DOI: https://doi.org/10.53486/cike2022.39
CZU: 330.4:517.987.3

A KEY EXCHANGE METHOD BASED ON BOOLEAN FUNCTIONS AS SUBSETS OF COLUMNS

AURELIU ZGUREANU
Department of Information Technology and Information Management
Academy of Economic Studies of Moldova
Chisinau, Republic of Moldova
Email account: zgureanu.aureliu@ase.md

Abstract: *The representation of Boolean functions as subsets of columns and one of its possible applications is discussed in this paper. Depending on the area of application, different Boolean function representations are used. Boolean functions as subsets of columns were investigated by the author together with other colleagues and published in many scientific works, which allow to apply this kind of representation in different domains. Based on the properties of the subsets of columns of Boolean functions, an algorithm of encryption key exchange between two or more entities is proposed. The algorithm consists of a long-lived secret key which consist of a family of n Boolean functions. The session key k_{ses} is defined by a subset of column of the partial derivative of one of the Boolean functions, randomly chosen from the secret key. The parameters that uniquely determine the secret key are generated randomly by one of the parties and may be sent nonencrypted to all other who are involved in the communication session. The main advantage of the algorithm is that it doesn't use public key cryptography, which is much more computationally demanding than calculation of the particular subset of column. The main challenge of the algorithm is choosing the correct type of functions that have as diverse subsets of columns as possible. The parameters of the table of partial derivatives of the Boolean functions also are very important and they need to best suit our purpose. These two particularities need further investigations.*

Keywords: Boolean function, subsets of column, Boolean function derivatives, key exchange, secret key, session key.

JEL Classification: C61, C63

1 Introduction

It is known (Bochmann & Posthoff, 1981) that determining a Boolean function knowing only its partial derivatives, is a very difficult task. The problem becomes even more complicated in the case when not all of the derivatives are known. This happens when the function and its partial derivatives are represented as subsets of columns (Булат, 1972), (Булат, 2002). This fact can be used to increase the processing speed of a cryptosystem without decreasing the cryptographic strength. For this, we need to know an effective way of calculating the derivatives of Boolean functions. Such a method was proposed in (Булат, 2002) and used in (Zgureanu, 2011), (Bulat, et al., 2012a), (Bulat, et al., 2010). In this paper we propose another application of Boolean functions partial derivatives, namely for cryptographic key exchange.

Boolean functions in their different representation are very important for a lot of domains. They have become the accepted model for designing circuits used in electronics nowadays. The theory of Boolean functions is also used in areas such as pattern recognition, coding theory, game theory, Reliability theory, Combinatorics, Logic, Cryptography, etc.

The function

$$F(x_1, \dots, x_\tau, \dots, x_n),$$

whose arguments

$$x_1, \dots, x_\tau, \dots, x_n$$

as well as the function itself, assume values from a two-element set {0,1}, is called Boolean function (Bochmann & Posthoff, 1981). Formally it is an *n*-ary relation R_{X^n}, where $X = \{0,1\}$ and can be represented by the *truth table*, which explicitly lists its values ε_i for all the possible arguments. The truth table for the function *F* is shown in Table 1, where $\varepsilon_i \in \{0,1\},\ i = \overline{1,u}$ and $u = 2^n - 1$. Truth table is very useful for functions of a quite small number of variables. When this number is raising truth tables became not unpractical because of its dimensions. For example, if *F* is a Boolean function of 10 variables $x_1, \dots, x_\tau, \dots, x_{10}$, its truth table has 2^{10} = 1024 rows. In such situations other Boolean function's representations could be used. The most popular ones are algebraic expressions: negation normal form, disjunctive normal form, conjunctive normal form, canonical normal form. A function may be expressed through several logically equivalent algebraic expressions, but there is only one unique truth table for every function. In each specific situation is used the representation which suits best for practical reasoning.

Table 1 Boolean function's truth table

	x_1	$\cdots$	x_τ	$\cdots$	x_n	$F(x_1, \dots, x_n)$
	0	$\cdots$	0	$\cdots$	0	ε_0
			$\ddots$			$\vdots$
	σ_1	$\cdots$	σ_τ	$\cdots$	σ_n	ε_i
			$\ddots$			$\vdots$
	1	$\cdots$	1	$\cdots$	1	ε_u

Source: based on Bochmann and Posthoff (1981)

This paper discus Boolean functions represented as subset of columns, which allows in many situations to operate more conveniently on Boolean functions even given in other different representations. This form of Boolean functions was investigated by the author of this paper together with other colleagues and published in many scientific works, which allow to apply this kind of representation in different domains because it needs less computational resources when operating on Boolean functions represented in this way. Based on the research on Boolean functions as subset of columns a key exchange algorithm is proposed.

2 Boolean Functions Represented as Subset of Columns

2.1 Representing Boolean Functions as Subsets of Columns

According to (Булат, 1972), (Булат & Горюк, 1978), (Bulat, et al., 2008b), on the set

$$X = \{x_1, \dots, x_n\} \tag{1}$$

we build a partition

$$\{\tilde{X}_1, \tilde{X}_2\} = \{\{x_1, \dots, x_\tau\}, \{x_{\tau+1}, \dots, x_n\}\} \tag{2}$$

and define two sets

$$Y = \{y_0, y_1, \dots, y_{2^\tau - 1}\}, Z = \{z_0, z_1, \dots, z_{2^{n-\tau} - 1}\} \tag{3}$$

which consist of binary sets built on variables from $\tilde{X}_1$ and $\tilde{X}_2$ respectively. Function $F(x_1, \dots, x_n)$ can be viewed as a binary relation R_{YZ} on the sets Y and Z and can be represented by the following matrix

$$R_{YZ} = \begin{array}{c|ccccc} & z_0 & \cdots & z_j & \cdots & z_p \\ \hline y_0 & a_{00} & \cdots & a_{0j} & \cdots & a_{0p} \\ \vdots & & & \ddots & & \\ y_i & a_{i0} & \cdots & a_{ij} & \cdots & a_{ip} \\ \vdots & & & \ddots & & \\ y_m & a_{m0} & \cdots & a_{mj} & \cdots & a_{mp} \end{array}, \tag{4}$$

where $m = 2^{\tau} - 1$, $p = 2^{n-\tau} - 1$, $\forall i, j$ $a_{ij} = \begin{cases} 1, \text{if } F(y_i, z_j) = 1, \\ 0, \text{if } F(y_i, z_j) = 0. \end{cases}$

Definition 1. Subset $S_{F^{\varepsilon}}^{z_j}$ of set Y, where

$$S_{F^{\varepsilon}}^{z_j} = \{y_i : \forall y_i \in Y, F(y_i, z_j) = \varepsilon,\ \varepsilon \in \{0,1\}\}, \tag{5}$$

is called a *subset of column* of function $F(x_1, \dots, x_n)$ for the column z_j.

According to this definition, a Boolean function can be represented using a table of subsets of columns (Table 2):

Table 3.1.2. Boolean function's table of subsets of columns

Table 2 Boolean function's table of subsets of columns

	z_0	…	z_j	…	z_p
F^0	$S_{F^0}^{z_0}$	…	$S_{F^0}^{z_j}$	…	$S_{F^0}^{z_p}$
F^1	$S_{F^1}^{z_0}$	…	$S_{F^1}^{z_j}$	…	$S_{F^1}^{z_p}$

Source: based on Zgureanu (2010)

It is obvious that $S_{F^0}^{z_j} = Y \setminus S_{F^1}^{z_j}$ and usually the subsets $S_{F^0}^{z_j}$, $j = \overline{0,p}$ are not included in the Table 2.

2.2 Subsets of columns of Boolean functions represented algebraically

Every Boolean function can be represented by an algebraic expression. Several properties of some of these representations are shown in Theorems 1, 2 and 3 (Zgureanu, 2011) and allow us to use the representation of subsets of columns when the Boolean function is expressed by a disjunction, conjunction, or a symmetric difference of some Boolean functions.

Theorem 1 If

$$F(x_1, \dots, x_\tau, x_{\tau+1}, \dots, x_n) = F_1 \vee \dots \vee F_i \vee \dots \vee F_k, \tag{6}$$

then

$$S_{F^1}^{z_j} = \bigcup_{i=1}^{k} S_{F_i^1}^{z_j},\ \ \forall j \in \{0, \dots, 2^{n-\tau} - 1\}. \tag{7}$$

Theorem 2 If

$$F(x_1, \dots, x_\tau, x_{\tau+1}, \dots, x_n) = F_1 \wedge \dots \wedge F_i \wedge \dots \wedge F_k, \tag{8}$$

then

$$S_{F^1}^{z_j} = \bigcap_{i=1}^{k} S_{F_i^1}^{z_j},\ \ \forall j \in \{0, \dots, 2^{n-\tau} - 1\}. \tag{9}$$

Theorem 3 If

$$F(x_1,\dots,x_\tau,x_{\tau+1},\dots,x_n) = F_1 \oplus\ldots\oplus F_i \oplus\ldots\oplus F_k, \tag{10}$$

then

$$S_{F^1}^{z_j} = \mathop{\Delta}_{i=1}^{k} S_{F_i^1}^{z_j}, \quad \forall j \in \{0,\dots,2^{n-\tau}-1\}, \tag{11}$$

where Δ denotes the symmetric difference of the subsets of columns.
A special interest represents Boolean functions in their disjunctive normal form:

$$F(x_1,\dots,x_n) = U_1 \vee\ldots\vee U_i \vee\ldots\vee U_k, \tag{12}$$

where

$$U_i = x_{i_1}^{\sigma_{i_1}} \wedge\ldots\wedge x_{i_s}^{\sigma_{i_s}} \wedge x_{j_1}^{\sigma_{j_1}}\ldots\wedge x_{j_q}^{\sigma_{j_q}}, \tag{13}$$

and

$$x_a^{\sigma_a} = \begin{cases} x_a, \text{if } \sigma_a = 1, \\ \bar{x}_a, \text{if } \sigma_a = 0, \end{cases} a \in \{i_1,\dots,i_s,j_1,\dots,j_q\}, x_{i_1},\dots,x_{i_s} \in \tilde{X}_1, x_{j_1},\dots,x_{j_q} \in \tilde{X}_2. \tag{14}$$

In (Zgureanu, 2011) was shown that if the subsets of columns of each conjunction are known, we can find out $S_{F^1}^{z_j}$, $\forall j \in \{0,\dots,2^{n-\tau}-1\}$ and so we can calculate easily all subsets of columns of a Boolean function given in its disjunctive normal form (Table 3).

There are three different cases related to the U_i structure. These cases are considered in Theorems 4 – 6 (Zgureanu, 2011).

Theorem 4 Let

$$U_i = x_{j_1}^{\sigma_{j_1}} \wedge\ldots\wedge x_{j_q}^{\sigma_{j_q}}, x_{j_1},\dots,x_{j_q} \in \tilde{X}_2. \tag{15}$$

Then

$$S_{U_i^1}^{z_j} = \begin{cases} \{0,\dots,2^\tau-1\}, \text{if } \forall a \in \{1,\dots,q\}: x_{j_a} = \sigma_{j_a}, \\ \emptyset, \text{if } \exists a \in \{1,\dots,q\}: x_{j_a} \neq \sigma_{j_a}. \end{cases} \tag{16}$$

Table 3 Table of subsets of columns of a Boolean function in disjunctive normal form

	z_0		z_j		z_p
	$S_{u_1^1}^{z_0}$		$S_{u_1^1}^{z_j}$		$S_{u_1^1}^{z_p}$
	$S_{u_2^1}^{z_0}$		$S_{u_2^1}^{z_j}$		$S_{u_2^1}^{z_p}$
	$\vdots$		$\vdots$		$\vdots$
	$S_{u_k^1}^{z_0}$	..	$S_{u_k^1}^{z_j}$		$S_{u_k^1}^{z_p}$
	$S_{F^1}^{z_0} = \bigcup_{i=1}^{k}$	$S_{u_i^1}^{z_0}.$	$S_{F^1}^{z_j} = \bigcup_{i=1}^{k}$	$S_{u_i^1}^{z_j}$	$S_{F^1}^{z_p} = \bigcup_{i=1}^{k} S_{u_i^1}^{z_p}$

Source: based on Zgureanu (2011)

Theorem 5 Let

$$U_i = x_{i_1}^{\sigma_{i_1}} \wedge\ldots\wedge x_{i_s}^{\sigma_{i_s}}, x_{i_1},\dots,x_{i_s} \in \tilde{X}_1. \tag{17}$$

Then

$$S_{U_i^1}^{z_j} = \bar{m}\begin{matrix}\sigma_{i_1} & \cdots & \sigma_{i_s}\\ i_1 & \cdots & i_s\end{matrix}, \forall j \in \{0,1,\dots,2^{n-\tau}-1\}. \qquad (18)$$

Theorem 6 Let

$$U_i = x_{i_1}^{\sigma_{i_1}} \wedge \dots \wedge x_{i_c}^{\sigma_{i_c}} \wedge x_{j_1}^{\sigma_{j_1}} \dots \wedge x_{j_q}^{\sigma_{j_q}}, \qquad (19)$$

where $x_{i_1},\dots,x_{i_c} \in \{x_1,\dots,x_\tau\}$ and $x_{j_1},\dots,x_{j_q} \in \{x_{\tau+1},\dots,x_n\}$.

Then

$$S_{U_i^1}^{z_j} = \begin{cases} \emptyset, \text{if } \exists\, a \in \{1,\dots,q\}: x_{j_a} \neq \sigma_{j_a}, \\ \bar{m}\begin{matrix}\sigma_{i_1} & \cdots & \sigma_{i_c}\\ i_1 & \cdots & i_c\end{matrix}, \text{if } \forall\, a \in \{1,\dots,q\}: \ x_{j_a} = \sigma_{j_a}. \end{cases} \qquad (20)$$

Theorems 1 – 6 allow to perform the transition from the algebraic representation of Boolean functions to their representation by subsets of columns. This transition makes it possible to perform operations with functions of a large number of variables.

3 Derivative Calculation of a Boolean Function

3.1 Derivative of a Boolean function

The Boolean Differential Calculus is a very powerful theory that significantly extends the basic concepts of Boolean Algebras. According to (Bochmann & Posthoff, 1981), the partial derivative of a Boolean function is defined as follow:

Definition 2 The partial derivative of an *n*-variable Boolean function $F(x_1,\dots,x_{i-1},x_i,x_{i+1},\dots,x_n)$ to the *i*-th variable x_i is another *n*-variable Boolean function $\frac{\partial F}{\partial x_i}$, defined as follows:

$$\frac{\partial F}{\partial x_i} = F(x_1,\dots,x_{i-1},1,x_{i+1},\dots,x_n) \oplus F(x_1,\dots,x_{i-1},0,x_{i+1},\dots,x_n). \qquad (21)$$

Also, according to (Bochmann & Posthoff, 1981), some of the properties of partial derivatives of a Boolean function are (22) – (24):

$$\frac{\partial(F_1 \vee F_2)}{\partial x_i} = \bar{F}_1 \wedge \frac{\partial F_2}{\partial x_i} \oplus \bar{F}_2 \wedge \frac{\partial F_1}{\partial x_i} \oplus \frac{\partial F_1}{\partial x_i} \wedge \frac{\partial F_2}{\partial x_i}, \qquad (22)$$

$$\frac{\partial(F_1 \wedge F_2)}{\partial x_i} = F_1 \wedge \frac{\partial F_2}{\partial x_i} \oplus F_2 \wedge \frac{\partial F_1}{\partial x_i} \oplus \frac{\partial F_1}{\partial x_i} \wedge \frac{\partial F_2}{\partial x_i}, \qquad (23)$$

$$\frac{\partial(F_1 \oplus F_2)}{\partial x_i} = \frac{\partial F_1}{\partial x_i} \oplus \frac{\partial F_2}{\partial x_i}. \qquad (24)$$

Partial derivatives of a higher order can be calculated using the formula (25)

$$\frac{\partial^s F}{\partial x_{i_1} \dots \partial x_{i_s}} = \frac{\partial}{\partial x_{i_s}}\left(\frac{\partial^{s-1} F}{\partial x_{i_1} \dots \partial x_{i_{s-1}}}\right), \qquad (25)$$

consecutively calculating

$$\frac{\partial F}{x_{i_1}}, \frac{\partial^2 F}{\partial x_{i_1} \partial x_{i_2}}, \dots, \frac{\partial^s F}{\partial x_{i_1} \partial x_{i_2} \dots \partial x_{i_s}}. \qquad (26)$$

However, finding partial derivatives according to (22) – (24) is associated with complex calculations. These calculations can be greatly simplified using Boolean functions represented as subsets of columns.

3.2 Derivative calculation of a Boolean function represented as subsets of columns

The result of calculating the derivative does not depend on the order of the variables by which the derivative is calculated (Bochmann & Posthoff, 1981). These calculations can be simplified by applying Theorems 7 – 10 (Bulat, et al., 2010), (Zgureanu, 2011), which play an essential role in calculating the derivatives of Boolean functions and offer a more accessible way than the mentioned traditional formulas.

Theorem 7 Let $x_{i_1}, \dots, x_{i_s} \in \tilde{X}_2$, $j \in \{0,1,\dots,2^{n-\tau}-1\}$. Then

$$S^{z_j}_{\left(\frac{\partial^s F}{\partial x_{i_1}\partial x_{i_2}\dots\partial x_{i_s}}\right)^1} = S^{j_1}_{F^1} \Delta S^{j_2}_{F^1} \Delta \dots \Delta S^{j_k}_{F^1}, \tag{27}$$

where Δ is the symmetric difference of subsets of columns, $k = 2^s$, $j_1, \dots, j_k \in \{0,1,2,\dots,2^{n-\tau}-1\}$, $j \in \{j_1, \dots, j_k\}$ for each z_{j_a}, $a = \overline{1,k}$ there are s neighbouring binary sets by the variables $x_{i_1}, \dots, x_{i_s}$.

Formula (3.1.10) allows calculating the partial derivative of the order s without calculating the derivatives of the lower orders. In order to do this, we need to find the elements of the set $J = \{j_1, j_2, j_3, \dots, j_k\}$, where $k = 2^s$ and $j_1 = j$. If we know j, the other elements can be calculated using the algorithm proposed in (Bulat, et al., 2010).

Definition 3 Two sets $\tilde{\alpha} = (\alpha_1, \dots, \alpha_{k-1}, 0, \alpha_{k+1}, \dots, \alpha_n)$ and $\tilde{\beta} = (\beta_1, \dots, \beta_{k-1}, 1, \alpha_{\beta+1}, \dots, \beta_n)$, differing only in the kth component are called neighbouring sets by the component k (Bochmann & Posthoff, 1981).

According to (Булат, 2002), if $x_i \in \tilde{X}_2$ and z_j and z_k are neighbouring binary sets by the variable x_i, then

$$S^{z_j}_{\left(\frac{\partial F}{\partial x_i}\right)^1} = S^{z_k}_{\left(\frac{\partial F}{\partial x_i}\right)^1} = S^{z_j}_{F^1} \,\Delta\, S^{z_k}_{F^1}, \tag{28}$$

where Δ is the symmetric difference of sets.

Theorem 8 Let $x_i \in \tilde{X}_1$ and $S^{z_j}_{F^1} = \{\alpha_1, \alpha_2, \dots, \alpha_k\}$. Then

$$S^{z_j}_{\left(\frac{\partial F}{\partial x_i}\right)^1} = \{\alpha_1, \beta_1\} \,\Delta\, \{\alpha_2, \beta_2\} \,\Delta\, \cdots \Delta \{\alpha_k, \beta_k\}, \tag{29}$$

where $\beta_1, \beta_2, \dots, \beta_k$ are neighbours with $\alpha_1, \alpha_2, \dots, \alpha_k$ respectively, by the variable x_i.

Corollary 1 If $S^{z_j}_{F^1} = \{0,1,2,\dots,2^\tau-1\}$ or $S^{z_j}_{F^1} = \emptyset$, then

$$S^{z_j}_{\left(\frac{\partial F}{\partial x_i}\right)^1} = \emptyset, \ \forall i \in \{1,\dots,\tau\}. \tag{30}$$

The calculation of partial derivatives could be improved as is shown in the next four Theorems 8 - 11 (Bulat, 2013):

Theorem 8 If

$$S^{z_j}_{F^1} = \bigcup_{i=1}^{m} Y_i, Y_i \subset Y = \{0,1,2,\dots,2^\tau-1\}, \tag{31}$$

then for any $a \in \{1,\dots\tau\}$

$$S^{z_j}_{\left(\frac{\partial F}{\partial x_a}\right)^1} = S^{z_j}_{\left(\frac{\partial F_1}{\partial x_a}\right)^1} \Delta \cdots \Delta S^{z_j}_{\left(\frac{\partial F_m}{\partial x_a}\right)^1} \Delta S^{z_j}_{\left(\frac{\partial (F_1 \wedge F_2)}{\partial x_a}\right)^1} \Delta \cdots \Delta S^{z_j}_{\left(\frac{\partial (F_1 \wedge F_2 \wedge \dots \wedge F_m)}{\partial x_a}\right)^1}, \tag{32}$$

where

$$S^{z_j}_{F_1^1} = Y_1, \dots, S^{z_j}_{F_m^1} = Y_m, S^{z_j}_{(F_1 \wedge F_2)^1} = Y_1 \cap Y_2, \dots\dots, S^{Z_j}_{(F_1 \wedge F_2 \wedge \dots \wedge F_m)^1} = Y_1 \cap Y_2 \cap \dots \cap Y_m. \quad (33)$$

Theorem 9 If $S^{z_j}_{F^1} = \bar{m}^{\sigma_i}_{i}$ then

$$S^{z_j}_{\left(\frac{\partial F}{\partial x_a}\right)^1} = \begin{cases} \emptyset, if \;\; a \neq i, \\ Y, if \;\; a = i, \end{cases} \forall i, a \in \{1,2,\dots,\tau\}, \forall \sigma_i \in \{0,1\}. \quad (34)$$

Theorem 10 Let $S^{z_j}_{F^1} = \underset{a=1}{\overset{k}{\Delta}} \bar{m}^1_{i_a}$, where $i_a \in \{1,2,\dots,\tau\}$. Then

$$S^{z_j}_{\left(\frac{\partial F}{\partial x_b}\right)^1} = \begin{cases} \emptyset, \;\; if \;\; b \notin \{i_1,\dots,i_k\}, \\ Y, \;\; if \;\; b \in \{i_1,\dots,i_k\}. \end{cases} \quad (35)$$

Theorem 11 If $S^{z_j}_{F^1} = \bar{m}^{\sigma_{i_1} \dots \sigma_{i_k}}_{i_1 \dots i_k}$, where $i_1,\dots,i_k \in \{1,\dots,\tau\}$, and $\sigma_{i_1},\dots,\sigma_{i_k} \in \{0,1\}$.

Then

$$S^{z_j}_{\left(\frac{\partial F}{x_b}\right)^1} = \begin{cases} \emptyset, \;\; if \;\; b \notin \{i_1,\dots,i_k\}, \\ \bar{m}^{\sigma_{i_1}}_{i_1} \cap \dots \cap \bar{m}^{\sigma_{i_{b-1}}}_{i_{b-1}} \cap \bar{m}^{\sigma_{i_{b+1}}}_{i_{b+1}} \cap \dots \cap \bar{m}^{\sigma_{i_k}}_{i_k}, if \;\; b \in \{i_1,\dots,i_k\}. \end{cases} \quad (36)$$

Applying Theorems 8 – 11 we can calculate all subsets of columns $S^{z_j}_{F^1}$ of Boolean function derivatives much more easily.

Now suppose the function F is represented as

$$F(x_1,\dots,x_n) = U_1 \oplus \dots \oplus U_i \oplus \dots \oplus U_k, \quad (37)$$

where

$$U_i = x^{\sigma_{i_1}}_{i_1} \wedge \dots \wedge x^{\sigma_{i_s}}_{i_s} \wedge x^{\sigma_{j_1}}_{j_1} \dots \wedge x^{\sigma_{j_q}}_{j_q}, x^{\sigma_a}_a = \begin{cases} x_a, \text{if } \sigma_a = 1 \\ \bar{x}_a, \text{if } \sigma_a = 0 \end{cases}, a \in \{i_1,\dots,i_s,j_1,\dots,j_q\},$$

$$x_{i_1},\dots,x_{i_s} \in \tilde{X}_1, \; x_{j_1},\dots,x_{j_q} \in \tilde{X}_2. \quad (38)$$

According to (Zgureanu & Cataranciuc, 2010), (Bulat, et al., 2012b), each conjunction U_i, through the variable $x_{i_1},\dots,x_{i_s} \in \tilde{X}_1$ determines a block, as shown below:

$$\bar{m}^{\sigma_{i_1} \dots \sigma_{i_s}}_{i_1 \dots i_1} = \bar{m}^{\sigma_{i_1}}_{i_1} \cap \dots \cap m^{\sigma_{i_s}}_{i_s}, \quad (39)$$

where

$$i_1,\dots,i_s \in \{1,\dots,\tau\}, \;\; \sigma_{i_1},\dots,\sigma_{i_s} \in \{0,1\}. \quad (40)$$

Variables $x_{j_1},\dots,x_{j_q} \in \tilde{X}_2$ determine the index z for which

$$S^{z_j}_{u_i^1} = \bar{m}^{\sigma_{i_1} \dots \sigma_{i_s}}_{i_1 \dots i_1}. \quad (41)$$

All the subsets of columns of the conjunctions U_i from (37) are shown in the Table 4. According to (Zgureanu & Cataranciuc, 2010), (Bulat, et al., 2012b), the last row of Table 4 contains the subsets of columns of the function F. Applying formulas (28) and (29), we can calculate the subsets of columns of the corresponding derivatives. The subsets of columns of the conjunctions U_i represent a block of partitions or the empty set $(\emptyset)$ or the universe (Y). Applying Theorem 9 and Theorem 11 and the property

$$S^{z_j}_{\left(\frac{\partial (F_1 \oplus F_2)}{\partial x_i}\right)^1} = S^{z_j}_{\left(\frac{\partial F_1}{\partial x_i}\right)^1} \Delta S^{z_j}_{\left(\frac{\partial F_2}{\partial x_i}\right)} \quad (42)$$

we can find $S^{z_j}_{\left(\frac{\partial F}{\partial x_i}\right)^1}$ without calculating $S^{z_j}_{F^1}$. This way, we can reduce the number of operations to calculate the subsets of columns of partial derivatives and, in consequence, reduce the time to calculate them.

Table 4 Table of subsets of columns of the conjunctions u_i

	z_0		z_j		z_p
	$S^{z_0}_{u_1^1}$		$S^{z_j}_{u_1^1}$		$S^{z_p}_{u_1^1}$
	$S^{z_0}_{u_2^1}$		$S^{z_j}_{u_2^1}$		$S^{z_p}_{u_2^1}$
	$\vdots$		$\vdots$		$\vdots$
	$S^{z_0}_{u_k^1}$	..	$S^{z_j}_{u_k^1}$		$S^{z_p}_{u_k^1}$
F^1	$S^{z_0}_{F^1} = \underset{i=1}{\overset{k}{\Delta}} S^{z_0}_{u_i^1}$	..	$S^{z_j}_{F^1} = \underset{i=1}{\overset{k}{\Delta}} S^{z_j}_{u_i^1}$		$S^{z_p}_{F^1} = \underset{i=1}{\overset{k}{\Delta}} S^{z_p}_{u_i^1}$

Source: based on Bulat *et al.* (2012)

4 A key exchange algorithm based on Boolean functions derivatives

One of the pressing problems of the information society is the secure transportation and storage of information using different computer technologies and local or global computer networks. An essential tool in information security represents cryptography, which covers encryption algorithms, protocols, cryptographic key handling facilities, etc. To obtain a reliable information protection system, it is necessary to anticipate many possible attacks on it because it is useless to protect one side of the system when the attack can be easily made on other, more sensitive side.

A cryptographic system is effective when it keeps a balance between what is necessary at the moment and what is possible in the future. In order to create such a system, it is needed to build a better infrastructure, which contains the following components: symmetric and asymmetric cryptographic algorithms, hash algorithms, digital signature, the required key infrastructure and many other things.

In practice, when the problem of implementing such a system arises, we choose between one of the two possibilities: either select an existing solution or create a new one. Each of these ways has advantages and disadvantages: existing solutions have been studied by experts in theory and applied in practice, therefore they would probably be safer. The problem is that these systems may not be suitable for our goals or for the information system we use. Therefore, there is often necessary to create a new algorithm or system, which is tuned to our needs.

Along with the improvement of computing facilities, the requirements for cryptosystems are also growing. The size of keys grows incredibly and arithmetic operations with very large numbers seriously slow down the operation speed of cryptosystems and, as a result, their performance decreases (Zgureanu, 2011), (Bulat, et al., 2008a). The situation may change when logical operations on Boolean functions are used instead of arithmetic operations on large numbers. Such solutions were proposed in (Bulat, 2013), (Zgureanu, 2010), (Bulat, et al., 2012a), (Bulat, et al., 2010).

Taking this into account, and based on the theory form chapters 2 and 3, we propose a concept of a key exchange algorithm between two parties. This algorithm use Boolean functions represented as subsets of columns and it is based on the complexity of the problem of determining a Boolean function, knowing only some of its partial derivatives.

As we know, the mainly common methods of key exchange are: using a Key Distribution Centre (KDC), which is a symmetric-key technique, or using a Certification Authority (CA) which provide certificates based on an asymmetric-key technique. Both of these approaches have advantages and disadvantages. For example, Kerberos, which is in widespread use is vulnerable to weak or repeated passwords; it only provides authentication for services and clients. On the other hand, the main disadvantage of asymmetric-key exchange techniques is that they are much computationally demanding than symmetric ones.

The algorithm we propose here also use certificates, but quite more rarely - only to certificate the long-lived secret key *k*, which will be used for session key generating.

Secret key generation. The secret key consists of the family of specific properties Boolean functions of *n* variables $F = \{F_1, F_2, \dots, F_g\}$ and will be used for session keys generation only. The secret key should be randomly generated by one party, then signed and sent to the other party through a protected channel (e.g., encrypted). The recipient will decrypt it, verify the signature and, after signature validation, both parties could start using it for session keys k_{ses} generation for a fixed, but quite a long period of time. In addition to the family *F* there should be generated a number τ, which define the partitions for subset of column representation of functions F_i and of their derivatives.

Session key generation. The main idea in session key generating is that k_{ses} represent a randomly chosen subset of column of a randomly chosen function F_i from the secret key *k*. For this the party which initiates the session should randomly choose three integers (n_1, n_2, n_3), where:

- $1 \le n_1 \le g$ and n_1 represents the index of function F_{n_1} in the secret key $k = F = \{F_1, F_2, \dots, F_g\}$;
- $1 \le n_2 \le 2^n$ and n_2 in its binary form, defines the variables with respect to which we calculate the partial derivative taking in mind only units indexes;
- $1 \le n_3 \le p$ and n_3 represents the index *z* of the subset of column of the derivative of the function F_{n_1}, where $p = 2^{n-\tau} - 1$.

The triplet (n_1, n_2, n_3) uniquely define a subset of column of the function F_{n_1} partial derivative. This subset of column is sent to the recipient unencrypted and is used to calculate the secret session key k_{ses}.

Let’s explain how to obtain k_{ses} for the triplet (13, 53, 9) and a family F which consists of functions of 10 variables (n = 10). Firstly, we choose the function F_{13} from the secret key k. Then we calculate the binary form of n_2, taking in mind that we have 10 variables, therefore we will use 10 bit’s binary representation for n_2: $53_{10} = 0000110101_2$. In this number, the units are situated in 4 positions: 5, 6, 8 and 10. Finally, $n_3 = 9$ define the column z_9. This means that for the function F_{13} we should calculate the subset of column z_9 of the partial derivative of the order 4 with respect to the variables x_5, x_6, x_8 and x_{10}:

$$S^{z_9}\left(\frac{\partial^4 F}{\partial x_5 \partial x_6 \partial x_8 \partial x_{10}}\right)^{1}.$$

This derivative, in its binary representation, is the material from which we choose for example 128, 192 or 256 bits for an AES encryption session.

5 Conclusion

The elaborated algorithm allows us to use public key cryptography only once for a long period, to validate the secret key generated by one party involved in communication process. This implies less computational necessities when the two parties exchange the session key, using only subset of column calculation and not using very demanding Diffie-Hellman or RSA key exchange every time. The cryptographic strength is based on the complexity of calculating the subsets of columns that were used for encryption without knowing the functions F_i in the secret key $k = F = \{F_1, F_2, \dots, F_g\}$.

On the other hand, to make this algorithm secure, further research on some additional parameters of the secret key is needed: the quantity of functions F_i in the secret key (secret key dimension), number n of their variables, bounds of τ. At the same time there are some necessary conditions that functions F_i must satisfy: the subsets of columns of derivatives of F_i must contain the lowest possible number of elements with the same minimal number of units in their binary representation and using only of so-called bent-functions in secret key generation [7]. In addition to this, further research on the functions F_i is needed in order to obtain as different as possible subsets of columns of their partial derivatives.

Acknowledgment: The authors would like to express appreciation for the support of the sponsors [Project Number = 20.80009.5007.26 Models, algorithms and technologies for control, optimization and protection of cyber-physical systems (MATCOSC)].

REFERENCES

1. Bochmann, D. & Posthoff, C., 1981. *Binäre dynamische Systeme.* Berlin: Academic-Verlag.
2. Bulat, M., 2013. *About applying derivatives of Boolean functions in encryption systems.* Chisinau, NCAA, pp. 247-254.
3. Bulat, M., Zgureanu, A., Cataranciuc, S. & Ciobanu, I., 2012a. *Encryption systems with wandering keys.* Chisinau, NCAA, pp. 238-246.
4. Bulat, M., Zgureanu, A., Cataranciuc, S. & Ciobanu, I., 2012b. *Криптосистемы на базе ряда Тейлора для булевых функций.* Chisinau, ATIC, pp. 229-251.
5. Bulat, M., Zgureanu, A., Ciobanu, I. & Bivol, L., 2008a. *Vector key encryption systems.* Chisianu, ATIC, pp. 281-285.
6. Bulat, M., Zgureanu, A., Ciobanu, I. & Bivol, L., 2008b. *Крипто-системы на базе n-арных отношений.* Москва, ИПУ РАН, pp. 66-67.
7. Bulat, M., Zgureanu, A., Ciobanu, I. & Izbaş, V., 2010. *Sistem de criptare bazat pe derivarea funcţiilor booleene*. Chişinău, ATIC, pp. 132-141.
8. Zgureanu, A., 2010. Information encryption systems based on Boolean functions. *Computer Science Journal of Moldova*, 3(54), pp. 319-335.

9. Zgureanu, A., 2011. *Securitatea informațională și metode de criptare bazate pe mulțimi de relații multi-are,* Chisinau: AZ.
10. Zgureanu, A. & Cataranciuc, S., 2010. Encryption systems based on multidimensional matrices. *Annals of the Tiberiu Popoviciu seminar,* 8(1), pp. 99-110.
11. Булат, М., 1972. Синтез таблиц состояний функций возбуждения внутренних элементов автомата. *Абстрактная и структурная теория релейных устройств,* pp. 49-64.
12. Булат, М., 2002. Об одном способе дифференцирования булевых функций. *Annals of ATIC - 2001 I(I)*, pp. 40-47.
13. Булат, М. & Горюк, И., 1978. *Синтез логических структур.* Chișinău: UTM.