A Sage Library For Analysis Of Nonlinear Binary Mappings

Anna Maria Eilertsen, Oleksandr Kazymyrov, Valentyna Kazymyrova, Maxim Storetvedt

Department of Informatics
University of Bergen, Norway
{Anna.Eilertsen,Valentyna.Kazymyrova,Maksim.Storetvedt}@student.uib.no
Oleksandr.Kazymyrov@ii.uib.no

Abstract. Many new ciphers are being introduced every year. Each well-educated researcher with a degree in computer science can create a good cryptoprimitive, which will be unbreakable awhile. The main blocks of modern ciphers are nonlinear components also known as substitutions. The toolbox (library) for decreasing time of analyzing such components is given in this paper. It can be used for investigation cryptographic properties of arbitrary nonlinear binary mappings used in symmetric primitives.
Keywords: substitution, Sage, cryptanalysis, cryptographic properties

1 Introduction

Today's information society has the conventional wisdom to enhance the security of information systems. A significant role in this process has the implementation of cryptographic primitives that provide the required level of cryptographic security. Enough time has passed since cryptography has become public science. As a result, in many areas ciphers have started to be used for protection of sensitive information.

Each year several new cryptoprimitives are issued [1,2]. Many designers justify the resistance of the developed algorithms to known attacks, such as differential, linear or algebraic. Meanwhile, an independent analysis requires tools to analyze both basic components and entire encryption algorithms. On the other hand, if you are a developer of a perspective algorithm, there is a need to analyze basic components as well. Choosing linear layers is a relatively simple task when only few indicators are considered [3]. The situation is completely opposite for nonlinear layers which are usually presented as parallel application of substitutions (S-boxes). Effective and simultaneous calculation of cryptographic properties of S-boxes is a nontrivial task.

Analysis of current state shows today there is a number of tools that can be considered as a partial solution of the problem [4,5,6,7,8]. However, the cryptographic community needs a universal approach to calculate cryptographic indicators for arbitrary binary mappings. In this paper we propose a tool for generating and analyzing arbitrary vectorial Boolean functions $F : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$.

The rest of the paper is organized as follows. Section 2 describes the theoretical background of substitution's criteria. Section 3 introduces general overview of the "Sbox" library and explains the main components and methods. Section 4 includes practical applications of the proposed library. Finally, Section 5 presents our conclusions.

2 Preliminaries

In this section we present theoretical aspects of representation and construction of vectorial Boolean functions. The relevant properties used in symmetric primitives are also considered in this section. All definitions and indicators are well-known and one can see [9] for more details.

2.1 Definitions and notations

Let n and m be two positive integers. Define $\mathbb{F}_2^n$ as a vector space of all binary vectors of length n, where $\mathbb{F}_2$ is the Galois field with elements $\{0,1\}$. Then (n,m)-function is a vectorial Boolean function $F : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$. Boolean functions $f_1, f_2, \ldots, f_m$, such that $F(x_1,\ldots,x_n) = (f_1(x_1,\ldots,x_n),\ldots,f_m(x_1,\ldots,x_n))$, and their linear combination are called coordinate and component functions of F, respectively. If $m = 1$ then a vectorial Boolean function has a single output bit and is equivalent to a Boolean function. To find algebraic properties of (n,m)-functions, a vector space often has a structure of finite field $\mathbb{F}_{2^n}$.

2.2 Cryptographic properties of Boolean functions

Suppose $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$ is a Boolean function of n variables. Algebraic normal form of such function is defined as

$$f(x_1, x_2, \ldots, x_n) = \sum_{I \in \mathbb{P}(1,\ldots,n)} \left(\prod_{i \in I} x_i \right),$$

where $\mathbb{P}(z)$ denotes the power set of z. The algebraic degree of f ($deg(f)$) is the maximum degree of the monomial with nonzero coefficient.

A Boolean function of n variables is called balanced if $hw(f) = 2^{n-1}$, where $hw(f) = \sum_{x=0}^{2^n-1} f(x)$. The correlation between arbitrary Boolean function $f(x)$, where $x = (x_1, x_2, \ldots, x_n)$, and the set of all linear functions is determined by Walsh transformation

$$W(w) = \sum_{x=0}^{2^n-1} (-1)^{f(x) \oplus l_w(x)},$$

where $l_w(x) = w \cdot x = w_1x_1 \oplus w_2x_2 \oplus \ldots w_nx_n$. The nonlinearity is related to the Walsh values as

$$NL(f) = \frac{1}{2}\left(2^n - \max_{\forall w, w \neq 0} |W(w)|\right).$$

Autocorrelation of f ($r_f(\alpha)$) shows how the function differs from itself shifted on several positions, i.e.

$$r_f(\alpha) = \sum_{x=0}^{2^n-1} (-1)^{f(x) \oplus f(x \oplus \alpha)},$$

where $\alpha \in \mathbb{F}_2^n$. For cryptography the maximal value of the function $r_f(\alpha)$ is of interest, which can be found as

$$AC_{max}(f) = \max_{\forall \alpha, \alpha \neq 0} |r_f(\alpha)|.$$

Let σ be the sum-of-square indicator, then

$$\sigma = \sum_{\alpha=0}^{2^n-1} r_f^2(\alpha).$$

Let $hw(\alpha)$ be a binary Hamming weight of $\alpha \in \mathbb{F}_2^n$ [9]. Then we say that $f(x)$ satisfies propagation criterion of order k ($PC(k)$) if and only if for all nonzero vectors $\alpha \in \mathbb{F}_2^n$ the following system is true

$$\begin{cases} 1 \le hw(\alpha) \le k; \\ \sum\limits_{x=0}^{2^n-1} f(x) \oplus f(x \oplus \alpha) = 2^{n-1}. \end{cases}$$

The strict avalanche criterion (SAC) corresponds to $PC(1)$.

A Boolean function is correlation immune of order m ($CI(m)$) if the system of equations

$$\begin{cases} 1 \le hw(w) \le m; \\ W(w) = 0. \end{cases}$$

is valid for all $w \in \mathbb{F}_2^n$. If the function is balanced and satisfy $CI(m)$ simultaneously, then such function is called m-resilient.

To prevent some misunderstanding we present the indicator called algebraic immunity of a Boolean function. The minimum algebraic degree of $g(x) \neq 0$ of the set $\{g \mid f(x) \cdot g(x) = 0\}$ $\cup \{g \mid (f(x) \oplus 1) \cdot g(x) = 0\}$ is called algebraic immunity (AI) of f.

2.3 Cryptographic properties of substitutions

While Boolean functions are adopted mainly as filtering functions in stream ciphers, vectorial Boolean function are used in block ciphers and hash functions as substitutions. For theoretical analysis the univariate representation is one of the best ways to consider cryptographic properties of the binary mappings. However, field operations are not good optimized in modern computers as operations with Boolean functions, especially for large n. Therefore, the representation of (n, m)-functions as the set of component functions is a better way for practical implementations.

Suppose substitution S is a table representation of a vectorial Boolean function $F = (f_1, \ldots, f_m)$ from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. Define $\{h_j = j \cdot F \mid 0 < j < 2^m\}$ as the set of component functions of F. Then

- nonlinearity of S is

$$NL(S) = \min_{0<j<2^m} (NL(h_j));$$

- minimum degree of S is

$$deg(S) = \min_{0<j<2^m} (deg(h_j));$$

- the maximum value of autocorrelation spectrum of S is

$$AC_{max}(S) = \max_{0<j<2^m} (AC_{max}(h_j));$$

- S satisfies strict avalanche criterion if every h_j satisfies SAC;
- S satisfies propagation criterion of order k if every h_j satisfies $PC(k)$;
- S is correlation immune of order k if every h_j is $CI(k)$;
- S is balanced (permutation) if every h_j is balanced;

– S is k-resilient if every h_j is k-resilient.

The similar properties for vectorial Boolean functions are given in [10].

While the maximum value of the approximation table (λ) can be calculated directly from the nonlinearity of the S-box as $\lambda = 2^{n-1} - NL(S)$, the maximum value of differential table (MDT) cannot be easily evaluated from the component functions. Let δ be the maximum number of times when the input difference maps to the output difference of the given S-box. Then

$$\delta = \max_{\alpha \in \mathbb{F}_2^n, \alpha \neq 0, \beta \in \mathbb{F}_2^m} \#\{x \mid S(x) \oplus S(x \oplus \alpha) = \beta\}.$$

This indicator is also know as δ-uniformity [10].

The ways to represent a substitution as a system of equations over $\mathbb{F}_2$ are given in [11,12]. Define density as the fraction of nonzero elements in a system of equations. Then, a substitution provides better protection against algebraic attacks if the system

– has higher degree;
– has fewer equations;
– is more dense.

Unambiguous theoretical relation between these parameters is an unsolved problem [12]. When we are talking about the algebraic immunity of an S-box ($AI(S)$), then we mean the smallest degree of the system.

2.4 Equivalence of vectorial Boolean functions

Two functions $F, G : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$ are called extended affine (EA) equivalent if there exist such affine permutations $A_1 = L_1(x) + c_1, A_2 = L_2(x) + c_2$ and arbitrary linear function $L_3(x)$ that

$$F(x) = A_1 \circ G \circ A_2(x) + L_3(x).$$

If $L_3(x) = const$, or $L_3(x) = 0, c_1 = 0$, and $c_2 = 0$ then F and G are affine, or linear equivalent, respectively. Moreover, for at least one missing element of $L_1(x), L_2(x), L_3(x), c_1, c_2$ the functions are called restricted EA (REA) equivalent [13].

In [14] F and G are considered as $G_F(x, y) = \{\{x, y\} \mid y = F(x)\}$. They are Carlet-Charpin-Zinoviev (CCZ) equivalent, if for $F_2(x) = L_3(x) + L_4 \circ G(x)$ and permutation $F_1(x) = L_1(x) + L_2 \circ G(x)$ the following equation is hold

$$F(x) = F_2 \circ F_1^{-1}(x),$$

where $L_1(x)$, $L_2(x)$, $L_3(x)$, $L_4(x)$ are arbitrary linear functions.

Both CCZ- and EA-equivalence preserve extended Walsh spectrum and δ-uniformity. However, the minimum degree remains the same only for the EA-equivalent functions. [10].

3 Design specification and main components

In this section we describe basic components of the "Sbox" library, and some methods for both generation and cryptanalysis. The source code is available at github and distributed under GPL v2 [15].

3.1 General overview of the library

There are many ready-made solutions [4,5,8]. However, all of them have certain limitations. For example, the class SBox from the package mq in Sage is optimized only for small values [4]. By increasing n, functions do not return the expected results, i.e. the absence of the system of equations of degree 2 for the AES substitution [16]. Most of the other programs or libraries are designed to work with a limited number of properties and/or just with Boolean functions. As a consequence, software to analyze arbitrary vectorial Boolean functions was developed taking into account publicly available optimized algorithms.

The proposed implementation is written as an extension to Sage and presented as a package with the main class "Sbox" [15]. Fig. 1 depicts the general overview of the library. It consists of three main parts: Sbox, CSbox and GSbox. While most cryptanalytic functions are presented in CSbox and connected to C/C++ code via Cython, GSbox includes methods for generation nonlinear binary mappings. Using Python as the main language is beneficial considering that the language is somewhat easy to learn and use. It does in general a slower run time, but this disadvantage may be offset by Cython and C/C++, making extensive mathematical calculations faster.

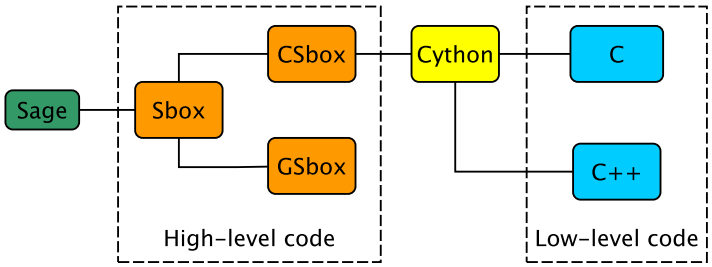

Fig. 1: The general overview of "Sbox"

It should also be noted that some methods use integrated in Sage packages (i.e. finite_rings) and cannot be transfered into other software. However, the C/C++ code is an independent implementation and one can easily port it to own software.

Performance comparison with other tools. The performance comparison of software, even written in one language, is a difficult task. It depends on many parameters including operation system, compilations, memory and processors. We have not created best of the best tool, but we took into account a number of articles with optimized algorithms for calculation of cryptographic indicators [17,18]. Hence, we will present time comparisons as the most critical indicator of the proposed implementation in Section 4.

3.2 Generation of vectorial Boolean functions

All functions that belong to this group start with "gen_" and defined in "GSbox.sage". Most of them are based on theoretical methods described in [10,19]. The correspondence between methods implemented in the library and well-known names are presented in Table 1.

The library has also methods to find CCZ-(CCZ) and EA-equivalence (EA). They do not allow to find vectorial Boolean functions with certain characteristics by themselves, however, they play very important roles in cryptography. For completeness and additional experiments

Table 1: Correspondence of vectorial Boolean functions to methods defined in "Sbox.sage"

Vectorial Boolean functions	Correspondent methods
Gold	gold
Kasami	kasami
Welch	welch
Niho	niho
Inverse	inverse
Dobbertin	dobbertin
Dicson	dicson
APN for $n = 6$	APN6
Optimal permutation polynomials for $n = 4$	OP4

the "Sbox" class includes methods for generating random substitutions and permutations that are specified by "random_substitution" and "random_permutation", respectively.

In order to unify different cases a single method "generate_sbox" was created, which has parameters "method" to specify generation method and "T" to define equivalence. The optional parameter "G" is used for setting a user-defined polynomial in the case of "method=polynomial".

3.3 Cryptanalysis of vectorial Boolean functions

This group contains methods starting with "cr_", which are described in "CSbox.sage". Most of the methods are also based on theoretical algorithms [10]. Nevertheless, the calculation of some indicators has been optimized, i.e. calculation of the algebraic immunity or cycles. Table 2 shows the correspondence between the indicators described in Section 2, and the methods from the "Sbox" library.

Table 2: Correspondence between cryptographic properties and methods in the library

Indicators	Methods in "Sbox"
Balanceness	balanced
Nonlinearity	nonlinearity
Absolute indicator	autocorrelation
Propagation criterion	PC
Correlation immunity	CI
Sum-of-square indicator	SSI
Minimum degree	minimum_degree
Resilience	resilient
Strict avalanche criterion	SAC
Bijection	is_bijection
Maximum of differential table	MDT
Maximum of linear approximation table	MLT
Cycles	cycles
Algebraic immunity	algebraic_immunity_sbox

To investigate the number of cryptographic properties in some applications, such as generation of pseudo-random sequences (i.e. using OFB mode), it is necessary to study the period lengths of sequences obtained at the output of the block cipher. For these purposes, the "cycles" method was implemented.

This group also contains a number of auxiliary functions, such as finding the univariate polynomial or the system of equations describing the substitution; checking APN, or CCZ-

equivalence properties [20]. Some examples of the use of different methods are given in Section 4.

3.4 Other useful methods

The last group of methods is optional and is used as an extension of functionality. Table 3 contains the most important methods and their short description. As it can be seen from the table, the methods' names define their functional purpose.

Table 3: Optional methods of "Sbox'

Methods	Comments
get_field	return the filed $\mathbb{F}_{2^{\max\{n,m\}}}$
get_ring	return the ring $\mathbb{F}_{2^{\max\{n,m\}}}[x]$
get_mg	return a multiplicative generator of the field
get_sbox	return the substitution
get_polynomial	return the univariate polynomial of the substitution
set_sbox	set a substitution for analysis
Tr_pol	return a trace of input polynomial
g2p	transform a given polynomial with multiplicative element to the internal representation
p2g	inverse method to g2p

4 Application of the "Sbox" library

To give an example of the library usage, we present the main steps of generation of the APN permutation for $n = 6$. First let us give a mathematical definition of such function [20].

Theorem 1. *Let α be a multiplicative generator of $\mathbb{F}_{2^6}$ with irreducible polynomial $f(x) = x^6 + x^4 + x^3 + x + 1$. Then the APN function*

$$F(x) = \alpha x^3 + \alpha^5 x^{10} + \alpha^4 x^{24}$$

is CCZ-equivalent to an APN permutation over $\mathbb{F}_{2^6}$.

For example, for the linear function

$$\mathcal{L}(x, y) = (tr_{6/3}(\alpha^4 x) + \alpha tr_{6/3}(y), tr_{6/3}(\alpha x) + \alpha tr_{6/3}(\alpha^4 y)),$$

where $tr_{6/3} = x + x^{2^3}$, $y = F(x)$, the function $G_H = \mathcal{L}(G_F)$ is the APN permutation.

Using the above description one can easily start repeating the following example in Sage. First we need to identify the main variables, including the ring $\mathbb{F}_{2^6}[x]$ (P), a multiplicative generator (g) of $\mathbb{F}_{2^6}$, and the trace (tr).

```
sage: %runfile ./Sbox.sage
sage: S = Sbox(n=6,m=6)
sage: P = S.get_ring()
sage: g = S.get_mg()
a
sage: tr = S.Tr_pol(x=P("x"),n=6,m=3)
sage: tr
x^8 + x
```

Next it is necessary to specify linear functions and convert them from polynomial representations to matrix forms [13].

```
sage: M1 = S.l2m(tr.subs(P("(%s)*x"%(g^4))))
sage: M2 = S.l2m(g*tr)
sage: M3 = S.l2m(tr.subs(P("(%s)*x"%(g))))
sage: M4 = S.l2m(g*tr.subs(P("(%s)*x"%(g^4))))
```

In the end, we apply CCZ-equivalence to the function F and check on APN properties.

```
sage: F = "g*x^3+g^5*x^10+g^4*x^24"
sage: S.generate_sbox(method="polynomial",G=F,T="CCZ",M1=M1,M2=M2,M3=
    ↪ M3,M4=M4)
sage: S.is_bijection()
True
sage: S.is_APN()
True
sage: S.MDT()
2
```

The same result can be achieved by the following commands.

```
sage: S = Sbox(n=6,m=6)
sage: S.generate_sbox(method='APN6')
sage: S.is_bijection()
True
sage: S.is_APN()
True
```

The above example shows, that hundred of strings of other libraries can be replaced by a few lines of the proposed library to achieve the same functionality. On the other hand, the performance needs to be at a high level. Fig. 2 shows the time complexity of several frequently used methods for $n = m$.

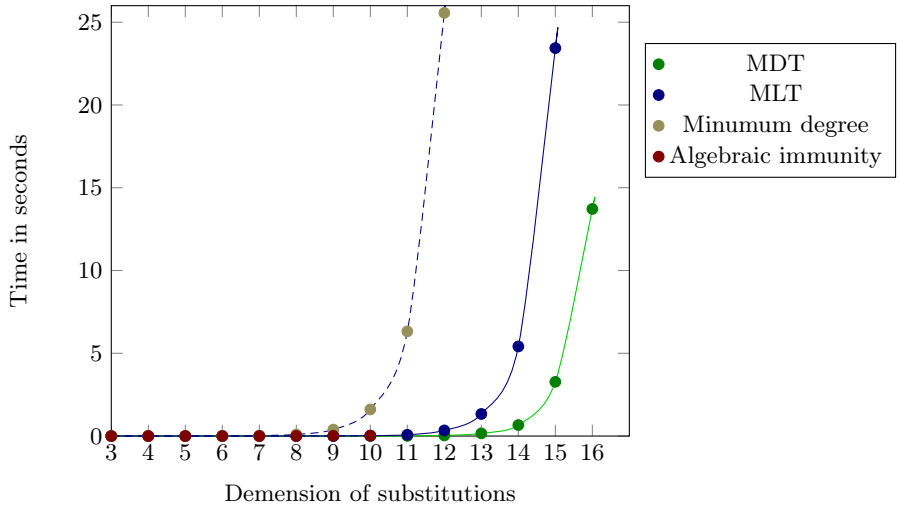

Fig. 2: The relationship between the dimension of random substitutions and time of calculation

All results are presented as the average values of 1000 runs on Macbook Pro Retina Mid 2012 [21]. One can easily notice that the figure doesn't show the results for every n. This is due to the fact that every function has limitations either in memory or in time. For example, the time needed for counting the value of algebraic immunity is negligible, while the memory is growing very fast. Consequently, every function has the maximum values of n or m after which the indicators become difficult to calculate on an ordinary computer.

Other examples for different values of n and m is given are Table 4. One can see that the library can work correctly with any dimension, but in the same time has some limitations.

Table 4: Performance of different methods from the "Sbox" library given in ms

Properties	(n,m)-functions								
	(6,10)	(8,1)	(8,4)	(10,2)	(10,8)	(3,12)	(12,3)	(12,6)	(12,9)
MDT	0.083	0.17	0.17	2.34	2.34	0.037	37.15	37.15	38.02
MLT	1.23	0.068	0.126	0.114	4.9	0.501	0.708	5.25	41.69
Algebraic immunity	1	8.91	2.24	6.17	22.4	-	39.8	75.9	-
Minimum degree	89.64	0.35	4.77	3.92	329.3	48.67	35.78	320.89	2631
Balancedness	0.102	0.331	0.338	1.26	1.32	0.025	5.13	5.13	5.01
$AC_{max}(S)$	5.89	109.6	110.6	2951.2	2818.4	0.1	112202	117490	114815
Interpolation polynomial	112.2	457.1	457.1	7586	14454	11.74	281838	288403	275422
Cycles	0.263	1.78	2.69	8.13	25.7	0.074	41.69	72.45	158.49
PC	6.03	89.1	89.1	1446	1446	0.098	23442	23442	24547

From a practical point of view the "Sbox" library can be used to analyze nonlinear components of the existing cryptographic primitives. The substitution comparison of AES, GOST R 34.11-2012, STB 34.101.31-2011, "Kalyna" ($S0$) [22], and a substitution proposed in [23] is given in Table 5.

Table 5: Comparison of 8-bit S-boxes

Properties	AES	GOST R 34.11-2012	STB 34.101.31-2011	Kalyna's S0	Proposed [23]
MDT	4	8	8	8	8
MLT	16	28	26	32	24
Absolute indicator	32	96	80	88	80
SSI	133120	258688	232960	244480	194944
Minimum degree	7	7	6	7	7
Algebraic immunity	2	3	3	3	3

5 Conclusions

It has been indicated quite clear that practical realization of theoretically proved results is time and resource consuming in most cases. Conducted analysis has shown that no sufficiently effective tools of finding characteristics of arbitrary substitutions exist to date.

The "Sbox" library implementation allows to solve many problems related to cryptanalysis. It includes lots of known generation methods and functions for computing permutations' properties. Moreover, the library is designed in the way that gives an opportunity to extent its functionality quite easily.

References

1. Canetti, R., and Garay, J. A., Eds. Advances in Cryptology – CRYPTO 2013 (2013), vol. 8042, Springer Berlin Heidelberg.
2. International Association for Cryptologic Research. Fast Software Encryption 2014, March 2014. http://fse2014.isg.rhul.ac.uk/.
3. Augot, D., and Finiasz, M. Direct construction of recursive MDS diffusion layers using shortened BCH codes. In FSE 2014 (2014).

4. Stein, W., et al. Sage Mathematics Software (Version 6.1.1). The Sage Development Team, 2014. http://www.sagemath.org.
5. Biryukov, A., De Canniere, C., Braeken, A., and Preneel, B. A toolbox for cryptanalysis: Linear and affine equivalence algorithms. In Advances in Cryptology – EUROCRYPT 2003. Springer, 2003, pp. 33–50.
6. Burnett, L. D. Heuristic Optimization of Boolean Functions and Substitution Boxes for Cryptography. PhD thesis, Queensland University of Technology, 2005.
7. Albrecht, M. Alorithmic Algebraic Techniques and Their Application to Block Cipher Cryptanalysis. PhD thesis, Royal Holloway, University of London, 2010.
8. Lafitte, F., Van Heule, D., et al. Cryptographic boolean functions with R. R Journal 3, 1 (2011).
9. Carlet, C. Boolean functions for cryptography and error correcting codes. Boolean Models and Methods in Mathematics, Computer Science, and Engineering 2 (2010), 257.
10. Carlet, C. Vectorial boolean functions for cryptography. Boolean Models and Methods in Mathematics, Computer Science, and Engineering 134 (2010), 398–469.
11. Kleiman, E. High performance computing techniques for attacking reduced version of AES using XL and XSL methods. PhD thesis, Iowa State University, 2010.
12. Kazymyrov, O., and Oliynykov, R. Choosing substitutions for symmetric cryptographic algorithms based on the analysis of the algebraic properties. In Mathematical modeling. Information Technology. Automated control systems., vol. 925. Kharkiv National University V.N. Karazina, 2010, pp. 79–86 (In Russian).
13. Budaghyan, L., Kazymyrov, O.: Verification of restricted EA-equivalence for vectorial Boolean functions. In Özbudak, F., Rodríguez-Henríquez, F. (eds.), Arithmetic of Finite Fields, vol. 7369 of Lecture Notes in Computer Science, pp. 108–118. Springer Berlin Heidelberg, 2012.
14. Carlet, C., Charpin, P., and Zinoviev, V. Codes, bent functions and permutations suitable for DES-like cryptosystems. Designs, Codes and Cryptography 15, 2 (1998), 125–156.
15. Kazymyrov, O. Source code of the "Sbox" library. https://github.com/okazymyrov/sbox, 2014.
16. Courtois, N. T., and Pieprzyk, J. Cryptanalysis of block ciphers with overdefined systems of equations. In Advances in Cryptology – ASIACRYPT 2002. Springer, 2002, pp. 267–287.
17. Mishra, P. R. Calculating cryptographic degree of an S-box. Cryptology ePrint Archive, Report 2014/145, 2014. http://eprint.iacr.org/.
18. Chmiel, K. Fast computation of approximation tables. In Information Processing and Security Systems. Springer, 2005, pp. 125–134.
19. Kazymyrov, O., and Oliynykov, R. Application of vectorial Boolean functions for substitutions generation used in symmetric cryptographic transformation. In Systems of information processing, vol. 6 (104). Kharkiv: KhUAF, 2012, pp. 97–102 (In Russian).
20. Browning, K., Dillon, J., McQuistan, M., and Wolfe, A. An APN permutation in dimension six. Finite Fields: Theory and Applications-FQ9, ser. Contemporary Mathematics 518 (2010), 33–42.
21. 15-inch macbook pro with retina display: 2.3GHz, 8GB of 1600MHz DDR3L, 2012. http://support.apple.com/kb/sp653.
22. Oliynykov, R., Gorbenko, I., Dolgov, V., Ruzhentsev, V.: Results of ukrainian national public cryptographic competition. Tatra Mountains Mathematical Publications 47(1), 99–113, 2010.
23. Kazymyrov, O., Kazymyrova, V., and Oliynykov, R. Method for generation of high-nonlinear S-boxes based on gradient descent. In Second workshop on Current Trends in Cryptology (CTCrypt 2013). Ekaterenburg, Russian, 2013, pp. 107–115.