) = g(m, \bar{K} \boxplus \sigma)$  or  $f_{\bar{K} \boxplus \sigma}(m)$  correspondingly. We can hope that in



the absence of specific vulnerabilities, the only possible attack is the parallel key guessing

$$\text{Adv}_{\mathbf{g}^\nabla}^{\text{PRF-RKA}_{\boxplus}}(t, q) \lesssim \frac{t \cdot q}{2^{k-1}}.$$

The more related keys are used, the easier it is to carry out the attack.

The complexity of solving basic problems should be confirmed by constructive cryptanalysis of the compression function. The impossible differential single-key attack against  $\mathbf{g}^\triangleright$  is presented in [24] and covers 6.75 out of 12 rounds. In [25], attacks on 7 rounds in the PRF model are proposed for  $\mathbf{g}^\triangleright$  and  $\mathbf{g}^\nabla$ . We managed to adapt methods [25] to build only 8-round attacks in the related-key settings.

Despite the many papers on the topic [26, 27, 28, 29, 30, 31, 32], to the best of our knowledge, no effective full-round algorithms for constructing preimages and collisions of various types have been published. This is implicit evidence of the good cryptographic properties of  $\mathbf{g}$ .

The existing results of cryptanalysis, as well as the conservative design of the underlying block cipher  $\mathbf{E}$  and its key schedule, suggest that there are no special attacks on full-round versions of the compression function also in the PRF-RKA settings. In other words, the two basic problems under consideration are actually computationally hard.

The appearance of more efficient cryptographic methods of the compression function  $\mathbf{g}$  will not render the presented security proof of **Streebog-K** incorrect. Specific attacks on  $\mathbf{g}$  can be taken into account in the security bounds due to the absence of heuristic arguments in the proof.

## 5 Proof of PRF-security

Next, we show the reducibility of **Streebog-K** security to the problems discussed in the previous section. The equivalent representation (figure 2) is the start point  $\text{KH}^{(0)}(K, M) = \text{KH}(K, M)$ .

**Step 1.** We define the padding transformation  $\text{pad} : V^* \rightarrow (V^n)^{\leq l_{\max}}$  that sequentially adds to  $M$ :

- the nonempty binary string  $10 \dots 0$  to achieve the multiplicity of the block length;
- the block  $\text{bin}(L)$  representing the length of  $M$  in bits.

Let  $M'$  consists of  $l + 1$  full-length blocks ( $l \geq 1$ ), then

$$\text{KH}^{(0)}(K, M) = \text{KH}^{(1)}(K, \text{pad}(M) = M || 10 \dots 0 || \text{bin}(L)), \quad M \in V^*,$$

$$\text{KH}^{(1)}(K, M') = \mathbf{g}(\text{Csc}(\mathbf{g}(IV, \bar{K}), M'), \bar{K} \boxplus \text{sum}'_{\boxplus}(M')), \quad M' \in (V^n)^{\leq l_{\max}},$$

and the «mixing»  $L$  is an implicit part of the cascade transformation

$$\mathbf{Csc}(K_{\mathbf{Csc}}, M') = \mathbf{g}(\dots \mathbf{g}(\mathbf{g}(K_{\mathbf{Csc}} \oplus \Delta_0, m_1) \oplus \Delta_1, m_2) \dots \oplus \tilde{\Delta}_l, m_{l+1}),$$

where  $K_{\mathbf{Csc}} = \mathbf{g}(IV, \bar{K})$  and  $m_{l+1} = \mathbf{bin}(L)$ .

The  $\mathbf{pad}$  is injective, and hence if  $\mathbf{KH}^{(1)}$  is secure with arbitrary block-length inputs, then  $\mathbf{KH}^{(0)}$  is also an equally good PRF with  $M \in V^*$ .

**Step 2.** We replace  $\mathbf{g}_K^\nabla = \mathbf{g}(\cdot, \bar{K} \boxplus \cdot)$  (the first and last compression functions) with a family of true random functions  $\mathbf{f}_{\bar{K} \boxplus}(\cdot)$  and obtain  $\mathbf{KH}^{(2)}$ . Algorithm  $\mathcal{A}$ , which distinguishes  $\mathbf{KH}^{(2)}$  from  $\mathbf{KH}^{(1)}$ , can be used to attack  $\mathbf{g}_K^\nabla$  in the model  $PRF\text{-}RKA_{\boxplus}$ . The corresponding algorithm  $\mathcal{B}_{RKA}$  works as follows. To process requests from  $\mathcal{A}$ , one preparatory query  $(IV, 0)$  to the oracle  $\mathbf{g}(\cdot, \bar{K} \boxplus \cdot)$  is required. So,  $\mathcal{B}_{RKA}$  obtains  $K_{\mathbf{Csc}}$ . Each query  $M \in (V^n)^{\leq l_{\max}}$  requires from  $\mathcal{B}_{RKA}$  no more than  $l_{\max}$  computations and one related-key query  $(\mathbf{Csc}(K_{\mathbf{Csc}}, M), \sigma = \mathbf{sum}'_{\boxplus}(M))$ . The result of work  $\mathcal{A}$  is equal to the result of work  $\mathcal{B}_{RKA}$  and the query complexity is  $q_{\mathcal{B}} = 1 + q_{\mathcal{A}}$ .

$$\Pr\left(\mathcal{A}^{\mathbf{KH}^{(1)}(\cdot)} \Rightarrow 1\right) - \Pr\left(\mathcal{A}^{\mathbf{KH}^{(2)}(\cdot)} \Rightarrow 1\right) \leq \text{Adv}_{\mathbf{g}^\nabla}^{PRF\text{-}RKA_{\boxplus}}(\mathcal{B}_{RKA}).$$

**Step 3.** The essence of the step is contained in the following statement.

**Lemma.** *The cascade  $\mathbf{Csc}(K_{\mathbf{Csc}}, M)$ ,  $M \in (V^n)^{\leq l_{\max}}$  is itself PRF-secure provided that  $\mathbf{g}^\triangleright$  is secure in the  $PRF\text{-}RKA_{\oplus}$  model*

$$\text{Adv}_{\mathbf{Csc}}^{PRF}(t, q, l_{\max}) \leq q \cdot l_{\max} \cdot \text{Adv}_{\mathbf{g}^\triangleright}^{PRF\text{-}RKA_{\oplus}}(t', q),$$

where  $t' = t + O(q \cdot l_{\max})$ .

The inequality presented above is similar to [9, Theorem 3.1] on the PRF-security of a «plain» cascade  $\mathbf{pCsc}$  (i.e. without addition with  $\Delta_0, \dots, \tilde{\Delta}_l$ ). The differences are as follows: the relevant threat model for  $\mathbf{g}^\triangleright$  has been changed from  $PRF$  to  $PRF\text{-}RKA_{\oplus}$ ; the sequence  $\Delta_0, \dots, \Delta_{l-1}, \tilde{\Delta}_l$  is used; the prefix-free restriction is not imposed on the adversary's queries.

Obviously, the cascade isn't secure if the adversary can predict some output for non-queried input. If the value  $\mathbf{pCsc}(K, M)$  is known for some message  $M$ , then  $\mathbf{pCsc}(K, M||p)$  can also be easily computed for any block  $p$  (length extension attack). Hence, the PRF-security of  $\mathbf{pCsc}$  is proved only for the case when none of the queried messages could be a prefix of any other.

Our situation is different. The value  $\mathbf{Csc}(K, m_1 || \dots || m_{l+1})$  does not give a direct opportunity to compute  $\mathbf{Csc}(K, m_1 || \dots || m_{l+1} || p)$  due to the  $\Delta_l \neq \tilde{\Delta}_l$ . If  $m_{l+1}$  is the last block, then  $\mathbf{g}(K_{\mathbf{g}} \oplus \tilde{\Delta}_l, m_{l+1})$  is computed, otherwise we have  $\mathbf{g}(K_{\mathbf{g}} \oplus \Delta_l, m_{l+1})$ , where  $K_{\mathbf{g}}$  is some intermediate state. The calculation is performed using related keys, and the relation is equal to  $\phi_l = \tilde{\Delta}_l \oplus \Delta_l$ .

We formalize this intuition using the  $PRF-RKA_{\oplus}$  notion and give the proof in Appendix B.

Thus, the last call of the compression function with checksum mixing *is not necessary* to ensure the security of **Streebog-K** (of course, under the two assumptions about security of  $\mathbf{g}$ ).

**Step 4.** Consider a special threat model called *FINAL*

$$\begin{aligned} \text{Adv}_{\text{Csc}}^{\text{FINAL}}(\mathcal{A}) &= \Pr \left( K_{\text{Csc}} \stackrel{R}{\leftarrow} V^k, \mathcal{A} \Rightarrow (M_1, \dots, M_q), \right. \\ &\quad \exists i, j : \text{Csc}(K_{\text{Csc}}, M_i) = \text{Csc}(K_{\text{Csc}}, M_j), M_i \neq M_j \text{ OR} \\ &\quad \left. \exists i : \text{Csc}(K_{\text{Csc}}, M_i) = IV \right). \end{aligned}$$

An adversary  $\mathcal{A}$  which is effective in this model allows to construct the algorithm  $\mathcal{B}_{\text{FINAL}}$  attacking  $\text{Csc}$  in the PRF model.  $\mathcal{B}_{\text{FINAL}}$  runs the algorithm  $\mathcal{A}$  and obtains  $q$  different messages  $(M_1, \dots, M_q)$ . For each message  $\mathcal{B}_{\text{FINAL}}$  requests from its oracle the value  $Y_i = \mathbf{F}(M_i)$  (resp.  $Y_i = \text{Csc}(K_{\text{Csc}}, M_i)$ ). If  $\mathcal{B}_{\text{FINAL}}$  obtains the collision or the value  $IV$  among  $(Y_1, \dots, Y_q)$  then the result is 1, otherwise 0. Hence

$$\begin{aligned} \Pr(\mathcal{B}_{\text{FINAL}}^{\text{Csc}(K_{\text{Csc}}, \cdot)} \Rightarrow 1) &= p_0 \geq \text{Adv}_{\text{Csc}}^{\text{FINAL}}(\mathcal{A}), \\ \Pr(\mathcal{B}_{\text{FINAL}}^{\mathbf{F}(\cdot)} \Rightarrow 1) &= p_1 \leq \frac{q \cdot (q-1)}{2} \frac{1}{2^n} + \frac{q}{2^n}, \\ \text{Adv}_{\text{Csc}}^{\text{PRF}}(\mathcal{B}_{\text{FINAL}}) &= p_0 - p_1 \geq \text{Adv}_{\text{Csc}}^{\text{FINAL}}(\mathcal{A}) - \left( \frac{q \cdot (q-1)}{2} \frac{1}{2^n} + \frac{q}{2^n} \right), \\ \text{Adv}_{\text{Csc}}^{\text{FINAL}}(\mathcal{A}) &\leq \text{Adv}_{\text{Csc}}^{\text{PRF}}(\mathcal{B}_{\text{FINAL}}) + \frac{q^2 + q}{2^{n+1}}. \end{aligned}$$

The last call of the compression function in **Streebog** «mixes» checksum with the state. At the second step, this transformation was replaced by a family of random functions  $\mathbf{f}_{\overline{K} \boxplus \sigma}(\cdot) \stackrel{R}{\leftarrow} \text{Func}(V^n, V^n)$ ,  $\forall \sigma \in V^n$ . One query has already been made  $K_{\text{Csc}} = \mathbf{f}_{\overline{K} \boxplus 0}(IV)$ .

The query  $M_i$  from  $\mathcal{A}$  produces the pair of values

$$(Y_i, \sigma_i) = (\text{Csc}(K_{\text{Csc}}, M_i), \text{sum}'_{\boxplus}(M_i))$$

If there are no collisions  $(Y_i, \sigma_i) \neq (Y_j, \sigma_j)$  for all  $i \neq j$  and for all  $i$ :  $(Y_i, \sigma_i) \neq (IV, 0)$ , then  $\mathbf{f}_{\overline{K} \boxplus \cdot}(\cdot)$  is not requested twice with the same query. Thus, the result is indistinguishable from a random function.

The transformation  $\text{KH}^{(2)}$  is represented as follows.

Initialization:  $K \stackrel{R}{\leftarrow} V^k$ ;  $H'_0, H'_1, \dots, H'_q \stackrel{R}{\leftarrow} V^n$ ;  $K_{\text{Csc}} = H'_0 = \mathbf{f}_{\overline{K} \boxplus 0}(IV)$ ;

On query  $M_i$ ,  $i = 1, \dots, q$  compute:

- $Y_i = \text{Csc}(K_{\text{Csc}}, M_i); H_i = H'_i; \sigma_i = \text{sum}'_{\boxplus}(M_i);$
- (\*) **if**  $(Y_i, \sigma_i) = (Y_j, \sigma_j)$  for some  $j < i$  **then**  $H_i = H_j;$
- (\*) **if**  $(Y_i, \sigma_i) = (IV, 0)$  **then**  $H_i = K_{\text{Csc}};$
- **return**  $H_i.$

If rows marked with (\*) are not executed, then the result is indistinguishable from a random function. Delete these rows and obtain  $\text{KH}^{(3)}$ .

The probability of the conditions (\*) being true does not exceed the probability of a successful attack in the *FINAL* model on  $\text{Csc}$  (if we remove checksums, then it is essentially the same thing). Hence, by «fundamental game-playing lemma»

$$\Pr\left(\mathcal{A}^{\text{KH}^{(3)}(\cdot)} \Rightarrow 1\right) - \Pr\left(\mathcal{A}^{\text{KH}^{(2)}(\cdot)} \Rightarrow 1\right) \leq \text{Adv}_{\text{Csc}}^{\text{PRF}}(\mathcal{B}_{\text{FINAL}}) + \frac{q^2 + q}{2^{n+1}}.$$

The set of transitions presented leads to the following theorem.

**Theorem (PRF-security of Streebog-K).** *For any adversary  $\mathcal{A}$  with time complexity at most  $t$  that makes  $q$  queries, where the maximal message length is at most  $(l_{\text{max}} - 1)$  blocks, there exist the adversaries  $\mathcal{B}'$  and  $\mathcal{B}''$  such that*

$$\text{Adv}_{\text{KH}}^{\text{PRF}}(\mathcal{A}) \leq \text{Adv}_{\text{g}^{\nabla}}^{\text{PRF-RKA}_{\boxplus}}(\mathcal{B}') + q \cdot l_{\text{max}} \cdot \text{Adv}_{\text{g}^{\triangleright}}^{\text{PRF-RKA}_{\oplus}}(\mathcal{B}'') + \frac{q^2 + q}{2^{n+1}}.$$

The query complexity of  $\mathcal{B}'$  and  $\mathcal{B}''$  is  $q + 1$  and  $q$  correspondingly. The time complexity of both adversaries is  $t' = t + O(q l_{\text{max}})$ .

Assuming  $t \gg q \cdot l_{\text{max}}$  and with the estimates of  $\text{Adv}_{\text{g}^{\nabla}}^{\text{PRF-RKA}_{\boxplus}}$  and  $\text{Adv}_{\text{g}^{\triangleright}}^{\text{PRF-RKA}_{\oplus}}$  based on generic attacks

$$\text{Adv}_{\text{KH}}^{\text{PRF}}(t, q, l_{\text{max}}) \lesssim \frac{t \cdot q}{2^{k-1}} + \frac{t \cdot q \cdot l_{\text{max}}}{2^{n-1}} + \frac{q^3 \cdot l_{\text{max}}}{2^n}.$$

It should be noted that the bound presented in the theorem almost coincides with the corresponding one in HMAC and can be considered tight in some sense (see, for example [15]). At the same time, the approximate estimate, which was given for illustrative purposes, is not accurate and significantly exaggerates the capabilities of the adversary. Despite this, **Streebog-K** can be used in practice *without any restrictions* on the amount of data processed. Of course, the presented estimates do not consider threats that are outside the formal model, e.g. side-channel attacks and others.

For example, let **Streebog-K** be used as MAC with 256-bit key. The output is truncated to  $\tau = 64$  bits,  $q = 2^{48}$  messages are processed with one key,

each message has a length of no more than  $l_{\max} = 2^{64}$  blocks. The computing power of the adversary is about  $t = 2^{128}$  operations. Hence, the probability of creating a forgery in one attempt is bounded by [12, Proposition 7.3] (SUF – Strong UnForgeability)

$$\text{Adv}_{\text{KH}}^{\text{SUF}}(t, q, l_{\max}) \leq \text{Adv}_{\text{KH}}^{\text{PRF}}(t, q, l_{\max}) + \frac{1}{2^\tau} \lesssim 2^{-63},$$

and the numerical value is close to the ideal  $2^{-\tau} = 2^{-64}$ .

## 6 Beyond the bound attacks

To complete the description of the properties and features of **Streebog-K**, we briefly present two attacks on it. Once again, we note that attacks have a significant probability of success only if the amount of material and computing resources of the adversary is greater than allowed according to the provable security bounds.

### 6.1 Existential forgery

The attack is carried out under the conditions of the adaptively chosen messages.

The set of  $l$ -block messages  $M_i = \text{bin}(i) \parallel \text{bin}(2^n - i) \parallel C$ ,  $i = 1, \dots, q$  is prepared, where  $C$  contains arbitrary blocks. The checksums of all messages are the same  $\text{sum}_{\boxplus}(M_i) = \text{sum}_{\boxplus}(M_j)$ ,  $1 \leq i, j \leq q$ .

The oracle is queried for the values of  $H_i = \text{KH}(K, M_i)$ .

For simplicity, we assume that after processing of the two first blocks  $\text{bin}(i) \parallel \text{bin}(2^n - i)$ , all intermediate states are different. Further transformations will be *identical* for each message.

Assuming that when processing the  $j$ -th block ( $j = 3, \dots, l$ ), a random mapping is applied to each intermediate state in parallel, the probability of a collision for an arbitrary pair is estimated by  $\Pr(H_i = H_j, i \neq j) = \Theta(l \cdot 2^{-n})$ , see [23, Lemma 1]. Then the probability of a collision among  $q$  messages  $\Pr(\exists i, j : H_i = H_j, i \neq j) = \Theta(q^2 \cdot l \cdot 2^{-n})$ .

A collision generated by a pair of messages  $M_i, M_j$  most likely occurs when processing one of the message blocks, and not when finalizing. Hence, we have

$$\text{KH}(K, M_i \parallel P) = \text{KH}(K, M_j \parallel P), \quad P \in V^n.$$

The adversary uses the possibility of adaptive setting, obtains  $H_{q+1} = \text{KH}(K, M_i \parallel P)$  and creates the forgery  $M_j \parallel P$  with the code  $H_{q+1}$ .

The attack is the same for both **Streebog-K** and **HMAC-Streebog**.

## 6.2 Key recovery

In [23], among other things, the key-recovery attack against **HMAC-Streebog** (with 512-bit key) was presented. The time complexity is at least  $t = 2^{419}$  operations.

The attack consists of two phases. Both of them have almost the same time complexity.

The first phase is the state-recovery attack. This method is *generic* for HMAC with HAIFA-like [21] hash function. The *optimal* time complexity is about  $t = 2^{419}$  operations. The oracle is queried about  $q = 2^{358}$  times. The length of each query is at least  $l = 2^{51}$  blocks. Other values of the  $q$  and  $l$  will result in more time complexity.

The target of the second phase is the secret key. This part of the attack can only be applied to **HMAC-Streebog** and similar cryptoalgorithms.

The state recovery attack can be used against **Streebog-K** without any modification. So, we omit its description and refer to [23]. As a result of the first phase, the adversary obtains the  $l$ -block message  $M$  and the corresponding secret state  $x$  (after processing the message and before finalization).

Key recovery phase is much easier for **Streebog-K**.

The adversary constructs  $2^u$ -collision starting with the state  $x$  (the time complexity is about  $u \cdot 2^{n/2}$  operations [20]). The value of the multicollision is  $x^* = \text{Csc}(K_{\text{Csc}}, M || P_i || 10..0 || \text{bin}(L))$  and  $P_i$  contains exactly  $u$  blocks,  $i = 1, \dots, 2^u$ . By queries to the oracle the values  $H_i = \text{KH}(\bar{K}, M || P_i)$  are collected and  $\mathbf{g}(x^*, \bar{K} \boxplus \sigma_i) = H_i$ , where  $\sigma_i = \text{sum}_{\boxplus}(M || P_i || 10..0)$  (see fig. 3). We assume that almost all  $\sigma_i$  are different and the same is true for  $H_i$ .

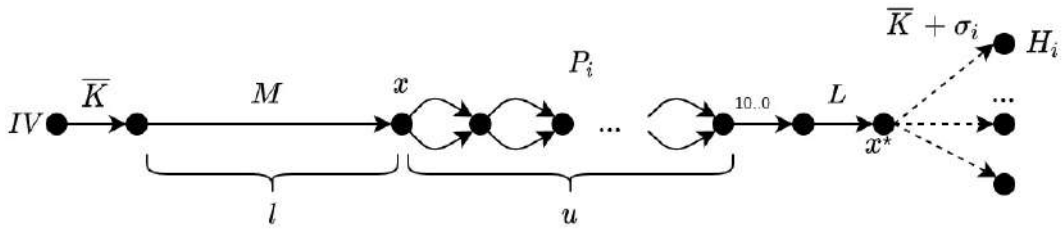


Figure 3: Key recovery attack (with known state  $x$ ).

Thus, we need to guess  $z_i = \bar{K} \boxplus \sigma_i$  for some  $i$ . We compute  $\mathbf{g}(x^*, \tilde{z}_j) = \tilde{H}_j$  and check the match  $\tilde{H}_j = H_i$ ,  $j = 1, \dots, 2^v$ . If  $\tilde{H}_j = H_i$  is true then  $\bar{K} \boxplus \sigma_i = \tilde{z}_j$  can also be true with high probability. If  $2^u \cdot 2^v = 2^n$ , we expect one true match to be found.

The time complexity of the key recovery phase is  $t \approx l \cdot 2^u + u \cdot 2^{n/2} + 2^v$  and with  $u = 230$ ,  $l = 2^{51}$ ,  $t \approx 2^{283}$ , the query complexity is  $q = 2^u$ . Consequently,

the complexity of the first phase is much greater than the second.

Thus, **Streebog-K** does not provide «512-bit security» in the sense of resistance to the key recovery. This is also true for **HMAC-Streebog**. We suggest using 256-bit keys in **Streebog-K**.

## 7 HMAC, S3G and GOST94

The obtained security proof for **Streebog-K** can be used with some modifications for a number of similar cryptoalgorithms.

**HMAC-Streebog** uses the key *four* times. The relation between keys is defined by two operations simultaneously. In the second step of the proof, the *stronger* model is used instead of  $PRF-RKA_{\oplus, \boxplus}$  (problem 2)

$$\begin{aligned} \text{Adv}_{\mathbf{g}^\nabla}^{PRF-RKA_{\oplus, \boxplus}}(\mathcal{A}) &= \Pr \left( K \stackrel{\mathbb{R}}{\leftarrow} V^k; \mathcal{A}^{\mathbf{g}(\cdot, (\overline{K} \oplus \cdot) \boxplus \cdot)} \Rightarrow 1 \right) - \\ &- \Pr \left( K \stackrel{\mathbb{R}}{\leftarrow} V^k; \mathbf{f}_i \stackrel{\mathbb{R}}{\leftarrow} \text{Func}(V^n, V^n), \forall i \in V^n; \mathcal{A}^{\mathbf{f}(\overline{K} \oplus \cdot) \boxplus (\cdot)} \Rightarrow 1 \right), \end{aligned}$$

the query is  $(m, \phi, \sigma)$ , the response is  $y = \mathbf{g}(m, (\overline{K} \oplus \phi) \boxplus \sigma)$ . The heuristic estimate of complexity for  $PRF-RKA_{\oplus, \boxplus}$  is the same as for  $PRF-RKA_{\boxplus}$ . Problem 1 and the rest of the steps remain unchanged, but one more step is added. The collision after the first call of the hash function  $\mathbf{H}((\overline{K} \oplus \text{ipad}) || M)$  is taken into account, as well as the collision between the last related key and the other three. The result is the following theorem (the proof is presented in Appendix).

**Theorem (PRF-security of HMAC-Streebog).** *For any adversary  $\mathcal{A}$  with time complexity at most  $t$  that makes  $q$  queries, where the maximal message length is at most  $(l_{\max} - 1)$  blocks, there exist adversaries  $\mathcal{B}'$  and  $\mathcal{B}''$  such that*

$$\begin{aligned} \text{Adv}_{\text{HMAC-Streebog}}^{PRF}(\mathcal{A}) &\leq \text{Adv}_{\mathbf{g}^\nabla}^{PRF-RKA_{\oplus, \boxplus}}(\mathcal{B}') + \\ &+ q \cdot l_{\max} \cdot \text{Adv}_{\mathbf{g}^\triangleright}^{PRF-RKA_{\oplus}}(\mathcal{B}'') + \frac{q^2 + q}{2^{n+1}} + \frac{3(q^2 + q)}{2^{\tau+1}}, \end{aligned}$$

where  $\tau \in \{256, 512\}$  is the bit length of the output. The query complexity of  $\mathcal{B}'$  and  $\mathcal{B}''$  is  $2 + 2 \cdot q$  and  $q$  correspondingly. The time complexity of both adversaries is  $t' = t + O(q l_{\max})$ .

Cryptoalgorithm **S3G** [4] is defined as  $\text{S3G}(K, M) = \mathbf{H}(K || M)$ , but the  $k$ -bit key is not padded to 512 bits,  $k \in \{128, 256\}$ . The number of calls to  $\mathbf{g}$  is always the same for the selected  $k$ . Yet another variant of the compression function is defined as  $\mathbf{g}_K^{\nabla\nabla}(m, m') = \mathbf{g}(m, K || m')$ ,  $m' \in V^{n-k}$  with the

corresponding threat model

$$\begin{aligned} \text{Adv}_{\mathbf{g}_{\nabla\nabla}}^{\text{PRF-RKA}_{\boxplus, \boxparallel}}(\mathcal{A}) &= \Pr\left(K \stackrel{\text{R}}{\leftarrow} V^k; \mathcal{A}^{\mathbf{g}(\cdot, (K||\cdot)_{\boxplus})} \Rightarrow 1\right) - \\ &- \Pr\left(K \stackrel{\text{R}}{\leftarrow} V^k; \mathbf{f}_i \stackrel{\text{R}}{\leftarrow} \text{Func}(V^n, V^n), \forall i \in V^n; \mathcal{A}^{\mathbf{f}(K||\cdot)_{\boxplus}(\cdot)} \Rightarrow 1\right), \end{aligned}$$

the query is  $(m, m', \sigma)$ , the response is  $y = \mathbf{g}(m, (K||m')_{\boxplus}) \boxplus \sigma$ . The second step of the proof changes accordingly. In the third step, the tree does not have a single root, the number of roots is arbitrary from 1 to  $q$ . The fourth step of the proof does not require significant changes.

Hash function **GOST94** [2] is based on the «plain» Merkle-Damgård scheme (the state and the message size  $n = 256$  bit) with one exception: the last call of the compression function  $\mathbf{g}_{94}$  «mixes» checksum of the message (modulo  $2^n$ ) to the state. The first step of the security proof **HMAC-GOST94** should take into account changes in the padding. The second step uses the same reduction as for **HMAC-Streebog** ( $\mathbf{g}_{94}^{\nabla}$  must be secure in the  $\text{PRF-RKA}_{\boxplus, \boxparallel}$  model). The third step is simply using the previously known result [9, Th. 3.1] ( $\mathbf{g}_{94}^{\triangleright}$  must be a secure PRF). At the fourth step we use a well-known «extension trick» as in the proof of HMAC [14, Sec. 7]. The last step is the same as for **HMAC-Streebog**. However, it should be noted that the security of  $\mathbf{g}_{94}$ , as far as we know, was not examined in the PRF model, unlike  $\mathbf{g}$  [24, 25]. Therefore, in the case of **HMAC-GOST94**, we cannot say without a doubt that the basic problems are really computationally hard.

## 8 Conclusion

The paper presents «**Streebog-K**» («Keyed Streebog»)

$$\text{KH}(K, M) = \text{H}(\overline{K}||M), \quad \overline{K} = K||0\dots 0,$$

based on keyless hash function **Streebog**. The proposed solution has almost the same cryptographic strength as **HMAC-Streebog**. This is true both from the provable security point of view, and with regard to the applicability of attacks. At the same time, the speed is doubled when processing short texts.

The suggested proof shows that **Streebog-K** is a pseudorandom function (PRF) and, therefore, a secure message authentication code (MAC). The security is reduced to the resistance of the underlying compression function to the related key attacks (PRF-RKA). The existing results indicate that the compression function is indeed secure in the relevant threat models.

The obtained results can be slightly modified to create security proofs of **HMAC-Streebog**, **HMAC-GOST94**, **S3G** and similar cryptoalgorithms. Such changes were also briefly listed.



We also propose two open problems to consider:

- 1) Is it possible to replace problem 1 ( $PRF-RKA_{\oplus}$ ) in the reduction with the simple PRF model?
- 2) Is there an attack in any model that would be more effective against Streebog-K than against HMAC-Streebog?

## 9 Acknowledgements

The author sincerely thanks Evgeny Alekseev, Liliya Akhmetzyanova and Alexandra Babueva for the inspiration, motivation and useful discussions. The quality of the paper has been significantly improved thanks to the great help of Andrey Scherbachenko. Special thanks to the anonymous reviewer(s) for the thorough verification of the result and a lot of detailed comments and suggestions.

## References

- [1] *GOST R 34.11-2012 – National standard of the Russian Federation – Information technology – Cryptographic data security – Hash function*, 2012.
- [2] *GOST R 34.11-94 – National standard of the Russian Federation – Information technology – Cryptographic data security – Hash function*, 1994.
- [3] *R 50.1.113-2016 – Information technology – Cryptographic data security – Cryptographic algorithms accompanying the use of electronic digital signature algorithms and hash functions*, 2016.
- [4] *R 1323565.1.003-2017 – Information technology – Cryptographic data security – Cryptographic algorithms for generating encryption keys and authentication vectors intended for implementation in hardware trust modules for use in mobile communication*, 2017.
- [5] Grebnev S., Dmukh A., Dygin D., Matyukhin D., Rudskoy V., Shishkin V., “Asymmetrical Reply to SHA-3: Russian Hash Function Draft Standard”, *CTCrypt 2012*, 2012.
- [6] Smyshlyaev S., Alekseev E., Oshkin I., Popov V., Leontiev S., PodobaeV., Belyavsky D., “RFC 7836 - Guidelines on the Cryptographic Algorithms to Accompany the Usage of Standards GOST R 34.10-2012 and GOST R 34.11-2012”, March 2016.
- [7] Damgård I., “A Design Principle for Hash Functions”, *LNCS*, CRYPTO 1989, **435**, ed. Brassar G., Springer, Heidelberg, 1990, 416–427.
- [8] Merkle R., “One way Hash Functions and DES”, *LNCS*, CRYPTO 1989, **435**, ed. Brassard G., Springer, Heidelberg, 1990, 428–446.
- [9] Bellare M., Canetti R., Krawczyk H., “Pseudorandom Functions Revisited: The Cascade Construction and its Concrete Security”, *Proceedings of 37th Conference on Foundations of Computer Science*, 1996, 514–523.
- [10] Bellare M., Canetti R., Krawczyk H., “Keying Hash Functions for Message Authentication”, *LNCS*, *Advances in Cryptology – Crypto’96*, **1109**, ed. Koblitz N., Springer, Berlin, Heidelberg, 1996, 1–15.
- [11] Preneel B., Paul C. van Oorschot, “On the Security of Iterated Message Authentication Codes”, *IEEE Transactions on Information Theory*, **45** (1999), 188–199.
- [12] Bellare M., Goldreich O., Mityagin A., “The power of verification queries in message authentication and authenticated encryption”, *Cryptology ePrint Archive: Report 2004/304*, 2004.
- [13] Koblitz N., Menezes A., “Another look at HMAC”, *J. Math. Cryptol.*, **7:3**, 2013, 225–251.

- 
- [14] Bellare M., “New Proofs for NMAC and HMAC: Security without Collision-Resistance”, *LNCS*, Advances in Cryptology – CRYPTO 2006, **4117**, ed. Dwork C., Springer, Berlin, Heidelberg, April 2006.
- [15] Gaži P., Pietrzak K., Rybár M., “The Exact PRF-Security of NMAC and HMAC”, *LNCS*, Advances in Cryptology – CRYPTO 2014, **8616**, Springer, Berlin, Heidelberg, August 2014, 113–130.
- [16] Alekseev E.K., Oshkin I.B., Popov V.O., Smyshlyaev S.V., “On the cryptographic properties of algorithms accompanying the applications of standards GOST R 34.11-2012 and GOST R 34.10-2012”, *Mat. Vopr. Kriptogr.*, **7:1** (2016), 5–38.
- [17] Nandi M., “A New and Improved Reduction Proof of Cascade PRF”, *Cryptology ePrint Archive: Report 2021/097*, 2021.
- [18] Bellare M., Rogaway P., “Introduction to Modern Cryptography”, 2005.
- [19] Goldreich O., “Foundations of Cryptography. Basic Tools”, 2004.
- [20] Joux A., “Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions”, Advances in Cryptology – CRYPTO 2004, 2004, 306–316.
- [21] Biham E., Dunkelman O., “A Framework for Iterative Hash Functions (HAIFA)”, *Cryptology ePrint Archive, Report 2007/278*, 2007.
- [22] Guo J., Jean J., Leurent G., Peyrin T., Wang L., “The Usage of Counter Revisited: Second-Preimage Attack on New Russian Standardized Hash Function”, *LNCS*, Selected Areas in Cryptography – SAC 2014, **8781**, ed. Joux A., Youssef A., Springer, Cham, 2014.
- [23] Dinur I., Leurent G., “Improved Generic Attacks Against Hash-based MACs and HAIFA”, *LNCS*, Advances in Cryptology – CRYPTO 2014, **8616**, ed. Garay J.A., Gennaro R., Springer, Berlin, Heidelberg, 2014.
- [24] Abdelkhalek A., AlTawy R., Youssef A. M., “Impossible Differential Properties of Reduced Round Streebog”, *LNCS*, Codes, Cryptology, and Information Security. C2SI 2015, **9084**, ed. El Hajji S., Nitaj A., Carlet C., Souidi E., Springer, Cham, 2015, 274–286.
- [25] Kiryukhin V., “Streebog compression function as PRF in secret-key settings”, *CTCrypt 2021*, 2021.
- [26] AlTawy R., Youssef A. M., “Preimage Attacks on reduced-round Stribog”, *LNCS*, Progress in Cryptology – AFRICACRYPT 2014, **8469**, ed. Pointcheval D., Vergnaud D., Springer, Cham, 2014.
- [27] AlTawy R., Kircanski A., Youssef A. M., “Rebound attacks on Stribog”, *LNCS*, Information Security and Cryptology – ICISC 2013, **8565**, ed. Lee HS., Han DG., Springer, Cham, 2014.
- [28] Lin D., Xu S., Yung M., “Cryptanalysis of the Round-Reduced GOST Hash Function”, *LNCS*, Information Security and Cryptology. Inscrypt 2013., **8567**, Springer, Cham, 2014.
- [29] Ma B., Li B., Hao R., Li X., “Improved cryptanalysis on reduced-round GOST and Whirlpool hash function”, *LNCS*, Applied Cryptography and Network Security. ACNS 2014., **8479**, ed. Boureanu I., Owesarski P., Vaudenay S., Springer, Cham, 2014.
- [30] Wang Z., Yu H., Wang X., “Cryptanalysis of GOST R Hash Function”, *Information Processing Letters*, **114** (2014), 655–662.
- [31] Kölbl S., Rechberger C., “Practical Attacks on AES-like Cryptographic Hash Functions”, *LNCS*, Progress in Cryptology – LATINCRYPT 2014, **8895**, ed. Aranha D., Menezes A., Springer, Cham, 2014.
- [32] Ma B., Li B., Hao R., Li X., “Improved (Pseudo) Preimage Attacks on Reduced-Round GOST and Grøstl-256 and Studies on Several Truncation Patterns for AES-like Compression Functions”, *LNCS*, Advances in Information and Computer Security. IWSEC 2015, **9241**, ed. Tanaka K., Suga Y., Springer, Cham, 2015, 79–96.

## A Detailed pictures

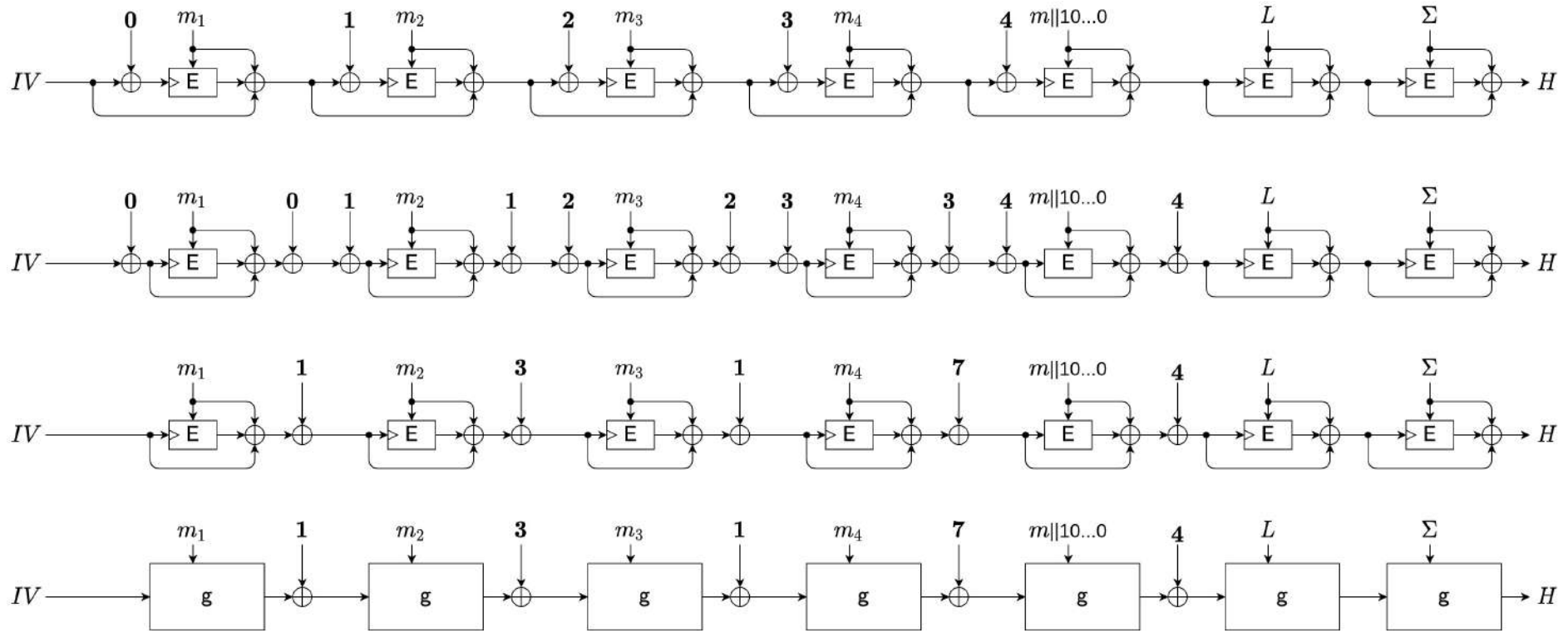


Figure 4: The equivalent representation of Streebog.

## B Proof of the lemma

**Lemma.** *The cascade*

$$\text{Csc}(K_{\text{Csc}}, M) = \mathbf{g}(\dots \mathbf{g}(K_{\text{Csc}} \oplus \Delta_0, m_1) \dots \oplus \tilde{\Delta}_l, m_{l+1}),$$

where  $M = m_1 || m_2 || \dots || m_{l+1} \in (V^n)^{\leq l_{\max}}$ ,  $1 \leq l+1 \leq l_{\max}$ , is PRF-secure provided that  $\mathbf{g}^\triangleright$  is secure in the PRF-RKA $_{\oplus}$  model

$$\text{Adv}_{\text{Csc}}^{\text{PRF}}(t, q, l_{\max}) \leq q \cdot l_{\max} \cdot \text{Adv}_{\mathbf{g}^\triangleright}^{\text{PRF-RKA}_{\oplus}}(t', q),$$

where  $t' = t + O(q \cdot l_{\max})$ .

*Proof.* Let's imagine queries to the cascade in the form of a tree:

the root  $v_0$  is  $K_{\text{Csc}}$ ; the nodes  $v_i$  are the intermediate states; the results are stored in leaves. Each edge of the tree is labeled with the the block  $m_i$  from the message  $M$

$$K_{\text{Csc}} = v_0 \xrightarrow{m_1} v_1 \xrightarrow{m_2} v_2 \xrightarrow{m_3} v_3 \dots \xrightarrow{m_{l+1}} v_{l+1}, \quad 1 \leq l+1 \leq l_{\max}.$$

At each level (height) after processing all requests, there will be no more than  $q$  nodes.

Consider an arbitrary node  $v_i$ , which is essentially an intermediate secret key. If  $m_{i+1}$  is not the last, then  $\mathbf{g}(v_i \oplus \Delta_i, m_{i+1}) = v_{i+1}$  is computed. If the block  $m'_{i+1}$  is the last, then  $\tilde{\Delta}_i$  is used  $\mathbf{g}(v_i \oplus \tilde{\Delta}_i, m'_{i+1}) = v'_{i+1}$ . The first secret key is  $K_{\mathbf{g}} = v_i \oplus \Delta_i$ , the second one is  $K'_{\mathbf{g}} = v_i \oplus \tilde{\Delta}_i$ . The relation between the keys is defined as

$$\phi_i = K_{\mathbf{g}} \oplus K'_{\mathbf{g}} = \Delta_i \oplus \tilde{\Delta}_i = (\mathbf{i} \oplus (\mathbf{i} \boxplus 1)) \oplus \mathbf{i} = \mathbf{i} \boxplus 1, \quad i = 0, \dots, l_{\max}.$$

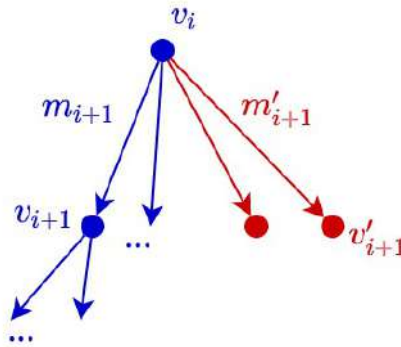


Figure 5: Node  $v_i$  of the tree. The internal (resp. external) edges and nodes are highlighted in blue (resp. red).

The adversary will never observe the values of the internal nodes of the tree. Hence, all adversary's queries will be *independent* of these values. For a

«plain» cascade from [9], this was ensured by limiting the queries (none of the queried message could be a prefix of any other). In our case, the mentioned independence is provided essentially by two different functions that compute internal and external nodes, respectively.

We use a «hybrid argument» for tree levels (from 1 to  $l_{\max}$ ) and for nodes of each level (from 1 to  $q$ ).

Denote by  $\mathbf{Csc}_i$  the cascade transformation starting from level  $i$ ,  $\mathbf{Csc}_0 = \mathbf{Csc}$ . At level  $i$ , we define the hybrid game and the corresponding oracle  $\mathcal{C}_i$  as follows:

- Initialization:  $F \stackrel{R}{\leftarrow} \text{Func}((V^n)^i, V^n)$ ;  $F' \stackrel{R}{\leftarrow} \text{Func}((V^n)^{\leq i}, V^n)$ ;
- On query  $M = (m_1, \dots, m_l) \in (V^n)^l$ ,  $1 \leq l \leq l_{\max}$  from  $\mathcal{A}$  compute:
  - if**  $l \leq i$  **then**  $y = F'(M)$ ; **return**  $y$ ;
  - if**  $l > i$  **then**  $M_{pre} = (m_1, \dots, m_i)$ ;  $M_{suff} = (m_{i+1}, \dots, m_l)$ ;
  - return**  $y = \mathbf{Csc}_i(F(M_{pre}), M_{suff})$ .

All internal (resp. external) nodes of the tree are calculated using  $F$  (resp.  $F'$ ). The oracle  $\mathcal{C}_0(\cdot)$  is identical to the  $\mathbf{Csc}(K_{\mathbf{Csc}}, \cdot)$ . Indeed, if  $i = 0$  then  $M_{pre}$  is an empty string,  $M_{suff} = M$ ,  $F(M_{pre}) = K_{\mathbf{Csc}}$  and  $y = \mathbf{Csc}_0(K_{\mathbf{Csc}}, M)$ . The algorithm  $\mathcal{C}_{l_{\max}}(\cdot)$  is essentially a random function

$$\text{Adv}_{\mathbf{Csc}}^{PRF}(\mathcal{A}) = \Pr\left(\mathcal{A}^{\mathcal{C}_{l_{\max}}(\cdot)} \Rightarrow 1\right) - \Pr\left(\mathcal{A}^{\mathcal{C}_0(\cdot)} \Rightarrow 1\right).$$

Let  $\mathcal{A}$  be able to effectively distinguish  $\mathcal{C}_0(\cdot)$  and  $\mathcal{C}_1(\cdot)$ , then it is possible to construct  $\mathcal{B}_0$  distinguishing the compression function from the random function in the  $PRF\text{-}RKA_{\oplus}$  model. Really, for the one-block message  $M = m_1$  algorithm  $\mathcal{B}_0$  queries the value  $f'(m_1)$  (this is either the value of the second random function, or  $\mathbf{g}(K \oplus \phi_1, m_1)$ ). For any multi-block query  $M = (m_1, m_2, \dots, m_l)$  received from  $\mathcal{A}$ , algorithm  $\mathcal{B}_0$  asks its oracle for the value  $\mathbf{f}(m_1)$  (resp.  $\mathbf{g}(K, m_1)$ ). Recall that the secret random key  $K$  is essentially the value  $K = K_{\mathbf{Csc}} \oplus \Delta_0$  (also distributed uniformly on  $V^n$ ), and therefore,  $\mathcal{B}_0$  correctly emulates the beginning of the cascade. Next, the value  $\mathbf{Csc}_1(\mathbf{f}(m_1), (m_2, \dots, m_l))$  is computed and sent to  $\mathcal{A}$  without queries to the oracle. The result of  $\mathcal{B}_0$  is equal to the result of  $\mathcal{A}$  and

$$\Pr\left(\mathcal{A}^{\mathcal{C}_1(\cdot)} \Rightarrow 1\right) - \Pr\left(\mathcal{A}^{\mathcal{C}_0(\cdot)} \Rightarrow 1\right) \leq \text{Adv}_{\mathbf{g}^b}^{PRF\text{-}RKA_{\oplus}}(\mathcal{B}_0).$$

Consider the general case. Let  $\mathcal{A}$  be able to effectively distinguish  $\mathcal{C}_i(\cdot)$  and  $\mathcal{C}_{i-1}(\cdot)$  (figure 6).

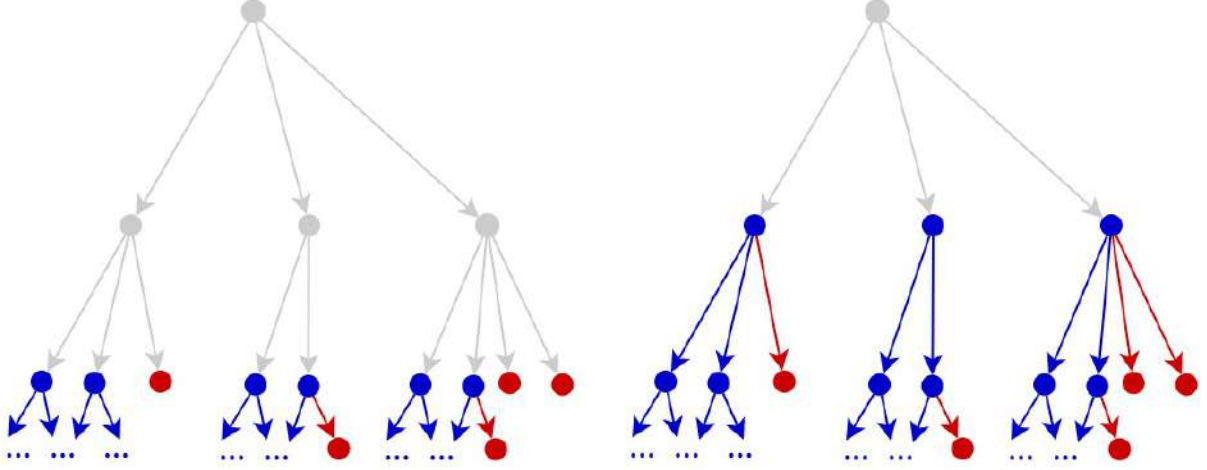


Figure 6: Trees formed by queries to  $\mathcal{C}_i(\cdot)$  (left) and  $\mathcal{C}_{i-1}(\cdot)$  (right).

We turn to the case of  $q$  parallel games in the  $PRF\text{-}RKA_{\oplus}$ . The corresponding adversary  $\mathcal{B}_i$  has access to  $q$  pairs of oracles ( $q$  pairs of random functions  $(f_j(\cdot), f'_j(\cdot))$  or  $q$  pairs of  $(\mathbf{g}_{K_j}^{\triangleright}(\cdot), \mathbf{g}_{K_j \oplus \phi_i}^{\triangleright}(\cdot))$ ,  $j = 1, \dots, q$ ). The query from  $\mathcal{B}_i$  consists of  $(j, b \in \{1, 2\}, m \in V^n)$ , where  $b$  specifies the first or the second oracle of the  $j$ -th pair. Due to the independence of the pairs, the hybrid argument is straightforward, and we have

$$\Pr\left(\mathcal{B}_i^{\mathbf{f}(\cdot), \mathbf{f}'(\cdot)} \Rightarrow 1\right) - \Pr\left(\mathcal{B}_i^{\mathbf{g}_{K_j}^{\triangleright}(\cdot), \mathbf{g}_{K_j \oplus \phi_i}^{\triangleright}(\cdot)} \Rightarrow 1\right) \leq \sum_{j=1}^q \text{Adv}_{\mathbf{g}^{\triangleright}}^{PRF\text{-}RKA_{\oplus}}(\mathcal{B}_{i,j}).$$

The value of  $\phi_i = \mathbf{i} \boxplus 1$  is chosen equally by all  $PRF\text{-}RKA_{\oplus}$ -adversaries  $\mathcal{B}_{i,j}$ ,  $j = 1, \dots, q$ . The algorithm of  $\mathcal{B}_{i,j}$  is similar to that of  $\mathcal{B}_0$ . In fact, the latter processes queries that affect the root of the tree, and  $\mathcal{B}_{i,j}$  uses messages that depend on the  $j$ -th node at depth  $i$ .

$M = (m_1, \dots, m_l)$  is the query from  $\mathcal{A}$  to the oracle  $\mathcal{C}_i(\cdot)$  or  $\mathcal{C}_{i-1}(\cdot)$ . The algorithm  $\mathcal{B}_i$  must perfectly simulate both of them:

- Initialization:  $\text{PrefixMap}[P] = \emptyset$ ,  $\forall P \in (V^n)^{i-1}$ ;  $\max_j = 1$ ;
- On query  $M = (m_1, m_2, \dots, m_l) \in (V^n)^l$ ,  $1 \leq l \leq l_{\max}$  from  $\mathcal{A}$  compute:
  - **if**  $l < i$  **then return**  $y \xleftarrow{R} V^n$ ; (recall that there are no duplicate queries  $M$ );
  - **if**  $\text{PrefixMap}[(m_1, \dots, m_{i-1})] = \emptyset$  **then**  
 $\text{PrefixMap}[(m_1, \dots, m_{i-1})] = \max_j$ ;  
 $\max_j = \max_j + 1$ ;
  - $j = \text{PrefixMap}[(m_1, \dots, m_{i-1})]$ ;

- **if**  $l = i$  **then**  $y = f'_j(m_i)$ ; (resp.  $y = \mathbf{g}_{K_j \oplus \phi_i}^\triangleright(m_i)$ ) by query  $(j, 2, m_i)$ ;  
**return**  $y$ ;
- **if**  $l > i$  **then**  $z = f_j(m_i)$ ; (resp.  $z = \mathbf{g}_{K_j}^\triangleright(m_i)$ ) by query  $(j, 1, m_i)$ ;  
 $M_{suff} = (m_{i+1}, \dots, m_l)$ ;  
 $y = \mathbf{Csc}_i(z, M_{suff})$ ; **return**  $y$ .

First of all, we note that requests shorter than  $l$  blocks always generate a random response (both  $\mathcal{C}_i$  and  $\mathcal{B}_i$ ). Different prefixes  $(m_1, \dots, m_{i-1})$  generate queries to different oracles. PrefixMap is used to store all queried prefixes  $(m_1, \dots, m_{i-1})$ . Initially, there is not a single entry in PrefixMap. If the prefix has not been queried before, a new entry is created in PrefixMap, otherwise, the  $j$  corresponding to the prefix is extracted. After all interactions, we have  $1 \leq \max_j \leq q$ . In other words, PrefixMap stores at least one and at most  $q$  elements. If  $\max_j = 1$ , then all queries had the same prefix. If  $\max_j = q$ , then the prefixes of all queries were different.

Let  $\mathcal{B}_i$  interact with  $q$  pairs of random functions. Therefore, if  $l = i$  then the response is really random (as  $y = \mathbf{F}'(M)$  in the case of  $\mathcal{C}_i$ ). If  $l > i$  then it is also truly random (as the intermediate value  $\mathbf{F}(M_{pre})$ ). Further computation of the cascade is identical in both cases. So,  $\mathcal{B}_i$  simulates  $\mathcal{C}_i(\cdot)$  for the adversary  $\mathcal{A}$ .

Let  $\mathcal{B}_i$  interact with  $q$  pairs of the compression functions  $(\mathbf{g}_{K_j}^\triangleright(\cdot), \mathbf{g}_{K_j \oplus \phi_i}^\triangleright(\cdot))$ . Imagine that  $M_{pre} = (m_1, \dots, m_{i-1})$  and instead of requesting a random function  $\mathbf{F}(M_{pre})$ , we implicitly use secret keys from the  $j$ -th pair of oracles. Next, the computations  $y = \mathbf{Csc}_i(\mathbf{g}(K_j, m_i), (m_{i+1}, \dots, m_l))$  are equivalent to the  $\mathbf{Csc}_{i-1}$  cascade, and the perfect simulation of  $\mathcal{C}_{i-1}(\cdot)$  is also constructed.

Thus, we consistently replace  $\mathcal{C}_{i-1}(\cdot)$  with  $\mathcal{C}_i(\cdot)$  ( $i = 1, \dots, l_{\max}$ ), and summing up the advantages, we obtain the statement of the lemma

$$\begin{aligned} \text{Adv}_{\mathbf{Csc}}^{PRF}(\mathcal{A}) &\leq \sum_{i=1}^{l_{\max}} \left( \Pr \left( \mathcal{A}^{\mathcal{C}_i(\cdot)} \Rightarrow 1 \right) - \Pr \left( \mathcal{A}^{\mathcal{C}_{i-1}(\cdot)} \Rightarrow 1 \right) \right) \leq \\ &\leq \sum_{i=1}^{l_{\max}} \sum_{j=1}^q \text{Adv}_{\mathbf{g}^\triangleright}^{PRF-RKA_\oplus}(\mathcal{B}_{i,j}). \end{aligned}$$

## C Adaptation of the proof for HMAC-Streebog

Recall that the HMAC is represented as

$$\text{HMAC}(K, M) = \text{H}((\overline{K} \oplus \text{opad}) || \text{H}(\overline{K} \oplus \text{ipad} || M)),$$

where  $\text{ipad}, \text{opad} \in V^n$ ,  $\text{ipad} \neq \text{opad}$ ,  $\overline{K} = (K || 0 \dots 0) \in V^n$ . We consider the case when  $\text{H}$  is **Streebog-256** or **Streebog-512** (fig. 7 and 8).

Denote by  $\tau \in \{256, 512\}$  the bit length of the hash function output. The intermediate output is  $H^I = \text{H}(\overline{K} \oplus \text{ipad} || M) \in V^\tau$  and the keys for cascades are  $K_{\text{Csc}}^I = \mathbf{g}(IV, \overline{K} \oplus \text{ipad})$ ,  $K_{\text{Csc}}^O = \mathbf{g}(IV, \overline{K} \oplus \text{opad})$ .

The proof of the PRF-security for **HMAC-Streebog** is similar to the corresponding one for **Streebog-K**. Next, we describe the changes.

The **first step** remains the same. We proceed to the analysis when the message consists of  $n$ -bit blocks ( $\text{HMAC}^{(1)}$ ).

In the **second step**, the *stronger* model is used instead of  $\text{PRF-RKA}_{\oplus, \boxplus}$  (problem 2)

$$\begin{aligned} \text{Adv}_{\mathbf{g}^\nabla}^{\text{PRF-RKA}_{\oplus, \boxplus}}(\mathcal{A}) &= \Pr\left(K \stackrel{\text{R}}{\leftarrow} V^k; \mathcal{A}^{\mathbf{g}(\cdot, (\overline{K} \oplus \cdot) \boxplus \cdot)} \Rightarrow 1\right) - \\ &- \Pr\left(K \stackrel{\text{R}}{\leftarrow} V^k; \mathbf{f}_i \stackrel{\text{R}}{\leftarrow} \text{Func}(V^n, V^n), \forall i \in V^n; \mathcal{A}^{\mathbf{f}_{(\overline{K} \oplus \cdot) \boxplus}(\cdot)}} \Rightarrow 1\right), \end{aligned}$$

the query is the triple  $(m, \phi, \sigma)$ , the response is  $y = \mathbf{g}(m, (\overline{K} \oplus \phi) \boxplus \sigma)$ .

We replace  $\mathbf{g}_K^\nabla = \mathbf{g}(\cdot, (\overline{K} \oplus \cdot) \boxplus \cdot)$  (the first and last compression functions in the inner and outer hash functions) with a family of true random functions  $\mathbf{f}_{(\overline{K} \oplus \cdot) \boxplus}(\cdot)$  and obtain  $\text{HMAC}^{(2)}$ .

Algorithm  $\mathcal{A}$ , which distinguishes  $\text{HMAC}^{(2)}$  from  $\text{HMAC}^{(1)}$ , can be used to attack  $\mathbf{g}_K^\nabla$  in the model  $\text{PRF-RKA}_{\oplus, \boxplus}$ . The corresponding algorithm  $\mathcal{B}_{\text{RKA}}$  works as follows. To process requests from  $\mathcal{A}$ , two preparatory queries  $(IV, \text{ipad}, 0)$  and  $(IV, \text{opad}, 0)$  to the oracle  $\mathbf{g}(\cdot, (\overline{K} \oplus \cdot) \boxplus \cdot)$  are required ( $K_{\text{Csc}}^I$  and  $K_{\text{Csc}}^O$  are obtained). Each query  $M \in (V^n)^{\leq l_{\max}}$  requires from  $\mathcal{B}_{\text{RKA}}$  no more than  $O(l_{\max})$  computations and two related-key queries

$$\begin{aligned} &(\text{Csc}(K_{\text{Csc}}^I, M), \text{ipad}, \sigma^I = \text{sum}'_{\boxplus}(M)), \\ &(\text{Csc}(K_{\text{Csc}}^O, H^I || 10 \dots 0 || \text{bin}(n + \tau)), \text{opad}, \sigma^O = \text{sum}_{\boxplus}(H^I || 10 \dots 0)). \end{aligned}$$

The result of work  $\mathcal{A}$  is equal to the result of work  $\mathcal{B}_{\text{RKA}}$  and the query complexity is  $q_{\mathcal{B}} = 2 + 2 \cdot q_{\mathcal{A}}$ .

$$\Pr\left(\mathcal{A}^{\text{HMAC}^{(1)}(\cdot)} \Rightarrow 1\right) - \Pr\left(\mathcal{A}^{\text{HMAC}^{(2)}(\cdot)} \Rightarrow 1\right) \leq \text{Adv}_{\mathbf{g}^\nabla}^{\text{PRF-RKA}_{\oplus, \boxplus}}(\mathcal{B}_{\text{RKA}}).$$

The **third** and **fourth steps** also remain the same.



The **fifth step**.

As a result of four steps we construct  $\mathbf{HMAC}^{(3)}$  as a truly random function generating  $H^I$ . The last compression function is defined in  $\mathbf{HMAC}^{(3)}$  as a family of random functions  $\mathbf{f}_{(\overline{K} \oplus \text{opad}) \boxplus \sigma^O}(x)$ . The value of  $H^I$  affects both inputs ( $\sigma^O$  and  $x$ ). We concern only about  $\sigma^O$  and ignore  $x$ . The adversary  $\mathcal{A}$  makes  $q$  queries to  $\mathbf{HMAC}^{(3)}$  and thereby generates  $2 + 2 \cdot q$  queries to  $\mathbf{f}_{(\overline{K} \oplus \cdot) \boxplus \cdot}(\cdot)$ . During the entire interaction, the following keys will be generated:

$$\begin{aligned} & \overline{K} \oplus \text{ipad}, \\ & \overline{K} \oplus \text{opad}, \\ & K_1^I = (\overline{K} \oplus \text{ipad}) \boxplus \sigma_1^I, \\ & K_1^O = (\overline{K} \oplus \text{opad}) \boxplus \sigma_1^O, \\ & \dots \\ & K_q^O = (\overline{K} \oplus \text{ipad}) \boxplus \sigma_q^I, \\ & K_q^O = (\overline{K} \oplus \text{opad}) \boxplus \sigma_q^O. \end{aligned}$$

We have two possible «bad» events:

- 1) collision in the sequence  $\mathbf{K}^O = \{K_1^O, \dots, K_q^O\}$ ;
- 2) nonempty intersection of  $\mathbf{K}^O$  and  $\mathbf{K}^I = \{K_1^I, \dots, K_q^I, \overline{K} \oplus \text{ipad}, \overline{K} \oplus \text{opad}\}$ .

If no bad events have occurred, then the last compression function never queried with the coinciding arguments and hence output of  $\mathbf{HMAC}^{(3)}$  is indistinguishable from a true random function.

The collision in  $\mathbf{K}^O$  is guaranteed to generate the collision in the output of  $\mathbf{HMAC}^{(3)}$ . Therefore, we are obliged to consider them.

Collisions in  $\mathbf{K}^I$  are not unsafe. This is taken into account in step 4, if the same keys are used, then the inputs are probably different.

The nonempty intersection of  $\mathbf{K}^O$  and  $\mathbf{K}^I$  may not lead to the reuse of random functions values, but to simplify the analysis, we consider only the worst case and assume that  $x$  value is the same for the same keys.

The probability of the first «bad» event is equal to the probability of the collision  $H^I \in V^\tau$

$$\Pr(\exists i \neq j : K_i^O = K_j^O) = \Pr(\exists i \neq j : \sigma_i^O = \sigma_j^O) \leq \frac{q \cdot (q - 1)}{2^{\tau+1}}.$$

The set  $\mathbf{K}^I$  contains at most  $(q + 2)$  elements. Due to the bijectivity of modular addition, we have at most  $(q + 2)$  values  $\sigma^O$  at which  $K^O \in \mathbf{K}^I$ . Hence, there are at most  $(q + 2)$  values  $H^I$  that lead to «bad» event 2

$$\Pr(\mathbf{K}^O \cap \mathbf{K}^I \neq \emptyset) \leq \frac{q \cdot (q + 2)}{2^\tau}.$$

and by the union bound and «fundamental game-playing lemma»

$$\text{Adv}_{\text{HMAC}^{(3)}}^{\text{PRF}}(q, l_{\max}) \leq \frac{q \cdot (q - 1)}{2^{\tau+1}} + \frac{q \cdot (q + 2)}{2^{\tau}} = \frac{3(q^2 + q)}{2^{\tau+1}}.$$

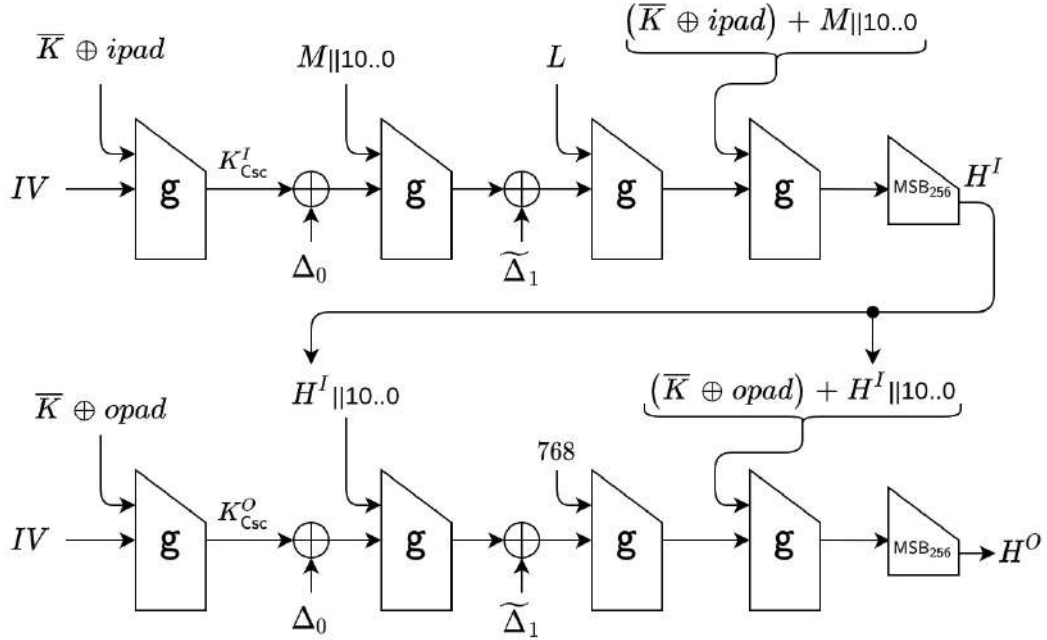


Figure 7: HMAC-Streebog-256 with equivalent representation. The message  $M$  consists of  $L < 512$  bits.

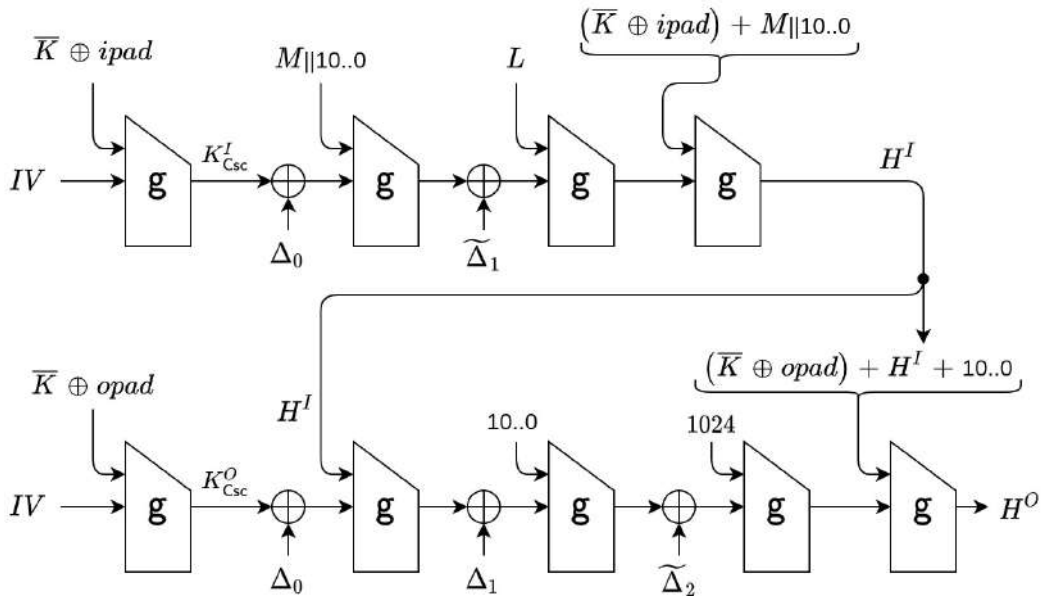


Figure 8: HMAC-Streebog-512 with equivalent representation. The message  $M$  consists of  $L < 512$  bits.

# Effective algorithm to compute guaranteed number of activations in XS-circuits and it's application to the cipher design

Denis Parfenov<sup>2</sup>, Aleksandr Bakharev<sup>1,2</sup>, Aleksandr Kutsenko<sup>1,2</sup>,  
Aleksandr Belov<sup>3</sup>, and Natalia Atutova<sup>1,2</sup>

<sup>1</sup>Sobolev Institute of Mathematics, Novosibirsk, Russia

<sup>2</sup>Novosibirsk State University, Novosibirsk, Russia

<sup>3</sup>Yaroslavl State University, Yaroslavl, Russia

d.parfenov@g.nsu.ru, a.bakharev@g.nsu.ru, alexandrkutsenko@bk.ru,  
ashmedey@gmail.com, n.atutova@g.nsu.ru

## Abstract

Guaranteed number of activations (GNA) is an important characteristics to determine effectiveness of differential cryptanalysis of a given XS-circuit. In this paper we propose an approach to optimize known algorithm for GNA computation, based on branch and bound method and the analysis of special matrices which define XS-circuit. The experiments show that the proposed algorithm significantly outperforms the existing approach. We prove that canonical forms of XS-circuit and its dual coincide that provides the strict connection between the guaranteed number of linear and differential activations. The circuits with extremal values of GNA are studied. Several hypotheses were made based on computational experiments. One of the hypotheses is that there are no XS-circuits of dimension greater then 2, which achieve an optimal GNA on every round.

**Keywords:** guaranteed number of activations, XS-circuit, differential cryptanalysis, linear cryptanalysis, branch and bound method.

## 1 Introduction

Many symmetric cryptographic algorithms may be described by circuits in which vertices are binary words of particular length  $m$  and operations belong to the following set:

1. **R** — cyclic shift (rotation);
2. **X** — bitwise modulo 2 addition;
3. **A** — addition of words as integers modulo  $2^m$ ;

4.  $\mathbf{L}$  — bitwise logical AND and OR;
5.  $\mathbf{M}$  — multiplication of words as elements of the field of order  $2^m$ ;
6.  $\mathbf{S}$  — substitution of words with preservation of their length  $m$ .

Usage of these operations in different combinations gives us different circuit types, for example ARX or LRX. Our work mostly focuses on  $\mathbf{XS}$ -circuits, develops framework for  $\mathbf{XS}$ -circuits cryptanalysis, proposed in [1]. We give new information about  $\mathbf{XS}$ -circuits gained via rigorous analysis and multiple experiments, which are made with a help of the optimizations that we apply to the guaranteed number of activations computation algorithm [2].

Number of activations is the amount of non-zero differences on S-block inputs within the rounds of the whole circuit. Guaranteed number of activations is the minimal number of activations for the given circuit. This characteristic gives us a lower bound of differential cryptanalysis complexity and allows to obtain preliminary estimates of the resilience of the given circuit to the differential cryptanalysis [3].

Another well-known method for cryptanalysis of block ciphers is a linear cryptanalysis [4]. Linear and differential methods are one of the main statistical methods, dual to each other in probability relations search [6]. For finding a lower bound of  $\mathbf{XS}$ -circuit linear cryptanalysis complexity it is possible to use the guaranteed number of activations for the dual  $\mathbf{XS}$ -circuit.

Current work consists of 7 sections. Section 1 is an introduction. In Section 2 notation and some necessary information about  $\mathbf{XS}$ -circuits and existing GNA algorithm is given. In Section 3 we give detailed explanation of the optimizations we applied to the original algorithm. In Section 4 we compare performance between original and proposed versions of GNA algorithm, and additionally compute activation times for some well-known circuits, including ones which were never computed before. In Section 5 we prove the equivalence of guaranteed number of differential and linear activations. In Section 6  $\mathbf{XS}$ -circuits with maximum possible GNA-values are discussed. In Section 7 there is a Conclusion.

## 2 $\mathbf{XS}$ -circuits and guaranteed number of activations

$\mathbf{XS}$ -circuits describe block ciphers that utilize 2 operations:  $\mathbf{X}$ ) bitwise modulo 2 addition of binary words and  $\mathbf{S}$ ) substitution of words using key-dependent S-boxes with possibly complicated internal structure. There are

a lot of ciphers based on **XS**-circuits. For example, MARS3, SMS4, Skipjack and the famous Feistel network.

Let  $\mathbb{F}$  be a field of characteristic 2. Instantiate the circuit over by substituting an oracle  $S : \mathbb{F} \rightarrow \mathbb{F}$  for the operation  $S$ . We get the mapping  $\mathbb{F}^n \rightarrow \mathbb{F}^n$ :

$$(a, B, c)[S](x) = xB + S(xa)c,$$

where  $a, c \in \mathbb{F}_2^n$  and elements of the matrix  $B$  belong to  $\mathbb{F}_2$ . **XS**-circuits could be described with a special *extended matrix of XS-circuit*

$$\begin{pmatrix} B & a \\ c & 0 \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1n} & a_1 \\ b_{21} & b_{22} & \dots & b_{2n} & a_2 \\ \vdots & \vdots & \ddots & \vdots & \\ b_{n1} & b_{n2} & \dots & b_{nn} & a_n \\ c_1 & c_2 & \dots & c_n & 0 \end{pmatrix}.$$

Any **XS**-circuit can be transformed to the equivalent form  $(a, B, c = (0, 0, \dots, 0, 1))$ , where  $B$  is a uniquely determined Frobenius cell. Such form is called *the first canonical form* [1].

Let us combine  $t$  instances of the circuit  $(a, B, c)$  by connecting one instance's output to the next instance's input. The resulting **XS**-circuit is called a  $t$ -round cascade and denoted by  $(a, B, c)^t$ .  $I$ -th activation time of the circuit  $(a, B, c)$  is denoted by  $\rho_i(a, B, c)$ . It equals to the minimum number of rounds  $t$  such that  $(a, B, c)^t$  guarantees  $i$  activations [1].

Matrix  $G$  of dimension  $(t + n) \times 2t$ , where  $n$  stands for circuit dimension and  $t$  for the amount of rounds, is built using extended matrix of **XS**-circuit in the first canonical form [2]

$$G = G(n, a, b, t)$$

Its columns have the following form:

$$\begin{array}{l} \tau - 1 \left\{ \begin{array}{l} 0 \ 0 \\ \vdots \ \vdots \\ 0 \ 0 \end{array} \right. \\ n + 1 \left\{ \begin{array}{l} a \ b \\ 0 \ 1 \end{array} \right. \\ t - \tau \left\{ \begin{array}{l} 0 \ 0 \\ \vdots \ \vdots \\ 0 \ 0 \end{array} \right. \end{array}$$

where  $b$  is the last column of  $B$ .

Existing GNA algorithm [2] is based on computation of the distance of a specific linear code built on the base of matrix  $G$ . The main idea of this algorithm is a search of partition of  $G$  into two submatrices  $G_0$  and  $G_1$  such that:

1. Submatrix  $G_0$  contains  $k + 1$  columns from  $G$ ;
2.  $\text{rank}(G_0) < t + n - 1$ , where  $t$  is the number of rounds,  $n$  is the length of  $a$  and  $b$ ;
3. Submatrix  $G_1$  does not contain any columns that are linearly dependent with columns from  $G_0$ ;
4. If such partition exists,  $k$  is incremented. Otherwise GNA of given circuit is equal to  $t - k$ .

### 3 Proposed optimization

Branch and bound is a well-known algorithm design paradigm for discrete and combinatorial optimization problems. The algorithm explores branches of a tree, which represent subsets of the solution set. Before enumerating the candidate solutions of a branch, the branch is checked against upper and lower estimated bounds on the optimal solution, and is discarded if it cannot produce a better solution than the best one found so far by the algorithm. The name «branch and bound» for the first time appeared in 1963 paper dedicated to traveling salesman problem [5], where proposed method was used for the optimization of exhaustive search.

In this section we will consider our proposed optimizations of the GNA algorithm inspired by branch and bound. To do this, we introduce the following

**Definition 1.** *We will call the partition of pairs of columns of the matrix  $G$  into matrices  $G_0$  and  $G_1$  «non-increasing» if  $G_1$  contains a column that linearly depends on the columns of the matrix  $G_0$ .*

The complexity of the existing algorithms grows exponentially in  $t$  and  $n$ . We propose a new approach based on usage of branch and bound method for building partitions of matrix  $G$ .

Consider a binary tree where each terminal leaf corresponds to a single partition of pairs of matrix  $G$  columns into matrices  $G_0$  and  $G_1$ . Root of given tree corresponds to an empty set of pairs of matrix  $G$  columns. The

first left node corresponds to the situation when the matrix  $G_0$  is empty and matrix  $G_1$  consists of the first pair of columns from  $G$  only. The first right node otherwise has the matrix  $G_1$  empty and the first pair of columns in  $G_0$ . On the  $i$ -th step of the tree construction a step to the left corresponds to addition of  $i$ -th pair to  $G_1$  and step to the right admits to  $G_0$ . As the result of such construction we obtain the tree of partitions of matrix  $G$ . In that tree each leaf corresponds to a partition of pairs of matrix  $G$  columns subset into matrices  $G_0$  and  $G_1$ .

**Proposition 1.** *If node corresponds to the «non-increasing» partition, then all it's children also correspond to «non-increasing» partitions.*

It means that during tree traversal we may stop on the first such node and do not traverse it's children at all. Such approach helps to noticeably speed up brute force, but still could be improved even more. Let  $n$  be a dimension of the XS-circuit.

**Lemma 1.** *For any  $n$  consecutive pairs of columns from  $G$ , the first column of the  $n + 1$ -th pair is linearly dependent on them.*

*Proof.* From Lemma 1 from the article [2] it follows that if  $t \geq n$ , then the matrix  $G$  has a full rank equal to the numbers of rows ( $n + t$ ). We call the rows that have nonzero elements significant. Consider  $n$  consecutive pairs of columns and leave only significant rows in them. The resulting matrix coincides with the matrix  $G$  at  $t = n$  and has a rank equal to  $2n$ . By addition to the first column of the  $n + 1$ -th pair to such matrix, we get matrix with dimension  $(2n) \times (2n + 1)$ , but the rank of the matrix cannot exceed the number of significant rows, therefore the rank of such matrix is equal to  $2n$ . Then the additional column is linearly dependent with the rest.  $\square$

**Lemma 2.** *Let  $a_1$  and  $b_1$  not be equal to 1 simultaneously and the conditions of Lemma 1 are fulfilled. Then one of the columns of the preceding pair will be linearly dependent on the columns of the following  $n$  pairs.*

*Proof.* Note that  $a_1$  and  $b_1$  cannot be equal to 0 at the same time, otherwise the round of such an XS-circuit will not be reversible. Having carried out arguments similar to the proof of the Lemma 1, we get that the column from the preceding pair, whose first element is 0, will be linearly expressed through the rest.  $\square$

The sufficient conditions for the partition to be «non-increasing» are provided by the following

**Theorem 1.** *Let  $G_0$  contains  $n$  consecutive pairs of columns. If one of the following cases holds, the partition is «non-increasing»:*

- 1) *there is a pair from  $G_1$  with a greater ordinal number than  $n$  consecutive pairs from  $G_0$ ;*
- 2)  *$a_1$  and  $b_1$  are not equal to 1 at the same time.*

*Proof.* 1) Consider a pair from  $G_1$  which ordinal number is minimal and is greater than the ordinal numbers of  $n$  consecutive pairs of columns from  $G_0$ . If such pair of columns follows  $n$  consecutive pairs from  $G_0$ , then according to Lemma 1 it follows that the first of its columns is linearly dependent on the columns from  $G_0$ . By definition, such partitioning is «non-increasing». If ordinal number of such pair is not equal to the ordinal number of the pair which follows after  $n$  consecutive pairs from  $G_0$ , then because this number is minimal, there exists  $n$  consecutive pairs of columns from  $G_0$  that their maximum ordinal number will be one less than ordinal number of pair from  $G_1$ . Then, according to the reasoning above, such a partition is «non-increasing».

2) From the construction of the matrix  $G_0$  and Lemmas 1, 2, it follows that if  $G_0$  contains  $n$  consecutive pairs of columns and  $a_1, b_1$  are not equal to 1 at the same time, then in  $G_1$  there is a column that is linearly dependent on the columns of the matrix  $G_1$ . Then, by definition, such partition is «non-increasing». □

Theorem 1 allows us to add criteria that cuts off obviously non-optimal branches of the search. Also, additional criteria can be deduced from it. For example, the following proposition gives us opportunity to cutoff branch with «non-increasing» partition even before occurrence of  $n$  consecutive pairs in  $G_0$ , as soon as their appearance becomes inevitable:

**Proposition 2.** *Let  $a_1$  and  $b_1$  are not equal to 1 simultaneously and for the partition, given by the processed node of the tree, it is impossible to complement the matrix  $G_0$  to  $k + 1$  pairs of columns without the appearance of  $n$  consecutive pairs of columns from  $G$ . Then such partition is «non-increasing».*

Consider Theorem 7 from the article [1]. Let  $(a, B, c)$  be a circuit of dimension  $n$  in which  $B$  is a Frobenius cell with a characteristic polynomial  $\lambda^n + b_n \lambda^{n-1} + \dots + b_1$ . The circuit is invertible if and only if one of the following cases holds:

1.  $b_1 = 1$  and  $a_1 (b_2 c_1 + b_3 c_2 + \dots + b_n c_{n-1} + c_n) + a_2 c_1 + a_3 c_2 + \dots + a_n c_{n-1} = 0$ ;



$$2. b_1 = 0, a_1 = 1 \text{ and } b_2c_1 + b_3c_2 + \dots + b_nc_{n-1} + c_n = 1.$$

By definition of the first canonical form we have  $c = (0, \dots, 0, 1)$ . Then if the circuit from the condition of Theorem 7 from [1] has the first canonical form, then both cases are rewritten as follows:

$$1. b_1 = 1 \text{ и } a_1 = 0;$$

$$2. b_1 = 0 \text{ и } a_1 = 1.$$

Then it is possible to exclude the condition « $a_1$  and  $b_1$  not be equal to 1 at the same time» from Proposition 2 for all invertible circuits.

So the main optimization of the algorithm is achieved by using the branch and bound method based on Proposition 1 and the cutoff criteria based on Theorem 1 and Proposition 2.

## 4 Computational experiments

In order to run computational experiments, for the original version of the algorithm we use the implementation written in Python by the author of the original article on GNA (All calculations were performed on Intel Core i5-7300HQ) [8]. Proposed algorithm is an algorithm built using additional criteria for a branch and bound algorithm based on corollaries from Theorem 1.

In this section we run existing and proposed versions of algorithm in order to compute guaranteed number of activations for some well-known circuits.

### 4.1 SMS4

SM4 (SMS4) is a block cipher used in the Chinese National Standard for Wireless LAN WLAN Authentication and Privacy Infrastructure since 2016 (GB/T 32907-2016). Encryption or decryption of one block of data is composed of 32 rounds. It easy to see from plots in Figure 1 that computation of GNA for 32 rounds would have take several hours, if the old version of algorithm was used.

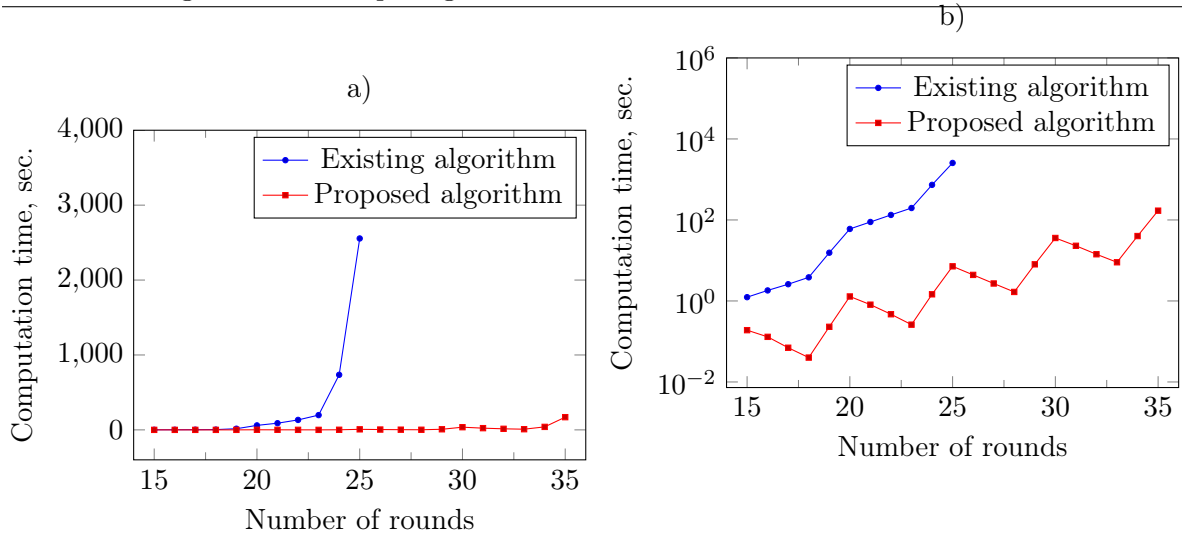


Figure 1: GNA computation time for SMS4 ( $n = 4$ ) a) with a linear time scale b) with a logarithmic time scale

## 4.2 Skipjackg-4

Skipjack was proposed as the encryption algorithm in a US government-sponsored scheme of key escrow, and the cipher was provided for use in the Clipper chip, implemented in tamperproof hardware. It is an unbalanced Feistel network with 32 rounds (the same as for SM4). b)

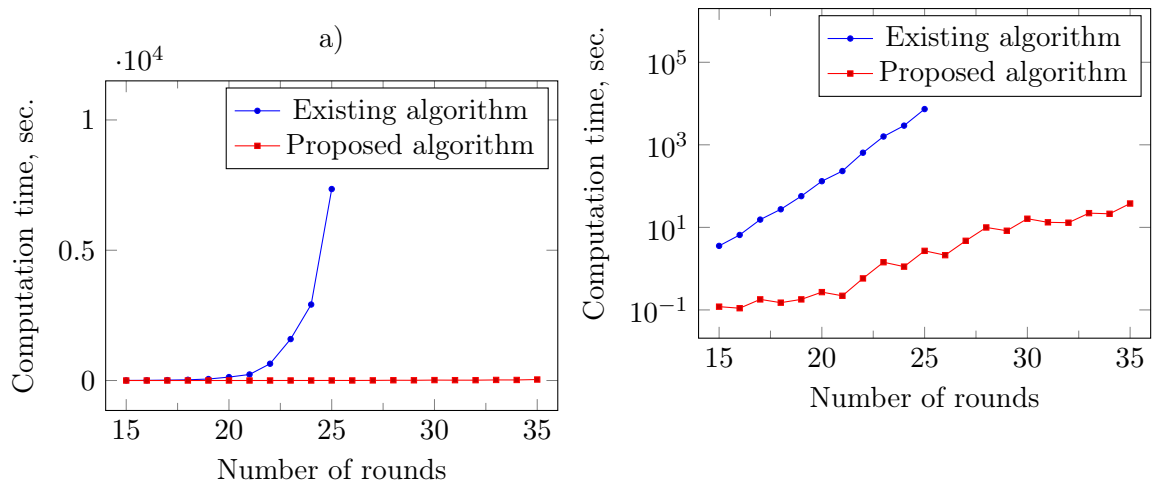


Figure 2: GNA computation time for skipjackg-4 ( $n = 4$ ) a) with a linear time scale b) with a logarithmic time scale

## 4.3 BeltWBL-4

This block cipher was described in standart [7]. According to [1], BeltWBL is the core of the belt-keywrap algorithm which provides confidentiality and integrity control of variable length keys.

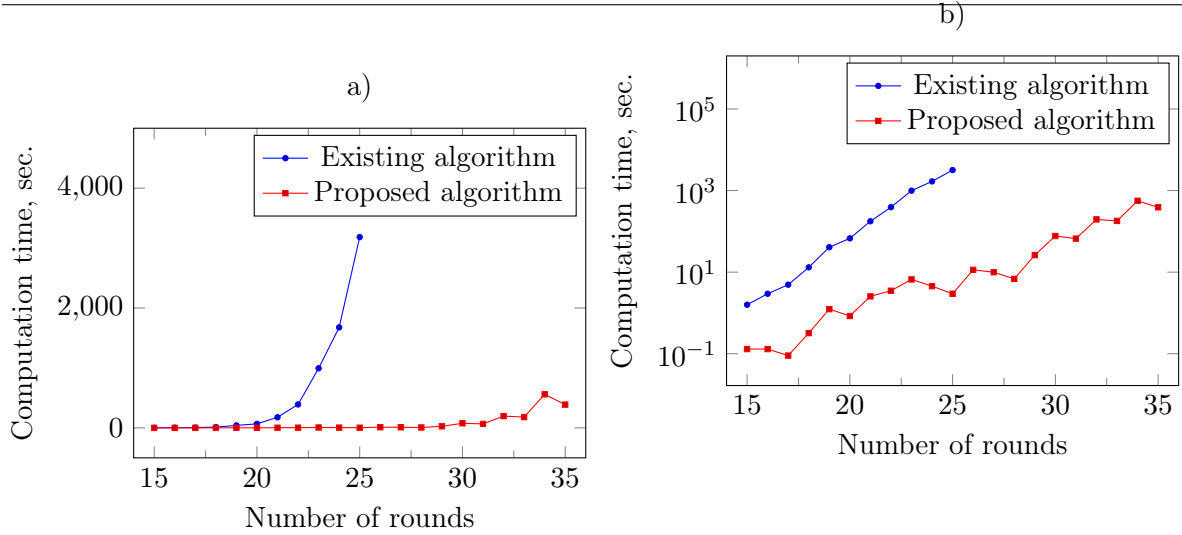


Figure 3: GNA computation time for beltWBL-4 ( $n = 4$ ) a) with a linear time scale b) with a logarithmic time scale

#### 4.4 Computation results

Number of rounds	SMS4		skipjackg-4		beltWBL-4	
	Ex. alg.	Pr. alg.	Ex. alg.	Pr. alg.	Ex. alg.	Pr. alg.
15	1,2366	0,1947	3,5638	0,1209	1,5898	0,134
16	1,818	0,1267	6,5453	0,1083	2,9466	0,1289
17	2,5896	0,0738	15,4309	0,1826	4,9232	0,0878
18	3,8312	0,0352	27,4088	0,1508	13,085	0,3157
19	15,5206	0,2303	57,0557	0,182	40,8592	1,2425
20	60,0129	1,2929	131,3367	0,2681	67,303	0,8394
21	89,1743	0,8057	231,5267	0,2155	176,574	2,5539
22	132,9802	0,4689	644,076	0,5805	391,1274	3,4973
23	197,0892	0,264	1589,793	1,4337	994,2484	6,6221
24	735,1466	1,4642	2916,3251	1,117	1677,1016	4,5253
25	2555,2445	7,1559	7348,0468	2,6989	3183,5355	2,9505
26	—	4,3972	—	2,1177	—	11,3771
27	—	2,7019	—	4,7252	—	9,9593
28	—	1,6709	—	9,9505	—	6,84
29	—	8,0172	—	8,2899	—	26,1846
30	—	35,8738	—	16,2128	—	76,8393
31	—	23,0073	—	13,2468	—	66,3372
32	—	14,217	—	12,983	—	196,2868
33	—	9,0241	—	22,1154	—	180,3136
34	—	40,0366	—	21,3304	—	560,1749
35	—	169,632	—	37,8486	—	388,0823

Table 1: GNA computation time in seconds for SMS4, skipjackg-4, beltWBL-4

As shown in table 1 and graphs with logarithmic time scale, the time complexity of the proposed algorithm remains exponentially dependent on

the number of rounds.

$i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\rho_i(\text{SMS4})$	4	5	9	10	14	15	19	20	24	25	29	30	34	35	39
$\rho_i(\text{Skipjackg-4})$	4	7	8	10	12	13	14	17	20	22	23	25	27	28	30
$\rho_i(\text{BeltWBL-4})$	4	7	8	10	12	15	18	19	21	23	26	29	30	32	34
$\rho_i(\text{GFN1-4})$	4	7	8	10	12	13	14	17	20	22	23	25	27	28	30
$\rho_i(\text{Feistel})$	2	3	5	6	8	9	11	12	14	15	17	18	20	21	23

Table 2: Computed activation times for a some well-known circuits

By using proposed algorithm it is possible to extend table of activation times presented in [1], that allows us to compute GNA for the full-round variants of the ciphers.

## 5 GNA application to linear cryptanalysis

As it was mentioned earlier, it is possible to use GNA in order to get preliminary estimate of resistance against both differential and linear cryptanalysis [1, 4]. In papers [1, 2] it was shown that in order to get the number of linear activations of the circuit, GNA of the dual circuit should be computed. The dual circuit to  $(a, B, c)$  is the circuit  $(c^T, B^T, a^T)$ .

Following Theorem shows that the guaranteed number of linear activations is the same as the guaranteed number of differential activations.

**Theorem 2.** *Let  $(a, B, c)$  be a XS-circuit in the first canonical form. Then the first canonical form of its dual circuit  $(c^T, B^T, a^T)$  coincides with  $(a, B, c)$ .*

*Proof.* Consider the idea of the proof. Denote the dual circuit to the  $(a, B, c)$  as  $(\bar{a}, \bar{B}, \bar{c})$ . By definition of duality:  $\bar{a} = c^T, \bar{B} = B^T, \bar{c} = a^T$ . We bring the  $(\bar{a}, \bar{B}, \bar{c})$  to the first canonical form  $(\tilde{a}, \tilde{B}, \tilde{c})$ . Further we prove that  $\tilde{B} = B$  and  $\tilde{a} = a$ . See appendix for details of the proof.  $\square$

It follows, that all the provided experiments also show the number of linear activations. In particular, Table 2 shows not only differential but linear activation times as well.

## 6 XS-circuits with extremal value of GNA

### 6.1 Maximum value of the GNA

The set of regular XS-circuits splits into equivalence classes with respect to the similarity relation [1]. Let us choose the representatives of classes XS-

circuits with matrix representation of the form

$$\begin{pmatrix} B & a \\ c & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & \dots & 0 & 0 & b_1 & a_1 \\ 1 & 0 & \dots & 0 & 0 & b_2 & a_2 \\ 0 & 1 & \dots & 0 & 0 & b_3 & a_3 \\ \vdots & \ddots & \dots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 & b_{n-1} & a_{n-1} \\ 0 & 0 & \dots & 0 & 1 & b_n & a_n \\ 0 & 0 & \dots & 0 & 0 & 1 & 0 \end{pmatrix}.$$

The following questions arise:

- how to find the maximum or minimum value of GNA for regular circuits for the fixed number of rounds?
- is there a regular circuit that reaches the maximum value of GNA for any number of rounds?
- is it possible to propose a construction of regular XS-circuits with a maximum or almost maximum value of GNA?

By using the proposed version of the algorithm for calculating GNA we computed maximum values of the GNA for different  $n$ . Figure 4 shows maximum values of the GNA for number of rounds from 2 to 23.

We found that for  $n > 2$  there are no regular XS-circuits that reach maximum number of activations in each round. For  $n = 2$ , there are 3 XS-circuits with maximum guaranteed number of activations:

$$S1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad S2 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad S3 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

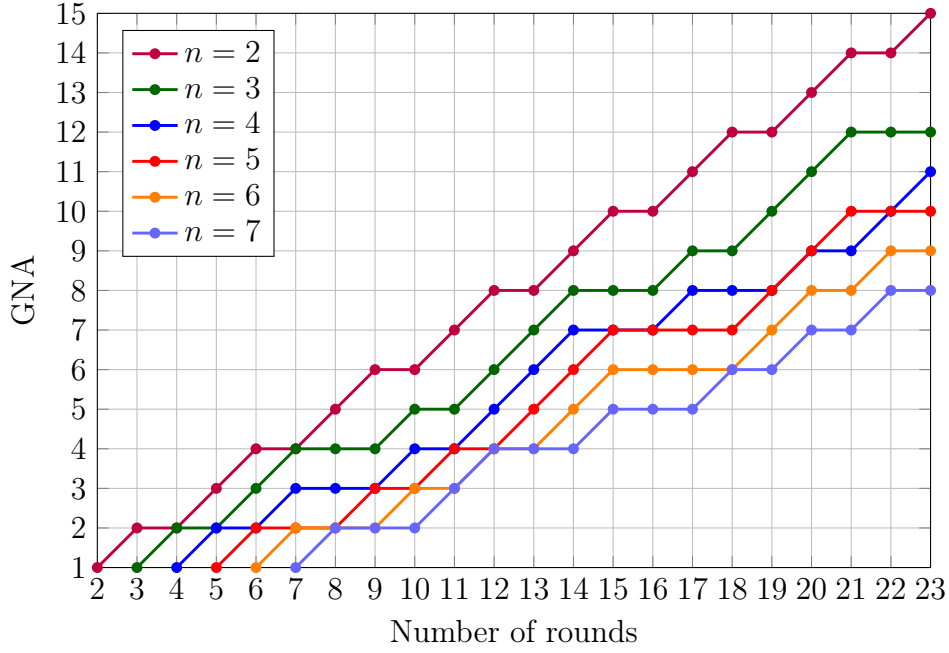


Figure 4: Maximum value of the GNA for number of rounds from 2 to 23

The circuit  $S1$  is equivalent to the Feistel network circuit and  $S2$  is the same as the circuit that describes the cipher MISTY2.

Let us denote the GNA for  $r$  rounds as  $\delta(r)$ .

**Proposition 3.** *For XS-circuit with matrix  $S3$  it holds*

$$\delta(3k - 1) = 2k - 1,$$

$$\delta(3k) = 2k,$$

$$\delta(3k + 1) = 2k,$$

where  $k \geq 1$ .

*Proof.* The induction basis is verified directly (fig. 4). Suppose the statement is true for  $k$ . Let us prove it for  $k + 1$ . Let's construct a matrix [2]

$$G = G(2, [1, 1]^T, [0, 0], t),$$

where

$$t \in \{3(k + 1) - 1, 3(k + 1), 3(k + 1) + 1\} \text{ and } t > 4.$$

Then it holds

$$(x_1, x_2, \dots, x_{t+2})G = W = (u, v),$$

where  $u$  is the vector of differences in the inputs of the XS-circuit after  $t - 3$  rounds, and the vector  $v$  contains the differences after additional three

rounds, i.e.

$$\begin{aligned} u &= (x_1 + x_2, x_3, x_2 + x_3, x_4, \dots, x_{t-3} + x_{t-2}, x_{t-1}), \\ v &= (x_{t-2} + x_{t-1}, x_t, x_{t-1} + x_t, x_{t+1}, x_t + x_{t+1}, x_{t+2}). \end{aligned}$$

By the induction hypothesis for  $t - 3$  rounds, the guaranteed number of activations is equal to  $2k - 1$  if  $t - 3 = 3k - 1$ , and equals  $2k$  if  $(t - 3) \in \{3k, 3k + 1\}$ . Therefore, there are such elements of the field, for which there are  $2k - 1$  or  $2k$  activations. Fix these elements. Our goal is to find such elements of the field which minimize number of activations.

Consider two cases:

- on the  $(t - 3)$ -th round activation occurred;
- on the  $(t - 3)$ -th round activation did not occur.

If activation did not occur, then  $x_{t-1} = 0$ . In this case if  $x_{t-2} = 0$ , then  $u$  is zero vector and activation in  $u$  did not occurred. That contradicts the choice of the vector  $u$ . So  $x_{t-2} = \alpha \neq 0$ . Since  $x_i + x_{i+1}$  and  $x_{i+2}$  either both zero or nonzero [2], we have  $x_t = \beta \neq 0$ . Hence

$$v = (\alpha + 0, \beta, 0 + \beta, x_{t+1}, \beta + x_{t+1}, x_{t+2}).$$

Similary  $x_{t+1} \neq 0$ . Hence there are two ore more activations in vector  $v$ . To get exactly 2 activations put  $x_{t+1} = x_t = \beta$ . Hence

$$v = (\alpha + 0, \beta, 0 + \beta, \beta, \beta + \beta, 0) = (\alpha, \beta, \beta, \beta, 0, 0)$$

and after  $t$ -th round number of activations equals  $2(k + 1) - 1$ , in case of  $t = 3(k + 1) - 1$ , and equals  $2(k + 1)$ , in case of  $t = 3(k + 1)$  or  $t = 3(k + 1) + 1$ .

The case when activation occurred on the  $t - 3$  round is considered by the same way. □

We also propose the following

**Hypothesis 1.** *For  $n = 2$  and arbitrary XS-circuit*

$$\begin{aligned} \delta(3k - 1) &\leq 2k - 1, \\ \delta(3k) &\leq 2k, \\ \delta(3k + 1) &\leq 2k, \end{aligned}$$

where  $k \geq 1$ , i.e.  $S3$  is optimal (achieves maximal GNA) for any number of rounds.

Although there are no optimal circuits for  $n > 2$ , there are circuits that reach the optimal number of activations starting from a certain round, see Figure 5. Therefore, for  $n > 2$  the concept of an optimal circuit can be replaced by a concept of a  $t$ -optimal circuit, i.e. circuit that achieve the optimal number of activations for all rounds starting from  $t$ . The same essential questions arise for  $t$ -optimal circuits:

1. What is the minimum value of  $t$  for which  $t$ -optimal circuits exist?
2. Is it possible to propose a construction of  $t$ -optimal circuit, and how to prove their  $t$ -optimality?

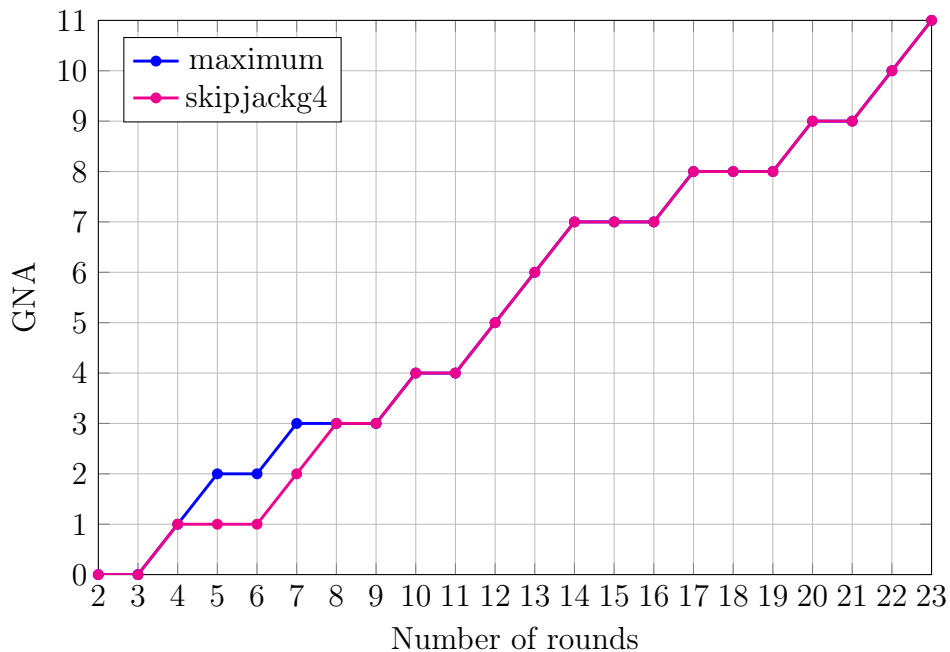


Figure 5: GNA for skipjackg4

## 6.2 Minimum value of the GNA

Also, during exhaustive search through all XS-circuits of low dimensions we observed a class of regular XS-circuits with a minimum number of activations. The structure of such circuits can be deduced theoretically. It is considered in the following

**Proposition 4.** *Let  $(a, B, c)$  be a regular XS-circuit of dimension  $n$  in the first canonical form and  $a + b = (1, 0, \dots, 0)$ . Then  $\rho_k(a, B, c) = kn$ .*

*Proof.* Let's find the number  $d(\mathbb{F}_2, n, a, b, t)$ . Since  $a + b = (1, 0, \dots, 0)$ , we have  $x_{\tau+n} = x_\tau$  (see [2], Section 4). If we take the input difference  $(x_1, \dots, x_n) = (0, \dots, 0, 1)$ , then  $d(\mathbb{F}_2, n, a, b, t) = \lfloor \frac{t}{n} \rfloor$ . From [2] we



have  $d(n, a, b, t) \leq d(\mathbb{F}_2, n, a, b, t)$  and  $\lfloor \frac{t}{n} \rfloor$  is the lower bound on the guaranteed number of activation  $d(n, a, b, t)$ , then  $d(n, a, b, t) = \lfloor \frac{t}{n} \rfloor$ .  $\square$

Therefore, we found the class of **XS**-circuit that is potentially vulnerable to differential and linear attacks and is not recommended for use.

## 7 Conclusion and further plans

In our work we propose an optimization to GNA algorithm which significantly speeds up its computation, which could be useful in cipher design because it allows to get preliminary estimate of resistance against differential and linear cryptanalysis. Implementation of proposed algorithm was approved by the author of the original papers on GNA, and currently is used as the main version [8].

We prove that the number of differential activations coincides with the number of linear activations that provides simultaneous estimate of efficiency for both types of attacks.

During algorithm analysis we additionally succeeded in finding some interesting facts: such as a structure of **XS**-circuits with minimum GNA, that are potentially vulnerable to attacks, or the fact that there are no **XS**-circuits of dimension  $n > 2$  which achieve an optimal GNA on any amount of rounds.

We plan to continue the investigation of the properties of **XS**-circuits, in particular, apply information received from analysis of computational experiments to design primitives based on **XS**-circuits with an optimal guaranteed number of activations. Such circuits are able to provide balanced high level of security against both linear and differential attacks. Possible primitives include block ciphers, hash-functions and random number generators. In this field, such primitives as sponge functions, based on **XS**-circuits, should also be studied.

### **Acknowledgments.**

We wish to thank Sergey Agievich for valuable discussions and advices.

The work is supported by Mathematical Center in Akademgorodok under agreement No. 075-15-2022-281 with the Ministry of Science and Higher Education of the Russian Federation.

## References

- [1] Agievich S.V., “**XS**-circuits in block ciphers”, *Mat. Vopr. Kriptogr.*, **10**:2 (2019), 7–30.

- [2] Agievich S.V., “On the guaranteed number of activations in XS-circuits”, *Mat. Vopr. Kriptogr.*, **12**:2 (2021), 7–20.
- [3] Biham E., Shamir A., “Differential Cryptanalysis of DES-like Cryptosystems”, *Journal of Cryptology*, **4**:1 (1991), 3–73.
- [4] Matsui M., “Linear Cryptanalysis Method for DES Cipher”, *Advances in Cryptology – EUROCRYPT ’93*, 1994, 386–397.
- [5] Little J., Murty K., Sweeney D., Karel C., “An algorithm for the traveling salesman problem”, *Operations Research*, **11**:6 (1963), 972–989.
- [6] Malyshev F.M., “The duality of differential and linear methods in cryptography”, *Mat. Vopr. Kriptogr.*, **5**:3 (2014), 35–47.
- [7] *STB 34.101.31-2011. Information Technology and Security. Data Encryption and Integrity Algorithms.*, 2011.
- [8] “Python implementation of algorithm for calculating the guaranteed number of activations in a given cascade”, <https://github.com/agievich/xs>.

## 8 Appendix

*Proof.* Denote the dual circuit to the  $(a, B, c)$  as  $(\bar{a}, \bar{B}, \bar{c})$ . By definition of duality:  $\bar{a} = c^T$ ,  $\bar{B} = B^T$ ,  $\bar{c} = a^T$ . We bring the  $(\bar{a}, \bar{B}, \bar{c})$  to the first canonical form  $(\tilde{a}, \tilde{B}, \tilde{c})$  by using the theory from [1]:

1) Let’s find the matrix  $\tilde{B} = A^{-1}\bar{B}A = A^{-1}B^T A$ , where  $A = (\bar{a} \ \bar{B}\bar{a} \ \dots \ \bar{B}^{n-1}\bar{a})$ .

Since  $\bar{a} = c^T$  and  $c = (0, \dots, 0, 1)$ , then in  $\bar{B}^i \bar{a}$  we are only interested in the last column.

$$\begin{aligned} \bar{B}^2 &= \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ b_1 & b_2 & b_3 & \dots & b_n \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ b_1 & b_2 & b_3 & \dots & b_n \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ b_1 & b_2 & b_3 & \dots & b_n \\ b_1 b_n & b_1 + b_2 b_n & b_2 + b_3 b_n & \dots & b_{n-1} b_n \end{pmatrix}. \end{aligned}$$

Note that the matrix  $\bar{B}$ , when multiplied on the left, lifts the last  $n - 1$  rows one up. Denote by  $\alpha_i$  the lower right element of the matrix  $\bar{B}^i$ . Then

the matrix  $A$  has the following form:

$$A = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 & \alpha_1 \\ 0 & 0 & 0 & \dots & 1 & \alpha_1 & \alpha_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & \dots & \alpha_{n-5} & \alpha_{n-4} & \alpha_{n-3} \\ 0 & 1 & \alpha_1 & \dots & \alpha_{n-4} & \alpha_{n-3} & \alpha_{n-2} \\ 1 & \alpha_1 & \alpha_2 & \dots & \alpha_{n-3} & \alpha_{n-2} & \alpha_{n-1} \end{pmatrix},$$

where  $\alpha_i = b_{n+1-i} + \sum_{j=1}^{i-1} \alpha_j b_{n+1-i+j}$ . Then from  $AA^{-1} = E$  find the elements of the matrix  $A^{-1}$ :

$$A^{-1} = \begin{pmatrix} \alpha_{n-1}^{-1} & \alpha_{n-2}^{-1} & \dots & \alpha_2^{-1} & \alpha_1^{-1} & 1 \\ \alpha_{n-2}^{-1} & \alpha_{n-3}^{-1} & \dots & \alpha_1^{-1} & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \alpha_1^{-1} & 1 & \dots & 0 & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 & 0 \end{pmatrix},$$

where  $\alpha_i^{-1} = \alpha_i + \sum_{j=1}^{i-1} \alpha_j^{-1} \alpha_{i-j}$ .

$$C = B^T A = \begin{pmatrix} 0 & 0 & \dots & 0 & 1 & \alpha_1 \\ 0 & 0 & \dots & 1 & \alpha_1 & \alpha_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 1 & \alpha_1 & \dots & \alpha_{n-3} & \alpha_{n-2} & \alpha_{n-1} \\ c_1 & c_2 & \dots & c_{n-2} & c_{n-1} & c_n \end{pmatrix},$$

where  $c_i = b_{n+1-i} + \sum_{j=1}^{i-1} \alpha_j b_{n+1-i+j} = \alpha_i$ .

$$\begin{aligned} \tilde{B} &= A^{-1} B^T A \\ &= \begin{pmatrix} \alpha_{n-1}^{-1} & \alpha_{n-2}^{-1} & \cdots & \alpha_1^{-1} & 1 \\ \alpha_{n-2}^{-1} & \alpha_{n-3}^{-1} & \cdots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_1^{-1} & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & \cdots & 1 & \alpha_1 \\ 0 & 0 & \cdots & \alpha_1 & \alpha_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & \alpha_1 & \cdots & \alpha_{n-2} & \alpha_{n-1} \\ c_1 & c_2 & \cdots & c_{n-1} & c_n \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 & \cdots & 0 & w_1 \\ 1 & 0 & \cdots & 0 & w_2 \\ 0 & 1 & \cdots & 0 & w_3 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & w_n \end{pmatrix}, \end{aligned}$$

where  $w_{n+1-i} = \alpha_i + \sum_{j=1}^{i-1} \alpha_j^{-1} \alpha_{i-j} = \alpha_i^{-1}$ ,  $i = 1, 2, \dots, n$ . Consider these numbers in details:

$$\begin{aligned} \alpha_i^{-1} &= \alpha_i + \sum_{j=1}^{i-1} \alpha_j^{-1} \alpha_{i-j} \\ &= b_{n+1-i} + \sum_{j=1}^{i-1} (\alpha_j^{-1} \alpha_{i-j} + \alpha_j b_{n+1-i+j}) \\ &= b_{n+1-i} + (\alpha_1^{-1} \alpha_{i-1} + \alpha_1 b_{n+2-i}) + (\alpha_2^{-1} \alpha_{i-2} + \alpha_2 b_{n+3-i}) \\ &\quad + \dots + (\alpha_{i-2}^{-1} \alpha_2 + \alpha_{i-2} b_{n-1}) + (\alpha_{i-1}^{-1} \alpha_1 + \alpha_{i-1} b_n) \\ &= b_{n+1-i} + \sum_{j=1}^{i-1} \alpha_j (b_{n+1-i+j} + \alpha_{i-j}^{-1}). \end{aligned}$$

We obtain that

$$\alpha_1^{-1} = b_n \Rightarrow \alpha_2^{-1} = b_{n-1} \Rightarrow \alpha_3^{-1} = b_{n-2} \Rightarrow \dots \Rightarrow \alpha_n^{-1} = b_1$$

and finally  $\tilde{B} = B$ .

**2)** Now let's find the vector  $\tilde{a} = PA^{-1}\bar{a} = PA^{-1}c^T$ .

We have

$$A^{-1}c^T = \begin{pmatrix} \alpha_{n-1}^{-1} & \alpha_{n-2}^{-1} & \dots & \alpha_1^{-1} & 1 \\ \alpha_{n-2}^{-1} & \alpha_{n-3}^{-1} & \dots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_1^{-1} & 1 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix},$$

so we are only interested in the first column of the matrix  $P$ .

$$P = P(\tilde{c}A) = \begin{pmatrix} \tilde{c}AM_1 \\ \tilde{c}AM_2 \\ \vdots \\ \tilde{c}AM_n \end{pmatrix},$$

where  $M_n = E, M_i = BM_{i+1} + b_{i+1}E = B^{n-i} + b_n B^{n-i-1} + \dots + b_{i-1}E$ .

$$\tilde{c}A = a^T \begin{pmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 1 & \alpha_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & \dots & \alpha_{n-3} & \alpha_{n-2} \\ 1 & \alpha_1 & \dots & \alpha_{n-2} & \alpha_{n-1} \end{pmatrix} = (v_1, v_2, \dots, v_{n-1}, v_n),$$

where  $v_i = a_{n+1-i} + \sum_{j=1}^{i-1} \alpha_j a_{n+1-i+j}$ ,  $i = 1, 2, \dots, n$ . Consider in details these matrices

$$B^k = \begin{pmatrix} 0 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \dots & * \\ 1 & * & \dots & * \\ 0 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \dots & * \end{pmatrix} \Rightarrow M_{n-t} = \begin{pmatrix} b_{n+1-t} & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ b_n & * & \dots & * \\ 1 & * & \dots & * \\ 0 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \dots & * \end{pmatrix}.$$

Then

$$P = \begin{pmatrix} vM_1 \\ vM_2 \\ \vdots \\ vM_{n-1} \\ vM_n \end{pmatrix} = \begin{pmatrix} p_1 & * & \dots & * \\ p_2 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ p_{n-1} & * & \dots & * \\ p_n & * & \dots & * \end{pmatrix},$$

where  $p_{n+1-i} = v_i + \sum_{j=1}^{i-1} v_j b_{n+1-i+j} = a_{n+1-i} + \sum_{j=1}^{i-1} (\alpha_j a_{n+1-i+j} + v_j b_{n+1-i+j})$ .

We are to prove that  $\sum_{j=1}^{i-1} (\alpha_j a_{n+1-i+j} + v_j b_{n+1-i+j}) = 0$ . It can be

$$\begin{aligned}
 & \sum_{j=1}^{i-1} (\alpha_j a_{n+1-i+j} + v_j b_{n+1-i+j}) \\
 &= \sum_{j=1}^{i-1} \left[ a_{n+1-i+j} \left( b_{n+1-j} + \sum_{k=1}^{j-1} \alpha_k b_{n+1-j+k} \right) \right. \\
 & \quad \left. + b_{n+1-i+j} \left( a_{n+1-j} + \sum_{k=1}^{j-1} \alpha_k a_{n+1-j+k} \right) \right] \\
 &= \sum_{j=1}^{i-1} \left( a_{n+1-i+j} b_{n+1-j} + b_{n+1-i+j} a_{n+1-j} \right) \\
 & \quad + \sum_{j=2}^{i-1} \sum_{k=1}^{j-1} \alpha_k \left( a_{n+1-i+j} b_{n+1-j+k} + b_{n+1-i+j} a_{n+1-j+k} \right)
 \end{aligned}$$

Consider the last two amounts separately. For the first one it holds

$$\begin{aligned}
 & \sum_{j=1}^{i-1} (a_{n+1-i+j} b_{n+1-j} + b_{n+1-i+j} a_{n+1-j}) \\
 &= (a_{n+2-i} b_n + b_{n+2-i} a_n) + (a_{n+3-i} b_{n-1} + b_{n+3-i} a_{n-2}) \\
 & \quad + \dots + (a_{n-1} b_{n+3-i} + b_{n-1} a_{n+3-i}) + (a_n b_{n+2-i} + b_n a_{n+2-i}) = 0,
 \end{aligned}$$

whereas for the second one we have the following multipliers with each  $\alpha_i$ ,  $i = 1, 2, \dots, i-2$ :

$$\alpha_1 : (a_{n+3-i} b_n + b_{n+3-i} a_n) + (a_{n+4-i} b_{n-1} + b_{n+4-i} a_{n-1}) + \dots + (a_{n-1} b_{n+4-i} + b_{n-1} a_{n+4-i}) + (a_n b_{n+3-i} + b_n a_{n+3-i}) = 0;$$

$$\alpha_2 : (a_{n+4-i} b_n + b_{n+4-i} a_n) + (a_{n+5-i} b_{n-1} + b_{n+5-i} a_{n-1}) + \dots + (a_{n-1} b_{n+5-i} + b_{n-1} a_{n+5-i}) + (a_n b_{n+4-i} + b_n a_{n+4-i}) = 0;$$

⋮

$$\alpha_{i-3} : (a_{n-1} b_n + b_{n-1} a_n) + (a_n b_{n-1} + b_n a_{n-1}) = 0;$$

$$\alpha_{i-2} : (a_n b_n + b_n a_n) = 0,$$

therefore, all the multipliers of  $\alpha_i$  are zero.

Thus, we obtain that  $p_i = a_i$  for  $i = 1, 2, \dots, n$ . Then it holds

$$\tilde{a} = PA^{-1}c^T = \begin{pmatrix} p_1 & * & \dots & * \\ p_2 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ p_{n-1} & * & \dots & * \\ p_n & * & \dots & * \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{n-1} \\ a_n \end{pmatrix} = a.$$

□

# On some algebraic aspects of heuristic algorithms for constructing cryptographically strong S-Boxes

Alejandro Freyre Echevarría, Oliver Coy Puente,  
and Reynier A. de la Cruz Jiménez

Institute of Cryptography, University of Havana, Cuba  
{freyreealejandro, coypuente.o, djr.antonio537}@gmail.com

## Abstract

The present article is concerned with the problem of constructing vectorial Boolean functions (S-Boxes) with strong cryptographic properties through the application of heuristic algorithms. We propose a (new) cost function for increasing performance of nonlinear vectorial Boolean functions generation. Also, we revisit the Spectral-linear and Spectral-difference methods and present a modification of these methods, derived from some new results related to the linear and differential characteristic of S-Boxes in the optimization of these properties. Combining the proposed modification with the assistance of some cost functions we provide some 8-bit S-Boxes having comparable properties with the best reported in public literature by using the heuristic approach, but (the proposed modification) outperform, in terms of the average number of S-Boxes evaluated to evolve the cryptographic parameters, most of the results given in the state-of-the-art.

**Keywords:** S-Box, nonlinearity, differential uniformity, heuristic algorithms.

## 1 Introduction

In Cryptography, many symmetric primitives are often iterations of several rounds. Each round, which must depend on the key, consists of a confusion layer and a diffusion layer. The confusion layers are usually formed by vectorial Boolean functions (S-Boxes) to provide nonlinear relationship between the input bits and the output bits while the diffusion layers are formed by global linear mappings mixing the output of the different S-Boxes.

The design criteria for S-Boxes, depending on the synthesis strategy of the cryptographic primitive, evolve over time. Each security requirement corresponds to an specific goal of resisting certain cryptographic attacks,



which use (most of them) the linear, differential and algebraic properties of S-Boxes.

The known methods for generating S-Boxes can be divided into four main classes: algebraic constructions, pseudo-random generation, heuristic techniques and constructions from small to large S-Boxes. Each approach has its advantages and disadvantages respectively (see, Section 4). In this article we particularly focus on the algebraic aspects of some heuristic methods, used to obtain cryptographically strong S-Boxes.

We propose a (new) cost function for increasing performance of non-linear vectorial Boolean functions generation. Several articles have been taken the advantage of the linear and differential characteristic of S-Boxes for optimizing its basic cryptographic properties and in this fashion, we present two efficient optimization methods derived from some new results related to the linear and differential characteristic of S-Boxes, which can be used for constructing cryptographically strong S-Boxes.

The article is structured as follows: In Section 2 we give some basic definitions and notations about (vectorial) Boolean functions. In Section 3, we present our design criteria for selecting S-Boxes with strong cryptographic properties. An updated overview of the most relevant results of the state-of-the-art by using the existing approaches is given in Section 4. In Section 5 we present new strategies for constructing cryptographically strong 8-bit S-Boxes which allows combine the proposed modifications of the spectral-linear and spectral-differential methods with the assistance of some cost functions. The result of our experiments and the corresponding comparison with other relevant methods are compiled in Section 6. In the final part we make conclusions of our research.

## 2 Basic definitions and notations

In this section, we introduce some basic notations and concepts needed to describe and analyse (vectorial) Boolean functions. Common mathematical notations are used. The notation  $\mathbb{Z}_+$  denotes the set of positive integers. For  $n \in \mathbb{Z}_+$ ,  $\mathbb{Z}_n$  denotes the set  $\{0, 1, 2, \dots, n-1\}$ .

Let  $\mathbb{F}_2$  be a finite field of two elements. For any  $n \in \mathbb{Z}_+$  we denote by  $\mathbb{F}_2^n = \mathbb{F}_2 \times \dots \times \mathbb{F}_2$ , the vector space of dimension  $n$  with the components from the field  $\mathbb{F}_2$ , let  $\mathbf{0} = (0, 0, \dots, 0)$  be the null vector of  $\mathbb{F}_2^n$  and by  $\oplus$  we denote the addition operation of  $\mathbb{F}_2^n$ . By  $S(\mathbb{F}_2^n)$  we denote the *symmetric group* on  $\mathbb{F}_2^n$ . We use the notation  $\#A$  to denote the size of the set  $A$ , by  $\lfloor x \rfloor$  we denote the *floor function* of the real number  $x$  and is equal to

$\lfloor x \rfloor = \max\{m \in \mathbb{Z} \mid m \leq x\}$ . The *scalar product* of  $a, b \in \mathbb{F}_2^n$  is denoted by  $\langle a, b \rangle$  and defined as  $\langle a, b \rangle = \bigoplus_{i=0}^{n-1} a_i b_i \in \mathbb{F}_2$ .

For any  $n \in \mathbb{Z}_+$ , the vectors from  $\mathbb{F}_2^n$  can be interpreted as integers, such that the leftmost bits correspond to the most significant bits. Let  $u \in \mathbb{F}_2^n$ , then the same vector can be written as follows:

$$u = (u_0, \dots, u_{n-1}) \in \mathbb{F}_2^n, \quad \Leftrightarrow \quad u = \sum_{i=0}^{n-1} u_i 2^{n-1-i} \in \mathbb{Z}_n.$$

We define the indicator function as follows

$$\text{Ind}(x, y) = \begin{cases} 1, & \text{if } x = y; \\ 0, & \text{if } x \neq y. \end{cases}$$

From mathematical point of view, an S-Box is a mapping from  $\mathbb{F}_2^n$  to  $\mathbb{F}_2^m$ . In what follows, we shall consider the case  $m = n$  and will focus only on vectorial Boolean functions from  $S(\mathbb{F}_2^n)$ , representing any such permutation as a vector of Boolean functions:

$$\Phi = (f_0, \dots, f_{n-1}), f_i: \mathbb{F}_2^n \rightarrow \mathbb{F}_2, i = 0, 1, \dots, n-1. \quad (1)$$

Alternatively, any  $n$ -bit S-Box  $\Phi \in S(\mathbb{F}_2^n)$  can be specified by the so-called tabular representation as follows:

$$\Phi = \begin{pmatrix} 0 & 1 & \dots & i & \dots & 2^n - 1 \\ \Phi(0) & \Phi(1) & \dots & \Phi(i) & \dots & \Phi(2^n - 1) \end{pmatrix}.$$

But very often for simplicity, we represent S-Boxes by the vector of its values (the so-called Look-Up Table) using the following notation:

$$\text{LUT}(\Phi) = (\Phi(0), \Phi(1), \dots, \Phi(2^n - 1)), \text{ where } \Phi(x) \in \mathbb{F}_2^n.$$

For some fixed  $i = 0, 1, \dots, n-1$ , every Boolean function  $f_i$  can be written as a sum over  $\mathbb{F}_2$  of distinct  $t$ -order products of its arguments,  $0 \leq t \leq n-1$ ; which is called the *algebraic normal form* (in brief, ANF) of  $f_i$ . The degree of the ANF of a Boolean function  $f$  with  $n$ -variables is called the algebraic degree of  $f$ , denoted by  $d_{alg}(f)$  and defined as the maximum order of terms appeared in its ANF [7].

Functions  $f_i$  written in (1) are called *coordinate Boolean functions* of the S-Box  $\Phi$  and it is well known (see, [7, p. 112]) that most of the desirable cryptographic properties of  $\Phi$  can be defined in terms of their non-trivial linear combinations, also called the S-Box *component functions*, denoted by  $\Phi_b$ , and defined as  $\Phi_b = \langle b, \Phi \rangle = b_0 f_0 \oplus \dots \oplus b_{n-1} f_{n-1}$  where  $\mathbf{0} \neq b = (b_0, \dots, b_{n-1}) \in \mathbb{F}_2^n, b_i \in \mathbb{F}_2, i \in \{0, \dots, n-1\}$ .

**Definition 1** (Walsh transform). *The Walsh transform  $\mathcal{W}_f$  of a Boolean function  $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  is defined as:*

$$\begin{aligned} \mathcal{W}_f: \mathbb{F}_2^n &\rightarrow \mathbb{Z}, \\ \mathcal{W}_f(a) &= \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) \oplus \langle a, x \rangle}, \end{aligned} \quad (2)$$

*The multiset of all values of the Walsh transform of  $f$  is called the Walsh spectrum of  $f$ .*

For the Walsh transform of a Boolean function  $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  it's well known the following property:

**Lemma 1** (Parseval identity). *For any Boolean function  $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ ,*

$$\sum_{a \in \mathbb{F}_2^n} \mathcal{W}_f(a)^2 = 2^{2n}.$$

As far as the number of all coefficients  $\mathcal{W}_f(a)$  of a Boolean function  $f$  is  $2^n$ , we have immediately the following lemma

**Lemma 2.** *For any Boolean function  $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$ ,*

$$\max_{a \in \mathbb{F}_2^n} |\mathcal{W}_f(a)| \geq 2^{\frac{n}{2}}.$$

**Definition 2** (Nonlinearity). *The nonlinearity of a Boolean function  $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  is defined as:*

$$\mathcal{NL}(f) = 2^{n-1} - \frac{1}{2} \max_{a \in \mathbb{F}_2^n} |\mathcal{W}_f(a)|. \quad (3)$$

In addition to concept of nonlinearity of Boolean functions, another characteristic of the difference with affine functions was treated in [11] namely curvature of a Boolean function  $f$  in  $n$ -variables.

**Definition 3** (Curvature). *Let  $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  be a Boolean function in  $n$ -variables. The curvature of  $f$ , denoted by  $\text{curv}(f)$ , is defined as*

$$\text{curv}(f) = \sum_{a \in \mathbb{F}_2^n} |\mathcal{W}_f(a)|. \quad (4)$$

As showed in [11], the curvature of a Boolean function of  $n$ -variables has the following bounds

$$2^n \leq \text{curv}(f) \leq 2^{\frac{3n}{2}}, \quad (5)$$

where the lower bound becomes an equality if and only if  $f$  is affine and the upper bound is achieved only when  $f$  is Bent. The curvature parameter (which is affine invariant) is useful tool for characterizing “how close” is a  $n$ -variables Boolean function  $f$  to being linear (or Bent) and in some sense this parameter can indicate some insight about the nonlinearity of  $f$ . In this article we discuss some aspects related to the notion of the curvature of Boolean functions and its application in the optimization of the property of nonlinearity.

From a cryptographic point of view S-Boxes with small values of Walsh coefficients offer better resistance against linear attacks [7].

Using the Lemma 1 we obtain the following lower bound for the sum of the  $k$  - power of the absolute values of Walsh coefficients:

**Lemma 3.** *For any Boolean function  $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  and any  $k \in \mathbb{N}$ ,  $k \geq 2$*

$$\sum_{a \in \mathbb{F}_2^n} |\mathcal{W}_f(a)|^k \geq \left\lfloor \frac{k}{2} \right\rfloor \cdot 2^{2n}.$$

*Proof.* Is not difficult to see that

$$\sum_{a \in \mathbb{F}_2^n} |\mathcal{W}_f(a)|^k \geq \sum_{a \in \mathbb{F}_2^n} (\mathcal{W}_f(a)^2)^{\lfloor \frac{k}{2} \rfloor} \geq \left\lfloor \frac{k}{2} \right\rfloor \cdot \sum_{a \in \mathbb{F}_2^n} \mathcal{W}_f(a)^2 = \left\lfloor \frac{k}{2} \right\rfloor \cdot 2^{2n}.$$

□

**Definition 4** (Linear Approximation table (LAT)). *Let  $\Phi: \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ . The linear approximation table (LAT) of  $\Phi$  is the mapping*

$$\begin{aligned} \text{LAT}_\Phi &: \mathbb{F}_2^n \times \mathbb{F}_2^n \rightarrow \mathbb{Q}, \\ \text{LAT}_\Phi[a, b] &= \frac{2}{2^n} \cdot \#\{x \in \mathbb{F}_2^n \mid \langle a, x \rangle = \langle b, \Phi(x) \rangle\} - 1 = \frac{\mathcal{W}_{\Phi_b}(a)}{2^n}. \end{aligned} \quad (6)$$

$\text{LAT}_\Phi$  naturally defines a  $2^n \times 2^n$  matrix over  $\mathbb{Q}$ .

**Definition 5** (Linearity). *Let  $\Phi: \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  be a  $n$ -bit S-Box of  $S(\mathbb{F}_2^n)$ . The linearity of  $\Phi$  is denoted by  $\mathcal{L}(\Phi)$  and can be defined as follows:*

$$\mathcal{L}(\Phi) = \max_{a, b \in \mathbb{F}_2^n \setminus \{0\}} |\text{LAT}_\Phi[a, b]|. \quad (7)$$

Alternatively, the *nonlinearity* of  $\Phi$  can be defined as:

$$\mathcal{NL}(\Phi) = 2^{n-1}(1 - \mathcal{L}(\Phi)) = 2^{n-1} - \frac{1}{2} \max_{a, b \in \mathbb{F}_2^n \setminus \{0\}} |\mathcal{W}_{\Phi_b}(a)|. \quad (8)$$

**Definition 6** (Difference distribution table (DDT)). *Let  $\Phi : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ . The difference distribution table (DDT) of  $\Phi$  is the mapping*

$$\begin{aligned} \text{DDT}_\Phi : \mathbb{F}_2^n \times \mathbb{F}_2^n &\rightarrow \mathbb{Q}_{\geq 0}, \\ \text{DDT}_\Phi[a, b] &= \frac{1}{2^n} \cdot \# \{x \in \mathbb{F}_2^n \mid \Phi(x \oplus a) \oplus \Phi(x) = b\}. \end{aligned} \quad (9)$$

$\text{DDT}_\Phi$  naturally defines a  $2^n \times 2^n$  matrix over  $\mathbb{Q}_{\geq 0}$ .

**Definition 7** (Differential Uniformity). *Let  $\Phi : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ . The differential uniformity of  $\Phi$  is denoted by  $\delta(\Phi)$  and is given by:*

$$\delta(\Phi) = \max_{a, b \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}} \text{DDT}_\Phi(a, b).$$

The resistance offered by an S-Box against differential attacks is related by the highest value of  $\delta$ , for this reason S-Boxes must have a small value of  $\delta$ -uniformity for a sufficient level of protection against this type of attacks (see, [7, 6]).

According to [29] we define the linear and the differential spectra of the permutation  $\Phi$  as follows.

For  $\Phi \in S(\mathbb{F}_2^n)$  and numbers  $p_1 \in P_{n-1}$  and  $p_2 \in P_{n-2}$ , let

$$P_j = \left\{ \frac{i}{2^j} \mid i = 0, \dots, 2^j \right\}, \#P_j = 2^j + 1, j \in \{n-2, n-1\},$$

then we define the next sets as follows

$$D(\Phi, p_1) = \left\{ (a, b) \in \mathbb{F}_2^n \times \mathbb{F}_2^n \setminus \{\mathbf{0}, \mathbf{0}\} \mid \text{DDT}_\Phi[a, b] = p_1 \right\},$$

and

$$L(\Phi, p_2) = \left\{ (a, b) \in \mathbb{F}_2^n \times \mathbb{F}_2^n \setminus \{\mathbf{0}, \mathbf{0}\} \mid |\text{LAT}_\Phi[a, b]| = p_2 \right\}.$$

**Definition 8** (Differential spectrum). *The differential spectrum of an S-Box  $\Phi \in S(\mathbb{F}_2^n)$  is defined as:*

$$D(\Phi) = \left\{ (p_1, \#D(\Phi, p_1)) \mid p_1 \in P_{n-1} \right\} \quad (10)$$

**Definition 9** (Linear spectrum). *The linear spectrum of an S-Box  $\Phi \in S(\mathbb{F}_2^n)$  is defined as:*

$$L(\Phi) = \left\{ (p_2, \#L(\Phi, p_2)) \mid p_2 \in P_{n-2} \right\} \quad (11)$$

**Definition 10** (Minimum degree). *The minimum algebraic degree (often called the minimum degree) of an  $n$ -bit S-Box  $\Phi$ , denoted by  $d_{\min}(\Phi)$ , is defined as the minimum algebraic degree of all the component functions, that is*

$$d_{\min}(\Phi) = \min_{\mathbf{0} \neq b \in \mathbb{F}_2^n} d_{\text{alg}}(\Phi_b(x)), \quad (12)$$

It is well-known that for any permutation  $\Phi \in S(\mathbb{F}_2^n)$ ,  $d_{min}(\Phi) \leq d_{alg}(\Phi)$  and these parameters are upper bounded by  $n - 1$  (see, [7]). In general, S-Boxes should have high values of  $d_{min}(\cdot)$ ,  $d_{alg}(\cdot)$  because S-Boxes with low values of these parameters are susceptible to algebraic attack, higher-order differential, interpolation, cube attacks, *etc* (see, [7, 13]).

**Definition 11** ( $r_{\Phi}^{(i)}$ -parameter). For  $i > 0$  the  $r_{\Phi}^{(i)}$  parameter of an  $n$ -bit S-Box  $\Phi$  is defined as

$$r_{\Phi}^{(i)} = \dim H_{\Phi}^{(i)}, \quad (13)$$

where

$$H_{\Phi}^{(i)} = \left\{ p \in \mathbb{F}_2[z_1, \dots, z_{2n}] \mid \forall x \in \mathbb{F}_2^n, p(x, \Phi(x)) = 0, 0 < d_{alg}(p) \leq i \right\}.$$

**Definition 12** ( $r_{\Phi}$ -parameter). The  $r_{\Phi}$ -parameter of an  $n$ -bit S-Box  $\Phi$  is defined as

$$r_{\Phi} = \min \left\{ i \mid r_{\Phi}^{(i)} > 0 \right\}. \quad (14)$$

It is well-known that there exists certain methods of analysis of block ciphers (see, [9]), exploiting the existence of polynomial relations involving the input  $x$  to the S-box  $\Phi$  and its output  $\Phi(x)$  and in order to increase the strength of a block cipher against these methods we need minimize parameter  $r_{\Phi}^{(i)}$ ,  $i = r_{\Phi}, \dots, n$  and maximize parameters  $d_{min}(\Phi)$  and  $r_{\Phi}$  (see, for example, [26, 29]).

It should be pointed that in [7, 37] parameter  $r_{\Phi}$  (defined slightly different) is called graph algebraic immunity of  $\Phi$  and in these references is denoted by  $AI_{gr}(\Phi)$ .

### 3 General S-Box Design Criteria

Our goal, is to find permutations having high values of its basic cryptographic parameters that satisfy the following criteria:

1. Maximum value of minimum degree;
2. Maximum value of  $r_{\Phi}$  with the minimum value of  $r_{\Phi}^{(i)}$ ;
3. Minimum value of  $\delta$ -uniformity limited by parameter listed below;
4. Maximum value of nonlinearity limited by parameter listed below.

In our case, when  $n = 8$  a cryptographically strong S-Box  $\Phi$  should satisfy the following

Main cryptographic criteria for selecting S – Boxes of  $S(\mathbb{F}_2^8)$  :

- $d_{min}(\Phi) = 7$ ;
- $r_{\Phi} = 3$  with  $r_{\Phi}^{(3)} = 441$ ;
- $\delta_{\Phi} \leq \frac{8}{256}$ ;
- $\mathcal{NL}(\Phi) \geq 100$ .

Our design criteria are basically the same as those included in the target set of criteria for the Gradient descent method [26]. We don't include the absence of fixed point criteria because it can be easily achieved by using the so-called linear (resp. affine) equivalence for S-Boxes<sup>a</sup>.

## 4 Existing approaches for constructing S-Boxes with high values of its basic cryptographic parameters

As stated at beginning of this work, the existing methods for constructing S-Boxes can be divided into four main classes: algebraic constructions, pseudo-random generation, constructions from small to large S-Boxes and heuristic techniques. In this section, we present an updated overview of the most relevant results in the state-of-the-art.

- **Algebraic constructions.** The best and well-known example are the so-called monomial permutations. Finite field inversion  $\mathcal{I} = x^{2^n-2}$  has best known differential uniformity and nonlinearity,  $\delta(\mathcal{I}) = \frac{4}{256}$ ,  $\mathcal{NL}(\mathcal{I}) = 112$  with the maximal value of minimum degree equal to 7. Nevertheless, these nonlinear transformations exhibit a very simple interpolation polynomial, and the number of such permutations is low, in addition further analysis has shown that this approach leads to existence of a system of polynomial equations with low degree (i.e.,  $r_{\mathcal{I}} = 2$  and  $r_{\mathcal{I}}^{(2)} = 39$ ) and a “potential vulnerability” to algebraic attacks when using it as nonlinear layer in block ciphers.
- **Pseudo-random generation.** The S-Box entries by using this method are generated from a table of random numbers and then one simply test whether the resulting S-Box is good or not with respect to a target set of cryptographic criteria. Pseudo-random generation of S-Boxes also covers new generation techniques based on chaotic maps, elliptic curves and quantum inspired methods [12, 28, 32, 1, 23, 2, 4, 3]. Permutations produced by pseudo-random generation have poor cryptographic properties, for example, in the case of 8-bit S-Boxes, the highest value for nonlinearity found is 98-100 and the differential uniformity are in the range of  $\frac{10}{256}$  up to a value of  $\frac{16}{256}$ , which are rather

---

<sup>a</sup>Two  $n$ -bit S-Boxes  $\Phi_1$  and  $\Phi_2$  are linear (resp. affine) equivalent if there exist linear (resp. affine) mappings  $A_1, A_2$ , such that  $\Phi_2(x) = A_2(\Phi_1(A_1(x)))$ ,  $\forall x \in \mathbb{F}_2^n$ .



low compared to the values of 112 and  $\frac{4}{256}$  for the finite field inversion-based case. However, these S-Boxes exhibits a complex interpolation polynomial and maximal possible value of its graph algebraic immunity. Although we always can generate a lot of such permutations there is a small number of good S-Boxes by using this approach among all in the search space.

- **Heuristic techniques.** This approach is aiming at using heuristic optimization techniques involving the hill climbing method [31], the simulated annealing method [8], the genetic algorithm [39, 25, 35], method of gradient descent [26], Spectral-linear and Spectral-differential methods [29] or a combination of these in a process of iteratively improving a given S-Box or S-Boxes with respect to one or more properties [36, 33, 17, 5, 14, 20]. Considering the case when  $n = 8$ , the S-Boxes obtained by using this method exhibits the following cryptographic properties: differential uniformity and nonlinearity are up to  $\frac{6}{256}$  and 104, maximal values of minimum degree and graph algebraic immunity equals to 7 and 3 respectively, which are the reality of nowadays when designing S-Boxes by using heuristic techniques [29, 24]. These S-Boxes frequently have a complex interpolation polynomial and although there are a lot of such permutations is hard to find them.

In almost all research [8, 39, 25, 24, 34, 19] devoted to problem of constructing S-Boxes with high values of its basics cryptographic parameters by using this approach, the values of its Walsh-Transform is taken under consideration in the optimization process and, in most of cases (for not being absolute), the best results cannot be achieved without the assistance of the so-called cost functions. Cost functions help to describe the behavior of coefficients in the Walsh spectrum of S-Boxes. Hence, a well descriptive cost function will undoubtedly help to improve the final nonlinearity of S-Boxes. It should be pointed that S-Boxes produced by heuristic techniques cannot achieve differential uniformity and nonlinearity values close to algebraic constructions unless they are seeded with S-Boxes having optimal properties (see [25]), but if we require an specific property that are not covered by algebraic constructions such as high graph algebraic immunity or optimized theoretical DPA metrics then, heuristic techniques can generate S-Boxes with high values of graph algebraic immunity and good enough values of DPA metrics having at the same time almost optimal differential



uniformity and nonlinearity. These are precisely their main advantages compared to those produced by algebraic constructions.

- **Constructions from small to large S-Boxes.** The idea of this method is to build larger S-Boxes from smaller ones. Because S-Boxes must have good cryptographic properties so that a cryptographic primitive (as a block cipher, for example) using them have good cryptographic properties as well, it seems natural to use the cryptographic phenomenon, whereby the strength of the smaller component is propagated upwards to the larger object in backwards direction as well — the S-Box itself may be constructed from smaller subcomponents combined with small block cipher structures (Feistel, SPN, Misty or a Lai-Massey) and certain algebraic operations like bitwise exclusive-or, modular addition or finite field multiplication, which leads to the appearance of permutations with good values of its basic cryptographic parameters. It should be pointed that when using this method we can use or combine all the previous approaches. There are a several reasons for constructing S-Boxes from smaller ones, for example: for table-based software implementations the table s are smaller, for hardware implementations the gate count is lower, for bit-sliced software implementation the instructions count is lower, for vectorized implementation, small S-Boxes can use vector permutations. In many cases, implementing several small S-Boxes requires less resources than implementing a large one. Therefore, constructing S-Boxes from smaller ones can reduce the implementation cost. The best S-Boxes (in term of its cryptographic properties) obtained by using this approach are given in [10, 16] and they attain the current record in the design of 8-bit S-Boxes with suboptimal properties.

## 5 New strategies for constructing 8-bit S-Boxes through optimization algorithms

Generally speaking, most of heuristic methods for optimizing the cryptographic properties of S-Boxes are based on two class of operations, swap and crossover<sup>b</sup> operations [15]. Some examples of successful application of these operators can be found in [26, 39, 25, 35, 29, 36, 33, 17, 5, 14, 20, 34, 19].

---

<sup>b</sup>In evolutionary computation, **crossover**, also called **recombination**, is a genetic operator used to combine the genetic information of two parents to generate a new offspring.

Following [21, Definition 9, p. 17] we define the *product* of two permutations  $\Phi_1, \Phi_2 \in S(\mathbb{F}_2^n)$  as  $(\Phi_2 \cdot \Phi_1)(x) = \Phi_1(\Phi_2(x))$ ,  $\forall x \in \mathbb{F}_2^n$ . In this section we are interested in swap operations (called transpositions, from an algebraic point of view) on S-Boxes, i.e., a class of transformations which convert the S-Box  $\Phi \in S(\mathbb{F}_2^n)$  having  $\text{LUT}(\Phi) = (\Phi(0), \dots, \Phi(i), \dots, \Phi(j), \dots, \Phi(2^n - 1))$  into a new S-Box  $\Phi' \in S(\mathbb{F}_2^n)$  with  $\text{LUT}(\Phi') = (\Phi(0), \dots, \Phi(j), \dots, \Phi(i), \dots, \Phi(2^n - 1))$ , where  $0 \leq i < j \leq 2^n - 1$ . The whole process can be synthesized using the group operation “.” of  $S(\mathbb{F}_2^n)$  as follows. Let  $(i, j)$  be a transposition of  $S(\mathbb{F}_2^n)$ , then the result of applying a swap operation to  $\Phi$ , yields a new S-Box  $\Phi' \in S(\mathbb{F}_2^n)$  given by the following relation<sup>c</sup>  $\Phi' = (i, j) \cdot \Phi$ , which in more details, can be defined as follows

$$\Phi'(x) = \begin{cases} \Phi(x), & \text{if } x \neq i, j; \\ \Phi(j), & \text{if } x = i; \\ \Phi(i), & \text{if } x = j. \end{cases}$$

With respect to the linear and differential characteristics of permutations multiplied by one transposition, it should be pointed that this issue has been studied in [40], where the authors showed that for two S-Boxes  $\Phi, \Phi' \in S(\mathbb{F}_2^n)$  related as follows  $\Phi' = (i, j) \cdot \Phi$ , the following relations holds:

- $|\mathcal{L}(\Phi') - \mathcal{L}(\Phi)| \leq 4 \cdot 2^{-n}$ ;
- $|\delta(\Phi') - \delta(\Phi)| \leq 4 \cdot 2^{-n}$ .

These relation tell us that, multiplying some permutation by a transposition, the (non)linear and differential properties of the resulting function will not change too much and in some cases they could be improved. However, improving these characteristics by using this approach is not a trivial task due to the time complexity required for optimizing such cryptographic properties.

We try to capture the behavior of the linear and differential characteristics of one S-Box before and after the multiplication by some transposition(s)  $(i, j) \in S(\mathbb{F}_2^n)$  by means of their linear and differential spectrum. Then, we check the fitness conditions of the new S-Box ( $\Phi'$ ) generated through the multiplication with respect to the original S-Box ( $\Phi$ ) using a function to decide whether  $\Phi'$  present better linear and differential characteristics than  $\Phi$ .

---

<sup>c</sup>In others words, when multiplying a transposition by a permutation, we perform an exchange of the values of a pair of inputs.

## 5.1 Selection of cost functions

As stated in Section 4 the best results achieved by optimization algorithms w.r.t nonlinearity are obtained by using of some cost functions related to the linear spectrum of S-Boxes. Example of these cost functions are presented in [8, 34, 19] and applied in [25, 17, 20, 24] with very good results. Furthermore, the performance of various optimization algorithms which use the aforementioned cost functions is analyzed in [34, 19], together with the correlation analysis of these functions towards nonlinearity property provided in [19]. Next, we present the definition of all these functions. Moreover, we refer the interested lecturer to read further about them in [8, 39, 34, 19].

**Definition 13** (The  $\mathcal{WHS}$ -fitness function). *Let  $\Phi : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  be a S-Box and  $X, R$  — two real-valued parameters. The fitness function  $\mathcal{WHS}$  presented in [8] is defined as:*

$$\mathcal{WHS}(\Phi) = \sum_{\mathbf{0} \neq b \in \mathbb{F}_2^n} \sum_{a \in \mathbb{F}_2^n} ||\mathcal{W}_{\Phi_b}(a)| - X|^R$$

**Definition 14** (The  $\mathcal{PCF}$ -fitness function ). *Let  $\vec{\mathcal{H}}$  be the histogram of the absolute values of the Walsh spectrum of an S-Box  $\Phi$  such that  $\vec{\mathcal{H}}$  is represented by a vector having in the  $i$ -th component the number of coefficients associated to absolute value  $4i$  from the Walsh spectrum of  $\Phi$  and let  $l$  indicate the last component of the vector with nonzero value. The fitness function  $\mathcal{PCF}(\Phi)$  presented in [34] is defined as:*

$$\mathcal{PCF}(\Phi) = \sum_{i=0}^{N-1} \frac{\vec{\mathcal{H}}(l-i)}{2^i}$$

where  $1 \leq N \leq l$ . In our case of study we maintain the configuration proposed in [34] which have  $N = 10$ .

**Definition 15** (The  $\mathcal{WCF}$ -fitness function ). *Let  $\Phi : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  be an S-Box and  $\mathcal{V} = \{0, 4, \dots, 2^{\frac{n}{2}+1}\}$ . The fitness function  $\mathcal{WCF}$  presented in [19] is defined as:*

$$\mathcal{WCF}(\Phi) = \sum_{\mathbf{0} \neq b \in \mathbb{F}_2^n} \sum_{a \in \mathbb{F}_2^n} \prod_{z \in \mathcal{V}} ||\mathcal{W}_{\Phi_b}(a)| - z|$$

### 5.1.1 On some bounds relating the curvature with some cost functions

When the curvature of a Boolean function  $f$  is near to the upper bound of inequality (3) then  $f$  is close to Bent functions which have maximum

nonlinearity. Furthermore, a Bent function satisfies that the absolute value of all the coefficients in its Walsh-Hadamard spectrum is equal to  $2^{\frac{n}{2}}$ . Based in these results, one can define the following for any Boolean function  $f$ :

$$\mathcal{C}_f = \sum_{x \in \mathbb{F}_2^n} ||\mathcal{W}_f(x)| - 2^{\frac{n}{2}}| \quad (15)$$

It is straightforward to notice that in (15) the minimum value of  $\mathcal{C}_f$  is achieved if  $f$  is Bent and it is equal to 0. Although for balanced Boolean functions we have that  $\mathcal{C}_f > 0$ , lower values of  $\mathcal{C}_f$  indicates that the coefficients in the Walsh-Hadamard spectrum of  $f$  are close to  $2^{\frac{n}{2}}$  and therefore one may expect their nonlinearity to be better. However, this is not a sufficient condition for the nonlinearity. In example, let us consider the following 6-variable Boolean functions written in hexadecimal notation:

$$f_1 = 0xf48091a920bf57d7 \quad f_2 = 0x1003b517aceb317f$$

With respect to nonlinearity we have  $\mathcal{NL}(f_2) = 24$  and  $\mathcal{NL}(f_1) = 22$  which clearly show that  $f_2$  is better than  $f_1$ . However, the values of  $\mathcal{C}_f$  indicates the opposite scenario given that  $\mathcal{C}_{f_1} = 240$  and  $\mathcal{C}_{f_2} = 256$ . Thus, we reformulate  $\mathcal{C}_f$  to add more importance to coefficients farther from  $2^{\frac{n}{2}}$

$$\mathcal{C}_f^k = \sum_{a \in \mathbb{F}_2^n} ||\mathcal{W}_f(a)| - 2^{\frac{n}{2}}|^k, \quad (16)$$

where  $k \in \mathbb{Z}, k \geq 2$ . For the particular case of functions  $f_1$  and  $f_2$ , if we set  $k = n = 6$ , then  $\mathcal{C}_{f_1}^6 = 9240576$  and  $\mathcal{C}_{f_2}^6 = 4325376$  which is in correspondence to the relation between the nonlinearity values of  $f_1$  and  $f_2$ . The next proposition establishes a lower bound on the value of  $\mathcal{C}_f^k$  which is useful in the case of balanced Boolean functions.

**Proposition 1.** *For any Boolean function  $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$  and any  $k \in \mathbb{N}, k \geq 2$*

$$\mathcal{C}_f^k \geq 2^n \left| k \cdot 2^{\frac{k-1}{2}} \cdot \text{curv}(f) - 2^n (2^{\frac{n}{2}} + 1)^{k-2} - 2^{\frac{k+2}{2}} \right|.$$

*Proof.* From the definition of the cost function  $\mathcal{C}_f^k$  and the binomial theorem we obtain that

$$\begin{aligned} \mathcal{C}_f^k &= \sum_{a \in \mathbb{F}_2^n} ||\mathcal{W}_f(a)| - 2^{\frac{n}{2}}|^k = \sum_{a \in \mathbb{F}_2^n} \left| \sum_{i=0}^k \binom{k}{i} (-2^{\frac{n}{2}})^i |\mathcal{W}_f(a)|^{k-i} \right| \geq \\ &\geq \left| \sum_{a \in \mathbb{F}_2^n} \sum_{i=0}^k \binom{k}{i} (-2^{\frac{n}{2}})^i |\mathcal{W}_f(a)|^{k-i} \right| = \left| \sum_{i=0}^k \binom{k}{i} (-2^{\frac{n}{2}})^i \sum_{a \in \mathbb{F}_2^n} |\mathcal{W}_f(a)|^{k-i} \right| = \end{aligned}$$

$$= \left| (-1)^k \cdot 2^{\frac{n(k+2)}{2}} + k \cdot (-1)^{k-1} \cdot 2^{\frac{n(k-1)}{2}} \cdot \text{curv}(f) + S \right|,$$

where by  $S$  is denoted the sum of the form

$$S = \sum_{i=0}^{k-2} \binom{k}{i} (-2^{\frac{n}{2}})^i \sum_{a \in \mathbb{F}_2^n} |\mathcal{W}_f(a)|^{k-i}.$$

Considering the lemma 2 we obtain that

$$\begin{aligned} S &\geq 2^{2n} \sum_{i=0}^{k-2} \binom{k}{i} (-2^{\frac{n}{2}})^i \left\lfloor \frac{k-i}{2} \right\rfloor \geq 2^{2n} (-1)^k \sum_{i=0}^{k-2} \binom{k}{i} 2^{\frac{ni}{2}} = \\ &= 2^{2n} (-1)^k (2^{\frac{n}{2}} + 1)^{k-2} \end{aligned}$$

and

$$\begin{aligned} \mathcal{C}_f^k &\geq \left| (-1)^{k-1} 2^n \left( k \cdot 2^{\frac{k-1}{2}} \cdot \text{curv}(f) - 2^n (2^{\frac{n}{2}} + 1)^{k-2} - 2^{\frac{k+2}{2}} \right) \right| = \\ &= 2^n \left| k \cdot 2^{\frac{k-1}{2}} \cdot \text{curv}(f) - 2^n (2^{\frac{n}{2}} + 1)^{k-2} - 2^{\frac{k+2}{2}} \right|. \end{aligned}$$

□

Finally, as the component functions of  $n$ -bit S-Boxes are  $n$ -variable Boolean functions, the notion of  $\mathcal{C}_f^k$  can be easily extended such that for any S-Box  $\Phi : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ :

$$\mathcal{C}_\Phi^k = \sum_{\mathbf{0} \neq b \in \mathbb{F}_2^n} \sum_{a \in \mathbb{F}_2^n} \left| |\mathcal{W}_{\Phi_b}(a)| - 2^{\frac{n}{2}} \right|^k. \quad (17)$$

**Proposition 2.** For any  $n$ -bit S-Box  $\Phi : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$  and any  $k \in \mathbb{N}, k \geq 2$

$$\mathcal{C}_\Phi^k \geq 2^n \left| k \cdot 2^{\frac{k-1}{2}} \sum_{\mathbf{0} \neq b \in \mathbb{F}_2^n} \text{curv}(\Phi_b) - (2^n - 1) \left( 2^n (2^{\frac{n}{2}} + 1)^{k-2} - 2^{\frac{k+2}{2}} \right) \right|.$$

*Proof.* From the definition of the cost function  $\mathcal{C}_\Phi^k$  and the proposition 1 we obtain that

$$\begin{aligned} \mathcal{C}_\Phi^k &= \sum_{b \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}} \sum_{a \in \mathbb{F}_2^n} \left| |\mathcal{W}_{\Phi_b}(a)| - 2^{\frac{n}{2}} \right|^k = \sum_{b \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}} \mathcal{C}_{\Phi_b}^k \geq \\ &\geq 2^n \sum_{b \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}} \left| k \cdot 2^{\frac{k-1}{2}} \cdot \text{curv}(\Phi_b) - 2^n (2^{\frac{n}{2}} + 1)^{k-2} - 2^{\frac{k+2}{2}} \right| \geq \end{aligned}$$

$$\begin{aligned} &\geq 2^n \left| \sum_{b \in \mathbb{F}_2^n \setminus \{0\}} k \cdot 2^{\frac{k-1}{2}} \cdot \text{curv}(\Phi_b) - 2^n (2^{\frac{n}{2}} + 1)^{k-2} - 2^{\frac{k+2}{2}} \right| = \\ &= 2^n \left| k \cdot 2^{\frac{k-1}{2}} \sum_{b \in \mathbb{F}_2^n \setminus \{0\}} \text{curv}(\Phi_b) - (2^n - 1) \left( 2^n (2^{\frac{n}{2}} + 1)^{k-2} - 2^{\frac{k+2}{2}} \right) \right|. \end{aligned}$$

□

In Table 1 we show the average number of solution evaluations of 1000 experiments using the hill climbing method described in *Appendix C* of [19] to obtain a nonlinearity value equal 100 for the cost function presented in this section and other relevant cost functions in the literature. For better comprehension, we include the results achieved for the values of parameter  $k$  ranging from  $k = 2$  to  $k = 8$ . To establish a fair ground to the later comparison on the results we calculate the average nonlinearity of the results presented in Table 1 of [29] for  $10^{10}$  random generated S-Boxes (92.69), which implies that the common nonlinearity value of a random generated S-Box is equal to 92. Therefore, one can assume this value as the initial nonlinearity value of the tested S-Boxes.

It should be pointed that the algorithm speeds up its performance when the value of parameter  $k$  increases. In addition, the average number of S-Boxes evaluated using the versions of  $\mathcal{C}_\Phi^k$  with parameter  $k \geq 7$  are comparable to the cost functions from [34, 19] which have the best average convergence ratios for different optimization algorithms. Also notice that the cost function presented in [29] for the linear characteristic of S-Boxes is the initial cost function analyzed in [34], which was later reformulated to the function  $\mathcal{PCF}$  in order to reduce the number of solutions that need to be evaluated by different heuristic algorithms to obtain S-Boxes with high nonlinearity. Furthermore,  $\mathcal{C}_f^k$  and  $\mathcal{C}_\Phi^k$  are special cases of Clark's cost function from [8] where the value of  $X$  is substituted based on the notion of the curvature of Boolean functions and  $R = k$ .

It is interesting that in [39] the author heuristically search the values of parameters  $X$  and  $R$  such that  $R \in [3, 25]$  and  $X \in [-25, 25]$  arriving at the conclusion that the best set of parameters for bijective 8-bit S-Boxes has  $R = 7$  and  $X = 21$ . However, the data in Table 1 contradict these results. Notice that using  $\mathcal{C}_\Phi^7$  and  $\mathcal{C}_\Phi^8$ , *i.e.* ( $R = 7, X = 16$ ) and ( $R = 8, X = 16$ ), the hill climbing method outperforms the configuration proposed in [39].

Function	$\mathcal{C}_{\Phi}^2$	$\mathcal{C}_{\Phi}^3$	$\mathcal{C}_{\Phi}^4$	$\mathcal{C}_{\Phi}^5$	$\mathcal{C}_{\Phi}^6$	$\mathcal{C}_{\Phi}^7$	$\mathcal{C}_{\Phi}^8$	$\mathcal{PCF}$ [34]	$\mathcal{WCF}$ [19]	$\mathcal{WHS}_{3,4}$ [8]	$\mathcal{WHS}_{7,21}$ [39]
Average	37461	1102	345	221	175	156	142	154	140	737	171

Table 1: Performance of the hill climbing method from [19] to obtain nonlinearity 100 using different values of parameter  $k$  for cost function  $\mathcal{C}_{\Phi}^k$  and some representative cost functions in the referred literature.

## 5.2 Adaptation of the Spectral-linear and Spectral-differential methods to improve the basic cryptographic properties of S-boxes

In the experiments conducted in this work, we use a variation of the spectral differential and the spectral linear methods from [29]. In our adaptation of the methods we use parameters  $w_1 = w_2 = 1$ , i.e. only the best S-Box is passed to the next round of the algorithm instead of a list containing the best  $w_i$  S-Boxes ( $i = 1, 2$ ). Each time the algorithm produce a new solution  $\Phi'$  from the round S-Box  $\Phi$  we evaluate the superiority of  $\Phi'$  over  $\Phi$  through the following binary function:

$$fitness_{spl}(\Phi, \Phi') = \begin{cases} 1, & \text{if } (\delta(\Phi') \leq \delta(\Phi) \text{ and } \mathcal{L}(\Phi') < \mathcal{L}(\Phi)) \text{ or} \\ & (\delta(\Phi') \leq \delta(\Phi) \text{ and } \mathcal{L}(\Phi') = \mathcal{L}(\Phi) \text{ and } \mathcal{F}_{\Phi'}^1 < \mathcal{F}_{\Phi}^1); \\ 0, & \text{otherwise.} \end{cases} \quad (18)$$

$$fitness_{spd}(\Phi, \Phi') = \begin{cases} 1, & \text{if } (\delta(\Phi') < \delta(\Phi) \text{ and } \mathcal{L}(\Phi') \leq \mathcal{L}(\Phi)) \text{ or} \\ & (\delta(\Phi') = \delta(\Phi) \text{ and } \mathcal{L}(\Phi') \leq \mathcal{L}(\Phi) \text{ and } \mathcal{F}_{\Phi'}^2 < \mathcal{F}_{\Phi}^2); \\ 0, & \text{otherwise.} \end{cases} \quad (19)$$

where  $\mathcal{F}_{\Phi}^1$  and  $\mathcal{F}_{\Phi}^2$  are the cost functions related to nonlinearity and differential uniformity respectively. With respect to  $\mathcal{F}_{\Phi}^1$  we use the cost function described in Section 5.1.1. For function  $\mathcal{F}_{\Phi}^2$  we use a variant the cost function over the differential spectrum of S-Boxes introduced by Ivanov *et al.* in [24] which we define as:

$$\mathcal{F}_{\Phi}^2 = \sum_{\mathbf{0} \neq a \in \mathbb{F}_2^n} \sum_{b \in \mathbb{F}_2^n} ((2^n \cdot \text{DDT}_{\Phi}[a, b])(2^n \cdot \text{DDT}_{\Phi}[a, b] - 2)(2^n \cdot \text{DDT}_{\Phi}[a, b] - 4))^2.$$

Notice that in  $\mathcal{F}_{\Phi}^2$  the values of the  $\text{DDT}_{\Phi}$  lower than  $\frac{6}{2^n}$  are not taken into account in the calculus of the function. Moreover, if  $\mathcal{F}_{\Phi}^2 = 0$  one can ensure that  $\delta(\Phi) \leq \frac{4}{2^n}$ .



### 5.2.1 Some issues related to the Spectral-linear and Spectral-differential methods

In this subsection, we perform an analysis of some issues related to the Spectral-linear and Spectral-differential methods for concrete 8-bit S-Boxes. Our results show the existence of some intermediate S-Boxes that don't meet the conditions given in Step 2 of these methods as indicated in [29]<sup>d</sup>.

First of all, it should be pointed that when multiplying a permutation by a transposition, the author of [29] perform an exchange of the values of a pair of outputs, in what follows we denote a such multiplication by  $\odot$ , defining the result of  $\Phi'' = (i, j) \odot \Phi$  as

$$\Phi''(x) = \begin{cases} \Phi(x), & \text{if } \Phi(x) \neq i, j; \\ j, & \text{if } \Phi(x) = i; \\ i, & \text{if } \Phi(x) = j. \end{cases}$$

The next proposition shows the natural relation between operations “ $\odot$ ” and “ $\cdot$ ”.

**Proposition 3.** *Let  $\Phi$  be an  $n$ -bit S-Box from  $S(\mathbb{F}_2^n)$ . Then for any  $\Phi', \Phi'' \in S(\mathbb{F}_2^n)$  such that  $\Phi' = (i, j) \cdot \Phi$ ,  $\Phi'' = (i, j) \odot \Phi$ ,  $0 \leq i < j \leq 2^n - 1$ , the following relations holds*

$$\Phi'' = (\Phi^{-1}(i), \Phi^{-1}(j)) \cdot \Phi. \quad (20)$$

*Proof.* To prove the proposition is sufficient to note that  $\text{LUT}(\Phi'') = (\Phi(0), \dots, j, \dots, i, \dots, \Phi(2^n - 1)) = (\Phi(0), \dots, \Phi(j_0), \dots, \Phi(i_0), \dots, \Phi(2^n - 1))$ , where by  $i_0, j_0$  we denote the elements  $\Phi^{-1}(i), \Phi^{-1}(j)$  respectively.  $\square$

In Step 2 of both methods, permutations  $g'_{i,j}$  belonging to one of the lists

$$\begin{aligned} I'_{\text{SDM}} &= \left\{ (g'_{i,j}, \delta(g'_{i,j}), \#D(g'_{i,j}, \delta(g'_{i,j})), \mathcal{L}(g'_{i,j})) \right\}, \\ I'_{\text{SLM}} &= \left\{ (g'_{i,j}, \mathcal{L}(g'_{i,j}), \#L(g'_{i,j}, \mathcal{L}(g'_{i,j})), \delta(g'_{i,j})) \right\}, \end{aligned}$$

are defined as  $g'_{i,j} = (x, x') \odot g_i$  and  $g'_{i,j} = (y, y') \odot g_i$ , where  $i = 0, \dots, \#I' - 1$ ,  $x, x' \in X_{g_i}$ ,  $y, y' \in Y_{g_i}$ ,  $x < x'$ ,  $y < y'$ ,  $j = j(x, x')$ ,  $j = j(y, y')$  are injective mappings (transpositions  $(x, x')$ ,  $(y, y')$  have special properties). Depending on which method is used, the list  $I'_{\text{SDM}}, I'_{\text{SLM}}$  accepts

<sup>d</sup>We refer to the full version of the Spectral-linear and Spectral-differential methods for better understanding of our results.



only permutations  $g'_{i,j}$  for which the following relations holds,  $\delta(g'_{i,j}) \leq \delta(g_i)$ ,  $\mathcal{L}(g'_{i,j}) \leq \mathcal{L}(g_i)$  and  $\#D(g'_{i,j}, \delta(g'_{i,j})) < \#D(g_i, \delta(g_i))$  if  $\delta(g'_{i,j}) = \delta(g_i)$  (or  $\mathcal{L}(g'_{i,j}) \leq \mathcal{L}(g_i)$ ,  $\delta(g'_{i,j}) \leq \delta(g_i)$ , and  $\#L(g'_{i,j}, \mathcal{L}(g'_{i,j})) < \#L(g_i, \mathcal{L}(g_i))$  if  $\mathcal{L}(g'_{i,j}) = \mathcal{L}(g_i)$ ). These relations seems to be natural and crucial at each iteration of Step 2 of the algorithms, because only the best S-Boxes in terms of its basic cryptographic parameters (differential and linear) are included in these lists and as a result of this iterative process the improvement of these parameters can be expected.

Now, using the list of transpositions given in [29, Table 6, p. 114], denoted here by **Transp** and defined as **Transp** =  $\{(208, 194), (45, 108), (48, 192), (0, 100), (148, 159), (121, 163), (67, 247), (36, 99), (245, 159), (189, 0), (84, 230), (109, 85), (53, 10), (25, 10), (178, 217), (41, 180), (156, 179), (112, 103), (181, 145), (42, 48), (200, 88), (235, 208), (211, 122), (6, 152), (160, 121), (5, 36), (1, 124), (129, 170), (241, 104), (153, 189), (147, 172), (219, 105), (197, 181), (218, 43), (251, 83), (110, 183)\}$ , we have recreated (by using a SAGE [38] script given in Section 7) the computation of the characteristics  $\delta(\cdot), \mathcal{L}(\cdot), \#D(\cdot), \#L(\cdot)$  involved in Step 2 of the methods when optimizing the cryptographic properties of S-Box of the national standard of Russian Federation GOST R 34.11-2015 [22], denoted here by  $\pi_{\text{Kuz}}$ . Let define  $\pi_{\text{Kuz}}^{(i+1)}$  as follows,  $\pi_{\text{Kuz}}^{(i+1)} = (\text{Transp}[i - 1] \odot \dots \odot \text{Transp}[0]) \odot \pi_{\text{Kuz}}$ , where for each  $i = 0, 1, \dots, 35$ ,  $\text{Transp}[i] \in \text{Transp}$ .

Characteristics of the S – Box $\pi_{\text{Kuz}}$	
$\delta(\pi_{\text{Kuz}}) = \frac{8}{256}, \mathcal{L}(\pi_{\text{Kuz}}) = \frac{28}{128}, \#D(\pi_{\text{Kuz}}, \delta(\pi_{\text{Kuz}})) = 25, \#L(\pi_{\text{Kuz}}, \mathcal{L}(\pi_{\text{Kuz}})) = 14$	
Characteristics of the S – Box $\pi_{\text{Kuz}}^{(i+1)}$	
$\delta(\pi_{\text{Kuz}}^{(1)}) = \frac{8}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(1)}) = \frac{28}{128}, \#D(\pi_{\text{Kuz}}^{(1)}, \delta(\pi_{\text{Kuz}}^{(1)})) = 24, \#L(\pi_{\text{Kuz}}^{(1)}, \mathcal{L}(\pi_{\text{Kuz}}^{(1)})) = 12$	
$\delta(\pi_{\text{Kuz}}^{(2)}) = \frac{10}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(2)}) = \frac{30}{128}, \#D(\pi_{\text{Kuz}}^{(2)}, \delta(\pi_{\text{Kuz}}^{(2)})) = 1, \#L(\pi_{\text{Kuz}}^{(2)}, \mathcal{L}(\pi_{\text{Kuz}}^{(2)})) = 1$	
$\delta(\pi_{\text{Kuz}}^{(3)}) = \frac{10}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(3)}) = \frac{30}{128}, \#D(\pi_{\text{Kuz}}^{(3)}, \delta(\pi_{\text{Kuz}}^{(3)})) = 1, \#L(\pi_{\text{Kuz}}^{(3)}, \mathcal{L}(\pi_{\text{Kuz}}^{(3)})) = 1$	
$\delta(\pi_{\text{Kuz}}^{(4)}) = \frac{10}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(4)}) = \frac{30}{128}, \#D(\pi_{\text{Kuz}}^{(4)}, \delta(\pi_{\text{Kuz}}^{(4)})) = 1, \#L(\pi_{\text{Kuz}}^{(4)}, \mathcal{L}(\pi_{\text{Kuz}}^{(4)})) = 1$	
$\delta(\pi_{\text{Kuz}}^{(5)}) = \frac{10}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(5)}) = \frac{30}{128}, \#D(\pi_{\text{Kuz}}^{(5)}, \delta(\pi_{\text{Kuz}}^{(5)})) = 1, \#L(\pi_{\text{Kuz}}^{(5)}, \mathcal{L}(\pi_{\text{Kuz}}^{(5)})) = 2$	
$\delta(\pi_{\text{Kuz}}^{(6)}) = \frac{10}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(6)}) = \frac{32}{128}, \#D(\pi_{\text{Kuz}}^{(6)}, \delta(\pi_{\text{Kuz}}^{(6)})) = 1, \#L(\pi_{\text{Kuz}}^{(6)}, \mathcal{L}(\pi_{\text{Kuz}}^{(6)})) = 1$	
$\delta(\pi_{\text{Kuz}}^{(7)}) = \frac{10}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(7)}) = \frac{32}{128}, \#D(\pi_{\text{Kuz}}^{(7)}, \delta(\pi_{\text{Kuz}}^{(7)})) = 1, \#L(\pi_{\text{Kuz}}^{(7)}, \mathcal{L}(\pi_{\text{Kuz}}^{(7)})) = 1$	
$\delta(\pi_{\text{Kuz}}^{(8)}) = \frac{10}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(8)}) = \frac{30}{128}, \#D(\pi_{\text{Kuz}}^{(8)}, \delta(\pi_{\text{Kuz}}^{(8)})) = 1, \#L(\pi_{\text{Kuz}}^{(8)}, \mathcal{L}(\pi_{\text{Kuz}}^{(8)})) = 1$	
...	
$\delta(\pi_{\text{Kuz}}^{(34)}) = \frac{8}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(34)}) = \frac{26}{128}, \#D(\pi_{\text{Kuz}}^{(34)}, \delta(\pi_{\text{Kuz}}^{(34)})) = 2, \#L(\pi_{\text{Kuz}}^{(34)}, \mathcal{L}(\pi_{\text{Kuz}}^{(34)})) = 9$	
$\delta(\pi_{\text{Kuz}}^{(35)}) = \frac{6}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(35)}) = \frac{26}{128}, \#D(\pi_{\text{Kuz}}^{(35)}, \delta(\pi_{\text{Kuz}}^{(35)})) = 513, \#L(\pi_{\text{Kuz}}^{(35)}, \mathcal{L}(\pi_{\text{Kuz}}^{(35)})) = 1$	
$\delta(\pi_{\text{Kuz}}^{(36)}) = \frac{6}{256}, \mathcal{L}(\pi_{\text{Kuz}}^{(36)}) = \frac{24}{128}, \#D(\pi_{\text{Kuz}}^{(36)}, \delta(\pi_{\text{Kuz}}^{(36)})) = 526, \#L(\pi_{\text{Kuz}}^{(36)}, \mathcal{L}(\pi_{\text{Kuz}}^{(36)})) = 122$	

Table 2: Computation of characteristics  $\delta(\cdot), \mathcal{L}(\cdot), \#D(\cdot), \#L(\cdot)$  for  $\pi_{\text{Kuz}}$  and  $\pi_{\text{Kuz}}^{(i+1)}$ .

From Table 2 we can see that, no matters which algorithm could be used to improve the differential and linear properties of  $\pi_{\text{Kuz}}$ , permutation  $\pi_{\text{Kuz}}^{(2)}$  (and others) can not be included in the lists  $I'_{\text{SDM}}, I'_{\text{SLM}}$  because the values

of  $\delta(\pi_{\text{Kuz}}^{(2)})$ ,  $\mathcal{L}(\pi_{\text{Kuz}}^{(2)})$  are in conflict (in fact, they are worse) with relations  $\delta(g'_{i,j}) \leq \delta(g_i)$ ,  $\mathcal{L}(g'_{i,j}) \leq \mathcal{L}(g_i)$  and this is inconsistent with the description of the algorithms implementing the Spectral-differential and Spectral-linear methods given in [29]. However, one can note from this Table that the “special properties” of the transpositions may have a significant impact when reducing the size of sets  $\#D(g'_{i,j}, \delta(g'_{i,j}))$ ,  $\#L(g_i, \mathcal{L}(g_i))$ .

For the rest of S-Boxes compiled in [29, Tables 7-9, p. 114-115] we have detected the same conflict related with the actual values  $\delta(\cdot)$ ,  $\mathcal{L}(\cdot)$  and restrictions  $\delta(g'_{i,j}) \leq \delta(g_i)$ ,  $\mathcal{L}(g'_{i,j}) \leq \mathcal{L}(g_i)$ . We emphasize that we do not question the efficiency of these methods, in fact we have obtained 8-bit S-Boxes with the desired criteria by using these algorithms. Surprisingly, when reproducing some experiments (optimization of differential and linear parameters of some known S-Boxes) made by the author of [29] we have found some intermediate S-Boxes which do not satisfy the conditions given by the description of the Spectral-linear and Spectral-differential algorithms. These S-boxes have worse properties, but the final S-Box has the desired cryptographic parameters and thus in sets  $I'_{\text{SDM}}$ ,  $I'_{\text{SLM}}$  could live S-Boxes with poor differential and linear properties, and yet the algorithms may output an S-Box with better parameters than the initial permutation. However we believe that the special properties (which we don't know) of transpositions may play a crucial role in such scenarios.

### 5.2.2 New results derived from the Spectral-linear method

There is no doubt that the Spectral-linear method is proven to be effective towards the optimization of S-Boxes with nonlinearity up to 104. Nonetheless, it consumes a high number of solution evaluations to produce such results. To give the reader some clarity on this topic, using the next propositions to make some estimations on the number of new S-Boxes reviewed for each one of the permutations of the  $i$  –  $th$  iteration of the Spectral-linear method assuming that  $\#I = w_2 = 1$ . The results are valid for any way of multiplying permutations by transpositions, i.e., by performing an exchange of the values of a pair of inputs or swapping a pair of outputs.

**Proposition 4.** *Let  $a, b \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}$ , then for any  $\Phi \in S(\mathbb{F}_2^n)$  such that  $\mathcal{L}(\Phi) = |\text{LAT}_{\Phi}[a, b]|$ , the following relations hold*

$$\#Y_{\Phi} = \begin{cases} 2^{n-1}(1 - \mathcal{L}(\Phi)), & \text{if } \text{LAT}_{\Phi}[a, b] < 0; \\ 2^{n-1}(1 + \mathcal{L}(\Phi)), & \text{if } \text{LAT}_{\Phi}[a, b] > 0. \end{cases} \quad (21)$$

where  $Y_{\Phi} = \{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\}$ .

*Proof.* From the definition of linearity (7) we have  $\mathcal{L}(\Phi) = \max_{a,b \in \mathbb{F}_2^n \setminus \{0\}} |\text{LAT}_\Phi[a, b]|$ . Let us fix such  $a, b \in \mathbb{F}_2^n \setminus \{0\}$ , for which

$$\mathcal{L}(\Phi) = |\text{LAT}_\Phi[a, b]|.$$

It is not difficult to see that the vector space  $\mathbb{F}_2^n$  can be decomposed as follows

$$\mathbb{F}_2^n = \{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} \sqcup \{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \neq \langle b, \Phi(y) \rangle\}. \quad (22)$$

From (22) we obtain

$$2^n = \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} + \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \neq \langle b, \Phi(y) \rangle\},$$

which is equivalent to

$$1 = \frac{\#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} + \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \neq \langle b, \Phi(y) \rangle\}}{2^n}. \quad (23)$$

Now, substituting (23) in the following equality

$$\mathcal{L}(\Phi) = |\text{LAT}_\Phi[a, b]| = \left| \frac{2}{2^n} \cdot \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} - 1 \right|$$

we have that

$$\mathcal{L}(\Phi) = \left| \frac{\#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} - \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \neq \langle b, \Phi(y) \rangle\}}{2^n} \right|. \quad (24)$$

From (24), by using the definition of  $|x|$ ,  $x \in \mathbb{R}$ , we deduce that:

If  $\text{LAT}_\Phi[a, b] > 0$ , then

$$\begin{aligned} \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} &= 2^n \mathcal{L}(\Phi) + \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \neq \langle b, \Phi(y) \rangle\} \\ &= 2^n \mathcal{L}(\Phi) + 2^n - \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\}, \end{aligned}$$

hence  $\#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} = 2^{n-1}(1 + \mathcal{L}(\Phi))$ .

If  $\text{LAT}_\Phi[a, b] < 0$ , then

$$\begin{aligned} \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} &= \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \neq \langle b, \Phi(y) \rangle\} - 2^n \mathcal{L}(\Phi) \\ &= 2^n - \#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} - 2^n \mathcal{L}(\Phi), \end{aligned}$$

and in this case we have  $\#\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\} = 2^{n-1}(1 - \mathcal{L}(\Phi))$ .  $\square$

We also found interesting that construction of the set  $Y_\Phi$  is always the same regardless the sign of  $\text{LAT}_\Phi[a, b]$ ,  $(a, b) \in L(\Phi, \mathcal{L}(\Phi))$ , i.e.  $Y_\Phi =$

$\{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\}$ . If  $\text{LAT}_\Phi[a, b] < 0$  then  $Y_\Phi$  is the complement of the set  $Y'_\Phi = \{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \neq \langle b, \Phi(y) \rangle\}$  which define the value of  $\text{LAT}_\Phi[a, b]$  as result of Proposition 4. The following proposition ensures that in these cases the linearity (resp. nonlinearity) of  $\Phi$  will not improve after multiplication by transpositions created from  $Y_\Phi$ .

**Proposition 5.** *Let  $(i, j)$  be a transposition of  $S(\mathbb{F}_2^n)$  and  $\Phi, \Phi' \in S(\mathbb{F}_2^n)$  such that  $\Phi' = (i, j) \cdot \Phi$ , where  $n \geq 4$ . If  $L(\Phi, \mathcal{L}(\Phi)) = \{(a, b)\}$  and  $\text{LAT}_\Phi[a, b] < 0$ , then one of the following conditions hold*

1.  $\mathcal{L}(\Phi') = \mathcal{L}(\Phi)$  and  $\#L(\Phi', \mathcal{L}(\Phi')) \geq \#L(\Phi, \mathcal{L}(\Phi))$ ;
2.  $\mathcal{L}(\Phi') > \mathcal{L}(\Phi)$ .

*Proof.* Let  $a, b \in \setminus\{\mathbf{0}\}$ , and  $Y_{\Phi'} = \{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi'(y) \rangle\}$ . Since  $\Phi' = (i, j) \cdot \Phi$ , then the following relation holds

$$\begin{aligned} \#Y_{\Phi'} &= \sum_{y \in \mathbb{F}_2^n} \text{Ind}(\langle a, y \rangle, \langle b, \Phi'(y) \rangle) = \\ &= \sum_{y \in \mathbb{F}_2^n \setminus \{i, j\}} \text{Ind}(\langle a, y \rangle, \langle b, \Phi'(y) \rangle) + \text{Ind}(\langle a, i \rangle, \langle b, \Phi(j) \rangle) + \\ &\quad + \text{Ind}(\langle a, j \rangle, \langle b, \Phi(i) \rangle) \\ &\leq \#Y_\Phi + 2, \end{aligned} \tag{25}$$

where  $\sum_{y \in \mathbb{F}_2^n \setminus \{i, j\}} \text{Ind}(\langle a, y \rangle, \langle b, \Phi'(y) \rangle) \leq \sum_{y \in \mathbb{F}_2^n} \text{Ind}(\langle a, y \rangle, \langle b, \Phi'(y) \rangle) = \#Y_{\Phi'}$ .

From (25) we derive

$$\#Y_{\Phi'} - \#Y_\Phi \leq 2. \tag{26}$$

Now, taking into account that  $\text{LAT}_\Phi[a, b] < 0$ , then from Proposition 4 we have

- $\#Y_\Phi = 2^{n-1}(1 - \mathcal{L}(\Phi))$ ;
- $\#Y_{\Phi'} \in \{2^{n-1}(1 - \mathcal{L}(\Phi')), 2^{n-1}(1 + \mathcal{L}(\Phi'))\}$ .

If  $\#Y_{\Phi'} = 2^{n-1}(1 + \mathcal{L}(\Phi'))$ , then from (26) we obtain

$$2^{n-1}(\mathcal{L}(\Phi') + \mathcal{L}(\Phi)) \leq 2, \tag{27}$$

which yields a contradiction because by using relations (6),(7) and lemma 2 we obtain that

$$2^{n-1}(\mathcal{L}(\Phi') + \mathcal{L}(\Phi)) = 2^{-1} \left( \max_{a, b \in \mathbb{F}_2^n, b \neq \mathbf{0}} |\mathcal{W}_{\Phi'}(a)| + \max_{a, b \in \mathbb{F}_2^n, b \neq \mathbf{0}} |\mathcal{W}_\Phi(a)| \right) \geq 2^{\frac{n}{2}}.$$

So we deduce that  $\#Y_{\Phi'} = 2^{n-1}(1 - \mathcal{L}(\Phi'))$  and from (26) we derive

$$2^{n-1} \left[ -(\mathcal{L}(\Phi') - \mathcal{L}(\Phi)) \right] \leq 2. \quad (28)$$

This inequality holds in the following two cases

1. When  $\mathcal{L}(\Phi') - \mathcal{L}(\Phi) = 0$ . In this case we obtain  $\mathcal{L}(\Phi') = \mathcal{L}(\Phi)$ , which means that  $(a, b) \in L(\Phi', \mathcal{L}(\Phi'))$ , and thus  $\#L(\Phi', \mathcal{L}(\Phi')) \geq 1 = \#L(\Phi, \mathcal{L}(\Phi))$ ;
2. When  $\mathcal{L}(\Phi') - \mathcal{L}(\Phi) > 0$ . In this case, we immediately obtain  $\mathcal{L}(\Phi') > \mathcal{L}(\Phi)$ .

Therefore, the proof is complete. □

For a given S-Box  $\Phi \in S(\mathbb{F}_2^n)$  that satisfies the premises of Proposition 5 the Spectral-linear method from [29] will not find any S-Box  $\Phi' = (i, j) \cdot \Phi$ ,  $(i, j) \in S(\mathbb{F}_2^n)$  which improves the linear spectrum of  $\Phi$  and therefore the algorithm stops. When an S-Box with this characteristics is supplied as input of the Spectral-linear method, then the algorithm is not able to improve the nonlinearity of such S-Box up to the best values reported in [29].

Next, let us analyse the best possible scenario for the Spectral-linear method using parameter  $w_2 = 1$ . Suppose the input S-Box  $\Phi$  as well all S-Boxes of the  $i$ -th iteration of the method does not satisfy Proposition 5. As consequence, it can be ensured that exist  $(a, b) \in L(\Phi, \mathcal{L}(\Phi))$  such  $\text{LAT}_{\Phi}[a, b] > 0$ . Hence, by Proposition 4 we have that the size of the set  $Y_{\Phi} = \{y \in \mathbb{F}_2^n \mid \langle a, y \rangle = \langle b, \Phi(y) \rangle\}$  is equal to  $2^{n-1}(1 + \mathcal{L}(\Phi))$ . Finally, we assume that on each iteration of the method the value of nonlinearity improves, which is not necessarily true. If we took the initial value of nonlinearity as 92 (see discussion in Section 5.1.1), then we can estimate the number of solution evaluations made by the algorithm as shown in Table 3.

Notice that this estimate is greater than the best average solution evaluations to obtain the same value of nonlinearity presented in [19]. Moreover, if the value of parameter  $w_2$  is greater, also the number of S-Boxes to be evaluated will be higher.

### 5.2.3 Modified Spectral-linear method

Our variant of the Spectral-linear method introduce some changes w.r.t the original implementation from [29] in addition to the aforementioned selection of parameter  $w_2 = 1$ . The creation of the set  $Y_{\Phi}$  for a S-Box  $\Phi$  and

Iteration	Initial $\mathcal{NL}$	Solution evaluations
1	92	$\binom{164}{2} = 13366$
2	94	$\binom{162}{2} = 13041$
3	96	$\binom{160}{2} = 12720$
4	98	$\binom{158}{2} = 12403$
5	100	$\binom{156}{2} = 12090$
6	102	$\binom{154}{2} = 11781$
Total		75401

Table 3: An estimation of the number of solution evaluations made by the Spectral-linear method to reach nonlinearity 104 int best possible scenario.

a pair  $(a, b) \in L(\Phi, \mathcal{L}(\Phi))$  is based on the sign of the  $\text{LAT}_{\Phi}[a, b]$ , instead of only using the values of  $y \in \mathbb{F}_2^n$  such that  $\langle a, y \rangle = \langle b, \Phi(y) \rangle$  as suggested in [29]. The new selection of the set  $Y_{\Phi}$  nullifies the result of Proposition 5 towards the optimization of the linear characteristic of the S-Box. The values of  $a$  and  $b$  to construct  $Y_{\Phi}$  are selected at random from the pairs of elements in the set  $L(\Phi, \mathcal{L}(\Phi))$  as requested by the optimization process. This turns the deterministic condition of the original Spectral-linear method to a probabilistic approach, *i.e.* we can obtain different output S-Boxes no matter if the initial substitution is the same. The general procedure of our proposal is described in Algorithm 1.

#### 5.2.4 Modified Spectral-differential method

Let  $X_{\Phi} = \{x \in \mathbb{F}_2^n \mid \Phi(x \oplus a) \oplus \Phi(x) = b, \exists (a, b) \in D(\Phi, \delta(\Phi))\}$ . Obviously, for any  $(a, b) \in D(\Phi, \delta_{\Phi})$  we have  $\delta_{\Phi} = \frac{\#X_{\Phi}}{2^n}$  and it is straightforward that the number of transpositions  $(i, j)$  generated from  $X_{\Phi}$  is equal to  $\binom{\#X_{\Phi}}{2}$ . Nonetheless, all these transpositions does not guarantee the improvement of the  $\delta$ -uniformity of the S-Box  $\Phi' = \Phi \cdot (i, j)$  as expressed in the next proposition.

**Proposition 6.** *Let  $\Phi, \Phi'$  be two S-Boxes such that  $\Phi' = (i, i \oplus a) \cdot \Phi$  where  $i, i \oplus a \in X_{\Phi}$ . Then the following equality holds  $\delta_{\Phi'} = \delta_{\Phi}$ .*

*Proof.* Let  $X_{\Phi'} = \{y \in \mathbb{F}_2^n \mid \Phi'(y \oplus a) \oplus \Phi'(y) = b, \exists (a, b) \in D(\Phi', \delta(\Phi'))\}$ . Since  $\Phi' = (i, i \oplus a) \cdot \Phi$  we have  $\Phi'(i) = \Phi(i \oplus a)$ ,  $\Phi'(i \oplus a) = \Phi(i)$  and  $\Phi'(x) = \Phi(x) \forall i, i \oplus a \neq x \in \mathbb{F}_2^n$ . Using these facts and taking into account

---

**Algorithm 1:** Modified Spectral-linear method (**MSLM**)

---

**Input:** Permutation  $\Phi \in S(\mathbb{F}_2^n)$  and the desired linearity value  $\text{Goal}_{\mathcal{L}}$ .

- 1 For permutation  $\Phi$  of the  $i$ -th iteration of the algorithm,  $i \geq 0$ , calculate the values of  $\mathcal{L}(\Phi)$ ,  $\delta(\Phi)$ , the list  $\ell_i = L(\Phi, \mathcal{L}(\Phi))$  and set  $Y_{\Phi} = \{y \in \mathbb{F}_2^n \mid \langle a, y \rangle \otimes \langle b, \Phi(y) \rangle\}$  where  $(a, b) \in \ell_i$  and the symbol  $\otimes \in \{=, \neq\}$  in accordance to the sign of  $\text{LAT}_{\Phi}[a, b]$ .
  - 2 Construct the list of transpositions  $Y_{\Phi}^{\mathcal{L}} = \{(y, y') \in Y_{\Phi} \times Y_{\Phi} \mid y < y'\}$ .
  - 3 Using transposition  $(y, y') \in Y_{\Phi}^{\mathcal{L}}$  construct a new S-Box  $\Phi' = (y, y') \cdot \Phi$  and go to **step 4**. If all transpositions in  $Y_{\Phi}^{\mathcal{L}}$  were used, then go to **step 6**.
  - 4 If  $\mathcal{L}(\Phi') \leq \text{Goal}_{\mathcal{L}}$  and  $\delta(\Phi') \leq \delta(\Phi)$ , then update  $\Phi \leftarrow \Phi'$  and go to **step 7**, otherwise go to **step 5**.
  - 5 If  $\Phi'$  satisfies the fitness conditions from (18) then:
    - (I) if  $\mathcal{L}(\Phi') < \mathcal{L}(\Phi)$ , then update  $\Phi \leftarrow \Phi'$  and go to **step 1**.
    - (II) if  $\mathcal{L}(\Phi') = \mathcal{L}(\Phi)$ , then update  $\Phi \leftarrow \Phi'$  and go to **step 3** starting on the next transposition of  $Y_{\Phi}^{\mathcal{L}}$ .
  - 6 Check if one of the following conditions holds:
    - (I) Update was made in **step 5 (II)**, then go to **step 1**.
    - (II) No update was made in **step 5** and  $\exists (a', b') \in \ell_i$  which is not used yet, then recalculate the set  $Y_{\Phi}$  and go to **step 2**.
    - (III) All the pairs in the list  $\ell_i$  were used and no improvement of  $\mathcal{L}(\Phi)$  was found, then go to **step 7**.
  - 7 Return permutation  $\Phi$ .
- 

that  $\Phi'(z \oplus a) \oplus \Phi'(z) = \Phi(z) \oplus \Phi(z \oplus a) = b$  for  $z = i, i \oplus a \in \mathbb{F}_2^n$  we obtain

$$\begin{aligned}
 \delta_{\Phi'} &= \frac{\#X_{\Phi'}}{2^n} = \frac{\sum_{i, i \oplus a \neq x \in \mathbb{F}_2^n} \text{Ind}(\Phi(x \oplus a) \oplus \Phi(x), b) + 2 \cdot \text{Ind}(\Phi'(i \oplus a) \oplus \Phi'(i), b)}{2^n} \\
 &= \frac{\sum_{i, i \oplus a \neq x \in \mathbb{F}_2^n} \text{Ind}(\Phi(x \oplus a) \oplus \Phi(x), b) + 2 \cdot \text{Ind}(\Phi(i \oplus a) \oplus \Phi(i), b)}{2^n} \\
 &= \frac{\sum_{x \in \mathbb{F}_2^n} \text{Ind}(\Phi(x \oplus a) \oplus \Phi(x), b)}{2^n} \\
 &= \frac{\#X_{\Phi}}{2^n} = \delta_{\Phi}.
 \end{aligned}$$

□

A direct consequence of Proposition (6) is that there is no need to form the transpositions  $(i, i \oplus a), i, i \oplus a \in X_{\Phi}$  because the  $\delta$ -uniformity of the



---

**Algorithm 2:** Modified Spectral-differential method (**MSDM**)

---

**Input:** Permutation  $\Phi \in S(\mathbb{F}_2^n)$  and the desired  $\delta$ -uniformity value  $\text{Goal}_\delta$ .

- 1 For permutation  $\Phi$  of the  $i$ -th iteration of the algorithm,  $i \geq 0$ , calculate the values of  $\mathcal{L}(\Phi)$ ,  $\delta(\Phi)$ , the list  $\ell_i = D(\Phi, \delta_\Phi)$  and the set of transpositions  $X_\Phi^\delta$  for some  $(a, b) \in \ell_i$ .
  - 2 Using transposition  $(y, y') \in X_\Phi^\delta$  construct a new S-Box  $\Phi' = (y, y') \cdot \Phi$  and go to **step 3**. If all transpositions in  $X_\Phi^\delta$  were used, then go to **step 5**.
  - 3 If  $\delta(\Phi') \leq \text{Goal}_\delta$  and  $\mathcal{L}(\Phi') \leq \mathcal{L}(\Phi)$ , then update  $\Phi \leftarrow \Phi'$  and go to **step 6**, otherwise go to **step 4**.
  - 4 If  $\Phi'$  satisfies the fitness conditions from (19) then:
    - (I) if  $\delta(\Phi') < \delta(\Phi)$ , then update  $\Phi \leftarrow \Phi'$  and go to **step 1**.
    - (II) if  $\delta(\Phi) = \delta(\Phi')$ , then update  $\Phi \leftarrow \Phi'$  and go to **step 2** starting on the next transposition of the list.
  - 5 Check if one of the following conditions holds:
    - (I) Update was made in **step 4 (II)**, then go to **step 1**.
    - (II) No update was made in **step 4** and  $\exists (a', b') \in \ell_i$  which is not used yet, then recalculate the set  $X_\Phi^\delta$  and go to **step 2**.
    - (III) All the pairs in the list  $\ell_i$  were used and no improvement of  $\delta(\Phi)$  was found, then go to **step 6**.
  - 6 Return permutation  $\Phi$ .
- 

S-Box  $\Phi' = (i, i \oplus a) \cdot \Phi$  will not be improved. So in order to improve this parameter we need multiply  $\Phi$  by any transposition belonging to the following set  $\{(i, j) \mid 0 \leq i < j \leq \#X_\Phi - 1, j \neq i \oplus a\}$ . This reduce the number of all transpositions from  $X_\Phi$ , (i.e.,  $\binom{\#X_\Phi}{2}$ ), to check only  $\binom{\#X_\Phi}{2} - \frac{\#X_\Phi}{2}$  transpositions. Furthermore, the lower the value of  $\delta$ -uniformity, the lower the number of different transpositions that can be generated from  $X_\Phi$ . This can be a disadvantage in some scenarios due the reduced exploration of the search space. To mitigate this side effect of decreasing the value of  $\delta$ -uniformity we introduce a minor modification w.r.t the original Spectral-differential method from [29].

We define the set  $X_\Phi^c = \{y \in \mathbb{F}_2^n \mid y \notin X_\Phi\}$  (also mentioned in *Remark 4* of [29, p. 107]) as the complement of  $X_\Phi$  in  $\mathbb{F}_2^n$ . Notice that  $X_\Phi^c$  contains all the values of  $\mathbb{F}_2^n$  which are not solution of the equation  $\Phi(y \oplus a) \oplus \Phi(y) = b$  for the pair  $(a, b) \in D(\Phi, \delta(\Phi))$ . However,  $X_\Phi^c$  may contain solutions of the equation  $\Phi(y) \oplus \Phi(y \oplus \alpha) = \beta$  for some  $(\alpha, \beta) \in D(\Phi, p_1)$  having  $p_1 \leq \delta(\Phi)$ , then multiplying  $\Phi$  by transposition  $(x, y)$  such that  $x \in X_\Phi$  and  $y \in X_\Phi^c$  there exist the possibility of reducing both  $\#D(\Phi, \delta(\Phi))$  and  $\#D(\Phi, p_1)$



which result in the later improvement of the  $\delta$ -uniformity of  $\Phi$ . Having both,  $X_\Phi$  and  $X_\Phi^c$ , we construct the set of transpositions

$$X_\Phi^\delta = \{(x, y) \in X_\Phi \times X_\Phi^c \mid x \in X_\Phi, y \in X_\Phi^c\} \cup \{(x, y) \in X_\Phi \times X_\Phi \mid x < y\}$$

Using the set of transpositions  $X_\Phi^\delta$  we add more exploration to the classical version of the Spectral-differential method. The pseudo-code of the modified Spectral-differential method is shown in Algorithm 2.

### 5.2.5 Comments on the modified Spectral-linear method and Spectral-differential method methods

Before introduce the results achieved in the experimental phase, we like to summarize some aspects that we consider need to be explained in relation to our proposal. Firstly, we supply as input a desired value of linearity (resp.  $\delta$ -uniformity) which is used in **step 3** of both algorithms to avoid the unnecessary execution of the method if the desired cryptographic criteria have been found already. Furthermore, by alternating between **steps 3** and **5 (II)** of the **MSLM** and **steps 2** and **4 (II)** of the **MSDM** we chain several multiplications of the round S-Box  $\Phi$  by different transpositions generated from  $Y_\Phi$  (resp.  $X_\Phi^\delta$ ) instead of multiply by one transposition per iteration as presented in [29]. Thus, the list of transpositions generated by our methods to obtain an S-Box with improved linear (resp. differential) characteristics will be larger than those presented in [29]. Finally, as we stated for the modified Spectral-linear method, the selection of the pairs  $(a, b) \in L(\Phi, \mathcal{L}(\Phi))$  (resp.  $D(\Phi, \delta(\Phi))$ ) is random, which makes our methods probabilistic on the contrary of the proposal from [29] which claim to be deterministic.

## 6 Experimental results and discussion

The algorithms described in the previous section can be applied separately until we obtain a desired value of  $\mathcal{L}(\Phi)$  (resp.  $\delta(\Phi)$ ). Although, we decide to join the optimization phase carried by these algorithms (as suggested by the author of [29]) as an iterative process where the **MSLM** is applied first to the input S-Box and its output is passed to the **MSDM**. Then if the linear and differential criteria are not satisfied, the process is repeated on the output S-Box from the **MSDM**. Moreover, for a round S-Box  $\Phi$  we use the value of  $\mathcal{L}_g$  (resp.  $\delta_g$ ) equal to  $\mathcal{L}(\Phi) - \frac{2}{2^n-1}$  (resp.  $\delta(\Phi) - \frac{2}{2^n}$ ).

The experimental phase was designed to run 200 experiments separated in two blocks of 100 executions according to the required values of nonlinearity and differential uniformity. On each block of experiments we record the average number of S-Boxes evaluated (see Table 4) by the proposed optimization scheme with the objective of establish a comparison between our proposal and the state-of-the-art papers on the optimization of S-Boxes.

Cryptographic criteria	Number of experiments	Average Evaluated S-Boxes
$\mathcal{NL}_\Phi = 102$ $\delta_\Phi = \frac{6}{256}$	100	52191
$\mathcal{NL}_\Phi = 104$ $\delta_\Phi = \frac{8}{256}$	100	65134

Table 4: Average number of S-Boxes evaluated by the proposed optimization method.

As one may notice our proposal is able to produce S-Boxes having nonlinearity up to 104 and differential uniformity down to  $\frac{6}{256}$ . If we compare our results in terms of nonlinearity with the results from [8, 39, 34, 19] there is no great difference because in the majority of these papers the authors also obtain S-Boxes having nonlinearity 104. Our proposal outperforms them because it also guarantees a good value of differential uniformity. Particularly, in the case of S-Boxes having 104 and  $\frac{8}{256}$  of nonlinearity and differential uniformity respectively, the average number of evaluated S-Boxes reported in table 4 is better than the best number reported in [19] where the authors does not take into account the differential uniformity parameter in the optimization process. Furthermore, in the case of the work of Picek *et al.* [34], the authors introduce the differential uniformity in a multi-objective optimization process carried out by NSGA-II algorithm and the results obtained w.r.t the average number of S-Boxes evaluated surpasses the half million of S-Boxes which is by magnitude greater than our results. Finally, none of the aforementioned papers achieve a substitution box with differential uniformity value of  $\frac{6}{256}$ .

Following the discussion, there exist some papers in the referred literature which achieve the combination of nonlinearity and differential uniformity equal to 104 and  $\frac{6}{256}$  respectively starting from random S-Boxes [29, 24].

Moreover, our proposal is based in one of these papers [29]. Hence, it is straightforward that our goal is to achieve the same values of nonlinearity and differential uniformity reported in these papers. Although in practice it consumes a high number S-Box evaluations, we were able to obtain S-Boxes having such values of nonlinearity and differential uniformity. In Table 5 we compile an S-Box (denoted by  $\pi$ ) which has the aforementioned cryp-

$\pi$															
$\mathcal{NL}(\pi) = 104, \delta(\pi) = \frac{6}{256}, d_{min}(\pi) = 7, r_\pi = 3, r_\pi^{(3)} = 441.$															
0B	70	7A	B4	DD	36	4D	EB	F9	47	55	D2	99	A5	92	C7
6A	50	EF	31	2E	BA	53	10	11	C5	5D	67	61	0E	A6	94
3A	88	21	AB	1F	5E	E2	F4	62	76	1E	C8	3B	4A	7E	7C
32	C4	3C	57	F6	86	15	60	1A	38	2F	ED	87	06	D1	CE
79	C1	8E	20	90	01	A3	80	D5	7F	03	66	E5	C6	D4	96
EE	B5	B6	63	EC	EA	37	6E	B1	84	6B	3F	E0	D7	8C	C0
8B	9B	52	3D	6F	3E	BD	0D	B0	58	74	0C	22	F3	E3	DC
89	D6	24	BE	BC	C3	AF	40	98	07	42	59	CD	00	2C	82
41	E1	FC	78	DE	D9	85	DF	44	0A	E7	30	A8	0F	2B	6C
56	18	9E	48	7D	8A	CA	46	14	1B	33	D0	FA	54	B3	69
CF	02	BF	93	05	5A	A2	68	64	45	E8	FF	9C	39	08	16
73	CB	34	DB	23	4F	1D	75	5F	35	95	F8	BB	A1	26	91
12	1C	F2	B7	71	72	A9	83	04	F1	B2	E6	29	25	77	A7
17	E9	D8	FD	09	51	F5	E4	4C	AD	81	2D	97	4B	9D	AA
C2	2A	F7	6D	AC	8D	65	DA	AE	C9	B9	9A	27	F0	5B	13
9F	FE	19	D3	5C	49	FB	8F	CC	A0	28	4E	B8	7B	43	A4

Table 5: The LUT( $\pi$ ) in hexadecimal and its properties.

$\pi'_{Kuz}$															
$\mathcal{NL}(\pi'_{Kuz}) = 104, \delta(\pi'_{Kuz}) = \frac{6}{256}, d_{min}(\pi'_{Kuz}) = 7, r_{\pi'_{Kuz}} = 3, r_{\pi'_{Kuz}}^{(3)} = 441.$															
2D	BE	DD	C4	CF	6E	EE	50	FC	11	12	DA	3F	C5	14	4D
E9	77	5C	61	93	09	3A	2E	79	36	BA	BB	4F	8A	5F	31
F9	1F	A1	DE	E2	8B	EF	21	CC	1C	30	42	13	01	FF	7A
22	D3	64	89	9D	6A	8F	A0	06	0B	ED	98	7F	D4	F1	05
63	34	B3	51	9F	02	DB	AB	52	2A	B1	A2	C6	48	96	68
C1	78	EC	43	08	0C	76	EB	67	72	3E	47	5E	B7	5D	87
15	B5	CE	9B	10	7B	9A	C7	F3	45	0E	EA	20	9E	C8	65
B0	AA	19	95	0A	35	CD	7E	73	54	F0	80	83	BD	B2	57
DF	F5	A5	A9	82	A8	FD	D6	D7	17	2C	44	7C	16	B9	03
E0	0F	29	5A	69	94	32	81	DC	E8	28	18	4E	33	8E	4A
A7	D1	60	56	37	00	62	F2	1A	B8	41	A3	84	BC	26	38
AD	F6	46	92	27	9C	FB	0D	8C	90	24	7D	75	D5	91	3B
07	2F	49	40	86	AC	1D	F7	6F	71	6B	58	88	C3	55	AE
E1	1B	D9	97	E6	6D	F8	FE	8D	E4	3D	53	CA	D8	85	99
C2	1E	BF	A4	CB	2B	F4	5B	C9	23	25	D0	70	FA	6C	C0
74	A6	59	D2	4C	E7	B4	3C	04	66	AF	E3	39	4B	E5	B6
Sequence of transpositions to convert $\pi_{Kuz}$ into $\pi'_{Kuz}$															
(0, 8), (0, 49), (0, 50), (0, 69), (0, 110), (0, 126), (0, 183), (0, 193), (0, 200), (0, 211),															
(0, 232), (1, 6), (1, 31), (1, 80), (1, 97), (1, 111), (1, 134), (1, 163), (23, 26), (23, 62),															
(33, 63), (48, 63), (105, 190), (0, 228), (72, 239), (167, 239), (239, 247), (171, 185),															
(185, 217), (185, 219), (37, 134), (161, 211), (161, 218), (78, 98), (82, 132), (132, 217),															
(42, 218), (218, 247), (52, 108), (50, 172), (51, 207), (99, 233), (35, 147), (3, 9), (7, 141),															
(7, 155), (44, 90), (170, 175), (108, 251), (1, 120), (46, 116), (116, 158), (130, 186),															
(194, 232), (212, 244), (19, 22), (22, 70), (40, 90), (8, 74), (18, 103), (88, 226), (146, 173),															
(182, 206), (8, 74), (23, 182), (23, 245), (108, 224), (148, 188), (161, 248), (1, 213),															
(12, 203), (14, 28), (205, 210), (92, 181), (42, 151), (103, 122), (42, 151), (82, 151),															
(105, 177), (37, 74), (111, 116), (19, 223), (47, 188), (1, 236), (48, 155), (64, 87),															
(139, 177), (188, 248), (64, 254), (90, 151), (18, 134), (37, 40), (115, 139), (34, 111),															
(34, 116), (46, 246), (237, 254), (46, 246), (113, 188), (83, 163), (124, 205), (107, 200),															
(4, 180), (103, 134), (4, 198), (21, 230), (4, 198), (23, 230), (4, 180), (68, 146), (28, 248),															
(139, 218), (146, 233), (81, 106), (106, 236), (201, 225), (10, 237), (49, 182), (106, 217),															
(107, 233), (203, 233), (76, 134), (164, 225), (24, 137), (240, 242), (29, 118), (12, 217),															
(135, 232), (40, 79), (112, 150), (82, 173), (66, 135), (135, 138), (22, 77)															

Table 6: The evolved S-Box  $\pi'_{Kuz} \in \mathcal{S}(\mathbb{F}_2^8)$  in hexadecimal, its properties and the sequence of 134 transpositions to convert  $\pi_{Kuz}$  into  $\pi'_{Kuz}$ . Transpositions in green were made by the **MSLM** while transpositions in blue were made by the **MSDM**.

tographic parameters obtained after evaluating approximately one million S-Boxes.

In addition, we apply our optimization scheme to  $\pi_{Kuz}$ , and obtain in

around three million evaluations an S-Box having the same cryptographic parameters reported in [29, 24]. The Look-Up Table of the optimized S-Box (denoted by  $\pi'_{\text{Kuz}}$ ), its cryptographic properties and the sequence of transpositions to obtain this permutation are presented in Table 6. The Look-Up Tables of  $\pi_{\text{Kuz}}$  and  $\pi'_{\text{Kuz}}$  differ in 139 values.

Reference	$\mathcal{NL}(\Phi)$	$\delta(\Phi)$	$d_{\min}(\Phi)$	$r_{\Phi}$
Ivanov <i>et al.</i> [25]	112	$\frac{6}{256}$	7	2
Bolufé & Tamayo [5]	112	<b>NR</b>	<b>NR</b>	<b>NR</b>
<b>This work</b>	<b>104</b>	$\frac{6}{256}$	<b>7</b>	<b>3</b>
Menyachikhin [29]	104	$\frac{6}{256}$	7	3
Ivanov <i>et al.</i> [24]	104	$\frac{6}{256}$	7	3
Kazymyrov <i>et al.</i> [26]	104	$\frac{8}{256}$	7	3
Picek <i>et al.</i> [34]	104	$\frac{8}{256}$	<b>NR</b>	<b>NR</b>
Freyre <i>et al.</i> [19]	104	$\frac{8}{256}$	6	3
Tesař [39]	104	<b>NR</b>	<b>NR</b>	<b>NR</b>
Freyre <i>et al.</i> [20]	102	$\frac{8}{256}$	7	3
Clark <i>et al.</i> [8]	102	<b>NR</b>	<b>NR</b>	<b>NR</b>
Freyre [17]	100	$\frac{8}{256}$	6	3
Millan [30]	100	<b>NR</b>	<b>NR</b>	<b>NR</b>
Zahid <i>et al.</i> [41]	96	$\frac{10}{256}$	6	3
Kim <i>et al.</i> [27]	96	$\frac{16}{256}$	2	2

Table 7: Comparison of nonlinearity and differential uniformity of some state-of-the-art S-Boxes obtained through optimization. NR states for not reported in the corresponding reference and no resulting S-Box was presented either.

Finally, we present in Table 7 a comparison of our results with some papers on optimization of S-Boxes w.r.t nonlinearity and  $\delta$ -uniformity. It worth to remark that the results shown in the table were taken regardless the seeding methods used to start the optimization process and, as far as we know, the top three results in the table receive as input S-Boxes produced by constructions that give them some advantages towards the properties of nonlinearity and differential uniformity rather than the random generation.

## 7 Conclusions

In this work we have presented a detailed analysis of a new cost function to evolve S-Boxes with good nonlinearity values. In addition, we study the algebraic characteristic of the linear spectrum of an S-Box and design a new optimization method which take advantage of some interesting results derived from the propositions of Section 5.2.2. We also extend the search space of the modified spectral differential method [29] through the construction of a larger set of transpositions to be used by Algorithm 2.

Finally, the experimental results obtained in this paper are comparable in quality to the best reported in public literature for evolution of random generated S-Boxes. Moreover, we outperform, in terms of the average number of S-Boxes evaluated to obtain the desired cryptographic parameters, most of the research papers taken as reference point in our investigation.

## References

- [1] Abd el-Latif A. A., Abd-el-Atty B., Amin M., Ilyyasu A. M., “Quantum-inspired cascaded discrete-time quantum walks with induced chaotic dynamics and cryptographic applications”, *Scientific reports*, **10** **1** (2020), 1–16.
- [2] Abd el-Latif A. A., Abd-el-Atty B., Venegas-Andraca S. E., “novel image steganography technique based on quantum Substitution Boxes”, *Optics & Laser Technology*, **116** (2019), 92–102.
- [3] Azam N. A., Hayat U., Ullah I., “An injective s-Box design scheme over an ordered isomorphic elliptic curve and its characterization”, *Security and communication networks*, 2018 .
- [4] Azam N. A., Hayat U., Ullah I., “Efficient construction of a Substitution Box based on a Model elliptic curve over a finite field”, *Frontiers of Information Technology & Electronic Engineering*, **20** **10** (2019), 1378–1389.
- [5] Bolufé-Röhler A., Tamayo-Vera D., “Machine learning based metaheuristic hybrids for S-box optimization”, *Journal of Ambient Intelligence and Humanized Computing*, 2020, 1–14.
- [6] Boura, C., Canteaut, A., Jean, J. et al., “Two notions of differential equivalence on Sboxes”, *Designs, Codes and Cryptography*, **87**, 2019 DOI: [10.1007/s10623-018-0496-z](https://doi.org/10.1007/s10623-018-0496-z), 185–202.
- [7] Carlet C., *Boolean Functions for Cryptography and Coding Theory*, Cambridge, Cambridge University Press, 2021 DOI: [10.1017/9781108606806](https://doi.org/10.1017/9781108606806).
- [8] Clark J. A., Jacob J., Stepney S., “The design of S-boxes by simulated annealing”, *New Generation Computing*, **23** **3** (2005), 219–231.
- [9] Courtois, N. T., and Pieprzyk, J., “Cryptanalysis of Block Ciphers with Overdefined Systems of Equations”, Cryptology ePrint Archive, Report 2002/044, <https://eprint.iacr.org/2002/044>.
- [10] de la Cruz-Jiménez, R. A., “Generation of 8-bit s-boxes having almost optimal cryptographic properties using smaller 4-bit s-boxes and finite field multiplication”, *International Conference on Cryptology and Information Security in Latin America*, Springer, 2017, 191–206.
- [11] de la Cruz Jiménez, R. A., Kamlovskiy, O. V., “The sum of modules of Walsh coefficients of Boolean functions”, *Discretnaya Matematika*, **26**:5 (2016), 259–272 DOI: [10.1515/dma-2016-0023](https://doi.org/10.1515/dma-2016-0023).
- [12] Dimitrov M. M., “On the Design of Chaos-Based S-Boxes”, *IEEE Access*, **8** (2020), 117173–117181.
- [13] Dinur A., Shamir A., “Cube Attacks on Tweakable Black Box Polynomials”.
- [14] Djurasevic M., Jakobovic D., Picek S., “One property to rule them all? On the limits of trade-offs for S-boxes”, *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, 2020, 1064–1072.
- [15] Eiben A. E., Smith J. E., “Introduction to evolutionary computing”, 2003.
- [16] Fomin D. B., “New classes of 8-bit permutations based on a butterfly structure”, *Mat. Vopr. Kriptogr.*, **10**:2 (2019), 169–180.
- [17] Freyre-Echevarría A., *Evolución híbrida de s-cajas no lineales resistentes a ataques de potencia*, Universidad de la Habana, 2020, BSc. Thesis DOI: [10.13140/RG.2.2.17037.77284/1](https://doi.org/10.13140/RG.2.2.17037.77284/1).



- [18] Freyre-Echevarría A., “On the generation of cryptographically strong substitution boxes from small ones and heuristic search”, *10th Workshop on Current Trends in Cryptology (CTCRYPT 2021)*, 2021, 112.
- [19] Freyre-Echevarría A., Alanezi A., Martínez-Díaz I., Ahmad M., Abd El-Latif A. A., Kolivand H., Razaq A. 11, “An External Parameter Independent Novel Cost Function for Evolving Bijective Substitution-Boxes”, *Symmetry*, **12** DOI: [10.3390/sym12111896](https://doi.org/10.3390/sym12111896) (2020).
- [20] Freyre-Echevarría A., Martínez-Díaz I., Legón-Pérez C. M., Sosa-Gómez G. Rojas O., “Evolving Nonlinear S-Boxes With Improved Theoretical Resilience to Power Attacks”, *IEEE Access*, **8** (2020), 202728–202737 DOI: [10.1109/ACCESS.2020.3035163](https://doi.org/10.1109/ACCESS.2020.3035163).
- [21] Glukhov M.M., Elizarov V.P., Nechaev A.A., *Algebra: Uchebnik. - izdanie vtoroe, ispravlennoe i dopolnennoe. [Algebra: Textbook. - second edition, revised and supplemented]*, Izdatelstvo Lan, Sankt-Peterburg, Moskva, Krasnodar, 2015, In Russian.
- [22] GOST R 34.12-2015, “Information technology. Cryptographic protection of information. Block ciphers”, *Standartinform*, Moscow, 2015.
- [23] Hematpour N., Ahadpour S., Behnia S., “Presence of dynamics of quantum dots in the digital signature using DNA alphabet and chaotic S-box”, *Multimedia Tools and Applications*, 2020, 1–23.
- [24] Ivanov G., Nikolov N., Nikova S., “Cryptographically strong S-boxes generated by modified immune algorithm”, *International Conference on Cryptography and Information Security in the Balkans*, Springer, 2015, 31–42.
- [25] Ivanov G., Nikolov N., Nikova S., “Reversed genetic algorithms for generation of bijective s-boxes with good cryptographic properties”, *Cryptography and Communications*, **8** 2 (2016), 247–276.
- [26] Kazymyrov O. V., Kazymyrova V. N., Oliynykov R. V., “A method for generation of high-nonlinear S-Boxes based on gradient descent”, *Mat. Vopr. Kriptogr.*, **5**:2 (2014), 71–78 DOI: [10.4213/mvk118](https://doi.org/10.4213/mvk118).
- [27] Kim G., Kim H., Heo Y., Jeon Y., Kim J., “Generating Cryptographic S-Boxes Using the Reinforcement Learning”, *IEEE Access*, **9** DOI: [0.1109/ACCESS.2021.3085861](https://doi.org/10.1109/ACCESS.2021.3085861) (2021).
- [28] Lu Q., Zhu C., Deng X., “An Efficient Image Encryption Scheme Based on the LSS Chaotic Map and Single S-Box”, *IEEE Access*, **8** (2020), 25664–25678.
- [29] Menyachikhin A. V., “Spectral-linear and spectral-differential methods for generating S-Boxes having almost optimal cryptographic parameters”, *Mat. Vopr. Kriptogr.*, **8**:2 (2017), 97–116 DOI: [10.4213/mvk227](https://doi.org/10.4213/mvk227).
- [30] Millan, W., “How to improve the nonlinearity of bijective S-boxes”, *Australian Conference on Information Security and Privacy*, Springer, 1998, 181–192.
- [31] Millan W., Clark A. and Dawson E., “Boolean functions design using hill climbing methods”, *LNCS, ACISP 1999*, **1587**, Springer, Berlin, Heidelberg, 1999.
- [32] Mohamed A. G., Korany N. O., El-Khamy S. E., “New DNA Coded Fuzzy Based (DNAFZ) S-Boxes: Application to Robust Image Encryption Using Hyper Chaotic Maps”, *IEEE Access*, **9** (2021), 14284–14305.
- [33] Picek S., *Applications of evolutionary computation to cryptology*, 2015, PhD. Thesis.
- [34] Picek S., Cupic M., Rotim L., “A new cost function for evolution of s-boxes”, *Evolutionary computation*, **24** 4 (2016), 695–718 DOI: [10.1162/EVCO\\_a\\_00191](https://doi.org/10.1162/EVCO_a_00191).
- [35] Picek S., Ege B., Batina L., Jakobovic D., Chmielewski L., Golub M., “On using genetic algorithms for intrinsic side-channel resistance: the case of AES S-box”, *Proceedings of the First Workshop on Cryptography and Security in Computing Systems*, 2014, 13–18.
- [36] Picek S., Papagiannopoulos K., Ege B., Batina L., Jakobovic D., “Confused by confusion: Systematic evaluation of DPA resistance of various s-boxes”, *International Conference on Cryptology in India*, Springer, 2014, 374–390.
- [37] Pokrasenko D. P., “On the maximal component algebraic immunity of vectorial Boolean functions”, *J. Appl. Industr. Math.*, **10**:2 (2016), 257–263 DOI: [10.1134/S1990478916020101](https://doi.org/10.1134/S1990478916020101).
- [38] *Sage Mathematics Software (Version 8.1)*, 2018, <http://www.sagemath.org>.
- [39] Tesar P., “A new method for generating high non-linearity s-boxes”, *Radioengineering*, **19** 1 (2010), 23–26.

- [40] Yu Y., Wang M., Li Y., *Constructing differentially 4 uniform permutation from know ones*, Cryptology ePrint Archive, Report 2011/047, <https://eprint.iacr.org/2011/047>.
- [41] Zahid A. H., *et al.*, “A Novel Construction of Dynamic S-Box With High Non-linearity Using Heuristic Evolution”, *IEEE Access*, **9** (2021), 67797–67812 DOI: [10.1109/ACCESS.2021.3077194](https://doi.org/10.1109/ACCESS.2021.3077194).

## Appendix

The following SAGE [38] script prints some characteristics (used in Section 5.2.1) of intermediate S-Boxes involved in Step 2 of algorithms implementing the spectral-linear and spectral-differential methods [29] for optimizing the cryptographic properties of the S-Box  $\pi_{\text{KUZ}}$  of the national standard of Russian Federation GOST R 34.11-2015 [22].

```
#!/usr/bin/sage
from sage.all import *
from sage.crypto.sbox import SBox
import copy

# multiplication of permutation by a transposition
def mult_by_transp(s, pos1, pos2):
    tmp = list(copy.deepcopy(s))
    a, b = tmp.index(pos1), tmp.index(pos2)
    tmp[a], tmp[b] = tmp[b], tmp[a]

    return tmp

# this function return the set D(s, DU(s))
def D(s, p1):
    n = SBox(s).input_size()
    T = SBox(s).difference_distribution_table()
    table = []
    for x in range(1, 2**n):
        for y in range(1, 2**n):
            if abs(T[x][y]) == p1:
                table.append([x, y])

    return table

# this function return the set L(s, LIN(s))
def L(s, p1):
    n = SBox(s).input_size()
    f_coef = "fourier_coefficient"
    T = SBox(s).linear_approximation_table(scale = f_coef)
    table = []
    for x in range(1, 2**n):
        for y in range(1, 2**n):
            if abs(T[x][y]) == p1:
                table.append([x, y])

    return table
```

```

# the Look-Up Table of Kuznyechik S-Box
pi_Kuz = [
    0xfc, 0xee, 0xdd, 0x11, 0xcf, 0x6e, 0x31, 0x16, 0xfb, 0xc4,
    0xfa, 0xda, 0x23, 0xc5, 0x04, 0x4d, 0xe9, 0x77, 0xf0, 0xdb,
    0x93, 0x2e, 0x99, 0xba, 0x17, 0x36, 0xf1, 0xbb, 0x14, 0xcd,
    0x5f, 0xc1, 0xf9, 0x18, 0x65, 0x5a, 0xe2, 0x5c, 0xef, 0x21,
    0x81, 0x1c, 0x3c, 0x42, 0x8b, 0x01, 0x8e, 0x4f, 0x05, 0x84,
    0x02, 0xae, 0xe3, 0x6a, 0x8f, 0xa0, 0x06, 0x0b, 0xed, 0x98,
    0x7f, 0xd4, 0xd3, 0x1f, 0xeb, 0x34, 0x2c, 0x51, 0xea, 0xc8,
    0x48, 0xab, 0xf2, 0x2a, 0x68, 0xa2, 0xfd, 0x3a, 0xce, 0xcc,
    0xb5, 0x70, 0x0e, 0x56, 0x08, 0x0c, 0x76, 0x12, 0xbf, 0x72,
    0x13, 0x47, 0x9c, 0xb7, 0x5d, 0x87, 0x15, 0xa1, 0x96, 0x29,
    0x10, 0x7b, 0x9a, 0xc7, 0xf3, 0x91, 0x78, 0x6f, 0x9d, 0x9e,
    0xb2, 0xb1, 0x32, 0x75, 0x19, 0x3d, 0xff, 0x35, 0x8a, 0x7e,
    0x6d, 0x54, 0xc6, 0x80, 0xc3, 0xbd, 0x0d, 0x57, 0xdf, 0xf5,
    0x24, 0xa9, 0x3e, 0xa8, 0x43, 0xc9, 0xd7, 0x79, 0xd6, 0xf6,
    0x7c, 0x22, 0xb9, 0x03, 0xe0, 0x0f, 0xec, 0xde, 0x7a, 0x94,
    0xb0, 0xbc, 0xdc, 0xe8, 0x28, 0x50, 0x4e, 0x33, 0x0a, 0x4a,
    0xa7, 0x97, 0x60, 0x73, 0x1e, 0x00, 0x62, 0x44, 0x1a, 0xb8,
    0x38, 0x82, 0x64, 0x9f, 0x26, 0x41, 0xad, 0x45, 0x46, 0x92,
    0x27, 0x5e, 0x55, 0x2f, 0x8c, 0xa3, 0xa5, 0x7d, 0x69, 0xd5,
    0x95, 0x3b, 0x07, 0x58, 0xb3, 0x40, 0x86, 0xac, 0x1d, 0xf7,
    0x30, 0x37, 0x6b, 0xe4, 0x88, 0xd9, 0xe7, 0x89, 0xe1, 0x1b,
    0x83, 0x49, 0x4c, 0x3f, 0xf8, 0xfe, 0x8d, 0x53, 0xaa, 0x90,
    0xca, 0xd8, 0x85, 0x61, 0x20, 0x71, 0x67, 0xa4, 0x2d, 0x2b,
    0x09, 0x5b, 0xcb, 0x9b, 0x25, 0xd0, 0xbe, 0xe5, 0x6c, 0x52,
    0x59, 0xa6, 0x74, 0xd2, 0xe6, 0xf4, 0xb4, 0xc0, 0xd1, 0x66,
    0xaf, 0xc2, 0x39, 0x4b, 0x63, 0xb6
]

# List of transpositions given by A.V. Menyachikhin in Table 6 p.114 of the
# article: Spectral-linear and spectral-differential methods for generating
# S-boxes having almost optimal cryptographic parameters

Transp = [
    (0x6e, 0xb7), (0xfb, 0x53), (0xda, 0x2b), (0xc5, 0xb5),
    (0xdb, 0x69), (0x93, 0xac), (0x99, 0xbd), (0xf1, 0x68),
    (0x81, 0xaa), (0x01, 0x7c), (0x05, 0x24), (0xa0, 0x79),
    (0x06, 0x98), (0xd3, 0x7a), (0xeb, 0xd0), (0xc8, 0x58),
    (0x2a, 0x30), (0xb5, 0x91), (0x70, 0x67), (0x9c, 0xb3),
    (0x29, 0xb4), (0xb2, 0xd9), (0x19, 0x0a), (0x35, 0x0a),
    (0x6d, 0x55), (0x54, 0xe6), (0xbd, 0x00), (0xf5, 0x9f),
    (0x24, 0x63), (0x43, 0xf7), (0x79, 0xa3), (0x94, 0x9f),
    (0x00, 0x64), (0x30, 0xc0), (0x2d, 0x6c), (0xd0, 0xc2)
]

# reversing the list of transpositions and calculating the size of Transp
Transp      = Transp[::-1]
size_of_Transp = len(Transp)

if __name__ == "__main__":
    # computation of some characteristics for Kuznyechik S-Box
    g          = pi_Kuz
    car_dif_g  = SBox(g).differential_uniformity()
    car_lin_g  = SBox(g).linearity()
    card_D_g   = len(D(g, car_dif_g))

```



```

card_L_g      = len(L(g, car_lin_g))

# printing the characteristics of Kuznyechik S-Box
print("#"*102)
print("\t\t\tCharacteristics of the S-Box")
print(
    "DU(pi_Kuz) = {0}".format(car_dif_g),
    "\tLIN(pi_Kuz) = {0}".format(car_lin_g),
    "\t#D(pi_Kuz, DU(pi_Kuz')) = {0}".format(card_D_g),
    "\t#L(pi_Kuz, LIN(pi_Kuz')) = {0}".format(card_L_g)
)
print("#"*102)
print('')

# printing the characteristics of intermediate S-Boxes involved in
# Step 2 of algorithms implementing the Spectral-linear and
# Spectral-differential methods
for i in range(size_of_Transp):
    g          = mult_by_transp(pi_Kuz, Transp[i][0], Transp[i][1])
    car_dif_g  = SBox(g).differential_uniformity()
    car_lin_g  = SBox(g).linearity()
    card_D_g   = len(D(g, car_dif_g))
    card_L_g   = len(L(g, car_lin_g))
    pi_Kuz     = g

    print("="*100)
    print("Characteristics of the previous S-Box multiplied by " +
          "transposition -> "+str(Transp[i]))
    print(
        "DU(pi_Kuz-" + str(i+1) + ") = {0}".format(car_dif_g),
        "\tLIN(pi_Kuz-" + str(i+1) + ") = {0}".format(car_lin_g),
        "\t#D(pi_Kuz-" + str(i+1) + ") = {0}".format(card_D_g),
        "\t#L(pi_Kuz-" + str(i+1) + ") = {0}".format(card_L_g)
    )

print("="*100)

```

SYMMETRIC CRYPTOGRAPHY  
ANALYSIS

# Related-key attacks on the compression function of Streebog

Vitaly Kiryukhin

LLC «SFB Lab», JSC «InfoTeCS», Moscow, Russia  
vitaly.kiryukhin@sfblaboratory.ru

## Abstract

Related-key attacks against block ciphers are often considered unrealistic. In practice, as far as possible, the existence of a known «relation» between the secret encryption keys is avoided. Despite this, related keys arise directly in some widely used keyed hash functions. This is especially true for HMAC-Streebog, where known constants and manipulated parameters are added to the secret key. The relation is determined by addition modulo 2 and  $2^n$ . The security of HMAC reduces to the properties of the underlying compression function. Therefore, as an initial analysis we propose key-recovery methods for 10 and 11 rounds (out of 12) of Streebog compression function in the related-key setting. The result shows that Streebog successfully resists attacks even in the model with such powerful adversaries.

**Keywords:** Streebog, related-key, truncated differentials, rebound

## 1 Introduction

A secure cryptographic keyless hash function  $H$  must meet many requirements, including the three most well-known: preimage resistance, second preimage resistance and collision resistance. Similar requirements are imposed on the compression function  $g(H, M)$  if the hash function is based on the Merkle-Damgård (MD) scheme [4, 3].

However, if the MD-like hash function is converted into the keyed one using HMAC [7] with the secret  $K$

$$\text{HMAC}(K, Msg) = H((K \oplus opad) || H((K \oplus ipad) || Msg)),$$

then other properties are expected from the compression function [8].

Firstly,  $g(H, \cdot)$  with the secret state  $H$  must be indistinguishable from a truly random function.

Secondly, the pair  $g(\cdot, K \oplus ipad)$  and  $g(\cdot, K \oplus opad)$  must be indistinguishable from a pair of random functions. In other words,  $g$  must be protected from attacks using two related keys [5].

Even more interesting is the situation when the Russian hash function Streebog [1] is used in HMAC. Streebog uses the Merkle-Damgård approach with some subtle differences, including the use of the checksum (modulo  $2^n$ ) of all message blocks in the finalization. In HMAC-Streebog there are four calls of the compression function where the secret key is used, one of them is

$$\mathbf{g}(\cdot, (K \oplus \text{ipad}) \boxplus \Sigma)$$

where  $\Sigma$  is the checksum that the attacker can freely manipulate by changing the message,  $\langle \boxplus \rangle$  denotes the addition modulo  $2^n$ .

Therefore, it would be reasonable for HMAC-Streebog to require  $\mathbf{g}(\cdot, (K \oplus \Phi) \boxplus \Sigma)$  with the random secret  $K$  to be indistinguishable from a family of random functions. In general, the input and parameters  $\Phi$ ,  $\Sigma$  are adaptively chosen by the adversary. One can consider such significant capabilities of the attacker mostly exaggerated, but they are convenient for security proofs.

Streebog and its underlying transformations have received a lot of attention from cryptographers. Basically, the articles on the topic were devoted to analysis in the keyless settings [9–17, 19, 20].

We can cite only three works [18, 22, 23] devoted to the analysis of Streebog when using secret keys.

In [22] the key-recovery attacks on HMAC-Streebog were presented as the extension of the generic state-recovery attacks on HMAC. The time and data complexities of attacks are significantly more than «provable secure» bounds of HMAC [8]. The method also does not use the properties of the compression function.

Impossible differential properties of the compression function are utilized in [18] to mount secret-state recovery attacks on 6.75-rounds  $\mathbf{g}(H, \cdot)$ . The article [23] presents 7-round key-recovery attacks against  $\mathbf{g}(H, \cdot)$  and  $\mathbf{g}(\cdot, M)$ , where  $H$  (resp.  $M$ ) is secret.

As far as we know, the Streebog compression function has not been previously considered in the related-key settings. We extend the approach presented in [23] to attack  $\mathbf{g}(\cdot, M)$  and propose the key-recovery method for  $\mathbf{g}(\cdot, (K \oplus \Phi) \boxplus \Sigma)$ . First, we construct the single-key method that works with a negligible success probability, but also with a relatively low time complexity. The rebound technique [24] and the truncated differentials [6] are the main parts of the method. Next, we present the effective way to convert this method into a highly probable one by using the sets of related keys. As a result, we have a key-recovery method against 10 and 11 rounds (of 12). Comparative characteristics are presented in table 1.

We are convinced that our results provide an additional argument showing that Streebog compression function has a sufficient security margin even in the related-key setting.

Setting	Rounds	Time	Memory	Data	Keys	Description
secret $H$	6.75	$2^{399.5}$	$2^{349}$	$2^{483}$	1	[18]
	6.75	$2^{261.5}$	$2^{205}$	$2^{495.5}$	1	[18]
	7	$2^{421}$	$2^{354}$	$2^{64}$	1	[23]
	12	$2^{256}$	$2^{256}$	$2^{256}$	1	birthday-paradox distinguisher
	12	$2^{512}$	$\sim$	2	1	key guessing
secret $M$	7	$2^{240}$	$2^{20}$	$2^{113}$	1	[23]
	10	$2^{224}$	$2^{94}$	$2^{225}$	$2^{198}$	Section 6 (any relation)
	10	$2^{232}$	$2^{91}$	$2^{168}$	$2^{145}$	Section 6 (only $\oplus$ )
	11	$2^{224}$	$2^{68}$	$2^{225}$	$2^{224}$	Section 6 (only $\oplus$ )
	12	$2^{367}$	$2^{145}$	$2^{145}$	$2^{145}$	parallel key guessing
	12	$2^{314}$	$2^{198}$	$2^{198}$	$2^{198}$	
	12	$2^{288}$	$2^{224}$	$2^{224}$	$2^{224}$	
	12	$2^{256}$	$2^{256}$	$2^{256}$	$2^{256}$	
12	$2^{512}$	$\sim$	2	1	key guessing	

Table 1: Attacks on the Streebog compression functions in secret-key settings. «Time» ( $t$ ) in  $\mathbf{g}$  computations, «Memory» in  $n$ -bit blocks, «Data» ( $q$ ) in chosen message-output pairs over all keys, «Keys» is the number of used related keys (single-key attack is denoted by «Keys = 1»).

## 2 Definitions

Let  $\mathbb{F}_{2^8}$  be a finite field. Each element of  $\mathbb{F}_{2^8}$  can be interpreted as an integer or a binary vector. Denote  $8 \times 8$  matrix space over  $\mathbb{F}_{2^8}$  by  $\mathbb{F}_{2^8}^{8 \times 8}$  (we also use symbol  $\mathbb{F}_{2^8}^8$  as a vector space). Elements from  $\mathbb{F}_{2^8}^{8 \times 8}$  will be denoted by capital letters:  $A, B$ . Blocks of states and messages also belong to  $\mathbb{F}_{2^8}^{8 \times 8}$ . Elements of a matrix are indexed by  $0 \leq i, j \leq v-1$  (for example,  $a = A[0, 0]$  is an element from the upper-left corner of the matrix).  $A[i, \cdot]$  is  $i$ -th row of  $A$ ,  $A[\cdot, j]$  is  $j$ -th column of  $A$ . Elements from  $\mathbb{F}_{2^8}/\mathbb{F}_{2^8}^8/\mathbb{F}_{2^8}^{8 \times 8}$  can be represented as 8-, 64-, 512-bit strings, respectively.

Denote addition modulo 2 and addition modulo  $2^n$  by symbols « $\oplus$ » and « $\boxplus$ » correspondingly,  $n = 512$ . These operations are defined naturally for all the objects under consideration.

We refer to  $\Delta \mathbf{B} = B \oplus B' \in \mathbb{F}_{2^8}^{8 \times 8}$  as a difference and indicate it in bold:  $\Delta \mathbf{M}, \Delta \mathbf{K}_4$ . If  $\Delta \mathbf{B}[i, j] \neq 0$  then we say that the position  $(i, j)$  is active, otherwise inactive. The differential trail is the sequence of the differences after each transformation in the cipher. The truncated differential trail is the set of the differential trails that have the same active positions.

The transformations over  $\mathbb{F}_{2^8}^{8 \times 8}$  (also over  $\mathbb{F}_{2^8}^8$  and  $\mathbb{F}_{2^8}$ ) are denoted in Sans Serif font:  $\mathbf{f}$ ,  $\mathbf{S}$ ,  $\mathbf{L}$ . The notation  $\mathbf{LS}$  indicates a composition of transformations, where  $\mathbf{S}$  applies first (the reverse order «left-to-right» is used on the figures). The inverse transformations are specified by  $\mathbf{f}^{-1}$ .

### 3 Streebog

The state size of Streebog consists of  $n = 512$  bits ( $8 \times 8$  bytes).

The message  $Msg$  is hashed as follows.

The text is always padded with bit string  $10 \dots 0$  and divided into  $l$  blocks of  $n$  bits  $Msg || 10 \dots 0 = M_1 || \dots || M_l$ . The compression function is sequentially applied to the previous bit state and block

$$H_{i+1} = \mathbf{g}_{i \cdot n}(H_i, M_{i+1}), \quad i = 0, \dots, l-1, \quad H_0 = IV \in \mathbb{F}_{2^8}^{8 \times 8},$$

where  $IV$  is a predefined constant. The counter  $N = i \cdot n \in \mathbb{F}_{2^8}^{8 \times 8}$  is the number of already hashed bits.

The bit length  $L$  and the checksum  $\Sigma = M_1 \boxplus \dots \boxplus M_l$  are «mixed» with the state at the finalizing stage

$$H_{l+1} = \mathbf{g}_0(H_l, L), \quad H_{l+2} = \mathbf{g}_0(H_{l+1}, \Sigma).$$

If 256-bit hash function is used, the output  $H_{l+2}$  is truncated to 256 bit.

The compression function  $\mathbf{g}_N(H, M)$  employs AES-like XSPL-cipher  $\mathbf{E}$  in the Miyaguchi-Preenel mode

$$\mathbf{g}_N(H, M) = \mathbf{E}(H \oplus N, M) \oplus H \oplus M = R, \quad \text{where}$$

$H \in \mathbb{F}_{2^8}^{8 \times 8}$  is the previous state of the hash function;

$M \in \mathbb{F}_{2^8}^{8 \times 8}$  is the message block;

$N \in \mathbb{F}_{2^8}^{8 \times 8}$  is the number of previously hashed bits;

$R \in \mathbb{F}_{2^8}^{8 \times 8}$  is the output (the next state of hash function).

The block cipher  $\mathbf{E}$  consists of 12 rounds and a post-whitening key addition. Each round consists of four operations:

$\mathbf{X}$  – modulo 2 addition of an input block with a round key;

$\mathbf{S}$  – parallel application of the fixed bijective substitution  $\mathbf{s}$  to each byte of the state;

$\mathbf{P}$  – transposition of the state;

$\mathbf{L}$  – parallel application of the linear transformation  $\mathbf{l}$  to each row of the state. In [21], it was shown that  $\mathbf{l}$ -transformation can be represented as the MDS matrix  $\mathbf{L}$  over  $\mathbb{F}_{2^8}^{8 \times 8}$ .

The block cipher formula is

$$E(K, M) = X[K_{13}]LPSX[K_{12}] \dots LPSX[K_2]LPSX[K_1](M).$$

The key schedule uses round constants  $RC_i \in \mathbb{F}_{2^8}^{8 \times 8}$ ,  $i = 1, 2, \dots, 12$ , and round keys  $K_i \in \mathbb{F}_{2^8}^{8 \times 8}$ ,  $i = 1, 2, \dots, 13$  are derived from a master key  $K_0$  as follows:

$$K_0 = H \oplus N, \quad K_1 = LPS(H \oplus N), \quad K_{i+1} = LPS(K_i \oplus RC_i), \quad i = 1, 2, \dots, 12.$$

We also denote the intermediate states before  $X$ ,  $S$ ,  $P$ ,  $L$  transformations in  $i$ -th round as  $X_i$ ,  $Y_i$ ,  $Z_i$ ,  $W_i$  correspondingly ( $X_1 = M$ ,  $Y_1 = M \oplus K_1$ ,  $Z_1 = S(Y_1)$ ,  $W_1 = P(Z_1)$ , etc.). The states in the key schedule are denoted in a similar way  $HX_i = K_i$ ,  $HY_i$ ,  $HZ_i$ ,  $HW_i$ , where  $H = HX_0$ ,  $HX_1 = LPS(H \oplus N)$  etc.

We define an  $r$ -round compression function with  $r + 1$  round keys as:

$$g(H, M) = (X[K_{r+1}]LPSX[K_r] \dots LPSX[K_1](M)) \oplus H \oplus M.$$

Next, we also assume that  $N$  is an arbitrary constant  $C_0$ .

HMAC-Streebog (see figure 1) is defined in [2] as

$$\text{HMAC-Streebog}(K, Msg) = H((\bar{K} \oplus opad) || H((\bar{K} \oplus ipad) || Msg)),$$

where  $\bar{K} \in \mathbb{F}_{2^8}^{8 \times 8}$  is obtained by padding the  $k$ -bit secret key  $K$  with zero bits,  $256 \leq k \leq 512$ ,  $opad$  and  $ipad$  are different nonzero constants.

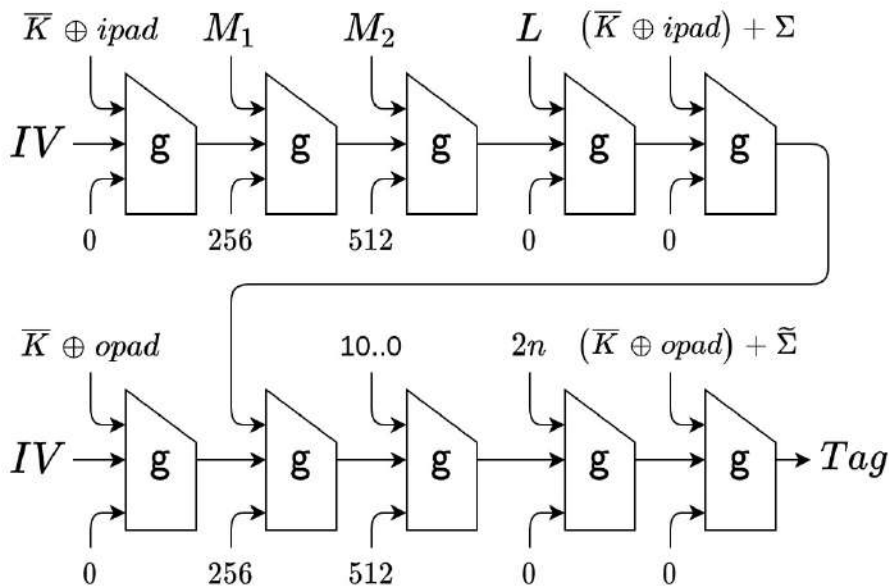


Figure 1: HMAC-Streebog-512,  $512 \leq L < 1024$ ,  $\Sigma = M_1 \boxplus M_2$ .

The secret key  $\overline{K}$  is used four times as part of the *message* input. The checksum  $\Sigma$  is directly controlled by the attacker and determines the relation between the keys. The sequence of the chosen messages  $Msg_1, \dots, Msg_q$  generates the sequence of the chosen relations  $\Sigma_1, \dots, \Sigma_q$ . Hence, the adversary has as many related keys as needed. The state  $H$  is usually not known to the attacker, but we assume the opposite. Firstly,  $H$  can be revealed as a result of some generic attack against HMAC, secondly, this may be convenient for a formal security proof. The output  $R$  is observed only after the last call of  $\mathbf{g}$ , but if, for example,  $\mathbf{H}(K || Msg)$  is used instead of HMAC, then  $R$  is known.

Thus, these considerations motivate us to examine the security of

$$\mathbf{g}(H, M) = R, \quad M = (K \oplus \Phi) \boxplus \Sigma,$$

where  $K$  is the secret 512-bit key, the output  $R$  is observed, and  $H, \Phi, \Sigma$  are chosen adaptively. If  $\mathbf{g}$  is secure even in the described setting, then there is no reason to worry about cases when the opponent has fewer opportunities.

## 4 Generic attack

The key-recovery attacks on the cryptoalgorithm are usually compared to a simple guessing of the key. Obviously, a  $k$ -bit key can be found with  $2^{k-1}$  trials on average.

However, in the related-key setting we have another generic attack. Let, for example, the adversary attacks an arbitrary block cipher  $\mathbf{E}$ . The sequence of ciphertexts  $C_1, \dots, C_q$  is the result of encryption of the same text  $P$ , but with the different key

$$C_i = \mathbf{E}(K \oplus \Phi_i, P), \quad i = 1, \dots, q.$$

Pairs  $(C_i, \Phi_i)$  are sorted by  $C_i$  and stored in memory. The attacker makes  $t$  guesses  $\tilde{K}$ . If the value of  $\tilde{C} = \mathbf{E}(\tilde{K}, P)$  exists in memory  $\tilde{C} = C_j$ , then surely  $K = \tilde{K} \oplus \Phi_j$ ,  $1 \leq j \leq q$ . One revealed key allows to trivially find all the others. The probability of successful guessing in one attempt is  $q \cdot 2^{-k}$ . Hence, if  $t \cdot q = 2^k$  then the probability of the successful attack exceeds  $\frac{1}{2}$ .

Therefore,  $2^r$  related keys allow to mount generic attack with  $2^{k-r}$  time and  $2^r$  memory complexities. We emphasize that the time complexity of any related-key attack should be compared with  $2^{k-r}$ , not  $2^k$ .

The optimal time complexity is  $2^{k/2}$  with  $r = \frac{k}{2}$ . Informally speaking, any cryptoalgorithm with a  $k$ -bit key provides only  $\frac{k}{2}$ -bit security if the number of the related keys available to the adversary is unlimited. Also note that the



type of relation can be rather arbitrary and include, for example, modular addition. The main thing is that the attacker has access to encryption with different keys and the relations between the keys are known.

## 5 Single-key attack

At the beginning, we consider the case when the message  $M = (K \oplus \Phi) \boxplus \Sigma$  is secret and  $\Phi = \Sigma = 0$ . The attack against 7 rounds in such conditions was considered earlier in [23]. We use the similar approach and construct the low-probability attack against 10 rounds of

$$g(H, M) = E(H, M) \oplus H \oplus M = R.$$

The master-key  $H$  of the underlying block cipher is directly chosen by the adversary

$$E(H, M) \oplus M = R \oplus H = \tilde{R}.$$

The key-recovery method consists of two stages.

«Offline» stage uses the rebound approach [24]. About  $2^{28}$  pairs  $(H, H')$  are generated. Each pair determines a truncated differential trail  $\Delta K_1 \rightarrow \dots \rightarrow \Delta K_{11}$ . Some precomputations are also performed under the assumption that  $\Delta Y_9 = \Delta K_9$ .

«Online» stage. For each input pair  $(H, H')$  we get the output  $(\tilde{R}, \tilde{R}')$ . The truncated related-key differential trail  $\Delta M \rightarrow \dots \rightarrow \Delta \tilde{R}$  is realized with a probability of at least  $p_{trail} = 2^{-224}$ . For each pair  $(\tilde{R}, \tilde{R}')$  we construct on average one possible value of the unknown internal state and check it directly. If the rare event actually occurred, then we definitely obtain the true key.

The patterns of the active S-boxes are

$$\begin{aligned} \Delta K_1 \rightarrow \dots \rightarrow \Delta K_{11} &: \ll 8-1-8-64-8-1-8-64-64-64-64 \gg, \\ \Delta M \rightarrow \Delta Y_1 \rightarrow \dots \rightarrow \Delta \tilde{R} &: \ll 0-8-0-8-0-8-0-8-0-64-64-64 \gg. \end{aligned}$$

The offline stage constructs the suitable round keys for the block cipher. Choose arbitrary nonzero bytes in one arbitrary column of the difference  $\Delta HW_3$  (highlighted with green on figure 2).

Propagate forward to  $\Delta HY_4 = X[C_4]L(\Delta HW_3)$ . Similarly in the backward direction  $\Delta HZ_4 = P^{-1}L^{-1}(\Delta K_5)$ . Thus, we have  $255^8 \cdot 8 \cdot 255^8 \cdot 8 \approx 2^{134}$  pairs  $(\Delta HY_4, \Delta HZ_4)$ .

Solve equation  $S(HY_4 \oplus \Delta HY_4) \oplus S(HY_4) = \Delta HZ_4$ . We get a total of more than  $2^{132}$  solutions (see Appendix A).

In «outbound phase» we compute

$$K_1 = X[C_1]S^{-1} \dots P^{-1}L^{-1}X[C_4](HY_4) \text{ and } K_{11} = LPSX[C_{10}] \dots LPS(HY_4).$$

We assume that the part  $\Delta K_1 \leftarrow \Delta K_2 \leftarrow \Delta K_3$  of the constructed trail match the pattern «8 – 1 – 8» with probability  $8 \cdot 255/255^8 \approx 2^{-53}$  due to the transition «1  $\leftarrow$  8». Note that any of eight possible patterns «1  $\leftarrow$  8» is suitable. Similar reasoning is true for  $\Delta K_6 \rightarrow \Delta K_7 \rightarrow \Delta K_8$  (and any values of  $\Delta K_9 \rightarrow \Delta K_{10} \rightarrow \Delta K_{11}$  is appropriate.). Actually 64 truncated trails are used, eight appropriate propagation possibilities in the backward and the same for forward.

As a result we obtain about  $q_{pair} = 2^{26} = 2^{132-53-53}$  pairs  $(H, H')$  and approximately  $2^{23} = 2^{26}/8$  of them have the active first column. The time complexity of the offline stage is about  $t_{offline} = 2^{134}$  operations.

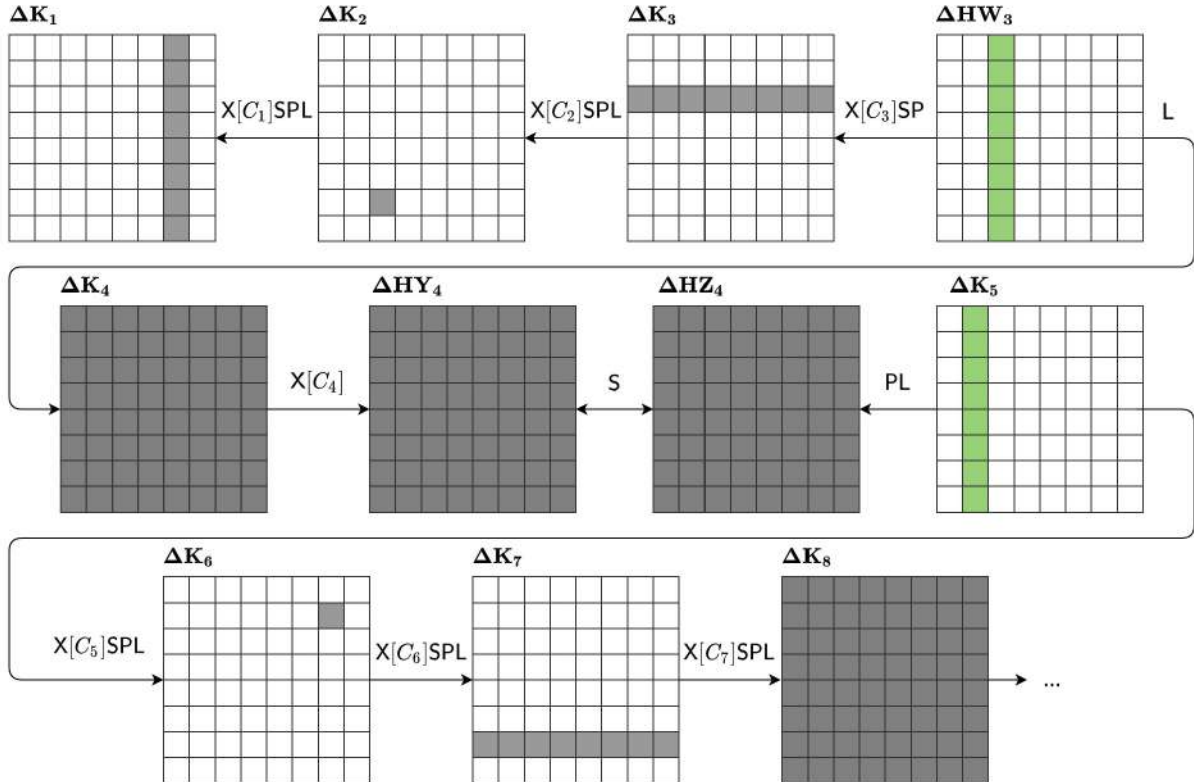


Figure 2: Offline stage. One of the possible truncated differential trail over first eight round keys.

At the online stage, pairs  $(\tilde{R}, \tilde{R}')$  are requested for each  $(H, H')$  from the «oracle».

We expect four internal collisions at the same time (figure 3). In the considered single-key setting,  $M = M'$  and  $\Delta M = 0$ . The differences  $\Delta K_1$ ,  $\Delta K_3$ ,  $\Delta K_5$ ,  $\Delta K_7$  induce eight active bytes (one row or one column of the

state) in the «encryption». If the transitions through  $S$  are the same in both «encryption» and «key schedule» then  $\Delta K_2, \Delta K_4, \Delta K_6, \Delta K_8$  make a zero difference in «encryption».

Before the first non-linear layer  $\Delta Y_1 = \Delta K_1 \oplus \Delta M = \Delta K_1$ . We hope that  $\Delta H Z_1 = \Delta Z_1$ . The transition  $\Delta H Y_1 \rightarrow \Delta H Z_1$  is possible, hence, the probability  $\Delta Y_1 \rightarrow \Delta Z_1$  is not less than  $p_{coll} = (2/256)^8 = 2^{-56}$ . If actually  $\Delta H Z_1 = \Delta Z_1$  then we obtain the first internal collision

$$\Delta Y_2 = \Delta K_2 \oplus \Delta X_2 = LP(\Delta H Z_1) \oplus LP(\Delta Z_1) = 0.$$

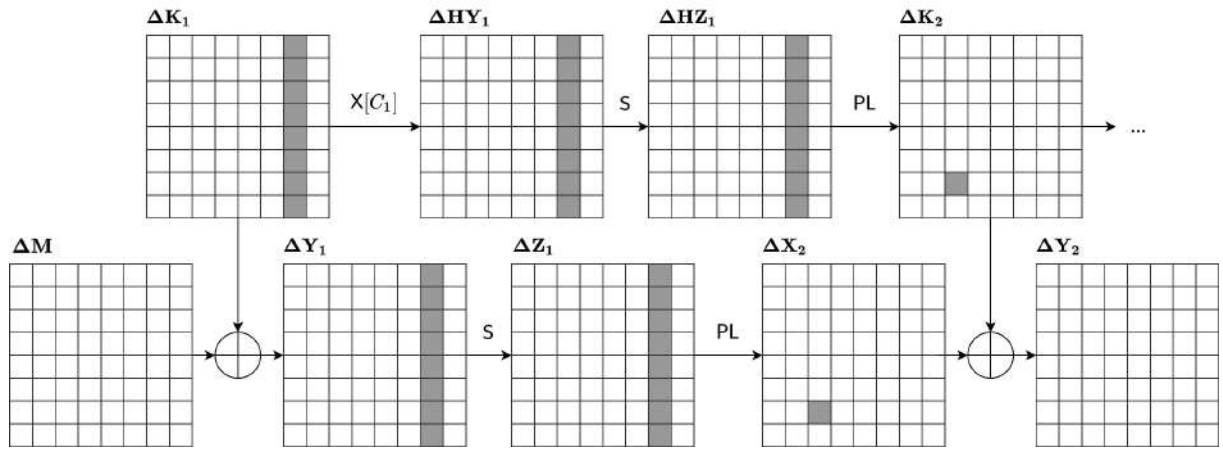


Figure 3: Online stage. Truncated related-key differential trail. The first round.

The same is true for  $\Delta Y_3 = \Delta K_3$  and «parallel» transitions  $\Delta H Y_3 \rightarrow \Delta H Z_3, \Delta Y_3 \rightarrow \Delta Z_3$  (figure 4). We also assume that  $\Pr(\Delta Z_3 = \Delta H Z_3) = p_{coll}$ .

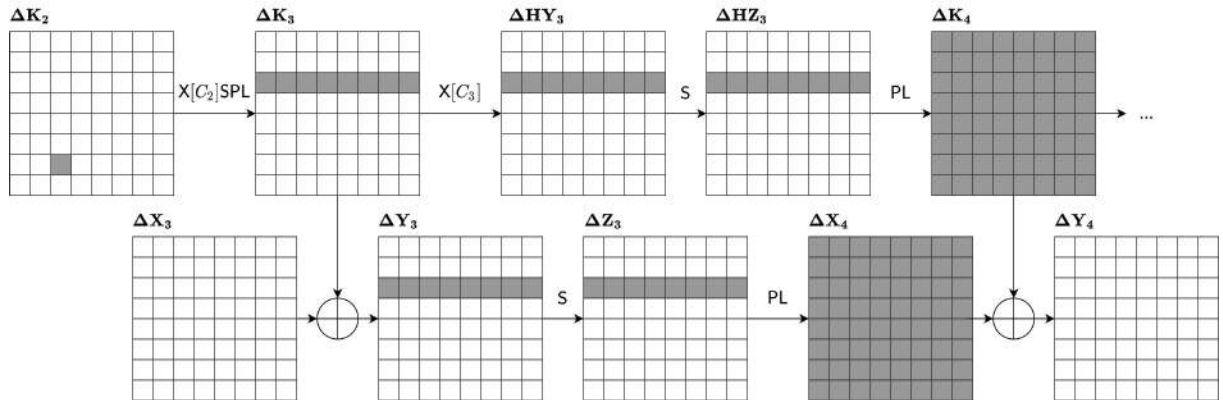


Figure 4: Online stage. The third round.

Similarly for the third and the fourth internal collision (figure 5),  $\Pr(\Delta Z_5 = \Delta H Z_5) = p_{coll}, \Pr(\Delta Z_7 = \Delta H Z_7) = p_{coll}$ .

Therefore, we have

$$p_{\text{trail}} = \Pr(\Delta \mathbf{X}_9 = 0) = \Pr(\Delta \mathbf{Y}_9 = \Delta \mathbf{K}_9) \geq (p_{\text{coll}})^4 = 2^{-56 \cdot 4} = 2^{-224}.$$

We use this distinguishing feature to construct the attack.

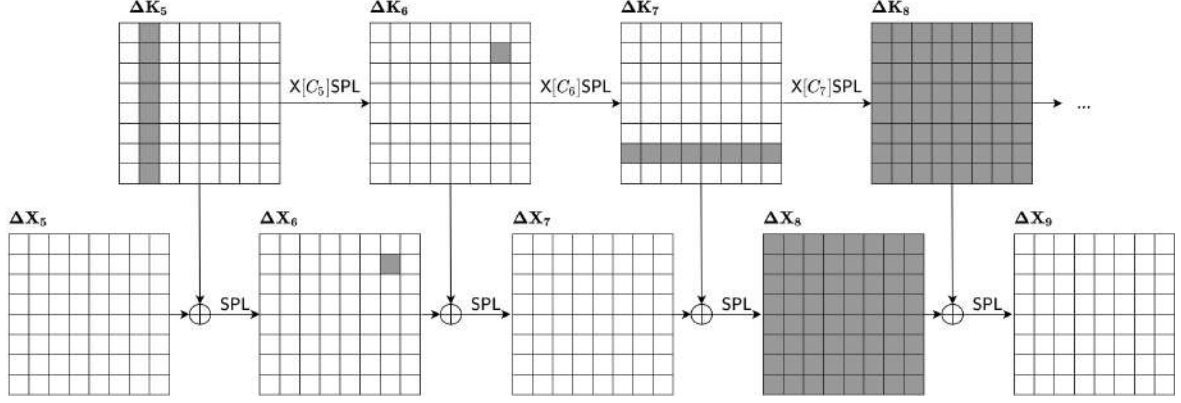


Figure 5: Online stage. Rounds 5, 6, 7 and 8.

After the rebound, the precomputations are performed at the offline stage. We have  $2^{26}$  pairs  $(H, H')$  and derived round keys  $(K_1, \dots, K_{11}), (K'_1, \dots, K'_{11})$ . Assuming that the trail is realized,  $(H, H')$  determines the only one  $\Delta \mathbf{Y}_9 = \Delta \mathbf{K}_9$ . Try all possible values in the column  $Y_9[\cdot, i]$  and propagate  $\Delta \mathbf{Y}_9[\cdot, i]$  to  $\Delta \mathbf{W}_{10}[\cdot, i]$ ,  $i = 0, \dots, 7$ .

For fixed  $(H, H')$  eight tables are stored in memory,  $i$ -th table contains the sequence of sorted values  $\Delta \mathbf{W}_{10}[\cdot, i] = W_{10}[\cdot, i] \oplus W'_{10}[\cdot, i]$ ,

$$\begin{aligned} W_{10}[\cdot, i] &= (\text{PS}(K_{10}[i, \cdot] \oplus \text{LPS}(Y_9[\cdot, i]))) , \\ W'_{10}[\cdot, i] &= (\text{PS}(K'_{10}[i, \cdot] \oplus \text{LPS}(\Delta \mathbf{Y}_9[\cdot, i] \oplus Y_9[\cdot, i]))) . \end{aligned}$$

and corresponding set of  $W_{10}[\cdot, i]$ . Assuming that after two nonlinear layers,  $\Delta \mathbf{W}_{10}[\cdot, i]$  is distributed uniformly, one value of  $\Delta \mathbf{W}_{10}[\cdot, i]$  corresponds to one value of  $W_{10}[\cdot, i]$  on average.

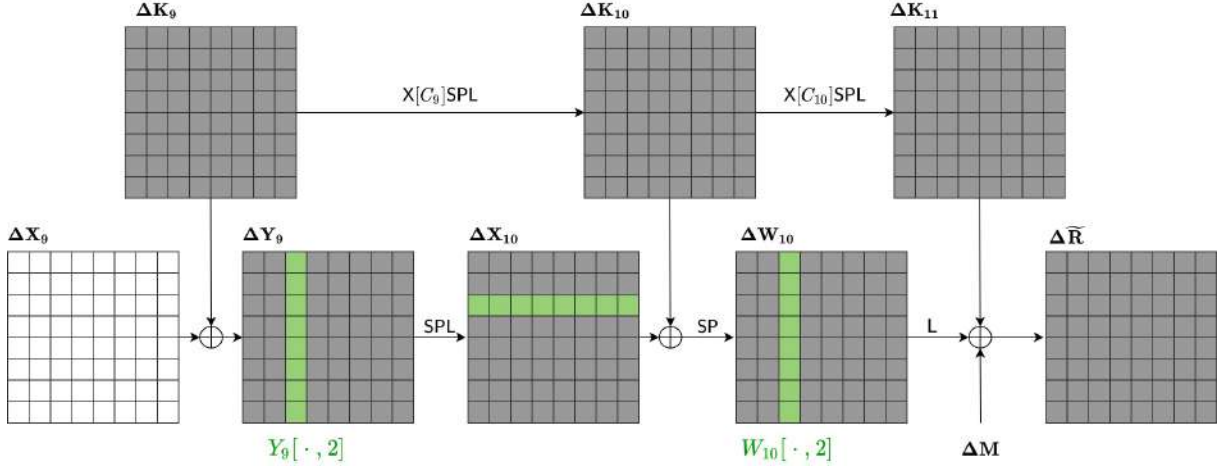


Figure 6: Additional precomputation at the offline stage.

For all  $(H, H')$  about  $2^{26} \cdot 8 \cdot 2^{64}$  tables are constructed. In total, this step requires about  $2 \cdot 2^{26} \cdot 8 \cdot 2^{64} = 2^{94}$  computations and the same number of  $n$ -bit blocks in memory. Hence, the complexity of the offline stage almost does not increase,  $t_{offline} = 2^{134} + 2^{93} \approx 2^{134}$ .

Consider again the online stage. The pair  $(H, H')$  defines the last round keys  $(K'_{11}, K'_{11})$  and the output pair  $(\tilde{R}, \tilde{R}')$ . The difference  $\Delta \mathbf{M}$  is also known to the adversary (in the single-key setting  $\Delta \mathbf{M} = 0$ ). If the differential trail really happened ( $\Delta \mathbf{X}_9 = 0$ ), then  $i$ -th column of

$$\Delta \mathbf{W}_{10} = \mathbf{L}^{-1}(\Delta \tilde{\mathbf{R}} \oplus \Delta \mathbf{K}_{11} \oplus \Delta \mathbf{M})$$

must be in  $i$ -th table. Otherwise, the pair will surely be discarded. Usually one solution  $W_{10}[\cdot, i]$  for  $i$ -th column is found. We construct internal state  $W_{10}$ , compute  $M = \mathbf{L}(W_{10}) \oplus K_{11} \oplus \tilde{R}$  and check them with the other input-output pair  $(H, R)$ . The average time complexity of the online stage is estimated as  $t_{online} \approx q_{pair}$ . The probability of success is negligible

$$p_{1k-attack} \approx q_{pair} \cdot p_{trail} = 2^{26} \cdot 2^{-224} = 2^{-198}.$$

## 6 Related-key attacks

The single-key low-probability attack presented above can be easily transformed into attack in the related-key setting.

Let's perform the offline stage once and store  $2^{26}$  convenient pairs  $(H, H')$  and precomputed tables in memory. We use about  $2^r = 2^{198} = (p_{1k-attack})^{-1}$  related keys and try the online stage against each of them independently. If one key is recovered, then all the others are can, too, be easily found from

known relations. The probability of success is now significant and is estimated as  $1 - (1 - p_{1k\text{-attack}})^{2^r} \approx 1 - e^{-1} \approx 0.63$ .

The time complexity is  $t = 2^{134} + 2^{26} \cdot 2^{198} \approx 2^{224}$  (for comparison, the generic method  $t = 2^{k-r} = 2^{314}$ ). It is not difficult to see that almost any possible relation can be used ( $M \oplus \Phi$ ,  $M \boxplus \Sigma$ ,  $M \oplus \Phi \boxplus \Sigma$  etc.).

However, a more effective attacks exists if the relation is bitwise xor (i.e.  $M \oplus \Phi$ ). We describe them in the following two subsections.

## 6.1 Reducing the number of related keys

At the offline stage, we select only those pairs  $(H, H')$  that activate only one chosen column  $\Delta K_1[\cdot, 0] \neq 0$ ,  $\Delta K_1[\cdot, 1] = \dots = \Delta K_1[\cdot, 7] = 0$ . The number of convenient pairs has been reduced to  $q_{pair} = 2^{23} = 2^{26}/8$ .

At the online stage, we use many sets

$$\mathbb{M}_i = \{M \oplus \Phi'_i \oplus \Phi_j\}, \Phi_j[\cdot, 0] \neq 0, \Phi_j[\cdot, 1] = \dots = \Phi_j[\cdot, 7] = 0, j = 0, \dots, 2^{64} - 1,$$

of the related keys. The values of  $\Phi'_i$  are chosen so that  $\mathbb{M}_{i_1} \cap \mathbb{M}_{i_2} = \emptyset$ ,  $\forall i_1 \neq i_2$ . The set induces  $(2^{128} - 2^{64}) \approx 2^{128}$  different pairs  $(M, M')$ , where also only the first column of the difference may be active  $\Delta M[\cdot, 0] \neq 0$ , other columns are obviously inactive. Note that the pairs  $(M, M)$  are also used. The pairs  $(M, M')$  and  $(M', M)$  are distinct if  $M \neq M'$ . Indeed,  $(M, M')$  and  $(H, H')$ ,  $H \neq H'$  generates two related-key differential trails,

$$(M \oplus H) \oplus (M' \oplus H') = (M' \oplus H) \oplus (M \oplus H'), \text{ but in general} \\ \mathcal{S}(M \oplus H) \oplus \mathcal{S}(M' \oplus H') \neq \mathcal{S}(M' \oplus H) \oplus \mathcal{S}(M \oplus H').$$

Hence, about  $2^{128} \cdot q_{pair} = 2^{151}$  starting points are obtained with one  $\mathbb{M}$  (at the same time, the number of the required queries is  $2 \cdot q_{pair} \cdot 2^{64} = 2^{88}$ ).

The probability of the resulting trail is slightly worse. If  $\Delta M = 0$ , then  $\Delta Y_1 = \Delta K_1$  and there is always a possibility to the transition  $\Delta Y_1 \rightarrow \Delta Z_1$ , where  $\Delta Z_1 = \Delta H Z_1$ . Otherwise,  $\Delta Y_1 \neq \Delta K_1$  and the target transition  $\Delta Y_1 \rightarrow \Delta Z_1$  may be impossible. However, assuming that  $\Delta Y_1[\cdot, 0]$  is random, we can treat  $\Delta Z_1$  also as random value and estimate

$$p_{rand\text{-}coll} = \Pr(\Delta Z_1 = \Delta H Z_1) = 2^{-64} < p_{coll}.$$

The probability of the modified truncated trail is

$$p'_{trail} = p_{rand\text{-}coll} \cdot (p_{coll})^3 = 2^{-232}.$$

The rest of the attack is the same.

Thus, we need about  $q_{set} = 2^{81} = (p'_{trail} \cdot 2^{151})^{-1}$  sets  $\mathbb{M}$ . The total number of the related keys is  $q_{key} = q_{set} \cdot 2^{64} = 2^{145} = 2^r$ . The number of the queries  $q = 2 \cdot q_{pair} \cdot 2^{64} \cdot q_{set} = 2^{169}$ . The memory for tables at the precomputation is  $2 \cdot q_{pair} \cdot 8 \cdot 2^{64} = 2^{91}$ . The time complexity slightly increases  $t = q_{set} \cdot 2^{128} \cdot q_{pair} = (p'_{trail})^{-1} = 2^{232}$ , but for the generic attack  $t = 2^{k-r} = 2^{369}$ .

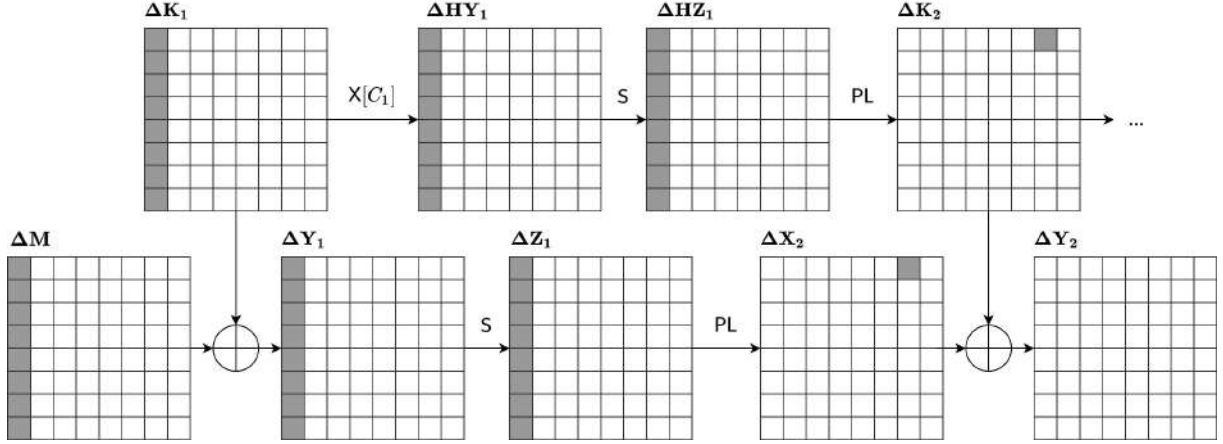


Figure 7: Online stage. Truncated related-key differential trail. The first round.  $\Delta M[\cdot, 0] \neq 0$ .

## 6.2 Extension to 11 round

We change the truncated differential trail used by adding one round at the beginning (figure 8). The patterns of the active S-boxes are

$$\begin{aligned} \Delta K_1 &\rightarrow \dots \rightarrow \Delta K_{12} : \ll 64-8-1-8-64-8-1-8-64-64-64-64 \gg, \\ \Delta M &\rightarrow \Delta Y_1 \rightarrow \dots \rightarrow \Delta Y_{12} : \ll 64-0-8-0-8-0-8-0-8-0-64-64-64 \gg. \end{aligned}$$

The rebound starts with  $\Delta HW_4$  and  $\Delta K_6$  instead of  $\Delta HW_3$  and  $\Delta K_5$ . The remaining steps are similarly «shifted to the right» (see Appendix B).

To provide  $\Delta Y_1 = \Delta X_2 = 0$ , we must use  $\Delta M = \Delta K_1$ . In this case, the probability of the rare event does not change  $p_{trail} = (p_{coll})^4 = 2^{-224}$ .

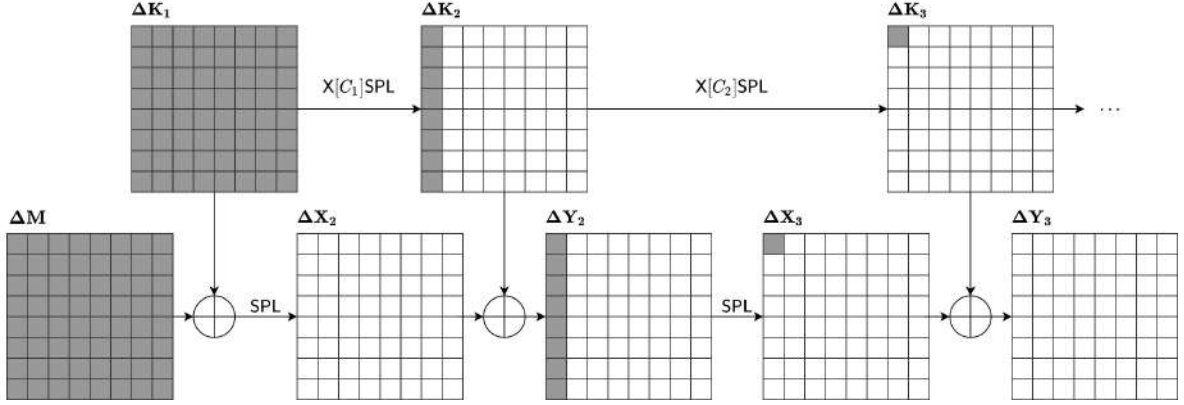


Figure 8: Truncated related-key differential trail,  $\Delta M = \Delta K_1$ .

We use only one pair  $(H, H')$  after the offline stage. Hence, only one sequence  $(K_1, K'_1) \rightarrow \dots \rightarrow (K_{12}, K'_{12})$  is used, and  $2 \cdot 8 \cdot 2^{64} = 2^{68}$   $n$ -bit blocks of memory are required to store the rows of  $\Delta W_{11}$  and  $W_{11}$  after the precomputation.

The  $i$ -th pair of the related keys is

$$(M_i, M'_i) = (M \oplus \Phi_i, M \oplus \Phi_i \oplus \Delta K_1), \quad i = 1, \dots, \frac{q_{key}}{2},$$

different values of  $\Phi_i$  should give  $q_{key}$  different keys, and  $\Delta M = \Delta K_1$  is always true. Again, each pair of keys gives two points to start for the online stage

$$((H, M_i), (H', M'_i)) \text{ and } ((H, M'_i), (H', M_i)).$$

The success probability is also  $(1 - e^{-1}) \approx 0.63$  with  $q_{key} = (p_{trail})^{-1}$ . The query complexity is  $q = 2 \cdot q_{key} = 2^{225}$ . As in previous attacks, the time complexity is equal to the number of starting points

$$t \approx t_{offline} + 2 \cdot \frac{q_{key}}{2} = q_{key} = 2^{224} = 2^r.$$

For comparison, the complexity of the generic method  $t = 2^{k-r} = 2^{288}$ .

## 7 Conclusion

In many practical cases, Streebog hashes the secret key joined to the message. Due to the checksum modulo  $2^n$  in the finalization, the related keys always arise. For example, in HMAC-Streebog the one processed block is  $M = (K \oplus \Phi) \boxplus \Sigma$ , where  $K$  is the secret key,  $\Phi$  is known,  $\Sigma$  is chosen adaptively by the adversary. Therefore, this motivates us to investigate round-reduced Streebog compression function  $g(H, M)$  with the secret  $M$  under above mentioned relations.



Among all the threat models for symmetric keyed cryptoalgorithms, the related-key setting is one of the most powerful. We present key-recovery algorithms up to 10 rounds (out of 12) when almost any relations exist (e.g. addition modulo 2 or modulo  $2^n$ ). If only bitwise xor is used then the attack can be extended for 11 rounds. The rebound approach and the related-key truncated differential trails are extensively used. The time complexity of the methods is close to that of the generic approaches.

Thus, we have significant evidence that Streebog compression function is hard to break even in the threat model under consideration. Therefore, another argument was obtained in favor of the security of Streebog-based keyed algorithms.

## 8 Acknowledgements

The author is grateful to Andrey Scherbachenko for the careful consideration of the article and many suggestions that significantly improved the quality of the text.

## References

- [1] *GOST R 34.11-2012 – National standard of the Russian Federation – Information technology – Cryptographic data security – Hash function*, 2012.
- [2] *R 50.1.113-2016 – Information technology – Cryptographic data security – Cryptographic algorithms accompanying the use of electronic digital signature algorithms and hash functions*, 2016.
- [3] Damgård I., “A Design Principle for Hash Functions”, *LNCS*, CRYPTO 1989, **435**, ed. Brassar G., Springer, Heidelberg, 1990, 416–427.
- [4] Merkle R., “One way Hash Functions and DES”, *LNCS*, CRYPTO 1989, **435**, ed. Brassard G., Springer, Heidelberg, 1990, 428–446.
- [5] Biham E., “New types of cryptanalytic attacks using related keys (extended abstract)”, *LNCS*, EUROCRYPT 93, **765**, ed. Helleseht T., Springer, Berlin, Heidelberg, 1993.
- [6] Knudsen L., “Truncated and Higher Order Differentials”, 2nd International Workshop on Fast Software Encryption (FSE 1994), 1994, 196–211.
- [7] Bellare M., Canetti R., Krawczyk H., “Keying Hash Functions for Message Authentication”, *LNCS*, Advances in Cryptology – Crypto’96, **1109**, ed. Kobitz N., Springer, Berlin, Heidelberg, 1996, 1–15.
- [8] Bellare M., “New Proofs for NMAC and HMAC: Security without Collision-Resistance”, *LNCS*, Advances in Cryptology – CRYPTO 2006, **4117**, ed. Dwork C., Springer, Berlin, Heidelberg, April 2014.
- [9] Guo J., Jean J., Leurent G., Peyrin T., Wang L., “The Usage of Counter Revisited: Second-Preimage Attack on New Russian Standardized Hash Function”, *LNCS*, Selected Areas in Cryptography – SAC 2014, **8781**, ed. Joux A., Youssef A., Springer, Cham, 2014.
- [10] AlTawy R., Youssef A. M., “Integral distinguishers for reduced-round Stribog”, *Information Processing Letters*, **114** (2014).
- [11] AlTawy R., Youssef A. M., “Preimage Attacks on reduced-round Stribog”, *LNCS*, Progress in Cryptology – AFRICACRYPT 2014, **8469**, ed. Pointcheval D., Vergnaud D., Springer, Cham, 2014.

- [12] AlTawy R., Kircanski A., Youssef A. M., “Rebound attacks on Stribog”, *LNCS*, Information Security and Cryptology – ICISC 2013, **8565**, ed. Lee HS., Han DG., Springer, Cham, 2014.
- [13] Lin D., Xu S., Yung M., “Cryptanalysis of the Round-Reduced GOST Hash Function”, *LNCS*, Information Security and Cryptology. Inscrypt 2013., **8567**, Springer, Cham, 2014.
- [14] Ma B., Li B., Hao R., Li X., “Improved cryptanalysis on reduced-round GOST and Whirlpool hash function”, *LNCS*, Applied Cryptography and Network Security. ACNS 2014., **8479**, ed. Boureanu I., Owesarski P., Vaudenay S., Springer, Cham, 2014.
- [15] Wang Z., Yu H., Wang X., “Cryptanalysis of GOST R Hash Function”, *Information Processing Letters*, **114** (2014), 655–662.
- [16] Kölbl S., Rechberger C., “Practical Attacks on AES-like Cryptographic Hash Functions”, *LNCS*, Progress in Cryptology – LATINCRYPT 2014, **8895**, ed. Aranha D., Menezes A., Springer, Cham, 2014.
- [17] Ma B., Li B., Hao R., Li X., “Improved (Pseudo) Preimage Attacks on Reduced-Round GOST and Grøstl-256 and Studies on Several Truncation Patterns for AES-like Compression Functions”, *LNCS*, Advances in Information and Computer Security. IWSEC 2015, **9241**, ed. Tanaka K., Suga Y., Springer, Cham, 2015, 79–96.
- [18] Abdelkhalek A., AlTawy R., Youssef A. M., “Impossible Differential Properties of Reduced Round Streebog”, *LNCS*, Codes, Cryptology, and Information Security. C2SI 2015, **9084**, ed. El Hajji S., Nitaj A., Carlet C., Souidi E., Springer, Cham, 2015, 274–286.
- [19] Rongjia Li, Chenhui Jin, Ruya Fan, “Improved Integral Distinguishers on Compression Function of GOST R Hash Function”, *Computer Journal*, **62** (2019), 535–544.
- [20] Tingting Cui, Wei Wang, Meiqin Wang, “Distinguisher on full-round compression function of GOST R”, *Information Processing Letters*, **156** (2019).
- [21] Kazymyrov O., Kazymyrova V., “Algebraic Aspects of the Russian Hash Standard GOST R 34.11-2012”, *Cryptology ePrint Archive, Report 2013/556*, 2013.
- [22] Dinur I., Leurent G., “Improved Generic Attacks Against Hash-based MACs and HAIFA”, *LNCS*, Advances in Cryptology – CRYPTO 2014, **8616**, ed. Garay J.A., Gennaro R., Springer, Berlin, Heidelberg, 2014.
- [23] Kiryukhin V., “Streebog compression function as PRF in secret-key settings”, *CTCrypt 2021*, 2021.
- [24] Mendel F., Rechberger C., Schl affer M., S oren S. Thomsen, “The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Gr ostl”, *LNCS*, Fast Software Encryption. FSE 2009, **5665**, ed. Dunkelman O., Springer, Berlin, Heidelberg, 2009.

## A Differential properties of Streebog’s S-box

The differential distribution table (DDT) is defined as follows

$$\text{DDT}[\Delta \mathbf{x}][\Delta \mathbf{y}] = |\{x : \mathbf{s}(x) \oplus \mathbf{s}(x \oplus \Delta \mathbf{x}) = \Delta \mathbf{y}\}|,$$

where  $\mathbf{s} : \mathbb{F}_{2^8} \rightarrow \mathbb{F}_{2^8}$ ,  $x, \Delta \mathbf{x}, \Delta \mathbf{y} \in \mathbb{F}_{2^8}$ .

The distribution of the number of solutions for Streebog’s S-box is shown in the table below.

Solutions	0	2	4	6	8	256
Number	38235	22454	4377	444	25	1

For random non-zero  $\Delta \mathbf{x}, \Delta \mathbf{y} \in \mathbb{F}_{2^8} \setminus 0$  the probability that at least some solution exists is

$$p = \Pr(|\{x : \Delta \mathbf{y} = \mathbf{s}(x) \oplus \mathbf{s}(x \oplus \Delta \mathbf{x})\}| > 0) = \frac{22454 + 4377 + 444 + 25}{255^2}.$$

Let  $\Delta \mathbf{x} \neq 0$ ,  $\Delta \mathbf{y} \neq 0$ , and it is also known that the equation

$$\mathbf{s}(x) \oplus \mathbf{s}(x \oplus \Delta \mathbf{x}) = \Delta \mathbf{y}$$

has a solution  $x$ . Then we get a conditional distribution of the number of solutions

$$\left( \begin{array}{cccc} 2 & 4 & 6 & 8 \\ \frac{22454}{27300} & \frac{4377}{27300} & \frac{444}{27300} & \frac{25}{27300} \end{array} \right).$$

The expected value of such a distribution (i.e., the average number of solutions provided that at least one solution exists) is

$$\frac{1}{27300} (2 \cdot 22454 + 4 \cdot 4377 + 6 \cdot 444 + 8 \cdot 25) = \frac{2^{16} - 2^8}{27300} = 2.39 \dots = z.$$

*The case « $\mathbf{S}(\Delta \mathbf{H}\mathbf{Y}_4 \oplus \mathbf{H}\mathbf{Y}_4) \oplus \mathbf{S}(\mathbf{H}\mathbf{Y}_4) = \Delta \mathbf{H}\mathbf{Z}_4$ »*

We assume, that  $\Delta \mathbf{H}\mathbf{Z}_4$  is a random difference. We also know that  $\Delta \mathbf{H}\mathbf{Z}_4$  consisting only of non-zero bytes. Fix the position of columns in  $\Delta \mathbf{H}\mathbf{W}_3$  and  $\Delta \mathbf{K}_5$ .

Each row in  $\Delta \mathbf{H}\mathbf{Y}_4$  is also completely non-zero and belongs to a set of 255 elements.

The probability that a single byte matches is  $p \approx 0.419$ . Hence a row matches with a probability of  $p^8 \approx 2^{-10}$ .

The probability that among the allowed  $\Delta \mathbf{H}\mathbf{Y}_4[\mathbf{0}, \cdot]$  there is a suitable one is  $1 - (1 - p^8)^{255} \approx 2^{-2.2}$ .

Therefore, the probability for a match of all 8 rows equals to  $2^{-2 \cdot 2 \cdot 8} = 2^{-17.6}$ .

Each pair  $(\Delta \mathbf{H}\mathbf{Y}_4, \Delta \mathbf{H}\mathbf{Z}_4)$  for which the equation is solvable gives on average of  $z^{64} \approx 2^{80.4}$  solutions.

We have  $255^8 \approx 2^{64}$  possible values  $\Delta \mathbf{H}\mathbf{Z}_4$ .

Repeat for all pairs of columns in  $\Delta \mathbf{H}\mathbf{W}_3$  and  $\Delta \mathbf{K}_5$ .

As a result we obtain about

$$8 \cdot 8 \cdot 2^{64+80.4-17.6} \approx 2^{132}$$

valid states  $\mathbf{H}\mathbf{Y}_4$ .

## B Detailed pictures for 11-round attack

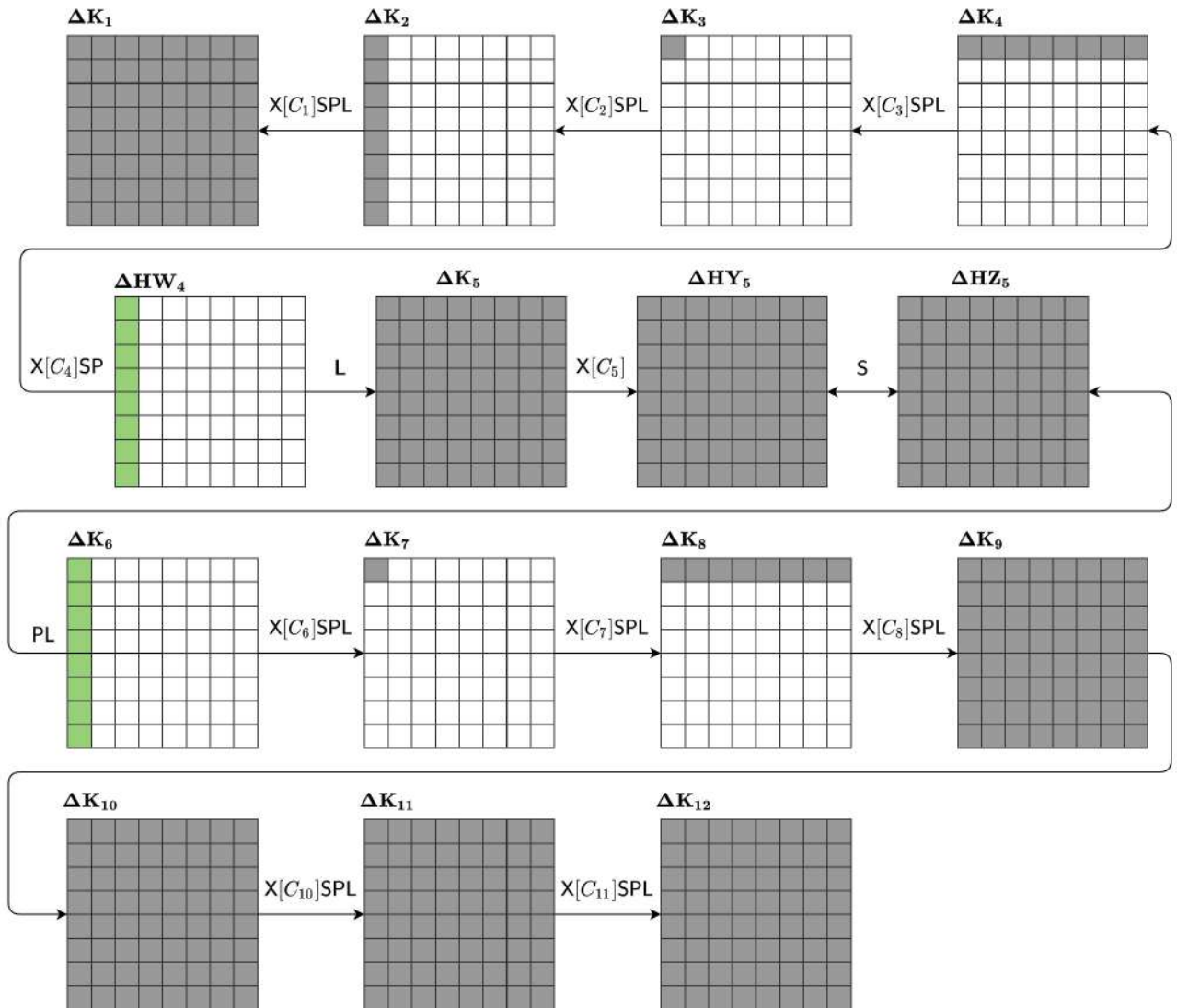


Figure 9: Offline stage. Rebound.

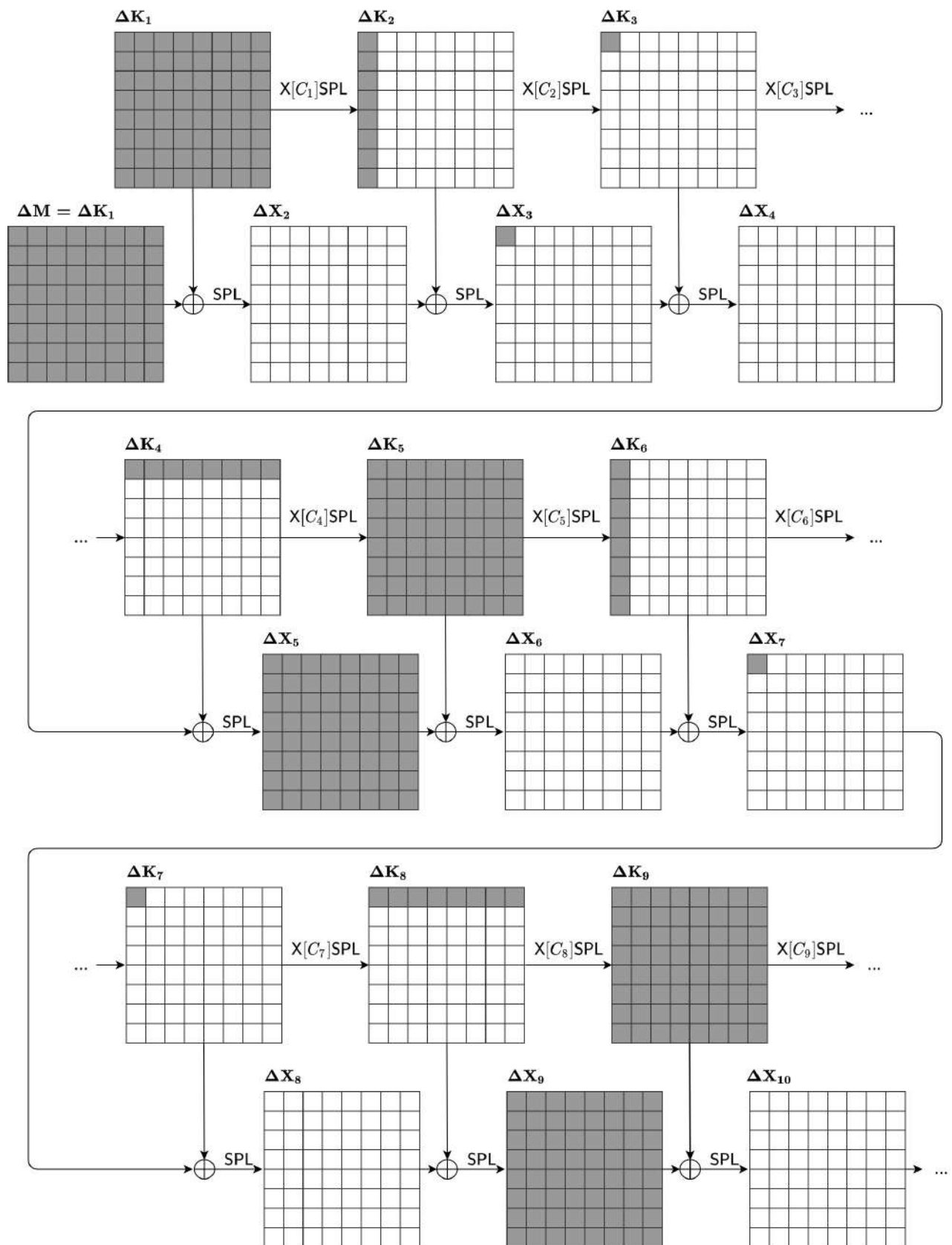


Figure 10: The truncated related-key differential trail. 9 rounds.

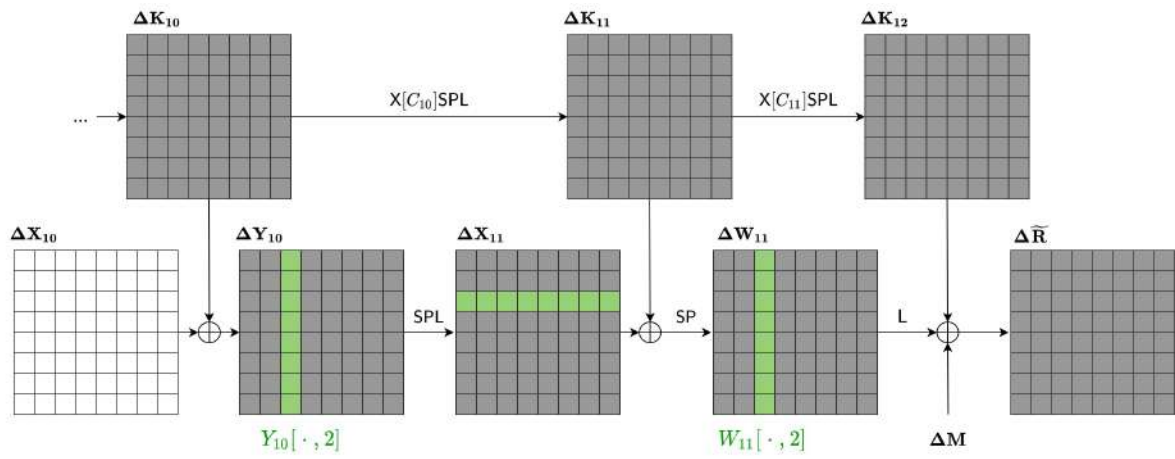


Figure 11: Additional precomputation at the offline stage.

# On differential characteristics modulo $2^n$ of the composition of bitwise exclusive-or and a bit rotation

Nikolay Kolomeec, Ivan Sutormin, Denis Bykov,  
Matvey Panferov, and Tatiana Bonich

Sobolev Institute of Mathematics, Russia  
kolomeec@math.nsc.ru, ivan.sutormin@gmail.com, den.bykov.2000i@gmail.com,  
magold1102@gmail.com, tanya50997@gmail.com

## Abstract

Properties of the additive differential probability  $\text{adp}^{\text{XR}}$  of the composition of bitwise XOR and a bit rotation are investigated where the differences are expressed using addition modulo  $2^n$ . This composition is widely used in ARX constructions consisting of additions modulo  $2^n$ , bit rotations and bitwise XORs. Differential cryptanalysis of such primitives may involve maximums of  $\text{adp}^{\text{XR}}$ , where some of its input or output differences are fixed. Although there is an efficient way to calculate this probability, many its properties are still unknown. In this work we find maximums of  $\text{adp}^{\text{XR}}$ , where the rotation is one bit left/right and one of its input differences is fixed. Some symmetries of  $\text{adp}^{\text{XR}}$  are obtained as well.

**Keywords:** ARX, differential cryptanalysis, XOR, bit rotation, modular addition.

## 1 Introduction

ARX is one of the modern architectures for symmetric cryptography primitives that uses only three operations: addition modulo  $2^n$  (Addition,  $\boxplus$ ), circular shift (Rotation,  $\lll$ ) and bitwise addition modulo 2 (XOR,  $\oplus$ ). Examples of such schemes include block ciphers TEA [1, 2], FEAL [3], Threefish [4], Speck [5], stream ciphers Salsa20 [6] and its modification ChaCha [7], SHA-3 finalists BLAKE [8] and Skein [4], MAC algorithm Chaskey [9]. ARX constructions have many advantages: fast performance and compactness of its program implementation, resistance to timing attacks. However, it is difficult to determine if they are secure against differential cryptanalysis [10] that studies how input differences transform to the output differences.

In this work we consider differences that are expressed using addition modulo  $2^n$ . In some cases they are more appropriate than XOR differences,

especially if round keys are added modulo  $2^n$  and the number of XOR operations is much less than the number of modulo  $2^n$  operations (see, for instance, [11]). The additive differential probability  $\text{adp}^f(\alpha_1, \dots, \alpha_k \rightarrow \alpha_{k+1})$  for a function  $f : (\mathbb{Z}_2^n)^k \rightarrow \mathbb{Z}_2^n$ , where  $\alpha_1, \dots, \alpha_{k+1} \in \mathbb{Z}_2^n$ , is defined as

$$\Pr_{x_1, \dots, x_k \in \mathbb{Z}_2^n} [f(x_1 \boxplus \alpha_1, \dots, x_k \boxplus \alpha_k) = f(x_1, \dots, x_k) \boxplus \alpha_{k+1}].$$

Here  $\alpha_1, \dots, \alpha_k$  and  $\alpha_{k+1}$  are its input differences and its output difference respectively. The additive differential probability of a rotation ( $\text{adp}^{\lll}$ ) and XOR ( $\text{adp}^{\oplus}$ ) were studied in [12, 13] and [14, 15, 16]. The formula for the additive differential probability  $\text{adp}^{\text{RX}}$  for the composition  $(x \lll r) \oplus y$  was obtained in [17]. Also, it was pointed out that it is inaccurate to calculate differential probability of the composition by assuming that the inputs of the basic operations are independent (the same is true for XOR differences, see, for instance, [18]). Note that we can add  $\boxplus$  operation to a composition and express new differential probabilities from the old ones in a direct way, since the considered differences go through  $\boxplus$  with probability one. For instance, it works for the function  $((x \boxplus z) \lll r) \oplus y$  if we know differential probabilities for  $(x \lll r) \oplus y$ .

We investigate the properties of  $\text{adp}^{\text{XR}}$  for the function  $(x \oplus y) \lll r$ . They are similar to the properties of  $\text{adp}^{\text{RX}}$  since  $\text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{r} \gamma) = \text{adp}^{\text{RX}}(\gamma, \beta \xrightarrow{n-r} \alpha)$ . Although the formula for  $\text{adp}^{\text{RX}}$  has been proposed, many its properties are still unknown. This work is devoted to maximums of  $\text{adp}^{\text{XR}}$ , where the rotation is one bit left/right and one of the first two arguments is fixed. More precisely, we find  $\beta', \gamma'$  such that  $\max_{\beta, \gamma} \text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{r} \gamma) = \text{adp}^{\text{XR}}(\alpha, \beta' \xrightarrow{r} \gamma')$ , where  $r = 1$  (one bit left rotation) and  $r = n - 1$  (one bit right rotation). In addition, we obtain some symmetries of  $\text{adp}^{\text{XR}}$ . Using these symmetries, we can get distinct differentials whose probabilities are the same. It may be interesting for maximum differentials. Note that we do not analyze maximums for other argument fixations as well as for other rotations. These cases look more difficult. At the same time, considered maximums for  $r = 1$  and  $r = n - 1$  are very similar to maximums for  $\text{adp}^{\oplus}$ . It may be an additional reason not to use these rotations in cryptographic primitives.

The paper is organized as follows. Necessary definitions are given in Section 2. In Section 3 we introduce auxiliary definitions of  $\text{adp}_c^{\oplus}$  and  $\text{adp}_{a,b}^{\oplus}$  and rewrite the formula from [17] to obtain an expression for  $\text{adp}^{\text{XR}}$  in these terms (Theorem 2 and Corollary 1). Section 4 is devoted to the symmetries of  $\text{adp}^{\text{XR}}$  which turned out to be similar to the symmetries of  $\text{adp}^{\oplus}$  (Theorem 4). Section 5 describes maximums of  $\text{adp}^{\text{XR}}$ , where the rotation is one bit



left and one of the first two arguments is fixed (Theorem 6). Similar results are obtained in Section 6 for the one bit right rotation (Theorem 7).

## 2 Preliminaries

Let  $x, y \in \mathbb{Z}_2^n$  be elements of the  $n$ -dimensional vector space over the two-element field. Note that  $x = (x_0, x_1, \dots, x_{n-1})$ , i.e. its least and most significant bits are  $x_0$  and  $x_{n-1}$  respectively. Since ARX schemes mix modulo 2 and modulo  $2^n$  operations, we denote that  $x + y$ ,  $x - y$  and  $-x$  mean  $x' + y' \pmod{2^n}$ ,  $x' - y' \pmod{2^n}$  and  $-x' \pmod{2^n}$  respectively, where  $x' = x_0 + x_1 2^1 + \dots + x_{n-1} 2^{n-1}$ . In other words,  $x$  is a binary representation of the integer  $x' \in \{0, \dots, 2^n - 1\}$ . The rotation is defined as  $x \lll r = (x_{n-r}, \dots, x_{n-1}, x_0, \dots, x_{n-r-1})$ . Let

$$\bar{x} = (x_0 \oplus 1, x_1 \oplus 1, \dots, x_{n-1} \oplus 1).$$

Recall that  $\bar{x} = 2^n - 1 - x$ . We define  $x^{\oplus a}$ ,  $a \in \mathbb{Z}_2$ , in the following way:

$$x^{\oplus a} = \begin{cases} x, & \text{if } a = 0 \\ \bar{x}, & \text{if } a = 1 \end{cases}.$$

The vector  $(a, x_0, x_1, \dots, x_{n-1})$  is denoted by  $xa$ . In terms of integers,  $xa = 2x + a \pmod{2^{n+1}}$ . Also,  $x \preceq y$  if and only if  $x_i \leq y_i$  for all  $i \in \{0, \dots, n-1\}$ .

We consider differences that are expressed using addition modulo  $2^n$ . The additive differential probability  $\text{adp}^f$  for  $f : (\mathbb{Z}_2^n)^k \rightarrow \mathbb{Z}_2^n$  is defined as follows:

$$\text{adp}^f(\alpha_1, \dots, \alpha_k \rightarrow \alpha_{k+1}) = 2^{-kn} \#\{x_1, \dots, x_k \in \mathbb{Z}_2^n : f(x_1 + \alpha_1, \dots, x_k + \alpha_k) = f(x_1, \dots, x_k) + \alpha_{k+1}\},$$

where  $\alpha_1, \dots, \alpha_k \in \mathbb{Z}_2^n$  and  $\alpha_{k+1} \in \mathbb{Z}_2^n$  are called input differences and an output difference respectively.

In this work we consider  $\text{adp}^{\oplus}(\alpha, \beta \rightarrow \gamma)$  for the function  $x \oplus y$  (see [14, 16]),  $\text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{r} \gamma)$  for the function  $(x \oplus y) \lll r$  and  $\text{adp}^{\text{RX}}(\alpha, \beta \xrightarrow{r} \gamma)$  for the function  $(x \lll r) \oplus y$  (see [17]).

Let  $e_0, \dots, e_7$  be standard basis vectors of  $\mathbb{Q}^8$ ,  $\mathbb{Q}$  is the field of rationals. They are vector-columns. There is a matrix approach for calculating  $\text{adp}^{\oplus}$ .

**Theorem 1** (Lipmaa et al. [14], 2004). *Let  $L = (1, 1, 1, 1, 1, 1, 1, 1)$ ,  $A_0, \dots, A_7$  be  $8 \times 8$  matrices, where*

$$A_0 = \frac{1}{4} \begin{pmatrix} 4 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

and  $A_k = ((A_k)_{i,j}) = ((A_0)_{i\oplus k,j\oplus k})$ , where  $i, j, k \in \mathbb{Z}_2^3$ . Then

$$\text{adp}^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}^\oplus(\omega) = LA_{\omega_{n-1}}A_{\omega_{n-2}}\dots A_{\omega_0}e_0,$$

where  $\alpha, \beta, \gamma \in \mathbb{Z}_2^n$  and the differential  $(\alpha, \beta \rightarrow \gamma)$  is written as the octal word  $\omega = \omega_{n-1}\dots\omega_0$  with  $\omega_i = \omega_i(\alpha, \beta, \gamma) = 4\alpha_i + 2\beta_i + \gamma_i$ .

Note that we will use both integer and binary vector notations in the indexes of matrices and coordinates, i.e. the matrices  $A_{p_2p_1p_0}$  and  $A_{4p_2+2p_1+p_0}$  (coordinates  $v_{p_2p_1p_0}$  and  $v_{4p_2+2p_1+p_0}$  of  $v \in \mathbb{Q}^8$ ) mean the same. Also, [16, Proposition 1] provides that  $\text{adp}^\oplus$  is symmetric. We will refer to the results of [16] taking into account this fact.

### 3 A formula for $\text{adp}^{\text{XR}}$

We will represent a formula for  $\text{adp}^{\text{XR}}$  by rewriting the formula from [17]. First of all, we introduce auxiliary terms. The function  $\text{carry} : \mathbb{Z}_2^n \times \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2$  is defined in the following way:

$$\text{carry}(x, y) = 0 \text{ if and only if } x + y < 2^n \text{ as integers, } x, y \in \mathbb{Z}_2^n.$$

Let us introduce  $\text{adp}_c^\oplus$  and  $\text{adp}_{a,b}^\oplus$ :

$$\begin{aligned} \text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) = & 2^{-2n} \#\{x, y \in \mathbb{Z}_2^n : (x + \alpha) \oplus (y + \beta) = \gamma + (x \oplus y), \\ & \text{carry}(x \oplus y, \gamma) = c\}, \text{ where } c \in \mathbb{Z}_2, \end{aligned}$$

$$\begin{aligned} \text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow \gamma) = & 2^{-2n} \#\{x, y \in \mathbb{Z}_2^n : (x + \alpha) \oplus (y + \beta) = \gamma + (x \oplus y), \\ & \text{carry}(x, \alpha) = a, \text{carry}(y, \beta) = b\}, \text{ where } a, b \in \mathbb{Z}_2. \end{aligned}$$

Their values for  $\alpha, \beta, \gamma \in \mathbb{Z}_2$  are given in Table 1.

**Proposition 1.** *We can calculate  $\text{adp}_c^\oplus$  and  $\text{adp}_{a,b}^\oplus$  in the following way:*

$$\begin{aligned} \text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow \gamma) &= L_{a,b}A_{\omega_{n-1}}\dots A_{\omega_0}e_0, \\ \text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) &= L_cA_{\omega_{n-1}}\dots A_{\omega_0}e_0, \end{aligned}$$

where

$$\begin{aligned} w_i &= 4\alpha_i + 2\beta_i + \gamma_i, \\ L_0 &= (1, 0, 1, 0, 1, 0, 1, 0), \quad L_1 = (0, 1, 0, 1, 0, 1, 0, 1) \\ L_{0,0} &= (1, 1, 0, 0, 0, 0, 0, 0), \quad L_{0,1} = (0, 0, 1, 1, 0, 0, 0, 0) \\ L_{1,0} &= (0, 0, 0, 0, 1, 1, 0, 0), \quad L_{1,1} = (0, 0, 0, 0, 0, 0, 1, 1) \end{aligned}$$

Table 1:  $\text{adp}_{a,b}^{\oplus}(\alpha, \beta \rightarrow \gamma)$  and  $\text{adp}_c^{\oplus}(\alpha, \beta \rightarrow \gamma)$ ,  $\alpha, \beta, \gamma \in \mathbb{Z}_2$ 

$\alpha\beta\gamma$	$\text{adp}_{0,0}^{\oplus}$	$\text{adp}_{0,1}^{\oplus}$	$\text{adp}_{1,0}^{\oplus}$	$\text{adp}_{1,1}^{\oplus}$	$\text{adp}_0^{\oplus}$	$\text{adp}_1^{\oplus}$
000	1	0	0	0	1	0
011	1/2	1/2	0	0	1/2	1/2
101	1/2	0	1/2	0	1/2	1/2
110	1/4	1/4	1/4	1/4	1	0
001	0	0	0	0	0	0
010	0	0	0	0	0	0
100	0	0	0	0	0	0
111	0	0	0	0	0	0

*Proof.* By Theorem 1, we know that  $\text{adp}^{\oplus}(\alpha, \beta \rightarrow \gamma) = Lv$ , where  $v = A_{\omega_{n-1}} \dots A_{\omega_0} e_0$  and  $L = (1, 1, 1, 1, 1, 1, 1, 1)$ . It remains to take into account the conditions for carries. S-function approach [15, §4] for calculating  $\text{adp}^{\oplus}$  gives us that the coordinate  $v_{4a+2b+c}$  ( $v_{abc}$  in binary form) equals to  $2^{-2n} \#\{x, y \in \mathbb{Z}_2^n : (x + \alpha) \oplus (y + \beta) = \gamma + (x \oplus y) \text{ where carry bits of } x + \alpha, y + \beta \text{ are } a \text{ and } b \text{ respectively, borrow bit of } (x + \alpha) \oplus (y + \beta) - (x \oplus y) \text{ is } c\}$ . Note that this borrow bit equals to the carry bit of  $(x \oplus y) + \gamma$ . Thus,  $\text{carry}(x, \alpha) = a$ ,  $\text{carry}(y, \beta) = b$  and  $\text{carry}(x \oplus y, \gamma) = c$ .

Next, we need to sum several coordinates of  $v$ . Let us calculate  $\text{adp}_{1,0}^{\oplus}$ , i.e.  $\text{carry}(x, \alpha) = 1$ ,  $\text{carry}(y, \beta) = 0$ ,  $\text{carry}(x \oplus y, \gamma) \in \{0, 1\}$ . We should take the sum  $v_4 + v_5$  (in other words,  $v_{100} + v_{101}$ ). Thus,  $\text{adp}_{1,0}^{\oplus} = L_{1,0}v$ .

It is not difficult to check all other formulas.  $\square$

Next, we prove the following lemma similar to [16, Lemma 4].

**Lemma 1.** *Let  $a, b, c \in \mathbb{Z}_2$ , and  $k, \omega_0, \dots, \omega_n \in \mathbb{Z}_2^3$ ,  $n \geq 0$ . Then*

- $L_c A_{\omega_n} \dots A_{\omega_0} e_k = L_{c \oplus k_0} A_{\omega_n \oplus k} \dots A_{\omega_0 \oplus k} e_0$ ,
- $L_{a,b} A_{\omega_n} \dots A_{\omega_0} e_k = L_{a \oplus k_2, b \oplus k_1} A_{\omega_n \oplus k} \dots A_{\omega_0 \oplus k} e_0$ .

*Proof.* Indeed, it is straightforward that  $A_{t \oplus k} = T_k A_t T_k$ ,  $e_k = T_k e_0$ ,  $L_{c \oplus k_0} = L_c T_k$  and  $L_{a \oplus k_2, b \oplus k_1} = L_{a,b} T_k$ , where  $t \in \mathbb{Z}_2^3$  and  $T_k$  is the  $8 \times 8$  involution matrix that swaps the  $i$  and  $i \oplus k$  coordinates,  $i \in \mathbb{Z}_2^3$ .  $\square$

Now we rewrite the formula from [17] in terms of  $\text{adp}_c^{\oplus}$  and  $\text{adp}_{a,b}^{\oplus}$ . Note that in the denotation  $\alpha'' = (\alpha, \alpha')$ , where  $\alpha \in \mathbb{Z}_2^n$ ,  $\alpha' \in \mathbb{Z}_2^m$ ,  $\alpha$  contains the least significant bits of  $\alpha''$ , i.e.  $\alpha'' = (\alpha_0, \dots, \alpha_{n-1}, \alpha'_0, \dots, \alpha'_{m-1})$ .

**Theorem 2.** *Let  $\alpha, \beta, \gamma \in \mathbb{Z}_2^{n-r}$ ,  $\alpha', \beta', \gamma' \in \mathbb{Z}_2^r$ . Then*

$$\begin{aligned} \text{adp}^{\text{XR}}((\alpha, \alpha'), (\beta, \beta') \xrightarrow{r} (\gamma', \gamma)) \\ = \sum_{a,b,c \in \mathbb{Z}_2} \text{adp}_{a,b}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus c}) \text{adp}_c^{\oplus}(\alpha'^{\oplus a}, \beta'^{\oplus b} \rightarrow \gamma'). \end{aligned}$$

*Proof.* Let  $x = (\alpha, \alpha'), y = (\beta, \beta'), z = (\gamma', \gamma)$ . According to the formula for  $\text{adp}^{\text{RX}}$  that was found in [17, Theorem 1] ( $\text{adp}^{\text{ARX}}$  without changing  $\Delta\gamma$ ),

$$\begin{aligned} \text{adp}^{\text{XR}}(x, y \xrightarrow{r} z) &= \text{adp}^{\text{RX}}(z, y \xrightarrow{n-r} x) \\ &\stackrel{[17]}{=} 2^{-2n} \sum_{j \in \{0,2,4,6\}} F_j B_{s_{n-1}} \dots B_{s_r} R B_{s_{r-1}} \dots B_{s_0} C_j, \end{aligned}$$

where  $s_i = z_i y_{i+n-r} x_{i+n-r}$  ( $4z_i + 2y_{i+n-r} + x_{i+n-r}$  as an integer),  $F_0 = L_{0,0}$ ,  $F_2 = L_{0,1}$ ,  $F_4 = L_{1,0}$ ,  $F_6 = L_{1,1}$  and  $C_j = e_j$ , see [17, §5.4]. The matrices  $B_{000}, \dots, B_{111}$ ,  $R$  are given in [17, Appendix A] (they are denoted by  $A$  there). It can be seen that  $R = e_4 L_0 + e_5 L_1$  and  $B_{abc} = 4A_{\bar{c}ba}$  where  $a, b, c \in \mathbb{Z}_2$ . Let  $v_i = \overline{x_{i+n-r} y_{i+n-r} z_i}$ ,  $i \in \{0, \dots, n-1\}$ . Using Lemma 1, we get that

$$\begin{aligned} \text{adp}^{\text{XR}}((\alpha, \alpha'), (\beta, \beta') \xrightarrow{r} (\gamma', \gamma)) \\ = \sum_{j \in \{0,2,4,6\}} F_j A_{v_{n-1}} \dots A_{v_r} (e_4 L_0 + e_5 L_1) A_{v_{r-1}} \dots A_{v_0} e_j \\ = \sum_{j \in \{0,2,4,6\}} L_{0,0} A_{v_{n-1} \oplus j} \dots A_{v_r \oplus j} (e_{4 \oplus j} L_0 + e_{5 \oplus j} L_1) A_{v_{r-1} \oplus j} \dots A_{v_0 \oplus j} e_0 \\ = \sum_{j \in \{0,2,4,6\}} L_{0,0} A_{v_{n-1} \oplus j} \dots A_{v_r \oplus j} e_{4 \oplus j} \times L_0 A_{v_{r-1} \oplus j} \dots A_{v_0 \oplus j} e_0 \\ \quad + \sum_{j \in \{0,2,4,6\}} L_{0,0} A_{v_{n-1} \oplus j} \dots A_{v_r \oplus j} e_{5 \oplus j} \times L_1 A_{v_{r-1} \oplus j} \dots A_{v_0 \oplus j} e_0 \\ = \sum_{a,b \in \mathbb{Z}_2} L_{0,0} A_{v_{n-1} \oplus ab0} \dots A_{v_r \oplus ab0} e_{100 \oplus ab0} \times L_0 A_{v_{r-1} \oplus ab0} \dots A_{v_0 \oplus ab0} e_0 \\ \quad + \sum_{a,b \in \mathbb{Z}_2} L_{0,0} A_{v_{n-1} \oplus ab0} \dots A_{v_r \oplus ab0} e_{101 \oplus ab0} \times L_1 A_{v_{r-1} \oplus ab0} \dots A_{v_0 \oplus ab0} e_0 \\ = \sum_{a,b \in \mathbb{Z}_2} L_{\bar{a},b} A_{v_{n-1} \oplus 100} \dots A_{v_r \oplus 100} e_0 \times L_0 A_{v_{r-1} \oplus ab0} \dots A_{v_0 \oplus ab0} e_0 \\ \quad + \sum_{a,b \in \mathbb{Z}_2} L_{\bar{a},b} A_{v_{n-1} \oplus 101} \dots A_{v_r \oplus 101} e_0 \times L_1 A_{v_{r-1} \oplus ab0} \dots A_{v_0 \oplus ab0} e_0 \\ = \sum_{a,b,c \in \mathbb{Z}_2} L_{\bar{a},b} A_{v_{n-1} \oplus 10c} \dots A_{v_r \oplus 10c} e_0 \times L_c A_{v_{r-1} \oplus ab0} \dots A_{v_0 \oplus ab0} e_0 \\ = \sum_{a,b,c \in \mathbb{Z}_2} L_{a,b} A_{v_{n-1} \oplus 10c} \dots A_{v_r \oplus 10c} e_0 \times L_c A_{v_{r-1} \oplus \bar{a}b0} \dots A_{v_0 \oplus \bar{a}b0} e_0 \end{aligned}$$

Finally,  $L_{a,b}A_{v_{n-1} \oplus 10c} \dots A_{v_r \oplus 10c}e_0 = \text{adp}_{a,b}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus c})$  and  $L_cA_{v_{r-1} \oplus \bar{a}b_0} \dots A_{v_0 \oplus \bar{a}b_0}e_0 = \text{adp}_c^{\oplus}(\alpha'^{\oplus a}, \beta'^{\oplus b} \rightarrow \gamma')$  by Proposition 1.  $\square$

**Corollary 1.** *Let  $\alpha, \beta, \gamma \in \mathbb{Z}_2^{n-r}$ ,  $\alpha', \beta', \gamma' \in \mathbb{Z}_2^r$ , and  $a = \alpha_0 \oplus \beta_0 \oplus \gamma_0$ ,  $a' = \alpha'_0 \oplus \beta'_0 \oplus \gamma'_0$ , i. e.  $a, a' \in \mathbb{Z}_2$ . Then*

$$\begin{aligned} \text{adp}^{\text{XR}}((\alpha, \alpha'), (\beta, \beta') \xrightarrow{r} (\gamma', \gamma)) &= \text{adp}_{a',0}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus a}) \text{adp}_a^{\oplus}(\alpha'^{\oplus a'}, \beta' \rightarrow \gamma') \\ &\quad + \text{adp}_{a',1}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus a}) \text{adp}_a^{\oplus}(\bar{\alpha}'^{\oplus a'}, \bar{\beta}' \rightarrow \gamma'). \end{aligned}$$

*Proof.* By Theorem 2, the following holds

$$\begin{aligned} \text{adp}^{\text{XR}}((\alpha, \alpha'), (\beta, \beta') \xrightarrow{r} (\gamma', \gamma)) &= \sum_{p,q,u \in \mathbb{Z}_2} \text{adp}_{p,q}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus u}) \text{adp}_u^{\oplus}(\alpha'^{\oplus p}, \beta'^{\oplus q} \rightarrow \gamma'). \end{aligned}$$

If  $\alpha_0 \oplus \beta_0 \oplus \gamma_0 = 1$ , then  $\text{adp}^{\oplus}(\alpha, \beta \rightarrow \gamma) = 0$ , see, for instance, [14]. It means that  $\text{adp}_{p,q}^{\oplus}(\alpha, \beta \rightarrow \gamma) = \text{adp}_u^{\oplus}(\alpha, \beta \rightarrow \gamma) = 0$ . Hence, we can exclude  $u \neq \alpha_0 \oplus \beta_0 \oplus \gamma_0 = a$ . Also, we can consider only

$$\alpha'_0 \oplus p \oplus \beta'_0 \oplus q \oplus \gamma'_0 = 0 \Rightarrow \begin{cases} p = \alpha'_0 \oplus \beta'_0 \oplus \gamma'_0 = a', & \text{if } q = 0, \\ p = \overline{\alpha'_0 \oplus \beta'_0 \oplus \gamma'_0} = \bar{a}', & \text{if } q = 1. \end{cases}$$

This completes the proof.  $\square$

## 4 Symmetries of $\text{adp}^{\text{XR}}$

In this section we prove some symmetries of  $\text{adp}^{\text{XR}}$ . We start with the following properties of the function carry.

**Proposition 2.** *Let  $x, y, \alpha \in \mathbb{Z}_2^n$ ,  $\alpha \neq 0$ . Then the following holds:*

1.  $\text{carry}(x, y) = \text{carry}(y, x)$ ,
2.  $\text{carry}(x - \alpha, \alpha) = \text{carry}(x, -\alpha) \oplus 1$ ,
3.  $\text{carry}(\bar{x}, \alpha) = \text{carry}(x, -\alpha) \oplus 1$ .

*Proof.* The first point is straightforward. To be precise, we will use  $+$ ,  $-$  and  $\boxplus$ ,  $\boxminus$  for the integer and modulo operations respectively. Also, it can be seen that  $2^n - \alpha = \boxminus \alpha$  as integers since  $\alpha \neq 0$ .

Let us prove the second point:  $\text{carry}(x \boxminus \alpha, \alpha) = 0 \iff (x \boxminus \alpha) + \alpha < 2^n \iff x \geq \alpha$  as integers. Indeed, if  $x \geq \alpha$ ,  $(x \boxminus \alpha) + \alpha = x - \alpha + \alpha = x < 2^n$ . Otherwise,  $(x \boxminus \alpha) + \alpha = 2^n + x - \alpha + \alpha = 2^n + x \geq 2^n$ .

At the same time,  $x \geq \alpha \iff x \geq 2^n - (2^n - \alpha) \iff x + (2^n - \alpha) \geq 2^n \iff x + (\Xi\alpha) \geq 2^n \iff \text{carry}(x, \Xi\alpha) = 1$ .

Finally,  $\text{carry}(\bar{x}, \alpha) = 0 \iff 2^n - 1 - x + \alpha < 2^n \iff 2^n - x + \alpha \leq 2^n \iff x + (2^n - \alpha) \geq 2^n \iff x + (\Xi\alpha) \geq 2^n \iff \text{carry}(x, \Xi\alpha) = 1$ .  $\square$

Next, we need to prove symmetries of  $\text{adp}_c^\oplus$  and  $\text{adp}_{a,b}^\oplus$ .

**Theorem 3.** *Let  $\alpha, \beta, \gamma \in \mathbb{Z}_2^n$ ,  $c \in \mathbb{Z}_2$ . Then*

1.  $\text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_c^\oplus(-\alpha, \beta \rightarrow \gamma) = \text{adp}_c^\oplus(\alpha, -\beta \rightarrow \gamma)$ ,
2.  $\text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_{c\oplus 1}^\oplus(\alpha, \beta \rightarrow -\gamma)$  for  $\gamma \neq 0$ ,
3.  $\text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_c^\oplus(\beta, \alpha \rightarrow \gamma) = \text{adp}_c^\oplus(\alpha + 2^{n-1}, \beta + 2^{n-1} \rightarrow \gamma)$ ,
4.  $\text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow -\gamma)$ ,
5.  $\text{adp}_{a,b}^\oplus(-\alpha, \beta \rightarrow \gamma) = \text{adp}_{a\oplus 1, b}^\oplus(\alpha, \beta \rightarrow \gamma)$  for  $\alpha \neq 0$ ,
6.  $\text{adp}_{a,b}^\oplus(\alpha, -\beta \rightarrow \gamma) = \text{adp}_{a, b\oplus 1}^\oplus(\alpha, \beta \rightarrow \gamma)$  for  $\beta \neq 0$ .

*Proof.* First of all, we refer to the symmetries of  $\text{adp}^\oplus$ . It can be seen (we may refer to the proofs of [16, Propositions 1, 3]) that the condition for  $x, y \in \mathbb{Z}_2^n$  to satisfy  $(x + \alpha) \oplus (y + \beta) = \gamma + (x \oplus y)$  is equivalent to any of the following:

1.  $x' = \overline{x + \alpha}$ ,  $y' = \overline{y + \beta}$  satisfy  $(x' + \alpha) \oplus (y' + \beta) = -\gamma + (x' \oplus y')$ ,
2.  $x' = \overline{x + \alpha}$ ,  $y' = y + \beta$  satisfy  $(x' + \alpha) \oplus (y' - \beta) = \gamma + (x' \oplus y')$ .

Let us start with the first two point. Indeed,

$$\text{adp}_c^\oplus(\alpha, -\beta \rightarrow \gamma) \stackrel{2}{=} \{x' = \overline{x + \alpha}, y' = y + \beta : (x' + \alpha) \oplus (y' - \beta) = \gamma + (x' \oplus y'), \text{carry}(x' \oplus y', \gamma) = c\}.$$

Taking into account the points of Proposition 2 (p.2 and p.3), we have that

$$\begin{aligned} \text{carry}(x' \oplus y', \gamma) &= \text{carry}(\overline{(x + \alpha) \oplus (y + \beta)}, \gamma) = \text{carry}(\overline{(x \oplus y) + \gamma}, \gamma) \\ &= \text{carry}(\overline{x \oplus y} - \gamma, \gamma) \stackrel{\text{p.2}}{=} \text{carry}(\overline{x \oplus y}, -\gamma) \oplus 1 \stackrel{\text{p.3}}{=} \text{carry}(x \oplus y, \gamma). \end{aligned}$$

Thus,  $\text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_c^\oplus(\alpha, -\beta \rightarrow \gamma)$ . The same is true for  $-\alpha$  due to their symmetry. Next,

$$\text{adp}_c^\oplus(\alpha, \beta \rightarrow -\gamma) \stackrel{1}{=} \{x' = \overline{x + \alpha}, y' = \overline{y + \beta} : (x' + \alpha) \oplus (y' + \beta) = -\gamma + (x' \oplus y'), \text{carry}(x' \oplus y', -\gamma) = c\}.$$

By the points of Proposition 2, we can see that

$$\begin{aligned} \text{carry}(x' \oplus y', -\gamma) &= \text{carry}((x + \alpha) \oplus (y + \beta), -\gamma) \\ &= \text{carry}((x \oplus y) + \gamma, -\gamma) \stackrel{\text{p.2}}{=} \text{carry}(x \oplus y, \gamma) \oplus 1. \end{aligned}$$

Thus,  $\text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_{c \oplus 1}^\oplus(\alpha, \beta \rightarrow -\gamma)$ .

Also, it is not difficult to check by definition that  $\text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_c^\oplus(\beta, \alpha \rightarrow \gamma) = \text{adp}_c^\oplus(\alpha + 2^{n-1}, \beta + 2^{n-1} \rightarrow \gamma)$ .

The proof of the rest points is very similar to the proof of the previous ones. We use the same  $x', y'$ . Let us consider the case of  $-\beta$ .

$$\begin{aligned} \text{adp}_{a,b}^\oplus(\alpha, -\beta \rightarrow \gamma) &\stackrel{2}{=} \{x' = \overline{x + \alpha}, y' = y + \beta : (x' + \alpha) \oplus (y' - \beta) \\ &= \gamma + (x' \oplus y'), \text{carry}(x', \alpha) = a, \text{carry}(y', -\beta) = b\}. \end{aligned}$$

Taking into account the points of Proposition 2, we have that

$$\begin{aligned} \text{carry}(y', -\beta) &= \text{carry}(y + \beta, -\beta) \stackrel{\text{p.2}}{=} \text{carry}(y, \beta) \oplus 1, \text{ also,} \\ \text{carry}(x', \alpha) &= \text{carry}(\overline{x + \alpha}, \alpha) = \text{carry}(\overline{x} - \alpha, \alpha) \\ &\stackrel{\text{p.2}}{=} \text{carry}(\overline{x}, -\alpha) \oplus 1 \stackrel{\text{p.3}}{=} \text{carry}(x, \alpha). \end{aligned} \quad (1)$$

Thus,  $\text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_{a,b \oplus 1}^\oplus(\alpha, -\beta \rightarrow \gamma)$ . The same is true for  $-\alpha$  due to their symmetry. Next, let us consider  $-\gamma$ .

$$\begin{aligned} \text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow -\gamma) &\stackrel{1}{=} \{x' = \overline{x + \alpha}, y' = \overline{y + \beta} : (x' + \alpha) \oplus (y' + \beta) \\ &= -\gamma + (x' \oplus y'), \text{carry}(x', \alpha) = a, \text{carry}(y', \beta) = b\}. \end{aligned}$$

The equality (1) gives us exactly what we need, the condition for  $\beta$  is the same for this case. Thus,  $\text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow -\gamma)$ .  $\square$

Now we obtain some symmetries of  $\text{adp}^{\text{XR}}$ .

**Theorem 4.** *The values of  $\text{adp}^{\text{XR}}$  satisfy the following properties:*

- $\text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{r} \gamma) = \text{adp}^{\text{XR}}(\beta, \alpha \xrightarrow{r} \gamma)$ ,
- $\text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{r} \gamma) = \text{adp}^{\text{XR}}(\alpha + 2^{n-1}, \beta + 2^{n-1} \xrightarrow{r} \gamma)$
- $\text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{r} \gamma) = \text{adp}^{\text{XR}}(\pm\alpha, \pm\beta \xrightarrow{r} \pm\gamma)$ , where  $\pm\alpha$  means that we can substitute either  $\alpha$  or  $-\alpha$ ,

*Proof.* The first two points are straightforward. Next, we do not consider  $\beta$  since  $\alpha$  and  $\beta$  are symmetric. Let  $\alpha, \beta, \gamma \in \mathbb{Z}_2^{n-r}$ ,  $\alpha', \beta', \gamma' \in \mathbb{Z}_2^r$ . According to Theorem 2,  $\text{adp}^{\text{XR}}((\alpha, \alpha'), (\beta, \beta') \xrightarrow{r} (\gamma', \gamma))$  is equal to

$$\sum_{a,b,c \in \mathbb{Z}_2} \text{adp}_{a,b}^\oplus(\alpha, \beta \rightarrow \gamma^{\oplus c}) \text{adp}_c^\oplus(\alpha'^{\oplus a}, \beta'^{\oplus b} \rightarrow \gamma'). \quad (2)$$

**Case 1.**  $\alpha \neq 0$  (resp.  $\gamma' \neq 0$ ). Thus,  $-(\alpha, \alpha') = (-\alpha, \overline{\alpha'})$ . Let us substitute  $(-\alpha, \overline{\alpha'})$  to (2):  $\text{adp}_c^\oplus(\overline{\alpha'}^{\oplus a}, \beta'^{\oplus b} \rightarrow \gamma') = \text{adp}_c^\oplus(\alpha'^{\oplus a \oplus 1}, \beta'^{\oplus b} \rightarrow \gamma')$ . Also,  $\text{adp}_{a,b}^\oplus(-\alpha, \beta \rightarrow \gamma^{\oplus c}) = \text{adp}_{a \oplus 1, b}^\oplus(\alpha, \beta \rightarrow \gamma^{\oplus c})$  by Theorem 3. But (2) does not change if we replace  $a$  by  $a \oplus 1$ . Similar reasons work for  $-(\gamma', \gamma) = (-\gamma', \overline{\gamma})$ .

**Case 2.**  $\alpha = 0$  (resp.  $\gamma' = 0$ ). Thus,  $-(0, \alpha') = (0, -\alpha')$ . Substituting  $(0, -\alpha')$  to (2),  $\text{adp}_{1,b}^\oplus(0, \beta \rightarrow \gamma^{\oplus c}) = 0$  by definition. Hence, we can only consider  $a = 0$ . But in this case  $\text{adp}_c^\oplus$  does not contain  $\overline{\alpha'}$ . By Theorem 3,  $\text{adp}_c^\oplus(-\alpha', \beta'^{\oplus b} \rightarrow \gamma') = \text{adp}_c^\oplus(\alpha', \beta'^{\oplus b} \rightarrow \gamma')$ . It means that (2) does not change. Similar reasons work for  $-(0, \gamma) = (0, -\gamma)$ .  $\square$

## 5 Maximums of $\text{adp}^{\text{XR}}$ for $r = 1$

In this section we consider maximums of  $\text{adp}^{\text{XR}}$  for  $r = 1$  (the rotation is one bit left), where one of the first two arguments is fixed. First of all, we prove the recurrence formulas for  $\text{adp}_c^\oplus$  and  $\text{adp}_{a,b}^\oplus$  which are similar to ones for  $\text{adp}^\oplus$  obtained in [16, Theorem 3].

**Theorem 5.** *Let  $\alpha, \beta, \gamma \in \mathbb{Z}_2^n, p \in \mathbb{Z}_2^3, a, b, c \in \mathbb{Z}_2$  and  $\text{wt}(p)$  be even. Then*

$$\begin{aligned} \text{adp}_c^\oplus(\alpha p_2, \beta p_1 \rightarrow \gamma p_0) &= \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} \text{adp}_{c \oplus q_0}^\oplus(\alpha^{\oplus q_2}, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_0}), \\ \text{adp}_{a,b}^\oplus(\alpha p_2, \beta p_1 \rightarrow \gamma p_0) &= \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} \text{adp}_{a \oplus q_2, b \oplus q_1}^\oplus(\alpha^{\oplus q_2}, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_0}). \end{aligned}$$

If  $\text{wt}(p)$  is odd,  $\text{adp}_c^\oplus(\alpha p_2, \beta p_1 \rightarrow \gamma p_0) = \text{adp}_{a,b}^\oplus(\alpha p_2, \beta p_1 \rightarrow \gamma p_0) = 0$ .

*Proof.* By Proposition 1,  $\text{adp}_c^\oplus(\alpha p_2, \beta p_1 \rightarrow \gamma p_0) = L_c A_{\omega_{n-1}} \dots A_{\omega_0} A_{p_2 p_1 p_0} e_0$ .

First of all,  $\text{adp}^\oplus(\alpha p_2, \beta p_1 \rightarrow \gamma p_0) = 0$  if  $\text{wt}(p)$  is odd. Next, let  $\text{wt}(p)$  be even. It can be seen that  $A_{p_2 p_1 p_0} e_0 = \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} e_q$ . Therefore,

$$\begin{aligned} \text{adp}_c^\oplus(\alpha p_2, \beta p_1 \rightarrow \gamma p_0) &= \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} L_c A_{\omega_{n-1}} \dots A_{\omega_0} e_q \\ &\stackrel{\text{Lemma 1}}{=} \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} L_{c \oplus q_0} A_{\omega_{n-1} \oplus q} \dots A_{\omega_0 \oplus q} e_0 \\ &= \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} L_{c \oplus q_0} A_{\alpha_{n-1} \beta_{n-1} \gamma_{n-1} \oplus q_2 q_1 q_0} \dots A_{\alpha_0 \beta_0 \gamma_0 \oplus q_2 q_1 q_0} e_0 \end{aligned}$$



$$= \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} \text{adp}_{c \oplus q_0}^{\oplus}(\alpha^{\oplus q_2}, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_0}).$$

The recurrence formula for  $\text{adp}_{a,b}^{\oplus}$  can be proven in the same way.  $\square$

Using these formulas, we prove the following lemma.

**Lemma 2.** *Let  $\alpha, \beta, \gamma \in \mathbb{Z}_2^n$ ,  $a \in \mathbb{Z}_2$ . Then the following holds:*

$$\begin{aligned} \text{adp}_{a,0}^{\oplus}(\alpha, \beta \rightarrow \gamma) + \text{adp}_{\bar{a},1}^{\oplus}(\alpha, \beta \rightarrow \gamma) \\ \leq \text{adp}_{0,0}^{\oplus}(\alpha, \alpha \rightarrow 0) + \text{adp}_{1,1}^{\oplus}(\alpha, \alpha \rightarrow 0). \end{aligned}$$

*Proof.* Let us prove the statement by induction. The base of the induction  $n = 1$  directly follows from Table 1. Suppose that it holds for all  $\alpha, \beta, \gamma \in \mathbb{Z}_2^n$ ,  $a \in \mathbb{Z}_2$ . We prove that it is true for  $\alpha', \beta', \gamma' \in \mathbb{Z}_2^{n+1}$ ,  $a' \in \mathbb{Z}_2$ . Let  $p = (\alpha_0, \beta_0, \gamma_0)$  and  $\alpha' = \alpha p_1, \beta' = \beta p_2, \gamma' = \gamma p_3$ . We suppose that  $\text{wt}(p)$  is even, since in other cases  $\text{adp}^{\oplus}(\alpha, \beta \rightarrow \gamma) = 0$  and the inequality holds. According to Theorem 5,

$$\begin{aligned} \text{adp}_{a,0}^{\oplus}(\alpha', \beta' \rightarrow \gamma') + \text{adp}_{\bar{a},1}^{\oplus}(\alpha', \beta' \rightarrow \gamma') \\ = \frac{1}{2^{\text{wt}(p)}} \sum_{q \in \mathbb{Z}_2^3, q \preceq p} (\text{adp}_{a \oplus q_0, q_1}^{\oplus}(\alpha^{\oplus q_0}, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_2}) \\ + \text{adp}_{\bar{a} \oplus q_0, \bar{q}_1}^{\oplus}(\alpha^{\oplus q_0}, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_2})). \end{aligned} \quad (3)$$

**Case 1.**  $p_0 = 0$ . In this case  $\text{adp}_{0,0}^{\oplus}(\alpha', \alpha' \rightarrow 0) + \text{adp}_{1,1}^{\oplus}(\alpha', \alpha' \rightarrow 0) = \text{adp}_{0,0}^{\oplus}(\alpha, \alpha \rightarrow 0) + \text{adp}_{1,1}^{\oplus}(\alpha, \alpha \rightarrow 0)$ . If  $(p_1, p_2) = 0$ , the equality (3) and the induction hypothesis prove the statement. Let  $(p_1, p_2) = (1, 1)$ . Then,

$$\begin{aligned} \text{adp}_{a,0}^{\oplus}(\alpha', \beta' \rightarrow \gamma') + \text{adp}_{\bar{a},1}^{\oplus}(\alpha', \beta' \rightarrow \gamma') \\ = \frac{1}{4} \sum_{(q_1, q_2) \in \mathbb{Z}_2^2} (\text{adp}_{a, q_1}^{\oplus}(\alpha, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_2}) + \text{adp}_{\bar{a}, \bar{q}_1}^{\oplus}(\alpha, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_2})). \end{aligned}$$

At the same time,  $\text{adp}_{a, q_1}^{\oplus}(\alpha, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_2}) + \text{adp}_{\bar{a}, \bar{q}_1}^{\oplus}(\alpha, \beta^{\oplus q_1} \rightarrow \gamma^{\oplus q_2}) \leq \text{adp}_{0,0}^{\oplus}(\alpha, \alpha \rightarrow 0) + \text{adp}_{1,1}^{\oplus}(\alpha, \alpha \rightarrow 0)$  by the induction hypothesis.

**Case 2.**  $p_0 = 1$ . First of all, we note that the equality (3) consists of 4 terms. Moreover, two of them are zero, since the last bits of their arguments are of odd weight. Next,

$$\begin{aligned} \text{adp}_{0,0}^{\oplus}(\alpha', \alpha' \rightarrow 0) + \text{adp}_{1,1}^{\oplus}(\alpha', \alpha' \rightarrow 0) \\ = \frac{1}{4} (\text{adp}_{0,0}^{\oplus}(\alpha, \alpha \rightarrow 0) + \text{adp}_{1,1}^{\oplus}(\alpha, \alpha \rightarrow 0)) \end{aligned} \quad (4)$$

$$= \frac{1}{4}(\text{adp}_{1,1}^{\oplus}(\bar{\alpha}, \bar{\alpha} \rightarrow 0) + \text{adp}_{0,0}^{\oplus}(\bar{\alpha}, \bar{\alpha} \rightarrow 0)). \quad (5)$$

Let  $(p_1, p_2) = (1, 0)$ . According to the equality (3),

$$\begin{aligned} & \text{adp}_{a,0}^{\oplus}(\alpha', \beta' \rightarrow \gamma') + \text{adp}_{\bar{a},1}^{\oplus}(\alpha', \beta' \rightarrow \gamma') \\ &= \frac{1}{4} \sum_{q \in \mathbb{Z}_2} (\text{adp}_{a,q}^{\oplus}(\alpha, \beta^{\oplus q} \rightarrow \gamma) + \text{adp}_{\bar{a},q}^{\oplus}(\alpha, \beta^{\oplus q} \rightarrow \gamma)) \end{aligned} \quad (6)$$

$$+ \frac{1}{4} \sum_{q \in \mathbb{Z}_2} (\text{adp}_{\bar{a},q}^{\oplus}(\bar{\alpha}, \beta^{\oplus q} \rightarrow \gamma) + \text{adp}_{a,q}^{\oplus}(\bar{\alpha}, \beta^{\oplus q} \rightarrow \gamma)). \quad (7)$$

Note that in both (6) and (7) a term for  $q = 0$  or for  $q = 1$  equals to zero. Therefore, the induction hypothesis provides that (6)  $\leq$  (4) and (7)  $\leq$  (5). The case of  $(p_1, p_2) = (0, 1)$  is the same.  $\square$

Finally, the next theorem describes how to find  $\max_{\beta, \gamma} \text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{1} \gamma)$ .

**Theorem 6.** *Let us fix the first argument of  $\text{adp}^{\text{XR}}$ . Then*

$$\max_{\beta, \gamma \in \mathbb{Z}_2^n} \text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{1} \gamma) = \text{adp}^{\text{XR}}(\alpha, \alpha \xrightarrow{1} 0), \text{ where } \alpha \in \mathbb{Z}_2^n.$$

*Proof.* Let  $\alpha, \beta, \gamma \in \mathbb{Z}_2^{n-r}$ ,  $\alpha', \beta', \gamma' \in \mathbb{Z}_2^r$ , and  $a = \alpha_0 \oplus \beta_0 \oplus \gamma_0$ ,  $a' = \alpha'_0 \oplus \beta'_0 \oplus \gamma'_0$ , i. e.  $a, a' \in \mathbb{Z}_2$ . Then  $\text{adp}^{\text{XR}}((\alpha, \alpha'), (\beta, \beta') \xrightarrow{1} (\gamma', \gamma)) = p \cdot \text{adp}_{a',0}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus a}) + q \cdot \text{adp}_{\bar{a}',1}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus a})$ , where  $p = \text{adp}_a^{\oplus}(\alpha' \oplus a', \beta' \rightarrow \gamma') \leq 1$  and  $q = \text{adp}_a^{\oplus}(\bar{\alpha}' \oplus a', \bar{\beta}' \rightarrow \gamma') \leq 1$ , see Corollary 1. At the same time,

$$\text{adp}^{\text{XR}}((\alpha, \alpha'), (\alpha, \alpha') \xrightarrow{1} 0) = \text{adp}_{0,0}^{\oplus}(\alpha, \alpha \rightarrow 0) + \text{adp}_{1,1}^{\oplus}(\alpha, \alpha \rightarrow 0).$$

Since  $p, q \leq 1$ , Lemma 2 provides that  $p \cdot \text{adp}_{a',0}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus a}) + q \cdot \text{adp}_{\bar{a}',1}^{\oplus}(\alpha, \beta \rightarrow \gamma^{\oplus a}) \leq \text{adp}^{\text{XR}}((\alpha, \alpha'), (\alpha, \alpha') \xrightarrow{1} 0)$ .  $\square$

## 6 Maximums of $\text{adp}^{\text{XR}}$ for $r = n - 1$

In this section we consider maximums of  $\text{adp}^{\text{XR}}$  for  $r = n - 1$  (in other word, the rotation is one bit right), where one of the first two arguments is fixed. Let us start with the following lemmas.

**Lemma 3.** *Let  $\alpha \in \mathbb{Z}_2^n$  and  $c \in \mathbb{Z}_2$ . Then it is true that  $\max_{\beta, \gamma \in \mathbb{Z}_2^n} \text{adp}_c^{\oplus}(\alpha, \beta \rightarrow \gamma) \leq \max_{\beta, \gamma \in \mathbb{Z}_2^n} \text{adp}_0^{\oplus}(\alpha, \beta \rightarrow \gamma) = \max_{\beta, \gamma \in \mathbb{Z}_2^n} \text{adp}^{\oplus}(\alpha, \beta \rightarrow \gamma) = \text{adp}^{\oplus}(\alpha, \alpha \rightarrow 0)$ .*

*Proof.* It is clear that  $\max_{\beta, \gamma} \text{adp}_c^\oplus(\alpha, \beta \rightarrow \gamma) \leq \max_{\beta, \gamma} \text{adp}^\oplus(\alpha, \beta \rightarrow \gamma)$ . Also, [16, Theorem 2] provides that  $\max_{\beta, \gamma} \text{adp}^\oplus(\alpha, \beta \rightarrow \gamma) = \text{adp}^\oplus(\alpha, \alpha \rightarrow 0)$ . At the same time,  $\text{adp}^\oplus(\alpha, \alpha \rightarrow 0) = \text{adp}_0^\oplus(\alpha, \alpha \rightarrow 0)$  by definition.  $\square$

**Lemma 4.** *Let  $\alpha \in \mathbb{Z}_2^n$ ,  $\alpha_0 = 0$ . Then  $\text{adp}^\oplus(\bar{\alpha}, \bar{\alpha} \rightarrow 0) \leq \text{adp}^\oplus(\alpha, \alpha \rightarrow 0)$ .*

*Proof.* The case of  $n = 1$  is straightforward. Next,  $\text{adp}^\oplus(\alpha'1, \alpha'1 \rightarrow 0) < \text{adp}^\oplus(\alpha'0, \alpha'0 \rightarrow 0)$  holds for any  $\alpha' \in \mathbb{Z}_2^{n-1}$  by [16, Corollary 5]. Let  $\alpha = \alpha'0$ . Due to the symmetries of  $\text{adp}^\oplus$  (see [16, Proposition 3]),  $\text{adp}^\oplus(\alpha'1, \alpha'1 \rightarrow 0) = \text{adp}^\oplus(-(\alpha'1), -(\alpha'1) \rightarrow 0)$ . Also,  $-(\alpha'1) = \bar{\alpha}'1 = \bar{\alpha}'0 = \bar{\alpha}$ .  $\square$

The next theorem describes how to find  $\max_{\beta, \gamma} \text{adp}^{\text{XR}}(\alpha, \beta \xrightarrow{n-1} \gamma)$ .

**Theorem 7.** *Let us fix the first argument of  $\text{adp}^{\text{XR}}$ . Then*

1.  $\max_{\beta, \gamma \in \mathbb{Z}_2^n} \text{adp}^{\text{XR}}(\alpha 0, \beta \xrightarrow{n-1} \gamma) = \text{adp}^{\text{XR}}(\alpha 0, \alpha 0 \xrightarrow{n-1} 0)$ , where  $\alpha \in \mathbb{Z}_2^{n-1}$ ,
2.  $\max_{\beta, \gamma \in \mathbb{Z}_2^n} \text{adp}^{\text{XR}}(\alpha 0 1, \beta \xrightarrow{n-1} \gamma) = \text{adp}^{\text{XR}}(\alpha 0 1, \alpha 0 0 \xrightarrow{n-1} 2^{n-1})$ ,  $\alpha \in \mathbb{Z}_2^{n-2}$ ,
3.  $\max_{\beta, \gamma \in \mathbb{Z}_2^n} \text{adp}^{\text{XR}}(\alpha 1 1, \beta \xrightarrow{n-1} \gamma) = \text{adp}^{\text{XR}}(\alpha 1 1, \bar{\alpha} 0 0 \xrightarrow{n-1} 2^{n-1})$ ,  $\alpha \in \mathbb{Z}_2^{n-2}$ .

*Proof.* According to Corollary 1,  $\text{adp}^{\text{XR}}((\alpha, \alpha'), (\beta, \beta') \xrightarrow{n-1} ((\gamma', \gamma))) = p \cdot \text{adp}_a^\oplus(\alpha' \oplus \alpha', \beta' \rightarrow \gamma') + q \cdot \text{adp}_a^\oplus(\bar{\alpha}' \oplus \alpha', \bar{\beta}' \rightarrow \gamma')$ , where  $p = \text{adp}_{\alpha', 0}^\oplus(\alpha, \beta \rightarrow \gamma^{\oplus a})$ ,  $q = \text{adp}_{\bar{\alpha}', 1}^\oplus(\alpha, \beta \rightarrow \gamma^{\oplus a})$ . Table 1 shows us possible values of  $p$  and  $q$ :

$\alpha$	0				1			
$\beta$	0		1		0		1	
$\alpha\beta\gamma^{\oplus a}$	000	000	011	011	101	101	110	110
$\alpha'$	0	1	0	1	0	1	0	1
$p$	1	0	1/2	0	1/2	1/2	1/4	1/4
$q$	0	0	0	1/2	0	0	1/4	1/4

**Case 1.**  $\alpha = 0$ , i.e.  $(\alpha, \alpha') = (0, \alpha') = \alpha'0$ . First of all,

$$\text{adp}^{\text{XR}}((0, \alpha'), (0, \alpha') \xrightarrow{n-1} 0) \stackrel{\alpha'=0}{=} \text{adp}_0^\oplus(\alpha', \alpha' \rightarrow 0) = \text{adp}^\oplus(\alpha', \alpha' \rightarrow 0).$$

According to the first four columns of the table above,  $\text{adp}^{\text{XR}}(\alpha'0, (\beta, \beta') \xrightarrow{n-1} (\gamma, \gamma'))$  takes one the following values:  $\text{adp}_a^\oplus(\alpha', \beta' \rightarrow \gamma')$ , 0,  $\frac{1}{2}\text{adp}_a^\oplus(\alpha', \beta' \rightarrow \gamma')$  and  $\frac{1}{2}\text{adp}_a^\oplus(\bar{\alpha}', \bar{\beta}' \rightarrow \gamma')$ . In light of Lemma 3, it is not difficult to see that any of them is not more than  $\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0)$ . The first point is proven.

**Case 2.**  $\alpha = 1$ , i.e.  $(\alpha, \alpha') = (1, \alpha') = \alpha'1$ . According to the last four columns of the table above,  $\text{adp}^{\text{XR}}(\alpha'1, (\beta, \beta') \xrightarrow{n-1} (\gamma, \gamma'))$  takes one of the following values:

$$\frac{1}{2}\text{adp}_a^\oplus(\alpha', \beta' \rightarrow \gamma') \quad (8)$$

$$\frac{1}{2}\text{adp}_a^\oplus(\bar{\alpha}', \beta' \rightarrow \gamma') \quad (9)$$

$$\frac{1}{4}\text{adp}_a^\oplus(\alpha', \beta' \rightarrow \gamma') + \frac{1}{4}\text{adp}_a^\oplus(\bar{\alpha}', \bar{\beta}' \rightarrow \gamma') \quad (10)$$

$$\frac{1}{4}\text{adp}_a^\oplus(\bar{\alpha}', \beta' \rightarrow \gamma') + \frac{1}{4}\text{adp}_a^\oplus(\alpha', \bar{\beta}' \rightarrow \gamma') \quad (11)$$

**Case 2.1**  $\alpha'_0 = 0$ . According to the second point of the theorem, let us define  $m_0 = \text{adp}^{\text{XR}}(\alpha'1, \alpha'0 \xrightarrow{n-1} 2^{n-1}) \stackrel{\alpha'_0=0}{=} \frac{1}{2}\text{adp}_0^\oplus(\alpha', \alpha' \rightarrow 0) = \frac{1}{2}\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0)$ . Then by Lemmas 3 and 4 (they are marked bellow as 3 and 4 respectively) we obtain that

$$m_0 = \frac{1}{2}\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0) \stackrel{3}{\geq} (8),$$

$$m_0 \stackrel{4}{\geq} \frac{1}{2}\text{adp}^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0) \stackrel{3}{\geq} (9),$$

$$\begin{aligned} m_0 &= \frac{1}{4}\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0) + \frac{1}{4}\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0) \\ &\stackrel{4}{\geq} \frac{1}{4}\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0) + \frac{1}{4}\text{adp}^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0) \stackrel{3}{\geq} (10), (11). \end{aligned}$$

The second point is proven.

**Case 2.2**  $\alpha'_0 = 1$ . According to the third point of the theorem, let us define  $m_1 = \text{adp}^{\text{XR}}(\alpha'1, \bar{\alpha}'0 \xrightarrow{n-1} 2^{n-1}) \stackrel{\alpha'_0=1}{=} \frac{1}{2}\text{adp}_0^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0) = \frac{1}{2}\text{adp}^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0)$ . Similarly to the previous case, we obtain that

$$m_1 \stackrel{4}{\geq} \frac{1}{2}\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0) \stackrel{3}{\geq} (8),$$

$$m_1 = \frac{1}{2}\text{adp}^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0) \stackrel{3}{\geq} (9),$$

$$\begin{aligned} m_1 &= \frac{1}{4}\text{adp}^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0) + \frac{1}{4}\text{adp}^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0) \\ &\stackrel{4}{\geq} \frac{1}{4}\text{adp}^\oplus(\alpha', \alpha' \rightarrow 0) + \frac{1}{4}\text{adp}^\oplus(\bar{\alpha}', \bar{\alpha}' \rightarrow 0) \stackrel{3}{\geq} (10), (11). \end{aligned}$$

The theorem is proven. □

Due to Theorem 4, Theorems 6 and 7 provide exactly the same if we fix the second argument of  $\text{adp}^{\text{XR}}$ .

## 7 Conclusion

By rewriting the formula from [17], we have obtained some symmetries of  $\text{adp}^{\text{XR}}$ . They turned out to be similar to the symmetries of  $\text{adp}^{\oplus}$  [16]. Also, if the rotation is one bit left/right, we have found maximums of  $\text{adp}^{\text{XR}}$ , where one of its input differences is fixed. Although these rotations are difficult to meet in real ciphers, the results obtained show us that the optimal differentials of  $\text{adp}^{\text{XR}}$  may be theoretically found in some cases. Also, anything that is true for  $\text{adp}^{\text{XR}}$  works for  $\text{adp}^{\text{RX}}$  as well (taking into account positions of arguments and the rotation parameter). The maximums for other argument fixations as well as for other rotations are the topics for future research.

**Acknowledgements** The work is supported by Mathematical Center in Akademgorodok under agreement No. 075–15–2022–281 with the Ministry of Science and Higher Education of the Russian Federation.

## References

- [1] David J. Wheeler and Roger M. Needham, “TEA, a Tiny Encryption Algorithm”, *FSE 1994*, LNCS, **1008**, Springer, 1994, 363–366.
- [2] Roger M. Needham and David J. Wheeler, “Tea extensions”, *Technical report, Computer Laboratory, University of Cambridge*, 1997 (<http://www.movable-type.co.uk/scripts/xtea.pdf>).
- [3] Akihiro Shimizu and Shoji Miyaguchi, “Fast Data Encipherment Algorithm FEAL”, *EUROCRYPT 1987*, LNCS, **304**, Springer, 1988, 267–278.
- [4] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas and Jesse Walker, “The Skein Hash Function Family”, 2009 (<http://www.skein-hash.info>).
- [5] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks and Louis Wingers, “The SIMON and SPECK Families of Lightweight Block Ciphers”, 2013 (<https://eprint.iacr.org/2013/404>).
- [6] D.J. Bernstein, “Salsa20 specification”, 2005 (<https://cr.yp.to/snuffle/spec.pdf>).
- [7] D.J. Bernstein, “ChaCha, a variant of Salsa20”, 2008 (<https://cr.yp.to/chacha/chacha-20080128.pdf>).
- [8] Jean-Philippe Aumasson and Willi Meier and Raphael C.-W. Phan and Luca Henzen, *The Hash Function BLAKE*, Information Security and Cryptography, Springer, 2014.
- [9] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel and Ingrid Verbauwhede, “Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers”, *SAC 2014*, LNCS, **8781**, Springer, 2014, 1–18.
- [10] Eli Biham and Adi Shamir, “Differential Cryptanalysis of DES-like Cryptosystems”, *Journal of Cryptology*, **4:1** (1991), 3–72.
- [11] Alex Biryukov and Vesselin Velichkov, “Automatic Search for Differential Trails in ARX Ciphers”, *CT-RSA 2014*, LNCS, **8366**, Springer, 2014, 227–250.
- [12] Thomas A. Berson, “Differential Cryptanalysis Mod  $2^{32}$  with Applications to MD5”, *EUROCRYPT 1992*, LNCS, **658**, Springer, 1992, 71–80.
- [13] Magnus Daum, “Cryptanalysis of Hash Functions of the MD4-Family”, *PhD thesis, Ruhr-University of Bochum*, 2005.
- [14] Helger Lipmaa, Johan Wallén and Philippe Dumas, “On the Additive Differential Probability of Exclusive-Or”, *FSE 2004*, LNCS, **3017**, Springer, 2004, 317–331.

- [15] Nicky Mouha, Vesselin Velichkov, Christophe De Cannière, Bart Preneel, “The Differential Analysis of S-Functions”, *SAC 2010*, LNCS, **6544**, Springer, 2011, 36-56.
- [16] Nicky Mouha, Nikolay Kolomeec, Danil Akhtiamov, Ivan Sutormin, Matvey Panferov, Kseniya Titova, Tatiana Bonich, Evgeniya Ishchukova, Natalia Tokareva and Bulat Zhan-tulikov, “Maximums of the Additive Differential Probability of Exclusive-Or”, *IACR Transactions on Symmetric Cryptology*, **2021:2** (2021), 292–313.
- [17] Vesselin Velichkov, Nicky Mouha, Christophe De Cannière and Bart Preneel, “The Additive Differential Probability of ARX”, *FSE 2011*, LNCS, **6733**, Springer, 2011, 342–358.
- [18] Seokhie Hong, Deukjo Hong, Youngdai Ko, Donghoon Chang, Wonil Lee, and Sangjin Lee, “Differential Cryptanalysis of TEA and XTEA”, *ICISC 2003*, LNCS, **2971**, Springer, 2003, 402–417.

## PROBABILISTIC ASPECTS

# Entropically secure cipher for messages generated by Markov chains with unknown statistics

Boris Ryabko<sup>1,2</sup>

<sup>1</sup>Federal Research Center for Information and Computational Technologies , Novosibirsk, Russia

<sup>2</sup>Novosibirsk State University, Russia  
`boris@ryabko.net`

## Abstract

In 2002, Russell and Wang proposed a definition of entropic security, which was developed within the framework of secret-key cryptography. An entropically secure system is unconditionally secure, that is, unbreakable regardless of the adversary's computing power. In 2004, Dodis and Smith further developed the results of Russell and Wang and, in particular, stated that the notion of an entropically secure symmetric encryption scheme is extremely important for cryptography because one can construct entropically secure symmetric encryption schemes with keys much shorter than the length of the input, thus circumventing Shannon's famous lower bound on key length.

In this report we suggest an entropically secure scheme for the case where the encrypted message is generated by a Markov chain with unknown statistics. The length of the required secret key is proportional to the logarithm of the message length (as opposed to the length of the message itself for the one-time pad).

**Keywords:** Information Theory, entropy security, indistinguishability, symmetric encryption scheme, unconditionally secure, Markov chain, unknown statistics.

## 1 Introduction

In 1949, K. Shannon, in his remarkable article [1], described the perfect secret system and showed that the one-time pad is such a system. Since then, it has been generally accepted that the length of the secret key should be equal to the length of the encrypted message (or at least its entropy). Russell and Wang [2] proposed the notion of entropic security, which gives a possibility to build a symmetric encryption scheme with a secret key much shorter than the length of the input, thus, in a sense, circumventing the mentioned Shannon's lower bound on key length. Informally, the entropy-secure symmetric encryption scheme uses the entropy of the input message to make the required secret key shorter.



The concept of entropic security has been generalized and developed by Dodis and Smith [3] and investigated by several other authors [4, 5, 6]. In order to describe it, suppose that there is a sender Alice and a receiver Bob who share a secret key  $K$ , and Alice wants to securely send some message  $M$  to Bob over a public channel. The message  $M$  is assumed to come from some a-priori distribution on  $\Lambda^n$  where  $\Lambda$  is a finite alphabet,  $n \geq 1$ , and  $K$  is a sequence of equally probable and independent binary digits. Informally, the goal is to compute  $E(M, K)$  which allows Bob to extract  $M$  from  $E(M, K)$  using  $K$  and (the decoder)  $D(E, K)$ , ( $D(E, K) = M$ ), in such a way as to reveal “no information” about  $M$  to the adversary Eve beyond what she already knew. It is assumed that  $E(M, K)$  is a probabilistic map, that is, it can also use random numbers, which are unknown to Bob.

The following formal definition of the entropic security belongs to Russell and Wang [2] (see also Dodis and Smith [3]):

**Definition 1.** *A probabilistic map  $E(M, K)$  is said to hide all functions  $f$  on  $\Lambda^n$  to  $\{0, 1\}^*$  with leakage  $\epsilon$ ,  $\epsilon > 0$ , if, for every adversary  $A$ , there exists some adversary  $\hat{A}$  (who does not know  $E(M, K)$ ) such that for all functions  $f$  from  $\Lambda^n$  to  $\{0, 1\}^*$ ,*

$$|Pr\{A(E(M, K)) = f(M)\} - Pr\{\hat{A}() = f(M)\}| \leq \epsilon. \quad (1)$$

(Note that  $\hat{A}$  does not know  $E(M, K)$  and, in fact, she guesses the meaning of the function  $f(M)$ , ignoring  $E(M, K)$ .)

The cipher  $E(M, K)$  is  $\epsilon$ -entropically secure for a probability distribution  $P$  on  $\Lambda^n$  if  $E(M, K)$  hides all functions  $f$  on  $\Lambda^n$  to  $\{0, 1\}^*$  with leakage  $\epsilon$  when  $M$  obeys the distribution  $P$ .

Another concept, namely, that of indistinguishability, provides another way evaluate the strength of the cipher. To describe it, we first need to define min-entropy.

For a probability distribution  $P$  on the alphabet  $S$  the min-entropy is defined as follows:

$$h_{min}(P) = -\log \max_{a \in S} P(a), \quad (2)$$

$\log = \log_2$ .

**Definition 2.** (Dodis and Smith [3].) *A randomized map  $Y()$  is  $(t, \epsilon)$ -indistinguishable if there is a random variable  $G$  such that for every distribution on a set  $\mathbf{M}$  with min-entropy at least  $t$ , we have*

$$SD(Y(M), G) \leq \epsilon,$$

where for two probability distributions  $A, B$

$$SD(A, B) = \frac{1}{2} \sum_{M \in \mathbf{M}} |Pr\{A = M\} - Pr\{B = M\}|.$$

Informally, in what follows the map  $Y()$  will be the cipher and, again,  $G$  does not depend on the ciphered message. So, Eve can guess the message regardless of its cipher.

Dodis and Smith [3] showed that entropy security and indistinguishability are equal (up to small constants in key length). In particular, they show that if a cipher is  $\epsilon$ -entropically secure, it is  $4\epsilon$ -indistinguishable.

The main result of this paper is as follows: We describe an  $\epsilon$ -entropically secure cipher for the case where the probability distribution  $\mu$  is unknown, but it is known that it belongs to class of stationary ergodic Markov chains with finite memory, or connectivity,  $m$ ,  $m \geq 0$ , whose definition is given in Appendix. (If  $m = 0$  then the symbols generated by  $\mu$  are independent and identically distributed – i.i.d.). The length of the required secret key is  $c_1 \log n + c_2 \log(1/\epsilon) + c_3$ , where  $n$  is the length of encrypted sequence,  $c_1, c_2$  and  $c_3$  are constants that depend on  $m$  and the size of the alphabet  $\Lambda$ . (Recall that all participants know  $m$ , but the secret key are known only to Alice and Bob and the key is used only once).

The proposed method is based on the concept of the  $\epsilon$ -entropically secure cipher and some results of universal coding, which makes it possible to efficiently “compress” messages with unknown statistics [7].

## 2 Preliminaries

### 2.1 Universal coding

First, we consider the simplest case where the alphabet is  $\{0, 1\}^n$ ,  $n \geq 1$  and letters are generated by some i.i.d. source  $\mu$  and  $\mu(0), \mu(1)$  are unknown. The goal is to build a lossless code which “compresses”  $n$ -letter sequences in such a way that the average length (per letter) of the compressed sequence is close to the Shannon entropy  $h(\mu)$ , which is the lower limit of the code-word length (lossless code is such that the encoded messages can be decoded without errors and  $h(\mu) = -(\mu(0) \log \mu(0) + (1 - \mu(0)) \log(1 - \mu(0)))$ ) [7, 8].

The first universal code was invented by Fitingoff [9] and we use this code as a part of the suggested entropically secure cipher. In order to describe this code we consider any word  $v \in \{0, 1\}^n$  and denote by  $\nu$  the number of ones in  $v$  and let  $S_\nu$  be the set of  $n$ -length words with  $\nu$  ones. Fitingoff proposed

to encode the word  $v$  by two subwords  $u$  (prefix) and  $w$  (suffix), where  $u$  is the binary notation of an integer  $\nu$  and  $w$  is the index of the word  $v$  in the subset  $S_\nu$ . It is assumed that the words in  $S_\nu$  are ordered 0 to  $(|S_\nu| - 1)$  (say, lexicographically) and the lengths of  $u$  and  $w$  are equal to  $\lceil \log(n + 1) \rceil$  and  $\lceil \log |S_\nu| \rceil$ , respectively. For example, for  $n = 3$ ,  $v = 100$  we obtain  $\nu = 1, u = 01, w = 10$ .

Recall the definition of the so-called prefix-free code. A set of words  $U$  is prefix-free if for any  $u, v \in U$  neither  $u$  is a prefix of  $v$  nor  $v$  is a prefix of  $u$  [8]. Clearly, the Fitingoff code is prefix-free. If some code  $\lambda$  is prefix-free, then for any sequence  $x_1 x_2 \dots x_n, n \geq 1, x_i \in \Lambda$ , the encoded sequence  $\lambda(x_1) \lambda(x_2) \dots \lambda(x_n)$  can be decoded to  $x_1 x_2 \dots x_n$  without errors. Hence, any prefix-free code is a lossless one.

If we denote the Fitingoff code by  $code_F$  we obtain from its description

$$|code_F(v)| = \lceil \log(n + 1) \rceil + \lceil \log |S_\nu| \rceil + 1. \quad (3)$$

For this code the ability to compress messages is based on the simple observation that probabilities of all messages from  $S_\nu$  are equal for any distribution  $\mu$  and, hence,  $\mu(v) \leq 1/|S_\nu|$  for  $\mu$  and any word  $v \in S_\nu$ . From this inequality and (3) we obtain

$$|code_F(v)| \leq \log(n + 1) + 3 + \log(1/\mu(v)). \quad (4)$$

(Let's explain the name "universal code." Clearly, the average code-length  $E_\mu(|code_F|)$  is not greater than  $\log(n + 1) + 3 + nh(\mu)$  and, hence, the average length per letter  $E_\mu(|code_F|)/n$  is not greater than  $h(\mu) + (\log n + 3)/n$ . We can see that  $E_\mu(|code_F|)/n \rightarrow h(\mu)$  if  $n \rightarrow \infty$ . So, one code compresses sequences generated by any  $\mu$ , that is, the code universal.)

The Fitingoff code described generalizes to i.i.d. processes with any finite alphabet  $\Lambda$ , as well as to Markov chains with memory or connectivity  $m$ , based on the same method as for binary i.i.d. [7]. Namely, the set of all  $n$ -letter words is divided into subsets of equiprobable words, and the code of any word is represented by a prefix and a suffix, where the prefix contains the number of the set with equiprobable words which contains the encoded one, and the suffix is the number in this set. It can be shown that the number of sets with equiprobable words is bounded above by  $(|\Lambda| - 1)|\Lambda|^m$  ([7, 8]), and similarly (4) we can deduce that

$$|code_F(v)| \leq \log((|\Lambda| - 1)|\Lambda|^m) + 3 + \log(1/\mu(v)). \quad (5)$$

It is important to note that there exists an algorithm to find the code-words which is based on method of fast calculation of numbers in  $S_\nu$ , see [10]. The complexity of this algorithm is  $O(n \log^3 n \log \log n)$ .

## 2.2 Entropically secure ciphers

Dodis and Smith [3], based on the results of Russell and Wang [2], proved the following

**Theorem (Russell-Wang, Dodis- Smith )** ([2], [3]). Let there be a probability distribution  $\sigma$  on an alphabet  $\Lambda = \{0, 1\}^l$ ,  $l \geq 1$ . Then, for any  $\epsilon > 0$ , there exists an  $\epsilon$ - entropically secure cipher  $E(M, K)$ ,  $M \in \{0, 1\}^l$  with the length of the key

$$|K| = l - h_{min}(\sigma) + 2\log(1/\epsilon) + 2. \quad (6)$$

Take any such cipher and denote it  $cipher_{RW-DS}(M, K)$ . Dodis and Smith described three algorithm of such ciphers with a key length (6) whose complexity grows polynomially in  $l$  and  $\log(1/\epsilon)$  (One such a cipher is described in Appendix).

It is important to note that each of the three constructions of the ciphers depends only on min-entropy, that is, the cipher construction is the same for all distributions with the same min-entropy (but, of course, depends on  $\epsilon$  and  $l$ ).

## 3 The cipher

### 3.1 Randomised prefix-free codes

Let  $\lambda$  be a prefix-free code for some alphabet  $\Lambda^*$  and  $L = \max_{a \in \Lambda^*} |\lambda(a)|$ . The randomized code  $\rho_\lambda$  maps elements from  $\Lambda^*$  to the set  $\{0, 1\}^L$  defined as follows:

$$\rho_\lambda(a_i) = \lambda(a_i) r_{|\lambda(a_i)|+1}^i r_{|\lambda(a_i)|+2}^i \cdots r_L^i, \quad (7)$$

where  $r_{|\lambda(a_i)|+1}^i, r_{|\lambda(a_i)|+2}^i, \dots, r_L^i$  are uniformly distributed and independent random bits (for all  $i$ ).

Let us define the probability distribution  $\pi_{\lambda, \mu}$  on  $\{0, 1\}^L$  as follows:

$$\begin{aligned} \pi_{\lambda, \mu}(y_1 y_2 \dots y_L) &= \mu(a) 2^{-(L-|\lambda(a)|)} \\ &\text{if } y_1 y_2 \dots y_{|\lambda(a_i)|} = \lambda(a). \end{aligned} \quad (8)$$

If for some  $y = y_1 \dots y_L$  any  $\lambda(a)$  is not a prefix of  $y$ , then  $\pi_{\lambda, \mu}(y) = 0$ .

Let us estimate the min-entropy of the distribution  $\pi_{\lambda, \mu}$ . From this equation and the definition of the min-entropy (2) we obtain the following:

$$h_{min}(\pi_{\lambda, \mu}) = L - \max_{a \in \Lambda} (|\lambda(a)| - \log(1/\mu(a))). \quad (9)$$

Now we consider the Fitingoff code applied to  $n$ -letter sequences generated by a Markov chain  $\mu$  of memory  $m$  over some alphabet  $\Lambda$ . The Fitingoff code is prefix-free and, hence, from (5) and (9) we obtain the following

**Statement.** For any distribution  $\mu$

$$h_{\min}(\pi_{code_F, \mu}) > L - (|\Lambda|^m (|\Lambda| - 1) \log n + 3). \quad (10)$$

In particular, for an i.i.d. source with binary alphabet

$$h_{\min}(\pi_{code_F}) > L - (\log n + 3).$$

### 3.2 Description of the cipher

Here we describe a cipher with the key of length  $const_1 \log n + const_2 \log(1/\epsilon) + const_3$ , which is  $\epsilon$ -entropically secure for  $n$ -letter sequences generated by any (unknown) Markov chain  $\mu$  of memory  $m$  over some alphabet  $\Lambda$ .

Briefly, the encryption is done as follows: first compress the message with the Fitingoff code, then randomize the encoded message according to (7) and then encrypt the received  $\rho_{code_F, \mu}()$  with an entropically secure cipher. (Note that the distribution of  $\mu$  is unknown.)

In detail, this algorithm is as follows:

**Parameters:**  $\epsilon > 0$ , the alphabet  $\Lambda$ , the memory of Markov chain  $m$  and the length of the ciphered message  $n$ .

**Input:** a word  $v \in \Lambda^n$ .

**1st step:** Encode  $v$  with the Fitingoff code  $code_F(v)$  (with parameters  $\Lambda, m$  and  $n$ ).

**2nd step:** Calculate the random word  $\rho_{code_F}(v) (\in \{0, 1\}^L)$ .

**3rd step:** Calculate the  $\epsilon$ -entropically secure cipher  $cipher_{RW-DS}(\rho_{code_F}(v), K)$  with the length of the secret key  $|K| = (|\Lambda|^m (|\Lambda| - 1) \log n + 2 \log(1/\epsilon) + 5)$  bits.

**Output:**  $cipher_{RW-DS}(\rho_{code_F}(v))$ .

The decryption algorithm is as follows: first Bob decrypts the word  $E(\rho_{code_F}(v), K)$  ( $= cipher_{RW-DS}(\rho_{code_F}(v))$ ) with the known secret key  $K$  and obtains the word  $\rho_{code_F}(v)$ . Then, based on the prefix-free property of the Fitingoff code, Bob finds the word  $code_F(v)$  and then decodes it to get  $v$ .

The described cipher uses compression and randomisation. Denote it  $cipher_{c\&r}$ .

The theorem of Russell-Wang and Dodis-Smith guarantees the entropic security and indistinguishability for the first cipher  $cipher_{RW-DS}$ , so, we

need to prove a similar property for the proposed  $cipher_{c\&r}$ . Despite the equivalence of the concepts of entropic security and indistinguishability [3], we will prove these properties separately due to the great importance of this fact for the described cipher  $cipher_{c\&r}$ .

The following theorem describes the entropic security property for this cipher:

**Theorem 1.** *Let  $\epsilon > 0$  and suppose that the cipher  $cipher_{c\&r}$  is applied to  $n$ -letter words  $M$  generated by a stationary ergodic Markov chain with memory  $m, m \geq 0$ , and an alphabet  $\Lambda$ , and let the length of the secret key  $K$  be  $(|\Lambda|^m(|\Lambda| - 1) \log n + 2 \log(1/\epsilon) + 5)$ . Then  $cipher_{c\&r}$  is  $\epsilon$ -entropically secure, that is, for any function  $A : \{0, 1\}^L \rightarrow \{0, 1\}^*$  and  $f : \Lambda^n \rightarrow \{0, 1\}^*$  there exists such a function  $\hat{A} : \{0, 1\}^L \rightarrow \{0, 1\}^*$  that*

$$|Pr\{A(cipher_{c\&r}(M, K) = f(M))\} - Pr\{\hat{A}() = f(M)\}| \leq \epsilon,$$

where  $\hat{A}$  does not use  $cipher_{c\&r}(M)$ .

*Proof.* The cipher  $cipher_{RW-DS}(\rho_{code_F}(v), K)$  with the length of the secret key  $|K| = (|\Lambda|^m(|\Lambda| - 1) \log n + 2 \log(1/\epsilon) + 5)$  is applied to  $\{0, 1\}^L$  (see the step 3). First we note that the cipher is  $\epsilon$ -entropically secure. Indeed, from Theorem of Russell-Wang and Dodis-Smith (see (6)) and the estimate of the min-entropy (10) we can see that such a cipher exists for the distribution  $\pi_{code_F, \mu}$  for any (unknown)  $\mu$ . So, from the definition of  $\epsilon$ -entropical security we can see that for any function  $g$

$$|Pr\{A(cipher_{RW-DS}(v) = g(v))\} - Pr\{\hat{A}() = g(v)\}| \leq \epsilon,$$

where  $v, v \in \{0, 1\}^L$ ,  $g$  is any function defined on  $\{0, 1\}^L$  ( $g : \{0, 1\}^L \rightarrow \{0, 1\}^*$ ) and  $\hat{A}()$  does not depend on  $v$  (to be short,  $\lambda = code_F$ ). Taking into account that the code  $\lambda$  is prefix-free, we can define such a function  $\phi$  that for any  $a \in \Lambda^n$  and  $u = \rho_\lambda(a)$ ,  $\phi(u) = a$ . For any function  $f : \Lambda^n \rightarrow \{0, 1\}^*$  and  $M$  consider the function  $g(\rho_\lambda(M)) = f(\phi(\rho_\lambda(M))) (= f(M))$ . This equation is valid for the function  $g$  and for  $v = \rho_\lambda(M)$ , hence

$$|Pr\{A(cipher_{ds}(\rho_\lambda(M)) = f(\phi(\rho_\lambda(M))))\} - Pr\{\hat{A}() = f(\phi(\rho_\lambda(M)))\}| \leq \epsilon.$$

Taking into account that  $cipher_{c\&r}(M) = cipher_{RW-DS}(\rho_\lambda(M))$  and  $f(\phi(\rho_\lambda(M))) = f(M)$ , we can see from the latter inequality that

$$|Pr\{A(cipher_{c\&r}(M)) = f(M)\} -$$

$$Pr\{\hat{A}() = f(M)\} \leq \epsilon.$$

The theorem is proven.

The following theorem establishes indistinguishability of  $cipher_{c\&r}$ .

**Theorem 2.** *Let  $\epsilon > 0$  and suppose that the cipher  $cipher_{c\&r}$  is applied to  $n$ -letter words  $M$  generated by a stationary ergodic Markov chain with memory  $m, m \geq 0$ , and an alphabet  $\Lambda$ , and let the length of the secret key  $K$  be  $(|\Lambda|^m(|\Lambda| - 1) \log n + 2 \log(1/\epsilon) + 5)$ . Then, this cipher is  $4\epsilon$ -indistinguishable.*

*Proof.* The cipher  $cipher_{RW-DS}$  is  $\epsilon$ -entropically secure (see Theorem 1). As we mentioned in Introduction, Dodis and Smith [3] showed that it means that this cipher is  $4\epsilon$ -indistinguishable. Our goal is to prove this property for  $cipher_{c\&r}$ . The  $4\epsilon$ -indistinguishability means that  $SD(cipher_{RW-DS}, G) \leq 4\epsilon$ , where  $G$  is a random variable on  $\{0, 1\}^L$  (which is independent on  $cipher_{RW-DS}$ ).

Define  $U_a = \{cipher_{RW-DS}(\lambda(a)r) : r \in \{0, 1\}^{L-\lambda(a)}\}$  and let the a random variable of  $G'(v)$  be defined as follows:

$$Pr\{G' = v\} = \sum_{w \in U_v} Pr\{G = w\}.$$

The following chain of equalities and inequalities is based on these definitions and the triangle inequality for  $L_1$ :

$$\begin{aligned} SD(cipher_{c\&r}, G') &= \\ \frac{1}{2} \sum_{u \in \Lambda^n} |Pr\{cipher_{c\&r} = u\} - Pr\{G' = u\}| &= \\ \frac{1}{2} \sum_{v \in \{0, 1\}^n} \left| \sum_{w \in U_v} (Pr\{cipher_{RW-DS} = w\} - Pr\{G = w\}) \right| &\leq \\ \frac{1}{2} \sum_{v \in \Lambda^n} \sum_{w \in U_v} |Pr\{cipher_{RW-DS} = w\} - Pr\{G = w\}| &= \\ \frac{1}{2} \sum_{w \in \{0, 1\}^L} |Pr\{cipher_{RW-DS} = w\} - Pr\{G = w\}| &= \\ SD(cipher_{RW-DS}, G) &\leq 4\epsilon. \end{aligned}$$

So,  $SD(cipher_{c\&r}, G') \leq 4\epsilon$ .

Theorem is proven.

Let us estimate the complexity of encoding and decoding. As we mentioned above, the encoding and decoding fitting complexity is  $O(n \log^{const})$ . The complexity of the Dodis and Smith cipher is polynomial in  $n$ . Thus, the complexity of the proposed cipher is also polynomial in  $n$ .

## 4 Appendix

### 4.1 The definition of a stationary ergodic Markov chain with memory, or connection, $m$ .

First we give a definition of stationary ergodic processes. The time shift  $T$  on  $\Lambda^\infty$  is defined as  $T(x_1, x_2, x_3, \dots) = (x_2, x_3, \dots)$ . A process  $P$  is called stationary if it is  $T$ -invariant:  $P(T^{-1}B) = P(B)$  for every Borel set  $B \subset \Lambda^\infty$ . A stationary process is called ergodic if every  $T$ -invariant set has probability 0 or 1:  $P(B) = 0$  or  $1$  whenever  $T^{-1}B = B$  [11, 12].

We denote by  $M_\infty(\Lambda)$  the set of all stationary and ergodic sources and let  $M_0(\Lambda) \subset M_\infty(\Lambda)$  be the set of all i.i.d. processes. We denote by  $M_m(\Lambda) \subset M_\infty(\Lambda)$  the set of Markov sources of order (or with memory, or connectivity) not larger than  $m$ ,  $m \geq 0$ . By definition  $\mu \in M_m(\Lambda)$  if

$$\begin{aligned} & \mu(x_{t+1} = a_{i_1} | x_t = a_{i_2}, x_{t-1} = a_{i_3}, \dots, x_{t-m+1} = a_{i_{m+1}}, \dots) \\ & = \mu(x_{t+1} = a_{i_1} | x_t = a_{i_2}, x_{t-1} = a_{i_3}, \dots, x_{t-m+1} = a_{i_{m+1}}) \end{aligned}$$

for all  $t \geq m$  and  $a_{i_1}, a_{i_2}, \dots \in \Lambda$ .

### 4.2 Entropically secure ciphers.

In this part we describe one entropically secure cipher from [3], part 3.2.

Let  $\{h_i\}_{i \in I}$  be some family of functions  $h_i : \{0, 1\}^k \rightarrow \{0, 1\}^n$ , indexed over the set  $I = \{0, 1\}^r$ . By definition, a collection of functions from  $n$ -bit words to  $n$ -bits is XOR-universal if:

$$\forall a, x, y \in \{0, 1\}^n, x \neq y, Pr\{h_i(x) \oplus h_i(y) = a\} \leq \frac{1}{2^{n-1}},$$

if  $i$  is randomly chosen from  $I$  according to the uniform distribution ( $\oplus$  is symbol-by-symbol modulo 2 summation). Also, suppose that there is a XOR-universal collection of functions whose description is public and, hence, it is known to Alice, Bob and Eve.

Dodis and Smith consider an encryption scheme of the form

$$E(m, K, i) = (i; m \oplus h_i(K))$$



where  $i$  is randomly chosen from  $I$  according to the uniform distribution, and  $K$  is a  $k$ -bit secret key. Note that  $m$  is a ciphered message of length  $n$ ,  $i$  is the number of  $h_i$  in the set  $I$  and  $i = \log |I| = r$ . (Dodis and Smith notice that this scheme is a special low-entropy, probabilistic one-time pad.) Decryption is obviously possible, since the description of the function  $h_i$  is public. It is shown [3] that this cipher is  $\epsilon$ -entropically secure for  $|k| \geq n - h_{\min} + 2 \log(1/\epsilon) + 2$  if the function family  $\{h_i\}_{i \in I}$  is XOR-universal.

An example of XOR-universal family is as follows [3]: View  $\{0, 1\}^n$  as  $\mathcal{F} = GF(2^n)$ , and embed the key set  $\{0, 1\}^k$  as a subset of  $\mathcal{F}$ . For any  $i \in \mathcal{F}$ , let  $h_i(K) = iK$ , with multiplication in  $\mathcal{F}$ . This yields a family of linear maps  $\{h_i\}$  with  $2^n$  members. For this family the complexity of ciphering and deciphering is  $O(n \log n \log \log n)$  [3].

It is important to note that the length of the secret key ( $k$ ) depends only on the min-entropy of the probability distribution and does not depend on other parameters of the distribution.

## References

- [1] Shannon C. E., "Communication theory of secrecy systems", *The Bell system technical journal*, 1949, 656-715.
- [2] Russell A, Wang H., "How to fool an unbounded adversary with a short key", *IEEE Transactions on Information Theory*, 2006, 1130-1140.
- [3] Dodis Y., Smith A., "Entropic security and the encryption of high entropy messages", Theory of Cryptography Conference, 2005, 556-577.
- [4] Li X., Tang Q., Zhang Z., "Fooling an Unbounded Adversary with a Short Key, Repeatedly: The Honey Encryption Perspective", 2nd Conference on Information-Theoretic Cryptography (ITC 2021), 2021. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 1-21.
- [5] Juels A, Ristenpart T., "Honey encryption: Security beyond the brute-force bound", Annual international conference on the theory and applications of cryptographic techniques, 2014 May 11, 293-310.
- [6] Ryabko B, "Using data compression and randomization to build an unconditionally secure short key cipher", Cryptology ePrint Archive, Report 2021/1667, 2021.
- [7] Krichevsky R., *Universal compression and retrieval*, Springer Science & Business Media, 1994.
- [8] Cover T. M. , Thomas J. A., *Elements of information theory*, Wiley-Interscience, 2006..
- [9] Fitingof B. M., "Optimal coding in the case of unknown and changing message statistics", *Problemy Peredachi Informatsii*, 2(2), 1966, 3-11.
- [10] Ryabko B.Ya., "The fast enumeration of combinatorial objects", *Discrete Math.and Applications*, v.10, n. 2, 1998, 163-182.
- [11] Billingsley P., *Ergodic Theory and Information*, John Wiley and Sons, Inc., 1965.
- [12] Ryabko D., *Asymptotic nonparametric statistical analysis of stationary time series*, Springer, 2019.

# Two Lempel-Ziv goodness-of-fit criteria for nonequiprobable random binary sequences

Vasiliy Kruglov

Steklov Mathematical Institute of Russian Academy of Sciences, Moscow, Russia  
kruglov@mi-ras.ru

## Abstract

Consider a random binary sequence  $X_1, \dots, X_n$  and hypothesis  $H_p$  that elements of this sequence are independent and identically distributed on the set  $\{0, 1\}$  with probabilities  $\mathbf{P}\{X_i = 1\} = p$ ,  $\mathbf{P}\{X_i = 0\} = 1 - p$ , where  $p \in (0, 1)$ . Earlier two goodness-of-fit criteria for the hypothesis  $H_{0.5}$  were proposed, these criteria were based on computation of Lempel-Ziv statistics. In this paper these criteria are generalized for any  $p \in (0, 1)$ .

For both criteria a sequence of the length  $n = mT$  is divided into  $m$  blocks of (equal) length  $T$ , for these blocks we compute values of Lempel-Ziv statistics  $W_1(T), \dots, W_m(T)$ . If the hypothesis  $H_p$  is true, these values are independent and their distributions are equal, so we may construct goodness-of-fit tests for hypothesis  $H_p$  based on these statistics via standard methods.

The first criterion is based on the statistic  $\tilde{W}(2mT) = (W_1 + W_2 + \dots + W_m) - (W_{m+1} + W_{m+2} + \dots + W_{2m})$ , the distribution of this statistic is symmetric about zero.

The statistic of the second criterion is the value  $\tilde{\chi}^2(mrT) = \max_{1 \leq k \leq m} \chi_k^2(T)$ , where  $\chi_1^2(T), \dots, \chi_m^2(T)$  are values of chi-square statistics corresponding to  $W_1(T), \dots, W_m(T)$ .

For both criteria we propose limit distributions of statistics, and for the first criterion we also obtain an estimation for the speed of convergence to the limit normal distribution.

**Keywords:** Lempel-Ziv, RNG testing, statistical criterion, computation.

## 1 Introduction

The most common type of binary sequences used in cryptography are sequences with independent elements that are equiprobably distributed on the set  $\{0, 1\}$ . However, non-equiprobable distributions are also a part of cryptography: lattice-based cryptography is based on samples from discrete Gaussian distributions on integers and in McEliece cryptosystem random binary vectors with fixed numbers of 0's and 1's are used.

The simplest non-equiprobable discrete distribution is Bernoulli distribution on the set  $\{0, 1\}$ , for this distribution probability of 1 is equal to  $p$  and

probability of 0 is equal to  $1 - p$  for some  $p \in (0, 1)$ . In this paper we generalize our previous results of studying distributions of Lempel-Ziv statistic for equiprobable binary sequences with  $p = 0.5$  and goodness-of-fit criteria based on this statistic for any  $p \in (0, 1)$ .

For computation of Lempel-Ziv statistic a sequence  $X_1, X_2, \dots$  of elements of alphabet  $\{0, 1\}$  is divided into subsequences of digits (words) in such a way that any next word is the least word that is not equal to any of previous words; the first word is the empty word. The statistic of Lempel-Ziv criterion is the amount  $W(T)$  of words obtained in such a way for a sequence  $X = (X_1, X_2, \dots, X_T)$  of the length  $T$ .

Examples:

Binary sequence 011101101011 of 12 digits is divided into 7 words  $\emptyset, (0), (1), (11), (01), (10), (101)$  and remainder 1 that is not considered because it is equal to the third word.

Binary sequence 010101101010 of 12 digits is divided into 7 words  $\emptyset, (0), (1), (01), (011), (010), (10)$  without remainder.

The main hypothesis considered in this paper is the hypothesis  $H_p$  that digits of sequence  $X$  are independent and distributed on  $\{0, 1\}$  with probabilities

$$\mathbf{P}\{X_i = 1\} = p, \mathbf{P}\{X_i = 0\} = 1 - p,$$

where  $p \in (0, 1)$ .

In section 5 of this paper we present a method for computation of distribution of random variable  $W(T)$  for the hypothesis  $H_p$ . This method was proposed in a previous paper of V.G. Mikhailov ([4]).

Remark that due to symmetry the distributions of  $W(T)$  for hypothesis  $H_p$  and  $H_{1-p}$  are exactly equal.

Distributions of  $W(T)$  for  $T = 1000$  and  $p = 0.1, 0.5, 0.9$  are given in Appendix, table 1. Values of  $\mathbf{E}W(T)$ ,  $\mathbf{D}W(T)$ ,  $\sigma(W(T)) = \sqrt{\mathbf{D}W(T)}$  and  $\mathbf{E}(W(T))^3$  for  $T = 1000, 2000, \dots, 6000$  and  $p = 0.1, 0.2, \dots, 0.9$  are presented in Appendix, table 2.

The Lempel-Ziv criterion is a goodness-of-fit criterion for the hypothesis  $H_p$ , this criterion is defined as follows:

$$\{|W(T) - \mu(T)| < C\sigma(T)\} \Rightarrow H_p,$$

$$\{|W(T) - \mu(T)| \geq C\sigma(T)\} \Rightarrow \bar{H}_p,$$

where  $C$  is a critical level,  $H_p$  is the full alternative for hypothesis  $H_p$ ,  $\mu(T) = \mathbf{E}W(T)$  and  $\sigma(T) = \sqrt{\mathbf{D}W(T)}$ .

Lempel-Ziv criterion for hypothesis  $H_{0.5}$  for a long time was a part of the NIST Statistical Test Suite — well-known collection of statistical criteria for testing the quality of random and pseudorandom equiprobable binary sequences designed by the National Institute of Standards and Technology, USA ([1], [2]).

Due to insufficient knowledge about speed of convergence of the distribution of statistic  $W(T)$  to the limit distribution, it was recommended to use Lempel-Ziv criterion only for long equiprobable binary sequences, i.e. for  $T \geq 10^6$  ([1]). This shortcoming was mentioned as one of reasons for removing this criterion from NIST Suite ([3]). Our method allows to compute the exact distribution of statistic  $W(T)$  and calculate probability to reject the hypothesis  $H_p$  if it is correct for any  $p \in (0, 1)$ .

In this paper we propose two goodness-of-fit criteria for hypothesis  $H_p, p \in (0, 1)$ . We also propose limit distributions for statistics of these criteria and for the first statistic we have obtained explicit estimations for the speed of convergence to limit (normal) distribution.

## 2 Criterion with summation

Divide a sample  $X = (X_1, X_2, \dots, X_{2mT})$  into  $2m$  nonintersecting blocks of the length  $T$  and for each of these  $2m$  blocks compute value of Lempel-Ziv statistic  $W(T)$ . Denote computed values as  $W_1, W_2, \dots, W_{2m}$  and compute statistic

$$\tilde{W}(2mT) = (W_1 + W_2 + \dots + W_m) - (W_{m+1} + W_{m+2} + \dots + W_{2m}).$$

Formula for this statistic may be rewritten as

$$\tilde{W}(2mT) = \sum_{i=1}^m V_i(2T) = \sum_{i=1}^m (W_i(T) - W_{i+m}(T)),$$

in this way random variable  $\tilde{W}(2mT)$  is represented as the sum of  $m$  independent values  $V_i(2T)$  that are identically distributed and

$$\mathbf{E}V_i(2T) = 0, \mathbf{E}\tilde{W}(2mT) = 0, \mathbf{D}\tilde{W}(2mT) = 2m\mathbf{D}W(T).$$

Goodness-of-fit criterion for hypothesis  $H_p$  for sample  $X$  that is based on statistic  $\tilde{W}(2mT)$  is defined as follows:

$$\left\{ |\tilde{W}(2mT)| < C\sqrt{\mathbf{D}\tilde{W}(2mT)} \right\} \Rightarrow H_p,$$

$$\left\{ |\tilde{W}(2mT)| \geq C\sqrt{\mathbf{D}\tilde{W}(2mT)} \right\} \Rightarrow \bar{H}_p.$$

Hypothesis  $\bar{H}_p$  is the full alternative to hypothesis  $H_p$ ,  $C$  is the critical level and for a given significance level  $\alpha > 0$  the critical level  $C$  may be chosen by rule  $2(1 - \Phi(C)) \approx \alpha$ . However, for given values  $m$  and  $T$  we can compute exact distribution of statistic  $\tilde{W}(2mT)$  and calculate exact probabilities for the first and the second type of errors for any critical level  $C$ .

For a given values of  $m$  and  $T$  the distribution of  $W(T)$  may be computed by formulae from section 5 and after that the distribution of  $V_i(2T)$  may be computed by formula

$$\mathbf{P}\{V_i(2T) = k\} = \sum_l \mathbf{P}\{W(T) = l\}\mathbf{P}\{W(T) = l - k\}.$$

For random variable  $V(2T)$  we have computed for  $T = 1000, \dots, 6000$  and  $p = 0.1, 0.2, \dots, 0.9$  values  $\mathbf{E}|V(2T)|$ ,  $\mathbf{D}V(2T)$ ,  $\sigma(V(2T)) = \sqrt{\mathbf{D}V(2T)}$  and  $\mathbf{E}|V(2T)|^3$ , these values are presented in Appendix, table 3.

The distribution of random variable  $\tilde{W}(2mT)$  may be computed as the  $m$ -fold convolution of the distribution  $V_i(2T)$ .

Examples of distributions of  $\tilde{W}(2mT)$  for  $T = 1000, p = 0.5, m = 1$  and  $m = 10$  are presented in Appendix, table 4. Values of expectation and variance of  $\tilde{W}(2mT)$  are not presented because expectation of  $\tilde{W}(2mT)$  is equal to zero for any  $T$  and  $m$ , and variance of  $\tilde{W}(2mT)$  may be easily calculated via equality  $\mathbf{D}\tilde{W}(2mT) = 2m\mathbf{D}W(T)$  and value of  $\mathbf{D}W(T)$  from Appendix, table 2.

## 2.1 On the accuracy of normal approximation for $\tilde{W}(2mT)$ .

The accuracy of normal approximation of the distribution of statistic  $\tilde{W}(2mT)$  may be estimated via well-known Berry–Esseen inequality (inequality for constant  $C_1$  may be found in [5]).

**Theorem 1.** *For distribution function of random variable  $\tilde{W}(2mT)$  the following inequality is valid:*

$$\sup_{-\infty < x < \infty} \left| \mathbf{P} \left\{ \frac{\tilde{W}(2mT)}{\sqrt{\mathbf{D}\tilde{W}(2mT)}} < x \right\} - \Phi(x) \right| \leq \frac{C_1 \mathbf{E}|V(2T)|^3}{(2m\mathbf{D}W(T))^{3/2}}, \quad (1)$$

where  $C_1 \leq 0.4774$ .

**Corollary 1.** *If  $m \rightarrow \infty$ , then for any  $-\infty < x < \infty$*

$$\mathbf{P} \left\{ \frac{\tilde{W}(2mT)}{\sqrt{\mathbf{D}\tilde{W}(2mT)}} < x \right\} \rightarrow \Phi(x).$$

Computed values of the right part of inequality (1) for  $T = 1000, \dots, 6000$ ,  $p = 0.1, 0.2, \dots, 0.9$  and  $m = 1000, 2000$  are given in Appendix, table 5. These values show that this upper bound for accuracy of normal approximation of  $\tilde{W}(2mT)$  significantly depends on amount of blocks  $m$  and almost does not depend on size of block  $T$  and on value of probability  $p$ . As it is reasonable to expect, for a sample of fixed size the higher accuracy is obtained if the sample is divided into the greater number of blocks. As an example, for a sample of size  $2mT = 4 \cdot 10^6$  the right part of inequality (1) for  $m = 2000$  is approximately 3.5 times smaller than for  $m = 1000$ .

## 2.2 Criterion with summation: exact values

A goodness-of-fit criterion for hypothesis  $H_p$  based on statistic  $\tilde{W}(2mT)$  may be constructed as follows.

For sample  $X_1, \dots, X_{2mT}$  we compute value of  $\tilde{W}(2mT)$  and accept or reject hypothesis  $H_p$  by following rules:

$$\left\{ |\tilde{W}(2mT)| < l \right\} \Rightarrow H_p,$$

$$\left\{ |\tilde{W}(2mT)| \geq l \right\} \Rightarrow \bar{H}_p.$$

Here  $\bar{H}_p$  is the full alternative to hypothesis  $H_p$ ,  $l$  is the critical level.

The probability to reject hypothesis  $H_p$  if it is correct is equal to

$$\begin{aligned} \alpha_l &= \mathbf{P}\{H_p \text{ rejected} | H_p \text{ correct}\} = \mathbf{P}\left\{|\tilde{W}(2mT)| \geq l | H_p\right\} = \\ &= 1 - \mathbf{P}\left\{|\tilde{W}(2mT)| < l | H_p\right\}. \end{aligned}$$

Now we present an example of such a criterion for equiprobable distribution  $p = 0.5$ , size of block  $T = 1000$  and  $m = 10$ . Remind that distributions of statistics  $\tilde{W}(2mT)$  for  $T = 1000$ ,  $p = 0.5$  and  $m = 1$  or  $m = 10$  may be found in Appendix, table 4. Values for probability of error  $\alpha_l$  for  $m = 10$  and different values of level  $l$  are presented in following table:

$l$	1	2	3	4	5	6	7	8
$\alpha_l$	0.9090	0.7317	0.5678	0.4238	0.3038	0.2089	0.1376	0.0867
$l$	9	10	11	12	13	14	15	16
$\alpha_l$	0.0522	0.0300	0.0165	0.0086	0.0043	0.0020	0.0009	0.0004

Values of  $\alpha_l$  for  $m = 10$  and  $l = 1, \dots, 16$ .

Similar goodness-of-fit criteria for hypothesis  $H_p$  may be constructed in the same way for any value of  $p \in (0, 1)$ .

### 3 Criterion of chi-square type

Consider amount of natural numbers with significant probability that random variable  $W(T)$  is equal to that number. As an example, for any  $T$  consider amount of  $n$  such that  $\mathbf{P}\{W(T) = n\} \geq 0.0001$ . Analysis of distributions of  $W(T)$  for different values of  $T$  and  $p$  shows that for fixed value of  $T$  amount of such numbers  $n$  grows as probability  $p$  moves away from equiprobable value  $p = 0.5$ : for  $T = 1000$  amount of such  $n$  is equal to 8 for  $p = 0.5$  and is equal to 33 for  $p = 0.1$ .

We propose a criterion of chi-square type that is based on well-known fact of mathematical statistics (e.g. [6], §3.2, section 2).

**Statement 1.** Let the support of a random variable  $\xi$  be divided into intervals  $\Delta_1, \dots, \Delta_N$ . Let simple hypothesis  $H_0$  for the distribution of  $\xi$  be considered and let  $\mathbf{P}\{\xi \in \Delta_j\} = p_j^0$ ,  $j = 1, \dots, N$ , if this hypothesis is true. For a given sample  $X_1, \dots, X_n$  of values of random variable  $\xi$  and any  $j = 1, \dots, N$  compute values  $v_j = \sum_{i=1}^n I_{\{X_i \in \Delta_j\}}$  and value

$$\chi^2 = \sum_{j=1}^N \frac{(v_j - np_j^0)^2}{np_j^0}.$$

Then for  $n \rightarrow \infty$  the distribution of random variable  $\chi^2$  converges to chi-square distribution with  $N - 1$  degrees of freedom.

Denote the distribution function of chi-square distribution with  $N - 1$  degrees of freedom as  $\chi_{N-1}^2(x)$ . For a given significance level  $\alpha \in (0, 1)$  define critical level  $C(N - 1, \alpha)$  by equality

$$\chi_{N-1}^2(C(N - 1, \alpha)) = \alpha. \quad (5)$$

For chi-square criterion the hypothesis  $H_0$  is accepted if

$$\{\chi^2 \leq C(N - 1, \alpha)\}$$

and is rejected if

$$\{\chi^2 > C(N - 1, \alpha)\}.$$

Now we propose goodness-of-fit criterion for hypothesis  $H_p$  that is based on dividing samples into blocks and using chi-square criterion. For clearness of explanation we use explicit values of probabilities of distribution  $W(T)$  for  $T = 1000$  and  $p = 0.1$  (ref. Appendix, table 1).

Consider sample  $X_1, \dots, X_n$  of  $n = mrT$  digits, each digit is equal to zero or one. We divide this sample into  $mr$  nonintersecting blocks of the length  $T$  and for each block compute value  $W(T)$ , as a result we obtain  $mr$  values

$$W_{1,1}(T), W_{1,2}(T), \dots, W_{1,r}(T),$$

$$W_{2,1}(T), W_{2,2}(T), \dots, W_{2,r}(T),$$

...

$$W_{m,1}(T), W_{m,2}(T), \dots, W_{m,r}(T).$$

For  $T = 1000$  and  $p = 0.1$  we divide the set of possible values of  $W(T)$  into  $N = 12$  intervals

$$\Delta_1 = \{0, \dots, 100\}, \Delta_2 = \{101\}, \Delta_3 = \{102\}, \Delta_4 = \{103\},$$

$$\Delta_5 = \{104\}, \Delta_6 = \{105\}, \Delta_7 = \{106\}, \Delta_8 = \{107\},$$

$$\Delta_9 = \{108\}, \Delta_{10} = \{109\}, \Delta_{11} = \{110\}, \Delta_{12} = \{111, 112, \dots\},$$

so, according to previously computed distribution of  $W(T)$ ,

$$p_1^0 = 0.113825, p_2^0 = 0.0473614, p_3^0 = 0.0592027, p_4^0 = 0.0706947,$$

$$p_5^0 = 0.0805426, p_6^0 = 0.0874356, p_7^0 = 0.0903182, p_8^0 = 0.0886453,$$

$$p_9^0 = 0.0825413, p_{10}^0 = 0.072801, p_{11}^0 = 0.0607218, p_{12}^0 = 0.145911.$$

In other aspects construction of chi-square criterion does not depend of values of  $T$  and  $p$ .

**Remark 1.** Probabilities  $p_1^0, \dots, p_{12}^0$  are presented here with the accuracy up to 6-th digit and the sum of these twelve values is equal to 1.0000006. During the computation these values were calculated with higher accuracy and for calculated values

$$1 - p_1^0 - p_2^0 - p_3^0 - \dots - p_{12}^0 = 5.42101 \cdot 10^{-20}.$$



For any  $k = 1, \dots, m$  and corresponding  $r$  values

$$W_{k,1}(T), W_{k,2}(T), \dots, W_{k,r}(T)$$

we compute values  $v_{k,1}(T), v_{k,2}(T), \dots, v_{k,12}(T)$  which are equal to amounts of values  $W_{k,i}(T)$  that turned out to be in one of the twelve intervals  $\Delta_j$ , i.e.

$$v_{k,j}(T) = |\{i = 1, \dots, r : W_{k,i}(T) \in \Delta_j\}|, \quad j = 1, \dots, 12.$$

We compute statistics

$$\chi_k^2(rT) = \sum_{j=1}^{12} \frac{(v_{k,j}(T) - np_j^0)^2}{np_j^0}, \quad k = 1, \dots, m,$$

and the final statistic

$$\tilde{\chi}^2(mrT) = \max_{1 \leq k \leq m} \chi_k^2(rT).$$

For a given significance level  $\alpha > 0$  we calculate the quantile  $C(11, \alpha^{1/m})$  and define the criterion by the rules

$$\begin{aligned} \{\tilde{\chi}^2(mrT) < C(N-1, \alpha^{1/m})\} &\Rightarrow H_p, \\ \{\tilde{\chi}^2(mrT) \geq C(N-1, \alpha^{1/m})\} &\Rightarrow \bar{H}_p, \end{aligned}$$

where  $N = 12$  and  $\bar{H}_p$  is the full alternative for main hypothesis  $H_p$ .

The form of the final statistic  $\tilde{\chi}^2(mrT)$  as maximum of values of partial statistics  $\chi_k^2(rT)$  gives us simplicity of its limit distribution.

**Theorem 2.** *Let the hypothesis  $H_p$  be true, so random variables  $X_1, \dots, X_{mrT}$  are independent and equiprobably distributed on  $\{0, 1\}$  with probabilities  $\mathbf{P}\{X_i = 1\} = p$ ,  $\mathbf{P}\{X_i = 0\} = 1 - p$ , where  $p \in (0, 1)$ . If parameters  $m$ ,  $T$  and  $N$  are fixed and  $r \rightarrow \infty$ , then*

$$\mathbf{P}\{\tilde{\chi}^2(mrT) < x\} \rightarrow 1 - (1 - \chi_{N-1}^2(x))^m, \quad x \in (-\infty, +\infty) \quad (6)$$

$$\mathbf{P}\{\tilde{\chi}^2(mrT) \geq C(N-1, \alpha^{1/m})\} \rightarrow \alpha. \quad (7)$$

There  $\chi_{N-1}^2(x)$  is the distribution function of the chi-square distribution with  $N-1$  degrees of freedom and  $C(N-1, \alpha)$  is the function of  $\alpha$  defined in (5).

So, if the hypothesis  $H_p$  is true and the size  $mrT$  of a sample increases, then the probability to reject  $H_p$  tends to  $\alpha$ .

The choice of parameter  $N$  depends on the distribution of random variable  $W(T)$  and this distribution changes as the parameter  $T$  grows. For a given  $T$

we should sort out from the support of  $W(T)$  such natural numbers that the probability that  $W(T)$  is equal to any of these numbers is significant. The set of such numbers (and the rest numbers in the support) may be divided into intervals  $\Delta_1, \dots, \Delta_N$  according to common principles of construction of chi-square criteria. If value  $T$  increases, then the number of values such that random variable  $W(T)$  is equal to this value with significant probability also increases.

## 4 On computation of the probabilities of distribution $W(T)$

Lempel-Ziv algorithm sequentially composes a dictionary of words of 0 and 1. Denote by  $S(n)$  the cumulative length of all words in a dictionary of  $n$  words, then the following events are equal:

$$\{W(T) < n\} = \{S(n) > T\}.$$

Via this observation the computation of distribution of  $W(T)$  may be implemented by significantly simpler computation of distribution of  $S(n)$ . Formulae for distributions of  $S(n)$  are presented in the next theorem.

**Theorem 3.** *Let for  $X_1, X_2, \dots$  the hypothesis  $H_p$  is true. Then*

$$\begin{aligned} \mathbf{P}\{S(n+1) = n+r\} &= \\ &= \sum_{k=0}^n (1-p)^k p^{n-k} C_n^k \sum_{l=0}^r \mathbf{P}\{S(k) = l\} \mathbf{P}\{S(n-k) = r-l\} \end{aligned}$$

for any  $r = 0, 1, \dots, n(n-1)/2$ .

Initial and boundary values of probabilities  $\mathbf{P}\{S(n+1) = r\}$  are defined by equalities

$$\begin{aligned} \mathbf{P}\{S(0) = 0\} &= 1, \mathbf{P}\{S(1) = 0\} = 1, \mathbf{P}\{S(2) = 1\} = 1, \\ \mathbf{P}\{S(n+1) = r\} &= 0, \quad r = 0, \dots, n-1. \end{aligned}$$

Proofs of these statements were given in [4].

So we may compute distributions of random variables  $S(n)$  and via these distributions easily compute the distribution of random variable  $W(T)$  by simple formula

$$\mathbf{P}\{W(T) = n\} = \sum_{k=0}^T (\mathbf{P}\{S(n) = k\} - \mathbf{P}\{S(n+1) = k\}).$$

## References

- [1] Rukhin A. et al., *NIST Special Publication 800-22. A statistical test suite for random and pseudorandom number generators for cryptographic applications*, 2000.
- [2] Rukhin A. et al., *NIST Special Publication 800-22r1a. A statistical test suite for random and pseudorandom number generators for cryptographic applications*, 2010.
- [3] Doganaksoy A., Gologlu F., “On Lempel-Ziv Complexity of Sequences”, *LNCS, Sequences and Their Applications – SETA 2006*, **4086**, ed. Gong G., Hellesteth T., Song HY., Yang K., Springer, Berlin, Heidelberg, 2006.
- [4] Mihailov V.G., “Formulae to Calculate Distributions of Lempel-Ziv Statistic and Relative Statistics”, *OPPM Surveys in Applied and Industrial Mathematics*, **14**:3 (2007), 461–473.
- [5] Tyurin I.S., “An improvement of the residual in the Lyapunov theorem”, *Teor. Veroyatnost. i Primenen.*, **56**:4 (2011), 808–811.
- [6] Ivchenko G.I., Medvedev Y.I., *Mathematical Statistics*, High School, Moscow, 1984, In Russian.
- [7] Ryabko B.Ya., Monarev V.A., “Using information theory approach to randomness testing”, *Journal of Statistical Planning and Inference*, **133**:1 (2003), 95–110.

## Appendix

Table 1 (in 2 parts). Probabilities of distributions of random variables  $W(T)$  for  $T = 1000$  and  $p = 0.1, 0.5, 0.9$ . Probabilities that are not presented in tables are smaller than 0.0001.

$$T = 1000, p = 0.5$$

$n$	$\mathbf{P}\{W(T) = n\}$		
169	0.0007489	173	0.4099
170	0.00899	174	0.2361
171	0.06482	175	0.03306
172	0.2457	176	0.0006201

$T = 1000, p = 0.1$  or  $p = 0.9$ .

$n$	$\mathbf{P}\{W(T) = n\}$	105	0.08744
89	0.0001293	106	0.09032
90	0.0002615	107	0.08865
91	0.0005096	108	0.08254
92	0.0009567	109	0.0728
93	0.00173	110	0.06072
94	0.00301	111	0.04782
95	0.00504	112	0.03549
96	0.008113	113	0.02478
97	0.01255	114	0.01625
98	0.01863	115	0.009986
99	0.02654	116	0.005744
100	0.03624	117	0.003086
101	0.04736	118	0.001545
102	0.05920	119	0.0007202
103	0.07069	120	0.0003117
104	0.08054	121	0.000125

Table 2. Values of  $\mathbf{E}W(T)$ ,  $\mathbf{D}W(T)$ ,  $\sigma(W(T)) = \sqrt{\mathbf{D}W(T)}$  and  $\mathbf{E}(W(T))^3$  for  $T = 1000, 2000, \dots, 6000$  and  $p = 0.1, 0.2, \dots, 0.9$ .

$p$	$1 - p$	$T$	$\mathbf{E}W(T)$	$\mathbf{D}W(T)$	$\sigma(W(T))$	$\mathbf{E}(W(T))^3$
0.9	0.1	1000	105.863	19.5466	4.42116	138.162
0.9	0.1	2000	181.480	31.4802	5.61073	282.414
0.9	0.1	3000	250.429	41.8801	6.47148	433.066
0.9	0.1	4000	315.589	51.5959	7.18303	591.918
0.9	0.1	5000	378.173	60.8781	7.80244	758.405
0.9	0.1	6000	438.839	69.8330	8.35662	931.577
0.8	0.2	1000	139.086	11.5378	3.39673	62.7145
0.8	0.2	2000	242.194	19.0237	4.36162	132.600
0.8	0.2	3000	337.030	25.6001	5.05966	206.903
0.8	0.2	4000	427.119	31.6750	5.62805	284.689
0.8	0.2	5000	513.950	37.4138	6.11668	365.409
0.8	0.2	6000	598.333	42.9038	6.55010	448.675
0.7	0.3	1000	158.736	5.71317	2.39022	21.9139
0.7	0.3	2000	278.223	9.30922	3.05110	45.4569
0.7	0.3	3000	388.475	12.4729	3.53171	70.4316
0.7	0.3	4000	493.409	15.3950	3.92364	96.5354
0.7	0.3	5000	594.683	18.1539	4.26073	123.583
0.7	0.3	6000	693.202	20.7917	4.55979	151.452
0.6	0.4	1000	169.466	2.15506	1.46801	5.10445
0.6	0.4	2000	297.917	3.34503	1.82894	9.83187
0.6	0.4	3000	416.617	4.37681	2.09208	14.6800
0.6	0.4	4000	529.691	5.32151	2.30684	19.6638
0.6	0.4	5000	638.888	6.20801	2.49159	24.7670
0.6	0.4	6000	745.165	7.05167	2.65550	29.9740
0.5	0.5	1000	172.899	0.96268	0.98116	1.53175
0.5	0.5	2000	304.220	1.34154	1.15825	2.49722
0.5	0.5	3000	425.627	1.65301	1.28569	3.40601
0.5	0.5	4000	541.309	1.92859	1.38874	4.29211
0.5	0.5	5000	653.046	2.18096	1.47681	5.16860
0.5	0.5	6000	761.811	2.41656	1.55453	6.02637

Table 3. Values  $\mathbf{E}|V(2T)|$ ,  $\mathbf{D}V(2T)$ ,  $\sigma(V(2T)) = \sqrt{\mathbf{D}V(2T)}$  and  $\mathbf{E}|V(2T)|^3$  for  $T = 1000, 2000, \dots, 6000$  and  $p = 0.1, 0.2, \dots, 0.9$ .

$p$	$1 - p$	$T$	$\mathbf{E} V(2T) $	$\mathbf{D}V(2T)$	$\sigma(V(2T))$	$\mathbf{E} V(2T) ^3$
0.9	0.1	1000	4.97597	39.0932	6.25246	390.533
0.9	0.1	2000	6.32021	62.9605	7.93476	798.119
0.9	0.1	3000	7.29320	83.7601	9.15206	1224.19
0.9	0.1	4000	8.09734	103.192	10.1583	1673.58
0.9	0.1	5000	8.79717	121.756	11.0343	2144.59
0.9	0.1	6000	9.42315	139.666	11.8180	2634.50
0.8	0.2	1000	3.81661	23.0756	4.80370	177.208
0.8	0.2	2000	4.90922	38.0475	6.16826	374.857
0.8	0.2	3000	5.69870	51.2003	7.15544	585.000
0.8	0.2	4000	6.34121	63.3499	7.95927	805.004
0.8	0.2	5000	6.89338	74.8276	8.65029	1033.31
0.8	0.2	6000	7.38305	85.8076	9.26324	1268.82
0.7	0.3	1000	2.67397	11.4263	3.38029	61.8622
0.7	0.3	2000	3.42516	18.6184	4.31491	128.442
0.7	0.3	3000	3.97006	24.9459	4.99459	199.080
0.7	0.3	4000	4.41393	30.7900	5.54887	272.906
0.7	0.3	5000	4.79543	36.3077	6.02559	349.400
0.7	0.3	6000	5.13374	41.5835	6.44852	428.213
0.6	0.4	1000	1.61960	4.31011	2.07608	14.3913
0.6	0.4	2000	2.03477	6.69007	2.58652	27.7349
0.6	0.4	3000	2.33560	8.75362	2.95865	41.4588
0.6	0.4	4000	2.58041	10.6430	3.26236	55.5470
0.6	0.4	5000	2.79065	12.4160	3.52364	69.9666
0.5	0.5	1000	1.05493	1.92537	1.38758	4.29894
0.5	0.5	2000	1.26349	2.68309	1.63801	7.04866
0.5	0.5	3000	1.41194	3.30601	1.81824	9.62922
0.5	0.5	4000	1.53127	3.85718	1.96397	12.1292
0.5	0.5	5000	1.63290	4.36191	2.08852	14.5852
0.5	0.5	6000	1.72236	4.83312	2.19844	17.0151

Table 4 (in 2 parts). Probabilities of distributions of random variables  $\tilde{W}(2mT)$  for  $T = 1000$ ,  $p = 0.5$ ,  $m = 1$  and  $m = 10$ .

$T = 1000, p = 0.5, m = 1$

$n$	$\mathbf{P}\{\tilde{W}(2mT) = n\}$	0	0.2895
-5	0.0005318	1	0.2218
-4	0.004736	2	0.1005
-3	0.02756	3	0.02756
-2	0.1005	4	0.004736
-1	0.2218	5	0.0005318

$T = 1000, p = 0.5, m = 10$

$n$	$\mathbf{P}\{\tilde{W}(2mT) = n\}$	0	0.09098
-16	0.0001207	1	0.08864
-15	0.000268	2	0.08198
-14	0.0005659	3	0.07198
-13	0.001136	4	0.05999
-12	0.002168	5	0.04747
-11	0.003931	6	0.03566
-10	0.006772	7	0.02543
-9	0.01108	8	0.01723
-8	0.01723	9	0.01108
-7	0.02543	10	0.006772
-6	0.03566	11	0.003931
-5	0.04747	12	0.002168
-4	0.05999	13	0.001136
-3	0.07198	14	0.0005659
-2	0.08198	15	0.000268
-1	0.08864	16	0.0001207

Table 5. Values of the right part of inequality (1) for  $T = 1000, 2000, \dots, 6000$ ,  $p = 0.1, 0.2, \dots, 0.9$  and  $m = 1000, 2000$ .

$p$	$1 - p$	$T$	$m = 1000$	$m = 2000$
0.9	0.1	1000	2.41206e-005	8.52792e-006
0.9	0.1	2000	2.41184e-005	8.52713e-006
0.9	0.1	3000	2.41088e-005	8.52375e-006
0.9	0.1	4000	2.41024e-005	8.52149e-006
0.9	0.1	5000	2.40985e-005	8.52009e-006
0.9	0.1	6000	2.40960e-005	8.51922e-006
0.8	0.2	1000	2.41344e-005	8.53281e-006
0.8	0.2	2000	2.41135e-005	8.52540e-006
0.8	0.2	3000	2.41063e-005	8.52286e-006
0.8	0.2	4000	2.41025e-005	8.52151e-006
0.8	0.2	5000	2.41002e-005	8.52071e-006
0.8	0.2	6000	2.40988e-005	8.52020e-006
0.7	0.3	1000	2.41795e-005	8.54874e-006
0.7	0.3	2000	2.41366e-005	8.53358e-006
0.7	0.3	3000	2.41219e-005	8.52839e-006
0.7	0.3	4000	2.41147e-005	8.52582e-006
0.7	0.3	5000	2.41105e-005	8.52435e-006
0.7	0.3	6000	2.41080e-005	8.52347e-006
0.6	0.4	1000	2.42801e-005	8.58431e-006
0.6	0.4	2000	2.41971e-005	8.55497e-006
0.6	0.4	3000	2.41667e-005	8.54422e-006
0.6	0.4	4000	2.41516e-005	8.53888e-006
0.6	0.4	5000	2.41435e-005	8.53600e-006
0.6	0.4	6000	2.41393e-005	8.53455e-006
0.5	0.5	1000	2.42924e-005	8.58868e-006
0.5	0.5	2000	2.42123e-005	8.56035e-006
0.5	0.5	3000	2.41834e-005	8.55011e-006
0.5	0.5	4000	2.41720e-005	8.54608e-006
0.5	0.5	5000	2.41702e-005	8.54547e-006
0.5	0.5	6000	2.41755e-005	8.54733e-006



# Experimental study of NIST Statistical Test Suite ability to detect long repetitions in binary sequences

Andrei Zubkov and Aleksandr Serov

Steklov Mathematical Institute of Russian Academy of Sciences, Moscow, Russia

zubkov@mi-ras.ru, serov@mi-ras.ru

## Abstract

We present and discuss the results of empirical testing the ability of the NIST Statistical Test Suite to detect very long repetitions in binary sequences. We construct the set of deterministic binary sequences that are not rejected by the NIST Suite and corrupt all of them in a deterministic way. To corrupt a binary sequence we choose several its substrings of fixed length and insert in this sequence the copy of each substring. The length of substrings was chosen to be significantly larger than the typical length of the longest repeated substring. If the number of repeated substrings in the corrupted sequence is moderate then the NIST Suite does not reject such explicit nonrandom binary sequences. We describe the algorithm of searching for the longest repetition of substrings in a binary sequence of length  $n$ . This algorithm is based on the suffix tree and its time and space complexities are of the order  $O(n)$ .

The examples of cryptographically weak sequences passing the standard statistical tests may stimulate the construction of new batteries of tests for random number generators.

**Keywords:** statistical testing, binary sequence, corrupted sequence, randomness, equiprobability, longest repeated substring.

## 1 Introduction

The problem of testing output binary sequences of Random Number Generators (RNG) is very important for cryptography (as well as for other applications of RNG). The aim of such testing is to decide whether RNG generates sequences which may be considered as realizations of equiprobable Bernoulli sequence and therefore may be used in cryptosystems. Without such testing a bad RNG outputs may destroy safety properties of cryptosystems.

The quality checking of binary sequences usually is based on some well-known batteries of tests, for example, NIST [3], TestU01 [4] etc. There are large intersections of types of tests included in different batteries. Any battery of tests consist of several statistical tests, all tests of the battery are

applied to each tested sequence. Of course, there are many properties of binary output sequences of RNGs which are undesirable for cryptographic applications, and it is hardly possible to construct a battery of tests detecting each such property. But it is reasonable to collect examples of sequences having such properties and use them to estimate the completeness of batteries. For example, in [5, 6] we show that simple combinations of pairs of binary linear recurrent sequences may be accepted as good by all tests of the NIST statistical package with high probability.

During the generation of the RNG sequence various failures may occur such as software and hardware faults, transmission noise and interference, program bugs etc. Some faults may result in repetition of long substrings in generated sequences, which is obviously bad property from a cryptographic point of view. It is interesting to assess the ability of the NIST battery of tests to reject binary sequences with substring repetitions having lengths significantly larger than the typical length of string repetition in the random equiprobable Bernoulli sequence.

## 2 Short description of NIST Statistical Test Suite

The full version of NIST Statistical Test Suite consists of 15 tests. The result of the execution of each test is the value of one or more statistics. To make a decision the values of statistics are transformed into  $p$ -values (the  $p$ -value is the probability to obtain the value of statistics at least as large as the actually observed one, under the assumption that the null hypothesis  $H_0$  on the randomness and equiprobability of binary sequences is true). The number of all  $p$ -values generated by the full version of test suite is 188. The Non-overlapping Template Matching Test, Random Excursions Test, Random Excursions Variant Test, Serial and Cumulative Sums Tests generate 148, 8, 18, 2 and 2  $p$ -values correspondingly, other tests generate one  $p$ -value each. We have not taken into account the results of Non-overlapping Template Matching Test (the number of  $p$ -values is too large and this test is not sensitive to a moderate number of repetitions in a sequence) and the results of two tests connected with Random Excursions (these tests do not generate any  $p$ -values for large part of tested sequences, detailed explanation see in [6]).

So, we have considered results of 12 tests generating 14  $p$ -values, see Table 1.

#	Name of Test	Used Statistics
1	<b>Frequency</b>	normed difference between frequencies of 1 and 0 in the sequence
2	<b>Block Frequency</b>	relative frequencies of ones in adjacent nonintersecting 128-bit blocks of the sequence
3	<b>Cumulative Sums</b>	maximal deviation from 0 for $\pm 1$ walk constructed by a segment of binary sequence, results in 2 statistics
4	<b>Runs</b>	the total number of 1-runs and 0-runs in the sequence
5	<b>Longest Run</b>	maximal lengths of 1-runs in adjacent nonintersecting $10^4$ -bit blocks of the sequence
6	<b>Binary Matrix Rank</b>	ranks of binary $32 \times 32$ -matrices formed from adjacent nonintersecting 1024-bit blocks of the sequence
7	<b>Discrete Fourier Transform</b>	the number of coefficients of discrete Fourier transform of the $n$ -bit sequence exceeding $h = \sqrt{n \log \frac{1}{0.05}}$ in absolute value
8	<b>Overlapping Template Matching</b>	numbers of 9-bit intersecting 1-series in 1032-bit adjacent blocks of the sequence
9	<b>"Universal statistical test"</b>	sum of base 2 logarithms of distances between equal nonintersecting 7-bit blocks of $n$ -bit sequence
10	<b>Approximate Entropy</b>	frequencies of intersecting 10- and 11-bit blocks in the sequence
11	<b>Serial</b>	frequencies of intersecting 16-, 15- and 14-bit blocks in the sequence, results in 2 statistics
12	<b>Linear Complexity</b>	lengths of shortest linear shift registers generating adjacent nonintersecting 500-bit blocks of the sequence

Table 1: List of NIST Statistical Tests considered

**Note.** *The decision made by each test depends on the closeness of  $p$ -value to 0. It should be noted that we didn't take into account the specific values of  $p$ -values at which hypothesis  $H_0$  is accepted or rejected on the base of all tests. That is, as it happens in practice, in our paper we are interested in a specific decisions taken by each individual test.*

### 3 Description of our experiments

We have applied 12 tests from Table 1 to pseudorandom sequences of two types:

- a) sequences generated by a block cipher,
- b) the same sequences corrupted by repetitions of blocks.

The first type sequences were obtained by means of AES block cipher in the CFB (Cipher Feedback Block) mode (see Fig.1) with the zero initialization vector (IV) and plaintext block (Plaintext), the key for each sequence was

randomly selected from the set of 128-bit binary sequences, the length in bit of all sequences was chosen to be the same and equal to  $n = 2^{20} = 1.048.576$  each (value corresponds to the NIST input size recommendation for the length of tested sequences).

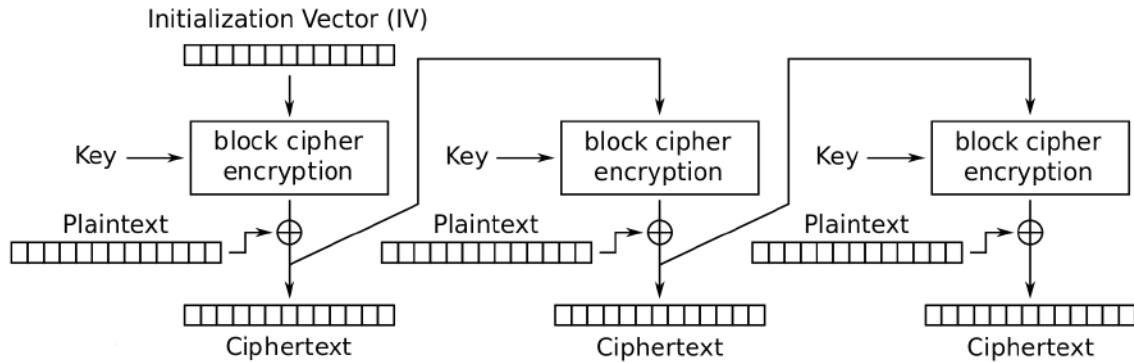


Figure 1:

The second type sequences were obtained by corruption the sequences of the first type. The process of corruption was as follows:

- generate list  $L = \{a_i\}_{i=1}^{2S}$  of  $2 \cdot S$  different random numbers in the range  $a_i \in \{1, 2, \dots, n - T + 1\}$ , where  $n$  is the length of the tested sequence,  $S$  is the number of repeated substrings of length  $T$ , then divide sorted or unsorted list  $L$  into pairs  $(a_{2i-1}, a_{2i})$ ,  $i = 1, \dots, S$ ,

- corrupte each initial sequence of the first type by replacing block of bits at positions  $a_{2i}, a_{2i} + 1, \dots, a_{2i} + T - 1$  with block of bits at positions  $a_{2i-1}, a_{2i-1} + 1, \dots, a_{2i-1} + T - 1$  for each  $i = 1, 2, \dots, S$ .

We have tested sequences with different parameters of corruptions: the values of list sizes  $S$  were 1, 10, 25, 50, 70, 100, 200, 400, 500; the number of different random keys is  $2^{14} = 16384$ , the values of repeated substring lengths  $T$  were 40, 50, 100, 400.

Keys (on Fig.1) and lists  $L$  of random numbers were obtained using the open source cryptographic library OpenSSL:

```
int RAND_bytes(unsigned char *buf, int num);
```

this function generates **num** bytes using the cryptographically secure generator of pseudorandom numbers (CSPRNG), and saves them in the **buf** array. By default, the CSPRNG generator provides a security level of 256 bits when initialized from a trusted entropy source.

Fig. 2 shows the first 256 output bytes of the `RAND_bytes` function, from which the first 16 keys were constructed.

```

7C 34 7E 21 19 1C A4 DC 25 D2 73 5F 40 29 02 4E
4E 4D B8 16 DF 76 42 94 25 9C 54 23 2E A2 7C C9
4B 9F 8F B0 4B A8 11 19 D6 F4 26 80 42 2A E4 4C
6C A3 F9 C2 28 10 55 70 03 66 BA 71 0A 44 45 B0
6C 3A 66 B7 00 BD 37 E9 87 53 FA 70 09 6B C9 4E
05 58 9D 52 9D FF 39 EF E2 32 6A 76 3A FC B4 B2
94 70 76 2F E2 5F D3 2D D1 43 11 D8 4D 7B AB 36
D2 C0 CE C3 18 BA F8 82 1E 5A 01 91 77 83 D8 11
E3 86 8E 87 60 5B EF 9C 1C B8 33 B4 FA 87 18 B7
29 D1 BE BD 84 19 F2 14 F8 74 E4 66 F1 E3 01 AC
EF 92 55 D2 FB 9D 1B D5 76 4E 78 8F 30 24 70 49
40 DD 97 93 54 23 2C 9A C2 B4 68 82 B4 6E 55 CA
02 07 C1 19 D7 9A 1F B6 A0 1A 60 EE BF C5 AA BB
2E 8C 61 BB 4B DF C0 6F CC CB A4 C9 EB 9E 08 A2
FC 2F 26 57 F5 81 0D F3 E6 F7 79 08 76 7E 62 E2
D1 22 F3 C6 91 16 3E 79 DD 1B 0B CF 17 6B 90 2B
    
```

Figure 2:

As an example, below is a sorted list  $L$  of 50 numbers used to corrupt sequence of length  $n = 1.048.576$  by repetitions of blocks of length 40:

```

6489 59901 69008 75675 94503 102388 105642 106546
116832 130615 145061 169698 183628 191460 225708 262244
276073 280945 299171 301080 382641 386106 410347 453254
454345 469001 548554 581574 604659 604697 646954 685538
689773 708682 738990 758877 761486 762812 777841 783019
804029 845813 877750 886345 922985 926170 954648 976396
982654 998914
    
```

Each sequence was tested by all 15 tests from the NIST Test Suite, but, as was mentioned above, the results of three tests not included in Table 1 were not taken into account. The application of 12 tests to each segment results in 14 different  $p$ -values (the tests with number 3 and 11 from Table 1 results in two  $p$ -values). The critical level of  $p$ -value was chosen as 0.01: if the  $p$ -value does not exceed 0.01, then test rejects the hypothesis  $H_0$  on the randomness and equiprobability for tested sequence. For each sequence we save the numbers of concrete tests rejecting the hypotheses  $H_0$ .

Table 2 contains the results of our experiments: the numbers of rejected sequences from 16384 sequences by the statistical tests from Table 1 with critical level of  $p$ -value is 0.01 (the numbers from the first column correspond to the test numbers from the Table 1).

№	$\frac{S}{T}$	10	25	50	70	1	50	50	100	100	200	400
		40	40	40	40	400	50	100	50	100	100	100
1	157	155	164	167	155	149	160	167	168	190	185	<b>218</b>
2	171	172	180	180	172	174	170	174	171	172	<b>208</b>	<b>263</b>
3	149	155	160	151	150	152	151	163	153	179	177	<b>215</b>
	162	159	160	162	168	161	167	163	169	187	185	<b>217</b>
4	156	153	155	153	158	155	161	157	160	160	179	<b>209</b>
5	189	186	191	188	191	190	193	191	183	190	194	<b>203</b>
6	149	152	158	138	145	148	153	167	138	<b>203</b>	<b>529</b>	<b>3682</b>
7	190	194	177	178	<b>203</b>	189	193	196	189	180	198	<b>216</b>
8	179	186	174	188	178	183	170	181	164	189	188	168
9	191	195	190	186	182	190	195	197	192	199	<b>205</b>	<b>311</b>
10	184	191	201	<b>234</b>	<b>223</b>	196	<b>228</b>	<b>286</b>	<b>287</b>	<b>479</b>	<b>1021</b>	<b>3732</b>
11	168	197	<b>255</b>	<b>382</b>	<b>496</b>	<b>220</b>	<b>500</b>	<b>1543</b>	<b>1218</b>	<b>6680</b>	<b>15755</b>	<b>16384</b>
	150	162	<b>202</b>	<b>269</b>	<b>332</b>	180	<b>342</b>	<b>899</b>	<b>685</b>	<b>3251</b>	<b>11485</b>	<b>16381</b>
12	189	188	183	186	171	182	182	182	187	170	164	179

Table 2: The numbers of rejected sequences for each test, two numbers in a cell correspond to two test statistics, the second column of the table corresponds to the noncorrupted sequences, the number of tested sequences is  $2^{14}$ ,  $S$  is the number of repetitions,  $T$  is the length of repetition

If the null hypothesis  $H_0$  is true then the distributions of  $p$ -values may be approximated by the uniform distribution on  $[0, 1]$ . The number of rejected sequences by any test in this case should have the Binomial distribution with parameters  $(N, p) = (16384, 0.01)$  ( $N = 16384$  is the number of trials,  $p = 0.01$  is the critical level of  $p$ -values) and expectation and standard deviation respectively  $Np = 163.84$  and  $\sqrt{Np(1-p)} \approx 12.74$ . The probability that the value of such random variable belongs to the set  $\{126, 127, \dots, 201\}$  is equal to 0.9962. Values outside this set are marked by boldface in Table 2. Note that, for example, values about 500 means that corresponding test rejects  $H_0$  approximately in one case of 32.

Table 2 shows that the NIST Test Suite accepts as random and equiprobable the binary sequences with a moderate number of repetitions of small length or with a small number of repetitions of large length.

In view of this results we propose to use the test based on the longest repeated substring (LRS). The algorithm for computing the longest repeated substring uses the well-known tree-like data structure representing all suffixes of an arbitrary string. Such data structures are called *suffix trees* and, in particular, allow to find the longest repeated substrings of the string in time linear in the length of the string. The method proposed by E. Ukkonen in [2] is developed as a linear-time version of simple algorithm (having quadratic complexity) for suffix tries. The complexity of construction a suffix tree for

a binary sequence of the length  $n$  is  $O(n)$  operations and the same storage space, the complexity of finding the longest substring or all the longest substrings by means of the constructed suffix tree is equal to  $O(n)$  also.

We could get the  $p$ -values for this new test from the limit theorem 3 of [1]: let  $X_1, X_2, \dots, X_n, \dots$  be the sequence of independent identically distributed random variables taking values  $1, 2, \dots, m$  and

$$\mathbf{P}\{X_i = k\} = p_k, \quad \sum_{k=1}^m p_k^2 = P,$$

if  $\mu(n)$  is the length of the longest substring in the random sequence of length  $n$ , then

$$\mathbf{P}\{\mu(n) \geq \tau\} \approx 1 - \exp\left(-\frac{1-P}{2} P^{\tau - \frac{2 \ln n}{|\ln P|}}\right).$$

If the sequence  $\{X_i\}_{i=1}^n$  is equiprobable Bernoulli sequence, then  $P$  is equal to  $\frac{1}{2}$  and we define  $p$ -value for outcome  $\tau$  as

$$\mathbf{P}\{\mu(n) \geq \tau\} \approx 1 - \exp\left(-\left(\frac{1}{2}\right)^{\tau + 2 - \frac{2 \ln n}{|\ln 2|}}\right). \quad (1)$$

We have tested all sequences of the first type with this LRS test; the test results are included in Table 3. The first and the fourth rows of the Table contain the lengths  $\tau$  of the longest repeated substrings, the second and the fifth rows contain the numbers of corresponding sequences, the third and sixth rows contain the average numbers of longest repeated substrings in the sequence under the condition that the longest repeated substring has length  $\tau$ .

$\tau$	34	35	36	37	38	39	40	41	42	43
#	9	295	1989	3720	3930	2797	1701	963	444	262
##	8.44	3.98	2.35	1.57	1.27	1.12	1.07	1.03	1.01	1.02
$\tau$	44	45	46	47	48	49	50	51	53	
#	139	63	36	12	14	4	4	1	1	
##	1	1	1	1	1	1	1	1	1	

Table 3: Numbers of sequences out of 16384 with longest repeated substring of length  $\tau$

If we use (1) as the  $p$ -value distribution law for LRS test, then for example

$$\begin{aligned} \mathbf{P}\{\mu(n) \geq 44\} &\approx 0.0155, & \mathbf{P}\{\mu(n) \geq 45\} &\approx 0.0078, \\ 2^{14} \cdot \mathbf{P}\{\mu(n) \geq 45\} &\approx 127.5. \end{aligned}$$

In our experiment the number of sequences out of 16384 with the length of the longest repeated substring at least 45 is equal to 135.

## 4 Conclusion

It may be concluded that tests included in the NIST Test Suite may fail to reject binary sequences with a moderate amount of repetitions of lengths which are significantly larger than the expected length of the longest repeated substring in the sequence. We propose the test for the longest repeated substring with linear time and space complexities.

## References

- [1] Zubkov A. M., Mikhailov V. G., “Limit distributions of random variables associated with long duplications in a sequence of independent trials”, *Theory Probab. Appl.*, **19**:1 (1974), 172–179.
- [2] Ukkonen E., “On-line construction of suffix trees”, *Algorithmica*, **14** (1995), 249–260.
- [3] Rukhin A., Soto J., Nechvatal J., Smid M., Barker E., Leigh S., Levenson M., Vangel M., Banks D., Heckert A., Dray J., Vo S., “A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications”, NIST Special Publication 800-22 Revision 1a, 27 April 2010.
- [4] L’Ecuyer P., Simard R., “TestU01”, 2013, 214 pp., <http://simul.iro.umontreal.ca/testu01/guideshorttestu01.pdf>.
- [5] Zubkov A. M., Serov A. A., “Testing the NIST Statistical Test Suite on artificial pseudo-random sequences”, *Matematicheskie Voprosy Kriptografii*, **10**:2 (2019), 89–96.
- [6] Zubkov A. M., Serov A. A., “A natural approach to the experimental study of dependence between statistical tests”, *Матем. вопр. криптогр.*, **12**:1 (2021), 131–142.



# ALGEBRAIC ASPECTS

# On the question of nonlinearity of vectorial functions over finite fields

Vladimir Ryabov

NP "GST", Moscow, Russia  
4vryabov@gmail.com

## Abstract

The nonlinearity of a vectorial function is defined as the Hamming distance to a set of affine mappings. Connections between the parameters characterizing the closeness of a vectorial function to affine mappings and the analogous parameters of its coordinate functions are established. Boundaries and estimates for the distribution of nonlinearity of arbitrary and balanced vectorial functions, as well as permutations, are obtained. The nonlinearity defined here is compared with the nonlinearity specified as minimal among nonlinearities of all nontrivial linear combinations of coordinate functions. Classes of mappings with a high nonlinearity are constructed.

**Keywords:** vectorial function, balanced mapping, permutation, Hamming distance, nonlinearity, probability distribution.

## 1 Introduction

Let  $\mathbf{F}_q$  denote a finite field of  $q$  elements, where  $q = p^m$ ,  $p$  is a prime number,  $m$  is a natural number, and  $\mathbf{F}_q^n$  is an  $n$ -dimensional vector space over the given field, where  $n$  is a natural number. Denote by  $P_q^{n,k}$  the set of all mappings of the space  $\mathbf{F}_q^n$  into the space  $\mathbf{F}_q^k$ . In what follows, the mapping  $F \in P_q^{n,k}$  will be called a  $q$ -valued vectorial function, and in the case  $k = 1$  we will use similar terms without the adjective “vectorial”.

Any vectorial function is uniquely determined by an ordered set of coordinate functions. The case of a finite field allows us to represent each coordinate function as a polynomial over the field. The algebraic degree of nonlinearity  $\mathit{deg}F$  is usually defined as the maximum of the degrees of polynomials representing its coordinate functions. Under the condition  $\mathit{deg}F \leq 1$  the mapping  $F$  is affine. If, in addition, the constant terms of all polynomials are equal to zero, the mapping  $F$  is linear. Denote by  $A_q^{n,k}$  and  $L_q^{n,k}$  the subsets of all affine and linear mappings from the set  $P_q^{n,k}$  respectively.

In the case  $k \leq n$ , a vectorial function  $F \in P_q^{n,k}$  is called balanced if it takes all values the same number of times, that is, for any  $\mathbf{y} \in \mathbf{F}_q^k$  the

condition  $|F^{-1}(\mathbf{y})| = q^{n-k}$  is satisfied. Obviously, every coordinate function of a balanced vectorial function is also balanced. Moreover, it follows from [12] that a vectorial function is balanced if and only if any nontrivial linear combination of its coordinate functions is balanced. Denote by  $B_q^{n,k}$  the subset of balanced vectorial functions from the set  $P_q^{n,k}$ . For  $k = n$  the set  $B_q^{n,n}$  coincides with the set of bijective mappings of space  $\mathbf{F}_q^n$  into itself or permutations of the space  $\mathbf{F}_q^n$ , which will be denoted by  $S_q^n$ .

The article [7] emphasizes that the problem of approximating discrete functions by linear analogs is of great importance in cryptography, where it is required to use discrete functions that are the most remote from linear functions<sup>1</sup>. In the same place, two approaches to determining the measure of closeness of any functions to linear functions are distinguished. The first approach is based on estimating the Hamming distance from a given function to a set of linear functions, the second one is based on the difference properties of the functions. The present paper is devoted to the development of the first approach in the case of  $q$ -valued vectorial functions.

In the case  $k = 1$ , the Hamming distance in the space  $\mathbf{F}_q^{q^n}$  from a function  $f \in P_q^{n,1}$  to the set  $A_q^{n,1}$  is usually called the nonlinearity of the  $q$ -valued function  $f$ . Let's denote it as  $N_f$ . Functions with the maximum value of nonlinearity among all functions of  $n$  variables are called maximally nonlinear. In the Boolean case, the set of distances from a function to all linear functions uniquely determines the nonlinearity of the function, and, as follows from the results of [16], for even values of  $n$ , the class of maximally nonlinear Boolean functions coincides with the class of Boolean bent functions. For  $q > 2$ , knowing the distances to all linear functions does not allow us to draw a conclusion regarding its nonlinearity, and even for even values of  $n$ , as it was shown in [20] (presented at CTCrypt 2021), not all  $q$ -valued bent functions are maximally nonlinear.

In the case of an arbitrary  $k$  and  $q$ , as a measure of closeness of vectorial functions, we take the Hamming distance in the space  $\mathbf{F}_q^{q^n}$  and denote the distance between the functions  $F_1$  and  $F_2$  from  $P_q^{n,k}$  by  $\rho(F_1, F_2)$ . Let's define the nonlinearity of the vectorial function  $F \in P_q^{n,k}$  by the formula

$$N_F = \min_{A \in A_q^{n,k}} \rho(F, A). \quad (1)$$

The nonlinearity  $N_F$  can be expressed as  $N_F = q^n - R_F$ , where  $R_F$  is the maximum size of the piecewise affinity region over all possible representations of  $F$  in piecewise affine form [26].

---

<sup>1</sup>In this case, linear functions on groups and quasigroups are understood as homomorphisms, while linear and affine functions on finite fields are considered in the conventional sense.

In the case of  $k > 1$ , the definition (1) differs from the popular definition introduced in [15], in which the nonlinearity of the mapping  $F \in P_q^{n,k}$  with a set of coordinate functions  $\mathbf{f} = (f_1, \dots, f_k)$  is given by the equality

$$NL_F = \min_{\mathbf{w} \in \mathbf{F}_q^k \setminus \{\mathbf{0}\}} N_{\langle \mathbf{w}, \mathbf{f} \rangle}, \quad (2)$$

where  $\langle *, * \rangle$  means the scalar product of vectors, that is, it is the minimum among the nonlinearities of all nontrivial linear combinations of coordinate functions. Although in both cases the Hamming distance is used, in (1) we are talking about the distance between the vectorial function  $F$  and the set of affine mappings  $A_q^{n,k}$  in the space  $\mathbf{F}_q^{q^n}$ , while in (2) we consider the distance between the set of all nontrivial linear combinations of coordinate functions of the mapping  $F$  and the set of affine functions  $A_q^{n,1}$  in the space  $\mathbf{F}_q^{q^n}$ .

Separately, it is worth highlighting the study of the characteristics of "nonlinearity" of mappings in the article [10], as well as the works [24, 9, 4] and subsequent papers by these authors using measures of closeness (consent) other than the Hamming distance were used. Note, that most of the results of these papers relate to the determination of the closeness of functions to homomorphisms, the class of which, in the case of a finite field that is not simple, includes the set of affine mappings, but doesn't reduce to it.

Unlike the other measures mentioned, the Hamming distance is a metric. The nonlinearity  $N_F$  vanishes only for affine mappings, and in this sense it agrees well with the algebraic degree of nonlinearity of a  $q$ -valued vectorial function. Finding affine analogues with a relatively small value of  $\rho(F, A)$  and the value of nonlinearity  $N_F$  can be used in cryptography, for example, when solving a system of nonlinear equations by methods using affine approximations. Thus, [6, 8] describes a method based on partitioning the space  $\mathbf{F}_2^n$  into  $r$  subsets on which the restrictions of  $F$  coincide with the restrictions of some affine mappings. The complexity of this method depends on the value of  $r$ , the minimum possible value of which is called the order affinity and denoted by  $\mathbf{ard}F$ . This method can be easily transferred to the  $q$ -valued case. To find partitions with a relatively small value of  $r$ , the algorithm of enumeration of affine analogs, ordered by increasing value  $\rho(F, A)$ , can be used, and the value of the nonlinearity  $N_F$  allows us to obtain a lower estimate of the order affinity of the form

$$\mathbf{ard}F \geq q^n / (q^n - N_F). \quad (3)$$

The last remarks speak in favor of studying the parameters  $\rho(F, A)$  and  $N_F$ . In what follows, unless otherwise stated, by nonlinearity we mean the nonlinearity of  $N_F$  in the sense of (1).

## 2 Properties of parameters characterizing the closeness of a vectorial function to affine mappings

For further presentation, we need one more notation for the Hamming distance to an affine mapping. In the case  $k = 1$ , any affine function  $a \in A_q^{n,1}$  can be uniquely represented by a scalar product

$$a(\mathbf{x}) = \langle \mathbf{x}^+, \mathbf{a} \rangle, \quad (4)$$

where  $\mathbf{x}^+ = (1, x_1, \dots, x_n)$ ,  $\mathbf{a} = (a_0, a_1, \dots, a_n) \in \mathbf{F}_q^{n+1}$ . Associating the function  $a$  with the vector  $\mathbf{a}$  from (4), let's denote the distance between the functions  $f \in P_q^{n,1}$  and  $a \in A_q^{n,1}$  in the space  $\mathbf{F}_q^{n+1}$  by  $\rho_f^{\mathbf{a}}$ . Then the nonlinearity formula for a  $q$ -valued function takes the form  $N_f = \min_{\mathbf{a} \in \mathbf{F}_q^{n+1}} \rho_f^{\mathbf{a}}$ .

Denote  $M_q^{s,t}$  the set of matrices over the field  $\mathbf{F}_q$  consisting of  $s$  rows and  $t$  columns. Then any affine mapping  $A \in A_q^{n,k}$  can be uniquely represented by multiplying a vector<sup>2</sup>  $\mathbf{x}^+ \in \mathbf{F}_q^{n+1}$  by a matrix  $\mathbf{A} \in M_q^{n+1,k}$  of the form

$$A(\mathbf{x}) = \mathbf{x}^+ \mathbf{A}. \quad (5)$$

Assume that a mapping  $A \in A_q^{n,k}$  have a set of coordinate functions  $\{a_1, \dots, a_k\}$ , and the matrix  $\mathbf{A}$  representing it in (5) has the form<sup>3</sup>  $\{a_{i,j}\}$  for  $i = 0, 1, \dots, n$  and  $j = 1, \dots, k$ . Then, for each coordinate function  $a_j$ , the  $j$ -th transposed column of the matrix  $\mathbf{A}$  acts as the vector  $\mathbf{a}$  in relation (4), and the ordered set of free terms of all coordinate functions coincides with the zero row of the matrix  $\mathbf{A}$ . Relation (5) can also be given by the more familiar expression  $A(\mathbf{x}) = \mathbf{x} \tilde{\mathbf{A}} \oplus \mathbf{a}_0$ , where  $\tilde{\mathbf{A}} \in M_q^{n,k}$  is a submatrix of the matrix  $\mathbf{A}$  obtained by excluding zero row  $\mathbf{a}_0$  from  $\mathbf{A}$ .

Taking into account the isomorphism of the vector space  $\mathbf{F}_q^k$  and the field  $\mathbf{F}_{q^k}$ , let's use the Hamming distance in the space  $\mathbf{F}_{q^k}^n$  and denote the distance between the vectorial functions  $F \in P_q^{n,k}$  and  $A \in A_q^{n,k}$  by  $\rho_F^{\mathbf{A}}$ , where the matrix  $\mathbf{A}$  corresponds to the mapping  $A$  in representation (5). Then the nonlinearity formula (1) takes the form

$$N_F = \min_{\mathbf{A} \in M_q^{n+1,k}} \rho_F^{\mathbf{A}}. \quad (6)$$

The nonlinearity  $N_F$  is invariant for  $\mathbf{EA}$ -equivalent mappings from  $P_q^{n,k}$ , as is the nonlinearity  $NL_F$ . Indeed, vectorial functions  $F$  and  $F'$  from  $P_q^{n,k}$  are considered to be  $\mathbf{EA}$ -equivalent if there exist bijective mappings  $B \in A_q^{n,n}$

<sup>2</sup>The symbol  $\mathbf{x}^+$  means adding a zero coordinate to the vector  $\mathbf{x}$  with the value of the field unit.

<sup>3</sup>In contrast to the conventional notation, the rows of a matrix, denoted by a latin letter without a superscript tilde, are numbered starting from zero.

and  $L \in L_q^{k,k}$ , as well as a mapping  $C \in A_q^{n,k}$ , such that the identity  $F' = L(F(B(\mathbf{x}))) \oplus C(\mathbf{x})$ . In the matrix representation, this identity has the form

$$F'(\mathbf{x}) = (F(\mathbf{x}^+ B))^+ L \oplus \mathbf{x}^+ C, \quad (7)$$

where the matrix  $B = \begin{pmatrix} b_0 \\ \tilde{B} \end{pmatrix} \in M_q^{n+1,n}$  and its submatrix  $\tilde{B} \in M_q^{n,n}$  is nonsingular, the matrix  $L = \begin{pmatrix} l_0 \\ \tilde{L} \end{pmatrix} \in M_q^{k+1,k}$  and its submatrix  $\tilde{L} \in M_q^{k,k}$  is nonsingular, and the matrix  $C = \begin{pmatrix} c_0 \\ \tilde{C} \end{pmatrix} \in M_q^{n+1,k}$ .

**Proposition 1.** *Let vectorial functions  $F$  and  $F'$  from  $P_q^{n,k}$  be **EA**-equivalent, and relation (7) is valid for them. Then the distances from these mappings to affine ones are related by the equality*

$$\rho_{F'}^{A'} = \rho_F^A, \quad (8)$$

where the matrices  $A = \begin{pmatrix} a_0 \\ \tilde{A} \end{pmatrix}$ ,  $A' = \begin{pmatrix} a'_0 \\ \tilde{A}' \end{pmatrix} \in M_q^{n+1,k}$ , the submatrix  $\tilde{A}' = \tilde{B}\tilde{A}\tilde{L} \oplus \tilde{C}$  and the zero row  $a'_0 = (a_0 \oplus b_0\tilde{A})\tilde{L} \oplus c_0$ .

The proof is similar to the proof of assertion 2 in [18].

**Corollary 1.** *For **EA**-equivalent vectorial functions from  $P_q^{n,k}$ , the unordered set  $\{\rho_F^A \mid A \in M_q^{n+1,k}\}$  and the nonlinearity  $N_F$  are invariants.*

Let us now consider the case when a vectorial function is obtained from another vectorial function by adding or eliminating several coordinate functions. Comparing the number of coincidences of the resulting vectorial function with affine functions obtained by adding or eliminating coordinate functions at the same places, we will see that when new coordinate functions are added, the nonlinearity doesn't decrease, and when some of the original ones are excluded, it doesn't increase. Adding any linear combinations of the original coordinate functions and an arbitrary affine function does not change the nonlinearity. For comparison, we note that the alternative nonlinearity of a vectorial function in the sense (2), on the contrary, doesn't increase with the addition of coordinate functions and doesn't decrease with exclusion. Adding at least one linear combination of the original coordinate functions and an arbitrary affine function nullifies the nonlinearity in the sense (2).

Since the nonlinearity doesn't increase when all but one of the coordinate functions are eliminated, we can conclude that the nonlinearity of the vectorial function is greater than or equal to the nonlinearity of each of its coordinate functions, and, therefore, the inequality  $N_F \geq NL_F$  is valid, which becomes into equality for  $k = 1$ . Moreover, the next assertion holds.

**Proposition 2.** *Let a vectorial function  $F \in P_q^{n,k}$  have a set of coordinate functions  $\mathbf{f} = (f_1, \dots, f_k)$ . Then the following inequality is true*

$$N_F \geq \max_{\mathbf{w} \in \mathbf{F}_q^k} N_{\langle \mathbf{w}, \mathbf{f} \rangle}.$$

*Proof.* In the case when the vector  $\mathbf{w}$  is zero, the inequality is obvious. To prove the nonzero case, it suffices to note that the **EA**-equivalent function of the form  $F(\mathbf{x})\widetilde{\mathbf{L}}_j$ , where  $w_j \neq 0$  and the matrix  $\widetilde{\mathbf{L}}_j \in \mathbf{M}_q^{k,k}$ , is obtained by replacing the  $j$ -th column in the identity matrix with the column vector  $\mathbf{w}$ , has a linear combination  $\langle \mathbf{w}, \mathbf{f} \rangle$  as the  $j$ -th coordinate function.  $\square$

By analogy with the parameters of the function  $f \in P_q^{n,1}$  defined in [17] by the equality  $\delta_f^{\mathbf{a}} = (q-1)q^{-1} - \rho_f^{\mathbf{a}} q^{-n}$ , let's define the parameters of the vectorial function  $F \in P_q^{n,k}$  by the relations

$$\delta_F^{\mathbf{A}} = (q^k - 1) q^{-k} - \rho_F^{\mathbf{A}} q^{-n} = \mu_F^{\mathbf{A}} q^{-n} - q^{-k}. \quad (9)$$

where  $\mu_F^{\mathbf{A}}$  is the number of coincidences of the vectorial function  $F$  with the affine mapping  $A \in A_q^{n,k}$ . It can be seen from the right side of the chain (9) that with a random choice of the argument  $\mathbf{x}$  from  $\mathbf{F}_q^n$ , the parameter  $\delta_F^{\mathbf{A}}$  shows the deviation of the probability of coincidence of the vectorial functions  $F$  and  $A$  from the average over all affine mappings, equal to  $q^{-k}$ .

Let's call the affinity index of the vectorial function  $F \in P_q^{n,k}$  the quantity

$$\delta_F = \max_{\mathbf{A} \in \mathbf{M}_q^{n+1,k}} \delta_F^{\mathbf{A}}. \quad (10)$$

Relations (6), (9), and (10) imply the formula

$$N_F = (q^k - 1) q^{n-k} - \delta_F q^n. \quad (11)$$

The formula expression for the relationship between the introduced parameters of the vector function and the analogous parameters of the coordinate functions gives the following assertion.

**Proposition 3.** *Let  $f_j \in P_q^{n,1}$ , where  $1 \leq j \leq k$ , be the  $j$ -th coordinate of the vectorial function  $F \in P_q^{n,k}$ . Then for any vector  $\mathbf{a} \in \mathbf{F}_q^{n+1}$  and any matrix  $\widetilde{\mathbf{B}} \in \mathbf{M}_q^{n,k-1}$  we have*

$$\delta_{f_j}^{\mathbf{a}} = \sum_{\mathbf{b}_0 \in \mathbf{F}_q^{k-1}} \delta_F^{\mathbf{A}(\mathbf{b}_0)}, \quad (12)$$

where the matrix  $\mathbf{A}(\mathbf{b}_0) \in \mathbf{M}_q^{n+1,k}$  is obtained by concatenating the column order of the matrix  $\mathbf{B} = \begin{pmatrix} \mathbf{b}_0 \\ \widetilde{\mathbf{B}} \end{pmatrix} \in \mathbf{M}_q^{n+1,k-1}$  into  $1, \dots, j-1, j+1, \dots, k$  places and a column vector  $\mathbf{a}$  at the  $j$ -th place.

*Proof.* Let us prove Proposition 3 without loss of generality in the case of  $k = 2$  and  $j = 1$ . For an affine function  $a \in A_q^{n,1}$  represented by a vector  $\mathbf{a} \in \mathbf{F}_q^{n+1}$  and any vector  $(b_1, \dots, b_n) \in \mathbf{F}_q^n$ , consider the set of  $q$  matrices  $\{\mathbf{A}(\mathbf{b}_0) \in \mathbf{M}_q^{n+1,2} \mid \mathbf{b}_0 \in \mathbf{F}_q\}$  obtained by adding to the column vector  $\mathbf{a}$  the second column vector  $\mathbf{b} = (b_0, b_1, \dots, b_n)$  with an arbitrary element  $b_0$ . Then the number of coincidences of functions  $f_1$  and  $a$  satisfies the relation  $\mu_{f_1}^{\mathbf{a}} = \sum_{b_0 \in \mathbf{F}_q} \mu_F^{\mathbf{A}(\mathbf{b}_0)}$ . Applying (9), we obtain the chain of equalities

$$\delta_{f_1}^{\mathbf{a}} = \mu_{f_1}^{\mathbf{a}} q^{-n} - q^{-k+1} = \sum_{b_0 \in \mathbf{F}_q} (\mu_F^{\mathbf{A}(\mathbf{b}_0)} q^{-n} - q^{-k}) = \sum_{b_0 \in \mathbf{F}_q} \delta_F^{\mathbf{A}(\mathbf{b}_0)}.$$

□

Using the relation of the form  $\sum_{a_0 \in \mathbf{F}_q} \delta_f^{(a_0, a_1, \dots, a_n)} = 0$  given in [17] for a function  $f \in P_q^{n,1}$ , we obtain an relation for the parameters  $\delta_F^{\mathbf{A}}$ .

**Corollary 2.** *Let  $F \in P_q^{n,k}$ . Then for any matrix  $\tilde{\mathbf{A}} \in \mathbf{M}_q^{n,k}$  we have*

$$\sum_{\mathbf{a}_0 \in \mathbf{F}_q^k} \delta_F^{\mathbf{A}(\mathbf{a}_0)} = 0, \quad (13)$$

where the matrix  $\mathbf{A}(\mathbf{a}_0) = \begin{pmatrix} \mathbf{a}_0 \\ \tilde{\mathbf{A}} \end{pmatrix} \in \mathbf{M}_q^{n+1,k}$ .

Relations (9), (10), (11) and (13) allow us to obtain bounds for affinity index and nonlinearity of any vectorial function  $F \in P_q^{n,k}$  of the form

$$0 \leq \delta_F \leq 1 - q^{-k} \quad (14)$$

and

$$0 \leq NL_F \leq N_F \leq q^n - q^{n-k}. \quad (15)$$

For the nonlinearity  $NL_F$  from [17] follow bounds of the form<sup>4</sup>

$$0 \leq NL_F \leq (q-1)q^{n-1} - q^{n/2-1}. \quad (16)$$

The upper bound of the affinity index in (14) and the lower bounds of the nonlinearity in (15) and (16) are reached when  $F$  itself is an affine mapping. The lower bound in (14) and the upper bound in (15) can be refined. The reachability of the upper bound in (16) will be discussed further.

In the case when a vectorial function is a permutation  $S$  of the space  $\mathbf{F}_q^n$ , it can be represented as a permutation  $s$  of the field  $\mathbf{F}_{q^n}$  in some basis. It follows from [18] that  $N_s \leq q^n - 2$ . In contrast to the case of a ring modulo

<sup>4</sup>For  $q = 2$ , these bounds follow from the results of [16], and for  $q = p$ , they were obtained in [13].



$q^n$ , here it follows from the condition that the permutation of the field  $\mathbf{F}_{q^n}$  is affine that the corresponding permutation of the space  $\mathbf{F}_q^n$  is also affine. Consequently, for a permutation of the space  $S \in S_q^n$  and the corresponding permutation of the field  $s \in S_{q^n}^1$ , the following inequalities hold:

$$0 \leq NL_S \leq N_S \leq N_s \leq q^n - 2. \quad (17)$$

In the case  $q = 2$ , the behavior of the nonlinearity in the sense (2) for arbitrary mappings, balanced vectorial functions, and permutations has been studied by many authors. The article [3] should be singled out, in which, in addition to reviewing the results in this direction, an attempt was made to integrate them and refine the bound (16). To study permutations of spaces over the field  $\mathbf{F}_2$ , of interest is the upper bound of the nonlinearity

$$NL_S \leq 2^{n-1} - 2^{(n-1)/2}, \quad (18)$$

resulting from the Sidelnikov–Chabaud—Vaudenay’s bound [23, 5], which can only be reached for odd values of  $n$ . Since every nontrivial linear combination of permutation coordinate functions is balanced, one can also use the nonlinearity bound for balanced functions from [22] and obtain the inequality

$$NL_S \leq 2^{n-1} - 2^{n/2-1} - 2. \quad (19)$$

The bounds (18) and (19) refine the bound (16) for the Boolean case.

### 3 Estimates for the distribution of nonlinearity of arbitrary mappings, balanced vectorial functions, and permutations

Consider a set of mappings with nonlinearity not exceeding a given value  $\{F \in P_q^{n,k} \mid N_F \leq r\}$ , where  $0 \leq r \leq q^n$  (in what follows, we will also use the short notation  $\{N_F \leq r\}$  for the indicated set). As follows from (15), for  $r \geq q^n - q^{n-k}$  the set  $\{F \in P_q^{n,k} \mid N_F \leq r\}$  coincides with the entire set  $P_q^{n,k}$ . For  $k = 1$ , in the article [27], in particular, it is shown that for Boolean functions for  $0 \leq r < 2^{n-1} - 2^{n/2-1}$  the following inequality holds:  $|\{N_f \leq r\}| \leq 2^{n+1} \sum_{i=0}^r \binom{2^n}{i}$ . In [17], this result was generalized to the case of  $q$ -valued functions.

**Theorem 1.** *Let the vectorial function  $F$  be chosen randomly and with equal probability from the set  $P_q^{n,k}$ . Then for  $0 \leq r < q^n - q^{n-k}$  for the probability of the event  $\{F \in P_q^{n,k} \mid N_F \leq r\}$  we have the following estimate*

$$\mathbf{P}(N_F \leq r) \leq q^{k(n+1-q^n)} \sum_{i=0}^r \binom{q^n}{i} (q^k - 1)^i. \quad (20)$$

*Proof.* To an affine mapping  $A_h \in A_q^{n,k}$ , where  $h = 1, \dots, q^{k(n+1)}$ , we associate the set  $\{F \in P_q^{n,k} \mid \rho(F, A_h) \leq r\}$ . Each such set is the union of pairwise disjoint  $r+1$  sets, that is, the equality  $\{F \in P_q^{n,k} \mid \rho(F, A_h) \leq r\} = \bigcup_{i=0}^r \{F \in P_q^{n,k} \mid \rho(F, A_h) = i\}$  is satisfied. In turn, the powers of the latter, in accordance with [21], are found by the formula

$$|\{F \in P_q^{n,k} \mid \rho(F, A_h) = i\}| = \binom{q^n}{i} (q^k - 1)^i. \quad (21)$$

Then the chain of relations is valid

$$\begin{aligned} \mathbf{P}(N_F \leq r) &= |\{N_F \leq r\}| / |P_q^{n,k}| \leq q^{-kq^n} \sum_{h=1}^{q^{k(n+1)}} |\{F \in P_q^{n,k} \mid \rho(F, A_h) \leq r\}| = \\ &= q^{-kq^n} \sum_{h=1}^{q^{k(n+1)}} \sum_{i=0}^r \binom{q^n}{i} (q^k - 1)^i = q^{k(n+1)-kq^n} \sum_{i=0}^r \binom{q^n}{i} (q^k - 1)^i. \end{aligned}$$

□

**Remark 1.** To simplify the calculation of the expression on the right side of relation (20), for  $r \leq (q^n + 1)/2$  and  $q^k \neq 2$ , we can use the inequality

$$\sum_{i=0}^r \binom{q^n}{i} (q^k - 1)^i < \binom{q^n}{r} (q^k - 1)^{r+1} / (q^k - 2).$$

Let's now consider the case balanced vectorial functions.

**Theorem 2.** Let the mapping  $G$  be chosen randomly and with equal probability from the set  $B_q^{n,k}$ . Then for  $0 \leq r < q^n - q^{n-1}$  for the probability of the event  $\{G \in B_q^{n,k} \mid N_G \leq r\}$  we have the following estimate

$$\mathbf{P}(N_G \leq r) \leq (q^{n-k}!)^k \prod_{h=1}^k (q^{n+1} - q^h) \sum_{i=0}^r 1/(q^n - i)! \sum_{j=0}^i (-1)^j / j!, \quad (22)$$

*Proof.* It is carried out similarly to the proof of Theorem 1. In this case, for any affine mapping  $A$  that isn't balanced, the inequality  $\rho(G, A) \geq q^n - q^{n-1}$  holds, and, therefore, such mappings can be excluded from consideration. The number of affine balanced mappings is  $\prod_{h=1}^k (q^{n+1} - q^h)$ . Based on the formula obtained for the Hamming distance in [21], for the number of balanced vectorial functions removed at a distance  $i$  from a balanced affine mapping  $B$ , as (21) the following inequality is used

$$|\{G \in B_q^{n,k} \mid \rho(G, B) = i\}| \leq q^n! / (q^n - i)! \sum_{j=0}^i (-1)^j / j!, \quad (23)$$

and the cardinality of the set  $B_q^{n,k}$  is  $q^n! / (q^{n-k}!)^k$ . □

Finally, consider the case of permutations. As follows from (17), for  $r \geq q^n - 2$  the set  $\{S \in S_q^n \mid N_S \leq r\}$  coincides with the set  $S_q^n$ .

**Corollary 3.** *Let the permutation  $S$  be chosen randomly and with equal probability from the set  $S_q^n$ . Then for  $0 \leq r < q^n - q^{n-1}$  for the probability of the event  $\{S \in S_q^n \mid N_S \leq r\}$  we have the following estimate*

$$\mathbf{P}(N_S \leq r) \leq \prod_{h=1}^n (q^{n+1} - q^h) \sum_{i=0}^r 1/(q^n - i)! \sum_{j=0}^i (-1)^j / j!. \quad (24)$$

To prove Corollary 3, it suffices to put  $k$  equal to  $n$  in (22). Note that in this case, for an affine permutation, the inequality (23) becomes an equality.

**Remark 2.** *To simplify the calculations of the expressions included in the right-hand sides of relations (22) and (24), we can use the approximation  $\sum_{t=0}^j (-1)^t / t! \approx e^{-1}$ , and also use the Stirling formula.*

Using the obtained results, it is easy to show, for example, that for most vectorial functions from  $P_2^{4,4}$ , space permutations from  $S_2^4$ , and field permutations from  $S_{16}^1$ , the nonlinearity is greater than or equal to 8, 8, and 11.

## 4 Construction of highly nonlinear vectorial functions

For  $q = 2$  and even values of  $n$ , the nonlinearity  $NL_F$  in the sense (2) reaches the upper bound in (16) only in the case of Boolean vectorial bent functions. It follows from the results of [14] that such mappings exist only under one more condition  $n \geq 2k$ . Under these conditions, various classes of Boolean vectorial bent functions have been constructed (see [25, 11]). For  $q > 2$  and  $k = 1$ , it was shown in [20], that the bent functions from Maiorana-McFarland's and Dillon's families are not maximally nonlinear, and a new family of maximally nonlinear bent functions was constructed. A similar situation with respect to nonlinearity  $NL_F$  in the sense (2) also takes place for vectorial functions.

Let's demonstrate this using the method proposed here for constructing vectorial functions belonging to the set  $P_p^{mn,m}$  from bent functions belonging to the set  $P_q^{n,1}$ , where  $q = p^m$ ,  $m \geq 2$ , and  $n$  is even. Let the bent function  $f \in P_q^{n,1}$ . We represent the field  $\mathbf{F}_q$  as an  $m$ -dimensional vector space over the prime field  $\mathbf{F}_p$  and define a vectorial function  $F$  whose coordinate functions are the corresponding coordinates of the bent function  $f$  in the given representation, and whose variables belong to the vector space  $\mathbf{F}_p^{mn}$ .

In accordance with the results of article [2], all nontrivial linear combinations of the vectorial function  $F$  are also bent functions. For  $p = 2$ , each such linear combination is a Boolean bent function, and, accordingly, the vectorial function  $F$  has the maximum possible nonlinearity  $NL_F$  equal to  $2^{mn-1} - 2^{mn/2-1}$ . But for  $p > 2$ , as follows from the work [19], in order to be guaranteed to obtain a nonlinearity  $NL_F$  equal to  $(p - 1)p^{mn-1} - p^{mn/2-1}$ , it is also necessary to take the maximally nonlinear bent function as the initial function  $f$ .

**Example 1.** Let  $q = 9$  and  $n = 2$ , and the generating polynomial  $g(x) = x^2 \oplus x \oplus 1$  is used to construct a field of nine elements, and, accordingly,  $\mathbf{F}_9 = \{0, 1, 2, x, x \oplus 1, x \oplus 2, 2x, 2x \oplus 1, 2x \oplus 2\} = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ . Consider two bent functions from  $P_9^{2,1}$ :  $f_1 = x_1x_2$  and  $f_2 = x_1^2 \oplus 3x_2^2$ . According to [18],  $f_1$  has nonlinearity  $N_{f_1}$  equal to 64, and  $f_2$  is a maximally nonlinear bent function with nonlinearity  $N_{f_2}$  equal to 71. The vectorial functions  $F_1$  and  $F_2$  from  $P_3^{4,2}$  obtained from them using the method described above have the following sets of coordinate functions:  $\{2x_1x_3 \oplus x_1x_4 \oplus x_2x_3, x_1x_3 \oplus x_2x_4\}$  and  $\{2x_1^2 \oplus 2x_3^2 \oplus x_4^2 \oplus 2x_1x_2 \oplus x_3x_4, x_1^2 \oplus x_2^2 \oplus 2x_3^2 \oplus 2x_3x_4\}$ . They satisfy the equalities  $NL_{F_1} = 48$  and  $NL_{F_2} = 51$  with respect to the nonlinearities in the sense (2), and the equalities  $N_{F_1} = 64$  and  $N_{F_2} = 68$  for the nonlinearities in the sense (1) are also valid. Taking into account formula (3), the affinity orders satisfy the relations  $\mathbf{ard}F_1 \geq 5$  and  $\mathbf{ard}F_2 \geq 7$ .

Since it follows from Theorem 1 that the inequality  $N_F \geq 58$  holds for most mappings from  $P_3^{4,2}$ , the above example shows that the use of vectorial functions with maximum nonlinearity  $NL_F$  provides high nonlinearity  $N_F$  and the order affinity  $\mathbf{ard} F$ . This is confirmed by the following result.

**Statement 1.** If the vectorial function  $F \in P_3^{n,2}$  satisfies the equality

$$NL_F = 2 \cdot 3^{n-1} - 3^{n/2-1}, \tag{25}$$

then its affinity index, nonlinearity, and affinity order satisfy the inequalities

$$\delta_F \leq 4 \cdot 3^{-n/2-2}, \tag{26}$$

$$N_F \geq 4 \cdot (2 \cdot 3^{n-2} - 3^{n/2-2}), \tag{27}$$

$$\mathbf{ard}F \geq \begin{cases} 4, & \text{if } n = 2; \\ 7, & \text{if } n = 4; \\ 8, & \text{if } n = 6; \\ 9, & \text{if } n \geq 8. \end{cases} \tag{28}$$

*Proof.* For a function  $f \in P_q^{n,1}$  and a vector  $\tilde{\mathbf{a}} \in \mathbf{F}_q^n$ , denote by  $\nabla_{\tilde{\mathbf{a}}} f$  the set of  $q$  parameters  $\{\delta_f^{(a_0, \tilde{\mathbf{a}})} \mid a_0 \in \mathbf{F}_q\}$ , and for a vectorial function  $F \in P_q^{n,k}$  and a matrix  $\tilde{\mathbf{A}} \in \mathbf{M}_q^{n,k}$ , denote by  $\nabla_{\tilde{\mathbf{A}}} F$  the set of  $q^k$  parameters  $\{\delta_F^{(\tilde{\mathbf{A}})} \mid \mathbf{a}_0 \in \mathbf{F}_q^k\}$ .

Let a mapping  $F \in P_3^{n,2}$  with a set of coordinate functions  $\mathbf{f} = (f_1, f_2)$  satisfy condition (25). Then  $n$  is even, and it follows from the results of [19] that, for any vectors  $\mathbf{w} \in \mathbf{F}_3^2 \setminus \{\mathbf{0}\}$  and  $\tilde{\mathbf{a}} \in \mathbf{F}_3^n$ , sets of the form  $\nabla_{\langle \mathbf{w}, \mathbf{f} \rangle} \tilde{\mathbf{a}}$  consist of a parameter equal to  $-2 \cdot 3^{-n/2-1}$  and two parameters equal to  $3^{-n/2-1}$ .

For any matrix  $\tilde{\mathbf{A}} = (\tilde{\mathbf{a}}_1, \tilde{\mathbf{a}}_2) \in \mathbf{M}_3^{n,2}$ , the set  $\nabla_{\tilde{\mathbf{A}}} F$  includes nine parameters  $\{\delta_F^{\mathbf{A}(a_{0,1}, a_{0,2})} \mid a_{0,1}, a_{0,2} \in \mathbf{F}_q\}$ , where  $\mathbf{A}(a_{0,1}, a_{0,2}) = \begin{pmatrix} a_{0,1} & a_{0,2} \\ \tilde{\mathbf{a}}_1 & \tilde{\mathbf{a}}_2 \end{pmatrix} \in \mathbf{M}_3^{n+1,2}$ . Applying sums (12) to them, we obtain two systems of linear equations

$$\begin{cases} \delta_F^{\mathbf{A}(0,0)} + \delta_F^{\mathbf{A}(0,1)} + \delta_F^{\mathbf{A}(0,2)} = \delta_{f_1}^{(0, \tilde{\mathbf{a}}_1)} \\ \delta_F^{\mathbf{A}(1,0)} + \delta_F^{\mathbf{A}(1,1)} + \delta_F^{\mathbf{A}(1,2)} = \delta_{f_1}^{(1, \tilde{\mathbf{a}}_1)} \\ \delta_F^{\mathbf{A}(2,0)} + \delta_F^{\mathbf{A}(2,1)} + \delta_F^{\mathbf{A}(2,2)} = \delta_{f_1}^{(2, \tilde{\mathbf{a}}_1)} \end{cases} \quad (29)$$

$$\begin{cases} \delta_F^{\mathbf{A}(0,0)} + \delta_F^{\mathbf{A}(1,0)} + \delta_F^{\mathbf{A}(2,0)} = \delta_{f_2}^{(0, \tilde{\mathbf{a}}_2)} \\ \delta_F^{\mathbf{A}(0,1)} + \delta_F^{\mathbf{A}(1,1)} + \delta_F^{\mathbf{A}(2,1)} = \delta_{f_2}^{(1, \tilde{\mathbf{a}}_2)} \\ \delta_F^{\mathbf{A}(0,2)} + \delta_F^{\mathbf{A}(1,2)} + \delta_F^{\mathbf{A}(2,2)} = \delta_{f_2}^{(2, \tilde{\mathbf{a}}_2)} \end{cases} \quad (30)$$

where their right sides are either  $(-2 \cdot 3^{-n/2-1}, 3^{-n/2-1}, 3^{-n/2-1})$ , or  $(3^{-n/2-1}, -2 \cdot 3^{-n/2-1}, 3^{-n/2-1})$ , or  $(3^{-n/2-1}, 3^{-n/2-1}, -2 \cdot 3^{-n/2-1})$ .

Let us now define a vectorial function by the relation  $F' = F\tilde{\mathbf{L}}'$ , where  $\tilde{\mathbf{L}}' = \begin{pmatrix} 1,1 \\ 0,1 \end{pmatrix} \in \mathbf{M}_3^{2,2}$ , with the set of coordinate functions  $\{f_1, f_1 \oplus f_2\}$ . Since the matrix  $\tilde{\mathbf{L}}'$  is nonsingular, the vectorial function  $F'$  is  $\mathbf{EA}$ -equivalent to the mapping  $F$  and, in accordance with relations (8) and (9), for any matrix  $\mathbf{A} \in \mathbf{M}_3^{n+1,2}$ , we have the equality  $\delta_{F'}^{\mathbf{A}'} = \delta_F^{\mathbf{A}}$ , where  $\mathbf{A}' = \begin{pmatrix} \mathbf{a}'_0 \\ \mathbf{A}' \end{pmatrix} \in \mathbf{M}_3^{n+1,2}$ , the submatrix  $\tilde{\mathbf{A}}' = \tilde{\mathbf{A}}\tilde{\mathbf{L}}'$  and the zero row  $\mathbf{a}'_0 = \mathbf{a}_0\tilde{\mathbf{L}}'$ . Taking into account the form of the matrix  $\tilde{\mathbf{L}}'$ , for all possible  $a_{0,1}, a_{0,2} \in \mathbf{F}_q$ , we have the relations

$$\delta_{F'}^{\mathbf{A}'(a_{0,1}, a_{0,1} \oplus a_{0,2})} = \delta_F^{\mathbf{A}(a_{0,1}, a_{0,2})}. \quad (31)$$

Applying sums (12) with respect to the second coordinate  $F'$ , we obtain a system of linear equations

$$\begin{cases} \delta_{F'}^{\mathbf{A}'(0,0)} + \delta_{F'}^{\mathbf{A}'(1,0)} + \delta_{F'}^{\mathbf{A}'(2,0)} = \delta_{f_1 \oplus f_2}^{(0, \tilde{\mathbf{a}}_1 \oplus \tilde{\mathbf{a}}_2)} \\ \delta_{F'}^{\mathbf{A}'(0,1)} + \delta_{F'}^{\mathbf{A}'(1,1)} + \delta_{F'}^{\mathbf{A}'(2,1)} = \delta_{f_1 \oplus f_2}^{(1, \tilde{\mathbf{a}}_1 \oplus \tilde{\mathbf{a}}_2)} \\ \delta_{F'}^{\mathbf{A}'(0,2)} + \delta_{F'}^{\mathbf{A}'(1,2)} + \delta_{F'}^{\mathbf{A}'(2,2)} = \delta_{f_1 \oplus f_2}^{(2, \tilde{\mathbf{a}}_1 \oplus \tilde{\mathbf{a}}_2)} \end{cases}$$

which, with the help of relations (31), is reduced to the form

$$\begin{cases} \delta_F^{\mathbf{A}(0,0)} + \delta_F^{\mathbf{A}(1,2)} + \delta_F^{\mathbf{A}(2,1)} = \delta_{f_1 \oplus f_2}^{(0, \widetilde{\mathbf{a}}_1 \oplus \widetilde{\mathbf{a}}_2)} \\ \delta_F^{\mathbf{A}(0,1)} + \delta_F^{\mathbf{A}(1,0)} + \delta_F^{\mathbf{A}(2,2)} = \delta_{f_1 \oplus f_2}^{(1, \widetilde{\mathbf{a}}_1 \oplus \widetilde{\mathbf{a}}_2)} \\ \delta_F^{\mathbf{A}(0,2)} + \delta_F^{\mathbf{A}(1,1)} + \delta_F^{\mathbf{A}(2,0)} = \delta_{f_1 \oplus f_2}^{(2, \widetilde{\mathbf{a}}_1 \oplus \widetilde{\mathbf{a}}_2)} \end{cases} \quad (32)$$

where the right side (32) can take only the three values indicated above.

By analogous reasoning, we obtain that one more **EA**-equivalent vectorial function given by the relation  $F'' = F\widetilde{\mathbf{L}}''$ , where  $\widetilde{\mathbf{L}}'' = \begin{pmatrix} 1,1 \\ 0,2 \end{pmatrix} \in \mathbf{M}_3^{2,2}$ , with the set of coordinate functions  $\{f_1, f_1 \oplus 2f_2\}$  leads to a system of linear equations

$$\begin{cases} \delta_F^{\mathbf{A}(0,0)} + \delta_F^{\mathbf{A}(1,1)} + \delta_F^{\mathbf{A}(2,2)} = \delta_{f_1 \oplus 2f_2}^{(0, \widetilde{\mathbf{a}}_1 \oplus 2\widetilde{\mathbf{a}}_2)} \\ \delta_F^{\mathbf{A}(0,1)} + \delta_F^{\mathbf{A}(1,2)} + \delta_F^{\mathbf{A}(2,0)} = \delta_{f_1 \oplus 2f_2}^{(1, \widetilde{\mathbf{a}}_1 \oplus 2\widetilde{\mathbf{a}}_2)} \\ \delta_F^{\mathbf{A}(0,2)} + \delta_F^{\mathbf{A}(1,0)} + \delta_F^{\mathbf{A}(2,1)} = \delta_{f_1 \oplus 2f_2}^{(2, \widetilde{\mathbf{a}}_1 \oplus 2\widetilde{\mathbf{a}}_2)} \end{cases} \quad (33)$$

where the right side (33) can again take only the three values indicated above.

Combining systems (29), (30), (32) and (33), we obtain a system of 12 linear equations. The use of other nontrivial linear combinations of coordinate functions does not lead to the emergence of new equations. The right side of the resulting unified system can take 81 values. Each of the possible 81 systems is compatible and has one solution. Using these solutions, we conclude that for any matrix  $\widetilde{\mathbf{A}} \in \mathbf{M}_3^{n,2}$  the collection  $\nabla_F^{\widetilde{\mathbf{A}}}$  does not go beyond two sets of the form  $\{-8 \cdot 3^{-n/2-2}, 3^{-n/2-2}, 3^{-n/2-2}, 3^{-n/2-2}, 3^{-n/2-2}, 3^{-n/2-2}, 3^{-n/2-2}, 3^{-n/2-2}, 3^{-n/2-2}\}$  and  $\{-5 \cdot 3^{-n/2-2}, -2 \cdot 3^{-n/2-2}, -2 \cdot 3^{-n/2-2}, -2 \cdot 3^{-n/2-2}, 4 \cdot 3^{-n/2-2}, 4 \cdot 3^{-n/2-2}\}$ . Therefore, inequalities (26) and (27) are valid. Applying (3) to (27) we obtain (28).  $\square$

**Corollary 4.** *In the case of  $q = 9$  and  $n$  divisible by 4, for the vectorial function  $F \in P_3^{n,2}$  composed of the coordinates of the maximally nonlinear bent function  $f \in P_9^{n/2,1}$ , the relations (25), (26), (27) and (28) are valid.*

**Remark 3.** *In a similar way, one can show that in the case of  $q = 4$  and  $n$  divisible by 4, for a Boolean mapping  $F \in P_2^{n,2}$  composed of the coordinates of the bent function  $f \in P_4^{n/2,1}$ , the relations  $NL_F = 2^{n-1} - 2^{n/2-1}$ ,  $\delta_F \leq 3 \cdot 2^{-n/2-2}$ ,  $N_F \geq 3 \cdot (2^{n-2} - 2^{n/2-2})$  and  $\mathbf{ard}F \geq \begin{cases} 3, & \text{if } n = 4; \\ 4, & \text{if } n \geq 8. \end{cases}$  are valid.*

**Remark 4.** *The classes of mappings presented in Corollary 4 and Remark 3 can be extended by **EA**-equivalent mappings and by adding coordinate functions. In the latter case, one should keep in mind that the nonlinearity  $NL_F$  of the resulting mappings may turn out to be less than the original one.*

In conclusion, using the example of permutation from  $S_2^4$ , we show that for the same value of the nonlinearity in the sense of (2), mappings can have different nonlinearities in the sense of (1) and differ significantly in resistance to methods using affine approximations.

Among the highly nonlinear permutations from  $S_2^4$ , researchers include the transformations generated by the power function  $x^d$  for some integer  $d$ , primarily  $d = 14$ , which is equivalent to  $x^{-1}$  for nonzero values of  $x$ . The function  $x^{14}$  generates the permutation  $S \in S_2^4$ . In accordance with (19), the permutation  $S$  has the maximum possible nonlinearity  $NL_S$  equal to 4, the nonlinearity  $N_S$  equal to 9, and the affinity order  $ardS$  greater than or equal 3. This value of the nonlinearity in the sense of (1) can be considered quite high, since most permutations from  $S_2^4$  and even most of the vectorial functions from  $P_2^{4,4}$  have a nonlinearity greater than or equal to 8.

In [1], taking into account various cryptographic criteria, "good" permutations from  $S_2^4$  were chosen according to the authors. The first among them was the permutation of the form  $S' = \{9, 13, 10, 15, 11, 14, 7, 3, 12, 8, 6, 2, 4, 1, 0, 5\}$ . This permutation  $S'$  has the maximum possible nonlinearity  $NL_{S'}$  equal to 4, and is really not inferior in this parameter to the permutation  $S$ . Moreover, unlike  $S$ , the permutation  $S'$  does not keep 0 and 1 in place. However, its nonlinearity in the sense (1) is less than that of the permutation  $S$  and is equal to 8. In this case, as can be seen from the table below of values for a given permutation  $S'$  and two affine mappings  $A_1$  and  $A_2$  from  $A_2^{4,4}$ , the latter can be considered as paired analogues of  $S'$  on the entire space  $\mathbf{F}_2^4$ . Thus, the order affinity  $ardS'$  is 2, and the system of equations given by the permutation  $S'$  is easily solved by the above method.

$x$	$S'$	$A_1$	$A_2$
0000	1001	<u>1001</u>	1110
0001	1101	<u>1101</u>	1011
0010	1010	0011	<u>1010</u>
0011	1111	0111	<u>1111</u>
0100	1011	1101	<u>1011</u>
0101	1110	1001	<u>1110</u>
0110	0111	<u>0111</u>	1111
0111	0011	<u>0011</u>	1010
1000	1100	<u>1100</u>	0001
1001	1000	<u>1000</u>	0100
1010	0110	<u>0110</u>	0101
1011	0010	<u>0010</u>	0000
1100	0100	1000	<u>0100</u>
1101	0001	1100	<u>0001</u>
1110	0000	0010	<u>0000</u>
1111	0101	0110	<u>0101</u>

## 5 Conclusion

In this work, the classical Hamming distance is used as a measure of closeness of vector spaces over finite fields. Previously, this approach was applied in cryptographic studies to functions with a one-dimensional space of values, primarily to Boolean functions. The need to turn to the classical measure in the vector case in the presence of other measures used in cryptography is due to a number of reasons. These include the fact that the Hamming distance is a metric that allows one to reasonably approach the measurement of proximity to the set of affine mappings over finite fields and avoid identification with affine vector functions that are not such. This approach creates the basis for studying the possibility of representing vectorial functions in a piecewise affine form, and the value of nonlinearity reflects the maximum size of the affinity region and allows us to estimate the order of affinity of the vectorial function, which is important when solving systems of nonlinear equations by methods using affine approximations.

In Section 2, we study the properties of parameters that characterize the closeness of a vectorial function to affine mappings, find connections between these parameters and similar parameters of its coordinate functions, and, based on the latter, draw conclusions about the relationship between the nonlinearity of a vectorial function and the nonlinearities of its coordinate functions and their linear combinations. This result is interesting because at present the definition of the nonlinearity of a vectorial function as minimal among nonlinearities of all nontrivial linear combinations of coordinate functions has become most widespread. The work compares these two approaches to nonlinearity. Due to the limited volume, the measures associated with the difference approach and the nonlinearities arising in this connection are not considered in this work.

Section 3 is devoted to meaningful estimates of the distribution of the nonlinearity introduced in the paper for arbitrary mappings, balanced vectorial functions, and permutations. Similar results for alternative nonlinearities are not known to the author.

In Section 4, we propose a method for constructing vectorial functions with a high nonlinearity using the coordinates of maximally nonlinear bent functions (in the Boolean case, any bent functions) given on a field of the same characteristic, but of a higher order. Here we also give an example of a permutation with the maximum possible nonlinearity for nontrivial combinations of coordinate functions, but with a relatively low nonlinearity introduced in this paper, which makes this permutation cryptographically vulnerable.



## References

- [1] Adams C., Tavares S., “Good S-boxes are easy to find”, *LNCS*, CRYPTO 1989., **435**, ред. Brassard G., Springer, Berlin, Heidelberg, 1990, 612–615.
- [2] Ambrosimov A.S., “Properties of bent functions of  $q$ -valued logic over finite fields”, *Discrete Mathematics and Applications*, **4:4** (1994), 341–350.
- [3] Carlet C., “Relating three nonlinearity parameters of vectorial functions and building APN functions from bent functions”, *Designs, Codes and Cryptography*, **59:1** (2011), 89–109.
- [4] Carlet C., Ding C., “Nonlinearities of S-boxes”, *Finite Fields and Their Applications*, **13:1** (2007), 121–135.
- [5] Chabaud F., Vaudenay S., “Links between differential and linear cryptanalysis”, *LNCS*, EUROCRYPT 1994., **950**, ред. De Santis A., Springer, Berlin, Heidelberg, 1995, 356–365.
- [6] Fomichev V.M., *Discrete mathematics and cryptology*, (Lecture course), Dialog-MIFI, Moscow, 2003, 397 pp., In Russian.
- [7] Glukhov M. M., “On the approximation of discrete functions by linear functions”, *Matematicheskie voprosy kriptografii*, **7:4** (2016), 29–50, In Russian.
- [8] Gorshkov S.P., Dvinyaninov A.V., “Lower and upper bounds for the affinity order of transformations of Boolean vector spaces”, *Prikladnaya diskretnaya matematika*, **2(20)** (2013), 14–18, In Russian.
- [9] Kuz'min A.S., Nechaev A.A., Shishkin V.A., “Bent and hyper-bent functions over the finite field”, *Trudy po diskretnoi matematike*, **10**, ред. Prokhorov Y.V., Fizmatlit, Moscow, 2007, 97–122, In Russian.
- [10] Logachev O.A., Sal'nikov A.A., Yashchenko V.V., “Some characteristics of “nonlinearity” of group mappings”, *Diskretnyi analiz i issledovanie operatsii, Ser. 1*, **8:1** (2001), 40–54, In Russian.
- [11] Mesnager S., *Bent functions: fundamentals and results*, Springer International Publishing AG, Cham, 2016, 544 pp.
- [12] Niederreiter H., “Orthogonal systems of polynomials in finite fields”, *Proceedings of the American Mathematical Society*, **28:2** (1971), 415–422
- [13] Nyberg K., “Constructions of bent functions and difference sets”, *LNCS*, EUROCRYPT 1990., **473**, ред. Damgård I.B., Springer, Berlin, Heidelberg, 1991, 151–160
- [14] Nyberg K., “Perfect nonlinear S-boxes”, *LNCS*, EUROCRYPT 1991., **547**, ред. Davies D.W., Springer, Berlin, Heidelberg, 1991, 378–386
- [15] Nyberg K., “On the construction of highly nonlinear permutations”, *LNCS*, EUROCRYPT 1992., **658**, ред. Rueppel R.A., Springer, Berlin, Heidelberg, 1993, 92–98
- [16] Rothaus O. S., “On “bent” functions”, *Journal of Combinatorial Theory, Series A*, **20:3** (1976), 300–305
- [17] Ryabov V.G., “Approximation of restrictions of  $q$ -valued logic functions to linear manifolds by affine analogues”, *Discrete Mathematics and Applications*, **31:6** (2021), 409–419
- [18] Ryabov V.G., “Maximally nonlinear functions over finite fields”, *Diskretnaya matematika*, **33:1** (2021), 47–63, In Russian.
- [19] Ryabov V.G., “Criteria for the maximum nonlinearity of a function over a finite field”, *Diskretnaya matematika*, **33:3** (2021), 79–91, In Russian.
- [20] Ryabov V.G., “Nonlinearity of bent functions over finite fields”, *Matematicheskie voprosy kriptografii*, **12:4** (2021), 87–98, In Russian.
- [21] Sachkov V.N., *Introduction to combinatorial methods discrete mathematics*, Nauka, Moscow, 1982, 384 pp., In Russian.
- [22] Seberry J., Zhang X.-M., Zheng Y., “Nonlinearity and propagation characteristics of balanced Boolean functions”, *Information and Computation*, **119:1** (1995), 1–13.
- [23] Sidelnikov V.M., “On the mutual correlation of sequences”, *Problemy kibernetikii*, **24**, ред. Lyapunov A.A., Nauka, Moscow, 1971, 15–42, In Russian.
- [24] Solodovnikov V. I., “Bent functions from a finite abelian group into a finite abelian group”, *Discrete Mathematics and Applications*, **12:2** (2002), 111–126.

- [25] Tokareva N., *Bent functions: results and applications to cryptography*, Academic Press, Elsevier, Global, 2015, 220 pp.
- [26] Yashchenko V.V., “On two characteristics of nonlinearity of Boolean mappings”, *Diskretnyi analiz i issledovanie operatsii, Ser. 1*, **5:2** (1998), 90–96, In Russian.
- [27] Zubkov A.M., Serov A.A., “Bounds for the number of Boolean functions admitting affine approximations of a given accuracy”, *Discrete Mathematics and Applications*, **20:5-6** (2010), 467–486.

# On subfunctions of self-dual bent functions

Aleksandr Kutsenko

Sobolev Institute of Mathematics, Novosibirsk, Russia  
Novosibirsk State University, Novosibirsk, Russia  
alexandr.kutsenko@bk.ru

## Abstract

Bent functions are Boolean functions in even number of variables that have maximal nonlinearity. They also have a flat Walsh–Hadamard spectrum and are of interest for their applications in algebra, coding theory and cryptography. A bent function is called self-dual if it coincides with its dual bent function. In current work we study the subfunctions of self-dual bent functions. We consider the subfunctions whose concatenation forms the vector of values of the function.

Based on a spectral characterization, we introduce a notion of self-duality for near-bent functions in odd number of variables. We describe subfunctions in  $n - 1$  variables of self-dual bent function in  $n$  variables and prove that there exists an one-to-one correspondence between the set of self-dual bent functions in  $n$  variables and the set of self-dual near-bent functions in  $n - 1$  variables.

We deduce the general form of the Gram matrix of sign functions of subfunctions in  $n - 2$  variables of an arbitrary bent function. Metrical relations between the subfunctions are obtained. By using the obtained form of the Gram matrix we prove that if sign functions of subfunctions in  $n - 2$  variables of self-dual bent function are linearly dependent then all subfunctions are bent functions. We also prove that for  $n \geq 6$  the converse does not hold, that is the singularity of the Gram matrix provides only sufficient condition for subfunctions to be bent.

Three new iterative constructions of self-dual bent functions are proposed. Some of them for the first time provide self-dual bent functions with bent subfunctions having nonsingular Gram matrix. One of the constructions allows to build a class of self-dual bent functions which cannot be decomposed into the concatenation of four bent functions. Based on the constructions a new iterative lower bound on the cardinality of the set of self-dual bent functions is obtained.

Functions of the form  $\mathbb{F}_2^n \rightarrow \mathbb{Z}_q$ , where  $q \geq 2$  is a positive integer, having flat generalized Walsh–Hadamard transform spectrum are known as generalized bent (gbent) functions. We study the open problem of the existence of (anti-)self-dual gbent functions in odd number of variables. It is known that such functions do not exist for  $q = 2, 4$ . We prove that they exist when  $q \equiv 0 \pmod{8}$ , in particular, for  $q = 2^k$  with  $k \geq 3$ .

**Keywords:** Self-dual bent, Rayleigh quotient, Near-bent, Subfunction of a Boolean function

## 1 Introduction

Bent functions are Boolean functions in even number of variables that have a maximal nonlinearity. They were firstly published by O.Rothaus

in [28]. They are mathematical objects of a great interest due to many applications in discrete mathematics, algebra, coding theory, cryptography. Due to a property of maximal nonlinearity these functions can be used for obtaining Boolean and vectorial Boolean functions with good cryptographic properties. Another applications use the fact that bent functions and only them have flat Walsh–Hadamard spectrum that is crucial for some approaches within the signals theory, including CDMA technology. More information about them one can find in monographies [32, 26]. For extensive data about cryptographic properties of Boolean functions and other their applications one can refer to the books [21, 6]. Despite the long history of study there are many open problems related to bent functions, in particular, their cardinality is still unknown, their affine classification is completely studied only for small number of variables, obtaining of new constructions is also the goal worth pursuing.

For every bent function it is possible to define its dual Boolean function that defines the signs of its Walsh–Hadamard transform. This functions is also bent and, in order, its dual coincides with the initial function, so bent functions come in pairs. More information about the duals and the properties of the duality mapping one can find in [4, 15].

Among different classes of bent functions the class of self-dual bent functions is emphasized. Self-dual are such bent functions that coincide with their duals. They are also important from the perspective of obtaining polyphase sequences with particular properties. The polyphase sequence of self-dual bent function is the eigenvector of the Sylvester Hadamard matrix that appears in many areas of discrete mathematics and also in quantum computation. So the construction and characterization of self-dual bent functions has strong relation with the problem of description of eigenvectors of the Sylvester Hadamard matrix [35]. Also self-dual bent functions are the fixed points of the duality mapping that has great interest in a scope of bent functions. Also note that on self-dual bent functions and only on them the Rayleigh quotient of a Boolean function has maximal value for the case of even number of variables.

Self-dual bent function were firstly studied by Carlet et al. in paper [5], though more general definition of a self-dual bent function on a finite group was introduced by Logachev, Sal’nikov and Yashchenko earlier in [20]. From that time there were a number of papers devoted to the study and characterization of self-dual bent functions. In particular, the classification of quadratic self-dual bent functions was provided by Hou in [9]. The classification of cubic self-dual bent functions in 8 variables was done in paper [8], while the bounds for the cardinality of this class were deduced in [10]. New constructions were

presented in [25, 22, 18]. Metrical properties of self-dual bent functions were studied in papers [11, 12, 13, 14].

There are known several generalizations of self-duality for the so called generalized Boolean functions, that is the mappings of the form  $\mathbb{F}_2^n \rightarrow \mathbb{Z}_q$ . For these functions the definition of a generalized bent function is that the function has flat Walsh–Hadamard spectrum. It is worth noting that there exist generalized bent functions in odd number of variables for the case  $q = 2^k$  [23]. The duality can be defined only for a subset of generalized bent functions that are called regular, the same holds for self-duality.

It was known that self-dual generalized bent functions exist for  $n$  and  $q$  both even. For the case of odd  $n$  it was known that for  $q = 4$  self-dual generalized bent functions do not exist [30]. The question of their existence for the case of an odd  $n$  for the general case remains an open one.

In current work we concentrate on the subfunctions of self-dual bent functions obtained by fixing one or two variables. Note that the best known for today lower and upper bounds on the cardinality of the set of self-dual bent functions are based on the analysis of its subfunctions. The problem of the existence of self-dual generalized bent functions in odd number of variables under some limitations is also considered.

The structure of the work is following. Necessary notation is in Section 2. In Section 3 we share a concept of self-duality on near-bent functions in odd number of variables and prove that there is an one-to-one correspondence between self-dual bent function in  $n$  variables and near-bent functions in  $n - 1$  variables having particular value of the Rayleigh quotient (Theorem 1). Note that this value coincides with the best known for today bound for the maximal value of the Rayleigh quotient for the case of odd number of variables. Further, in Section 4.2 we study the Gram matrix obtained via sign functions of subfunctions obtained by fixing two variables of bent function. The general form of this matrix is deduced (Theorem 2). We use it for obtaining metrical relations between subfunctions of every bent function (Theorem 3). The Rayleigh quotients of subfunctions are characterized in Section 4.3 and their general form is obtained (Proposition 2). The form of the Gram matrix is explicitly used in Section 4.4, where we prove that given a self-dual bent function with linearly dependent sign functions of subfunctions, all these functions are necessarily bent (Theorem 4, Corollary 1). New constructions and lower bound on the number of self-dual bent functions are presented in Section 5, whereas the converse of Theorem 4 is considered in Section 5.1. In Section 6 we study the open problem of the existence of self-dual bent functions in odd number of variables and prove that such functions exist

for every  $q \equiv 0 \pmod 8$  (Theorem 7), in particular  $q = 2^k$  with  $k \geq 3$ . The Conclusion is in Section 7.

## 2 Notation

Let  $\mathbb{F}_2^n$  be a set of binary vectors of length  $n$ . For  $x, y \in \mathbb{F}_2^n$  denote  $\langle x, y \rangle = \bigoplus_{i=1}^n x_i y_i$ , where the sign  $\oplus$  denotes a sum modulo 2.

A *Boolean function*  $f$  in  $n$  variables is any map from  $\mathbb{F}_2^n$  to  $\mathbb{F}_2$ . The set of Boolean functions in  $n$  variables is denoted by  $\mathcal{F}_n$ . A *sign function* (also known as polyphase sequence or  $\{\pm 1\}$ -sequence) of  $f \in \mathcal{F}_n$  is an integer function  $F = (-1)^f$ , we will also refer to it as to an integer vector  $((-1)^{f_0}, (-1)^{f_1}, \dots, (-1)^{f_{2^n-1}})$  of length  $2^n$ , where  $(f_0, f_1, \dots, f_{2^n-1})$  is a vector of values (truth table) of the function  $f$ .

The *Hamming weight*  $\text{wt}(x)$  of the vector  $x \in \mathbb{F}_2^n$  is the number of nonzero coordinates of  $x$ . The *Hamming distance*  $\text{dist}(f, g)$  between Boolean functions  $f, g$  in  $n$  variables is the cardinality of the set  $\{x \in \mathbb{F}_2^n \mid f(x) \neq g(x)\}$ .

The Walsh–Hadamard transform of the function  $f \in \mathcal{F}_n$  is the integer function

$$W_f(y) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) \oplus \langle x, y \rangle}, \quad y \in \mathbb{F}_2^n.$$

A Boolean function  $f$  in an odd number  $m$  of variables is said to be *near-bent* if

$$W_f(y) \in \{0, \pm 2^{(m+1)/2}\}, \quad y \in \mathbb{F}_2^m.$$

For the case of an even number of variables, say  $n$ , the function  $f$  in  $n$  of variables is said to be *near-bent* if

$$W_f(y) \in \{0, \pm 2^{(n+2)/2}\}, \quad y \in \mathbb{F}_2^n.$$

A Boolean function  $f$  in an even number  $n$  of variables is said to be *bent* if

$$|W_f(y)| = 2^{n/2}, \quad y \in \mathbb{F}_2^n.$$

The set of bent functions in  $n$  variables is denoted by  $\mathcal{B}_n$ . The Boolean function  $\tilde{f} \in \mathcal{F}_n$  such that  $W_{\tilde{f}}(y) = (-1)^{\tilde{f}(y)} 2^{n/2}$  for any  $y \in \mathbb{F}_2^n$  is said to be *dual* of  $f$ . Note that the dual function is uniquely defined for every bent function, moreover the function  $\tilde{f}$  is bent as well. A bent function  $f$  is said to be *self-dual* if  $f = \tilde{f}$ , and *anti-self-dual* if  $f = \tilde{f} \oplus 1$ . The set of self-dual bent functions in  $n$  variables is denoted by  $\mathcal{SB}_n^+$ .

The *Rayleigh quotient* of a Boolean function in  $n$  variables is a number

$$S_f = \sum_{x,y \in \mathbb{F}_2^n} (-1)^{f(x) \oplus f(y) \oplus \langle x,y \rangle} = \sum_{y \in \mathbb{F}_2^n} (-1)^{f(y)} W_f(y).$$

This characteristics of a Boolean function in a scope of bent functions was studied in [7]. It is interesting for bent functions since it completely characterizes the Hamming distance between the function and its dual. Indeed, for any bent function  $f$  it holds

$$S_f = \sum_{y \in \mathbb{F}_2^n} (-1)^{f(y)} W_f(y) = 2^{n/2} \sum_{y \in \mathbb{F}_2^n} (-1)^{f(y) \oplus \tilde{f}(y)} = 2^{3n/2} - 2^{n/2+1} \text{dist}(f, \tilde{f}).$$

For a Boolean function  $f$  in  $n$  variables we call the number

$$\mathcal{S}_f = \frac{S_f}{2^{n/2}}.$$

the *sub-normalized Rayleigh quotient*. In terms of sign functions the Rayleigh quotient of  $f$  has the expression

$$S_f = \langle F, H_n F \rangle,$$

where  $F$  is its sign function.

Let  $I_n$  be the identity matrix of size  $n$  and  $H_n = H_1^{\otimes n}$  be the  $n$ -fold tensor product of the matrix  $H_1$  with itself, where

$$H_1 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

This matrix is known as Sylvester Hadamard matrix. It is known the Hadamard property of this matrix

$$H_n H_n^T = 2^n I_{2^n},$$

where  $H_n^T$  is transpose of  $H_n$  (it holds  $H_n^T = H_n$  by symmetricity of  $H_n$ ). Denote by  $\mathcal{H}_n = 2^{-n/2} H_n$  its normalized version. This matrix describes the Walsh–Hadamard transform in matrix form.

It is clear that sign functions if self-dual bent functions are eigenvectors of the normalized Sylvester Hadamard matrix that correspond to the eigenvalue 1. At the same time sign functions of anti-self-dual bent functions are eigenvectors of the normalized Sylvester Hadamard matrix that correspond to the eigenvalue  $(-1)$ . In terms of subspaces these facts imply that sign functions belong to the spaces  $\text{Ker}(\mathcal{H}_n - I_{2^n}) = \text{Ker}(H_n - 2^{n/2} I_{2^n})$  and  $\text{Ker}(\mathcal{H}_n + I_{2^n}) = \text{Ker}(H_n + 2^{n/2} I_{2^n})$  correspondingly.

### 3 Subfunctions in $n - 1$ variables

In this section we study the subfunctions of self-dual bent functions that are obtained by fixing the first variable. It is known that for any bent function in  $n$  variables such subfunctions are near-bent functions in  $n - 1$  variables with disjoint Walsh–Hadamard spectrum (see [34], for example).

In [5] and [8] the subfunctions in  $n - 1$  variables were used in proposed algorithms for the enumeration of all self-dual bent functions of prescribed degree. These algorithms explicitly exploit the fact that the vector  $(Y, Z)$ , where  $Y, Z \in \{\pm 1\}^{2^{n-1}}$ , is the sign function of some self-dual bent function in  $n$  variables if and only if

$$Y = Z + \frac{2H_{n-1}}{2^{n/2}}Z. \tag{1}$$

It is known [5] that for any Boolean function, say  $f$ , in even number  $n$  of variables it holds

$$|S_f| \leq 2^{3n/2}$$

with equality if and only if  $f$  is either self-dual  $(+2^{3n/2})$  or anti-self-dual  $(-2^{3n/2})$  bent. It follows that extremal values are achieved if and only if

$$(-1)^{f(y)}W_f(y) = 2^{n/2} \text{ for any } y \in \mathbb{F}_2^n$$

or

$$(-1)^{f(y)}W_f(y) = -2^{n/2} \text{ for any } y \in \mathbb{F}_2^n.$$

We are to introduce similar notation for the case of an odd number of variables based on the spectral characterization. Let  $m$  be an odd positive integer. We call a near-bent function  $g$  in  $m$  variables a *self-dual* if

$$(-1)^{g(y)}W_g(y) \geq 0 \text{ for any } y \in \mathbb{F}_2^m.$$

In order,  $g$  is called an *anti-self-dual* near-bent if

$$(-1)^{g(y)}W_g(y) \leq 0 \text{ for any } y \in \mathbb{F}_2^m.$$

An example of such function is the concatenation of an (anti-)self-dual bent function  $f$  with itself  $(f, f)$ .

Finding the exact maximal (minimal) value of the Rayleigh quotient of a Boolean function in an odd number  $m$  of variables is an open problem. Note that as was shown in [5], it holds

$$\max_{f \in \mathcal{F}_m} |S_f| \geq 2^{(3m-1)/2}.$$



The authors used the concatenation of two self-dual bent functions in  $m - 1$  variables, so the obtained value is called the *bent-concatenation bound*. Nevertheless the experiments have shown that this bound is not tight at least for small values of  $m$ .

**Proposition 1.** *Let  $g$  be a near-bent function in  $m$  variables, then*

$$|S_g| \leq 2^{(3m-1)/2}$$

*with equality if and only if  $g$  is either self-dual or anti-self-dual near-bent.*

*Proof.* By the definition of the Rayleigh quotient we have

$$S_g = \sum_{y \in \mathbb{F}_2^m} (-1)^{g(y)} W_g(y). \quad (2)$$

The multiplicities of Walsh coefficients of any near-bent function in  $m$  variables are well known (see [24], for example), we list them in Table 1.

Table 1: Multiplicities of Walsh coefficients of the near-bent function  $g$

Value	Size
0	$2^{m-1}$
$2^{(m+1)/2}$	$2^{m-2} + (-1)^{g(\mathbf{0})} 2^{(m-3)/2}$
$-2^{(m+1)/2}$	$2^{m-2} - (-1)^{g(\mathbf{0})} 2^{(m-3)/2}$

Consider two nonnegative integers  $a_1, a_2$  describing the signs of nonzero terms in the sum (2):

$$a_1 = \left| \left\{ y \in \mathbb{F}_2^m : (-1)^{g(y)} W_g(y) > 0 \right\} \right|,$$

$$a_2 = \left| \left\{ y \in \mathbb{F}_2^m : (-1)^{g(y)} W_g(y) < 0 \right\} \right|.$$

Then we have a following system

$$\begin{cases} 2^{(m+1)/2} a_1 - 2^{(m+1)/2} a_2 = S_g, \\ a_1 + a_2 = 2^{m-1}. \end{cases}$$

It is clear that the maximal value of  $S_g$  corresponds to the case when  $a_2 = 0$ . Then  $a_1 = 2^{m-1}$  and

$$S_g = 2^{(m+1)/2} \cdot 2^{m-1} = 2^{(3m-1)/2}.$$

By the same arguments the minimal value of  $S_g$  corresponds to the case when  $a_1 = 0$ . Then  $a_2 = 2^{m-1}$  and

$$S_g = \left( -2^{(m+1)/2} \right) \cdot 2^{m-1} = -2^{(3m-1)/2}.$$

Thus, it holds  $|S_g| \leq 2^{(3m-1)/2}$  with the equality only when either

$$(-1)^{g(y)}W_g(y) \geq 0 \text{ for any } y \in \mathbb{F}_2^m$$

or

$$(-1)^{g(y)}W_g(y) \leq 0 \text{ for any } y \in \mathbb{F}_2^m,$$

that is  $g$  is either self-dual or anti-self-dual near-bent.  $\square$

Thus, the value of the Rayleigh quotient of a self-dual near-bent function coincides with the bound for its maximal value, which is the best known one for today. Moreover, on self-dual near-bent functions and only on them the value of the Rayleigh quotient is maximal within the set of near-bent functions. Just the same holds for the minimal value and anti-self-dual near-bent functions.

Further we show that there exists a bijection between two types of self-duality with a descent step from even to odd number of variables.

**Theorem 1.** *There exists an one-to-one correspondence between the set of all self-dual bent functions in  $n \geq 4$  variables and the set of (anti-)self-dual near-bent functions in  $n - 1$  variables.*

*Proof.* Put  $\mathcal{H} = \mathcal{H}_{n-1}$ . Let  $f$  be a self-dual bent functions in  $n$  variables and  $(f_0, f_1)$  be its truth table, where  $f_i \in \mathcal{F}_{n-1}$ ,  $i = 1, 2$ . Denote by  $F_i$  the sign function of  $f_i$ ,  $i = 1, 2$ . Then it holds

$$\frac{1}{\sqrt{2}} \begin{pmatrix} \mathcal{H} & \mathcal{H} \\ \mathcal{H} & -\mathcal{H} \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \end{pmatrix} = \begin{pmatrix} F_0 \\ F_1 \end{pmatrix},$$

that is equal to the system

$$\begin{cases} \mathcal{H}F_0 + \mathcal{H}F_1 = \sqrt{2}F_0, \\ \mathcal{H}F_0 - \mathcal{H}F_1 = \sqrt{2}F_1. \end{cases} \quad (3)$$

Firstly one can notice that

$$\begin{aligned} \langle \sqrt{2}F_0, \sqrt{2}F_0 \rangle &= \langle \mathcal{H}F_0 + \mathcal{H}F_1, \mathcal{H}F_0 + \mathcal{H}F_1 \rangle \\ &= \langle \mathcal{H}F_0, \mathcal{H}F_0 \rangle + 2 \langle \mathcal{H}F_0, \mathcal{H}F_1 \rangle + \langle \mathcal{H}F_1, \mathcal{H}F_1 \rangle \\ &= \langle F_0, F_0 \rangle + 2 \langle F_0, F_1 \rangle + \langle F_1, F_1 \rangle \\ &= 2^{n-1} + 2 \langle F_0, F_1 \rangle + 2^{n-1} \\ &= 2 \cdot 2^{n-1}, \end{aligned}$$

therefore it holds  $\langle F_0, F_1 \rangle = 0$ . Now consider the first equation from the system above:

$$\mathcal{H}F_0 = -\mathcal{H}F_1 + \sqrt{2}F_0.$$

Since  $\mathcal{H}^2 = I_{n-1}$ , it is the same as

$$F_0 = -F_1 + \sqrt{2}\mathcal{H}F_0.$$

Consider the inner product

$$\langle F_0, F_0 \rangle = -\langle F_0, F_1 \rangle + \sqrt{2}\langle F_0, \mathcal{H}F_0 \rangle.$$

The orthogonality of  $F_0$  and  $F_1$  implies the condition

$$\sqrt{2}\langle F_0, \mathcal{H}F_0 \rangle = 2^{n-1}.$$

Under the used notation we have

$$S_{f_0} = \langle F_0, H_{n-1}F_0 \rangle = 2^{\frac{3(n-1)-1}{2}}.$$

By considering the second equation from (3) by the same way one can show that

$$S_{f_1} = \langle F_1, H_{n-1}F_1 \rangle = -2^{\frac{3(n-1)-1}{2}}.$$

Since from (1) it immediately follows that function  $f_1$  can be characterized by  $f_0$ , there exists an injective mapping from the set of all self-dual bent functions in  $n$  variables to the set of self-dual near-bent Boolean functions in  $n - 1$  variables. This mapping essentially maps every self-dual bent function to its subfunction obtained by fixing the first coordinate with 0.

Now let  $f_0$  be a self-dual near-bent Boolean function in  $n - 1$  variables. From Proposition 1 it follows that the value of  $(-1)^{f_0(y)}$  and the sign of the Walsh coefficient  $W_{f_0}(y)$  of  $f_0$  are agreed in a sense that their product is nonnegative for every  $y \in \mathbb{F}_2^{n-1}$ . Let  $F_0$  be a sign function of  $f_0$ . Define

$$F_1 = \frac{2H_{n-1}}{2^{n/2}}F_0 - F_0. \tag{4}$$

From (1) it follows that if  $F_1 \in \{\pm 1\}^{2^{n-1}}$ , then the vector  $(F_0, F_1)$  is the sign function of a self-dual bent function in  $n$  variables. Indeed, the relation (1) for  $(F_0, F_1)$  is

$$F_0 = \left( I_{2^{n-1}} + \frac{2H_{n-1}}{2^{n/2}} \right) F_1,$$

and its multiplication by  $\left( I_{2^{n-1}} - \frac{2H_{n-1}}{2^{n/2}} \right)$  from the left yields (4) since

$$\left( I_{2^{n-1}} + \frac{2H_{n-1}}{2^{n/2}} \right) \left( I_{2^{n-1}} - \frac{2H_{n-1}}{2^{n/2}} \right) = -I_{2^{n-1}}.$$

It is clear that  $W_{f_0}(y)$  and  $(-1)^{f_0(y)}$  are being agreed imply that

$$\left( \frac{2}{2^{n/2}} W_{f_0}(y) - (-1)^{f_0(y)} \right) \in \{\pm 1\}, \quad y \in \mathbb{F}_2^{n-1},$$

then we can define a Boolean function in  $n - 1$  variables, say  $f_1$ , which has a sign function  $F_1$  and consider the relation (4) in componentwise form

$$(-1)^{f_1(y)} = \frac{2}{2^{n/2}} W_{f_0}(y) - (-1)^{f_0(y)}, \quad y \in \mathbb{F}_2^{n-1}.$$

So the vector  $(F_0, F_1)$  is the sign function of a self-dual bent function in  $n$  variables. Note that this self-dual bent function is unique since the pair of subfunctions of any self-dual bent function is defined uniquely.

Thus, it follows that for any self-dual near-bent Boolean function in  $n - 1$  variables there exists a self-dual bent function in  $n$  variables, moreover this function is an unique one.

Finally, we have that the mapping that maps every self-dual bent function to its subfunction obtained by fixing the first coordinate with 0, is an injective and surjective one from the set of all self-dual bent functions in  $n$  variables to the set of self-dual near-bent functions in  $n - 1$  variables. Therefore there exists a bijection between these sets of Boolean functions.  $\square$

There is an interesting consequence that if we want to obtain Boolean function in even number  $n$  of variables that has the maximal value of the Rayleigh quotient, by using the concatenation of two Boolean functions in  $n - 1$  variables, we likely should not take functions that have extremal values of the Rayleigh quotient. The same holds if we are to construct a Boolean function in odd number  $m$  of variables that also has the maximal value of the Rayleigh quotient. Since in the first case we will not choose self-dual or anti-self-dual near-bent functions, therefore the obtained Boolean functions will not be self-dual bent. In the second case we obtain a bent-concatenation bound that is likely to be not tight.

The reason of that can be explained by the following. Assume we have the Boolean function  $f$  in  $k$  variables (even or odd) with sign function  $F$  which is a concatenation of functions  $f_0$  and  $f_1$  in  $k - 1$  variables. Then it holds

$$\begin{aligned} S_f &= \langle F, H_n F \rangle = \left\langle (F_0, F_1), \begin{pmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \end{pmatrix} \right\rangle \\ &= \left\langle (F_0, F_1), \begin{pmatrix} H_{n-1}(F_0 + F_1) \\ H_{n-1}(F_0 - F_1) \end{pmatrix} \right\rangle \end{aligned}$$

$$\begin{aligned}
 &= \langle F_0, H_{n-1}F_0 \rangle - \langle F_1, H_{n-1}F_1 \rangle + \langle F_0, H_{n-1}F_1 \rangle + \langle F_1, H_{n-1}F_0 \rangle \\
 &= S_{f_0} - S_{f_1} + \langle F_0, H_{n-1}F_1 \rangle + \langle F_1, H_{n-1}F_0 \rangle \\
 &= S_{f_0} - S_{f_1} + 2 \langle F_0, H_{n-1}F_1 \rangle,
 \end{aligned}$$

where we have different signs for the Rayleigh quotients of the subfunctions and also the term comprising both subfunctions, one of which is given in its Walsh–Hadamard transform form.

Thus, the maximization of the Rayleigh quotient of subfunctions may lead to the “instability” and decrease of the Rayleigh quotient of the whole function. The maximization of the Rayleigh quotient for the case of an odd number of variables is a problem of a complex optimization, in particular, of the values of the Rayleigh quotient of its subfunctions.

## 4 Properties of subfunctions in $n - 2$ variables

In this section we study the subfunctions of self-dual bent functions that are obtained by fixing the first and the second coordinates of the argument. Metrical properties of subfunctions and interconnections between them are considered.

Subfunctions of a bent function, in more general form, comprising the restriction of a bent function on all subspaces of codimension 2, were extensively studied in works [2, 3]. The considered sets of subfunctions were referred to as *4-decompositions* of a bent function. In particular, it was shown that such subfunctions of a bent function in  $n$  variables have the same extended Fourier spectrum: either all of them are bent, all are the three valued almost optimal (these are precisely near-bent functions with the spectrum having three values  $0, \pm 2^{n/2}$ ), or they have the same extended Fourier spectrum with five values  $0, \pm 2^{(n-2)/2}, \pm 2^{n/2}$ .

Throughout this section given a function  $f$  in  $m$  variables we will refer to four Boolean functions  $f_i, i = 0, 1, 2, 3$ , in  $m - 2$  variables as to its subfunctions obtained by fixing the first and the second coordinates of the argument with the values  $\{(00), (01), (10), (11)\}$ , correspondingly. In order, vector of values of  $f$  will have the form  $(f_0, f_1, f_2, f_3)$ . The sign function of  $f_i, i = 0, 1, 2, 3$ , will be denoted by  $F_i$ . Let the notation  $\mathcal{H}$  states for  $\mathcal{H}_{n-2}$ .

### 4.1 Concatenation of four bent functions

The case when all four subfunctions are bent essentially leads to the idea of an iterative construction of a bent function in  $n + 2$  variables through four

bent functions in  $n$  variables. In [27] Preneel et al. proved that given four bent functions  $f_i$ ,  $i = 0, 1, 2, 3$ , in  $n$  variables, the concatenation of vectors of values of  $f_i$  yields a bent function in  $n + 2$  variables if and only if

$$W_{f_0}(y)W_{f_1}(y)W_{f_2}(y)W_{f_3}(y) = -2^{2n} \text{ for any } y \in \mathbb{F}_2^n.$$

In terms of duals this condition is equivalent to the following

$$\tilde{f}_0(y) \oplus \tilde{f}_1(y) \oplus \tilde{f}_2(y) \oplus \tilde{f}_3(y) = 1 \text{ for any } y \in \mathbb{F}_2^n.$$

Note that the idea of concatenation also appears in a scope of so called “bent based” bent sequences, see [1]. The approach allows to obtain a bent sequence of length  $4l$  through the concatenation of four bent sequences of length  $l$  provided the similar conditions on these sequences are satisfied.

Bent functions in  $n + 2$  variables obtained by the concatenation of four bent functions in  $n$  variables were also studied in [31] from the point of view of obtaining lower bounds on the cardinality of the set of bent functions. Such functions were referred to as *bent iterative* functions.

There are known two constructions of self-dual bent functions in  $n + 2$  variables, based on the concatenation of four bent functions in  $n$  variables. They are

- the construction **C1**:

$$(f, \tilde{f}, \tilde{f}, f \oplus 1),$$

where  $f$  is bent function in  $n$  variables [5];

- the construction **C2**:

$$(f, g \oplus 1, g, f),$$

where  $f$  is a self-dual bent function,  $g$  is an anti-self-dual bent function both in  $n$  variables [12].

It is worth noting that the best known for today lower bound on the cardinality of the set of self-dual bent functions is the sum of cardinalities of **C1** and **C2**.

In [12] the criteria of self-duality of a bent function in  $n + 2$  variables obtained via concatenation of four bent functions in  $n$  variables was presented.

## 4.2 Metrical characteristics of the subfunctions of a bent function

In this subsection we will study the Gram matrix of vectors  $F_i$ ,  $i = 0, 1, 2, 3$  which are sign functions of subfunctions of a

Boolean function  $f$ . Recall that elements  $g_{ij}$  of the Gram matrix of vectors  $\{v_k\}_{k \in M} \subset \mathbb{R}^d$  are inner products between  $v_i$  and  $v_j$ ,  $i, j \in M$ . The determinant of the Gram matrix is called the Gramian of the corresponding system of vectors. The basic properties of the Gram matrix are:

- symmetricity;
- positive semi-definiteness;
- the Gramian is zero if and only if the vectors are linearly dependent.

The form of the Gram matrix of bent functions is characterized by the following

**Theorem 2.** *The Gram matrix of any bent function in  $n$  variables has form*

$$\begin{pmatrix} 2^{n-2} & b & b & -a \\ b & 2^{n-2} & a & -b \\ b & a & 2^{n-2} & -b \\ -a & -b & -b & 2^{n-2} \end{pmatrix},$$

for some even integers  $a, b$  such that

$$-2^{n-2} + 2|b| \leq a \leq 2^{n-2}.$$

*Proof.* Let  $f$  be a bent function in  $n$  variables, then

$$\frac{1}{2} \begin{pmatrix} \mathcal{H} & \mathcal{H} & \mathcal{H} & \mathcal{H} \\ \mathcal{H} & -\mathcal{H} & \mathcal{H} & -\mathcal{H} \\ \mathcal{H} & \mathcal{H} & -\mathcal{H} & -\mathcal{H} \\ \mathcal{H} & -\mathcal{H} & -\mathcal{H} & \mathcal{H} \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \end{pmatrix} = \begin{pmatrix} R_0 \\ R_1 \\ R_2 \\ R_3 \end{pmatrix}$$

or, equivalently,

$$\begin{cases} F_0 + F_1 + F_2 + F_3 = 2\mathcal{H}R_0, \\ F_0 - F_1 + F_2 - F_3 = 2\mathcal{H}R_1, \\ F_0 + F_1 - F_2 - F_3 = 2\mathcal{H}R_2, \\ F_0 - F_1 - F_2 + F_3 = 2\mathcal{H}R_3, \end{cases}$$

where  $R_i \in \{\pm 1\}^{2^{n-2}}$ ,  $i = 0, 1, 2, 3$ .

Denote  $g_{ij} = \langle F_i, F_j \rangle$ ,  $i, j = 0, 1, 2, 3$ , and consider pairwise inner products of right parts of all equations in the system above. The symmetricity of  $\text{Gram}(f)$  implies that we have at most six different coefficients outside the main diagonal in fact.

1) The 1st equation with itself

$$\begin{aligned} \langle 2\mathcal{H}R_0, 2\mathcal{H}R_0 \rangle &= \langle F_0 + F_1 + F_2 + F_3, F_0 + F_1 + F_2 + F_3 \rangle \\ &= \sum_{i,j=0}^3 g_{ij} = 4 \cdot 2^{n-2} + \sum_{\substack{i,j=0, \\ i \neq j}}^3 g_{ij} = 2^n. \end{aligned}$$

It yields the equation

$$g_{01} + g_{02} + g_{03} + g_{12} + g_{13} + g_{23} = 0;$$

2) The 2nd equation with itself yields

$$g_{01} - g_{02} + g_{03} + g_{12} - g_{13} + g_{23} = 0;$$

3) The 3d equation with itself yields

$$g_{01} - g_{02} - g_{03} - g_{12} - g_{13} + g_{23} = 0;$$

4) The 4th equation with itself yields

$$g_{01} + g_{02} - g_{03} - g_{12} + g_{13} + g_{23} = 0.$$

Finally we have the following system of equations that describe necessary relations between the entries of the Gram matrix

$$\begin{cases} g_{01} + g_{02} + g_{03} + g_{12} + g_{13} + g_{23} = 0, \\ g_{01} - g_{02} + g_{03} + g_{12} - g_{13} + g_{23} = 0, \\ g_{01} - g_{02} - g_{03} - g_{12} - g_{13} + g_{23} = 0, \\ g_{01} + g_{02} - g_{03} - g_{12} + g_{13} + g_{23} = 0. \end{cases}$$

The system has rank 4, its general solution is

$$\begin{aligned} g_{01} &= -g_{23}, & g_{02} &= -g_{23}, & g_{03} &= -g_{12}, \\ & & g_{12} &= g_{12}, & g_{13} &= g_{23}, \\ & & & & g_{23} &= g_{23} \end{aligned}$$

for  $g_{12}$  and  $g_{23}$  being free variables. Denote  $b = -g_{23}$  and  $a = g_{12}$ , then we obtain the desired form of the Gram matrix:

$$\text{Gram}(f) = \begin{pmatrix} 2^{n-2} & b & b & -a \\ b & 2^{n-2} & a & -b \\ b & a & 2^{n-2} & -b \\ -a & -b & -b & 2^{n-2} \end{pmatrix}.$$



Now we are to point essential bounds on values of  $a$  and  $b$  and deduce some relations between them. In order to do it recall that any Gram matrix is positive semi-definite, hence all its eigenvalues must be nonnegative. The matrix  $\text{Gram}(f)$  has four eigenvalues, they are

$$\begin{aligned}\lambda_{1,2} &= 2^{n-2} - a, \\ \lambda_3 &= 2^{n-2} + a - 2b, \\ \lambda_4 &= 2^{n-2} + a + 2b.\end{aligned}$$

Note that the eigenvalue  $2^{n-2} - a$  has algebraic multiplicity 2, also its non-negativity is obvious. The rest imply that

$$\begin{aligned}a &\geq -2^{n-2} + 2b, \\ a &\geq -2^{n-2} - 2b,\end{aligned}$$

that is

$$a \geq -2^{n-2} + \max\{2b, -2b\},$$

and, consequently,

$$a \geq -2^{n-2} + 2|b|,$$

where  $|b|$  is essentially bounded by  $2^{n-2}$  from above. □

If  $f$  is a self-dual bent function in  $n$  variables, then we have

$$\frac{1}{2} \begin{pmatrix} \mathcal{H} & \mathcal{H} & \mathcal{H} & \mathcal{H} \\ \mathcal{H} & -\mathcal{H} & \mathcal{H} & -\mathcal{H} \\ \mathcal{H} & \mathcal{H} & -\mathcal{H} & -\mathcal{H} \\ \mathcal{H} & -\mathcal{H} & -\mathcal{H} & \mathcal{H} \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \end{pmatrix} = \begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \end{pmatrix}$$

or, equivalently,

$$\begin{cases} F_0 + F_1 + F_2 + F_3 = 2\mathcal{H}F_0, \\ F_0 - F_1 + F_2 - F_3 = 2\mathcal{H}F_1, \\ F_0 + F_1 - F_2 - F_3 = 2\mathcal{H}F_2, \\ F_0 - F_1 - F_2 + F_3 = 2\mathcal{H}F_3. \end{cases}$$

In this case we can use another inner products between right parts of the system of equations that describe necessary relations between the elements of the Gram matrix. Joined with the previous four equations, new equations

yield the system

$$\begin{cases} g_{01} + g_{02} + g_{03} + g_{12} + g_{13} + g_{23} = 0, \\ g_{01} - g_{02} + g_{03} + g_{12} - g_{13} + g_{23} = 0, \\ g_{01} - g_{02} - g_{03} - g_{12} - g_{13} + g_{23} = 0, \\ g_{01} + g_{02} - g_{03} - g_{12} + g_{13} + g_{23} = 0, \\ 2g_{01} = g_{02} - g_{13}, \\ 2g_{02} = g_{01} - g_{23}, \\ g_{03} = -g_{12}, \\ 2g_{13} = g_{23} - g_{01}, \\ 2g_{23} = g_{13} - g_{02}. \end{cases}$$

The system also has rank 4, so we obtain the same form of the Gram matrix as in Theorem 2.

For example, the constructions **C1** and **C2** provide the following matrices:

$$\text{Gram}(\mathbf{C1}) = \begin{pmatrix} 2^{n-2} & \mathcal{S}_f & \mathcal{S}_f & -2^{n-2} \\ \mathcal{S}_f & 2^{n-2} & 2^{n-2} & -\mathcal{S}_f \\ \mathcal{S}_f & 2^{n-2} & 2^{n-2} & -\mathcal{S}_f \\ -2^{n-2} & -\mathcal{S}_f & -\mathcal{S}_f & 2^{n-2} \end{pmatrix},$$

which has rank 1 in the case when  $\mathcal{S}_f = 2^{n-2}$  that is  $f$  is self-dual bent, and 2 otherwise, and

$$\text{Gram}(\mathbf{C2}) = \begin{pmatrix} 2^{n-2} & 0 & 0 & 2^{n-2} \\ 0 & 2^{n-2} & -2^{n-2} & 0 \\ 0 & -2^{n-2} & 2^{n-2} & 0 \\ 2^{n-2} & 0 & 0 & 2^{n-2} \end{pmatrix}$$

with rank equal to 2. It is obvious that for both constructions the sets  $\{F_i\}$  are linearly dependent.

Inner products between sign functions are interesting since it is easy to deduce the Hamming distance between two Boolean functions provided the inner product between their sign functions is known. Indeed,

$$\text{dist}(f_i, f_j) = 2^{n-3} - \frac{1}{2} \langle F_i, F_j \rangle = 2^{n-3} - \frac{1}{2} g_{ij}, \quad i, j = 0, 1, 2, 3.$$

Thus, Theorem 2 can be reformulated in terms of Hamming distances between subfunctions:

**Theorem 3.** *Let  $f$  be a bent function in  $n$  variables. The distances between  $\{f_i\}_{i=0}^3$  are characterized by the matrix*

$$\text{Dist}(f) = \begin{pmatrix} 0 & 0 & 0 & 2^{n-2} \\ 0 & 0 & 0 & 2^{n-2} \\ 0 & 0 & 0 & 2^{n-2} \\ 2^{n-2} & 2^{n-2} & 2^{n-2} & 0 \end{pmatrix} + \begin{pmatrix} 0 & d_1 & d_1 & -d_2 \\ d_1 & 0 & d_2 & -d_1 \\ d_1 & d_2 & 0 & -d_1 \\ -d_2 & -d_1 & -d_1 & 0 \end{pmatrix}$$

for some positive even integers  $d_1, d_2$  such that

$$|2^{n-2} - 2d_1| \leq 2^{n-2} - d_2.$$

*Proof.* The relation between the inner product and the Hamming distance yields the matrix whereas the inequality

$$|2^{n-2} - 2d_1| \leq 2^{n-2} - d_2$$

is obtained from

$$-2^{n-2} + 2|b| \leq a \leq 2^{n-2}$$

with

$$\begin{aligned} b &= 2^{n-2} - 2d_1, \\ a &= 2^{n-2} - 2d_2. \end{aligned}$$

□

### 4.3 Rayleigh quotients of subfunctions

Let  $f$  be a self-dual bent function in  $n$  variables. Recall that we have

$$\begin{cases} F_0 + F_1 + F_2 + F_3 = 2\mathcal{H}F_0, \\ F_0 - F_1 + F_2 - F_3 = 2\mathcal{H}F_1, \\ F_0 + F_1 - F_2 - F_3 = 2\mathcal{H}F_2, \\ F_0 - F_1 - F_2 + F_3 = 2\mathcal{H}F_3. \end{cases}$$

Consider four inner products using the equations above

$$\begin{cases} \langle F_0, F_0 \rangle + \langle F_0, F_1 \rangle + \langle F_0, F_2 \rangle + \langle F_0, F_3 \rangle = \langle F_0, 2\mathcal{H}F_0 \rangle, \\ \langle F_1, F_0 \rangle - \langle F_1, F_1 \rangle + \langle F_1, F_2 \rangle - \langle F_1, F_3 \rangle = \langle F_1, 2\mathcal{H}F_1 \rangle, \\ \langle F_2, F_0 \rangle + \langle F_2, F_1 \rangle - \langle F_2, F_2 \rangle - \langle F_2, F_3 \rangle = \langle F_2, 2\mathcal{H}F_2 \rangle, \\ \langle F_3, F_0 \rangle - \langle F_3, F_1 \rangle - \langle F_3, F_2 \rangle + \langle F_3, F_3 \rangle = \langle F_3, 2\mathcal{H}F_3 \rangle. \end{cases}$$

The Gram matrix provides the expression of the Rayleigh quotients of the subfunctions in terms of the coefficients  $a$  and  $b$ .

$$\begin{cases} 2^{n-2} + 2b - a = \langle F_0, 2\mathcal{H}F_0 \rangle, \\ -2^{n-2} + a + 2b = \langle F_1, 2\mathcal{H}F_1 \rangle, \\ -2^{n-2} + 2b + a = \langle F_2, 2\mathcal{H}F_2 \rangle, \\ 2^{n-2} - a + 2b = \langle F_3, 2\mathcal{H}F_3 \rangle. \end{cases}$$

Finally we have expressions

$$\begin{aligned} S_{f_0} &= 2^{n/2-2} (2^{n-2} - a + 2b), & S_{f_1} &= 2^{n/2-2} (-2^{n-2} + a + 2b), \\ S_{f_2} &= 2^{n/2-2} (-2^{n-2} + a + 2b), & S_{f_3} &= 2^{n/2-2} (2^{n-2} - a + 2b), \end{aligned}$$

and

$$S_{f_0} + S_{f_1} = S_{f_2} + S_{f_3} = 2^{n/2}b.$$

It follows that the Rayleigh quotients of  $f_0$  and  $f_3$  coincide, as well as of  $f_1$  and  $f_2$ . The sum of all Rayleigh quotients is equal to

$$S_{f_0} + S_{f_1} + S_{f_2} + S_{f_3} = 2^{n/2+1}b.$$

We collect all this to the following statement

**Proposition 2.** *Let  $f$  be a self-dual bent function in  $n$  variables with the Gram matrix*

$$\text{Gram}(f) = \begin{pmatrix} 2^{n-2} & b & b & -a \\ b & 2^{n-2} & a & -b \\ b & a & 2^{n-2} & -b \\ -a & -b & -b & 2^{n-2} \end{pmatrix},$$

then

$$\begin{aligned} S_{f_0} &= S_{f_3} = 2^{n/2-2} (2^{n-2} - a + 2b), \\ S_{f_1} &= S_{f_2} = 2^{n/2-2} (-2^{n-2} + a + 2b). \end{aligned}$$

#### 4.4 Sufficient condition for the subfunctions of self-dual bent function to be bent

In this subsection we study the special cases of parameters  $a, b$  for which the Gram matrix is singular.

Recall that the Gramian is equal to the multiplication of the eigenvalues of the matrix so for a bent function  $f$  with the Gram matrix  $\text{Gram}(f)$  it has the following expression

$$\det(\text{Gram}(f)) = (2^{n-2} - a)^2 (2^{n-2} + a - 2b) (2^{n-2} + a + 2b). \quad (5)$$

Further the values such that the Gramian is zero will be considered for a self-dual case.

But before, we can characterize all self-dual bent functions that possess  $a = 2^{n-2}$ , that is  $f_1 = f_2$ . In order to do it consider the general system

$$\begin{cases} F_0 + F_1 + F_2 + F_3 = 2\mathcal{H}F_0, \\ F_0 - F_1 + F_2 - F_3 = 2\mathcal{H}F_1, \\ F_0 + F_1 - F_2 - F_3 = 2\mathcal{H}F_2, \\ F_0 - F_1 - F_2 + F_3 = 2\mathcal{H}F_3, \end{cases}$$

which is transformed to

$$\begin{cases} F_0 + 2F_1 + F_3 = 2\mathcal{H}F_0, \\ F_0 - F_3 = 2\mathcal{H}F_1, \\ F_0 - F_3 = 2\mathcal{H}F_1, \\ F_0 - 2F_1 + F_3 = 2\mathcal{H}F_3. \end{cases}$$

By the triangle inequality we obtain

$$\|F_0 - F_3\| \leq \|F_0\| + \|F_3\| = 2 \cdot 2^{(n-1)/2} = 2^{(n+1)/2}.$$

From the other side by orthogonality of the matrix  $\mathcal{H}$  we obtain that  $\|2\mathcal{H}F_1\| = 2 \cdot 2^{(n-1)/2} = 2^{(n+1)/2}$ . So we have an equality

$$\|F_0 - F_3\| = \|F_0\| + \|F_3\|,$$

hence  $F_0$  and  $F_3$  are linearly dependent vectors, that is either  $F_0 = F_3$  or  $F_0 = -F_3$ . But from the second and third equalities it follows that  $F_0$  and  $F_3$  can not coincide, therefore  $F_3 = -F_0$ . Finally we obtain  $F_0 = \mathcal{H}F_1$ , that is all subfunctions are bent and  $f_0$  and  $f_1$  are dual of each other. This situation is exactly the construction **C1**.

**Proposition 3.** *If for a self-dual bent function  $f$  it holds  $f_1 = f_2$ , then it is constructed via **C1**.*

In Section 4.2 it was mentioned that sign functions of subfunctions mentioned in constructions **C1** and **C2** are linearly dependent. Also all that subfunctions are bent. The next results covers all combinations for which the Gramian is zero.

**Theorem 4.** *If the Gram matrix of a self-dual bent function  $f$  is singular then all subfunctions are bent.*

*Proof.* At first notice that the condition (1) for the case of subfunctions in  $n - 2$  variables has the following form

$$\begin{pmatrix} F_0 \\ F_1 \end{pmatrix} = \begin{pmatrix} F_2 \\ F_3 \end{pmatrix} + \begin{pmatrix} \mathcal{H} & \mathcal{H} \\ \mathcal{H} & -\mathcal{H} \end{pmatrix} \begin{pmatrix} F_2 \\ F_3 \end{pmatrix}. \quad (6)$$

Also by the condition  $\mathcal{H}^2 = I_{2^{n-2}}$  we obtain

$$\begin{pmatrix} \mathcal{H}F_0 \\ \mathcal{H}F_1 \end{pmatrix} = \begin{pmatrix} \mathcal{H}F_2 \\ \mathcal{H}F_3 \end{pmatrix} + \begin{pmatrix} F_2 + F_3 \\ F_2 - F_3 \end{pmatrix}. \quad (7)$$

Both of conditions (6) and (7) allow to characterize all possible combinations of signs of  $F_i$  and  $\mathcal{H}F_i$ ,  $i = 0, 1, 2, 3$ . As it was mentioned in Section 4.1, either all of subfunctions are bent, all are near-bent, or they have the same extended Fourier spectrum with five values  $0, \pm 2^{(n-2)/2}, 2^{n/2}$  [2, 3]. It means that in general case  $\mathcal{H}F_i \in \{0, \pm 1, \pm 2\}$ ,  $i = 0, 1, 2, 3$ . For every row of the

Table 2: All possible relations between values of subfunctions

$i$	$F_0(y)$	$F_1(y)$	$F_2(y)$	$F_3(y)$	$\mathcal{H}F_0(y)$	$\mathcal{H}F_1(y)$	$\mathcal{H}F_2(y)$	$\mathcal{H}F_3(y)$
1	+1	+1	+1	+1	+2	0	0	0
2	+1	-1	+1	-1	0	+2	0	0
3	-1	+1	-1	+1	0	-2	0	0
4	-1	-1	-1	-1	-2	0	0	0
5	-1	+1	+1	-1	0	0	0	-2
6	+1	-1	-1	+1	0	0	0	+2
7	+1	+1	-1	-1	0	0	+2	0
8	-1	-1	+1	+1	0	0	-2	0
9	+1	+1	-1	+1	+1	-1	+1	+1
10	+1	-1	-1	-1	-1	+1	+1	+1
11	-1	+1	+1	+1	+1	-1	-1	-1
12	-1	-1	+1	-1	-1	+1	-1	-1
13	+1	+1	+1	-1	+1	+1	+1	-1
14	-1	+1	-1	-1	-1	-1	+1	-1
15	+1	-1	+1	+1	+1	+1	-1	+1
16	-1	-1	-1	+1	-1	-1	-1	+1

table above by  $c_i$  we denote the number of vectors  $y \in \mathbb{F}_2^{n-2}$  for which the corresponding sequence of values and signs stands. The Gram matrix from Theorem 2 for the function  $f$  gives six equations:

$$\begin{aligned} \langle F_0, F_1 \rangle &= c_1 - c_2 - c_3 + c_4 - c_5 - c_6 + c_7 + c_8 + c_9 - c_{10} - c_{11} + c_{12} \\ &\quad + c_{13} - c_{14} - c_{15} + c_{16} = b, \end{aligned}$$

$$\langle F_0, F_2 \rangle = c_1 + c_2 + c_3 + c_4 - c_5 - c_6 - c_7 - c_8 - c_9 - c_{10} - c_{11} - c_{12}$$

$$+ c_{13} + c_{14} + c_{15} + c_{16} = b,$$

$$\begin{aligned} \langle F_0, F_3 \rangle &= c_1 - c_2 - c_3 + c_4 + c_5 + c_6 - c_7 - c_8 + c_9 - c_{10} - c_{11} + c_{12} \\ &\quad - c_{13} + c_{14} + c_{15} - c_{16} = -a, \end{aligned}$$

$$\begin{aligned} \langle F_1, F_2 \rangle &= c_1 - c_2 - c_3 + c_4 + c_5 + c_6 - c_7 - c_8 - c_9 + c_{10} + c_{11} - c_{12} \\ &\quad + c_{13} - c_{14} - c_{15} + c_{16} = a, \end{aligned}$$

$$\begin{aligned} \langle F_1, F_3 \rangle &= c_1 + c_2 + c_3 + c_4 - c_5 - c_6 - c_7 - c_8 + c_9 + c_{10} + c_{11} + c_{12} \\ &\quad - c_{13} - c_{14} - c_{15} - c_{16} = -b, \end{aligned}$$

$$\begin{aligned} \langle F_2, F_3 \rangle &= c_1 - c_2 - c_3 + c_4 - c_5 - c_6 + c_7 + c_8 - c_9 + c_{10} + c_{11} - c_{12} \\ &\quad - c_{13} + c_{14} + c_{15} - c_{16} = -b. \end{aligned}$$

Finally, taking into account the cardinality of the space  $\mathbb{F}_2^{n-2}$ , we obtain the system of 7 linear equations

$$\left\{ \begin{array}{l} c_1 - c_2 - c_3 + c_4 - c_5 - c_6 + c_7 + c_8 + c_9 - c_{10} - c_{11} + c_{12} + c_{13} - c_{14} - c_{15} + c_{16} = b \\ c_1 + c_2 + c_3 + c_4 - c_5 - c_6 - c_7 - c_8 - c_9 - c_{10} - c_{11} - c_{12} + c_{13} + c_{14} + c_{15} + c_{16} = b \\ c_1 - c_2 - c_3 + c_4 + c_5 + c_6 - c_7 - c_8 + c_9 - c_{10} - c_{11} + c_{12} - c_{13} + c_{14} + c_{15} - c_{16} = -a \\ c_1 - c_2 - c_3 + c_4 + c_5 + c_6 - c_7 - c_8 - c_9 + c_{10} + c_{11} - c_{12} + c_{13} - c_{14} - c_{15} + c_{16} = a \\ c_1 + c_2 + c_3 + c_4 - c_5 - c_6 - c_7 - c_8 + c_9 + c_{10} + c_{11} + c_{12} - c_{13} - c_{14} - c_{15} - c_{16} = -b \\ c_1 - c_2 - c_3 + c_4 - c_5 - c_6 + c_7 + c_8 - c_9 + c_{10} + c_{11} - c_{12} - c_{13} + c_{14} + c_{15} - c_{16} = -b \\ c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8 + c_9 + c_{10} + c_{11} + c_{12} + c_{13} + c_{14} + c_{15} + c_{16} = 2^{n-2} \end{array} \right.$$

The system has rank 7, its equations in a row echelon form yield the relations

$$\begin{aligned} c_1 + c_4 + c_{14} + c_{15} &= \frac{2^{n-2} - a}{4}, \\ c_2 + c_3 + c_{14} + c_{15} &= \frac{2^{n-2} - a}{4}, \\ c_5 + c_6 + c_{14} + c_{15} &= \frac{2^{n-2} - a}{4}, \\ c_7 + c_8 + c_{14} + c_{15} &= \frac{2^{n-2} - a}{4}, \\ c_9 + c_{12} - c_{14} - c_{15} &= 0, \\ c_{10} + c_{11} - c_{14} - c_{15} &= \frac{a - b}{2}, \\ c_{13} - c_{14} - c_{15} + c_{16} &= \frac{a + b}{2}. \end{aligned}$$

Now we are to consider all combinations of  $a, b$  such that the Gramian (5) is zero. In order to do it we consider  $c_i, i = 1, 2, \dots, 16$ , as nonnegative integer variables. Before one can note that one of subfunctions is bent (consequently all of them are bent) if and only if  $c_i = 0$  for  $i = 1, 2, \dots, 8$ . So we introduce an auxiliary equation

$$c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8 = k,$$

where  $k$  is nonnegative integer and put it to the system. The rank of the resulting system of equations is 8. From the nonnegativity of variables it follows that if for a fixed pair  $a, b$  provided by some self-dual bent function, the system has no solutions with positive  $k$ , all subfunctions of this function are bent.

At first, note that for every eigenvalue of the Gram matrix given in the general form there exists a self-dual bent function with  $a, b$  such that the eigenvalue is zero. Indeed, for  $\lambda_{1,2} = 2^{n-2} - a = 0$  the construction **C1** is suitable, since it provides  $a = 2^{n-2}, b = \mathcal{S}_f$ . For  $\lambda_3 = 2^{n-2} + a - 2b = 0$  and  $\lambda_4 = 2^{n-2} + a + 2b = 0$  the construction **C2** meets the desired condition because it is clear that it admits  $a = -2^{n-2}, b = 0$ .

Now consider all pairs of  $a, b$ , vanishing the corresponding eigenvalue, and analyze the obtained system:

–  $2^{n-2} - a = 0$ : in this case

$$c_1 = c_2 = c_3 = c_4 = c_5 = c_6 = c_7 = c_8 = 0,$$

that holds if only if  $k = 0$ .

–  $2^{n-2} + a - 2b = 0$ : the relations between variables must satisfy

$$c_1 + c_4 = \frac{k}{4}, \quad c_2 + c_3 = \frac{k}{4}, \quad c_5 + c_6 = \frac{k}{4}, \quad c_7 + c_8 = \frac{k}{4}, \quad c_{10} + c_{11} = -\frac{k}{4},$$

so the solution does not exist if  $k > 0$ , therefore  $k = 0$ .

–  $2^{n-2} + a + 2b = 0$ : we have relations

$$c_1 + c_4 = \frac{k}{4}, \quad c_2 + c_3 = \frac{k}{4}, \quad c_5 + c_6 = \frac{k}{4}, \quad c_7 + c_8 = \frac{k}{4}, \quad c_{13} + c_{16} = -\frac{k}{4},$$

so again the solution does not exist if  $k > 0$ , therefore  $k = 0$ .

□

From Theorem 4 and properties of the Gram matrix we conclude that



**Corollary 1.** *If sign functions of subfunctions of a self-dual bent function are linearly dependent then all subfunctions are bent.*

Thus, we obtain a sufficient condition for bentness of the subfunctions. The interesting question arises: is it **necessary** to have linear dependence for sign functions of subfunctions in order to obtain a self-dual bent function with bent subfunctions? In particular, the experiments show that

**Remark 1.** *For  $n = 4$  all self-dual bent functions with bent subfunctions have singular Gram matrices.*

We consider this question further in Section 5.1.

From Table 2 we can also deduce an interesting property that can be useful

**Proposition 4.** *Let  $f$  be a self-dual bent function in  $n$  variables, then  $\{f_i\}_{i=0}^3$  are bent if and only if*

$$f_0(y) \oplus f_1(y) \oplus f_2(y) \oplus f_3(y) = 1 \text{ for any } y \in \mathbb{F}_2^{n-2}.$$

This condition does not follow directly from the results of [12] but can be deduced from [3]. By using it we can clarify the decomposition of a self-dual bent function with bent subfunctions:

$$\begin{aligned} f(y_1, y_2, x) &= (y_1 \oplus 1)(y_2 \oplus 1)f_0(x) \oplus (y_1 \oplus 1)y_2f_1(x) \\ &\quad \oplus y_1(y_2 \oplus 1)f_2(x) \oplus y_1y_2f_3(x) \\ &= y_1y_2(f_0(x) \oplus f_1(x) \oplus f_2(x) \oplus f_3(x)) \\ &\quad \oplus y_1(f_0(x) \oplus f_2(x)) \oplus y_2(f_0(x) \oplus f_1(x)) \oplus f_0(x) \\ &= f_0 \oplus y_1(f_0 \oplus f_2) \oplus y_2(f_0 \oplus f_1) \oplus y_1y_2, \quad y_1, y_2 \in \mathbb{F}_2, x \in \mathbb{F}_2^{n-2}. \end{aligned}$$

## 5 New iterative constructions and lower bound for the cardinality of the set of self-dual bent functions

At first, we propose three new constructions **C3**, **C4** and **C5** of self-dual bent functions. The constructions use a 4-variables step. Let  $h$  be a bent function in  $n - 4$  variables,  $f$  be a self-dual bent function in  $n - 4$  variables and  $g$  be an anti-self-dual bent function in  $n - 4$  variables.

– the construction **C3**:

$$(h, g, g \oplus 1, h, \tilde{h}, f, f \oplus 1, \tilde{h}, \tilde{h}, f \oplus 1, f, \tilde{h}, h \oplus 1, g, g \oplus 1, h \oplus 1)$$

It is clear that the subfunctions in  $n - 2$  variables are bent;

– the construction **C4**:

$$(h, g, \tilde{h}, f, g \oplus 1, h, f \oplus 1, \tilde{h}, \tilde{h}, f \oplus 1, h \oplus 1, g, f, \tilde{h}, g \oplus 1, h \oplus 1)$$

The subfunctions in  $n - 2$  variables are bent if and only if  $h \oplus \tilde{h} \oplus f \oplus g = 0$ , so in some cases we do not obtain bent decompositions. Thus, this construction also yields a class of bent functions which cannot be decomposed into the concatenation of four bent functions;

– the construction **C5**:

$$(h, h \oplus 1, \tilde{h}, \tilde{h}, h, h, \tilde{h} \oplus 1, \tilde{h}, \tilde{h}, \tilde{h}, h \oplus 1, h, \tilde{h} \oplus 1, \tilde{h}, h \oplus 1, h \oplus 1)$$

It is clear that the subfunctions in  $n - 2$  variables are bent.

It is possible to slightly estimate the **C4** case related with the (im)possibility of a bent decomposition:

**Proposition 5.** *The number of self-dual bent functions in  $n \geq 8$  variables constructed via **C4**, which cannot be (can be) decomposed into the concatenation of four bent functions, is at least  $2 |\mathcal{SB}_{n-6}^+|^2$ .*

*Proof.* Let  $r_1$  and  $r_2$  be two bent functions in  $n - 6$  variables  $x_7, x_8, \dots, x_n$ , such that the first one is either self-dual or anti-self-dual while the second is a self-dual one. Define

$$\begin{aligned} f(x_5, x_6, x_7, \dots, x_n) &= x_5 x_6 \oplus r_1(x_7, x_8, \dots, x_n), \\ g(x_5, x_6, x_7, \dots, x_n) &= x_5 x_6 \oplus x_5 \oplus x_6 \oplus r_1(x_7, x_8, \dots, x_n), \\ h(x_5, x_6, x_7, \dots, x_n) &= x_5 x_6 \oplus x_5 \oplus r_2(x_7, x_8, \dots, x_n), \end{aligned}$$

one can check that

$$\tilde{h}(x_5, x_6, x_7, \dots, x_n) = x_5 x_6 \oplus x_6 \oplus r_2(x_7, x_8, \dots, x_n).$$

Thus, the self-dual bent function constructed via **C4**, is the concatenation of four bent functions.

Finally, note that in order to obtain the function which is not the concatenation of four bent functions, it is enough to take a negation of the two-variables part of either  $f$  or  $g$ , for instance

$$f(x_5, x_6, x_7, \dots, x_n) = x_5 x_6 \oplus 1 \oplus r_1(x_7, x_8, \dots, x_n).$$

□

The constructions **C3**, **C4** and **C5** have the following Gram matrices:

$$\begin{aligned} \text{Gram}(\mathbf{C3}) &= \begin{pmatrix} 2^{n-2} & 2\mathcal{S}_h & 2\mathcal{S}_h & 0 \\ 2\mathcal{S}_h & 2^{n-2} & 0 & -2\mathcal{S}_h \\ 2\mathcal{S}_h & 0 & 2^{n-2} & -2\mathcal{S}_h \\ 0 & -2\mathcal{S}_h & -2\mathcal{S}_h & 2^{n-2} \end{pmatrix}, \\ \text{Gram}(\mathbf{C4}) &= \begin{pmatrix} 2^{n-2} & 0 & 0 & 0 \\ 0 & 2^{n-2} & 0 & 0 \\ 0 & 0 & 2^{n-2} & 0 \\ 0 & 0 & 0 & 2^{n-2} \end{pmatrix}, \\ \text{Gram}(\mathbf{C5}) &= \begin{pmatrix} 2^{n-2} & 0 & 0 & -4\mathcal{S}_h \\ 0 & 2^{n-2} & 4\mathcal{S}_h & 0 \\ 0 & 4\mathcal{S}_h & 2^{n-2} & 0 \\ -4\mathcal{S}_h & 0 & 0 & 2^{n-2} \end{pmatrix} \end{aligned}$$

with parameters  $a = 0, b = 2\mathcal{S}_h, a = b = 0$  and  $a = 4\mathcal{S}_h, b = 0$ , correspondingly. The Gramian of **C3** is equal to  $2^{4n-8} - 2^{2n}\mathcal{S}_h^2$  hence it is nonzero besides the case  $|\mathcal{S}_h| = 2^{n-4}$ , that is when  $h$  is either self-dual or anti-self-dual bent. The Gramian of **C4** is equal to  $2^{4n-8}$ . The Gramian of **C5** is equal to  $(2^{2n-4} - 16\mathcal{S}_h^2)^2$  that is it is nonzero besides the case  $|\mathcal{S}_h| = 2^{n-4}$ , that is again when  $h$  is either self-dual or anti-self-dual bent.

Note that the constructions **C1**, **C2**, **C3** and **C4** provide disjoint sets of self-dual bent functions whereas **C5** has clear intersection with **C1** and **C2**. So we conclude that the sum of the cardinalities of the first four constructions and the disjoint part of **C5** is a lower bound for the cardinality of the set of self-dual bent functions in  $n$  variables.

**Theorem 5.** *The number of self-dual bent functions in  $n \geq 6$  variables is at least*

$$|\mathcal{B}_{n-2}| + |\mathcal{SB}_{n-2}^+|^2 + |\mathcal{B}_{n-4}| \left( 2|\mathcal{SB}_{n-4}^+|^2 + 1 \right) - 2|\mathcal{SB}_{n-4}^+|.$$

Thus, it increases the previous iterative bound  $|\mathbf{C1}| + |\mathbf{C2}|$  by the summand that corresponds to the constructions that exploit functions in  $n - 4$  variables.

### 5.1 Linear independence of sign functions and bentness of the subfunctions of self-dual bent function

Here we consider the converse of Theorem 4. Now we can give an answer to the question about necessity of singularity of the Gram matrix for the subfunctions to be bent:

**Theorem 6.** *For every even  $n \geq 6$  there exist self-dual bent functions in  $n$  variables with invertible Gram matrices, such that all their subfunctions are bent.*

*Proof.* The proof is just the usage of partial cases of the constructions that we introduce in current work.  $\square$

From Theorem 6 and properties of the Gram matrix we again conclude that

**Corollary 2.** *For every even  $n \geq 6$  there exist self-dual bent functions in  $n$  variables whose subfunctions are bent functions with linearly independent sign functions.*

Thus, the converse of Theorem 4 does not hold for  $n \geq 6$ , that is the linear dependence of sign functions provides only **sufficient** condition for subfunctions in  $n - 2$  variables to be bent.

## 6 Existence of self-dual generalized bent functions in odd number of variables

A *generalized Boolean function*  $f$  in  $n$  variables is any map from  $\mathbb{F}_2^n$  to  $\mathbb{Z}_q$ , the integers modulo  $q$ . The set of generalized Boolean functions in  $n$  variables is denoted by  $\mathcal{GF}_n^q$ . Let  $\omega = e^{2\pi i/q}$ . The (*generalized*) *Walsh-Hadamard transform* of  $f \in \mathcal{GF}_n^q$  is the complex valued function

$$H_f(y) = \sum_{x \in \mathbb{F}_2^n} \omega^{f(x)} (-1)^{\langle x, y \rangle},$$

where  $y \in \mathbb{F}_2^n$ . A generalized Boolean function  $f$  in  $n$  variables is said to be *generalized bent* (gbent) if  $|H_f(y)| = 2^{n/2}$  for all  $y \in \mathbb{F}_2^n$  [29]. If there exists such  $\tilde{f} \in \mathcal{GF}_n^q$  that  $H_f(y) = \omega^{\tilde{f}(y)} 2^{n/2}$  for any  $y \in \mathbb{F}_2^n$ , the gbent function  $f$  is said to be *regular* and  $\tilde{f}$  is called its *dual*. Note that  $\tilde{f}$  is generalized bent as well. A regular gbent function  $f$  is said to be *self-dual* if  $f = \tilde{f}$ , and *anti-self-dual* if  $f = \tilde{f} + q/2$ . Consequently, it is the case only for even  $q$ .

The problem of the existence of (regular) gbent functions is non-trivial for different combinations of parities of  $n$  and  $q$ , see [19, 17]. In particular, it is known that regular gbent functions exist for  $n$  and  $q$  both even, and for  $n$  odd and  $q = 2^k$ ,  $k \geq 3$  [23]. The nonexistence for the case of  $q = 4$  and odd  $n$  follows from [29] (Lemma 3.3) while the Boolean case is a trivial one.

The existence of (anti-)self-dual gbent functions, essentially forming a remarkable subclass of regular gbent functions, is clear for the case when  $n, q$

are both even [16]. Some examples of such functions are provided by generalized Maiorana–McFarland bent functions and the generalization of Dillon’s class  $\mathcal{PS}_{ap}$ . It also follows that (anti-)self-dual gbent functions do not exist for  $q = 4$  and odd  $n$  [30]. Other cases comprising odd  $n$  remained open.

In current work we consider the problem of the existence of self-dual gbent functions in  $n$  variables for the case of an odd  $n$  and even  $q$  under some limitations. Firstly, it is easy to show that self-dual gbent functions in one variable do not exist, hence the considered problem makes sense only for  $n \geq 3$ . Our result is the following

**Theorem 7.** *Let  $q$  be an even positive integer such that  $q \equiv 0 \pmod{8}$ , then for every odd  $n \geq 3$  there exist self-dual and anti-self-dual gbent functions in  $n$  variables.*

*Proof.* Denote  $z = e^{3\pi i/4}$  and consider the vector  $Z = (z, 1, 1, -z)$ . Since  $q \equiv 0 \pmod{8}$ , we have  $\pm z \in \{1, \omega, \omega^2, \omega^3, \dots, \omega^{q-1}\}$ . Define vector

$$Y = Z + \frac{1}{\sqrt{2}}H_2Z.$$

In coordinate form we have

$$\begin{aligned} Y &= \begin{pmatrix} z \\ 1 \\ 1 \\ -z \end{pmatrix} + \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} z \\ 1 \\ 1 \\ -z \end{pmatrix} = \begin{pmatrix} z \\ 1 \\ 1 \\ -z \end{pmatrix} + \frac{2}{\sqrt{2}} \begin{pmatrix} 1 \\ z \\ z \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} z + \sqrt{2} \\ 1 + \sqrt{2}z \\ 1 + \sqrt{2}z \\ -z - \sqrt{2} \end{pmatrix} = \begin{pmatrix} e^{\pi i/4} \\ i \\ i \\ -e^{\pi i/4} \end{pmatrix}, \end{aligned}$$

and it holds  $\pm e^{\pi i/4}, i \in \{1, \omega, \omega^2, \omega^3, \dots, \omega^{q-1}\}$ . Based on the decomposition (1) of sign function of self-dual bent function it is possible to deduce the analog of this decomposition for self-dual gbent functions at least for the case of even  $q$ . That is the vector  $(Y, Z)$  with  $Y, Z \in \{1, \omega, \omega^2, \omega^3, \dots, \omega^{q-1}\}^{2^{n-1}}$  is the polyphase sequence of a self-dual gbent function in  $n$  variables if and only if

$$Y = Z + \frac{2H_{n-1}}{2^{n/2}}Z. \tag{8}$$

So, by the relation (8) the vector

$$F = (Y, Z) = \left( e^{\pi i/4}, i, i, e^{5\pi i/4}, e^{3\pi i/4}, 1, 1, e^{7\pi i/4} \right),$$

is the polyphase sequence of a self-dual gbent function in 3 variables having vector of values

$$(q/8, q/4, q/4, 5q/8, 3q/8, 0, 0, 7q/8).$$

Then by the construction **C1** with  $f$  being the mentioned above self-dual gbent function in 3 variables, we obtain a polyphase sequence

$$(F, F, F, -F),$$

which is a polyphase sequence of a self-dual gbent function in 5 variables.

This iterative process with a two-variables step can be continued further so we obtain the existence of self-dual gbent functions in  $n$  variables for every odd  $n \geq 3$  provided  $q$  is divisible by 8.  $\square$

In particular, from Theorem 7 it follows that self-dual gbent functions in both even and odd number of variables exist for  $q = 2^k$  with integer  $k \geq 3$ .

## 7 Conclusion

In this work we studied subfunctions of self-dual bent functions in  $n - 1$  and  $n - 2$  variables and introduced the notation of the Gram matrix of a bent function. It is interesting to continue the study of this matrix and obtain new metrical relations between subfunctions of an arbitrary (self-dual) bent function. The search of the constructions of bent functions with particular Gram matrix is also a goal worth pursuing. Regarding generalized bent functions it is interesting to know if there self-dual gbent functions in odd number of variables outside the considered case, in particular, for an odd  $q$ .

Considering the Gram matrices for the constructions **C1** and **C3** we can conclude that the problem of the classification of Gram matrices of self-dual bent functions (even in the case when all subfunctions are bent) comprises the problem of finding all values of the Rayleigh quotient that can be achieved by some of their subfunctions. The last problem has intersection with the investigation of the Hamming distances spectrum between bent functions and their duals.

### Acknowledgments.

The work is supported by Mathematical Center in Akademgorodok under agreement No. 075-15-2022-281 with the Ministry of Science and Higher Education of the Russian Federation.

## References

- [1] Adams C.M., Tavares S.E., “Generating bent sequences”, *Discrete Appl. Math.*, **39** (1992), 155–159.
- [2] Canteaut A., Carlet C., Charpin P., Fontaine C., “On cryptographic properties of the cosets of  $R(1, m)$ ”, *IEEE Trans. Inform. Theory*, **47**:4 (2001), 1494–1513.
- [3] Canteaut A., Charpin P., “Decomposing bent functions”, *IEEE Trans. Inform. Theory*, **49**:8 (2003), 2004–2019.
- [4] Carlet C., Mesnager S., “Four decades of research on bent functions”, *Des. Codes Cryptogr.*, **78**:1 (2016), 5–50.
- [5] Carlet C., Danielsen L.E., Parker M.G., Solé P., “Self-dual bent functions”, *Int. J. Inform. Coding Theory*, **1** (2010), 384–399.
- [6] Carlet C., *Bent Functions, Results and Applications to Cryptography*, Cambridge University Press, 2020, 620 p.
- [7] Danielsen L.E., Parker M.G., Solé P., “The Rayleigh quotient of bent functions”, *Springer Lect. Notes in Comp. Sci.*, **5921** (2009), 418–432.
- [8] Feulner T., Sok L., Solé P., Wassermann A., “Towards the classification of self-dual bent functions in eight variables”, *Des. Codes Cryptogr.*, **68**:1 (2013), 395–406.
- [9] Hou X.-D., “Classification of self dual quadratic bent functions”, *Des. Codes Cryptogr.*, **63**:2 (2012), 183–198.
- [10] Hyun J.Y., Lee H., Lee Y., “MacWilliams duality and Gleason-type theorem on self-dual bent functions”, *Des. Codes Cryptogr.*, **63**:3 (2012), 295–304.
- [11] Kutsenko A.V., “The Hamming Distance Spectrum Between Self-Dual Maiorana-McFarland Bent Functions”, *Journal of Applied and Industrial Mathematics*, **12**:1 (2018), 112–125.
- [12] Kutsenko A., “Metrical properties of self-dual bent functions”, *Des. Codes Cryptogr.*, **88**:1 (2020), 201–222.
- [13] Kutsenko A., “The group of automorphisms of the set of self-dual bent functions”, *Cryptogr. Commun.*, **12**:5 (2020), 881–898.
- [14] Kutsenko A., Tokareva N., “Metrical properties of the set of bent functions in view of duality”, *Prikl. Diskr. Mat. (Applied Discrete Math.)*, **49** (2020), 18–34.
- [15] Kutsenko A., Gorodilova A., “The Duality Mapping and Unitary Operators Acting on the Set of All Generalized Boolean Functions”, Proceedings of the 10th Workshop on Current Trends in Cryptology (CTCrypt 2021), 2021.
- [16] Kutsenko A.V., “On constructions and properties of self-dual generalized bent functions”, arXiv: <https://arxiv.org/abs/2107.13538>.
- [17] Leung K.H., Wang Q., “New nonexistence results on  $(m, n)$ -generalized bent functions”, *Des. Codes Cryptogr.*, **88**:4 (2020), 755–770.
- [18] Li Y., Kan H., Mesnager S., Peng J., Tan C.H., Zheng L., “Generic constructions of (Boolean and vectorial) bent functions and their consequences”, *IEEE Trans. Inform. Theory*, **68**:4 (2022), 2735–2751.
- [19] Liu H., Feng K., Feng R., “Nonexistence of generalized bent functions from  $\mathbb{Z}_2^n$  to  $\mathbb{Z}_m$ ”, *Des. Codes Cryptogr.*, **82**:3 (2017), 647–662.
- [20] Logachev O.A., Sal’nikov A.A., Yashchenko V.V., “Bent functions on a finite abelian group”, *Discrete Math. Appl.*, **7**:6 (1997), 547–564.
- [21] Logachev O.A., Sal’nikov A.A., Smyshlyaev S.V., Yashchenko V.V., *Boolean functions in coding theory and cryptology*, 2020, 576.
- [22] Luo G., Cao X., Mesnager S., “Several new classes of self-dual bent functions derived from involutions”, *Cryptogr. Commun.*, **11**:6 (2019).
- [23] Martinsen T., Meidl W., Stănică P., “Generalized bent functions and their Gray images”, *Arithmetic of Finite Fields*, **10064** (2017), 160–173.
- [24] Mesnager S., “On semi-bent functions and related plateaued functions over the Galois field  $\mathbb{F}_{2^n}$ ”, *Open Problems in Mathematics and Computational Science*, 2014, 243–273.

- [25] Mesnager S., “Several New Infinite Families of Bent Functions and Their Duals”, *IEEE Trans. Inf. Theory*, **60**:7 (2014), 4397–4407.
- [26] Mesnager S., *Bent Functions: Fundamentals and Results*, Springer, Berlin, 2016, 544 p.
- [27] Preneel B., Van Leekwijck W., Van Linden L., Govaerts R., Vandewalle J., “Propagation characteristics of Boolean functions”, *Advances in Cryptology-EUROCRYPT, Lecture Notes in Computer Science*, **473**, Springer, Berlin, Heidelberg, 1990, 161–173.
- [28] Rothaus O.S., “On bent functions”, *J. Combin. Theory. Ser. A*, **20**:3 (1976), 300–305.
- [29] Schmidt K.-U., “Quaternary constant-amplitude codes for multicode CDMA”, *IEEE Trans. Inform. Theory*, **55**:4 (2009), 1824–1832.
- [30] Sok L., Shi M., Solé. P., “Classification and Construction of quaternary self-dual bent functions”, *Cryptogr. Commun.*, **10**:2 (2018), 277–289.
- [31] Tokareva N., “On the number of bent functions from iterative constructions: lower bounds”, *Adv. Math. Commun.*, **5**:4 (2011), 609–621.
- [32] Tokareva N., *Bent Functions, Results and Applications to Cryptography*, Acad. Press. Elsevier, 2015, 230 p.
- [33] Wada T., “Characteristic bit sequences applicable to constant amplitude orthogonal multi-code systems”, *IEICE Trans. Fundamentals*, **E83-A**:11 (2000), 2160–2164.
- [34] Wolfmann J., “Special bent and near-bent functions”, *Adv. Math. Commun.*, **8**:1 (2014), 21–33.
- [35] Yarlagadda R., Hershey J.E., “A note on the eigenvectors of Hadamard matrices of order  $2^n$ ”, *Linear Algebra & Appl.*, **45** (1982), 43–53.



# Proper families of functions and their applications

Alexei Galatenko, Valentin Nosov, Anton Pankratiev,  
and Kirill Tsaregorodtsev

Lomonosov Moscow State University, Moscow, Russia  
agalat@msu.ru, vnosov40@mail.ru, apankrat@intsys.msu.ru, kirill194\_12@mail.ru

## Abstract

Proper families of functions provide a convenient and memory-efficient method for specification of large parametric families of quasigroups and  $d$ -quasigroups of a large order. We present a number of examples of proper families, outline the connections between proper families and quasigroups and  $d$ -quasigroups, discuss equivalent definitions of properness and various operations on proper families, analyze the complexity of deciding properness and study possible configurations of essential dependence inside proper families.

**Keywords:** proper family of functions, quasigroup, Latin square.

## 1 Introduction

Quasigroups and their analogues of higher dimensions are attracting attention as a platform for the construction of cryptographic algorithms (see e.g. the review [1] and an example of an algorithm [2]). In case of quasigroups of a high order tabular specification is inefficient due to high memory requirements. The situation is even worse for higher-dimensional structures. A possible way around is to switch from tabular specification to formula-based specification. In particular large parametric families of quasigroups and  $d$ -quasigroups of a high order can be specified by proper families of functions. In our paper we present a number of examples of proper families, outline the connections between proper families and quasigroups and  $d$ -quasigroups, discuss equivalent definitions of properness and various operations on proper families, analyze the complexity of deciding properness and study possible configurations of essential dependence inside proper families.

## 2 Definitions and examples

### 2.1 Definition of proper family

The notion of proper family of Boolean functions was introduced in [3] and investigated further in [4, 5, 6]. A generalization to Abelian groups is given in [7]. Here we give a more general definition for the case of quasigroups [8].

**Definition 1.** *Let  $Q$  be a finite nonempty set. Suppose that we have a collection of functions  $F = (f_1(x_1, \dots, x_n), \dots, f_n(x_1, \dots, x_n))$ , where  $f_i: Q^n \rightarrow Q$ . Then  $F$  is called a family of functions on  $Q^n$ .*

**Definition 2.** *A family  $F$  of functions on  $Q^n$  is said to be proper if for any two unequal elements  $\alpha, \beta \in Q^n$  it holds that  $\exists i: \alpha_i \neq \beta_i, f_i(\alpha) = f_i(\beta)$ .*

If  $|Q| = k$ , then without loss of generality assume that  $Q = \{0, 1, \dots, k-1\}$  (this set is denoted by  $E_k$ ) and  $f_i$  are functions of  $k$ -valued logic.

### 2.2 Examples of proper families

**Example 1** ([7]). *Triangular family of size  $n$  is a family of functions  $F = (f_1(x_1, \dots, x_n), \dots, f_n(x_1, \dots, x_n))$  with the property that there exists an ordering  $\sigma \in S_n$  (a consistent renumbering of functions and variables) such that after reordering the family can be written as:*

$$\left[ \begin{array}{c} f_{\sigma(1)}(\cdot) \\ f_{\sigma(2)}(x_{\sigma(1)}) \\ \vdots \\ f_{\sigma(n)}(x_{\sigma(1)}, \dots, x_{\sigma(n-1)}) \end{array} \right],$$

*i.e. after reordering each function can depend only on «previous» variables.*

It can be easily seen that triangular families are proper: if two distinct elements  $\alpha = (\alpha_1, \dots, \alpha_n)$  and  $\beta = (\beta_1, \dots, \beta_n)$  are such that  $\alpha_{\sigma(1)} = \beta_{\sigma(1)}, \dots, \alpha_{\sigma(k)} = \beta_{\sigma(k)}, \alpha_{\sigma(k+1)} \neq \beta_{\sigma(k+1)}$ , then the desired index is  $\sigma(k+1)$ , since  $f_{\sigma(k+1)}(\alpha) = f_{\sigma(k+1)}(\beta)$ .

A very special case of a triangular family is an assembly of constant functions (i. e.,  $f_i \equiv \text{const}_i$ ).

It can be shown that in the Boolean case triangular families comprise a small fraction of all proper families of a given size  $n$ :

**Theorem 1** ([9]). *Let  $\Delta(n)$  be the number of triangular Boolean families of size  $n$ ,  $T(n)$  be the number of proper Boolean families of size  $n$ . Then it holds that  $\frac{\Delta(n)}{T(n)} = o\left(\frac{1}{n^{D \cdot 2^n}}\right)$  as  $n \rightarrow \infty$ , for some  $D > 0$ .*

Generalization of Theorem 1 to the case of arbitrary values of  $|Q|$  is the subject of future research.

**Example 2** ([5]). *Triangular extension.* Let  $F^0 = (g_{1,0}, \dots, g_{n,0})$  be a family of functions depending on  $(x_{1,0}, \dots, x_{n,0})$ . Let  $s_1, \dots, s_n \in \mathbb{N}$  be a collection of arbitrary natural numbers. Define functions  $f_{i,j}$  as follows:

$$\begin{aligned} f_{i,1} &= F_{i,1}(g_{i,0}), \\ f_{i,2} &= F_{i,2}(g_{i,0}, x_{i,1}), \\ &\vdots \\ f_{i,s_i} &= F_{i,s_i}(g_{i,0}, x_{i,1}, \dots, x_{i,s_i-1}), \\ f_{i,0} &= F_{i,0}(g_{i,0}, x_{i,1}, \dots, x_{i,s_i}). \end{aligned}$$

If the family  $F^0$  is proper, then the family  $F = (f_{i,j})_{i=1,\dots,n,j=0,\dots,s_i}$  is also proper for any functions  $F_{i,j}$ .

**Definition 3.** Two functions  $f, g: E_k^n \rightarrow E_k$  are orthogonal, if for any  $x \in E_k^n$  it holds that either  $f(x) = 0$  or  $g(x) = 0$ .

**Example 3** ([5, 10]). *Family of orthogonal functions.* Let  $F = (f_1, \dots, f_n)$  be a family of pairwise orthogonal functions such that  $f_i$  does not depend essentially on  $x_i$ . Then  $F$  is proper. For instance the family

$$\begin{aligned} f_1 &= \bar{x}_2 x_3 \cdots x_{n-1} x_n, \\ f_2 &= \bar{x}_3 x_4 \cdots x_n x_1, \\ &\vdots \\ f_n &= \bar{x}_1 x_2 \cdots x_{n-2} x_{n-1} \end{aligned} \tag{1}$$

consists of pairwise orthogonal Boolean functions, and each  $f_i$  does not depend essentially on  $x_i$ . Hence, the family  $F$  is proper.

**Remark 1.** The requirement in Example 3 can be generalized. Namely, the functions  $f_i$  should not depend essentially on  $x_i$  and have the following property: there exists  $q \in Q$  such that for any  $i \neq j$  and any  $x \in Q^n$  at least one of the values  $f_i(x), f_j(x)$  equals  $q$ . In particular, for  $Q = E_k$  and  $q = 0$  the second condition means that the vectors of values of functions comprising orthogonal proper families are orthogonal.

### 2.3 Special classes of proper families

In this subsection we give some examples of proper families that do not fall into the categories defined above.

**Example 4** ([5, Theorem 5]). A family  $F$  of functions over a prime field  $\mathbb{F}_p$  defined by the rule

$$\begin{bmatrix} f_1(x_1, \dots, x_n) \\ \vdots \\ f_n(x_1, \dots, x_n) \end{bmatrix} = \begin{bmatrix} \phi(x_2 + 1) \cdot \dots \cdot \phi(x_2 + p - 1) \cdot \phi(x_3) \\ \vdots \\ \phi(x_1 + 1) \cdot \dots \cdot \phi(x_1 + p - 1) \cdot \phi(x_2) \end{bmatrix},$$

where  $\phi$  is a permutation polynomial, is proper iff  $n$  is odd.

**Remark 2.** A special case of the construction from Example 4 for Boolean functions is the following example from [13]:

$$\begin{bmatrix} f_1(x_1, \dots, x_n) \\ \vdots \\ f_n(x_1, \dots, x_n) \end{bmatrix} = \begin{bmatrix} \overline{x_2} \cdot x_3 \\ \vdots \\ \overline{x_1} \cdot x_2 \end{bmatrix}. \tag{2}$$

**Example 5** ([9]). The following Boolean family is proper for any  $n \geq 1$  :

$$\begin{bmatrix} 0 \\ x_1 \\ x_1 \oplus x_2 \\ \vdots \\ x_1 \oplus x_2 \oplus \dots \oplus x_{n-1} \end{bmatrix} \oplus \begin{bmatrix} \bigoplus_{i < j, i, j \neq 1}^n x_i x_j \\ \bigoplus_{i < j, i, j \neq 2}^n x_i x_j \\ \bigoplus_{i < j, i, j \neq 3}^n x_i x_j \\ \vdots \\ \bigoplus_{i < j, i, j \neq n}^n x_i x_j \end{bmatrix}; \tag{3}$$

The second sum in each  $f_i$  contains all pairwise products except  $x_i \cdot x_j$ ,  $i < j$ .

### 3 Generation of quasigroups and $n$ -quasigroups

Recall that a finite quasigroup is a pair  $(Q, g)$ , where  $Q$  is a nonempty finite set, and  $g: Q \times Q \rightarrow Q$  is invertible in both variables, i.e. for any  $a, b \in Q$  the equations  $g(x, a) = b$  and  $g(a, y) = b$  are uniquely solvable. Throughout this paper all objects are finite, thus for the sake of brevity the word “finite” will be omitted.

Obviously Cayley tables of quasigroups are Latin squares, i.e. elements in any row and any column are distinct, and vice versa, any Latin square is the Cayley table of a finite quasigroup.

The notion of a quasigroup can be naturally generalized to the case of operations of greater arity. Assume that  $d \in \mathbb{N}$ ,  $d \geq 2$ ,  $Q$  is a nonempty finite set. A pair  $(Q, g)$ , where  $g: Q^d \rightarrow Q$  is invertible in any variable, is a  $d$ -quasigroup. Note that if  $d = 2$ , then we obtain the definition of a “regular” quasigroup. Similar to the case  $d = 2$  Cayley tables of  $d$ -quasigroups are  $d$ -dimensional Latin cubes.

Assume that  $d, k, n \in \mathbb{N}$ ,  $d, k \geq 2$ ,  $Q = E_k^n$ ,  $(Q, g)$  is a  $d$ -quasigroup. Then elements of the set  $Q$  can be naturally encoded by  $n$ -tuples, and  $g$  can be considered as a vector function  $(g_1, \dots, g_n)$ , where  $g_i: (E_k^n)^d \rightarrow E_k$  are functions of  $k$ -valued logic. In other words the  $d$ -quasigroup operation  $g$  is represented in the form

$$\begin{aligned} z_1 &= g_1(x_1^1, \dots, x_n^1, \dots, x_1^d, \dots, x_n^d) \\ &\vdots \\ z_n &= g_n(x_1^1, \dots, x_n^1, \dots, x_1^d, \dots, x_n^d) \end{aligned} \quad (4)$$

where  $z_i, x_j^i$  take values in  $E_k$ . Further assume that  $(E_k, h_1), \dots, (E_k, h_n)$  are  $(d+1)$ -quasigroups and consider the following special case of the representation (4):

$$\begin{aligned} z_1 &= h_1(x_1^1, \dots, x_n^d, f_1(\pi_1(x_1^1, \dots, x_1^d), \dots, \pi_n(x_n^1, \dots, x_n^d))) \\ &\vdots \\ z_n &= h_n(x_n^1, \dots, x_n^d, f_n(\pi_1(x_1^1, \dots, x_1^d), \dots, \pi_n(x_n^1, \dots, x_n^d))) \end{aligned} \quad (5)$$

where  $f_i$  are  $n$ -ary functions of  $k$ -valued logic,  $\pi_j$  are  $d$ -ary functions of  $k$ -valued logic. It turned out that the representation (5) is tightly connected with the notion of a proper family. Namely the following assertion holds.

**Theorem 2.** *Representation (5) specifies a  $d$ -quasigroup operation for any choice of the functions  $\pi_1, \dots, \pi_n$  iff the family  $(f_1, \dots, f_n)$  is proper.*

Theorem 2 was originally proved for the case  $k = 2$ ,  $d = 2$ ,  $h_i(x, y, z) = x \oplus y \oplus z$  in [4], then the result was extended to the case of an arbitrary  $k \geq 2$  and  $h_i = x + y + z$  (here  $+$  is the addition operation in some Abelian group  $(E_k, +)$ ) in [7], and the final form for the case  $d = 2$  was established in [8]. Transition from Abelian groups to quasigroups was proved to be essential: in the group case all quasigroups generated by proper families contain a fixed point, i.e. an element  $a$  such that  $g(a, a) = a$ .

The  $d$ -quasigroup form of Theorem 2 was originally proposed for  $h_i(x_1, \dots, x_{d+1}) = x_1 + \dots + x_{d+1}$  in [14] by I. A. Plaksina and further generalized in [15]. Note that the condition on  $h_i$  can not be relaxed, since the fact that  $h_i$  is not invertible in some variable directly implies that for any family  $(f_1, \dots, f_n)$  there exists a fixation of the functions  $\pi_1, \dots, \pi_n$  such that the operation specified by the relations (5) is not a  $d$ -quasigroup one.

Reducing the representation (4) to the form (5) leads to the loss of generality: if the operations  $h_1, \dots, h_n$  are fixed, then not all  $d$ -quasigroups operations can be represented in the form (5). For example, in [16] it was shown that in the case  $k = n = 2$  only 60 out of 576 quasigroups of the order 4

can be specified by proper families. On the other hand if the value of  $n$  is essentially large, then the representation (5) provides a memory-efficient way of specifying large families of  $d$ -quasigroups. Indeed, the Cayley table of a  $d$ -quasigroup of the order  $k^n$  consists of  $(k^n)^d$  elements, whereas tabular specification of a proper family consists of  $d \cdot k^n$  elements, the tables for the functions  $h_i$  require at most  $n \cdot k^{d+1}$  elements, and the functions  $\pi_j$  can be defined by  $n$  tables consisting of  $k^d$  elements. Variation of the internal functions  $\pi_j$  in the representation (5) generates  $(k^{k^d})^n$   $d$ -quasigroups, some of which may coincide (e.g. if all functions comprising a proper family are constants, then all  $d$ -quasigroups generated are the same). The number of distinct  $d$ -quasigroups generated satisfies the following assertion.

**Theorem 3.** *The number of distinct  $d$ -quasigroups specified by a proper family of order  $n$  in  $k$ -valued logic is at most  $(k^{k^d})^{n-1}$ ; this bound is sharp.*

The upper bound is a direct corollary of the definition of a proper family, since the range of a proper family does not contain tuples that differ in all positions. The lower bound is achieved e.g. on the triangular family  $f_1 = \text{const}$ ,  $f_2 = x_1, \dots, f_n = x_{n-1}$ .

Let  $(Q, g_1)$  and  $(Q, g_2)$  be  $d$ -quasigroups,  $\alpha_1, \dots, \alpha_d, \beta$  be permutations on  $Q$  such that the identity  $g_1(x_1, \dots, x_d) = \beta^{-1}(g_2(\alpha_1(x_1), \dots, \alpha_d(x_d)))$  holds. Then the  $d$ -quasigroups  $(Q, g_1)$  and  $(Q, g_2)$  are said to be isotopic, and the transformation that maps  $(Q, g_1)$  to  $(Q, g_2)$  is referred to as an isotopy. It can be easily noticed that permuting indices of the variables in the representation (5) is an isotopy, so the following assertion holds.

**Theorem 4.** *Assume that  $d, k, n \in \mathbb{N}$ ,  $k, d \geq 2$ ,  $(E_k, h_1), \dots, (E_k, h_n)$  are  $(d+1)$ -quasigroups,  $\sigma_1, \dots, \sigma_d, \tilde{\sigma} \in S_n$  are some permutations,  $g: (E_k^n)^d \rightarrow E_k^n$ ,  $g = (g_1, \dots, g_n)$ , is specified by the relations*

$$g_{\tilde{\sigma}(i)}(x_1^1, \dots, x_n^1, \dots, x_1^d, \dots, x_n^d) = h_i(x_{\sigma_1(i)}^1, \dots, x_{\sigma_d(i)}^d, f_i(\pi_1(x_{\sigma_1(1)}^1, \dots, x_{\sigma_d(1)}^d), \dots, \pi_n(x_{\sigma_1(n)}^1, \dots, x_{\sigma_d(n)}^d))), \quad (6)$$

where  $f_i$  are  $n$ -ary functions of  $k$ -valued logic,  $\pi_j$  are  $d$ -ary functions of  $k$ -valued logic. Then  $(E_k^n, g)$  is a  $d$ -quasigroup for any choice of the functions  $\pi_1, \dots, \pi_n$  if and only if the family  $(f_1, \dots, f_n)$  is proper.

The construction was originally proposed for the case  $k = 2$  in [16] and then generalized in [15]. Besides increasing the cardinality of the sets of  $d$ -quasigroups generated, permuting variable indices improves such  $d$ -quasigroup properties as simplicity, non-affinity and polynomial completeness; e.g. in [16] it was shown that all 60 quasigroups of the order 4 specified

by proper families of Boolean functions are non-simple, whereas applying permutation construction yields 240 distinct quasigroups including 112 simple and non-affine (and thus polynomially complete) ones. Note that  $d + 1$  permutations can be stored using just  $(d + 1) \cdot n \cdot \lceil \log_2 n \rceil$  bits, so additional memory load is negligible.

Another generalization consists in using different internal functions in the representation (5). In general it may lead to non-quasigroup operations, but for the case of triangular families the following assertion holds.

**Theorem 5.** *Suppose that  $(f_1, \dots, f_n)$  is a triangular proper family,  $(E_k, h_1), \dots, (E_k, H_n)$  are  $(d + 1)$ -quasigroups. Then the representation*

$$z_i = h_i(x_i^1, \dots, x_i^d, f_i(\pi_{i,1}(x_1^1, \dots, x_1^d), \dots, \pi_{i,n}(x_n^1, \dots, x_n^d))) \quad (7)$$

*specifies a  $d$ -quasigroup operation for any choice of the functions  $\pi_{i,j}$ .*

The proof of this theorem is similar to the poof of Theorem 2 in [13].

Further investigations of the connection between proper families and  $d$ -quasigroups include generation of  $d$ -quasigroups with additional properties (degrees of polynomials in algebraic normal form of the functions  $g_i$  in representation (4), e.g. to produce so-called multivariate quadratic quasigroups [17]; simplicity, non-affinity and polynomial completeness; various forms of non-linearity, etc.), thorough study of the construction with respect to variation of the functions  $f_i$  and/or  $h_j$ , and efficiency of implementing the construction in hardware and/or software. Some results connected to generation of multivariate quadratic quasigroups are discussed in Section 8; other problems are the subject of future research.

## 4 Equivalent definitions of properness

There are a number of equivalent definitions of proper families.

In paper [7] the regularity criteria was proved.

**Theorem 6** ([7, Theorem 2]). *A family  $F$  of functions over  $Q = G^n$ , where  $G$  is an Abelian group, is proper iff for any collection of arbitrary mappings  $\psi_i: G \rightarrow G$  the following mapping is a bijection  $G^n \rightarrow G^n$ :*

$$\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \rightarrow \begin{bmatrix} x_1 + \psi_1(f_1(x_1, \dots, x_n)) \\ \vdots \\ x_n + \psi_n(f_n(x_1, \dots, x_n)) \end{bmatrix}, \quad x_i \in G.$$

In the special case of  $p$ -valued logic, i.e. when  $G = \mathbb{F}_p$ ,  $p$  is prime, Theorem 6 can be strengthened in the following way.

**Theorem 7** ([5, Theorem 3]). *A family  $F$  of functions over  $Q = \mathbb{F}_p^n$ , where  $\mathbb{F}_p$  is a field,  $p$  is prime, is proper iff for any elements  $a_i \in \mathbb{F}_p$  the mapping*

$$\begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \rightarrow \begin{bmatrix} x_1 + a_1 \cdot f_1(x_1, \dots, x_n) \\ \vdots \\ x_n + a_n \cdot f_n(x_1, \dots, x_n) \end{bmatrix}, \quad x_i \in \mathbb{F}_p$$

*is a bijection  $\mathbb{F}_p^n \rightarrow \mathbb{F}_p^n$ .*

A number of equivalent definitions can be obtained for the Boolean case. The first one uses generalization of the concept of an essential variable.

**Definition 4.** *Let  $I = \{i_1, \dots, i_s\} \subseteq \{1, \dots, n\}$  be a subset of indices. A collection of variables  $x_I = (x_{i_1}, \dots, x_{i_s})$  is called essential for Boolean function  $f(x_1, \dots, x_n)$ , if*

$$\sum_{\alpha_{i_1}, \dots, \alpha_{i_s}} f(x_1, \dots, x_{i_1-1}, \alpha_{i_1}, x_{i_1+1}, \dots, x_{i_s-1}, \alpha_{i_s}, x_{i_s+1}, \dots, x_n) \not\equiv 0 \pmod{2}.$$

**Theorem 8** ([3]). *A family of Boolean functions  $F = (f_1, \dots, f_n)$  is proper iff for any  $I \subseteq \{1, \dots, n\}$  the collection  $x_I$  **is not** essential for the function  $f = \prod_{i \in I} f_i$ .*

**Remark 3.** *From Theorem 8 (as well as from the definition of properness) it follows that  $f_i$  can not depend essentially on  $x_i$ .*

In the Boolean case there also exists another (more geometric) characterization. Let us introduce some auxiliary definitions.

**Definition 5.** *The graph of the Boolean cube  $\mathbb{E}_n$  of dimension  $n$  is a graph with  $2^n$  vertices labelled by  $(\alpha_1, \dots, \alpha_n)$ ,  $\alpha_i \in \{0, 1\}$ , two vertices  $\alpha, \beta$  of which are adjacent iff the Hamming distance between  $\alpha$  and  $\beta$  is 1.*

**Definition 6.** *The subcube of  $\mathbb{E}_n$  of dimension  $n - m$  is a subgraph of  $\mathbb{E}_n$  induced by the vertices with fixed coordinates  $i_1, \dots, i_m \in \{1, \dots, n\}$ .*

**Definition 7** ([18]). *Unique sink orientation (USO) of  $\mathbb{E}_n$  is an orientation of the edges of  $\mathbb{E}_n$  such that in every subcube of  $\mathbb{E}_n$  there is exactly one vertex for which all adjoining edges are oriented inward (i.e. towards that vertex).*

**Definition 8.** *For a given Boolean family  $F$  of size  $n$  we introduce the following directed graph (the family graph  $\Gamma_F$ ). Vertices of  $\Gamma_F$  are all binary  $n$ -tuples:  $V = \{(\alpha_1, \dots, \alpha_n) \mid \alpha_i \in \{0, 1\}\}$ . Suppose  $\alpha, \beta \in V$  such that the Hamming distance between them is 1:  $\alpha_i \neq \beta_i$ . If  $f_i(\alpha) = \alpha_i$ , then we have an oriented edge  $(\beta, \alpha) \in E$ .*



Now we are ready to give a following geometrical characterization of Boolean proper families.

**Theorem 9** ([19]). *Graph  $\Gamma_F$  of a family  $F$  is USO iff  $F$  is proper.*

The criterion above can be viewed more algebraically using the notion of fixed point.

**Definition 9.** *A family  $F$  on  $Q^n$  is said to have a fixed point  $\alpha \in Q^n$  if  $F(\alpha) = \alpha$ , i.e., if  $f_i(\alpha_1, \dots, \alpha_n) = \alpha_i$  for any  $i = 1, \dots, n$ .*

As it was shown in [20], the geometrical characterization from Theorem 9 can be reformulated in the following way:

**Corollary 1** ([20]). *Let  $F$  be a family of Boolean functions. Then  $F$  is proper if and only if for  $F$  and any of its projections (in the sense of theorem 13, see below) there exists a unique fixed point.*

In [20] Corollary 1 was generalized to the case of  $k$ -valued logic.

**Definition 10** ([20]). *Let  $F$  be a family of functions on  $E_k^n$ ,  $\sigma_i, \tau_i \in S_{E_k}$  be permutations on  $E_k$ . Then the family  $\hat{F} = (\hat{f}_1, \dots, \hat{f}_n)$  defined as  $\hat{f}_i(x_1, \dots, x_n) = \tau_i(f_i(\sigma_1(x_1), \dots, \sigma_n(x_n)))$  is a reencoding of the family  $F$ .*

**Theorem 10** ([20]). *A family  $F$  on  $E_k^n$  is proper iff every reencoding of  $F$  and every reencoding of any projection of  $F$  has a unique fixed point.*

## 5 Operations on proper families

Let us consider the following situation. Given a proper family  $F$  of size  $n$ , how one can generate new proper families out of the old one?

**Theorem 11** ([7, Remark 3]). *Let  $\sigma \in S_n$  be a permutation. Define  $\sigma(F)$  as the family obtained from  $F$  by a simultaneous permutation of the indices of variables and functions:  $f_i(x_1, \dots, x_n) \rightarrow f_{\sigma(i)}(x_{\sigma(1)}, \dots, x_{\sigma(n)})$ . If  $F$  is a proper family, then  $\sigma(F)$  is also proper.*

**Theorem 12** ([7, Remark 1]). *For any  $n$ -tuple  $A = (a_1, \dots, a_n) \in G^n$  let us consider the following family  $\hat{F}$ :*

$$\hat{F} = F + A = \begin{bmatrix} f_1(x_1, \dots, x_n) + a_1 \\ \vdots \\ f_n(x_1, \dots, x_n) + a_n \end{bmatrix}.$$

*A family  $F$  on  $G^n$ , where  $G$  is a group, is proper if and only if the family  $\hat{F} = F + A$  is proper.*

**Theorem 13** ([8, Lemma 1]). *For any  $i \in \{1, \dots, n\}$  and any constant  $a \in Q$  the family  $F'$  obtained from  $F$  by substituting the value  $a$  for the variable  $x_i$  and cancelling the function  $f_i$  is a proper family of the order  $(n - 1)$ :*

$$F'(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = \begin{bmatrix} f_1(x_1, \dots, x_{i-1}, a, x_{i+1}, \dots, x_n) \\ \vdots \\ f_{i-1}(x_1, \dots, x_{i-1}, a, x_{i+1}, \dots, x_n) \\ f_{i+1}(x_1, \dots, x_{i-1}, a, x_{i+1}, \dots, x_n) \\ \vdots \\ f_n(x_1, \dots, x_{i-1}, a, x_{i+1}, \dots, x_n) \end{bmatrix}.$$

As we see, shifts, «consistent» permutations and «projections» preserves properness of a family.

As an inverse to the operation of projection, we can expand a collection of families of size  $n$  to the family of size  $n + 1$  using the following construction.

**Example 6** ([21, Lemma 2]). *Suppose that  $F_m: E_k^n \rightarrow E_k^n$ ,  $m \in E_k$  is a collection of proper families in  $k$ -valued logic. Denote by  $I_r(x)$  a function of the following form:  $I_m(x) = k - 1$  if  $x = m$ , otherwise  $I_m(x) = 0$ . Let*

$$F(x_1, \dots, x_n, x_{n+1}) = \bigvee_{m \in E_k} \left( I_m(x_{n+1}) \bigwedge F_m(x_1, \dots, x_n) \right), \quad (8)$$

where  $\bigvee$  is the maximum,  $\bigwedge$  is the minimum. Define  $f_{n+1}(x_1, \dots, x_{n+1}) \equiv \text{const} \in E_k$ . Then the family  $(F, f_{n+1}): E_k^{n+1} \rightarrow E_k^{n+1}$  is proper.

Using the construction from Example 6 one can create new families out of old ones. However, we note that  $f_{n+1}$  is constant in the construction, hence there are proper families of size  $n + 1$  that cannot be generated in this way.

The following approach generalizing the Example 6 can be used to construct any family of size  $n + 1$  out of families of size  $n$ .

**Theorem 14** ([21, Theorem 3]). *Suppose that  $k$  proper families  $F_0, \dots, F_{k-1}$  on  $E_k^n$  of size  $n$  are given.*

1. *Let us construct an undirected graph  $\mathcal{G} = (V, E)$ , where  $V = E_k^n$ , and two vertices  $u, v \in V$  are adjacent iff the following condition holds:*

$$\exists i, j : F_i(u) \text{ and } F_j(v) \text{ are differ in all positions where } u \text{ and } v \text{ differ,}$$

*i.e. for these two  $u, v \in V$  the properness condition does not hold.*

2. *Find all components of the graph  $\mathcal{G}$ .*

3. Define a function  $f_{n+1}(x_1, \dots, x_{n+1}) = f_{n+1}(x_1, \dots, x_n)$  (variable  $x_{n+1}$  is dummy for  $f_{n+1}$ ) in such a way that  $f_{n+1}$  is a constant value on each of the components of  $\mathcal{G}$ .

Then the family  $(F, f_{n+1}): E_k^{n+1} \rightarrow E_k^{n+1}$  is proper, where  $F$  is defined by formula (8), and any family of size  $n + 1$  can be obtained using this construction by selecting suitable  $F_m$ .

A possible application of Theorem 14 is the construction of all proper families of the order  $n + 1$  by exhausting all possible combinations of proper families  $F_0, \dots, F_{k-1}$  of the order  $n$  and adding all possible functions  $f_{n+1}$ . Another application is specifying the transformation of proper families that consists in selection of a variable  $x_i$ , construction of projections defined by the equalities  $x_i = j$  and restoration of the “new” function  $f_i$  based on the projections. It can be shown that equiprobable selection of the index  $i$  and the new function generates the Markov chain that converges to uniform distribution of the set of all proper families of the given order.

## 6 Recognition of properness

It was shown earlier that for the Boolean case [3] and (as a consequence) for the case of  $k$ -valued logic [22] the problem of recognizing whether the family is proper is coNP-complete. Hence, it seems that no efficient algorithm for the general case is possible.

As it was shown in [21], the number of proper families among all families of size  $n$  over  $E_k^n$  with the property that  $x_i$  is dummy for  $f_i$  tends to 0 as  $n \rightarrow \infty$ . Hence, generating random family and checking whether it is proper or not (using the basic definition or any of the equivalent definitions of properness listed in Section 4) is rather inefficient.

Restricting the class of functions to a more specific allows one to use more effective criteria for checking properness. For instance, the graph of essential dependence (see Definition 11 below) in some cases (linear functions,  $g$ -functions, monotonic Boolean functions [23]) can provide enough information for significant speedup of checking the properness condition.

## 7 Graphs and matrices of essential dependence

The intrinsic structure of families of functions can be visualized using the constructions of their graphs and matrices of essential dependence.

**Definition 11.** Let  $F = (f_i)_{i=1}^n$ ,  $f_i = f_i(z_1, z_2, \dots, z_n)$ , be a family of  $n$  functions, each in  $n$  variables. The graph of essential dependence of the family  $F$  is the directed graph  $G_F = (V, E)$  on the set of vertices  $V = \{1, 2, \dots, n\}$  whose edges are defined by the condition that  $(i, j) \in E$  if and only if  $f_j$  essentially depends on  $z_i$ .

It is known [7] that a linear family is proper if and only if its graph of essential dependence contains no directed cycles.

The class of functions for which the properness of a family is equivalent to the absence of directed cycles in the graph can be extended to the so-called  $g$ -functions. Namely, given an element  $g = (g_1, g_2, \dots, g_n)$  we say that a function  $f = f(x_1, \dots, x_n)$  is a  $g$ -function if for any its essential variable  $x_i$  the unary function  $f'(x_i) = f(g_1, \dots, g_{i-1}, x_i, g_{i+1}, \dots, g_n)$  is not a constant. It can be shown [12] that for a fixed  $g = (g_1, g_2, \dots, g_n)$  a family of  $g$ -functions is proper iff its graph of essential dependence is free of directed cycles.

On the other hand, the graph of a proper family of functions may be rather rich in cycles. For instance, it is easily seen that the proper family of pairwise orthogonal functions (1) has a complete graph of essential dependence.

It has been already mentioned that in a proper family  $F = (f_i)_{i=1}^n$  each function  $f_i = f_i(z_1, z_2, \dots, z_n)$  does not essentially depend on  $z_i$ . At the same time, with this necessary condition being fulfilled, any family of functions can be extended to a proper family of size  $n' \leq n + \lceil \log_2 n \rceil$ . In terms of graphs this result is formulated as follows.

**Theorem 15** ([11, Theorem 7]). Let  $G(V, E)$  be an arbitrary directed graph without loops and multiple edges on  $n$  vertices  $V = \{1, 2, \dots, n\}$ . Then there exists a graph  $G'(V', E')$  on  $n' \leq n + \lceil \log_2 n \rceil$  vertices  $V' = \{1, 2, \dots, n'\}$  that can be treated as the graph of essential dependence of a proper family of functions and such that its subgraph induced by the subset  $V \subseteq V'$  coincides with  $G(V, E)$ . Moreover, for any family of functions  $F = (f_i)_{i=1}^n$  realizing the original graph  $G(V, E)$  one can find a proper family of functions  $F' = (f'_i)_{i=1}^{n'}$  that realizes the graph  $G'(V', E')$  and such that for every  $i$ ,  $1 \leq i \leq n$ , there exists an evaluation of arguments  $x_{n+1}, \dots, x_{n'}$  such that  $f'_i$  as a function of  $n$  arguments  $x_1, \dots, x_n$  coincides with  $f_i$ .

**Remark 4** ([11]). If the set  $V$  of vertices of the graph  $G(V, E)$  can be partitioned into two subsets  $V = V_0 \sqcup V_1$  in such a way that any directed cycle of the graph  $G(V, E)$  contains vertices of both subsets  $V_0, V_1$ , then the graph  $G'(V', E')$  mentioned in Theorem 15 needs only one additional vertex.

Various properties of families of functions can also be formulated in terms of their matrices of essential dependence.

**Definition 12.** *The matrix of essential dependence of a family  $F = (f_i)_{i=1}^n$  of functions  $f_i = f_i(z_1, z_2, \dots, z_n)$  is a square  $(0, 1)$ -matrix  $A$  of size  $n$  whose entry  $A_{ij}$  equals one if and only if  $f_j$  essentially depends on  $z_i$ .*

Examples of results on the matrices of essential dependence of proper families of functions can be found in the paper [11].

## 8 Quadratic and strongly quadratic proper families

Assume that  $d = 2$  (we consider quasigroups) and  $k$  in representation (4) is a prime power. Then the functions  $g_i$  can be represented in algebraic normal form, i.e. by polynomials (see e.g. [24, Theorem 1.4.3]). If the degrees of all polynomials are at most 2 and there exists at least one quadratic polynomial, then the quasigroup is referred to as a multivariate quadratic quasigroup (MQQ). If additionally all non-trivial linear combinations of the functions  $g_i$  are quadratic, then the quasigroup is said to be strongly quadratic. MQQ can be used to design public-key cryptographic algorithms (see e.g. [17]); strongly quadratic quasigroups guarantee that the order of a system of equations on key bits can not be reduced by computing linear combinations of equations. A number of methods for generation of MQQ are proposed in [25].

Proper families of functions are a convenient apparatus to specify large families of MQQs of a large order [13]. Representation (5) suggests two strategies: use a proper family specified by linear polynomials and substitute arbitrary internal functions  $\pi_j$  so that at least one of the functions  $g_i$  in the representation (4) is quadratic, or use a quadratic proper family and linear internal functions  $\pi_j$ . In the former case it is known that all linear proper families are triangular [7], thus strongly quadratic MQQs can not be generated by this construction. Possible direct enhancements include switching to the representation (7) and/or considering quadratic triangular families. For the case  $k = 2$  the cardinality of the set of MQQs generated by this construction is  $2^{n^3/6+o(n)}$ ,  $n \rightarrow \infty$  [13]. All of these MQQ can also be generated by the  $t$ -construction described e.g. in [25], however a proper family-based representation consumes less memory.

Another possibility is using a proper family such that all functions are quadratic. An example for  $k = 2$  and  $n \geq 3$  is the family (3). This family is not strongly quadratic in a sense that a non-trivial linear combination is admitted. A strongly quadratic example is the family (2) which is proper for  $k = 2$  and odd  $n \geq 3$ . Combining the family (2), a triangular extension and a generalization of Theorem 5 to the case of triangular extensions

allows one to generate  $2^{n^3/6+o(n)}$ ,  $n \rightarrow \infty$ , strongly quadratic MQQs of the order  $2^n$  [13], some of which can not be generated by methods from [25].

Generalization of the first construction to the case of  $k$  equal to an arbitrary prime power is straightforward. Generalization of the second construction requires a strongly quadratic proper family. Construction of such a family for  $k \neq 2$  is the subject of future research.

## 9 Concluding remarks

Proper families of functions are a promising apparatus for memory-efficient specification of large parametric families of quasigroups and  $d$ -quasigroups of a high order, possibly with additional constraints, e.g. the degrees of polynomials in ANF. Quasigroups and 3-quasigroups are currently extensively used to construct various cryptographic primitives such as symmetric and public-key ciphers, hash functions, digital signatures, etc. (see e.g. the review [1]); proper families of functions can be used to select quasigroup-based parameters for future solutions providing a number of cryptographically beneficial properties (for example high cardinality of the key space ensured by variation of internal functions). There exist proper family-based methods for constructing quasigroups and  $d$ -quasigroups with additional properties, e.g. MQQ [13] suitable for multivariate quadratic cryptography. Other important properties such as simplicity, non-affinity and absence of proper subquasigroups can be decided after generation using one of the existing decision algorithms (see e.g. [26]); selection of generation parameters that guarantee one of these properties is the subject of future research.

## References

- [1] Shcherbacov V. A., “Quasigroups in cryptology”, arXiv:1007.3572.
- [2] Dömösi P., Horváth G., “A novel cryptosystem based on abstract automata and Latin cubes”, *Studia Scientiarum Mathematicarum Hungarica*, **52:2** (2015), 221–23.
- [3] Nosov V. A., “The criterion of regularity of a Boolean non-autonomous automaton with split input”, *Intelligent Systems (Intellektualnye Sistemy)*, **3:3–4** (1998), 269–280, In Russian.
- [4] Nosov V. A., “Construction of classes of Latin squares in a Boolean database”, *Intelligent Systems (Intellektualnye Sistemy)*, **4:3–4** (1999), 307–320, In Russian.
- [5] Nosov V. A., “Construction of a parametric family of Latin squares in a vector database”, *Intelligent Systems (Intellektualnye Sistemy)*, **8:1–4** (2004), 517–528, In Russian.
- [6] Nosov V. A., “Constructing Families of Latin Squares over Boolean Domains”, *Boolean Functions in Cryptology and Information Security*, IOS Press, Amsterdam, 2008, 200–2007.
- [7] Nosov V. A., Pankratiev A. E., “Latin squares over Abelian groups”, *Journal of Mathematical Sciences*, **163:5** (2009), 53–542.

- 
- [8] Galatenko A. V., Nosov V. A., Pankratiev A. E., “Latin squares over quasigroups”, *Lobachevskii Journal of Mathematics*, **41**:2 (2020), 194–203.
- [9] Tsaregorodtsev K. D., “Properties of proper families of Boolean functions”, *Diskr. Mat.*, **33**:1 (2021), 91–102, In Russian.
- [10] Nosov V. A., Pankratiev A. E., “On functional specification of Latin squares”, *Journal of Mathematical Sciences*, **169**:4 (2010), 533–540.
- [11] Kozlov A. A., Nosov V. A., Pankratiev A. E., “Matrices and Graphs of Essential Dependence of Proper Families of Functions”, *Journal of Mathematical Sciences*, **163**:5 (2009), 534–542.
- [12] Nosov V. A., Pankratiev A. E., “Families of functions specifying Latin squares over Abelian groups”, *Vestnik Moskovskogo Gosudarstvennogo Universiteta Lesa – Lesnoi Vestnik (Vestnik of Moscow State Forest University – Forestry Bulletin)*, **2** (2007), 141–144, In Russian.
- [13] Galatenko A. V., Nosov V. A., Pankratiev A. E., “Generation of multivariate quadratic quasigroups by proper families of Boolean functions”, *Fundamental and Applied Mathematics (Fundamentalnaya i Prikladnaya Matematika)*, **23**:2 (2020), 57–73, In Russian.
- [14] Plaksina I. A., “Construction of a parametric family of multidimensional Latin squares”, *Intelligent Systems. Theory and Applications (Intellektualnye Sistemy. Teoria i Prilojenia)*, **18**:2 (2014), 323–330, In Russian.
- [15] Galatenko A. V., Nosov V. A., Pankratiev A. E., “Functional specification of quasigroup operations”, Proceedings of the XIX International Conference “Problems of Theoretical Cybernetics” (Kazan, 2021), 2021, 35–37.
- [16] Piven N. A., “Investigation of quasigroups generated by proper families of Boolean functions of the order 2”, *Intelligent Systems. Theory and Applications (Intellektualnye Sistemy. Teoria i Prilojenia)*, **22**:1 (2018), 21–35, In Russian.
- [17] Gligoroski D., Markovski S., Knapskog J., “Multivariate quadratic trapdoor functions based on multivariate quadratic quasigroups”, Proceedings of the American Conference on Applied Mathematics (MATH’08), 2008, 44–49.
- [18] Szabo T., Welzl E., “Unique sink orientations of cubes”, Proceedings 42nd IEEE Symposium on Foundations of Computer Science, 2001, 547–555.
- [19] Tsaregorodtsev K. D., “One-to-one correspondense between proper families of Boolean functions and unique sink orientations of cubes”, *Applied Discrete Mathematics (Prikladnaya Diskretnaya Matematika)*, 2020, 16–21, In Russian.
- [20] Galatenko A. V., Nosov V. A., Pankratiev A. E., Staroverov V. M., “A criterion of properness of a family of functions”, Proceedings of the XIX International Conference “Algebra, Number Theory, Discrete Geometry and Multiscale Modeling: modern problems, applications and problems of history” (Tula, 2021), 2021, 103–106, In Russian.
- [21] Galatenko A. V., Nosov V. A., Pankratiev A. E., “An algorithm for generation of proper families of functions”, Proceedings of the XVIII International Conference “Algebra, Number Theory and Discrete Geometry: modern problems, applications and problems of history” (Tula, 2020), 2020, 141–145, In Russian.
- [22] Galatenko A. V., Nosov V. A., Pankratiev A. E., Staroverov V. M., “Generation of proper families of functions”, *Intelligent Systems. Theory and Applications (Intellektualnye Sistemy. Teoria i Prilojenia)*, **25**:4 (2021), 100–103, In Russian.
- [23] Rykov D. O., “Algorithms for checking properness of family of functions”, *Intelligent Systems. Theory and Applications (Intellektualnye Sistemy. Teoria i Prilojenia)*, **14**:1–4 (2010), 261–276, In Russian.
- [24] Lau D., *Function algebras on finite sets: a basic course on many-valued logic and clone theory*, Springer, 2006.
- [25] Samardjiska S., Chen Y., Gligoroski D., “Construction of Multivariate Quadratic Quasigroups (MQQs) in arbitrary Galois fields”, *2011 7th International Conference on Information Assurance and Security (IAS)*, 2011, 314–319.
- [26] Galatenko A. V., Pankratiev A. E., Staroverov V. M., “Algorithms for Checking Some Properties of  $n$ -Quasigroups”, *Programming and Computer Software*, **48**:1 (2022), 36–48.

# PUBLIC-KEY CRYPTOGRAPHY



# On the (im)possibility of ElGamal blind signatures

Liliya Akhmetzyanova, Evgeny Alekseev, Alexandra Babueva, and  
Stanislav Smyshlyaev

CryptoPro LLC, Russia  
{lah, alekseev, babueva, svb}@cryptopro.ru

## Abstract

In the current paper we investigate the possibility of designing secure blind signature scheme based on ElGamal signature equation. We define the generalized construction and analyze its security. We consider two types of schemes with the proposed construction, that cover all existing schemes. For schemes of the first type we provide generic ROS-style attack that violates unforgeability in the parallel setting. For schemes of the second type we prove that they do not provide either blindness, or unforgeability. As the result, we prove that all known ElGamal blind signature schemes are not secure. Moreover, these results show that the existence of secure ElGamal blind signature scheme is potentially possible only for small set of signature equations and requires the non-standard way of generating the first component of the signature.

**Keywords:** ElGamal signature scheme, blind signature scheme, ROS attack.

## 1 Introduction

Blind signature schemes are widely used in many applications that guarantee user anonymity, e.g. e-voting [8] and e-cash [4] systems. They allow the Requester to obtain a signature for an arbitrary message after interacting with the Signer in such a way that the Signer does not receive any information about either the message or the signature value (blindness property) and the Requester can compute only one single signature per interaction with the Signer (unforgeability property).

ElGamal signature scheme [6] is one of the most well-studied and widely-deployed signature schemes. Thus, development of blind signature scheme based on it is a relevant task. And sure enough, there exists a variety of blind signature schemes based on ElGamal signature equation [5, 10, 12, 13, 14, 17, 19, 20, 21, 24]. However, the unforgeability of these schemes was not formally

proven under some relevant assumptions. At the same time, no attacks on these schemes were proposed. So, their security remains an open question. The only exception is the scheme introduced in [24], which additionally uses homomorphic encryption and non-interactive zero-knowledge proof (NIZK) for providing blindness. Its unforgeability was proven in [16] in the so-called algebraic bijective random oracle model. However, this scheme is not nearly as interesting for us since it uses the additional cryptographic mechanisms.

In the current paper we examine the possibility of constructing secure blind signature scheme based only on ElGamal signature equation. We introduce generalized ElGamal blind signature scheme called **GenEG-BS**. The signing protocol in this scheme is fixed only on the Signer side, where the ElGamal signature generation algorithm is performed for masked hash-value  $e$  generated on the Requester side in an arbitrary way. **GenEG-BS** construction covers all existing blind signature schemes based on ElGamal equation except for the scheme [24], in which the Signer side involves, in particular, verifying the NIZK proof.

We study the security of the **GenEG-BS** schemes. It turned out that the ROS attack [3], that breaks the security of blind Schnorr signature [18], can be adapted to break several **GenEG-BS** schemes. We provide the generic ROS-style attack on these schemes violating unforgeability in the parallel setting and the necessary condition for its applicability. Further we consider the schemes that are not vulnerable to the ROS-style attack. More specifically, we study the particular case of these schemes for which the way of generating the first component of the signature on the Requester side is fixed. We prove that such schemes do not provide either unforgeability, or blindness. As the consequence, we show that all existing **GenEG-BS** schemes [5, 10, 12, 13, 14, 17, 19, 20, 21] are not secure. Moreover, we identify the form of ElGamal signature equation that can potentially lie in the heart of the secure **GenEG-BS** scheme. However, the construction of such scheme requires the radically new method of generating the first component of the signature.

## 2 Basic notations and definitions

By  $\{0, 1\}^*$  we denote the set of all bit strings of finite length including the empty string. If  $p$  is a prime number then the set  $\mathbb{Z}_p$  is a finite field with characteristic  $p$ . We assume the canonic representation of the elements in  $\mathbb{Z}_p$  as integers in the interval  $[0 \dots p - 1]$ . Each non-zero element  $x$  in  $\mathbb{Z}_p$  has an inverse  $1/x$ . We define  $\mathbb{Z}_p^*$  as the set  $\mathbb{Z}_p$  without zero element.

We denote the group of points of elliptic curve over the field  $\mathbb{Z}_p$  as  $\mathbb{G}$ , the order of the prime subgroup of  $\mathbb{G}$  as  $q$  and elliptic curve point of order  $q$  as  $P$ . We denote by  $H$  the hash function that maps binary strings to elements from  $\mathbb{Z}_q$  and assume that all field operations are performed modulo  $q$ .

If the value  $s$  is chosen from a set  $S$  uniformly at random, then we denote  $s \stackrel{\mathcal{U}}{\leftarrow} S$ . If the variable  $x$  gets the value  $val$  then we denote  $x \leftarrow val$ . Similarly, if the variable  $x$  gets the value of the variable  $y$  then we denote  $x \leftarrow y$ . If the variable  $x$  gets the result of an algorithm  $A$  we denote  $x \leftarrow A$ .

The blind signature scheme is determined by three algorithms:

- $(sk, pk) \leftarrow \mathbf{KGen}$ : a key generation algorithm that outputs a secret key  $sk$  and a public key  $pk$ ;
- $(b, \sigma) \leftarrow \langle \mathbf{Signer}(sk), \mathbf{Requester}(pk, m) \rangle$ : an interactive signing protocol that is run between a Signer with a secret key  $sk$  and a Requester with a public key  $pk$  and a message  $m$ ; the Signer outputs  $b = 1$  if the interaction completes successfully and  $b = 0$  otherwise, while the Requester outputs a signature  $\sigma$  if it terminates correctly, and a fail indicator  $\perp$  otherwise.
- $b \leftarrow \mathbf{Vf}(pk, m, \sigma)$ : a (deterministic) verification algorithm that takes a public key  $pk$ , a message  $m$ , and a signature  $\sigma$ , and returns 1 if  $\sigma$  is valid on  $m$  under  $pk$  and 0 otherwise.

### 3 ElGamal blind signature scheme

**Standard ElGamal signature scheme.** The generalised ElGamal type signature scheme was introduced in [11] and further extended in [7]. A key generation algorithm involves picking random  $d$  uniformly from  $\mathbb{Z}_q^*$  (secret signing key) and defining  $Q = dP$  (public verifying key).

A signing algorithm for message  $m$  involves computing hash-value  $e = H(m)$ , picking random  $k$  uniformly from  $\mathbb{Z}_q^*$  and defining  $r$  value as  $kP.x \bmod q$ . To ensure functionality and security, certain such values need to be excluded. The  $s$  value is determined from the ElGamal signature equation. According to [11], ElGamal signature equation is defined as follows:

$$G_d(r, e, s) \cdot d + G_k(r, e, s) \cdot k + G_0(r, e, s) = 0, \quad (1)$$

where  $G_d, G_k, G_0$  are the functions  $\mathbb{Z}_q^3 \rightarrow \mathbb{Z}_q^*$  that are affine by  $z$  or  $z^{-1}$  for all  $z \in \{r, e, s\}$ . If there exists a unique  $s$  such that the equation (1) is

satisfied, then the signing algorithm returns  $(r, s)$  pair as the signature value, otherwise it returns the fail indicator.

For example, GOST [25] signature equation refers to ElGamal signature equations, where  $s$  is calculated as  $ke + dr$ , i.e.  $G_d(r, e, s) = r, G_k(r, e, s) = e, G_0(r, e, s) = -s$ . In [11] all possible ElGamal signature equations are listed (here the difference between  $+z$  and  $-z$  and the difference between  $z$  and  $z^{-1}$  is neglected, where  $z \in \{r, e, s, k, d\}$ ):

$$\begin{array}{lll}
 1 : & ed = rk + s & 7 : \quad red = k + s & 13 : \quad (r + e)d = k + s \\
 2 : & ed = sk + r & 8 : \quad d = rek + s & 14 : \quad d = (r + e)k + s \\
 3 : & rd = ek + s & 9 : \quad sd = k + re & 15 : \quad sd = k + (r + e) \\
 4 : & rd = sk + e & 10 : \quad d = sk + re & 16 : \quad d = sk + (r + e) \\
 5 : & sd = rk + e & 11 : \quad red = sk + 1 & 17 : \quad (r + e)d = sk + 1 \\
 6 : & sd = ek + r & 12 : \quad sd = rek + 1 & 18 : \quad sd = (r + e)k + 1
 \end{array}$$

Figure 1: ElGamal signature equations

In the current paper we rely on this list and do not consider its completeness.

The verify procedure for the message  $m$  and the signature  $(r, s)$  assumes verifying the equality

$$r = R \cdot x \pmod{q},$$

where  $R = -\frac{1}{G_k(r, e, s)} (G_d(r, e, s) \cdot Q + G_0(r, e, s) \cdot P)$ ,  $e = H(m)$ .

**ElGamal blind signature scheme.** We define the general ElGamal blind signature scheme. A key generation algorithm is the same as in the standard ElGamal signature scheme.

The signing protocol is defined at Figure 2. The value  $e$  is always generated on the Requester side and forwarded to the Signer. The Signer performs ElGamal signature generating algorithm.

The verify procedure is the same as in the standard ElGamal signature scheme. We denote all blind signature schemes of this type as **GenEG-BS** schemes.

## 4 Security notions

Blind signature schemes should provide two security properties: unforgeability and blindness. In the current section we introduce the corresponding security notions by defining the threat and the adversary capabilities in each

The signing protocol

---

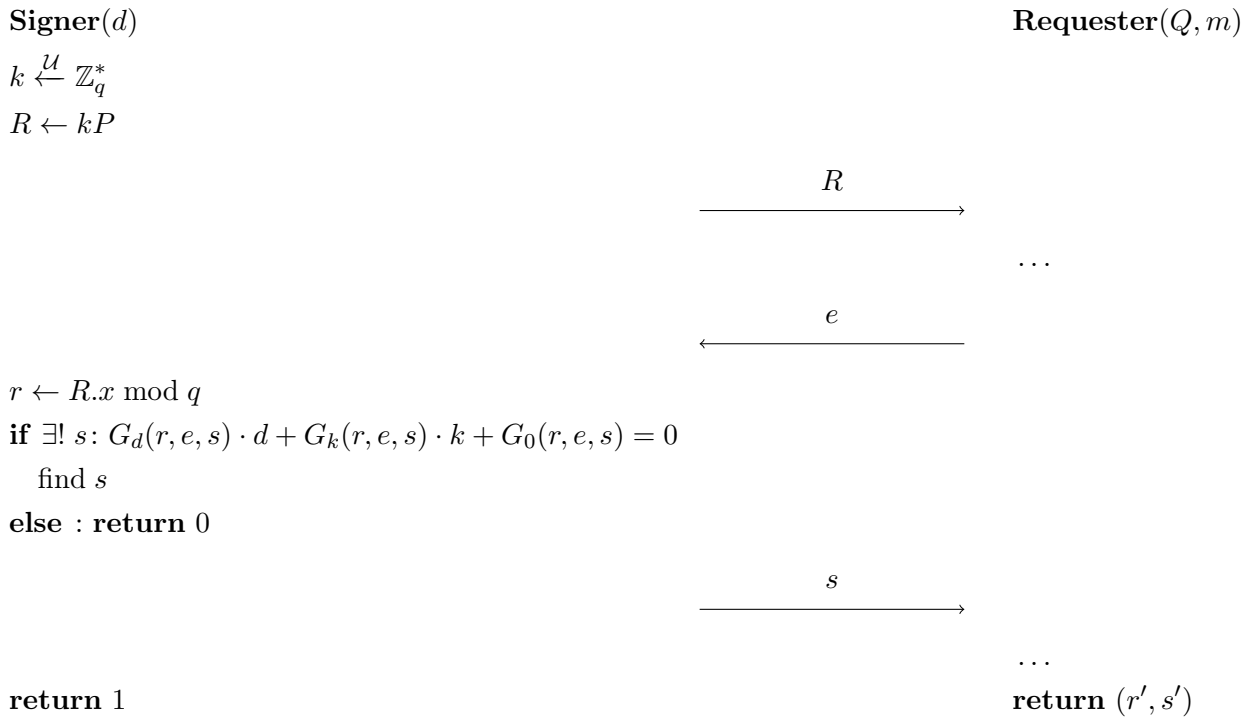


Figure 2: GenEG-BS scheme: the signing protocol

case. The formal definitions of these notions for two-round blind signature schemes are introduced, for example, in [9].

**Unforgeability.** An adversary acts as a malicious Requester and is powered to run the signing protocol with the Signer, scheduling and interleaving the sessions in any arbitrary way. In particular, it can open many parallel sessions with the Signer. It is assumed that the Signer behaves correctly (according to the protocol).

An adversary’s task (threat), after interacting arbitrary many times with the Signer and  $l$  of these interactions were considered successful by the Signer, is to produce more than  $l$  valid (message, signature) pairs. The threat is considered strong if all messages should be distinct and weak if all (message, signature) pairs should be distinct.

In some cases the weak notion, in which the adversary is powered to open only the sequential sessions with the Signer, is considered.

**Blindness.** Informally, the blind signature scheme provides blindness if there is no way to link a (message, signature) pair to the certain execution of the signing protocol. In other words, the blindness is broken if the particular

protocol execution for some fixed message leads to fixing the signature value in an unambiguous way or at least to significant narrowing the set of possible signature values.

Here an adversary acts as a malicious Signer and is powered to run the signing protocol with the Requester twice. It is assumed that the Requester behaves correctly (according to the protocol). After two successful interactions the Requester outputs two (message, signature) pairs simultaneously. If at least one of the interactions failed, the Requester outputs fail indicator.

An adversary's task (threat) is to link the transcription of the protocol to the corresponding (message, signature) pair with success probability significantly greater than  $1/2$ . The unlinkability can be either computational, in which case we talk about computational blindness, or information-theoretical, we then talk about perfect blindness.

## 5 Security of the GenEG-BS schemes

We study the possibility of constructing secure ElGamal blind signature scheme **GenEG-BS**. Note that all existing **GenEG-BS** schemes were introduced without formal unforgeability proof, the blindness proof is presented only for some of them. Therefore, the security of these schemes remains an open question.

Well in our research, we identified two types of **GenEG-BS** schemes. They cover all existing schemes of such type [5, 10, 13, 14, 17, 19, 20, 21]. We show that schemes of both types are not secure and do not provide either unforgeability, or blindness.

The starting point for distinguishing two types of the **GenEG-BS** schemes was the study of the possibility of applying the ROS attack [3] to such schemes. ROS (Random inhomogeneities in an Overdetermined, Solvable system of linear equations) problem was introduced by Schnorr [18] and was considered intractable for some time. However, later it was reduced to the  $(l + 1)$ -sum problem, for which Wagner's [23] generalized birthday algorithm (with sub-exponential complexity) can be used. Finally, polynomial-time attack against ROS problem (ROS-attack) was proposed in 2020 in [3], that implies polynomial-time attack against blind Schnorr signature scheme in case an adversary is able to open  $l \geq \lceil \log q \rceil$  parallel sessions with the Signer. In fact, not only the Schnorr scheme [15] was broken, but also the Okamoto-Schnorr scheme [15] and the partially blind Abe scheme [1]. Therefore, the question of the applicability of the attack to the **GenEG-BS** schemes is quite natural.

**First type.** It turned out that the modification of the ROS attack is applicable to a significant number of existing schemes [5, 12, 13, 14, 17, 19, 21]. We provide the necessary condition for its applicability as the restrictions on the signature equation.

**Condition 1:** at least one of the function  $\frac{1}{G_k(r, e, s)} \cdot G_d(r, e, s)$  or  $\frac{1}{G_k(r, e, s)} \cdot G_0(r, e, s)$  does not significantly depend on  $s$ .

All GenEG-BS schemes with signature equation satisfying the Condition 1 will be called the schemes of Type I. For such schemes we construct generic ROS-style attack, violating unforgeability, thereby proving the following theorem.

**Theorem 1.** *If GenEG-BS scheme satisfies the Condition 1, then it does not provide unforgeability when the number of parallel sessions  $l \geq \lceil \log q \rceil$ .*

See Section 5.1 for attack description and discussion on Condition 1. Note that these attack is applicable in the standard model in which the adversary can open parallel sessions with the Signer. The security of such schemes relative to the weak adversary that can open only sequential sessions is the open question.

**Second type.** Consider ElGamal signature equations for which the Condition 1 is not satisfied. These are equations 2, 4, 10, 11, 16 at Figure 1, all of them have the following form:

$$sk = F_1(r, e)d + F_2(r, e) \quad (2)$$

or

$$s^{-1}k = F_1(r, e)d + F_2(r, e), \quad (3)$$

where  $F_1$  and  $F_2$  functions are affine functions by  $z$  or  $z^{-1}$  for all  $z \in \{r, e\}$ . Moreover, only one of the functions  $F_1$  and  $F_2$  significantly depends on  $r$ .

We obtain the result for the particular case of the GenEG-BS schemes based on these equations, in which the  $r'$  component of the signature is generated on the Requester side in the following way:

$$R' \leftarrow \alpha R + \beta Q + \gamma P, \quad r' \leftarrow R'.x \bmod q, \quad (4)$$

where each of the  $\alpha, \beta, \gamma$  values (called blinding factors) are chosen uniformly from  $\mathbb{Z}_q^*$  by the Requester or are equal to zero. We consider exactly uniform distribution on the blinding factor values, since other distributions seem not

to allow to reach perfect blindness. All existing schemes known to the authors assume exactly this way of generation of the  $r'$  component (regardless of the signature equation type). Thus, these results are important in terms of practice.

Finally, we call **GenEG-BS** scheme a scheme of Type II, if:

- the signature equation has the form (2) or (3);
- the  $r'$  component is generated according to (4).

The only known blind signature scheme of Type II is the scheme, introduced in [10]. However, the attack, violating blindness, on this scheme was presented in [2]. This attack leads us to consider the following condition.

Let  $(R, e, s)$  be the transcription of the signing protocol execution and  $r = R.x \bmod q$ . Let  $(r', s')$  be the signature value produced by the Requester for some message  $m$  with hash-value  $e' = H(m)$  after that execution.

**Condition 2:** for all possible key pairs  $(d, Q)$  the equation  $F_1(r, e) \cdot F_2(r', e') = F_1(r', e') \cdot F_2(r, e)$  holds with the overwhelming probability.

Here the probability space consists of all values representing random choices made by the Signer and the Requester randomized algorithms.

It turned out that this condition provides the criteria to link the given protocol transcription and the (message, signature) pair. We state the following theorem, see Section 5.2 for its proof.

**Theorem 2.** *If GenEG-BS scheme of Type II satisfies the Condition 2, then it does not provide blindness.*

To the best of our knowledge, there exist no **GenEG-BS** schemes of Type II, for which the Condition 2 is not satisfied. This observation allowed us to prove the following theorem, justifying the impossibility of constructing a secure blind signature scheme of this type.

**Theorem 3.** *If there exists GenEG-BS scheme of Type II that does not satisfy the Condition 2, then it does not provide unforgeability.*

The main idea of the proof is to show that the existence of such scheme leads either to the secret signing key recovering from the protocol transcription and the signature value obtained after the protocol execution, or to the ability to make valid signatures without secret key knowledge. See Section 5.3 for the full proof.

Summing up, we show that **GenEG-BS** schemes of Types I and II are not secure. Which means that if the secure **GenEG-BS** scheme exists, then



it is based on the equations (2) or (3) and assumes radically new way of generating the  $r'$  component, not according to (4).

## 5.1 ROS-style attack

Consider the general ElGamal signature equation (1). According to the Condition 1, at least one of the function  $\frac{G_d(r, e, s)}{G_k(r, e, s)}$  or  $\frac{G_0(r, e, s)}{G_k(r, e, s)}$  does not significantly depend on  $s$ . We denote the function that does not depend on  $s$  by  $Y_1(r, e)$  and the another one function by  $Y_2(r, e, s)$ . Therefore, the signature equation can be represented in the following way:

$$k + Y_1(r, e) \cdot G_1(d) + Y_2(r, e, s) \cdot G_2(d) = 0, \quad (5)$$

where  $G_1$  and  $G_2$  functions are equal to tautology or identity function, and  $G_1(d) \cdot G_2(d) = d$ . Note that  $Y_1$  function significantly depends on  $e$  and  $r$  values in all signature equations listed at Figure 1 and satisfied the above representation.

Verify procedure for message  $m$  and signature  $(r, s)$  assumes verifying the equality

$$r = R.x \bmod q,$$

where  $R = -Y_1(r, e) \cdot G_1(d)P - Y_2(r, e, s) \cdot G_2(d)P$ ,  $e = H(m)$ . Note that  $G_z(d)P = P$  or  $G_z(d)P = Q$  for  $z \in \{1, 2\}$ .

The attack, presented below, allows an adversary to construct  $(l + 1)$  valid (message, signature) pairs after  $l \geq \lceil \log q \rceil$  successful interactions with the Signer. The adversary acts as follows:

1. Selects message  $m_l \in \{0, 1\}^*$  for which a signature will be forged, let  $e_l = H(m_l)$ .
2. Opens  $l$  parallel sessions, querying the Signer, and receives corresponding points  $R_0, \dots, R_{l-1}$ .
3. Calculates  $r_i = R_i.x \bmod q, 0 \leq i \leq l - 1$ .
4. Selects  $m_i^0, m_i^1 \in \{0, 1\}^*, 0 \leq i \leq l - 1$ , such that  $r'_{i0} = Y_1(r_i, e_i^0) \neq Y_1(r_i, e_i^1) = r'_{i1}$ , where  $e_i^0 = H(m_i^0), e_i^1 = H(m_i^1)$ .
5. Defines  $(\rho_0, \rho_1, \dots, \rho_l)$  as the vector of coefficients placed before  $x_i$  in the

$$\text{function } f : \mathbb{Z}_q^l \rightarrow \mathbb{Z}_q; f(x_0, \dots, x_{l-1}) = \sum_{i=0}^{l-1} 2^i \underbrace{\frac{x_i - r'_{i0}}{r'_{i1} - r'_{i0}}}_{b'_i} = \sum_{i=0}^{l-1} \rho_i x_i + \rho_l.$$

Note that if  $x_i = r'_{i0}$  then  $b'_i = 0$ , if  $x_i = r'_{i1}$  then  $b'_i = 1$ .

6. Defines  $R_l = \sum_{i=0}^{l-1} \rho_i R_i - \rho_l G_1(d)P$ .
7. Defines  $r_l = R_l \cdot x \pmod q$ .
8. Defines  $b_0, \dots, b_{l-1}$  from the following equation:  $Y_1(r_l, e_l) = \sum_{i=0}^{l-1} 2^i b_i$ .
9. Defines  $r'_i = r'_{ib_i}, e_i = e_i^{b_i}, m_i = m_i^{b_i}, 0 \leq i \leq l-1$ ; therefore, according to step 5,  $Y_1(r_l, e_l) = \sum_{i=0}^{l-1} \rho_i r'_i + \rho_l = \sum_{i=0}^{l-1} \rho_i Y_1(r_i, e_i) + \rho_l$ .
10. Sends  $e_0, \dots, e_{l-1}$  values to the Signer in the corresponding sessions;
11. Obtains responses  $s_0, \dots, s_{l-1}$  such that:
 
$$R_i + Y_1(r_i, e_i) \cdot G_1(d)P + Y_2(r_i, e_i, s_i) \cdot G_2(d)P = 0, \quad 0 \leq i \leq l-1.$$
12. Defines  $s_l$  in such a way that the following equality is satisfied:

$$\sum_{i=0}^{l-1} \rho_i Y_2(r_i, e_i, s_i) = Y_2(r_l, e_l, s_l).$$

According to our notations, the  $Y_2(r_l, e_l, s_l)$  function is equal to  $\frac{G_z(r_l, e_l, s_l)}{G_k(r_l, e_l, s_l)}$ , where  $z \in \{0, d\}$ . Thus, the above equation is equivalent to the following:

$\sum_{i=0}^{l-1} \rho_i Y_2(r_i, e_i, s_i) G_k(r_l, e_l, s_l) = G_z(r_l, e_l, s_l)$ , and is affine by  $s_l$  since  $G_k, G_z$  functions are affine by  $s_l$ . It can be represented as  $a_1 s_l + a_2 = 0$ , where  $a_1, a_2$  are the fixed values from  $\mathbb{Z}_q$  that depend on  $d, e_l, R_i, e_i^0, e_i^1, 0 \leq i \leq l-1$ , values. Thus, if  $a_1 \neq 0$ , it is possible to efficiently find the  $s_l$  value such that the equation is satisfied. If  $a_1 = 0$ , the adversary returns to step 1. For all ElGamal equations listed at Figure 1, for any fixed signing key  $d$  and for any values  $e_l, e_i^0, e_i^1, 0 \leq i \leq l-1$ , selected by the adversary, the condition  $a_1 = 0$  holds with the negligible probability over the random choice of  $R_i$  values by the Signer algorithm.

13. Outputs  $\{m_i, (r_i, s_i)\}_{i=0}^l$ .

Indeed, for  $0 \leq i \leq l-1$  signature  $(r_i, s_i)$  is valid for  $m_i$  by attack construction, see step 11. Consider the case  $i = l$ . Summarize the equations

obtained at step 11 with the corresponding coefficients:

$$\sum_{i=0}^{l-1} \rho_i R_i + \sum_{i=0}^{l-1} \rho_i Y_1(r_i, e_i) \cdot G_1(d)P + \sum_{i=0}^{l-1} \rho_i Y_2(r_i, e_i, s_i) \cdot G_2(d)P = 0.$$

Subtract and add the term  $\rho_l G_1(d)P$  in the left part of the equation:

$$\underbrace{\sum_{i=0}^{l-1} \rho_i R_i - \rho_l G_1(d)P}_{=R_l} + \underbrace{\left( \sum_{i=0}^{l-1} \rho_i Y_1(r_i, e_i) + \rho_l \right)}_{=Y_1(r_l, e_l)} \cdot G_1(d)P + \underbrace{\sum_{i=0}^{l-1} \rho_i Y_2(r_i, e_i, s_i) \cdot G_2(d)P}_{=Y_2(r_l, e_l, s_l)} = 0.$$

According to the steps 6, 9, 12, this equation is equivalent to the following equation:

$$R_l = -Y_1(r_l, e_l) \cdot G_1(d)P - Y_2(r_l, e_l, s_l) \cdot G_2(d)P,$$

and  $R_l \cdot x \bmod q = r_l$  by construction at step 7. Hence, the signature  $(r_l, s_l)$  is valid for  $m_l$ .

The condition  $l \geq \lceil \log q \rceil$  is needed to make possible the field element binary representation (see step 8) of length  $l$ .

The attack works due to the ability of varying  $Y_1(r_i, e_i)$  values by message changing on step 4. This, in turn, is possible because of the summand, that does not depend on  $s$  value, in the equation (5). That explains the form of the Condition 1.

## 5.2 Attack on blindness

Consider GenEG-BS schemes of Type 2. Remind that for such schemes the Condition 2 is satisfied, i.e. the equation

$$F_1(r, e) \cdot F_2(r', e') = F_1(r', e') \cdot F_2(r, e) \quad (6)$$

holds with the overwhelming probability.

We claim that such schemes do not provide blindness. Namely, we show that for fixed protocol transcription and message there exists only the small set of valid signature values that could be produced during the given protocol execution. Indeed, if the protocol transcription  $(R, e, s)$  and message  $m$  are

fixed, then the  $r = R.x \bmod q$  and  $e' = H(m)$  values are also fixed. The equation (6) is affine by  $r'$  since  $F_1(r', e')$  and  $F_2(r', e')$  functions are affine by  $r'$  and only one of them significantly depends on  $r'$ . Thus,  $r'$  component of the signature is defined unambiguously from equation (6). Note that  $\alpha, \beta, \gamma$  are equal to zero or chosen uniformly at random from  $\mathbb{Z}_q^*$ . The probability to choose  $\alpha, \beta, \gamma$  during several protocol executions such that  $(\alpha R + \beta Q + \gamma P).x \bmod q = r'$  is negligible. Therefore, with overwhelming probability there exists the unique signature that could be produced for message  $m$  during the given protocol transcription.

### 5.3 Unforgeability attack

Suppose, that there exists **GenEG-BS** scheme of Type II, for which the Condition 2 does not hold. It means that there exists an algorithm **User**, that works on the Requester side as follows. For arbitrary public key  $pk$ , outputted by key generation algorithm, arbitrary message  $m$ , point  $R$  and  $\alpha, \beta, \gamma$  values, generated according to the distributions specified by the scheme, it outputs some value  $e$ . Then, after receiving the  $s$  value, generated according to (2) or (3), algorithm **User** outputs a valid signature  $(r', s')$  for message  $m$  with the overwhelming probability. Here the probability space consists of all values representing random choices made by the **User** randomized algorithm. Otherwise, it returns the fail indicator.

We construct an adversary  $\mathcal{A}$  for such **GenEG-BS** scheme that violates unforgeability and uses algorithm **User**. It can interact with the Signer in the way described in Section 4. The adversary  $\mathcal{A}$  knows the public key  $Q$  and acts as follows:

1. Selects message  $m$  and computes  $e' = H(m)$ .
2. Selects  $\alpha, \beta, \gamma$  values uniformly from  $\mathbb{Z}_q^*$  or defines them equal to zero (depending on the **User** algorithm).
3. Opens the session, querying the Signer, and receives point  $R$  as the response, computes  $r = R.x \bmod q$ .
4. Computes  $r' = (\alpha R + \beta Q + \gamma P).x \bmod q$ .
5. Runs algorithm **User**, giving it public key  $Q$ , point  $R$ , message  $m$  and  $\alpha, \beta, \gamma$  values.
6. Receives  $e$  value from the **User**.

7. If  $\gamma F_1(r, e) - \beta F_2(r, e) = 0$ , goes to the next step.

If  $\gamma F_1(r, e) - \beta F_2(r, e) \neq 0$ , computes

$$s^* = (\gamma F_1(r, e) - \beta F_2(r, e))^{-1} (F_1(r, e) F_2(r', e') - F_2(r, e) F_1(r', e'))$$

and checks if the signature is valid, computing  $b = \text{GenEG-BS.Vf}(Q, m, (r', s^*))$ . If  $b = 1$ , the adversary  $\mathcal{A}$  outputs  $(m, (r', s^*))$  pair as the forgery and stops.

8. Sends  $e$  value to the Signer and forwards the obtained  $s$  value to the User.

9. Receives the signature  $(r', s')$  from the User. This signature must be valid for message  $m$  under public key  $Q$ , thus  $s' \neq s^*$ . If User outputs the fail indicator, the adversary  $\mathcal{A}$  stops its work with the fail indicator.

10. If the equation (6) is not fulfilled, computes secret signing key  $d$  using the Algorithm 1 described below. After that, it computes valid signature  $(r'_1, s'_1)$  for arbitrary message  $m_1 \neq m$ , using the knowledge of  $d$ , and outputs two pairs  $(m, (r', s'))$  and  $(m_1, (r'_1, s'_1))$ . If the equation (6) holds true, the adversary  $\mathcal{A}$  stops its work with the fail indicator.

This attack is shown schematically in the Figure 3.

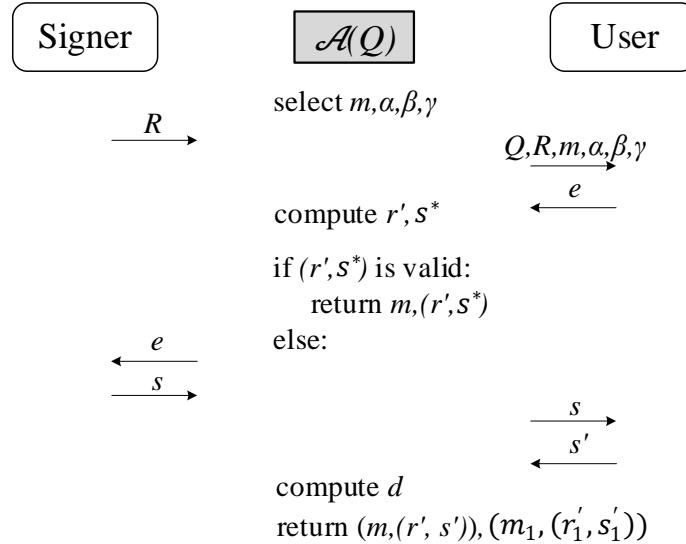


Figure 3: Attack on the GenEG-BS scheme of Type II

If the adversary  $\mathcal{A}$  finishes the work on step 7, it completes successfully 0 interactions with the Signer and outputs 1 forgery. Otherwise, the adversary

$\mathcal{A}$  makes 1 successful interaction with the Signer and outputs 2 forgeries, if **User** outputs a valid signature and the equation (6) holds true. According to the assumptions of Theorem 3, the probability of Condition 2 (and, thus, equation (6)) fulfillment and returning the fail indicator by **User** is negligible. Thus, the adversary  $\mathcal{A}$  violates unforgeability with the overwhelming probability.

**Algorithm 1.** We consider the case when **GenEG-BS** scheme of Type II is based on the equation (2), the case of the equation (3) is proved analogously.

Having a valid signature  $(r', s')$  for message  $m$  with hash-value  $e'$  and protocol transcription  $(R, e, s)$ , the adversary  $\mathcal{A}$  can construct the following system of linear equations with respect to unknown  $k$  and  $d$ :

$$\begin{cases} sk = F_1(r, e)d + F_2(r, e), \\ s'(\alpha k + \beta d + \gamma) = F_1(r', e')d + F_2(r', e'), \end{cases} \quad (7)$$

where  $r = R.x \bmod q$ . The first equation follows from the procedure of  $s$  value computation according to the equation (2). The second equation follows from the fact, that  $r' = R'.x \bmod q = (\alpha R + \beta Q + \gamma P).x \bmod q$  and the signature  $(r', s')$  is valid, i.e.  $s'R' = F_1(r', e')Q + F_2(r', e')P$ .

Due to the construction of the scheme the system (7) must have a solution relative to  $k$  and  $d$ . According to the Kronecker-Capelli theorem [22], a system has a solution iff the rank of its coefficient matrix  $A$  is equal to the rank of its augmented matrix  $A'$ . We write out these matrices for system (7):

$$A = \begin{pmatrix} s & -F_1(r, e) \\ s'\alpha & s'\beta - F_1(r', e') \end{pmatrix},$$

$$A' = \begin{pmatrix} s & -F_1(r, e) & F_2(r, e) \\ s'\alpha & s'\beta - F_1(r', e') & F_2(r', e') - s'\gamma \end{pmatrix}.$$

Further we show that  $\text{rank}(A) = \text{rank}(A') = 2$ . Then the solution of the system is unique, and  $\mathcal{A}$  finds secret key  $d$  by solving the system.

Suppose the opposite. Let  $\text{rank}(A) = \text{rank}(A') \leq 1$ . Then any two columns of matrix  $A'$ , in particular, second and third columns, are linearly dependent. This means that the determinant of the square submatrix formed by these columns is equal to zero. We write out this condition:

$$\begin{aligned} 0 &= \begin{vmatrix} -F_1(r, e) & F_2(r, e) \\ s'\beta - F_1(r', e') & F_2(r', e') - s'\gamma \end{vmatrix} = \\ &= F_1(r, e)(s'\gamma - F_2(r', e')) - (s'\beta - F_1(r', e'))F_2(r, e) = \\ &= s'(\gamma F_1(r, e) - \beta F_2(r, e)) - (F_1(r, e)F_2(r', e') - F_2(r, e)F_1(r', e')). \end{aligned}$$

Since the equation (6) is not fulfilled,  $F_1(r, e)F_2(r', e') - F_2(r, e)F_1(r', e') \neq 0$ . Then if  $\gamma F_1(r, e) - \beta F_2(r, e) = 0$ , the determinant can not be equal to zero and we come to the contradiction, from where  $\text{rank}(A) = \text{rank}(A') = 2$ . Let  $\gamma F_1(r, e) - \beta F_2(r, e) \neq 0$ , then we have the following condition on  $s'$ :

$$s' = (\gamma F_1(r, e) - \beta F_2(r, e))^{-1}(F_1(r, e)F_2(r', e') - F_2(r, e)F_1(r', e')) = s^*.$$

However,  $s' \neq s^*$  according to the adversary  $\mathcal{A}$  construction (see step 9), so we come to the contradiction and  $\text{rank}(A) = \text{rank}(A') = 2$ .

## 6 Conclusion

The obtained results show that the development of secure ElGamal blind signature scheme is non-trivial task. There exist no such schemes to date. If such a scheme potentially exists, then either its Signer side differs from the one defined in the **GenEG-BS** scheme, or the method of generating the first component of the signature on the Requester side is entirely new and signature equation necessarily has the form (2) or (3).

Therefore, the direction for further research is the analysis of more general blind signature constructions based on ElGamal signature equations and providing either the attacks on them, or their formal security proof.

## References

- [1] Abe M., Okamoto T., “Provably secure partially blind signatures”, *Advances in Cryptology – CRYPTO 2000*, Springer, Berlin, Heidelberg, 2000, 271–286.
- [2] Babueva, A. A., Akhmetzyanova, L. R., Alekseev, E. K., Taraskin, O. G., “On Blindness of Several ElGamal-Type Blind Signatures”, *Proceedings of the 6th International Conference "Convergent Cognitive Information Technologies"*, Convergent 2021. Communications in Computer and Information Science.
- [3] Benhamouda, F., Lepoint, T., Loss, J., Orru, M., Raykova, M., “On the (in) security of ROS”, *Advances in Cryptology – EUROCRYPT 2021*, Springer, Cham, 2021, 33–53.
- [4] Chaum D., “Blind signatures for untraceable payments”, *Advances in cryptology*, ed. Chaum D., Rivest R.L., Sherman A.T., Springer, Boston, MA, 1983, 199–203.
- [5] Camenisch, J. L., Piveteau, J. M., Stadler, M. A., “Blind signatures based on the discrete logarithm problem”, *LNCS, Advances in Cryptology – EUROCRYPT’94*, **950**, ed. De Santis A, Springer, Berlin, Heidelberg, 1994.
- [6] ElGamal T., “A public key cryptosystem and a signature scheme based on discrete logarithms”, *IEEE transactions on information theory*, **31:4** (1985), 469–472.
- [7] Fersch M., *The provable security of Elgamal-type signature schemes*, Diss. Bochum, Ruhr-Universität Bochum, 2018.
- [8] Fujioka A., Okamoto T., Ohta K., “A practical secret voting scheme for large scale elections”, *LNCS, Advances in Cryptology – AUSCRYPT ’92*, **718**, Springer, Berlin, Heidelberg, 1992.
- [9] Fuchsbauer G., Plouviez A., Seurin Y., “Blind Schnorr signatures and signed ElGamal encryption in the algebraic group model”, *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, Cham, 2020, 63–95.

- [10] Gorbenko I., Yesina M., Ponomar V., “Anonymous electronic signature method”, 2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T), 2016, 47–50.
- [11] Harn, L., Xu, Y., “Design of generalised ElGamal type digital signature schemes based on discrete logarithm”, *Electronics Letters*, **30**:24 (1994), 2025–2026.
- [12] Jena D., Panigrahy S. K., Acharya B., Jena S. K., “A Novel ECDLP-Based Blind Signature Scheme”, National Conference on Information Security – Issues & Challenges, NCISIC 08, 2008.
- [13] Khater, M. M., Al-Ahwal, A., Selim, M. M., Zayed, H. H., “New Blind Signature Scheme Based on Modified ElGamal Signature for Secure Electronic Voting”, *International Journal of Scientific & Engineering Research*, **9**:3 (2018).
- [14] Moldovyan, N. A. Blind Signature Protocols from Digital Signature Standards. In: *Int. J. Netw. Secur.*, 13(1), pp. 22–30. 2011.
- [15] Pointcheval D., Stern J., “Security arguments for digital signatures and blind signatures”, *Journal of cryptology*, **13**:3 (2000), 361–396.
- [16] Qin X., Cai C., Yuen T.H., “One-More Unforgeability of Blind ECDSA”, *LNCS*, Computer Security – ESORICS 2021, **12973**, ed. Bertino E., Shulman H., Waidner M., Springer, Cham, 2021.
- [17] Rostovtsev, A. G., “Blind signature on elliptic curve for e-cash.”, *Information Security Problems. Computer Systems*, **1** (2000), 40–45, In Russian.
- [18] Schnorr, C. P., “Security of blind discrete log signatures against interactive attacks”, *LNCS*, International Conference on Information and Communications Security, **2229**, ed. Qing S., Okamoto T., Zhou J., Springer, Berlin, Heidelberg, 2001.
- [19] Shen, V. R., Chung, Y. F., Chen, T. S., Lin, Y. A., “A blind signature based on discrete logarithm problem”, *International Journal of Innovative Computing, Information and Control*, **7**:9 (2011), 5403–5416.
- [20] Tan D. N., Nam H. N., Van H. N., Thi, L. T., Hieu M. N., “New blind mutisignature schemes based on signature standards”, 2017 International Conference on Advanced Computing and Applications (ACOMP), 2017, 23–27.
- [21] Tan, D. N., Nam, H. N., Hieu, M. N., Van, H. N., “New Blind Muti-signature Schemes based on ECDLP”, *International Journal of Electrical and Computer Engineering*, **8**:2 (2018), 1074–1083.
- [22] Vinberg E. B., *A course in algebra*, **56**, American Mathematical Soc, 2003.
- [23] Wagner D., “A generalized birthday problem”, *LNCS*, Advances in Cryptology – CRYPTO 2002, **2442**, Springer, Berlin, Heidelberg, 2002.
- [24] Yi, X., Lam, K. Y., “A new blind ECDSA scheme for bitcoin transaction anonymity”, *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, 2019, 613–620.
- [25] *GOST R 34.10-2012. Information technology. Cryptographic data security. Signature and verification processes of electronic digital signature. National standard of the Russian Federation, STANDARTINFORM*, 2012, In Russian.



# Solving some cryptanalytic problems for lattice-based cryptosystems with quantum annealing method

Ivan Lysakov

Lomonosov Moscow State University, Moscow, Russia  
lysakoviv@my.msu.ru

## Abstract

In this paper we study closest vector problem (CVP) and bounded distance decoding problem (BDD) which arise in cryptanalysis of lattice-based cryptosystems. We propose an algorithm for solving bounded distance decoding (BDD) problem using quantum annealing. We provide estimates for number of qubits required to run this algorithm. We also estimate number of qubits required for lattices that have Hermite normal form with a single pivot element not equal to 1, and lattices defined by the public keys of NTRUEncrypt cryptosystem.

**Keywords:** closest vector problem, bounded distance decoding, NTRUEncrypt, quantum annealing.

## 1 Introduction

It is widely known that in quantum computation model there exist algorithms for solving hard mathematical problems which provide a speedup compared to classical computational model. Algorithms for solving integer factorization and discrete logarithm problems are examples of those [18]. The security of most modern asymmetric cryptographic schemes (like RSA [17], El Gamal [5] and etc.) is based on the assumption of hardness of this two problems.

The model of quantum computations for a long time remained purely theoretical, but different prototypes of quantum computers have been created recently. The performance of these prototypes does not allow to attack existing cryptosystems, but many researchers believe that computers with enough computational power to do it may appear in the next few years. In this regard, synthesis of asymmetric schemes that are resistant to attacks with use of a quantum computer, or the so called post-quantum schemes, becomes very important.

There are different computational models based on the principles of quantum mechanics. In this paper we consider quantum annealing. It is believed that quantum annealer would be able to effectively solve a number of optimization problems over discrete sets.

It is assumed that the family of cryptographic schemes, which security is based on problems from the lattice theory, is resistant to attacks with the use of quantum computer. The closest vector problem (CVP) is often used as such problem. However, it is not always necessary to solve it for breaking the system. Sometimes it is enough to solve an easier problem - the problem of bounded distance decoding (BDD).

## 1.1 Related work

Quantum computer and quantum annealer are two different devices that work with their own computational models. Therefore below we represent results for both of them.

**Quantum gate model** SVP and CVP are assumed to be in some analogue of NP class for quantum gate model [2]. This makes it possible for us to consider them hard to solve on quantum computer.

The vast majority of quantum algorithms for solving SVP and CVP is based on Grover's algorithm for searching unsorted list [7] which offers polynomial speed-up. In this regard, it is not expected to gain any significant benefits from using a quantum computer.

Lattice enumeration is a classical method for solving SVP. It was proposed in the first half of 1980's in [6], [12] and [16]. This algorithm runs  $2^{O(n \log n)}$  in time and  $O(n)$  in space on classic computer, where  $n$  - lattice size [11]. Usage of Grover's algorithm hardly improves these estimates.

Many probabilistic algorithms are based on lattice sieving technique which was introduced in 2001 in [1]. These algorithms are known to be the fastest for solving SVP as they run  $2^{O(n)}$  in time. But at the same time, this comes with a cost of space, as sieving requires  $2^{O(n)}$  of memory [11]. Grover's algorithm can be applied to sieving algorithms. It helps to speed up running time, but not enough to get out of  $2^{O(n)}$  [11].

For solving CVP the Voronoi cell may be computed [13]. In both classic and quantum gate models the time complexity of it is equal to  $2^{2n+o(n)}$ , and it requires  $2^{n+o(n)}$  space [11].

**Quantum annealers** Two algorithms for solving SVP were introduced in [10]. In this paper, the authors reduced SVP to the Ising problem which is supposedly can be solved fast enough by quantum annealer [8]. Estimates on qubits number also were given. For the first algorithm to work with lattices, which has Hermite normal form with only one pivot element not equal to one, it requires  $O(n \log n + \log D)$  qubits, where  $n$  - lattice dimension and  $D$  - it's absolute value of determinant. At the same time, the second algorithm requires  $O(n^{5/2} D^{1/n})$  qubits for the same type of matrices, but it is much more resistant to quantum noise.

## 1.2 Our contribution

In this paper we research the possibilities of solving CVP and BDD with quantum annealers. Approach proposed by us is a generalization of shortest vector problem (SVP) solving algorithm in paper [10]. We present an algorithm that allows to solve these problems and estimate the required number of qubits. We also provide estimates for number of qubits required to make it work with matrices, used in NIST PQC candidates like NTRU.

## 2 Preliminaries

### 2.1 Lattices

**Definition 1.** Let  $\mathbf{b}_1, \dots, \mathbf{b}_n \in \mathbb{Z}^m$  be a set of lineary independent vectors. The set  $\Lambda = \{\sum_{i=1}^n z_i \mathbf{b}_i \mid z_i \in \mathbb{Z}\}$  is called a lattice. The matrix  $B$  composed of column vectors  $\mathbf{b}_i$  is called the lattice basis.

**Statement 1.** [3] The bases  $B_1$  and  $B_2$  specify the same lattice if and only if  $B_2 = U B_1$ , where  $U$  is an unimodular matrix.

From the previous statement, the following definition is correct:

**Definition 2.** For a given lattice  $\Lambda$  define its determinant as  $|\Lambda| = \sqrt{\det(B^T B)}$ .

Each lattice is an additive abelian subgroup of  $(\mathbb{R}^m, +)$ . Hence, it contains a vector with minimal Euclidean norm. The length (Euclidean norm) of such a vector for the lattice  $\Lambda$  is called the *first minimum* of the lattice.

$$\lambda_1(\Lambda) = \min\{\|v\| \mid v \in \Lambda, v \neq 0\}$$

**Statement 2.** (Minkowski's theorem) [9] For a lattice  $\Lambda$  of full rank and dimension  $N$ , it holds  $\lambda_1(\Lambda) \leq \sqrt{N} \cdot |\Lambda|^{1/N}$ .

**Definition 3.** A matrix  $H = \{h_{i,j}\}_{1 \leq i,j \leq n}$  is an Hermite normal form (HNF) of matrix  $B$  if there is a unimodular matrix  $U$  such that  $H = BU$  and  $H$  has the following restrictions:

1.  $H$  is upper triangular ( $h_{i,j} = 0$  for  $i > j$ ).
2. The pivot element of any nonzero row is always positive and is strictly to the right of the pivot element of the row above it.
3. elements below the pivot element are 0, and the elements above it are positive and strictly smaller than it.

If a matrix  $H$  has only one pivot element not equal to 1, then it is called the optimal Hermite normal form of matrix  $B$ .

We will say that the lattice  $\Lambda$  defined by the basis  $B$ , has an optimal HNF if for  $B^T$  there exists an optimal Hermite normal form.

**Statement 3.** [3] Every matrix  $B \in \mathbb{Z}^{n \times n}$  has HNF.

Next, let us introduce several difficult problems from lattice theory.

**Definition 4.** The Shortest Vector Problem (SVP): for a given lattice  $\Lambda$  find a vector  $\mathbf{v} \in \Lambda$  such that  $\|\mathbf{v}\| = \lambda_1(\Lambda)$ .

**Definition 5.** The Closest Vector Problem (CVP): For a given point  $\mathbf{t} \notin \Lambda$  find a vector  $\mathbf{v} \in \Lambda$  such that  $\|\mathbf{v} - \mathbf{t}\| \rightarrow \min$ .

**Definition 6.** Bounded Distance Decoding problem ( $BDD_\gamma$ ): For a given point  $\mathbf{t} \notin \Lambda$ , find the closest vector  $\mathbf{v} \in \Lambda$  provided that  $\|\mathbf{v} - \mathbf{t}\| \leq \gamma \cdot \lambda_1(\Lambda)$ .

The SVP problem can be reduced to the minimization problem of a positively defined quadratic form.

Denote by  $G = \{g_{ij}\}$  the Gram matrix for the lattice basis  $B$ . Then for any vector  $\mathbf{v} \in \Lambda$ :

$$\begin{aligned} \|\mathbf{v}\|^2 &= \|B\mathbf{x}\|^2 = \left\| \sum_{i=1}^n x_i \mathbf{b}_i \right\|^2 = \\ & \left( \sum_{i=1}^n x_i \mathbf{b}_i, \sum_{i=1}^n x_i \mathbf{b}_i \right) = \sum_{i=1}^n \sum_{j=1}^n (\mathbf{b}_i, \mathbf{b}_j) x_i x_j = \sum_{i,j=1}^n g_{ij} x_i x_j. \end{aligned}$$

Thus, to solve the SVP problem it is sufficient to minimize the following quadratic form on the set of all nonzero integer vectors.

$$f(\mathbf{x}) = \sum_{i,j=1}^n g_{ij} x_i x_j = \mathbf{x}^T G \mathbf{x} = \mathbf{x}^T B^T B \mathbf{x}.$$

Similarly, to solve CVP and BDD problems it is sufficient to find the minimum of the quadratic form presented below

$$g(\mathbf{x}) = \|B\mathbf{x} - \mathbf{t}\|^2 = \|B\mathbf{x}\|^2 - (B\mathbf{x}, \mathbf{t}) + \|\mathbf{t}\|^2 = \mathbf{x}^T B^T B \mathbf{x} - 2\mathbf{x}^T B^T \mathbf{t} + \mathbf{t}^T \mathbf{t}.$$

## 2.2 Ising and QUBO problems and quantum annealing

Quantum annealing is an optimization process based on quantum mechanics that is used for finding global minimum of a given objective function. The objective function is defined over a discrete space. Quantum annealer starts from a superposition of all possible states in the search space. Then the annealer's state evolves following the quantum-mechanical evolution. At the end of evolution, the state will correspond to a solution of the initial problem.

Consider an example of objective function:

$$E_{ising}(\mathbf{s}) = \sum_{p=1}^N h_p s_p + \sum_{i=1}^N \sum_{j=i+1}^N J_{i,j} s_i s_j,$$

$$h_p \in \mathbb{R}, p = \overline{1, N}, J_{i,j} \in \mathbb{R}, i = \overline{1, N}, i < j \leq N$$

$$s_k \in \{-1, 1\}, k = \overline{1, N}.$$

Annealer's qubits correspond to the values of variables  $s_k$ ,  $k = \overline{1, N}$ . The problem of minimizing  $E_{ising}$  is called *the Ising problem*.

With a change of variables in the function  $E_{ising}$ :  $s_k = 2x_k - 1$ ,  $k = \overline{1, N}$  up to a constant we get the following:

$$E_{QUBO}(\mathbf{x}) = \sum_{i=1}^N Q_{i,i} x_i + \sum_{i < j} Q_{i,j} x_i x_j, x_i \in \{0, 1\}, i = \overline{1, N},$$

$$Q_{i,j} \in \mathbb{R}, i, j = \overline{1, N}.$$

The problem of minimization of the function  $E_{QUBO}$  over the set  $\{0, 1\}^N$  is called *Quadratic unconstrained binary optimization problem (QUBO)*.

It is assumed that a quantum annealer would be able to solve the Ising and QUBO problems more efficiently than a classical computer.

## 3 Solving CVP and BDD with quantum annealing

In this section we describe an algorithm for solving the SVP problem with quantum annealing method proposed in [10]. Next we generalize this approach and propose an algorithm for solving CVP and BDD problems with quantum annealing.

### 3.1 SVP to QUBO reduction

Let the lattice  $\Lambda$  defined by the basis  $B = \{\mathbf{b}\}_{i=1}^N$ . and  $|\Lambda| = D$ . The lattice  $\Lambda$  corresponds to a positively determined quadratic form:

$$f(\mathbf{x}) = \sum_{i,j=1}^N g_{ij} x_i x_j, \quad g_{ij} = (\mathbf{b}_i, \mathbf{b}_j).$$

Then to solve the SVP problem it will be enough to find the vector on which  $f(\mathbf{x})$  reaches it's minimum among all nonzero integer vectors.

To reduce the problem of minimization of positively determined quadratic form to the QUBO problem it is necessary to replace the set of variables  $\mathbf{x} \in \mathbb{Z}^N$  to the set  $\mathbf{q} \in \{0, 1\}^L$  respectively.

In [10] the following method of reducing the functional  $f(\mathbf{x})$  to the functional used in the QUBO problem is presented.

Suppose that  $|x_j| \leq 2^k$ ,  $j = \overline{1, N}$ . Let us make next substitution of variables:

$$x_j = \sum_{p=0}^k 2^p q_{pj} - 2^k, \quad q_{pj} \in \{0, 1\}.$$

This will reduce the given quadratic form to the QUBO problem.

As shown in [10], if the matrix  $B$  is an optimal HNF, then the parameter  $k$  can be chosen as  $1 + \frac{3}{2} \log N + \frac{1}{N} \log D$ . So the following theorem holds:

**Theorem 1.** [10] *For a lattice  $\Lambda$  of dimension  $N$  with an optimal HNF and  $|\Lambda| = D$  there is a quantum algorithm for solving the SVP problem, that requires at most  $(\frac{3N}{2} \log N + N + \log D)$  qubits.*

### 3.2 CVP and BDD to QUBO reduction

For a given point  $\mathbf{t} \in \mathbb{Z}^n$ , that does not belong to the lattice  $\Lambda$ , in order to solve the BDD or CVP problem, it is enough to find  $\mathbf{z} \in \mathbb{Z}^N$  such that

$$\mathbf{z} = \underset{\mathbf{x} \in \mathbb{Z}^N}{\operatorname{argmin}} (\mathbf{x}^T B^T B \mathbf{x} - 2 \mathbf{x}^T B^T \mathbf{t} + \mathbf{t}^T \mathbf{t}) = \underset{\mathbf{x} \in \mathbb{Z}^N}{\operatorname{argmin}} (g(\mathbf{x})).$$

Similarly to the case of reducing SVP to QUBO, we will encode each vector coordinate of  $\mathbf{x}$  by its binary representation. Thus we replace the

variables:

$$\mathbf{x} = \underbrace{\begin{bmatrix} 1 & 2 & \dots & 2^k & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 1 & 2 & \dots & 2^k & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 1 & 2 & \dots & 2^k \end{bmatrix}}_{N \times (N \cdot (k+1))} \cdot \underbrace{\begin{bmatrix} x_{0,1} \\ \vdots \\ x_{k,1} \\ \vdots \\ x_{0,N} \\ \vdots \\ x_{k,N} \end{bmatrix}}_{(N \cdot (k+1)) \times 1} - \underbrace{\begin{bmatrix} 2^k \\ \vdots \\ 2^k \end{bmatrix}}_{N \times 1}.$$

Let us rewrite it in other terms:  $\mathbf{x} = T\mathbf{x}' - \mathbf{d}$ .

After the change of variables, the function  $g(\mathbf{x})$  will take the following form:

$$\begin{aligned} g(\mathbf{x}') &= \mathbf{x}' Q \mathbf{x}' + L \mathbf{x}' + c, \\ Q &= T^T B^T B T, \quad L = -2\mathbf{d}^T B^T B T - 2\mathbf{t}^T B T, \\ c &= \mathbf{d}^T B^T B \mathbf{d} + 2\mathbf{d}^T B^T \mathbf{t} + \mathbf{t}^T \mathbf{t}, \\ \mathbf{x}' &\in \{0, 1\}^{N \cdot (k+1)}. \end{aligned}$$

The obtained functional is a QUBO problem and can be fed to quantum annealer.

**Theorem 2.** *For a lattice  $\Lambda$  of dimension  $N$  with an optimal HNF and  $|\Lambda| = D$  there exists a quantum algorithm for solving  $BDD_\gamma$  problem that requires at most  $N \log(2\gamma N^{3/2} D^{1/N} + \|\mathbf{t}\|_{L_1})$  qubits.*

*Proof.* Let it is required to solve  $BDD_\gamma$  problem for the lattice  $\Lambda$  given by it's basis  $B$  in optimal HNF, and the point  $\mathbf{t}$ .

Since  $B^T$  is an optimal HNF, then

$$B^T = \begin{bmatrix} 1 & 0 & \dots & 0 & b_1 \\ 0 & 1 & \dots & 0 & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & b_{N-1} \\ 0 & 0 & \dots & 0 & D \end{bmatrix}.$$

Then for any vector  $\mathbf{x} \in \mathbb{Z}^N$ :

$$\mathbf{t} - B \mathbf{x} = [t_1 - x_1, \dots, t_{N-1} - x_{N-1}, t_N - (x_1 b_1 + \dots x_{N-1} b_{N-1} + x_N D)]^T.$$

If  $\mathbf{x}$  is a solution to the  $\text{BDD}_\gamma$ , then, using the Minkowski's theorem, we obtain  $\|\mathbf{t} - B\mathbf{x}\| \leq \gamma \sqrt{N} D^{1/N}$ . It follows that:

$$\begin{aligned} |t_i - x_i| &\leq \gamma \sqrt{N} D^{1/N}, \quad i = 1, \dots, N-1, \\ |t_N - (x_1 b_1 + \dots x_{N-1} b_{N-1} + x_N D)| &\leq \gamma \sqrt{N} D^{1/N}. \end{aligned}$$

Since  $\forall x, y \in \mathbb{R}$  is true  $|x - y| \geq |x| - |y|$ , let us rewrite these inequalities:

$$|x_i| \leq \gamma \sqrt{N} D^{1/N} + |t_i|, \quad i = 1, \dots, N-1, \quad (1)$$

$$|x_N D| \leq \gamma \sqrt{N} D^{1/N} + |t_N - (x_1 b_1 + \dots x_{N-1} b_{N-1})|. \quad (2)$$

With the triangle inequality it follows:

$$|x_N D| \leq \gamma \sqrt{N} D^{1/N} + |t_N| + \sum_{i=1}^{N-1} |x_i b_i|.$$

Since  $B$  is an optimal HNF, then  $b_i < D$  for  $i = 1, \dots, N-1$ . Therefore

$$|x_N D| \leq \gamma \sqrt{N} D^{1/N} + D \sum_{i=1}^{N-1} |x_i| + D t_N.$$

Using the inequalities (1) we obtain:

$$\begin{aligned} |x_N D| &\leq \gamma \sqrt{N} D^{1/N} + \gamma (N-1) \sqrt{N} D^{1/N+1} + D \sum_{i=1}^N |t_i| \leq \\ &\gamma N^{3/2} D^{1/N+1} + D \|t\|_{L_1}. \end{aligned}$$

Thus, the values of the last coordinate are in the interval from  $-\gamma N^{3/2} D^{1/N} - \|t\|_{L_1}$  to  $\gamma N^{3/2} D^{1/N} + \|t\|_{L_1}$ . Therefore, its binary representation will require at most  $\log(2 \gamma N^{3/2} D^{1/N} + \|t\|_{L_1})$  qubits.  $\square$

## 4 NTRUEncrypt

The NTRUEncrypt [15] cryptosystem is one of the candidates, participating NIST PQC [14] effort. The main goal of NIST PQC is to select asymmetric cryptographic algorithms that are resistant to attacks that use quantum computers. This section explores the capabilities of the algorithm presented above when dealing with lattices, providing security for this scheme.



Next, we will consider lattices of dimension  $2N \times 2N$  of the following kind:

$$L = \left[ \begin{array}{cccc|cccc} 1 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & 0 & 0 & \dots & 0 \\ \hline h_0 & h_1 & \dots & h_{N-1} & q & 0 & \dots & 0 \\ h_{N-1} & h_0 & \dots & h_{N-2} & 0 & q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ h_1 & h_2 & \dots & h_0 & 0 & 0 & \dots & q \end{array} \right]$$

Here  $N$  and  $q$  are defined by the cryptosystem parameters, and the coefficients  $h_i$ ,  $i = 0, \dots, N - 1$  are defined by its public key.

Let us estimate how many qubits are needed to solve  $\text{BDD}_\gamma$  for this lattice. For any vector  $\mathbf{x} \in \mathbb{Z}^N$  and a point  $\mathbf{t} \notin L$  it holds:

$$\mathbf{t} - L\mathbf{x} = [t_0 - x_0, \dots, t_{N-1} - x_{N-1}, t_N - (x_0 h_0 + \dots x_{N-1} h_{N-1} + x_N q), \dots, t_{2N-1} - (x_0 h_1 + \dots x_{N-1} h_0 + x_{2N-1} q)]^T.$$

If  $\mathbf{x}$  is a solution of  $\text{BDD}_\gamma$ , then, using the Minkowski theorem, we obtain  $\|\mathbf{t} - L\mathbf{x}\| \leq \gamma \sqrt{qN}$ . From this it follows that

$$\begin{aligned} |t_i - x_i| &\leq \gamma \sqrt{2qN}, \quad i = 0, \dots, N - 1, \\ |t_N - (x_0 h_0 + \dots x_{N-1} h_{N-1} + x_N q)| &\leq \gamma \sqrt{2qN}, \\ &\dots \\ |t_{2N-1} - (x_0 h_1 + \dots x_{N-1} h_0 + x_{2N-1} q)| &\leq \gamma \sqrt{2qN}. \end{aligned}$$

Since  $\forall x, y \in \mathbb{R}$  it holds  $|x - y| \geq |x| - |y|$ , let us rewrite these inequalities:

$$\begin{aligned} |x_i| &\leq \gamma \sqrt{2qN} + |t_i|, \quad i = 0, \dots, N - 1, \\ |x_N q| &\leq \gamma \sqrt{2qN} + |t_N - (x_0 h_0 + \dots x_{N-1} h_{N-1})|, \\ &\dots \\ |x_{2N-1} q| &\leq \gamma \sqrt{2qN} + |t_{2N-1} - (x_0 h_1 + \dots x_{N-1} h_0)|. \end{aligned}$$

Let's transform the inequality for  $|x_N q|$  using the triangle inequality:

$$|x_N q| \leq \gamma \sqrt{2qN} + |t_N| + \sum_{i=0}^{N-1} |x_i h_i|.$$

From the key generation procedure [15] it follows that  $|h_i| \leq q$ ,  $i = 0, \dots, N - 1$ , then

$$|x_N q| \leq \gamma \sqrt{2qN} + q \sum_{i=0}^{N-1} |x_i| + q t_N.$$

Using the inequalities  $|x_i| \leq \gamma \sqrt{qN} + |t_i|$ ,  $i = 0, \dots, N - 1$  we get:

$$|x_N q| \leq \gamma \sqrt{2qN} + \gamma \sqrt{2} N^{3/2} q^{3/2} + q \sum_{i=0}^{2N-1} |t_i| \leq \gamma \sqrt{2} N^{3/2} q^{3/2} + q \|t\|_{L_1}.$$

Thus, the coordinate values with the index  $N$  are in the interval from  $-\gamma \sqrt{2} N^{3/2} \sqrt{q} - \|t\|_{L_1}$  to  $\gamma \sqrt{2} N^{3/2} \sqrt{q} + \|t\|_{L_1}$ . Therefore, to its binary representation will require  $\log(2 \gamma \sqrt{2} N^{3/2} \sqrt{q} + \|t\|_{L_1})$  qubits.

With similar reasoning for coordinates with an index from  $N$  to  $2N - 1$  the algorithm will require  $2N \log(2 \gamma \sqrt{2} N^{3/2} \sqrt{q} + \|t\|_{L_1})$  qubits.

As a point  $\mathbf{t}$  let's take one of the possible ciphertexts in the NTRUEncrypt cryptosystem. Then, according to the encryption procedure in [15], we'll get that  $\|t\|_{L_1} \leq Nq$ . As a result, we obtain the following estimation on the number of qubits:

$$2N \log(2 \gamma \sqrt{2} N^{3/2} \sqrt{q} + Nq).$$

To achieve a 128-bit security level, the cryptosystem designers suggested to choose  $N = 509$  and  $q = 2048$ . This means that to solve the  $BDD_\gamma$  problem in such lattice with  $\gamma = 1$  it is required about 21,000 qubits.

As proposed algorithm is intended to run on quantum annealer it is quite difficult to compare it with algorithms in quantum gate model. But it is possible to analyze result taking into account possibilities of existing today prototypes of quantum annealers.

D-Wave [4] is a company constructing computers which exploit quantum effects in their operation. It is claimed that this machines are using quantum annealing to solve optimization problem.

In 2020 D-Wave Advantage was introduced. This machine contains more than 5000 qubits. Company also announced D-Wave Advantage 2 with approximately 7000 qubits. This numbers means that appearance of computer which will have enough space to run suggested algorithm is the matter of near future.

## 5 Discussion

The paper gives an estimate on the number of qubits required to solve problems from lattice theory in a theoretical model of quantum computation. But in practical implementations quantum noises will play a significant role.

This means that a real quantum computer might need a lot more qubits than the estimate suggests.

We should also note that in this paper we do not provide any estimates for the running time of the algorithm. This problem requires additional study.

## Acknowledgments

This paper contains part of the results from the author's master thesis. I am grateful to V. Rudskoy and E. Primenko for helpful comments and discussions.

## References

- [1] Ajtai M., Kumar R., Sivakumar D., "A sieve algorithm for the shortest lattice vector problem", STOC, 601-610.
- [2] Aharonov D., Regev O., "A Lattice Problem in Quantum NP", 2003.
- [3] Cohen H., *A Course in Computational Algebraic Number Theory.*, Springer Publishing Company, Incorporated, 2010.
- [4] D-Wave, <https://www.dwavesys.com>.
- [5] Elgamal T., "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Trans. Inf. Theory, **4** (1985), 469–472.
- [6] Fincke, U., Pohst, M., "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis", Math. Comp. **44**, 1985, 463-471.
- [7] Grover L. K., "A fast quantum mechanical algorithm for database search", 28th Annual ACM Symposium on the Theory of Computing, 1996, 212–219.
- [8] Kadowaki T., Nishimori H., "Quantum annealing in the transverse Ising model", Physical Review E - Statistical Physics, **58**, 5 (1998), 5355–5363.
- [9] Minkowski H., *Geometrie der Zahlen.*, 1910.
- [10] Joseph D., Callison A., Ling C., Mintert F., "Two quantum Ising algorithms for the shortest-vector problem", Physical Review A, **103**, 3 (2021).
- [11] Laarhoven T., Mosca M., J. van de Pol, "Solving the Shortest Vector Problem in Lattices Faster Using Quantum Search", CoRR, **abs/1301.6176** (2013).
- [12] Kannan R., "Improved algorithms for integer programming and related lattice problems.", STOC, 1983, 193-206.
- [13] Micciancio, D., Voulgaris, P., "A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations", STOC, 2010, 351–35.
- [14] NIST PQC, <https://csrc.nist.gov/Projects/post-quantum-cryptography>.
- [15] NTRU NIST PQC submission 2020, <https://ntru.org/release/NIST-PQSubmission-NTRU-20201016.tar.gz>.
- [16] Post M., "On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications", ACM SIGSAM Bulletin, **15**, 1 (1981), 37–44.
- [17] Rivest R. L., Shamir A., Adleman L., "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, **21**, 2 (1976), 120-126.
- [18] Shor P. W., "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer", SIAM review, **41**, 3 (1999), 303-322.