ISSN 1990-4789, Journal of Applied and Industrial Mathematics, 2020, Vol. 14, No. 4, pp. 623–647. © Pleiades Publishing, Ltd., 2020.
Russian Text © The Authors, 2020, published in Diskretnyi Analiz i Issledovanie Operatsii, 2020, Vol. 27, No. 4, pp. 21–57.

On the Sixth International Olympiad in Cryptography NSUCRYPTO

A. A. Gorodilova[1*], N. N. Tokareva[1,2*], S. V. Agievich[3*], C. Carlet[4*], E. V. Gorkunov[1,5*], V. A. Idrisova[1*], N. A. Kolomeec[1*], A. V. Kutsenko[1,5*], R. K. Lebedev[5*], S. Nikova[6*], A. K. Oblaukhov[1*], I. A. Pankratova[7*], M. A. Pudovkina[8*], V. Rijmen[6*], and A. N. Udovenko[9*]

[1] *Sobolev Institute of Mathematics, pr. Akad. Koptyuga 4, Novosibirsk, 630090 Russia*
[2] *Laboratory of Cryptography JetBrains Research, ul. Pirogova 1, Novosibirsk, 630090 Russia*
[3] *Belarusian State University, pr. Nezavisimosti 4, Minsk, 220030 Belarus*
[4] *University of Paris 8, 2 Rue de la Liberté, 93200 Saint-Denis, France*
[5] *Novosibirsk State University, ul. Pirogova 2, Novosibirsk, 630090 Russia*
[6] *ESAT-COSIC, KU Leuven, Kasteelpark Arenberg, 10, B-3001 Leuven, Belgium*
[7] *Tomsk State University, pr. Lenina 36, Tomsk, 634050 Russia*
[8] *Bauman Moscow State Technical University, ul. Vtoraya Baumanskaya 5/2, Moscow, 105005 Russia*
[9] *SnT, University of Luxembourg, 2 Avenue de l'Universite, L-4365 Esch-sur-Alzette, Luxembourg*

Received May 20, 2020; in final form, August 18, 2020; accepted August 21, 2020

Abstract—NSUCRYPTO is the unique cryptographic Olympiad containing scientific mathematical problems for professionals, school and university students from any country. 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. The Olympiad history starts in 2014. In 2019, it was held for the sixth time. We present the problems and their solutions of the Sixth International Olympiad in cryptography NSUCRYPTO'2019. Under consideration are the problems related to attacks on ciphers and hash functions, protocols, Boolean functions, Dickson polynomials, prime numbers, rotor machines, etc. We discuss several open problems on mathematical countermeasures to side-channel attacks, APN involutions, S-boxes, etc. The problem of finding a collision for the hash function `Curl27` was partially solved during the Olympiad.

DOI: 10.1134/S1990478920040031

Keywords: *cryptography, cipher, hash function, Hamming code, slide attack, threshold implementation, Dickson polynomial, APN function, Olympiad, NSUCRYPTO*

INTRODUCTION

NSUCRYPTO (Non-Stop University Crypto) is the International Olympiad in cryptography that was held for the sixth time in 2019.

Interest in the Olympiad around the world is significant. This year, there were hundreds of participants from 26 countries; 42 participants in the first round and 21 teams in the second round from 16 countries were awarded with prizes and honorable diplomas. The Olympiad Program Committee includes specialists from Belgium, France, the Netherlands, the USA, Norway, India, Luxembourg, Belarus', Kazakhstan, and Russia.

Let us shortly formulate the format of the Olympiad. One of the Olympiad main ideas is that everyone can participate! Each participant chooses his/her category when registering on the Olympiad website [1]. There are three categories: "*school students*" (for junior researchers: pupils and high

* E-mail: nsucrypto@nsu.ru

school students), "*university students*" (for participants who are currently studying at universities) and "*professionals*" (for participants who have already completed education or just want to be in the restriction-free category). Awarding of the winners is held in each category separately.

The Olympiad consists of the two independent Internet rounds: the first one is individual (duration 4 hours 30 minutes) while the second round is a team one (duration 1 week). The first round is divided into two sections: A—for "school students," B—for "university students" and "professionals." The second round is common to all participants. Participants read the Olympiad problems and submit their solutions through the Olympiad website. The language of the Olympiad is English.

The Olympiad participants are always interested in solving various problems of any complexity at the intersection of mathematics and cryptography. The participants show their knowledge, creativity, and professionalism. That is why the Olympiad not only includes interesting tasks with known solutions but also offers unsolved problems. This year, one of such open problems, "`Curl27`" (see Section 2.14), was partially solved during the second round! All open problems stated during the Olympiad history can be found in [2].

On the website we also mark the current status of each problem. For example, in addition to "`Curl27`", the problem "Sylvester matrices" was solved by three teams in 2018, and the problem "Algebraic immunity" was completely solved during the Olympiad in 2016. And what is important for us, some participants were trying to find solutions after the Olympiad was over. For example, a partial solution for the problem "A secret sharing" (2014) was proposed in [3]. We invite everybody who has ideas on solving the problems to send solutions to us!

The paper is organized as follows: We start with the problem structure of the Olympiad in Section 1. Then we present formulations of all problems stated during the Olympiad and give their detailed solutions in Section 2. Finally, we publish the lists of NSUCRYPTO'2019 winners in Section 3.

Mathematical problems and their solutions of the previous International Olympiads in cryptography NSUCRYPTO from 2014 to 2018 can be found in [4], [5], [6], [7], and [8] respectively.

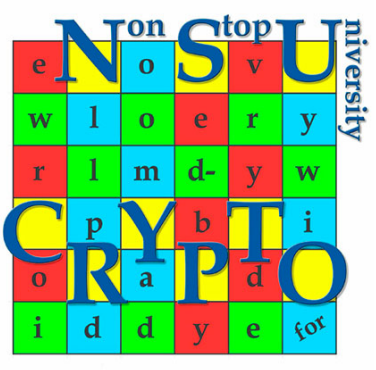

Fig. 1. NSUCRYPTO logo.

1. PROBLEM STRUCTURE OF THE OLYMPIAD

There were 16 problems stated during the Olympiad; some of them were included in both rounds (Tables 1 and 2). Section A of the first round consisted of six problems, whereas the section B contained seven problems. Three problems were common for both sections. The second round was composed of eleven problems. Five problems of the second round included unsolved questions (with special awards of the Program Committee).

2. PROBLEMS AND THEIR SOLUTIONS

In this section, we formulate all problems of NSUCRYPTO'2019 and present their detailed solutions paying attention to the solutions by the participants.

2.1. Problem "A 1024-Bit Key"

2.1.1. Formulation. Alice has a 1024-bit key for a symmetric cipher (the key consists of 0s and 1s). Alice is afraid of malefactors, so she changes her key everyday in the following way:

1. Alice chooses a subsequence of key bits such that the first bit and the last bit are equal to 0. She also can choose a subsequence of length 1 that contains only 0.

2. Alice inverts all bits in this subsequence (0 turns into 1 and vice versa); bits outside of this subsequence remain as they are.

Prove that the process will stop. Find the key that will be obtained by Alice in the end of the process.

Example of an operation. 11001 01101110 011... turns to 11001 10010001 011...

2.1.2. Solution. Let us encode the binary vector of the key as the corresponding decimal number. It is obvious that this number will increase on the next day since all bits on the left from the sequence are not changing, but the first bit of the sequence turns from 0 to 1. Let us note that this number can not increase infinitely since the size of the key is restricted by 1024 bits, so, in the very end the key will be maximal possible and, thus, will consist of all 1s.

Almost all participants successfully solved the problem.

2.2. Problem "The Magnetic Storm"

2.2.1. Formulation. A hardware random number generator is a device that generates random sequences consisting of 0s and 1s. Unfortunately, a disturbance caused by a magnetic storm affected this random

Table 1. Problems of the first round

N	Problem title	Maximum score
1	A 1024-bit key	4
2	The magnetic storm	4
3	Autumn leaves	4
4	A rotor machine	4
5	Broken `Calculator`	4
6	A promise	6

Section A

N	Problem title	Maximum score
1	Autumn leaves	4
2	The magnetic storm	4
3	A rotor machine	4
4	16QAM	8
5	A promise and money	6
6	`Calculator`	6
7	APN + Involutions	7

Section B

Table 2. Problems of the second round

N	Problem title	Maximum score
1	A 1024-bit key	4
2	Sharing	6 + additional scores for open questions
3	Factoring in 2019	8
4	`TwinPeaks-3`	8
5	`Curl27`	10 + additional scores for open questions
6	8-bit S-box	Unlimited (open problem)
7	A rotor machine	4
8	16QAM	8
9	`Calculator`	6
10	APN + Involutions (extended)	12 + additional scores for open questions
11	Conjecture	Unlimited (open problem)

Fig. 2. Autumn Leaves.

number generator. As a result, the device had generated a sequence of 0s of length k (where k is a positive integer), and then started to generate an infinite sequence of 1s.

Prove that at some point the generator will produce the number $1\ldots10\ldots0$ that is divisible by 2019.

2.2.2. Solution. Let us prove that a number of form $1\ldots11\ldots1$ is divisible by 2019. Consider all numbers that consists only of 1s. Since there are infinitely many of these numbers, there can be found a pair of numbers A and B such that they have the same remainder when divided by 2019. Therefore, $C = A - B = 1\ldots10\ldots0$ consisting of m 1s for some natural m is divisible by 2019, and, since 2019 is not divisible by 2 and 5,

$$C^* = C \times 10\ldots0 = 1\ldots10\ldots0$$

is divisible by 2019 for any number of 0s.

There were many correct solutions by the participants.

2.3. Problem "Autumn Leaves"

2.3.1. Formulation. Read a hidden message (see Fig. 2)!

2.3.2. Solution. We see different leaves and spaces between them. It looks like a simple substitution cipher was used there and distinct leaves corresponded to distinct English letters. By English grammar, we can suppose that the second and the third words are "`is a`." Then the first word starts with "`a`" and by its structure can be "`autumn`" (which is very likely as the autumn landscape is depicted). Also, the leaf  is the most common letter in the text and we can guess that it is "`e`." Then we see "`*ea*`" in the third line that seems to be "`leaf`". As a result the last word becomes "`fl**e*`" that is "`flower`." Finally, we get "`Autumn is a second spring when every leaf is a flower`" that is a famous quote by Albert Camus. Almost all participants read the message.

2.4. Problem "A Rotor Machine"

2.4.1. Formulation. In a country rotor machines were very useful for encryption of information (see examples in Fig. 3).

Eve knows that for some secret communication a simple rotor machine was used. It works with letters `O, P, R, S, T, Y` only and has an input circle with lamps (start), one rotor, and a reflector. See Fig. 4.

Fig. 3. Examples of rotor machines.

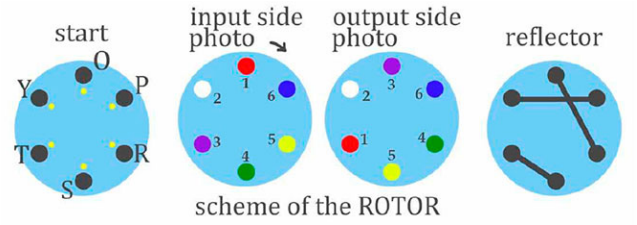

Fig. 4. Scheme of the rotor.

The input circle and the reflector are fixed in their positions, while the rotor can be in one of the six possible positions. After pressing a button on a keyboard, an electrical signal corresponding to the letter goes through the machine, comes back to the input circle, and the appropriate lamp shows the result of encryption. After each letter is encrypted, the rotor turns right (i.e. clockwise) on 60 degrees. Points of different colors (enumerated) on the rotor sides indicate different noncrossing signal lines within the rotor.

For instance, if the rotor is fixed as shown on the picture above then if you press the button `O`, it will be encrypted as `T` (the signal enters the rotor via red (color 1) point, is reflected, and then comes back via purple (color 5) line). If you press `O` again, it will be encrypted as `R`. If you press `T` then, you will get `S`, and so on.

Eve intercepted the secret message

TRRYSSPRYRYROYTOPTOPTSPSPRS.

Help her to decrypt it keeping in mind that Eve does not know the initial position of the rotor.

2.4.2. Solution. To solve the problem and decrypt the message, we need to correctly understand the scheme of work. A key for the cipher is the initial position of the rotor. We denote it by a color of the circle (enumerated) on the input side of the rotor that corresponds to the letter `O`. Table 3 represents the encryption tables depending on the key.

Table 3. Encryption tables

	O	P	R	S	T	Y
red (color 1)	T	Y	S	R	O	P
white (color 2)	R	S	O	P	Y	T
purple (color 3)	Y	R	P	T	S	O

	O	P	R	S	T	Y
green (color 4)	S	R	P	O	Y	T
yellow (color 5)	S	T	Y	O	P	R
blue (color 6)	R	T	O	Y	P	S

Trying all six possible keys, we find the only one meaningful message `POST TO TOP OOPS SORRY STOP ROTOR` that corresponds to the "yellow" (color 5) key.

Almost all participants solved the problem. The most interesting solutions were obtained by creating real models for this rotor machine, for example, by a school student Varvara Lebedinskaya (The Specialized Educational Scientific Center of Novosibirsk State University), by the team of Kristina Geut, Sergey Titov, and Dmitry Ananichev (Ural State University of Railway Transport).

2.5. Problem "Broken `Calculator`*"*

2.5.1. Formulation. Alice and Bob are practicing in developing toy cryptographic applications for smartphones. This year they have invented `Calculator` that allows one to perform the following operations modulo 2019 (that is to get the result as the reminder of division by 2019):

- to insert at most 4-digit positive integers (digits from 0 to 9);
- to perform addition, subtraction, and multiplication of two numbers;
- to store temporary results and read them from the memory.

Suppose that Alice wants to send Bob a ciphertext y (given by a 4-digit integer). She sends y from her smartphone to Bob's `Calculator` memory. To decrypt y, Bob needs to get the plaintext x (using his `Calculator`) by the rule: x is equal to the remainder of dividing $f(y) = y^5 + 1909y^3 + 401y$ by 2019.

At the most inopportune moment, Bob dropped his smartphone and broke its screen (see Fig. 5). Now, the button "+" as well as all digits except "1" and "5" are not working.

Help Bob to invent an efficient algorithm of how to decrypt any ciphertext y using `Calculator` in his situation. More precisely, suggest a short list of commands such that each command has one of the following types ($1 \leqslant j, k < i$):

$$S_i = y, \qquad S_i = a, \qquad S_i = S_j - S_k, \qquad S_i = S_j * S_k,$$

where a is an at most 4-digit integer consisting of digits 1 and 5 only; for example, $a = 1$, $a = 15$, $a = 551$, $a = 5115$, etc.

The first command has to be $S_1 = y$. In the last command, the resulting plaintext x has to be calculated. We remind that all calculations are modulo 2019. In particular, the integer 2500 becomes 481 and -1000 becomes 1019 immediately after entering or calculations. The shorter the list of commands you suggest, the more scores you get for this problem.

Example. The following list of commands calculates $x = y^2 - 55$:

Command	Result
$S_1 = y$	y
$S_2 = S_1 * S_1$	y^2
$S_3 = 11$	11
$S_4 = 5$	5
$S_5 = S_3 * S_4$	55
$S_6 = S_2 - S_5$	$y^2 - 55$

Fig. 5. Broken `Calculator`.

2.5.2. Solution. Let us present the original solution in 14 steps by the Program Committee.

Let $a \equiv_m b$ mean that integers a and b are congruent modulo m. The following relations hold:

$$\begin{aligned} f(y) &\equiv_{2019} y^5 + 1909y^3 + 401y \equiv_{2019} y(y^4 - 110y^2 + 401) \\ &\equiv_{2019} y(y^4 - 2 * 55y^2 + 55^2 - 55^2 + 401) \equiv_{2019} y((y^2 - 55)^2 - 55^2 + 5 * 22^2) \\ &\equiv_{2019} y((y^2 - 55)^2 - 11^2 * (5^2 - 5 * 2^2)) \equiv_{2019} y((y^2 - 55)^2 - 11^2 * 5) \\ &\equiv_{2019} y((y^2 - 55)^2 - 11 * 55). \end{aligned}$$

Thus, the reminder of division of $f(y)$ by 2019 can be calculated for any y by the list of commands in Table 4. A similar solution was found by Borislav Kirilov (Bulgaria, The First Private Mathematical Gymnasium).

Note. The polynomial $f(y) = y^5 + 1909y^3 + 401y$ is the Dickson polynomial $D_5(y, a) = y^5 - 5y^3a + 5ya^2$ for $a = 22$ with coefficients taken modulo 2019.

Table 4. List of commands for the Program Committee solution

Command	Result	Command	Result	Command	Result
$S_1 = y$	y	$S_4 = S_2 - S_3$	$y^2 - 55$	$S_7 = S_3 * S_6$	$11 * 55$
$S_2 = S_1 * S_1$	y^2	$S_5 = S_4 * S_4$	$(y^2 - 55)^2$	$S_8 = S_5 - S_7$	$(y^2 - 55)^2 - 11 * 55$
$S_3 = 55$	55	$S_6 = 11$	11	$S_9 = S_1 * S_8$	$y((y^2 - 55)^2 - 11 * 55)$

2.6. Problem "`Calculator`*"*

2.6.1. Formulation. Alice and Bob are practicing in developing toy cryptographic applications for smartphones. This year they have invented `Calculator` that allows one to perform the following operations modulo 2019:

- to insert at most 4-digit positive integers (digits from 0 to 9);
- to perform addition, subtraction, and multiplication of two numbers;
- to store temporary results and read them from the memory.

Suppose that Alice wants to send Bob a ciphertext y (given by a 4-digit integer). She sends y from her smartphone to Bob's `Calculator` memory. To decrypt y, Bob needs to get the plaintext x (using his `Calculator`) by the rule $x = f(y) \bmod 2019$, where f is a secret polynomial known to Alice and Bob only.

At the most inopportune moment, Bob dropped his smartphone and broke its screen (see Fig. 6). Now, the button "+" as well as all digits except "2" are not working.

Help Bob to invent an efficient algorithm of how to decrypt any ciphertext y using `Calculator` in his situation if the current secret polynomial is $f(y) = y^5 + 1909y^3 + 401y$. More precisely, suggest a short list of commands, where each command has one of the following types ($1 \leqslant j, k < i$):

$$S_i = y, \quad S_i = 2, \quad S_i = 222, \quad S_i = S_j - S_k, \quad S_i = 22, \quad S_i = 2222, \quad S_i = S_j * S_k.$$

The first command has to be $S_1 = y$. In the last command, the resulted plaintext x has to be calculated. We remind that all calculations are modulo 2019. In particular, the integer 2222 becomes 203 immediately after entering. The shorter the list of commands you suggest, the more scores you get for this problem.

Example. The following list of commands calculates $x = y^2 - 4$:

Command	Result
$S_1 = y$	y
$S_2 = S_1 * S_1$	y^2
$S_3 = 2$	2
$S_4 = S_3 * S_3$	4
$S_5 = S_2 - S_4$	$y^2 - 4$

Fig. 6. Broken `Calculator`.

2.6.2. Solution. The polynomial $f(y) = y^5 + 1909y^3 + 401y$ is the Dickson polynomial $D_5(y, a) = y^5 - 5y^3a + 5ya^2$ for $a = 22$ with coefficients taken modulo 2019. The following relations hold:

$$\begin{aligned} D_5(y, a) &= yD_4(y, a) - aD_3(y, a) = yD_2(D_2(y, a), a^2) - aD_3(y, a) \\ &= y((y^2 - 2a)^2 - 2a^2) - ay(y^2 - 2a - a). \end{aligned}$$

For $a = 22$, the value $f(y)$ can be calculated for any y by the list of commands given in Table 5.

What was surprising that the participants found two solutions that has 11 and 13 steps! These solutions were awarded by additional points. The solution with 11 steps were found by Madalina Bolboceanu (Romania, Bitdefender) during the first round (Table 6). The solution with 13 steps were given by Henning Seidler and Katja Stumpp team (Germany, TU Berlin) during the second round. Both solutions were based on the representation $f(y) = y((y^2 - 44)(y^2 - 66) - 22^2)$.

Table 5. List of commands for the Program Committee solution

Command	Result	Command	Result
$S_1 = y$	y	$S_8 = S_7 * S_7$	$(y^2 - 2a)^2$
$S_2 = 2$	2	$S_9 = S_8 - S_5$	$(y^2 - 2a)^2 - 2a^2$
$S_3 = 22$	a	$S_{10} = S_1 * S_9$	$y((y^2 - 2a)^2 - 2a^2)$
$S_4 = S_2 * S_3$	$2a$	$S_{11} = S_7 - S_2$	$y^2 - 2a - a$
$S_5 = S_3 * S_4$	$2a^2$	$S_{12} = S_1 * S_{11}$	$y(y^2 - 2a - a)$
$S_6 = S_1 * S_1$	y^2	$S_{13} = S_3 * S_{12}$	$ay(y^2 - 2a - a)$
$S_7 = S_6 - S_4$	$y^2 - 2a$	$S_{14} = S_{10} - S_{13}$	$f(y)$

Table 6. List of commands for the 11-step solution

Command	Result	Command	Result
$S_1 = y$	y	$S_7 = S_6 - S_4$	$y^2 - 44 - 22$
$S_2 = S_1 * S_1$	y^2	$S_8 = S_6 * S_7$	$(y^2 - 44) * (y^2 - 44 - 22)$
$S_3 = 2$	2	$S_9 = S_4 * S_4$	22^2
$S_4 = 22$	22	$S_{10} = S_8 - S_9$	$(y^2 - 44) * (y^2 - 44 - 22) - 22^2$
$S_5 = S_3 * S_4$	44	$S_{11} = S_1 * S_{10}$	$f(y)$
$S_6 = S_2 - S_5$	$y^2 - 44$		

2.7. Problem "A Promise"

2.7.1. Formulation. Young cryptographers, Alice, Bob and Carol, are interested in quantum computings and really want to buy a quantum computer. A millionaire gave them some certain amount of money (say, X_A for Alice, X_B for Bob, and X_C for Carol). He also made them promise that they would not tell anyone including each other, how much money everyone of them had received.

• Could you help the cryptographers to invent an algorithm of how to find out (without breaking the promise) whether the total amount of money they have, $X_A + X_B + X_C$, is enough to buy a quantum computer?

• What weaknesses does your algorithm have (if someone breaks the promise)? Does it always protect the secret of the honest participants from the dishonest ones?

2.7.2. Solution. This problem is a particular case for the problem "A promise and money" for only three participants (see Section 2.8).

2.8. Problem "A Promise and Money"

2.8.1. Formulation. A group of young cryptographers are interested in quantum computings and really want to buy a quantum computer. A millionaire gave them a certain amount of money (say, n cryptographers; X_i for each of them, $i = 1, \ldots, n$). He also made a promise from them that they would not tell anyone, including each other, how much money everyone of them had received.

• Could you help the cryptographers to invent an algorithm of how to find out (without breaking the promise) whether the total amount of money they have, $\sum_{i=1}^{n} X_i$, is enough to buy a quantum computer?

• What do you think whether there are such algorithms protecting the secrets of honest participants from dishonest ones?

• What weaknesses does your algorithm have (if someone breaks the promise)? Does it always protect the secret of honest participants from dishonest ones?

2.8.2. Solution. Here we give an idea of the solution proposed by Mikhail Kudinov (Bauman Moscow State Technical University).

First of all, it is supposed that no one can buy a quantum computer himself without other participants. Let us assume that N' is the amount of money that one needs to buy a quantum computer and

$$N = nN',$$

where n is the number of participants. The millionaire gave them X_i money for $i \in \{1, \dots, n\}$. Each participant chooses random secrets $s_{i,j}$ uniformly so that

$$\sum_{j=1}^{n} s_{i,j} \equiv X_i \pmod{N}.$$

Then each of them gives the share $s_{i,j}$ to the owner of X_j by the secure channel. After this procedure, the owner of X_i has shares $s_{k,i}$ for each $k \in \{1, \dots, n\}$. It is obvious that

$$\sum_{j=1}^{n} \sum_{i=1}^{n} s_{i,j} = \sum_{i=1}^{n} X_i \pmod{N}.$$

Under the first suggestion, all participants can together calculate the common amount of money.

The main disadvantage of the algorithm, in addition to the suggestion, is a big amount of private communication (though the number of keys can be n for asymmetric schemes).

By analogy, many participants described algorithms similar to Schneier's calculating average salary algorithm [9]. In general, all these algorithms are vulnerable if $n - 1$ participants are dishonest. Some participants tried to describe a possibility of using a cryptosystem that is homomorphic by "+" and preserves relation "<," as some general analysis.

The problem of the first school round is the same problem for $n = 3$ (score assignment was more loyal). Despite there was quite a few solutions for this problem in the student round, each solution had big or small lacks in analysis of the general case, in analysis of the algorithm advantages and disadvantages, in description of communications (the number of the private communications, what kind of cryptography is used, the number of required private keys), and so on. As a result, there was no possibility to chose the "best of the best" for 6 scores, and we decided to give 5 scores as maximum. There were nine maximal-scored solutions.

2.9. Problem "16QAM"

2.9.1. Formulation. For sending messages, Alice and Bob use a fiber-optic communication via 16QAM technology. This technology allows them to send messages whose alphabet consists of 16 letters, where each letter is usually encoded with a 4-bit Gray code. While a message is transmitted in the channel, single errors in codewords of the Gray code are possible.

Alice has read an interesting book and would like to share her enthusiasm with Bob! Alice sent a short fragment from the book to Bob. Owing to the characteristics of the communication channel used, she divided the text into two parts and sent them separately. In the first part, she placed all of the 16 consonants that occurred in this fragment; in the second part, she placed vowels ("y" is a vowel), a space, a hyphen, and punctuation marks. Then Alice also encoded the letters with a Hamming code to be able to correct single errors. She applied a 7-bit Hamming code with the parity-check matrix whose columns are written in lexicographical order.

Bob received the two parts of ciphertext given in hexadecimal notation (see Table 7).

Also, he received the following number sequence:

22, 19, 3, 3, 36, 53, 3, 33, 20, 28.

Each number indicates how many consonants are contained between the punctuation marks.

Recover the text and find the main character of the book Alice has read!

Table 7. The ciphertext that Bob received from Alice (Problem "16QAM")

Part 1	Part 2
66674C36666F43D3C199900AA1AA325992A	66CA61967319CCD2CE76998CE6433332D19
67A59D9B4A8B69330D1BC000153367A5E33	B46784C65334E999A402ADA0265A99A6633
D30E6692D0F349D3321FFFF0ED706667A7F	33319B32D3299698CCC96986619967134CC
670D999679F4AA67561BA679B4AA54F34D5	B4CE23333334CC6730CE90170CCCD2CE669
AB0F4AACCF000055CE633670D9DA54CE37F	996A61999EA63332CCA4C3332D4CD3334CC
660DE19CD995335495523CCAAA8F1E03325	D3319994730CCCD3A6669D96A66999699B3
86CF48A98CD9B387FD9D546A99E9D200033	98640CC86CE619676AD4CD3308999866D33
3201513FE5B4AA00CCCE9667554CD2CCCB3	79321C33210B4C6732199B53218019A404C
330F32A666553CD756AC3E0674E9D369E1D	D2DE65A986663398CCCCCB5319CC6665997
C6A9999780007F00961E66465519FEA8B25	B96A63398CD9CCD2CD9A399A66339866619
14CCCB332AA63332CCCE6D2A99AACCCC004	98CD9CC325A6339CCE619998C04C66CE633
	996A61998CF66967334CC66CA6199865E$(0)_2$

2.9.2. Solution. Some details in the problem statement are insignificant. Namely, we could omit the step with the Gray code and mind that Alice substitutes 7-bit codewords of the Hamming code for each symbol in each part of the plaintext.

The crucial idea to broke the cipher Alice and Bob use is analyzing the frequency distribution in each part of the ciphertext. This helps them to deduce the probable meaning of the most common symbols and form partial words. Tentative search for the combinations of consonants and vowels giving actual words in English expands the partial solution. Frequencies of the pairs of letters also give an improvement but it could seem inessential. At last, one can employ search engine on the Internet to find the fragment of the book that Alice sent to Bob.

Let us consider a possible solution. Alice uses the Hamming code with the parity check matrix H and the corresponding generator matrix G, where

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}, \qquad G = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

First, rewrite each part of the given ciphertext in the binary form. Split them into 7-bit words and correct errors using the parity check matrix H. One can decode the Hamming code into a 4-bit Gray code but it is not a necessary step for the solution. Calculating the frequencies of the codewords separately in each part of the given ciphertext, we put them in Table 8.

Compare the frequencies obtained with those of letters in the English language. The suitable frequency distribution can be found in [10] cited, e.g., at [11]. According to Lewand, arranged from most to least common in appearance, the letters are:

e t a o i n s h r d l c u m w f g y p b v k j x q z.

We start with vowels, punctuation marks, spaces, and a hyphen, which are placed in Part 2. Make a guess that the most frequent symbol in Part 2 is the space. It is also worth to note that most of the punctuation marks are followed by a space in contrast to a hyphen, which is usually embraced by letters. Using letter frequencies, we determine the probable spaces, vowels, and hyphen and construct the following partial solution for this part of the plaintext (the sign # substitutes punctuation):

```
ee ae e oe o e ua iaia# e oo oy-oy i o ea ee# u# ea# auae o ie ea o e aoy a oe
o i a i eae# a i o o o eae a oo o i o iee ay ue aeii o aa aie# uuay# e uai uy oy
oe i a e ea i e eae# i e ee oeee o e a a ee a# e e a uy ee e i a e oe o ee a a#
```

Let us turn to Part 1 which contains 16 consonants occurring in the fragment of the book. Let us order the codewords of the Hamming code from most to least frequent in Part 1, as it is shown in Table 8, a. Denote the 7-bit codewords by hexadecimal numbers from 0 till F. Then we get the following ciphertext

Table 8. Frequencies of Hamming codewords in the text

Gray code	Hamming code	Frequency
1011	0110011	46
0010	0101010	30
1001	0011001	24
0001	1101001	24
0011	1000011	19
0000	0000000	15
0110	1100110	13
1100	0111100	8
1111	1111111	8
1101	1010101	7
0100	1001100	6
1110	0010110	5
1010	1011010	5
0101	0100101	4
1000	1110000	4
0111	0001111	2

(a) Part 1

Gray code	Hamming code	Frequency
0100	1001100	85
1011	0110011	50
1001	0011001	33
0001	1101001	26
1010	1011010	17
0011	1000011	9
0000	0000000	8
1110	0010110	7
1100	0111100	2
0010	0101010	1
1000	1110000	1
0111	0001111	0
0101	0100101	0
1101	1010101	0
0110	1100110	0
1111	1111111	0

(b) Part 2

of 220 symbols in length that is splitted into 10 pieces (according to the number sequence given in the task):

```
023402C43E0251412B0103  02C1B32407551003703  4A3  B46  33A4884CE02E804020631094106311739943
1675510A0040C1068047266101D10619FF56D4031A00048090103  355
025108B315023021A3020246102173994  E2333C72410275585D46  021281BD102021A0202631016055
```

Then we match the symbol frequencies in Part 1 of the ciphertext with those of consonants in the English alphabet. The first five pairs are like as follows: `0 - t`, `1 - n`, `2 - s/h`, `3 - s/h`, and `4 - r`.

The bigram "`th`" is the most frequent in English. This allows us to make a suggestion that "`2`" substitutes "`h`" and "`3`" substitutes "`s`." Then we obtain a partial solution for Part 1 and, combining with one for Part 2, get the following pieces of the plaintext given in Table 9. It is not difficult to recognize words "`these are the`" at the beginning in (1). Also, we can see "`the`" as the first word in (2) and (8).

The best idea for the next step is to search through the English dictionary for the words that have given vowels in the prescribed order. It is possible to use one of the tools for the pattern recognition available on the Internet, e.g., [12]. Advanced participants of the Olympiad implemented some computer programs on their own.

Consider several examples. We have a word with consonants "`s55`" and vowels "`uuay`" in (7), and the last two consonants are identical. The only match is "`usually`", so we assume that "`5`" substitutes the letter "`l`." The pattern "`auae`" in combination with double "`s`" gives us two possibilities in (5): "`assuage`" and "`sausage`." In any case, it seems like "`A`" means "`g`." Then we have "`rugs`" in (3). The pattern "`uai`" and consonants "`5nt8B`" lead us to "`lunatic`" in (8), so "`8`" probably means "`c`."

At this point we revise our matching the letters and their frequencies corresponding to the Part 1 of the ciphertext. Let us look at the first eight letters with large frequencies: "`t n h s r l 6 7/c`."

Table 9. Partial plaintext

No.	Partial plaintext
(1)	thsrthCrsEth5nrnhBtnts ee ae e oe o e ua iaia#
(2)	thCnBshrt755ntts7ts e oo oy-oy i o ea ee#
(3)	rAs u#
(4)	Br6 ea#
(5)	ssAr88rCEthE8trtht6snt9rnt6snn7s99rs auae o ie ea o e aoy a oe o i a i eae#
(6)	n6755ntAttrtCnt68tr7h66ntnDnt6n9FF56DrtsnAtttr8t9tnts a i o o o eae a oo o i o iee ay ue aeii o aa aie#
(7)	s55 uuay#
(8)	th5nt8Bsn5thsthnAsththr6nthn7s99r e uai uy oy oe i a e ea i e eae#
(9)	EhsssC7hrnth75585Dr6 i e ee oeee o e a a ee a#
(10)	thnh8nBDnththnAthth6sntn6t55 e e a uy ee e i a e oe o ee a a#

We can see that the letter "d" has still been hidden. According to the Lewand distribution it is the most probable that "6" means "d." Then (4) contains "Brd" and "ea" that gives us possible words "beard" and "bread." Therefore, it seems like "B" substitutes "b."

A thorough analysis of the remaining ciphertext and search for words by patterns and number of letters eventually lead us to the plaintext (with punctuation replaced by #):

these are the mores of the lunar inhabitants# the moon boy-shorty will not eat sweets# rugs# bread# sausage or ice cream of the factory that does not print ads in newspapers# and will not go to treatment a doctor who did not invented any puzzle advertising to attract patients# usually# the lunatic buys only those things that he read in the newspaper# if he sees somewhere on the wall a clever ad# then he can buy even the thing that he does not need at all#

This is a fragment of the fairytail novel "Dunno on the Moon" by the Russian writer Nikolay Nosov. The title character of the novel is a boy-shorty Dunno. The problem was completely solved by 13 teams in the second round and by Samuel Tang (Hong Kong, Black Bauhinia) in the first round. The best solutions were proposed by the team of Irina Slonkina, Mikhail Sorokin, and Vladimir Bobrov (Bauman Moscow State Technical University) and the team of Vladimir Paprotski, Dmitry Zarembo, and Karina Kruglik (Belarusian State University).

2.10. Problem "APN + Involutions"

The first three questions **Q1**, **Q2**, and **Q3** were given as the problem "APN + Involutions" in the first round. The extended version of the task for the second round included also Question **Q4** that contains some open problems.

2.10.1. Formulation. Alice wants to construct a block cipher with heavy use of **involutions** as subcomponents; this minimizes the difference between the algorithms for encryption and decryption. She knows that APN *permutations* are the best choice of subcomponents to resist the attacks based on differential technique. She wants to construct some set of APN permutations that are involutions for every $n \geqslant 2$.

Alice knows that every involution can be expressed as the product of disjoint *transpositions*. So, she decides to study the following involution

$$g = \prod_{i=1}^{d} (\alpha_i, \alpha'_i),$$

where $\{\alpha_i, \alpha'_i\} \cap \{\alpha_j, \alpha'_j\} = \varnothing$ for all $i, j \in \{1, ..., d\}$, $i \neq j$, and $1 \leqslant d \leqslant 2^{n-1}$.

Alice needs your help to get APN permutations among such involutions g. Find answers to the following questions!

Q1: Let

$$\Lambda(g) = \{\alpha_i \oplus \alpha'_i : \ i = 1, ..., d\}, \qquad \widehat{\Lambda}(g) = [\alpha_i \oplus \alpha'_i : \ i = 1, ..., d],$$
$$\mathrm{B}(g) = \{x \oplus y : \ \{x, y\} \subseteq \mathrm{FixP}(g), \ x \neq y\}, \qquad \widehat{\mathrm{B}}(g) = [x \oplus y : \ \{x, y\} \subseteq \mathrm{FixP}(g), \ x \neq y],$$

where $\mathrm{FixP}(g)$ is the set of all *fixed points* of g; i.e. $\mathrm{FixP}(g) = \{x \in \mathbb{F}_2^n : \ g(x) = x\}$.

Suppose that g is an APN permutation. Get necessary conditions for multisets $\widehat{\Lambda}(g)$, $\widehat{\mathrm{B}}(g)$ and sets $\Lambda(g)$, $\mathrm{B}(g)$. Prove that if your conditions do not hold then g is not an APN permutation.

Q2: Let $\mathrm{d}_{a,b}(g) = |\{x \in \mathbb{F}_2^n : g(x \oplus a) \oplus g(x) = b\}|$, $a, b \in \mathbb{F}_2^n$. Let g be an involution and APN. Find $\mathrm{d}_{a,a}(g)$ for each nonzero $a \in \mathbb{F}_2^n$.

Q3: Can you get the nontrivial upper bound on $|\mathrm{FixP}(g)|$?

Q4: Let M_n be the set of all n-bit involutions that are APN permutations.

(1) Can you find the size of M_n for $n = 2, 3, 4$?

(2) Can you find the size of M_n for $n = 5$?

(3) *A Bonus Problem (extra scores, a special prize!)*

Let $n \geqslant 6$. Can you get the lower and the upper bounds for the size of M_n? Can you describe involutions from M_n? Can you suggest constructions for involutions from M_n?

Note that the mapping $x \mapsto x^{-1}$ in the Galois field $GF(2^n)$ belongs to M_n for odd $n \geqslant 3$.

Remark. Let us recall some relevant definitions:

• $\mathbb{F}_2^n$ is the vector space of dimension n over $\mathbb{F}_2 = \{0, 1\}$.

• A vector $x \in \mathbb{F}_2^n$ has the form $x = (x_1, ..., x_n)$, where $x_i \in \mathbb{F}_2$. For two vectors $x, y \in \mathbb{F}_2^n$ their sum is $x \oplus y = (x_1 \oplus y_1, ..., x_n \oplus y_n)$, where $\oplus$ stands for XOR operation.

• Let $\widehat{X} = [x_1, ..., x_d]$ be a multiset with the underlying set $\mathbb{F}_2^n$, where $x_1, ..., x_d \in \mathbb{F}_2^n$. Note that all elements in a set are distinct. Unlike a set, a multiset allows for multiple instances for each of its elements.

• A *permutation* s is a mapping from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ such that $s(x) \neq s(y)$ for all $x, y \in \mathbb{F}_2^n$, $x \neq y$.

• An *involution* s is a permutation that is its own inverse, $s^2(x) = s(s(x)) = x$ for all $x \in \mathbb{F}_2^n$.

• For every different vectors $\alpha, \beta \in \mathbb{F}_2^n$, a permutation s is called a *transposition* if $s(\alpha) = \beta$, $s(\beta) = \alpha$, and $s(x) = x$ for all $x \in \mathbb{F}_2^n \backslash \{\alpha, \beta\}$; it is denoted by $s = (\alpha, \beta)$.

• A permutation s is called APN (Almost Perfect Nonlinear) if, for every nonzero $a \in \mathbb{F}_2^n$ and every $b \in \mathbb{F}_2^n$, the equation $s(x \oplus a) \oplus s(x) = b$ has at most 2 solutions.

2.10.2. Solution. Consider the solutions of the problem.

Q1: Let $a \in \Lambda(g)$. Hence, $a = x \oplus y$, where $y = g(x)$ and $(x, y) = (\alpha_i, \alpha_i')$ for some i. Then

$$g(x \oplus a) = g(y) = x = y \oplus a = g(x) \oplus a.$$

Let $a \in \mathrm{B}(g)$. Hence, $a = x \oplus y$, where $x, y \in \mathrm{FixP}(g)$. Then

$$g(x \oplus a) = g(y) = y = x \oplus a = g(x) \oplus a.$$

Thus, $\mathrm{d}_{a,a}(g) \geqslant 2$ for every vector $a \in \Lambda(g) \cup \mathrm{B}(g)$.

Let g be an APN permutation. Then $\mathrm{d}_{a,a}(g) = 2$. Hence, the multiplicity of all elements from $\Lambda(g)$ and $\mathrm{B}(g)$ is 1. Thus, $\Lambda(g) = \widehat{\Lambda}(g)$ and $\mathrm{B}(g) = \widehat{\mathrm{B}}(g)$. Note that $\Lambda(g) \cap \mathrm{B}(g) = \varnothing$.

Q2: Since g is an APN permutation; therefore, $\mathrm{d}_{a,a}(g) \leqslant 2$. As we get in **Q1**, $\mathrm{d}_{a,a}(g) = 2$ for every vector $a \in \Lambda(g) \cup \mathrm{B}(g)$. Let us prove that $\mathrm{d}_{a,a}(g) = 0$ for $a \notin \Lambda(g) \cup \mathrm{B}(g)$.

Let a be a nonzero vector and x be a solution of $g(x \oplus a) \oplus g(x) = a$. Since g is a permutation, either $x \in \mathrm{FixP}(g)$ or $x = \alpha_i$ $(x = \alpha_i')$ for some i. Consider the two cases:

1. Let $x \in \mathrm{FixP}(g)$. Then, $g(x \oplus a) \oplus g(x) = a$ implies $g(x \oplus a) = x \oplus a$. Hence, $x \oplus a \in \mathrm{FixP(g)}$. As a result, $a \in B(g)$.

2. Without loss of generality, let $x = \alpha_i$ for some i and $y = x \oplus a$. If $y \in \mathrm{FixP}(g)$ then $g(x \oplus a) \oplus g(x) = a$ implies $g(x) = x$, which is a contradiction. Hence, without loss of generality, $y = \alpha_j'$ for some j (so, we have $\alpha_i \oplus \alpha_j' = a$). Then

$$g(\alpha_i \oplus a) \oplus g(\alpha_i) = a \;\;\Rightarrow\;\; g(\alpha_j') \oplus \alpha_i' = a \;\;\Rightarrow\;\; \alpha_j \oplus \alpha_i' = a.$$

Let us show that α_i' and α_j is also solutions. Indeed,

$$g(\alpha_i' \oplus a) \oplus g(\alpha_i') = g(\alpha_j) \oplus \alpha_i = \alpha_j' \oplus \alpha_i = a, \quad g(\alpha_j \oplus a) \oplus g(\alpha_j) = g(\alpha_i') \oplus \alpha_j' = \alpha_i \oplus \alpha_j' = a.$$

Thus, if $i \neq j$ then we get at least 3 solutions that is a contradiction for the APN property of g. Hence, $j = i$ and $a \in \Lambda(g)$.

Q3: Let us prove that $|\mathrm{FixP}(g)| \leqslant 1 + (2^{n-1} - 1)^{1/2}$.

The involution g is APN. From **Q1** we have

$$\mathrm{B}(g) \cap \Lambda(g) = \varnothing. \tag{1}$$

Let $q = |\mathrm{FixP}(g)|$. Since g is an involution, q is even. Owing to (1) and $\Lambda(g) \cup \mathrm{B}(g) \subseteq \mathbb{F}_2^n \backslash \{\mathbf{0}\}$, we have

$$|\Lambda(g)| + |\mathrm{BB}(g)| \leqslant 2^n - 1. \tag{2}$$

Since $|\mathrm{B}(g)| = \binom{q}{2}$, $|\Lambda(g)| = 2^{n-1} - q/2$, we have $|\Lambda(g)| + |\mathrm{BB}(g)| = q(q-1)/2 + 2^{n-1} - q/2$.

From (2), we have $q(q-1)/2 + 2^n - q \leqslant 2^n - 1$. Thus, $q(q-2)/2 \leqslant 2^{n-1} - 1$; i.e.,

$$q \leqslant 1 + (2^{n-1} - 1)^{1/2}.$$

Q4: (a) It could be computationally verified that $M_2 = \varnothing$ and $|M_3| = 224$. Then, it is known [13] that there are no APN permutations for $n = 4$. Hence, $M_4 = \varnothing$.

(b) Recall some definitions:. A function $A : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is *affine* if $A(x \oplus y) = A(x) \oplus A(y) \oplus A(\mathbf{0})$ for all $x, y \in \mathbb{F}_2^n$. Two functions $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ are called *affine equivalent* if there exist affine permutations A_1 and A_2 such that $F = A_1 \circ F \circ A_2$. It is easy to see that the APN permutation property of a function is an invariant under the affine equivalence. There exist [13] only five affine equivalence classes of APN permutations. Moreover, by [13, theorem 3], only one class contains functions together with their inverses. Hence, only this class of APN permutations can contain involutions. The representative of this class is the famous inverse function over the finite field: $F(x) = x^{-1}$ for nonzero x and $F(0) = 0$ (here, functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ are considered as functions over the finite field of order 2^n). The inverse function is an involution. Thus, all APN involutions for $n = 5$ are affine equivalent to the inverse function.

(c) There were no interesting suggestions by the participants for these open problems.

The unique full correct solution in the first round was proposed by Henning Seidler (Germany, TU Berlin). In the second round, the best solution for 11 scores was proposed by the team of Kristina Geut, Sergey Titov, and Dmitry Ananichev (Russia, Ural State University of Railway Transport, Ural Federal University).

2.11. Problem "Sharing"

2.11.1. Formulation. Bob is interested in studying mathematical countermeasures to side-channel attacks on block ciphers. He found out that the techniques such as special sharings of functions can be applied. Now he is thinking about the following mathematical problem in this approach:

Let $\mathcal{F}$ denote the set of *invertible functions* (*permutations*) from $\mathbb{F}_2^4$ to $\mathbb{F}_2^4$ and let $\mathcal{F}^n$ denote the set of invertible functions from $(\mathbb{F}_2^4)^n$ to $(\mathbb{F}_2^4)^n$. Let $F \in \mathcal{F}^n$ be

$$F(x_1, x_2, \ldots, x_n) = \big(F_1(x_1, x_2, \ldots, x_n), F_2(x_1, x_2, \ldots, x_n), \ldots, F_n(x_1, x_2, \ldots, x_n)\big),$$

with component functions $F_i : (\mathbb{F}_2^4)^n \to \mathbb{F}_2^4$, $i = 1, \ldots, n$.

For every $f \in \mathcal{F}$, a function $F \in \mathcal{F}^n$ is called a *sharing* of f if

$$\sum_{i=1}^{n} F_i(x_1, x_2, \ldots, x_n) = f\left(\sum_{i=1}^{n} x_i\right) \quad \text{for all } (x_1, x_2, \ldots, x_n) \in (\mathbb{F}_2^4)^n.$$

Moreover, F is an *noncomplete* sharing of f if F is a sharing of f with the additional property that each component function F_i is independent of x_i.

Bob needs your help to study functions for which a noncomplete sharing exists. Find answers to the following questions!

Q1: Let $\mathcal{A}$ denote the set of *affine functions* from $\mathbb{F}_2^4$ to $\mathbb{F}_2^4$. Two functions $f, g \in \mathcal{F}$ are *affine equivalent* if there exist $a, b \in \mathcal{A}$ such that $g = b \circ f \circ a$.

Let f and g be two functions in the same affine equivalence class of $\mathcal{F}$ and let F be a noncomplete sharing of f. Derive from F a noncomplete sharing for g.

All functions of the same affine equivalence class have the same degree. It is known [14] that this equivalence relation partitions $\mathcal{F}$ into 302 classes: 1 class corresponds to $\mathcal{A}$, 6 classes contain quadratic functions, and 295 classes contain cubic functions.

Also, Bob knows that when $n \geqslant 5$ then there exists a noncomplete sharing for each $f \in \mathcal{F}$ (it can be shown by construction). When $n = 2$ then a noncomplete sharing exists only for the functions in $\mathcal{A}$. When $n = 3$ then a noncomplete sharings exist for $\mathcal{A}$ and also for 5 out of the 6 equivalence classes containing quadratic functions. When $n = 4$ then noncomplete sharings exist for $\mathcal{A}$, for all 6 quadratic equivalence classes, and for 5 cubic classes.

Q2: *A Bonus problem (extra scores, a special prize!)*

Find a concise mathematical property that $f \in \mathcal{F}$ must have in order that a noncomplete sharing F exists for $n = 3$ and $n = 4$.

Q3: *A Bonus problem (extra scores, a special prize!)*

Generalize to functions over $\mathbb{F}_2^5$ and $\mathbb{F}_2^6$.

2.11.2. Solution. **Q1:** Let f and g be two functions in the same affine equivalence class of $\mathcal{F}$; i.e., $g = b \circ f \circ a$ for some $a, b \in \mathcal{A}$, and let $F \in \mathcal{F}^n$ be a noncomplete sharing of f. At first, one can notice that since f and g are invertible, the mappings a and b must be invertible as well. Let us denote

$$a(x) = Ax + a', \qquad x \in \mathbb{F}_2^4, \qquad\qquad b(x) = Bx + b', \qquad x \in \mathbb{F}_2^4,$$

where A and B are nonsingular binary matrices of order 4×4 and $a', b' \in \mathbb{F}_2^4$.

Using the components functions $\{F_i\}_{i=1}^n$ of F, we define the invertible function $G \in \mathcal{F}^n$ with components functions

$$G_j\,(x_1, x_2, ..., x_n) = \begin{cases} BF_1\,(Ax_1 + a', Ax_2, ..., Ax_n) + b', & j = 1, \\ BF_j\,(Ax_1 + a', Ax_2, ..., Ax_n)\,, & j \neq 1, \end{cases}$$

where $j = 1, 2, ..., n$.

Then for every $(x_1, x_2, ..., x_n) \in \left(\mathbb{F}_2^4\right)^n$, we have

$$\sum_{j=1}^{n} G_j(x_1, x_2, ..., x_n) = BF_1\left(Ax_1 + a', Ax_2, ..., Ax_n\right) + b'$$

$$+ \sum_{j=2}^{n} BF_j\left(Ax_1 + a', Ax_2, ..., Ax_n\right) = B\left(\sum_{j=1}^{n} F_j\left(Ax_1 + a', Ax_2, ..., Ax_n\right)\right) + b'$$

$$= Bf\left(Ax_1 + a' + Ax_2 + \ldots + Ax_n\right) + b'$$

$$= Bf\left[A\left(\sum_{i=1}^{n} x_i\right) + a'\right] + b' = b \circ f \circ a\left(\sum_{i=1}^{n} x_i\right) = g\left(\sum_{i=1}^{n} x_i\right).$$

Therefore, the function $G \in \mathcal{F}^n$ defined as

$$G(x_1, x_2, ..., x_n) = \left(G_1(x_1, x_2, ..., x_n), G_2(x_1, x_2, ..., x_n), ..., G_n(x_1, x_2, ..., x_n)\right),$$

is a sharing of g.

From noncompleteness of F it follows that G_j, which is in fact an affine transformation of F_j, does not depend on x_j. Hence, G is a noncomplete sharing of g.

Q2–Q3: These open problems were not solved completely during the Olympiad. Nevertheless, one perspective solution was proposed by the team of Victoria Vlasova, Mikhail Polyakov, and Alexey Chilikov (Bauman Moscow State Technical University). They found a sufficient condition for the existence of a noncomplete sharing for $n = 3$. Let us describe it here.

Let $\mathrm{wt}(y)$ be the Hamming weight of a binary vector y. Given $\sigma \in \mathbb{F}_2$, put

$$\delta_\sigma(y) = \begin{cases} y, & \sigma = 1, \\ \mathbf{0}, & \sigma = 0, \end{cases}$$

where $\mathbf{0}$ is the zero vector of the same dimension as y.

Let V be a vector space over the field K and assume that for the invertible function $f : V \to V$ it holds

$$\sum_{\sigma \in \mathbb{F}_2^n} (-1)^{\mathrm{wt}(\sigma)} f\left(\sum_{i=1}^{n} \delta_{\sigma_i}(x_i)\right) = 0. \tag{3}$$

Then there exists a noon-complete sharing for f. Further we consider the case $n = 3$.

Indeed, given $(x_1, x_2, x_3) \in V^3$, put

$$F_1(x_1, x_2, x_3) = f(x_2) - f(x_2 + x_3), \qquad F_2(x_1, x_2, x_3) = f(x_3) - f(x_1 + x_3),$$

$$F_3(x_1, x_2, x_3) = f(x_1) - f(x_1 + x_2).$$

It is clear that every $F_i : V^3 \to V$ does not depend on x_i, where $i = 1, 2, 3$. Consider the expression

$$\sum_{i=1}^{3} F_i(x_1, x_2, x_3) = f(x_2) - f(x_2 + x_3) + f(x_3) - f(x_3 + x_1) + f(x_1) - f(x_1 + x_2)$$

$$= \sum_{\sigma \in \mathbb{F}_2^3} (-1)^{\mathrm{wt}(\sigma)} f\left(\sum_{i=1}^{3} \delta_{\sigma_i}(x_i)\right) + f(x_1 + x_2 + x_3) - f(0) = f(x_1 + x_2 + x_3) - f(0).$$

Without loss of generality we assume that $f(0) = 0$. Otherwise, we can consider the initial problem for the function $g(x) = f(x) - f(0)$ with $g(0) = 0$ and which, by the arguments from **Q1**, has a noncomplete sharing if and only if f does.

Finally, $\sum_{i=1}^{3} F_i(x_1, x_2, x_3) = f(x_1 + x_2 + x_3)$, which completes the proof.

It was also shown by the authors that the condition (3) is necessary for the existence of a noncomplete sharing of f for all n.

Taking $V = \mathbb{F}_2^m$ with $m = 4, 5, 6$ and $K = \mathbb{F}_2$, we can obtain a solution of **Q2** and **Q3** for the case $n = 3$.

2.12. Problem "Factoring in 2019"

2.12.1. Formulation. Nicole is learning about the RSA cryptosystem. She has chosen random 500-bit prime numbers p and q, $2^{499} \leqslant p, q < 2^{500}$, and computed $n = p \cdot q$. Being a curious and creative person, she has also combined the three numbers in funny ways. Her favorite one is an integer h such that

$$h \equiv 3^{2019}p^2 + 5^{2019}q^2 \pmod{n^2 + 8 \cdot 2019}.$$

Unfortunately, she has lost the paper where she wrote the two prime numbers. Luckily, she remembers n and h. Help Nicole to recover p and q.

$$\begin{aligned} n = {} & 40763613025504836845249840044831561583564626405535158138667037\\ & 18791672670905308860844304055285019651507728831663677166092475\\ & 16155419756121537288444995708421977847213953345126368990185271\\ & 10259760189356588305406519080647582874212687596214191915933827\\ & 67252094717222418132289251314647500491996323400002019, \end{aligned}$$

$$\begin{aligned} h = {} & 78307999278336577586961528110240026923828914927526911949501196\\ & 64549497756373569985393554661132717198368717093111812566649031\\ & 17342818449633588647098544612151278035131454234786653136500887\\ & 08830470996542888912418213532073622903727205396807848603735835\\ & 72653630883685906916701587362236649126895719656663293825501223\\ & 97088799629252601249428062432254738935764304610281613264225641\\ & 74990272864680012560095992125783832230234589257650929348364268\\ & 48117494065463529201859600747521892957258104033195441014023432\\ & 36581529201392185327635674923459290749241831590661903965132514\\ & 21544515183088866585058200066678369344118 81. \end{aligned}$$

2.12.2. Solution. This problem is based on a (simplified) variation of the Coppersmith method.

Let $m = n^2 + 8 \cdot 2019$. It is a composite number with unknown factors. The idea is to find an integer a such that the numbers $a_1 = a \cdot 3^{2019} \mod m$ and $a_2 = a \cdot 5^{2019} \mod m$ are small enough and $a_1p^2 + a_2q^2$ exceeds the modulus m by a small amount and can be recovered from $a \cdot h \mod m$. This can be done using the Lagrange–Gauss algorithm (which is a special case and the building block of the LLL algorithm). Let Λ be the lattice spanned by the two vectors

$$v_1 = \big(1,\ (5^{2019} \cdot (3^{2019})^{-1} \mod m)\big), \qquad v_2 = \big(0,\ m\big).$$

Consider an arbitrary vector $v = (a_1, a_2)$ in this lattice. It is easy to verify that

$$a_1p^2 + a_2q^2 \equiv a_1 \cdot h \cdot (3^{2019})^{-1} \pmod{m}.$$

The lattice reduction guarantees to find such vector v with the norm

$$\|v\| = \sqrt{a_1^2 + a_2^2} \leq 2^{(d-1)/4}(\det \Lambda)^{1/d} = \sqrt{m}/\sqrt[4]{2},$$

where $d = 2$ is the dimension of the lattice. In particular,

$$|a_1p^2 + a_2q^2| \leq n(p^2 + q^2) < n(p + q)^2 < 10n^2,$$

where the last two inequalities follow from the balancedness of the primes (i.e., $\max(p, q) \leq 2\min(p, q)$).

It follows that there exists an integer z, $|z| < 10$, such that

$$a_1 \cdot h \cdot (3^{2019})^{-1} \mod m + zm = a_1p^2 + a_2q^2.$$

In result, we obtain an equation in p^2 and q^2. By replacing $p = n/q$, we obtain a biquadratic equation in q which is easy to solve and factor n.

The final solution is:

$$p = 2019000075878154181681129810414477022346818209175194524879208890921501144547048007953722271285690350264116081579241189587393202602664199899594021414383,$$

$$q = 2019000073973494194521339805682093959182265746083995594826393753631669289175827851666668014167119439386543289850940734885806826120718179729242641026893.$$

The best solution was proposed by Alexey Zelenetskiy, Mikhail Kudinov, and Denis Nabokov team (Russia, Bauman Moscow State Technical University).

2.13. Problem "`TwinPeaks3`*" (online)*

2.13.1. Formulation. As Bob's previous cipher `TwinPeaks2` (NSUCRYPTO-2018) was broken again, he finally decided to read some books on cryptography. His new cipher is now inspired by practical ciphers, while the number of rounds was reduced a bit for better performance.

Not only the best techniques were adopted by Bob, but also he decided to enhance his cipher by security through obscurity, so the round functions are now unknown. The only thing known about these functions is that they are the same for odd and even rounds.

New Bob's cipher works as follows: A message X is represented as a binary word of length 128. The latter is divided into four 32-bit words a, b, c, and d; then the following round transformation is applied 32 times:

$$(a, b, c, d) \leftarrow (b, c, d, a \oplus (F_i(b, c, d))),$$
$$F_i = F_1 \text{ for odd rounds and } F_i = F_2 \text{ for the rest.}$$

Here F_1 and F_2 are secret functions accepting three 32-bit words and returning one word; and $\oplus$ is the binary bitwise XOR. The concatenation of the final a, b, c, d is the resulting ciphertext Y for the message X.

Agent Cooper again wants to read the Bob's messages. He caught the ciphertext

$$Y = \texttt{e473f19a247429ab33b66268d57dd241}$$

(the ciphertext is given in hexadecimal notation, the first byte is `e4`).

He was also able to gain access to Bob's testing server with encryption and decryption routines, using the secret key (see [15]). Unfortunately, the version of software available on this server is not final. So, the decryption routine is incomplete and only uses keys in the reverse order, which is not sufficient for decryption:

$$(a, b, c, d) \leftarrow (b, c, d, a \oplus (F_i(b, c, d))),$$
$$F_i = F_2 \text{ for odd rounds and } F_i = F_1 \text{ for the rest.}$$

The server can also process multiple blocks of text at a time: they will be processed one-by-one and then concatenated, as in the regular ECB cipher mode of operation. Ciphertexts and plaintexts are given and processed by the server in hexadecimal notation.

Help Cooper to decrypt Y.

2.13.2. Solution. Let f_i be the round transformation of round i:

$$f_i : \ (a, b, c, d) \leftarrow (b, c, d, a \oplus (F_{k(i)}(b, c, d))),$$

where $k(i) = 1$ for odd i and $k(i) = 2$ otherwise.

Hence, we can represent the encryption transformation E as $E = (f_1 f_2)^{16}$.

Let I be the incomplete decryption transformation described in the problem statement. The encryption and the incomplete decryption processes only differ in the key order, so I can be written as $I = (f_2 f_1)^{16}$.

The decryption transformation E^{-1} can be represented as $E^{-1} = (f_2^{-1} f_1^{-1})^{16}$, where f_i^{-1} is the inverse of f_i and is given by the transformation

$$f_i^{-1} : (a, b, c, d) \leftarrow (d \oplus (F_{k(i)}(a, b, c)), a, b, c).$$

Thus, to apply E^{-1} to the ciphertext one should be able to compute $F_1(x, y, z)$ and $F_2(x, y, z)$ that are secret. To recover these functions a *slide attack* can be used.

The idea is to find the words $x = (x_1, x_2, x_3, x_4)$ and $y = (y_1, y_2, y_3, y_4)$ such that $f_i(x) = y$. If such a pair is found then F_i can be found as $F_i(x_2, x_3, x_4) = y_4 \oplus x_1$.

We use the following idea to find a desired pair: If $Ef_i(x) = E(y)$ then $f_i(x) = y$. Let us start with F_1. We need a pair of x and y such that $Ef_1(x) = E(y)$. This relation can be written as

$$(f_1 f_2)^{16} f_1(x) = (f_1 f_2)^{16}(y), \qquad f_1 (f_2 f_1)^{16}(x) = (f_1 f_2)^{16}(y), \qquad f_1 I(x) = E(y).$$

We come to a conclusion that if $f_1 I(x) = E(y)$ then $f_1(x) = y$. The condition $f_1 I(x) = Ey$ can be checked by using the definition of f_1: if

$$(I(x))_2 = (E(y))_1, \qquad (I(x))_3 = (E(y))_2, \qquad (I(x))_4 = (E(y))_3$$

then it is *likely* that $f_1 I(x) = E(y)$. The probability of false positives is approximately 2^{-96} for random F_i functions. So, it can be considered as negligible. Both $I(x)$ and $E(y)$ are available on the encryption oracle for arbitrary x and y as the incomplete decryption and the encryption routines respectively.

To find $F_i(a, b, c)$, let us brute force over x and y of the following forms: $x = (X, a, b, c)$ and $y = (a, b, c, X')$. According to the birthday paradox, a desired pair can be found in $2 * 2^{16}$ operations average (instead of 2^{32} if we lock X or X' to some constant value).

As soon as we find such a pair x and y, we can compute $F_1(a, b, c)$ and apply f_1^{-1} to the ciphertext and decrypt the last round. Then F_2 can be found in the same way by replacing I and E with each other due to the symmetry. By doing this round by round, we decrypt the whole ciphertext and get the desired message (in hexadecimal notation)

`acherrypieplease`

The reference implementation of this attack requires 2^{22} blocks of text to be encrypted and 10 minutes of time average. It is important to use the server's ability to process multiple blocks of text at a time to minimize the amount of HTTP requests.

Four teams successfully solved the problem using the same method.

2.14. Problem "`Curl27`*"*

2.14.1. Formulation. Bob is developing the 3OTA infrastructure and has designed a new hash function Curl27 for it. A distinguishing feature of the infrastructure is the ternary logic: Trits from the set $\mathbf{T} = \{0, 1, -1\}$ are used instead of bits, ternary strings and words are used instead of binary ones. The Curl27 hash function is defined below. Its implementation in Java can be found in [16].

Find a collision for Curl27; i.e., different ternary strings X and X' such that $\mathsf{Curl27}(X) = \mathsf{Curl27}(X')$. Submit colliding strings as two lines of trits separated by commas. An example of a (wrong!) solution is:

$$-1, 1, 0, 1, 1, 0 \qquad -1, -1, 1, 0, 1, 1, -1, 0$$

Description of Curl27. The Curl27 function maps a ternary string X of arbitrary length to a hash value from $\mathbf{T}^{243}$. When hashing, an auxiliary sponge function Curl27-f: $\mathbf{T}^{729} \to \mathbf{T}^{729}$ is used. The hashing algorithm is as follows:

(1) Pad X with zeros to make its length a multiple of 243. Divide the resulting string into blocks $X_1, X_2, \ldots, X_d \in \mathbf{T}^{243}$.

(2) Prepare the state $W = W_0 W_1 W_2 \in \mathbf{T}^{729}$ consisting of words $W_i \in \mathbf{T}^{243}$. Initialize the state by filling W_0 and W_2 with zeros and W_1 with the encoded initial (before padding) length of X. The length is encoded by a ternary word according to the little-endian conventions: less significant trits go first. For example, the length $25 = 1 - 3^1 + 3^3$ is presented by the word $\underbrace{1\bar{1}01000\ldots0}_{243}$. Here $\bar{1}$ stands for -1.

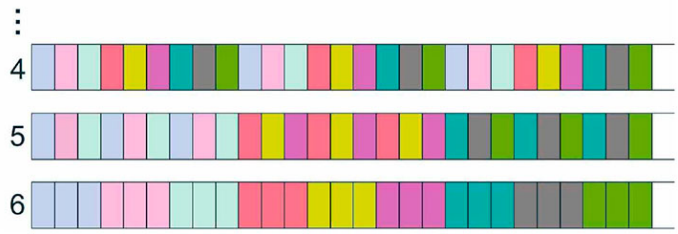

Fig. 7. Groupings (3 last steps, grouped trits are painted the same color).

(3) For $i = 1, 2, \ldots, d$, do: $W_0 \leftarrow X_i$, $W \leftarrow$ Curl27-f(W).
(4) Return W_0.

Description of Curl27-f. In Curl27-f the S-box

$$S\colon \mathbf{T}^3 \to \mathbf{T}^3, \qquad (a, b, c) \mapsto (F(a, b, c), F(b, c, a), F(c, a, b))$$

is used. Here

$$F(a, b, c) = a^2b^2c + a^2bc^2 - ab^2c^2 + a^2b^2 - a^2bc + a^2c^2 + ab^2c - a^2c + ab^2 - ac^2 + b^2c + bc^2 - a^2 - b^2 + bc - c^2 - c + 1,$$

where the calculations are carried out modulo 3 while the residue 2 is represented by the trit -1.

To transform the state W, 27 rounds are performed. A round consists of 6 steps. At each step triplets of trits of W are grouped in a certain way. Then each triplet (a, b, c) is replaced with $S(a, b, c)$.

Groupings are organized as follows (see Fig. 7): At the first step, the state is divided into 3 words of 243 trits. Trits of these words in the same positions are grouped. At the second step, the state is divided into 9 words of 81 trits. Trits of the 1st, 2nd and 3rd words in the same positions are grouped, then trits of the 4th, 5th and 6th words, and so on. After that, the state is divided into words of length 27, then length 9, then length 3, while maintaining the logic of groupings. At the last sixth step, consecutive triplets of trits are grouped.

A bonus problem (extra scores, a special prize!). Find a collision when the state is initialized in a different way: Now W_0 and W_2 are not filled with 0s; in each of them, $\underbrace{01\bar{1}01\bar{1}\ldots01\bar{1}}_{243}$ is written instead.

2.14.2. Solution. For a word u in the alphabet $\mathbf{T}$, let u^m be the word of m copies of u. Supposing $u = u_0u_1\ldots u_{n-1}$ denote $u^{[m]} = u_0^m u_1^m \ldots u_{n-1}^m$. We call a word of the form $u^{[m]}$ *m-fragmented*.

Theorem. *Let m be a power of 3, $m \le 729$. The sponge function Curl27-f preserves m-fragmentation; i.e., if W is m-fragmented then Curl27-f(W) is also m-fragmented.*

Proof. At the ith step of the Curl27-f round function, the state W is divided into words of length

$$n = 3^{6-i}, \qquad i = 1, 2, \ldots, 6.$$

For $n \le m$ the step function preserves equality of trits inside fragments. It follows from the fact that $S(a, a, a) = (b, b, b)$. For $n > m$ equality is also preserved since in each fragment trits at the different positions are processed in the same way. □

Let m be a small power of 3 (interesting cases are $m = 3$, 9, and 27). Consider a ternary string X of length

$$1 + 3 + 3^2 + \ldots + 3^{m-1} = (3^m - 1)/2.$$

The length is given by a word of m ones. Consequently, the initial state of Curl27 when processing X is m-fragmented (one fragment of 1s, the remaining fragments of 0s).

Let us choose trits of X so as to preserve m-fragmentation of the state during hashing. This is easy to do using the Theorem: Each full m-fragment of X must have the form α^m, $\alpha \in \mathbf{T}$, and, in addition, trits

of the last (incomplete) fragment must be zero to be consistent with the padding trits. Having achieved m-fragmentation of states, we automatically obtain m-fragmentation of hash values. Now a hash value is determined by $243/m$ trits each of which is repeated m times. We can find a collision for Curl27 after processing of about $\sqrt{3^{243/m}}$ strings X of the described structure, that is, in time of order

$$3^m \cdot \sqrt{3^{243/m}} = 3^{m+121.5/m}.$$

The minimum of the above function is achieved at $m = 9$. During the attack with $m = 9$ it is required to process approximately $\sqrt{3^{13.5}}$ strings of $9841 = 243 \cdot 40 + 121$ trits each.

An example of colliding messages:

$$X = 0^{243\cdot 39}(101100110101111100101100000)^{[9]}0^{121},$$
$$X' = 0^{243\cdot 39}(000011110100111111001000000)^{[9]}0^{121}.$$

This collision was found by Jeremy Jean (National Cybersecurity Agency of France), the only participant who solved the problem.

The preservation of fragmentation is an invariant of Curl27-f which allows to decrease the dimension and thereby effectively solve the basic problem. To solve the bonus problem, Jeremy Jean proposed to use another invariant for Curl27-f: If each part W_0, W_1, W_2 of the state W is 3-expanded then this fact also holds for Curl27-f(W). Here we call a word $U \in \mathbf{T}^{243}$ 3-*expanded* if it has the form $(abc)^{81}$, $abc \in \mathbf{T}^3$.

At the initial state, the parts W_0 and W_2 are indeed 3-expanded. To comply with the invariant, the part W_1 representing the length of a hashed string X must have one of the forms $(ab1)^{81}$, $(a10)^{81}$ or $(100)^{81}$ (the length is nonzero and positive). As a result, X consists of at least

$$1 + 27 + \cdots + 27^{80} > 3^{240} \text{ trits.}$$

It is easy to maintain the invariant during hashing: Full 243-fragments of X must be 3-expanded and the last incomplete fragment (if it exists) must be filled with zeros. The resulting hash values are 3-expanded, there are only 27 choices for them, and a collision will surely be found after processing only 28 strings X. Of course, the attack is impractical: The time of order 3^{240}, which is required only for recording colliding messages, is unacceptably large even compared to the time $3^{243/2}$ of the standard birthday attack.

2.15. Problem "8-Bit S-Box"

2.15.1. Formulation. Permutations S of the set $\{0,1\}^n$ or $\mathbb{F}_2^n$ are usually called *n-bit S-boxes*. We will focus on the following cryptographic properties of S-boxes:

(1) The (*minimal*) *algebraic degree* of S denoted by $\deg(S)$ is the minimum of algebraic degrees of all component functions of S.

(2) The *nonlinearity* of S denoted by $\mathrm{nl}(S)$ is the minimal Hamming distance between all component functions of S and the set of all affine functions.

(3) The *differential uniformity* of S denoted by $\mathrm{du}(S)$ is the maximal number of solutions of the equation $S(x) \oplus S(x \oplus \alpha) = \beta$ for any nonzero vector α and any vector β.

(4) The (*graph*) *algebraic immunity* of S denoted by $\mathrm{ai}(S)$ is the minimal algebraic degree of all nonzero Boolean functions f in $2n$ variables such that $f(x, y) = 0$ for all $x \in \mathbb{F}_2^n$ and $y = S(x)$.

In modern symmetric cryptography, S-boxes of dimension $n = 8$ are probably most popular. For example, such an S-box is used in the AES block cipher. The characteristics of S_{AES}:

$$(\deg, \mathrm{nl}, \mathrm{du}, \mathrm{ai})(S_{\mathrm{AES}}) = (7, 112, 4, 2).$$

The value $\mathrm{ai}(S_{\mathrm{AES}}) = 2$ means that S_{AES} (and the whole AES) can be compactly described by quadratic equations. This can be a weakness in the context of algebraic attacks.

Imposing the restrictions $(\deg, \mathrm{ai})(S) = (7, 3)$ (optimal values), we need to maximize $\mathrm{nl}(S)$ and minimize $\mathrm{du}(S)$. The current best result [17, 18] is $(\deg, \mathrm{nl}, \mathrm{du}, \mathrm{ai})(S) = (7, 108, 6, 3)$.

A problem for a special prize! You need to improve this result: Find 8-bit S with $\mathrm{nl}(S) > 108$ and/or $\mathrm{du}(S) < 6$ while preserving $\deg(S) = 7$ and $\mathrm{ai}(S) = 3$.

Remarks. Let us recall the relevant definitions:

(1) A Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ can be uniquely represented in the *algebraic normal form* (ANF) in the following way:

$$f(x) = \bigoplus_{I \in \mathcal{P}(N)} a_I \Big(\prod_{i \in I} x_i\Big),$$

where $\mathcal{P}(N)$ is the power set of $N = \{1, \dots, n\}$ and $a_I \in \mathbb{F}_2$.

(2) The *algebraic degree* of F is the degree of its ANF:

$$\deg(F) = \max\{|I| : \ a_I \neq 0, \ I \in \mathcal{P}(N)\}.$$

(3) Boolean functions of the algebraic degree at most 1 are called *affine*.

(4) The Hamming distance between Boolean functions f and g is the number of vectors $x \in \mathbb{F}_2^n$ such that $f(x) \neq g(x)$.

(5) A function $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ can be given as $S = (s_1, \dots, s_n)$, where s_i is a Boolean function; a nontrivial linear combination of $s_1, \dots, s_n$ is a *component* function of S.

2.15.2. Solution. There were no valuable ideas from the Olympiad participants. The problem remains unsolved for the considered configuration of cryptographic properties. There exist several dozen of constructions based on the well-known butterfly structure that provide current record $(7, 108, 6, 3)$, see [17, 18]. This leads to the idea that if candidates for improvement exist then they are likely outside the known structures and constructions of cryptographic permutations.

2.16. Problem "Conjecture"

2.16.1. Formulation. Let $\mathbb{F}_2$ be the finite field with two elements and let n be a positive integer at least 3. Let $f(X)$ be an irreducible polynomial of degree n over $\mathbb{F}_2$. It is known that the set of the equivalence classes β of polynomials over $\mathbb{F}_2$ modulo $f(X)$ is a finite field of order 2^n, that we denote by $\mathbb{F}_{2^n}$. It is known that different choices of the irreducible polynomial give automorphic finite fields and such choice has then no incidence on the algebraic problems on the corresponding fields.

A problem for a special prize! Prove or disprove the following

Conjecture. *Let k be co-prime with n. For every $\beta \in \mathbb{F}_{2^n}$, let $F(\beta) = \beta^\xi$, $\xi = 4^k - 2^k + 1$. Let*

$$\Delta = \{F(\beta) + F(\beta + 1) + 1; \ \beta \in \mathbb{F}_{2^n}\}.$$

For every distinct nonzero v_1 and v_2 in $\mathbb{F}_{2^n}$, we have

$$\big|\{(x, y, z) \in \Delta^3; \ v_1 x + v_2 y + (v_1 + v_2) z = 0\}\big| = 2^{2n-3}.$$

Example for $n = 3$: We can take $f(X) = X^3 + X + 1$, then each element β of the field $\mathbb{F}_{2^3}$ can be written as a polynomial of degree at most 2: $a_0 + a_1 X + a_2 X^2$, $a_0, a_1, a_2 \in \mathbb{F}_2$. The element 0 corresponds to the null polynomial; and the unity, denoted by 1, corresponds to the constant polynomial 1. We can calculate the table of multiplication in $\mathbb{F}_{2^3}$ (the table of addition just corresponds to adding polynomials of degree at most 2); this allows us to calculate any power of any element of the field and check the property.

2.16.2. Solution. This mathematical problem is open and difficult. It was presented in [19] for the first time and discussed in [20]. The conjecture was verified for small n (odd values $n \leqslant 11$, even values $n \leqslant 8$). The Olympiad participants suggested several ideas. Unfortunately, none of them gave significant advances to prove the conjecture or search for a counterexample.

The team of Kristina Geut, Sergey Titov, and Dmitry Ananichev (Ural State University of Railway Transport) and the team of Alexey Zelenetskiy, Mikhail Kudinov, and Denis Nabokov (Bauman Moscow State Technical University) proved the conjecture for a particular case $k = 1$. Nevertheless, this case is peculiar since the function is then quadratic and the result is known for quadratic functions. The proofs cannot be generalized to the common case.

3. WINNERS OF THE OLYMPIAD

Summing up the results of the Olympiad, 42 participants in the first round and 21 teams in the second round from 16 countries were awarded by prizes and honorable diplomas. Tables 10, 11, 12, 13, and 14 illustrate the information about the prize winners of NSUCRYPTO'2019.

All information about the winners can be found on the official website [21].

Table 10. Winners of the first round in School Section A ("School Student")

Place	Name	Country, City	School	Score
1	Borislav Kirilov	Bulgaria, Sofia	The First Private Mathematical Gymnasium	16
1	Alexey Lvov	Russia, Novosibirsk	Gymnasium 6	16
2	Lenart Bucar	Slovenia, Ljubljana	Gymnasium Bezigrad	15
3	Varvara Lebedinskaya	Russia, Novosibirsk	The Specialized Educational Scientific Center of Novosibirsk State University	14
3	Gabriel Ericson	Sweden, Örebro	Tullangsskolan	14

Table 11. Winners of the first round, Section B (in the category "University Student")

Place	Name	Country, City	University	Score
1	Maxim Plushkin	Russia, Moscow	Lomonosov Moscow State University	22
1	Mikhail Kudinov	Russia, Moscow	Bauman Moscow State Technical University	21
2	Narendra Patel	India, Roorkee	Indian Institute of Technology Roorkee	19
2	Vladimir Schavelev	Russia, Saint Petersburg	Saint Petersburg State University	19
3	Thanh Nguyen Van	Vietnam, Ho Chi Minh City	Ho Chi Minh City University of Technology	16
3	Daria Grebenchuk	Russia, Yaroslavl	Yaroslavl State University	16
3	Roman Gibadulin	Russia, Yaroslavl	Yaroslavl State University	16
3	Tuong Nguyen	Vietnam, Ho Chi Minh City	Ho Chi Minh City University of Technology	15

Table 12. Winners of the first round, Section B (in the category "Professional")

Place	Name	Country, City	Organization	Score
1	Henning Seidler	Germany, Berlin	TU Berlin	26
2	Samuel Tang	Hong Kong, Hong Kong	Black Bauhinia	20
2	Madalina Bolboceanu	Romania, Bucharest	Bitdefender	20
3	Irina Slonkina	Russia, Moscow	National Research Nuclear University MEPhI	16

Table 13. Winners of the second round (in the category "University Student")

Place	Name	Country, City	University	Score
1	Alexey Zelenetskiy, Mikhail Kudinov, Denis Nabokov	Russia, Moscow	Bauman Moscow State Technical University	51
2	Ngoc Ky Nguyen, Dung Truong, Phuoc Nguyen Ho Minh	Vietnam, Ho Chi Minh City; France, Paris	Ho Chi Minh City University of Technology, Ecole Normale Superieure	43
2	Thanh Nguyen Van, Quoc Bao Nguyen, Ngan Nguyen	Vietnam, Ho Chi Minh City	Ho Chi Minh City University of Technology	40
3	Maxim Plushkin	Russia, Moscow	Lomonosov Moscow State University	34
3	Ilya Trusevich, Maxim Bibik, Alexander Shulga	Belarus, Minsk	Belarusian State University	38

Table 14. Winners of the second round (in the category "Professional")

Place	Names	Country, City	Organization	Score
1	Irina Slonkina, Mikhail Sorokin, Vladimir Bobrov	Russia, Moscow	Bauman Moscow State Technical University	48
1	Kristina Geut, Sergey Titov, Dmitry Ananichev	Russia, Yekaterinburg	Ural State University of Railway Transport, Ural Federal University	46
2	Henning Seidler, Katja Stumpp	Germany, Berlin	Berlin Technical University	42
3	Victoria Vlasova, Mikhail Polyakov, Alexey Chilikov	Russia, Moscow	Bauman Moscow State Technical University	37
3	Duc Tri Nguyen, Quan Doan, Tuong Nguyen	Vietnam, Ho Chi Minh City	Cryptographic Engineering Research Group, pwnphofun, Ho Chi Minh City University of Technology	36
3	Madalina Bolboceanu, Andrei Mogage, Radu Titiu	Romania, Bucharest	Bitdefender, Alexandru Ioan Cuza University	34
Special prize	Jeremy Jean	France, Paris	National Cybersecurity Agency of France	20

FUNDING

The work of the first two authors and the sixth author was supported by the Mathematical Center in Akademgorodok under Agreement No. 075–15–2019–1613 with the Ministry of Science and Higher Education of the Russian Federation and the Laboratory of Cryptography JetBrains Research. The work of the fifth author was supported by the State Task to the Sobolev Institute of Mathematics (project no. 0314–2019–0016). The work of the seventh, eighth, and eleventh authors was supported by the

Russian Foundation for Basic Research (projects nos. 20–31–70043, 18–07–01394, and 19–31–90093).

REFERENCES

1. https://nsucrypto.nsu.ru/.
2. https://nsucrypto.nsu.ru/unsolved-problems/.
3. K. Geut, K. Kirienko, P. Sadkov, R. Taskin, and S. Titov, "On Explicit Constructions for Solving the Problem 'A Secret Sharing'," Prikl. Diskret. Mat. Pril. No. 10, 68–70 (2017).
4. S. Agievich, A. Gorodilova, N. Kolomeec, S. Nikova, B. Preneel, V. Rijmen, G. Shushuev, N. Tokareva, and V. Vitkup, "Problems, Solutions, and Experience of the First International Student's Olympiad in Cryptography," Prikl. Diskret. Mat. (Appl. Discret. Math.) No. 3, 41–62 (2015).
5. S. Agievich, A. Gorodilova, V. Idrisova, N. Kolomeec, G. Shushuev, and N. Tokareva, "Mathematical Problems of the Second International Student's Olympiad in Cryptography," Cryptologia **41** (6), 534–565 (2017).
6. N. Tokareva, A. Gorodilova, S. Agievich, V. Idrisova, N. Kolomeec, A. Kutsenko, A. Oblaukhov, and G. Shushuev, "Mathematical Methods in Solutions of the Problems from the Third International Students' Olympiad in Cryptography," Prikl. Diskret. Mat. (Appl. Discret. Math.) No. 40, 34–58 (2018).
7. A. Gorodilova, S. Agievich, C. Carlet, E. Gorkunov, V. Idrisova, N. Kolomeec, A. Kutsenko, S. Nikova, A. Oblaukhov, S. Picek, B. Preneel, V. Rijmen, and N. Tokareva, "Problems and Solutions of the Fourth International Students Olympiad in Cryptography (NSUCRYPTO)," Cryptologia **43** (2), 138–174 (2019).
8. A. Gorodilova, S. Agievich, C. Carlet, X. Hou, V. Idrisova, N. Kolomeec, A. Kutsenko, L. Mariot, A. Oblaukhov, S. Picek, B. Preneel, R. Rosie, and N. Tokareva, "The Fifth International Students' Olympiad in Cryptography—NSUCRYPTO: Problems and Their Solutions," Cryptologia **44** (3), 223–256 (2020).
9. B. Schneier, *Applied Cryptography: Protocols, Algorithms and Source Code in C*, 2nd Ed. (Wiley, Indianapolis, 1996).
10. R. E. Lewand, *Cryptological Mathematics* (MAA, Washington, 2000).
11. "Letter Frequency," in *Wikipedia*. Available at https://en.wikipedia.org/wiki/Letter_frequency.
12. "Find Words Using Pattern Matching," in *Litscape.com*. Available at http://www.litscape.com/word_tools/pattern_match.php.
13. M. Brinkmann and G. Leander, "On the Classification of APN Functions up to Dimension Five," Designs, codes and cryptography **49**, 273–288 (2008).
14. C. De Canni‘ere, *Analysis and Design of Symmetric Encrytption Algorithms*, Ph.D. Thesis (Katholieke Universiteit Leuven, Heverlee, 2007).
15. https://nsucrypto.nsu.ru/archive/2019/round/2/task/4/.
16. https://nsucrypto.nsu.ru/media/Olympiads/2019/Round_2/Tasks/curl27.java.
17. R. A. de la Cruz Jimènez, "Generation of 8-Bit S-Boxes Having Almost Optimal Cryptographic Properties Using Smaller 4-Bit S-Boxes and Finite Field Multiplication," in *Progress in Cryptology—LATINCRYPT 2017: 5th International Conference on Cryptology and Information Security in Latin America (Havana, Cuba, September 20–22, 2017): Revised Selected Papers*, Ed. by T. Lange and O. Dunkelman (Springer, Cham, 2019), pp. 191–206 [*Lecture Notes in Computer Science*, Vol. 11368].
18. D. B. Fomin, "New Classes of 8-Bit Permutations Based on a Butterfly Structure," Mat. Vopr. Kript. **10** (2), 169–180 (2019). https://ctcrypt.ru/files/files/2018/09_Fomin.pdf.
19. C. Carlet, "Componentwise APNness, Walsh Uniformity of APN Functions, and Cyclic-Additive Difference Sets," Finite Fields and Their Applications **53**, 226–253 (2018).
20. C. Carlet, "On APN Exponents, Characterizations of Differentially Uniform Functions by the Walsh Transform, and Related Cyclic-Difference-Set-Like Structures," in *Proceedings of WCC 2017* Designs, Codes and Cryptography **87** (2), 203–224 (2018).
21. https://nsucrypto.nsu.ru/archive/2019/total_results/#data.