Logical cryptoanalysis on the example of the cryptosystem DES

A.D. Plotnikov[*]

Abstract

In the paper on the example of the cryptosystem DES, the successful method of a cryptanalysis is presented. As a result, it is offered as a criterion of the cryptographic security to use a complexity of building and solving the system of Boolean functions, describing the cipher construction procedure.

MSC2000: 94A60, 68P25.
Keywords: block ciphers, DES, Boolean functions, cryptographic security, maximum independent set.

Introduction

In the modern encryption systems it is assumed that a plaintext $M = (m_1, m_2, \dots, m_N)$, a key $K = (k_1, k_2, \dots, k_{N_1})$ and a ciphertext $C = (c_1, c_2, \dots, c_{N_2})$ are Boolean vectors. Then a map f:

$$(M \times K_1) \to C$$

is called *enciphering* of the plaintext M with the use of the key K_1, and the backward transformation F:

$$(C \times K_2) \to M$$

is called *decryption* of the ciphertext C by means of the key K_2.

A set of all plaintexts $\widetilde{M}$, a space of the keys $\widetilde{K}$, a set of all ciphertexts $\widetilde{C}$, and also maps f and F form together a *cryptosystem*.

[*]The department of "Computer systems and networks" Lugansk East-Ukrainian national university. E-mail address: `a.plotnikov(AT)list.ru`

A cryptosystem is called *symmetric* if $K_1 = K_2$, and *asymmetrical* if $K_1 \neq K_2$. In last case the key K_1 is public, and the key K_2 is secret.

As all vectors M, K and C are binary then the maps f and F can be in each special case be presented by the system of the Boolean functions. In the case of the map f we have the system:

$$c_j = f_j(m_{i_1}, m_{i_2}, \dots, m_{i_p}, k_{r_1}, k_{r_2}, \dots, k_{r_q}), \tag{1}$$

where $1 \leq p \leq N$, $1 \leq q \leq N_1$ $j = 1, 2, \dots, N_2$.

The use of several keys, obviously, does not change the done conclusion, because they can be considered as parts of one (hyper) key. Thus, each ciphertext C is assigned the system of N_2 of Boolean functions (1). Such system of the Boolean functions will be called *direct* for the given cryptosystem. For example, for the cryptosystem DES, the built system will have 64+56+64=184 variables.

From other side, the map F, obviously, also in each special case is assigned the system of N of Boolean functions:

$$m_i = F_i(c_{j_1}, c_{j_2}, \dots, c_{j_s}, k_{r_1}, k_{r_2}, \dots, k_{r_q}), \tag{2}$$

where $1 \leq s \leq N_2$, $1 \leq q \leq N_1$ $i = 1, 2, \dots, N$

Thus, the vector M of plaintext is assigned the system of N of Boolean functions. Such system of the Boolean functions will be called *inverse* for the given cryptosystem.

At the building of any cryptosystem, a central problem is its cryptographic security. *Cryptographic security* of a cryptosystem is defined as a complexity of solution of the problem for finding its secret key K. Suppose that the cryptosystem is not secure if for finding the key it is required to execute less 2^{80} operations (see [9]) at enough hard conditions [8]:

(1^0) algorithms of enciphering and decryption (the maps f and F) are known;

(2^0) there are a specimen of the plaintext and the corresponding ciphertext.

The purpose of this paper is to show on the example of successful cryptoanalysis of the cryptosystem DES that if on basis of algorithms of enciphering and decryption for the analyzing cryptosystem there is possibility for reasonable time to build the systems of Boolean functions (1) and (2), at the same conditions (1^0) and (2^0), then such cryptosystem is not secure. Therefore, it is expedient as one of criterions of cryptographic security to use a complexity of building and solving of the system of Boolean functions (1) and/or (2).

Note that the solution of the systems (1) and (2), obviously, exists and it is unique for cryptosystem DES at the given conditions. Also, the cryptosystem DES was picked exceptionally as a demonstration model.

1 Features of a logical cryptanalysis

Record of transformations, related to the construction of ciphertext as a system of Boolean functions, certainly, is not newly. So, the first time C.E. Shannon [7] was specified that the value of the cryptosystem key can be found as solving a system of nonlinear equations.

The most fundamentally, such approach was examined, mainly, in works of N. Courtois and other authors (see, for example, [1], [2], [3]). The offered approach, named an algebraic attack, generated the series of researches of constructing of nonlinear Boolean functions (bent-functions) for providing of safety of a cryptosystem. Unfortunately, this approach did not avoid a necessity to execute enumeration of possibilities at the search of solution of the built system of equations.

We offer the new approach to a logical cryptanalysis. If the algebraic attack tries to use the classic methods of solution of the systems of non-linear equations, our method uses the tools of discrete mathematics. The characteristic feature of our approach is simplicity.

In the examined cryptosystem, the input vector is subjected to transformation in iteration (round). Each transformation, allowing obtaining a new Boolean vector, makes the stage of the iteration. In accordance with it, we divide an iteration of data conversions on the stages (see Fig. 1).

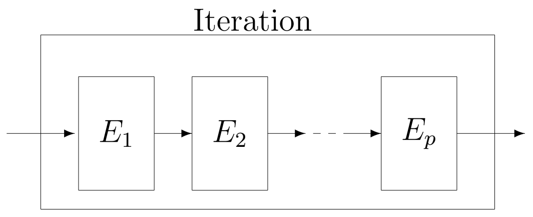

Fig. 1: Stages of data conversions on single iteration

The input of each stage be a Boolean vector X, the components of which we examine as Boolean variables. The output of any stage also be a Boolean vector Y, the components of which are examined as values of a system of Boolean functions. Solving such system of Boolean functions, we will find values components of the vector X, which, in turn, are examined as values of the system of Boolean functions, describing the previous stage of data conversion.

All Boolean functions we record in the perfect conjunctive normal form (CNF), consisting of elementary disjunctions. (We will remind that each elementary disjunction in the perfect CNF contains all variables of the Boolean function and includes single literals only, i.e. variables with negation ($\bar{x}$) and without negation (x)).

Sequence of consideration of the systems of Boolean functions, which describe data conversion on each stage, is inverted their succession in the iteration. That is, at first we examine the system of Boolean functions, which describe the last stage E_p. We find the solution of such system. Then, we examine a system, which describe the stage E_{p-1}, and so on. We complete calculation of the current iteration by consideration and by solving of the proper system of Boolean functions, which describe the first stage of the current iteration.

The sequence of iterations is also examined in an order, inverted their executing.

The accepted approach to the analysis of cryptosystem allows using the effective method of solution of the system of Boolean functions. It consists of the following.

The system of Boolean functions is assigned an undirected graph. Then the maximum independent set of such graph determines a solution of the examined system. Although, in general case, the problem of finding the maximum independent set of a graph is NP-complete, the features of the built graphs in cryptosystem DES allow to use a simple solution algorithm, which being not faithful theoretically in general case.

Thus, the key moments of the offered method of cryptanalysis consist of the following.

1. Consider an iteration (round) of data conversion in examined cryptosystem in an order, inverted their succession.

2. Divide an iteration of data conversion on the stages. Each stage determines a transformation Boolean vector in other one. Such transformation is elementary in that sense, that the obtaining some intermediate vectors is not foreseen in it.

3. Consider each stage in an order, inverted their succession in the current iteration.

4. Record the system of Boolean functions, describing the examined stage.

5. Build the appropriate undirected graph if the system of Boolean functions is not the simple renaming of variables (by permutation of components of a Boolean vector).

6. Find the maximum independent set in the built graph. Record the obtained solution as the output value of Boolean vector of the previous stage.

7. Go to examination of the previous stage.

2 Estimation of construction complexity for the Boolean functions in DES

We will analyze the cryptosystem DES.

DES is the block ciphering system, in which a plaintext (in binary presentation) is broken on 64-bits blocks, each of which is subjected to enciphering. The following operations are used in the enciphering process of the cryptosystem DES: a permutation of block bits, bit-by-bit addition modulo 2 and the constructing the system of Boolean functions by S-boxes (within the Feistel cipher construction). If the permutation of bits determines the list of variables of the Boolean functions then addition modulo 2 (the Boolean function of nonequivalence) and S-boxes determine type of the formed system of functions.

As DES is a symmetric cryptosystem, it is enough for its analysis to consider the direct system of Boolean functions (1). To have an evident picture of forming this system in the enciphering process, we will consider as it can be built for the cryptosystem DES (see [10]).

It is convenient to examine a vector $C_i = (c_1^i, c_2^i, \dots, c_{64}^i)$ as a result of execution of i-th iteration of the enciphering process. The vector C_i is also by an input vector for executing $(i+1)$-th iteration. An output vector of this iteration is a vector $C_{i+1} = (c_1^{i+1}, c_2^{i+1}, \dots, c_{64}^{i+1})$ $(i = 0, 1, 2, \dots, 15)$. Here $C_0 = M$ is a plaintext, and $C_{16} = C$ is a ciphertext. Then each component c_q^{i+1} $(q = 1, 2, \dots, 64)$ of the vector C_{i+1} be a function of components of the vector C_i and the vector of the key K_{i+1}. Therefore

$$C_{i+1} = f(C_i, K_{i+1}), \qquad (i = 0, 1, 2, \dots, 15). \tag{3}$$

We will examine a system of the Boolean functions is formed by the S_1-box. The S_1-box as well as other seven boxes will transform six input bits in four bits, that is, this box can be presented by a combinational circuit, having six inputs and four outputs (Fig. 2).

Thus, each output variable y_j $(j = 1, 2, 3, 4)$ of this circuit can be presented as a Boolean function of variables $x_1, x_2, x_3, x_4, x_5, x_6$.

Each S-box in the cryptosystem DES can be given by (4×16)-matrix. So, the matrix of the S_1-box has the following kind:

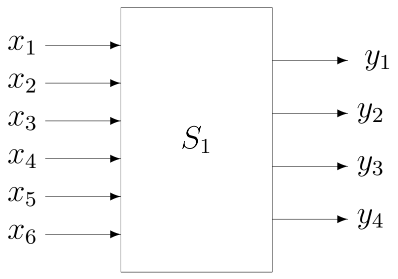

Fig. 2: Circuit of a S-box

No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

The first column of this matrix presents values of variables x_1 and x_2, analogously the first row of the matrix presents the values of variables x_3, x_4, x_5 and x_6, are read as a binary notation of the decimal number. Each element of the matrix is also read as binary notation of values of variables y_1, y_2, y_3 y_4.

The Karnaugh diagrams for the S_1-box are presented on Fig. 3.

Notice that the number of unities and zeros in each diagram equally and equals 32, i.e. each output function of y_1, …, y_4 takes on a unity value exactly on the half of sets of values of input variables x_1, …, x_6. Therefore complexity of the perfect conjunctive normal form (PCNF) for a function and its negation will be equal.

We will write down the perfect conjunctive normal form (PCNF) of the output function y_1:

	0	1	1	0	0	1	1	0	x_5
	0	0	1	1	1	1	0	0	x_4
	0	0	0	0	1	1	1	1	x_3
000	1			1	1	1	1		
001		1	1				1		
011	1		1	1		1		1	
010		1			1			1	
110	1	1		1			1		
111		1	1			1		1	
101	1	1		1			1		
100			1	1		1		1	

$x_1x_6x_2$ y_1

	0	1	1	0	0	1	1	0	x_5
	0	0	1	1	1	1	0	0	x_4
	0	0	0	0	1	1	1	1	x_3
000	1	1		1			1		
001			1	1		1		1	
011		1		1			1		
010		1	1	1	1			1	
110	1	1				1		1	
111	1		1		1	1			
101	1	1	1		1				
100	1			1			1	1	

$x_1x_6x_2$ y_2

	0	1	1	0	0	1	1	0	x_5
	0	0	1	1	1	1	0	0	x_4
	0	0	0	0	1	1	1	1	x_3
000	1				1		1	1	
001	1	1		1		1			
011	1	1	1		1				
010		1		1			1	1	
110	1		1			1			
111		1	1	1	1			1	
101	1		1				1	1	
100				1	1	1	1		

$x_1x_6x_2$ y_3

	0	1	1	0	0	1	1	0	x_5
	0	0	1	1	1	1	0	0	x_4
	0	0	0	0	1	1	1	1	x_3
000			1	1	1		1		
001	1					1	1	1	
011			1		1		1	1	
010		1		1	1	1			
110	1				1	1	1		
111	1	1		1		1			
101	1		1	1	1			1	
100		1				1		1	

$x_1x_6x_2$ y_4

Fig. 3: Karnaugh diagrams for outputs of the S_1-box

$$
\begin{aligned}
y_1 = {} & (x_1 \vee x_2 \vee x_3 \vee x_4 \vee \bar{x}_5 \vee x_6)(x_1 \vee x_2 \vee x_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee x_6)(\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee x_4 \vee x_5 \vee x_6)\& \\
& (x_1 \vee \bar{x}_2 \vee x_3 \vee x_4 \vee x_5 \vee x_6)(x_1 \vee \bar{x}_2 \vee x_3 \vee \bar{x}_4 \vee x_5 \vee x_6)(x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5 \vee x_6)\& \\
& (x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee x_6)(x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4 \vee x_5 \vee x_6)(x_1 \vee \bar{x}_2 \vee x_3 \vee x_4 \vee \bar{x}_5 \vee \bar{x}_6)\& \\
& (x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5 \vee \bar{x}_6)(x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4 \vee \bar{x}_5 \vee \bar{x}_6)(x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5 \vee \bar{x}_6)\& \\
& (x_1 \vee x_2 \vee x_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee \bar{x}_6)(x_1 \vee x_2 \vee x_3 \vee \bar{x}_4 \vee x_5 \vee \bar{x}_6)(x_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee \bar{x}_6)\& \\
& (x_1 \vee x_2 \vee \bar{x}_3 \vee x_4 \vee \bar{x}_5 \vee \bar{x}_6)(\bar{x}_1 \vee x_2 \vee x_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee \bar{x}_6)(\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5 \vee \bar{x}_6)\& \\
& (\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee \bar{x}_6)(\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee x_4 \vee x_5 \vee \bar{x}_6)(\bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee x_4 \vee x_5 \vee \bar{x}_6)\& \\
& (\bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee \bar{x}_4 \vee x_5 \vee \bar{x}_6)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5 \vee \bar{x}_6)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4 \vee \bar{x}_5 \vee \bar{x}_6)\& \\
& (\bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee x_6)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5 \vee x_6)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_5 \vee x_6)\& \\
& (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4 \vee x_5 \vee x_6)(\bar{x}_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5 \vee x_6)(\bar{x}_1 \vee x_2 \vee x_3 \vee x_4 \vee \bar{x}_5 \vee x_6)\& \\
& (\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5 \vee x_6)(\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee x_4 \vee \bar{x}_5 \vee x_6).
\end{aligned}
$$

The output functions of the other S-boxes have analogical kind.

Now we will estimate the construction complexity of the system of Boolean functions (3) in the cryptosystem DES. Such system is built as a result of executing an iteration of ciphertext construction. Flow-chart of such iteration is presented on Fig. 4 (see also [10]).

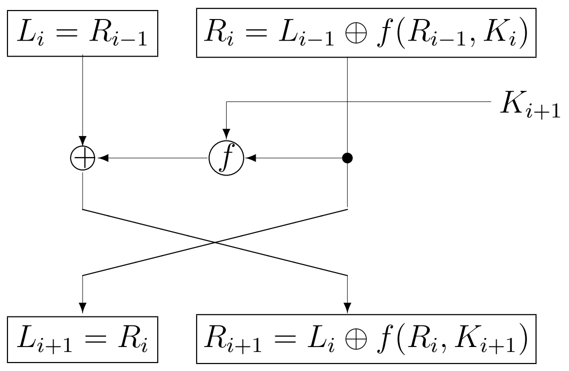
$L_i = R_{i-1}$
$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$
K_{i+1}
f
$L_{i+1} = R_i$
$R_{i+1} = L_i \oplus f(R_i, K_{i+1})$

Fig. 4: An iteration of DES

We will estimate the construction complexity of the Boolean functions as the number of elementary operations which must be executed to obtain the result vector C_{i+1} of the vector C_i and the key K_{i+1}. We suppose that any indivisible operation is elementary. Such operation is a write operation and any Boolean operation. As we want to obtain estimation we will coarsen it toward an increase.

So, a 64-bit input vector C_i on $(i+1)$-th iteration $(i = 0, 1, 2, \ldots, 15)$ is divided on two 32-bit blocks: left L_i and right R_i. The right block R_i forms the left block of a 64-bit output vector C_{i+1}, i.e. 32 operations are simply executed in this case $c_q^{i+1} = c_j^i$ $(q = 1, 2, \ldots, 32;\ j = 33, 34, \ldots, 64)$. And the following is executed for forming of the right block of the output vector C_{i+1}. The construction of such transformation (the Feistel cipher construction) is represented on Fig. 5 (see also [10]).

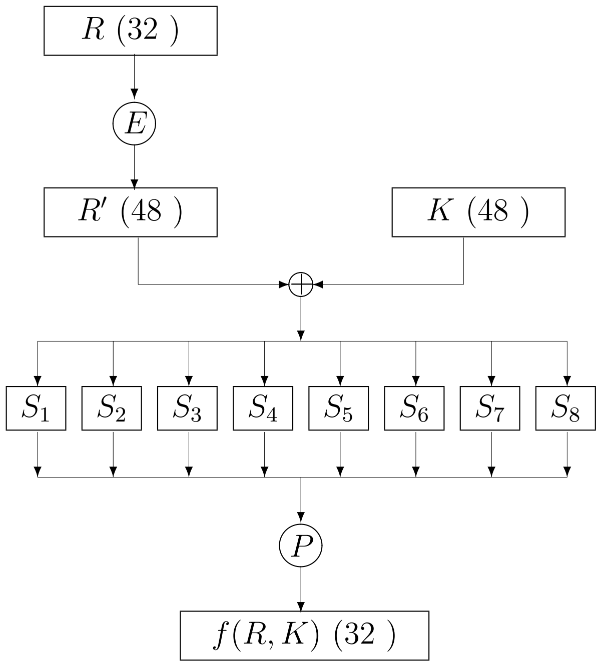

Fig. 5: Forming the cipher function f

Foremost, according to the defined in the cryptosystem rule, a new vector is formed of 32-bit vector R_i by means of permutation and reiteration some elements of R_i. The new vector will be designated as $R_i' = E(R_i)$. Clear that for this purpose it is required to execute 48 operations of assignment. A key vector is exposed to transformations (see [10]) of shift and permutation. We will estimate it by the number of operations $2 \cdot 28 + 2 \cdot 28 + 56 = 168$ (shifts of each half of the key and its final permutation). Further, the new vector R_i' is added (bit-by-bit) modulo 2 with the 48-bit key K_{i+1}. It requires also 48 operations and it is necessary still to add 48 operations of assignation for a new result vector. We will designate it through B_{i+1}.

Then, each 6 bits of vector B_{i+1} are processed by the appropriate S-box. There are only 8 $(8 \cdot 6 = 48)$ such boxes. We have 4 bits of a result of transformations of 6 bits in each of S-box (see of Fig. 2). Clear, that for the construction of Boolean functions on each output of a S-box it is necessary to examine everything $2^6 = 64$ possible sets of values of variables of each function. As follows from Fig. 3, a half of these sets are unities (i.e. a function on such sets is equal to unity). For each zero set (on which a function is equal to the zero) it is necessary to build the appropriate elementary disjunction (as it is done at construction of CNF). Length of each disjunction is equal 6 and, we will suppose that a half of variables in it has negations. Consequently, it needs to expend 8 operations (for a record of 5 disjunctions and 3 negations) for making one elementary disjunction.

Because it is needed to write 32 (the number of zero sets) elementary disjunctions, on the construction of one Boolean function, presenting each one output of the appropriate S-box, it is required to expend $32 \cdot 8 + 64 = 320$ operations. As there are 4 outputs the same S-box it is necessary to execute $320 \cdot 4 = 1280$ operations. In the total, for the construction of Boolean functions for all 8 S-boxes it is necessary to execute $1280 \cdot 8 = 10240$ operations. Calculation of function f ends by permutation that requires 32 operations. As a result we have values of intermediate 32-bit vector H_{i+1}.

Finally, the calculation of right block is completed by the bit-by-bit addition modulo 2 results of permutation of H_{i+1} and 32-bit vector L_i that requires 32 operations plus the operation of assignment

$$R_{i+1} = L_i \oplus f(R_i, K_{i+1}) = L_i \oplus H_{i+1}.$$

Theorem 1. *For cryptosystem of DES, the number of elementary operations, which must be expended on the construction of the system of Boolean functions (3) for describing an iteration, approximately equal* $2^{14} = 10648$.

Indeed, taking into account the obtained results above, on the construction of the system of Boolean functions for describing an iteration of cryptosystem DES, it is required to expend $32+48+168+48+48+10240+32+32 = 10648 < 2^{14}$ operations. Q.E.D.

The Boolean functions, describing transformations of input vector C_i on $(i+1)$-th iteration into an output vector C_{i+1}, convenient to write, reflecting each stage of data conversion. Then the system (3) can be rewritten in the following kind:

$$c_{j_1}^{i+1} = c_{j_2}^{i}, \tag{4a}$$
$$r'_j = E(c_{33}^{i}, c_{34}^{i}, \ldots, c_{64}^{i}), \tag{4b}$$
$$b_j^{i+1} = r'_j \oplus k_j^{i+1}, \tag{4c}$$
$$h_{j_1}^{i+1} = \phi_{j_1}(b_{q_1}^{i+1}, b_{q_2}^{i+1}, b_{q_3}^{i+1}, b_{q_4}^{i+1}, b_{q_5}^{i+1}, b_{q_6}^{i+1}), \tag{4d}$$

$$c_{j_2}^{i+1} = c_{j_1}^{i} \oplus h_{j_1}^{i+1}, \tag{4e}$$

where $i = 0, 1, 2, \ldots, 15$, $j_1 = 1, 2. \ldots, 32$; $j_2 = 33, 34, \ldots, 64$, $j = 1, 2, \ldots, 48$.

Notice that the obtained system will contain the embedded Boolean functions. In this system, except for the operations of disjunction, conjunction and negations, will be used also operation of nonequivalence (addition modulo 2).

In principle, executing superposition of functions of (4), it is possible to obtain the system of Boolean functions in an explicit form (1). However it makes inexpediently and it is not needed because such system is intricate.

Separately, it is not difficult to write transformations of the key K within the framework of the key schedule [10].

3 Construction of the graph problems sequence

Each Boolean function of the system (1) can be written as an equation:

$$c_j \oplus f_j(m_{i_1}, m_{i_2}, \ldots, m_{i_p}, k_{r_1}, k_{r_2}, \ldots, k_{r_q}) = 0,$$

or

$$1 \oplus c_j \oplus f_j(m_{i_1}, m_{i_2}, \ldots, m_{i_p}, k_{r_1}, k_{r_2}, \ldots, k_{r_q}) = 1,$$

and, at last, the transformed system (1) will accept a kind:

$$\bar{c}_j \oplus f_j(m_{i_1}, m_{i_2}, \ldots, m_{i_p}, k_{r_1}, k_{r_2}, \ldots, k_{r_q}) = 1, \tag{5}$$

where $1 \leq p \leq N$, $1 \leq q \leq N_1$ $j = 1, 2, \ldots, N_2$.

Multiplying logically the left and right parts of the obtained system of equations, we will reduce it to one equation:

$$\bigwedge_{j=1}^{N_2} \bar{c}_j \oplus f_j(m_{i_1}, m_{i_2}, \ldots, m_{i_p}, k_{r_1}, k_{r_2}, \ldots, k_{r_q}) = 1. \tag{6}$$

A famous satisfiability problem (SAT) is obtained [4]. If we will present the left part of the equation (6) in standard form as a conjunctive normal form (CNF) then it is possible to apply all known methods being used for the solution of this problem. The review of these methods for solving of SAT problem is presented in the paper [5].

However, it is expedient to reduce the obtained problem to the maximum independent set problem on a graph. Although this problem, as well as the SAT problem is also NP-complete, there exist ways of its effective solution in the most of cases. This question will discuss below.

We will explain on the example how the SAT problem can be reduced to the maximum independent set problem. Let the SAT problem is presented the following CNF:

$$(a \vee b \vee \bar{c})(\bar{a} \vee b \vee c) = 1. \tag{7}$$

Each literal of CNF be assigned to a vertex of a graph G. We will join the vertices of the same elementary disjunction by edges such that these vertices form a clique of G. Besides, we will join the alternative literals (x and $\bar{x}$) by edges. The obtained graph is presented on Fig. 6.

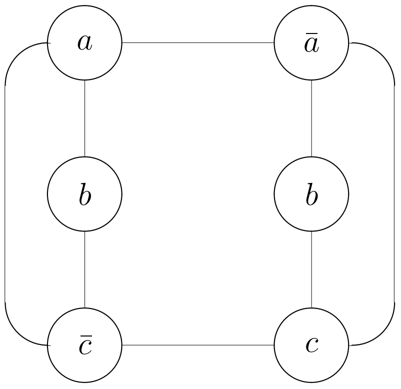

Fig. 6: The graph for the CNF $(a \vee b \vee \bar{c})(\bar{a} \vee b \vee c)$

It is easy to see that the maximum independent set of vertices (MMIS) ($\{b, b\} = \{b\}$) of the built graph satisfies the equation (7), that is, if values of literals, the appropriate vertices of MMIS, equal to unity then the value of the equation equals to unity too.

As mentioned above, a process of the executing of one iteration of the DES algorithm, and also the integrated results of different iterations, it is described by the Boolean functions in a parenthesis notation. Within iteration, it is necessary to insert the functions of one stage of calculations instead of variables of a Boolean function, which are formed on the subsequent stage of calculations, i.e. to execute superposition of the Boolean functions. And for transition from i-th iteration to $(i + 1)$-th iteration, it needs take the results of previous iteration as input variables of C_i.

We will define that is the superposition operation of Boolean functions in the graph presentation.

Comment 1. *Because Boolean functions, which substitute of variables, also can be presented by a subgraph, a meaning of graph edges on Fig. 6 consists* *in the following. If vertices a, b and c (we will call them by hypervertices) will be transferable by the appropriate subgraphs then each vertex of one subgraph must be joined with each vertex of other subgraph by an edge, when the appropriate hypervertices are joined by an edge.*

In the process of superposition of operation of addition modulo 2, we will use the followings transformations:

$$x \oplus y = (x \vee y)(\bar{x} \vee \bar{y}), \tag{8a}$$
$$\overline{x \oplus y} = (\bar{x} \vee y)(x \vee \bar{y}). \tag{8b}$$

For example, let $a = x_1 \oplus y_1$, $b = x_2 \oplus y_2$, $c = x_3 \oplus y_3$. Then, substituting variables a, b and c in relation (7) and taking into account the relations (8), we will obtain:

$$\begin{aligned}&((x_1 \vee y_1)(\bar{x}_1 \vee \bar{y}_1) \vee (x_2 \vee y_2)(\bar{x}_2 \vee \bar{y}_2) \vee (\bar{x}_3 \vee y_3)(x_3 \vee \bar{y}_3))\&\\ &\&((\bar{x}_1 \vee y_1)(x_1 \vee \bar{y}_1) \vee (x_2 \vee y_2)(\bar{x}_2 \vee \bar{y}_2) \vee (x_3 \vee y_3)(\bar{x}_3 \vee \bar{y}_3))\end{aligned} \tag{9}$$

A graph, the appropriate obtained Boolean function, is presented on Fig. 7.

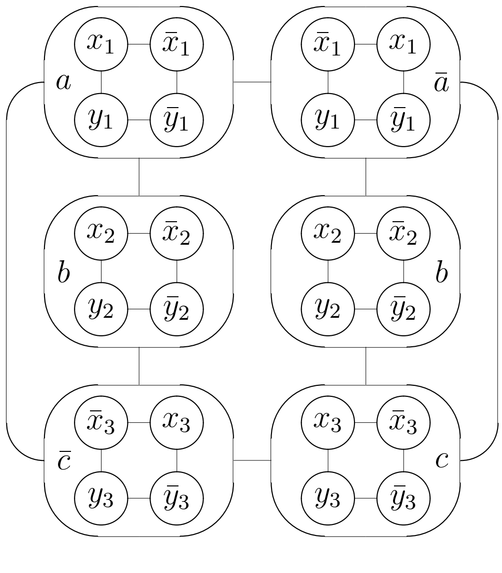

Fig. 7: Extended graph

Comment 2. *Note that in a new graph, obtained as a result of superposition, except edges, indicated in Comment 1, it is necessary to enter edges, joining alternative literals.*

Obviously, that the extended graph, obtained after a substitution, is redundant for our aims – for finding of the satisfying set of values of Boolean variables of the equation (9). It is enough to examine a graph, substituted in place of vertices of MMIS of the primary graph. This graph is presented by the vertex b in our example.

Features of execution of each $(i+1)$-th iteration ($i = 0, 1, 2, \ldots, 15$) of the cryptosystem DES consist in that for its description it is necessary to form Boolean functions in three stages. On the first stage the Boolean function of nonequivalence (addition modulo 2) is formed, and on the second one we obtain by means of S-boxes some CNF and, finally, on the third stage– again the Boolean function of nonequivalence is formed. All other functions of the system (4) execute renaming of Boolean variables.

The first stage ends by forming the intermediate vector B_{i+1} (see Part 2). This components are functions of the vector R' and the appropriate key K_{i+1}. Clear, that any such function (see (4c)) can put in accordance of the subgraph G'''_j as it is done in an example above. Components of the key K_{i+1} and components of the vector R' is assigned the subgraph vertices.

On the second stage the subgraph, induced by Boolean functions, obtained by means of S-boxes, is formed. We will designate it as G''_{j_1}. The components of the intermediate vector B_{i+1} are assigned the vertices of this subgraph (see (4d)).

At last, the calculations on $(i+1)$-th iteration end by forming the function (4e), which induces the subgraph G'_{j_2}. The vertices of this subgraph are assigned to components of vectors L_i and H_{i+1}.

Let, in obedience to the enciphering process of DES, the space of the keys is formed $K_1, K_2, \ldots, K_{16}$.

Then for solving of the system (4) it is necessary to use the following procedure:

1. Find all function of the current $(i+1)$-th iteration in obedience to (4).

2. Construct the graph G'_{i+1} as union of subgraphs G'_{j_2} ($j_2 = 33, 34, \ldots, 64$) and find its MMIS U'_{i+1}.

3. Record the vertices of the MMIS U'_{i+1}, delete of the subgraph G'_{i+1}.

4. Build the extended graph G''_{i+1}, replacing each vertex of the MMIS U'_{i+1} by the appropriate subgraph G''_{j_1} ($j_1 = 1, 2. \ldots, 32$), and find its MMIS U''_{i+1}.

5. Record the vertices of the MMIS U''_{i+1}, delete the graph G''_{i+1}.

6. Build the extended graph G'''_{i+1}, replacing each vertex of the MMIS U''_{i+1} by the appropriate subgraph G'''_j $(j = 1, 2, \dots, 48)$, and find its MMIS U_{i+1}.

7. Record the vertices of the MMIS U_{i+1}, delete the graph G'''_{i+1}.

8. Complete the calculations, connected with solving the system (4).

Remind that at the construction of any extended graph, it is necessary to join all alternative literals in it by edges.

Theorem 2. *The variables, that correspond to the MMIS U_{i+1}, are the satisfying set for the system* (4)*.*

It is a consequence from the way of constructing of graphs for appropriate Boolean equations. Q.E.D.

Thus, for solving all 16 systems of the Boolean equations, describing the enciphering process in the cryptosystem DES, it is necessary to construct a sequence of graphs

$$\underbrace{G'_{16}, G''_{16}, G'''_{16}}, \underbrace{G'_{15}, G''_{15}, G'''_{15}}, \cdots, \underbrace{G'_1, G''_1, G'''_1}. \tag{10}$$

The described above graphs are being formed first for 16-th iteration, after, for 15-th iteration, and so on. Calculations are completed after a construction and solution of Boolean functions on the first iteration. The graphs, formed on each (i)-th iteration $(i = 15, \dots, 0)$, are always put instead the vertices of the recorded MMIS of the graph G'''_{i+1}. Naturally the existing ciphertext $C = C_{16}$ is examined as the first MMIS.

Theorem 3. *The sequence (10) solves the system of the Boolean functions (1) for the cryptosystem DES.*

It is a consequence of Theorem 2. Q.E.D.

The described above procedure of constructing of the graph sequence implies some features of forming the system of Boolean functions (4).

First of all, we will notice that the presentation of the system of Boolean functions in kind of the equation (5) for superposition is inexpediently. We will suppose that the system is given by equations (4). In this case a question appears about how a Boolean function substitutes the literal with a negation. Already at consideration of function on 16-th iteration, the output vector C_{16} can contain zero components. This fact we will be to examine as a necessity of substitution of the function instead the literal with a negation.

Comment 3. *Before to build of one of the Boolean functions* (4c) – (4e)*, it is necessary to determine the presence of a negation in such literal of a MMIS, which will be replaced by this function. If the literal of the MMIS does not have a negation, for presentation of the functions* (4e) *and* (4e)*, we will use the equation* (8a)*, and we will write the PCNF for the function* (4d) *on its zero sets. If the literal of the MMIS has a negation, for presentation of the functions* (4e) *and* (4e) *we will use the equation* (8b)*, and we will write CNF for the function* (4d) *on its unity sets, i.e. the CNF is formed for the negation of this function.*

Lemma 1. *The graph vertices of* G'_{i+1}*,* G''_{i+1} *and* G'''_{i+1} *(*$i = 0, 1, 2, \dots, 15$*) are assigned to only variables (with negation or without negation), which the appropriate Boolean functions depend on.*

It follows from Comment 3. Q.E.D.

4 Complexity of solution of the graph problems sequence

First of all, we will estimate the size of graphs in the sequence (10).

In accordance with the procedure of construction of the graph sequence (10), on 16-th iteration of the algorithm DES, first the graph G'_{16} is formed as a union of 32 subgraphs G'_{j_2} ($j_2 = 33, 34, \dots, 64$). These subgraphs present the Boolean functions of type (8), depending on 64 variables $c^{15}_{j_1}$ and h_{j_1} ($j_1 = 1, 2. \dots, 32$). Hence, the graph G'_{16} will contain $32 \cdot 4 = 128$ vertices and its MMIS U'_{16} cannot contain more than 64 vertices, since any MMIS cannot contain alternative literals simultaneously.

Further, the graph G''_{16} is formed also as an union subgraphs G''_{j_1} ($j_1 = 1, 2. \dots, 32$), presenting the Boolean functions of type (4d). These Boolean functions depend on 6 vertices each and the appropriate PCNF of any Boolean function contains 32 elementary disjunctions with 6 literals. Therefore, each subgraph will contain $32 \cdot 6 = 192$ vertices, which replace one of vertices of the earlier found MMIS U'_{16}. Hence, the graph G''_{16} will contain $64 \cdot 32 \cdot 6 = 12288$ vertices. However, its MMIS U''_{16} cannot contain more than 48 variables that is the number of component of the vector B_{16}.

At last, 16-th iteration is being completed by building the graph G'''_{16} and by substitution subgraphs G'''_j ($j = 1, 2, \dots, 48$) into the MMIS U''_{16}. As each such subgraph contains 4 vertices then the total number of vertices of the graph G'''_{16} equal to $48 \cdot 4 = 192$. However, the MMIS U_{16} of the graph G'''_{16} cannot contain more than $32 + 48 = 80$ vertices (they correspond to 32 components of left part of the input vector L_{15} and 48 components of the key vector K_{16}).

So, 80 vertices of the MMIS U_{16} of the graph G'''_{16} and 32 vertices of the vector C_{16} develop the input vector for calculation on 15-th iteration. Thus, this input vector contains at the most $80 + 32 = 112$ vertices.

It is easy to see that starting with 15-th iteration down to 1-st iteration of (10), the appropriate MMIS U_i ($i = 15, 14, \ldots, 1$) has the same size, i.e. 112 vertices. It increases only the number of graph vertices in G'_i but, in accordance with Lemma 1, this does not change the size of the following graphs in (10).

Hence, the following assertion is true.

Theorem 4. *In the sequence (10), the most large graph contains not more than* 2^{14} *vertices.*

The most large graph in (10) is the graph G''_{i+1} ($i = 0, 1, \ldots, 15$). It contains not more $12288 < 2^{14}$ vertices. Q.E.D.

5 Complexity of solving graph problems

The question arises about the complexity of finding the MMIS for graphs of the sequence (10).

As specified, this problem is NP-complete. Therefore there are not the proved effective (polynomial-time) solving algorithms for finding a MMIS in any graph. However there are effective heuristic algorithms, which show good results in practice. The algorithm MIN for finding the maximum independent set of graph vertices can serve by the example of such algorithm.

The algorithm MIN is simple. In the initial graph G, the local degrees of vertices are being found, and a vertex, having the minimum local degree, is being picked. After picking, the found vertex is being recorded and all adjacent vertices are deleted from the graph G. Further all begins at first. It is continued until the empty graph will be obtained. Suppose that the selected vertices form the MMIS of the graph G.

The different researchers, including author of this article, repeatedly re-opened this algorithm. Theoretical estimation of complexity of this algorithm equal $O(n^2)$, where n is the number of graph vertices. However, an error in the solution (on the specially built examples) can be arbitrary large.

The existing situation with the solution of the MMIS problem to some extent is analogical to the earlier existed situation with the solution of the linear programming problem. It was proved that simplex-algorithm has an exponential-time estimation. However, in practice, it shows good results, that is, theoretically bad algorithm successfully solves many practical problems. Therefore there is a hope that the algorithm MIN correctly cans solution the MMIS problem in the simple graphs of the sequence (10).

There is a question: can the algorithm MIN solve graph problems correctly, which formed in the process of analysis of the cryptosystem DES?

Now, we consider what type of graphs was obtained in the process of analysis of the stages on any iteration.

In the system (4), two functions (4c) and (4e) present addition modulo 2. It means that in accordance with Comment 3 each of such functions will be presented simple subgraph, the vertices of which form 4-vertex cycle (see the example in Section 3). Clear, that in MMIS of the graph, presenting the join of 4-vertex cycles, exactly two vertices can enter from every subgraph, because different subgraphs do not contain vertices, proper alternative literals. Therefore these subgraphs are not joined by an edge between itself.

It was conducted also a calculating experiment in solving the systems of functions, generated by S-boxes. In the process of experiment there were found the maximum independent sets in subgraphs, generated every function y_i $(i = 1, 2, 3, 4)$, and also by their joins. The number of vertices in MMIS always was exactly equal to the number of elementary disjunctions in each function y_i or in joins of functions. Certainly, the vertices of MMIS corresponded to literals, the number of which is not exceeded 6.

The experiment allows to assert that the algorithm MIN solves the systems of Boolean equation, generated the algorithm of DES correctly.

Theorem 5. *If the algorithm MIN correctly solves the MMIS problem in each of graphs of the sequence (10), then the time of determination of the used key in cryptosystem DES equals at the most* 2^{32}.

The most number of vertices is in the graph sequence (10) equal 2^{14} approximately. Hence, the algorithm MIN will require 2^{28} unities of the time for the solution of the system of Boolean functions, describing one iteration of ciphering data. Then all 16 iterations will demand at the most 2^{32} unities of the time Q.E.D.

Corollary 1. *The cryptosystem DES is not secure.*

The attempts were done to solve this problem exactly (as well as other NP-complete problems). Some of developed (polynomial-time) algorithms allow finding the exact solution for many graphs. To test these algorithms, the numerous graph instances, having thousands of vertices, are developed (see http://www.nlsde.buaa.edu.cn/∼kexu/benchmarks/graph-benchmarks.htm).

Unfortunately, while it is not proved the theoretical correctness of the constructed algorithms.

One of such polynomial-time algorithms is developed in the work [6] for the solution of the MMIS problem. The algorithm uses the search procedure of the maximum antichain in a finite partially ordered set. For this purpose, the edges of the initial graph are oriented such that its vertices correspond the elements of a partially ordered set. If it is necessary, fictitious arcs are entered in the constructed digraph. The solution algorithm is based on the hypothesis of author about properties of the special constructed digraph. Theoretical estimation of complexity of the solution algorithm equal to $O(n^8)$, where n is the number of graph vertices.

In accordance with the developed algorithm, Thomas Karbe from Berlin technical university wrote the Java-program

(see http://www.vinnica.ua/∼aplot/solver.html).

The numerous tests of this program did not find errors in solutions of the developed algorithm.

6 Conclusions

On basis of the executed cryptanalysis of the cryptosystem DES it is possible to do the following conclusions.

Firstly, the successful cryptanalysis on the example of the cryptosystem DES shows its low cryptographic security. This is consequence of simplicity and possibility to construct the system of Boolean functions, describing an iteration of the cipher algorithm. Consequently, for the increase of cryptographic security of any ciphering system it is desirable that it was impossible to make the system of the Boolean functions, describing an enciphering iteration that can be achieved in some ways. Simple complication of the systems of the Boolean functions will influence only complication process of enciphering and expenses of time on this process.

In the second, it is necessary in the block ciphering system that a length of the key must exceed a length of ciphering block substantially such that the afore-mentioned system of Boolean functions, when it can be written, had not unique solution, but it must have a set of solutions even at conditions when a plaintext and the appropriate ciphertext are known. A size of this set of solution must be such that the direct enumeration of possible system solutions would be unacceptable.

At last, a few words about the presented method of logical cryptanalysis. Classic approach to solving the system of Boolean equations as the system of nonlinear algebraic equations, obviously, did not justify hopes of cryptanalysts. In our view, tools of discrete mathematics are more suitable in this case. In particular, the advantage of the offered method consists in reduction of solving the system of Boolean equations as the graph problem in which non-linearity of Boolean equations does not play an important role.

References

[1] N. T. Courtois and W. Meier. *Algebraic Attacks on Stream Ciphers with Linear Feedback.* Proceedings of EUROCRYPT 2003, Lecture Notes in Computer Science 2656, pp. 346-359, 2002.

[2] N. T. Courtois. *Fast Algebraic Attacks on Stream Ciphers with Linear Feedback.* Proceedings of CRYPTO 2003, Lecture Notes in Computer Science 2729, pp. 177-194, 2003.

[3] J.-C. Faugere and G. Ars. *An Algebraic Cryptanalysis of Nonlinear Filter Generators using Grobner bases.* Rapport de Recherche INRIA 4739, 2003.

[4] M. R. Garey and D. S. Johnson *Computers and Intractability.* W.H.Freeman and Company, San Francisco, 1979.

[5] J. Gu, P. W. Purdom, J. Franco, and B. W. Wah *Algorithms for the Satisfiability (SAT) Problem: A Survey.* DIMACS. Series in Discrete Mathematics and Theoretical Computer Science.

[6] A. D. Plotnikov *Experimental Algorithm for the Maximum Independent Set Problem..* http://lanl.arxiv.org/abs/0706.3565

[7] C. E. Shannon. *Communication theory of secrecy systems.* Bell system technical journal, 28, pp. 656-715, 1949.

[8] B. Schneier *Applied Cryptography: Protocols, Algorithms, and Source Code in C* Paperback, John Wiley & Sons, 1996.

[9] N. P. Smart *Cryptography.* A McCraw-Hili Publication McGraw-Hill, 2003.

[10] U.S. Department of Commerce/National Institute of Standards and Technology *Data Encryption Standard (DES)* Federal Information, Processing Standards Publication 46-3, 1999 October 25.