ISSN 2320 - 2602

Volume 7 No.6, June 2018
International Journal of Advances in Computer Science and Technology
Available Online at http://www.warse.org/IJACST/static/pdf/file/ijacst03762018.pdf
https://doi.org/10.30534/ijacst/2018/03762018

About the algorithm of data encryption BTS

Sattarov Alijon Bozorboyevich
National University of Uzbekistan, Uzbekistan, asb2602@mail.ru

ABSTRACT

In this article presents data encryption algorithm BTS, based on SP network. The length of the created algorithm block is 128, 256, 384 and 512 bit, and also a number of its rounds is equal to 8, 10, 12, and 14 respectively. In the algorithm named B, T, S and K transformation are used.

Key words: cryptography, decryption, encryption, key, round function, round, symmetrical block algorithm.

1. INTRODUCTION

Information systems need to use cryptographic encryption algorithms to ensure confidentiality of processed data. According to Shannon's work, the following requirements are for modern block-symmetric encryption algorithm (BSEA).

- diffusion – changing of one character open text leads to change in the almost all ciphertext;
- confusion – mixing information is used to complicate the relation between ciphertext and key.

Nowadays most of the BSEA have "substitute" and "permutation" transforms which are used sequent and repeatedly in order to ensure above requirements. However, there are very few ciphers that satisfy the properties of "stability", "simplicity" and "speed". In general, these concepts are relative. Therefore, the research in the direction of creating new encryption algorithms with higher cryptographic performance than existing ones is one of the main tasks of cryptography.

The following is a process of encrypting and decrypting the BTS algorithm that has been developed as a result of a research on the creation of a new modern cipher.

2. BTS DATA ENCRYPTION ALGORITHM

BTS (Bardoshli, Tezkor, Sodda – Stable, Fast, Simple) – block symmetric encryption algorithm is based on the SP network, the block length (key) is 128, 256, 384, and 512 bits. The encryption process uses B, T, S and K transformations, and the decryption procedure uses invB, T, invS and invK transformations.

The algorithm encryption and decryption processes are shown in Figure 1.

Transformations of the BTS algorithm, if the block length is 256 bits, are performed as follows.

K (from the word "Kalit" (key)) – performs the addition of the round keys modulo 2^{32}. For encryption and decryption require 88 keys (k_i), each of which is 32 bits long. An input 256 bit X-block is transformed as follows:

Round-0:
$K(X)=(x_0[+]k_0)||(x_1[+]k_1)||(x_2[+]k_2)||\ldots||(x_7[+]k_7)$
Round-1:
$K(X)=(x_0[+]k_8)||(x_1[+]k_9)||(x_2[+]k_{10})||\ldots||(x_7[+]k_{15})$
Round-2:
$K(X)=(x_0[+]k_{16})||(x_1[+]k_{17})||(x_2[+]k_{18})||\ldots||(x_7[+]k_{23})$
. . . . . .
Round-10:
$K(X)=(x_0[+]k_{80})||(x_1[+]k_{81})||(x_2[+]k_{82})||\ldots||(x_7[+]k_{87})$

where: [+] – is the addition operation modulo 2^{32}, $X=x_0||x_1||x_2||x_3||x_4||x_5||x_6||x_7$.

invK – is carried out as follows:

Round-0:
$invK(Y)=(y_0[-]k_{80})||(y_1[-]k_{81})||(y_2[-]k_{82})||\ldots||(y_7[-]k_{87})$
Round-1:
$invK(Y)=(y_0[-]k_{72})||(y_1[-]k_{73})||(y_2[-]k_{74})||\ldots||(y_7[-]k_{79})$
Round-2:
$invK(Y)=(y_0[-]k_{64})||(y_1[-]k_{65})||(y_2[-]k_{66})||\ldots||(y_7[-]k_{71})$
. . . . . .
Round-10:
$invK(Y)=(y_0[-]k_0)||(y_1[-]k_1)||(y_2[-]k_2)||\ldots||(y_7[-]k_7)$

where: [–] – is the subtraction operation modulo 2^{32}, $Y=y_0||y_1||y_2||y_3||y_4||y_5||y_6||y_7$.

B (from the word "Bardoshli" (stable)) – is a non-linear transformation which performs the substitution of bytes by block S_1 and S_2. That is, an input 256-bit X-block is transformed as follows:

$$B(X)=S_1(x_0)||S_2(x_1)||S_1(x_2)||S_2(x_3)||S_1(x_4)||\ldots||S_2(x_{31})$$

where: $X=x_0||x_1||x_2||\ldots||x_{29}||x_{30}||x_{31}$.

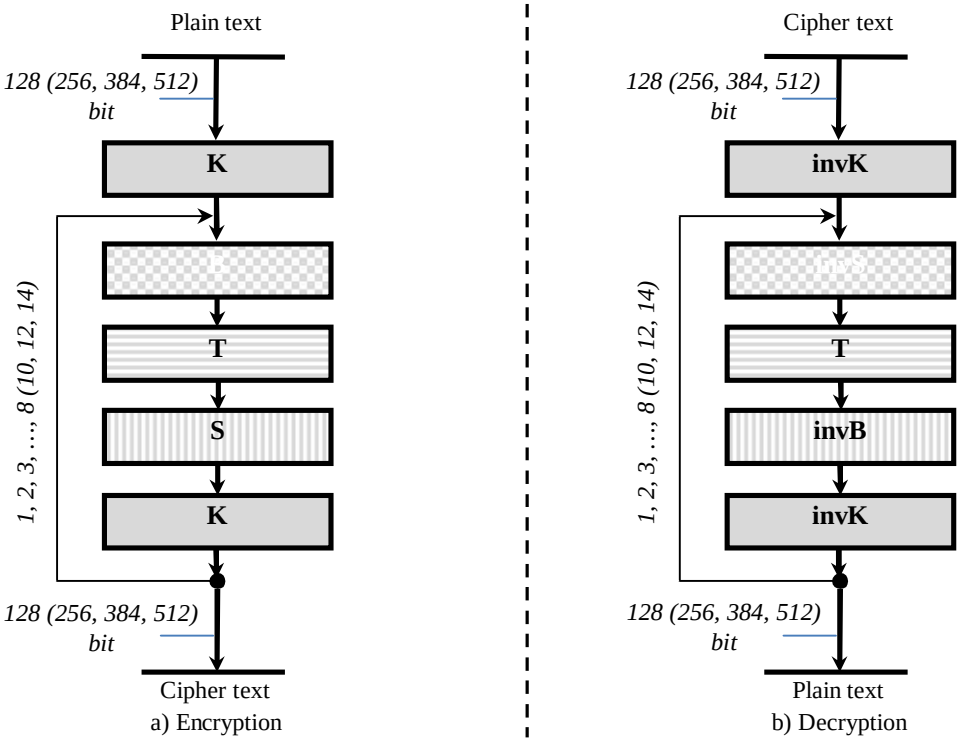

Figure 1: BTS symmetric encryption algorithm

S_1={37, 39, 220, 117, 216, 187, 140, 180, 90, 84, 18, 246, 113, 168, 108, 174, 227, 13, 29, 105, 70, 20, 181, 160, 14, 68, 226, 91, 0, 107, 153, 192, 62, 120, 177, 92, 56, 172, 2, 31, 109, 100, 61, 170, 236, 161, 103, 60, 200, 47, 148, 146, 190, 79, 99, 10, 54, 88, 251, 55, 250, 72, 86, 52, 80, 123, 11, 183, 110, 58, 152, 175, 43, 142, 96, 94, 207, 132, 193, 239, 128, 143, 133, 141, 41, 182, 27, 59, 64, 144, 231, 209, 252, 36, 169, 173, 211, 145, 89, 17, 125, 163, 6, 217, 16, 51, 104, 158, 126, 65, 74, 93, 213, 131, 26, 155, 50, 208, 85, 247, 178, 28, 147, 229, 228, 240, 44, 185, 159, 254, 114, 151, 202, 71, 21, 198, 249, 83, 210, 204, 122, 101, 25, 139, 219, 233, 136, 232, 7, 225, 97, 75, 40, 196, 244, 49, 214, 176, 57, 137, 118, 111, 8, 73, 116, 102, 241, 156, 34, 222, 149, 33, 67, 253, 82, 167, 22, 12, 255, 124, 188, 78, 95, 248, 201, 245, 165, 121, 98, 179, 224, 19, 166, 164, 127, 238, 154, 171, 63, 115, 9, 162, 30, 212, 77, 76, 218, 46, 191, 205, 215, 199, 130, 135, 1, 234, 112, 35, 150, 66, 106, 243, 24, 194, 221, 242, 15, 5, 195, 186, 3, 138, 87, 203, 223, 230, 157, 38, 53, 81, 23, 42, 184, 237, 134, 189, 69, 45, 197, 129, 206, 48, 32, 119, 235, 4};

S_2={148, 151, 37, 76, 201, 220, 172, 74, 190, 188, 181, 232, 160, 95, 173, 66, 49, 91, 133, 249, 129, 250, 193, 211, 138, 71, 68, 140, 136, 42, 135, 41, 199, 39, 253, 93, 153, 54, 166, 255, 159, 189, 23, 219, 191, 156, 3, 243, 47, 89, 77, 214, 252, 176, 157, 92, 155, 174, 127, 186, 152, 175, 206, 104, 13, 27, 142, 119, 78, 40, 204, 115, 150, 202, 112, 48, 56, 108, 20, 247, 36, 180, 128, 26, 101, 117, 2, 12, 52, 11, 149, 223, 107, 168, 19, 86, 124, 170, 246, 251, 248, 241, 0, 187, 165, 179, 130, 134, 144, 239, 245, 242, 34, 105, 57, 5, 84, 122, 21, 75, 147, 254, 60, 82, 9, 113, 235, 53, 216, 234, 97, 80, 224, 72, 85, 200, 73, 123, 62, 109, 14, 205, 43, 67, 32, 103, 15, 192, 226, 58, 240, 184, 195, 1, 228, 114, 212, 29, 111, 207, 194, 8, 132, 88, 154, 182, 178, 65, 233, 81, 225, 6, 110, 25, 221, 55, 196, 183, 106, 237, 145, 79, 4, 100, 90, 164, 10, 125, 99, 69, 64, 24, 229, 121, 63, 185, 16, 203, 18, 98, 163, 158, 167, 44, 218, 131, 50, 141, 137, 143, 51, 177, 46, 96, 162, 30, 146, 70, 28, 120, 161, 213, 22, 227, 126, 210, 238, 139, 198, 102, 217, 94, 244, 169, 87, 230, 222, 116, 236, 31, 35, 7, 17, 83, 197, 118, 38, 45, 209, 59, 231, 33, 208, 61, 171, 215}.

In developing the S_2 block, an algorithm developed in [1] was used.

invB – performs the "substitution" transformation, using $invS_1$ and $invS_2$ blocks, which are inverse to blocks S_1 and S_2. That is, an input 256-bit X-block is transformed as follows:

$$invB(X)=invS_1(x_0)||invS_2(x_1)||invS_1(x_2)||\ldots||invS_2(x_{31})$$

where: $X=x_0||x_1||x_2||\ldots||x_{29}||x_{30}||x_{31}$.

$invS_1$={28, 214, 38, 230, 255, 227, 102, 148, 162, 200, 55, 66, 177, 17, 24, 226, 104, 99, 10, 191, 21, 134, 176, 240, 222, 142, 114, 86, 121, 18, 202, 39, 252, 171, 168, 217, 93, 0, 237, 1, 152, 84, 241, 72, 126, 247, 207, 49, 251, 155, 116, 105, 63, 238, 56, 59, 36, 158, 69, 87, 47, 42, 32, 198, 88, 109, 219, 172, 25, 246, 20, 133, 61, 163, 110, 151, 205, 204, 181, 53, 64, 239, 174, 137, 9, 118, 62, 232, 57, 98, 8, 27, 35, 111, 75,

182, 74, 150, 188, 54, 41, 141, 165, 46, 106, 19, 220, 29, 14, 40, 68, 161, 216, 12, 130, 199, 164, 3, 160, 253, 33, 187, 140, 65, 179, 100, 108, 194, 80, 249, 212, 113, 77, 82, 244, 213, 146, 159, 231, 143, 6, 83, 73, 81, 89, 97, 51, 122, 50, 170, 218, 131, 70, 30, 196, 115, 167, 236, 107, 128, 23, 45, 201, 101, 193, 186, 192, 175, 13, 94, 43, 197, 37, 95, 15, 71, 157, 34, 120, 189, 7, 22, 85, 67, 242, 127, 229, 5, 180, 245, 52, 208, 31, 78, 223, 228, 153, 248, 135, 211, 48, 184, 132, 233, 139, 209, 250, 76, 117, 91, 138, 96, 203, 112, 156, 210, 4, 103, 206, 144, 2, 224, 169, 234, 190, 149, 26, 16, 124, 123, 235, 90, 147, 145, 215, 254, 44, 243, 195, 79, 125, 166, 225, 221, 154, 185, 11, 119, 183, 136, 60, 58, 92, 173, 129, 178};

$invS_2$={102, 153, 86, 46, 182, 115, 171, 241, 161, 124, 186, 89, 87, 64, 140, 146, 196, 242, 198, 94, 78, 118, 222, 42, 191, 173, 83, 65, 218, 157, 215, 239, 144, 251, 112, 240, 80, 2, 246, 33, 69, 31, 29, 142, 203, 247, 212, 48, 75, 16, 206, 210, 88, 127, 37, 175, 76, 114, 149, 249, 122, 253, 138, 194, 190, 167, 15, 143, 26, 189, 217, 25, 133, 136, 7, 119, 3, 50, 68, 181, 131, 169, 123, 243, 116, 134, 95, 234, 163, 49, 184, 17, 55, 35, 231, 13, 213, 130, 199, 188, 183, 84, 229, 145, 63, 113, 178, 92, 77, 139, 172, 158, 74, 125, 155, 71, 237, 85, 245, 67, 219, 193, 117, 137, 96, 187, 224, 58, 82, 20, 106, 205, 162, 18, 107, 30, 28, 208, 24, 227, 27, 207, 66, 209, 108, 180, 216, 120, 0, 90, 72, 1, 60, 36, 164, 56, 45, 54, 201, 40, 12, 220, 214, 200, 185, 104, 38, 202, 93, 233, 97, 254, 6, 14, 57, 61, 53, 211, 166, 105, 81, 10, 165, 177, 151, 195, 59, 103, 9, 41, 8, 44, 147, 22, 160, 152, 176, 244, 228, 32, 135, 4, 73, 197, 70, 141, 62, 159, 252, 248, 225, 23, 156, 221, 51, 255, 128, 230, 204, 43, 5, 174, 236, 91, 132, 170, 148, 223, 154, 192, 235, 250, 11, 168, 129, 126, 238, 179, 226, 109, 150, 101, 111, 47, 232, 110, 98, 79, 100, 19, 21, 99, 52, 34, 121, 39}.

T (from the word "Tezkor" (fast)) – performs matrix multiplication (as MixColumns in AES) in the $GF(2^8)/\varphi(x)$ field of the input X-block as follows:

$$T(X)=\begin{bmatrix} y_0, y_1, y_2, y_3 \\ y_4, y_5, y_6, y_7 \\ y_8, y_9, y_{10}, y_{11} \\ y_{12}, y_{13}, y_{14}, y_{15} \\ y_{16}, y_{17}, y_{18}, y_{19} \\ y_{20}, y_{21}, y_{22}, y_{23} \\ y_{24}, y_{25}, y_{26}, y_{27} \\ y_{28}, y_{29}, y_{30}, y_{31} \end{bmatrix} = \begin{bmatrix} 1,2,3,4,5,112,145,225 \\ 2,1,4,3,112,5,225,145 \\ 3,4,1,2,145,225,5,112 \\ 4,3,2,1,225,145,112,5 \\ 5,112,145,225,1,2,3,4 \\ 112,5,225,145,2,1,4,3 \\ 145,225,5,112,3,4,1,2 \\ 225,145,112,5,4,3,2,1 \end{bmatrix} \bullet \begin{bmatrix} x_0, x_1, x_2, x_3 \\ x_4, x_5, x_6, x_7 \\ x_8, x_9, x_{10}, x_{11} \\ x_{12}, x_{13}, x_{14}, x_{15} \\ x_{16}, x_{17}, x_{18}, x_{19} \\ x_{20}, x_{21}, x_{22}, x_{23} \\ x_{24}, x_{25}, x_{26}, x_{27} \\ x_{28}, x_{29}, x_{30}, x_{31} \end{bmatrix}$$

where: $X=x_0||x_1||x_2||\ldots||x_{30}||x_{31}$, $T(X)=y_0||y_1||y_2||\ldots||y_{30}||y_{31}$, $\varphi(x)=x^8 \oplus x^7 \oplus x^6 \oplus x \oplus 1$.

The fixed matrix is an involution and lightweight ([8]) MDS (Maximum Distance Separable [7]) matrix. The following rules of multiplication of elements of the finite field $GF(2^8)/\varphi(x)$ can be used for effective implementation and fast execution of this transformation.

Table 1: The multiplication rules in the fields $GF(2^8)/\varphi(x)$

Multiplication ($a=a_7\|\|a_6\|\|a_5\|\|\ldots\|\|a_0$; $a_i \in \{0,1\}$)	**Rules (result) of multiplication** ($a`_7\|\|a`_6\|\|a`_5\|\|a`_4\|\|a`_3\|\|a`_2\|\|a`_1\|\|a`_0$)
$1 \cdot a$	$a_7\|\|a_6\|\|a_5\|\|a_4\|\|a_3\|\|a_2\|\|a_1\|\|a_0$
$2 \cdot a$	$a_6 \oplus a_7\|\|a_5 \oplus a_7\|\|a_4\|\|a_3\|\|a_2\|\|a_1\|\|a_0 \oplus a_7$
$3 \cdot a$	$a_6\|\|a_5 \oplus a_6 \oplus a_7\|\|a_4 \oplus a_5\|\|a_3 \oplus a_4\|\|a_2 \oplus$ $a_3\|\|a_1 \oplus a_2\|\|a_0 \oplus a_1 \oplus a_7\|\|a_0 \oplus a_7$
$4 \cdot a$	$a_5 \oplus a_6\|\|a_4 \oplus a_6 \oplus a_7\|\|a_3\|\|a_2\|\|a_1\|\|a_0 \oplus a_7\|\|a_6\|\|a_6 \oplus a_7$
$5 \cdot a$	$a_5 \oplus a_6 \oplus a_7\|\|a_4 \oplus a_7\|\|a_3 \oplus a_5\|\|a_2 \oplus a_4\|\|a_1 \oplus$ $a_3\|\|a_0 \oplus a_2 \oplus a_7\|\|a_1 \oplus a_6\|\|a_0 \oplus a_6 \oplus a_7$
$112 \cdot a$	$a_1\|\|a_0 \oplus a_1 \oplus a_7\|\|a_0 \oplus a_1 \oplus a_6 \oplus a_7\|\|a_0 \oplus$ $a_5 \oplus a_6\|\|a_4 \oplus a_5\|\|a_3 \oplus a_4\|\|a_2 \oplus a_3\|\|a_2$
$145 \cdot a$	$a_0 \oplus a_1\|\|a_1\|\|a_1 \oplus a_7\|\|a_0 \oplus a_6\|\|a_5\|\|a_4\|\|a_3\|\|a_0 \oplus a_1 \oplus a_2$
$225 \cdot a$	$a_0\|\|a_0 \oplus a_7\|\|a_0 \oplus a_6\|\|a_5\|\|a_4\|\|a_3\|\|a_2\|\|a_0 \oplus a_1$

For example, $225 \cdot 37$ ($37_{10}=00100101_2$) the product is equal to $1||1 \oplus 0||1 \oplus 0||1||0||0||1||1 \oplus 0=11110011_2=243$.

By the multiplication of $GF(2^8)/\varphi(x)$ field, it is possible to check that the number 243 is actually the result of $225 \cdot 37$ multiplication.

$$225 \cdot 37 = [(x^7 \oplus x^6 \oplus x^5 \oplus 1) \cdot (x^5 \oplus x^2 \oplus 1)] \bmod (x^8 \oplus x^7 \oplus x^6 \oplus x \oplus 1) =$$

$$[x^{12} \oplus x^{11} \oplus x^{10} \oplus x^9 \oplus x^8 \oplus x^6 \oplus x^2 \oplus 1] \bmod (x^8 \oplus x^7 \oplus x^6 \oplus x \oplus 1) =$$

$$x^7 \oplus x^6 \oplus x^5 \oplus x^4 \oplus x \oplus 1 = 243$$

The product can also be calculated using a substitution table corresponding to the multiplication result.

S (from the word "Sodda" (simple)) – linear transformation. An input X-block is transformed as follows:

$$S(X) = z_0||z_1||z_2||z_3||z_4||z_5||z_6||z_7,$$
$$z_i = w_{4i}||w_{4i+1}||w_{4i+2}||w_{4i+3},$$
$$w_{4i+j} = x_{4i} \oplus x_{4i+1} \oplus x_{4i+2} \oplus x_{4i+3} \oplus x_{4i+[(i+j) \bmod 4]}$$

where: $X=x_0||x_1||x_2||\ldots||x_{29}||x_{30}||x_{31}$.

invS – is the inverse of the transformation S, and the input X-block is transformed as follows:

$$invS(X) = z_0||z_1||z_2||z_3||z_4||z_5||z_6||z_7,$$
$$z_i = w_{4i}||w_{4i+1}||w_{4i+2}||w_{4i+3},$$
$$w_{4i+j} = x_{4i} \oplus x_{4i+1} \oplus x_{4i+2} \oplus x_{4i+3} \oplus x_{4i+[(8-i+j) \bmod 4]}$$

where: $X=x_0||x_1||x_2||\ldots||x_{29}||x_{30}||x_{31}$.

It is known that the evaluation of the cryptographic stability of the block symmetric encryption algorithms includes the following 2 main steps: evaluation for compliance with General cryptographic requirements and evaluation by cryptanalysis methods. Table 2 shows the cryptographic indicators S_1 and S_2 of the blocks of the BTS encryption algorithm.

Table 2: Cryptographic indicators of S_1 and S_2 blocks

№	Indicators	S_1	S_2
1.	*deg* (algebraic degree of non-linearity) [6]	7	7
2.	*NL* (non-linearity) [1, 2, 6]	104	112
3.	λ (stability for linear cryptanalysis) [6]	0,406	0,437
4.	δ (stability for differential cryptanalysis) [6]	0,031	0,016
5.	Return period [9]	878220	31050
6.	Fixed point [6]	–	–
7.	*AI* (algebraic immunity) [1, 2, 6]	3 (441)	2 (39)

The most frequently used methods of cryptanalysis for block symmetric ciphers include linear, differential, linear-differential, integral and algebraic methods of cryptanalysis [4]. Degrees and *AI* parameters of the S_1 block of the BTS algorithm have the maximum value. This is the basis for the algorithm to be stable to algebraic method of cryptanalysis [2, 3, 6]. For S_1 block *NL* parameters have maximum and λ, δ parameters have minimum value. These indicators are the basis for stability of linear, differential and linear-differential cryptanalysis methods [4, 6]. Adding round keys modulo 2^{32} will increase the stability to linear, integral and algebraic cryptanalysis methods [5]. Developed involution MDS matrix allows the same transformation to be used encryption and decryption processes. In addition, this saves memory when implementing algorithm operating modes and results in keeping the same feature.

3. CONCLUSION

In general, the B transformation of BTS algorithm serves to provide "confusion" as well as T and S serves to "diffusion". After the second round, the avalanche effect ([10]) coefficient reaches ≈50%, which means good performance of the avalanche criterion. Moreover, the simple operations on bytes used in the algorithm can be the basis for the simple and faster operation of the BTS encryption algorithm.

REFERENCES

1. Abdurakhimov B.F., Sattarov A.B. **An algorithm for constructing S-boxes for block symmetric encryption** // *International Journal: "Universal Journal of Mathematics and Applications". 1 (1) (2018) 29-32 Journal Homepage:* www.dergipark.gov.tr/ujma.
2. Abdurakhimov B.F., Sattarov A.B. **The method of constructing S-boxes with maximum algebraic immunity** // *Transactions of the international scientific conference "Modern problems of applied mathematics and information technologies–Al–Khorezmiy 2016", – Tashkent, 2016, pp. 130–132.*
3. Kuryazov D.M., Sattarov A.B. **Method of the constructing an algebraic system of equations describing the S-box** // *"Information Security in the light of the Strategy Kazakhstan–2050": proceedings III International scientific–practical conference (15–16 October 2015, Astana). – Astana. 2015. pp. 222–229.*
4. Babenko L., Ischukova E. **Modern algorithms of the block encryption and methods of their analysis** // *M., "Gelios ARV", 2006. pp. 38.*
5. Babenko L., Maro E. **Algebraic Cryptanalysis of GOST Encryption Algorithm** // *Journal of Computer and Communications, 2014, 2, 10-17, Published Online March 2014 in SciRes. http://www.scirp.org/journal/jcc,* http://dx.doi.org/10.4236/jcc.2014.24002.
6. Kazymyrov O. **Methods and tools for analysis of symmetric cryptographic primitives** // *Dissertation for the degree of philosophiae doctor, Harkov 2013. pp. 190.*
7. Malik M. and No J. **Dynamic MDS Matrices for Substantial Cryptographic Strength** // *Seoul National University. 2011.*
8. Siang M., Khoongming K., Frederique O. and Thomas P. **Lightweight MDS Involution Matrices**. *pp. 38.*
9. Sokolov A. **New methods of the syntheses of the nonlinear transformations modern cipher** // *LAP LAMBERT Academic Publishing (Saarbrucken, Germany), 2015. ISBN: 978-3-659-67440-2. pp. 94.*
10. https://ru.wikipedia.org/wiki/Avalanche_effect