IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 September 2023, accepted 12 October 2023, date of publication 27 October 2023, date of current version 2 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3328215

RESEARCH ARTICLE

Using Linearizing Sets to Solve Multivariate Quadratic Equations in Algebraic Cryptanalysis

ALEXANDER SEMENOV[1], KIRILL ANTONOV[2], STEPAN KOCHEMAZOV[1], AND ARTEM PAVLENKO[1]

[1]Information Technologies and Programming Faculty, ITMO University, 197101 Saint Petersburg, Russia
[2]Department of Cryptography and Computer Systems Security, MEPhI University, 115409 Moscow, Russia

Corresponding author: Alexander Semenov (alex.a.semenov@itmo.ru)

This work was supported by the Ministry of Science and Higher Education of Russian Federation under Project 075-03-2020-139/2 (goszadanie no. 2019-1339).

ABSTRACT In this paper we describe a class of cryptographic guess-and-determine attacks which is based on the notion of a linearizing set. A linearizing set-based attack is applied to a system of Multivariate Quadratic equations (MQ) over $GF(2)$ field, which encodes how a considered cryptographic function works. By substituting into such MQ system a random (in some strict sense) assignment of variables from a linearizing set we aim to transform the system into a linear one. We introduce a probability of such an event and call it a probability of linearization. Then we describe a guess-and-determine attack, the hardness of which can be expressed via a probability of linearization. To estimate the latter it is possible to use a simple Monte Carlo algorithm. Also we describe a technique that allows to augment a considered MQ system by new linear equations and to construct a new MQ system, for which the probability of linearization is usually larger than that for an original one. For this purpose we apply a SAT oracle to a Boolean formula that is naturally associated with a considered MQ system. Finally, we reduce the problem of searching for a linearizing set that yields the best effectiveness of a constructed guess-and-determine attack to a pseudo-Boolean optimization problem, which can be solved using metaheuristic optimization algorithms. The important consequence of this is that this way we can construct guess-and-determine attacks automatically by solving the corresponding optimization problem. In the computational experiments we used the proposed methodology to construct attacks on several well-known stream ciphers. The runtime estimations of some of the attacks make it possible to implement them in reasonable time.

INDEX TERMS Algebraic cryptanalysis, guess-and-determine attack, system of multivariate quadratic equations, Boolean satisfiability, pseudo-Boolean optimization, evolutionary algorithm.

I. INTRODUCTION

The main object of the study in this paper are the Multivariate Quadratic systems of algebraic equations over $GF(2)$ field, or MQ-systems. They are interesting because for any total effectively computed discrete function (i.e. a function that transforms binary words into binary words) its inversion problem (also known as preimage finding problem or preimage attack) can be effectively reduced to MQ-system. The problem of solving an MQ-system or proving its inconsistency is NP-hard (see e.g. [1]). However, in many practical cases one can construct an algorithm for solving such systems, that has the complexity significantly smaller than that of a brute force attack. The corresponding methods can be useful for analysis of existing systems of lightweight cryptography or for the development of such systems.

In the present paper, we describe both theoretical and computational instruments, that can be employed for inverting an arbitrary discrete function specified by some fast algorithm. The proposed approach is based on the notion of a *linearizing set*: a set of variables of a considered system, such that assigning values to these variables in a system with some

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa M. Fouda.

probability transforms this system into linear. The probability we mention here is referred to as *probability of linearization*.

We would like to specifically note that the introduced term *linearizing set* is the base for constructing a special class of *guess-and-determine attacks* [2] and the corresponding attacks can *not* be viewed in the context of the widely known eXtended Sparse Linearization (XSL) method [3], [4]. Thus, the notion of "linearizing set" is not related to XS linearization and similar techniques.

To construct a preimage attack that can actually be used in practice, one needs a linearizing set of as small size as possible which at the same time has a relatively high probability of linearization. We propose an algorithm for finding such sets for an MQ-system associated with a considered cryptographic function. It optimizes a special fitness function which evaluates how good a specific set of variables is when viewed as a linearizing set candidate. The value of this fitness function can be viewed as the *hardness* of solving an MQ-system using this particular set of variables. Thus, we are looking for a linearizing set that yields an attack with minimal hardness. The introduced fitness function is minimized via metaheuristic optimization algorithm in a completely automatic mode. Apart from that we propose a new technique for generating additional linear equations that outline the connections between some of the variables in a considered MQ-system. For this purpose we invoke a SAT solver to check the consistency of additional constraints with an original MQ-system.

In the computational experiments we applied the proposed method to constructing the attacks on several lightweight stream ciphers, in particular, the A5/1, ASG, Bivium and Trivium keystream generators. The runtime estimations that we obtained for A5/1 and Bivium show that the corresponding attacks can be implemented in practice.

Thus, the main contributions of our paper are as follows.

1) We introduce a new class of guess-and-determine attacks based on a special technique for solving MQ-systems over $GF(2)$ field. The technique associates with an MQ-system, which encodes a considered cryptographic function, a set of linear systems over $GF(2)$ using a set of Boolean variables which is referred to as *linearizing set*.
2) With each linearizing set we naturally associate the notion called *probability of linearization*, using which we can define the hardness of the linearising set-based attack. We show that the probability of linearization and consequently the hardness of the corresponding attack can be effectively estimated using a simple Monte Carlo algorithm.
3) The problem of finding a linearizing set attack with the smallest runtime estimation is formulated as a pseudo-Boolean optimization problem and is solved using metaheuristic algorithms.
4) We also develop a special technique that can be used to augment the obtained linear systems over $GF(2)$ with new linear equations derived using a SAT oracle.
5) In the computational experiments we use the proposed method to automatically construct linearizing set-based attacks on a number of stream ciphers. In several cases the constructed attacks have a runtime estimation that allows one to implement them in reasonable time.

Let us give a brief outline of our paper. In Section II we introduce the necessary notions and basic results about Boolean circuits and satisfiability. In Section III we demonstrate the interconnection between cryptanalysis problem considered as an instance of the Boolean satisfiability problem (SAT) and tne corresponding system of Multivariate Quadratic Equations over $GF(2)$. In Section IV we introduce the notion of a linearizing set, describe a class of guess-and-determine attacks based on linearizing sets and reduce the problem of constructing an attack with a good runtime estimation to a metaheuristic pseudo-Boolean optimization problem. In Section V we describe the technique employed for generating additional linear equations using a SAT oracle. Section VI contains the results of computational experiments and Section VII – the review of related work. Finally, in Section VIII we briefly discuss the obtained results, consider the strengths and limitations of the proposed methods, draw conclusions and outline the directions for future research.

II. PRELIMINARIES

Remind that $GF(2)$ is a field of two elements which are usually denoted as 0 and 1, and can be viewed as the remainders from division of an arbitrary natural number by 2. Thus, $GF(2) = \langle\{0, 1\}, \oplus, \wedge\rangle$, where $\oplus$ and $\wedge$ are the *operations* in the field. If we define these operations using a table, then we obtain exactly the binary Boolean functions $\oplus$ (xor) and $\wedge$ (conjunction). The expressions over $GF(2)$ that we consider are *polynomials* of the kind $P(x_1, \ldots, x_s)$, where x_i, $i \in \{1, \ldots, s\}$ are variables with values from $\{0, 1\}$, as well as algebraic equations of the kind $P(x_1, \ldots, x_s) = 0$ or $P(x_1, \ldots, x_s) = 1$.

Apart from expressions over $GF(2)$ we will consider *Boolean formulas*, i.e. the expressions over an alphabet that includes Boolean variables $X = \{x_1, \ldots, x_k\}$, braces and Boolean connectives ($\neg, \vee, \wedge, \oplus, \rightarrow$, etc.). The simplest Boolean formulas are formulas x and $\neg x$, where x is a Boolean variable and $\neg$ is the unary connective called *negation* ($\neg 1 = 0$, $\neg 0 = 1$). These formulas are called *literals*. A *clause* is a formula which is the disjunction of literals, e.g. $\neg x_1 \vee \neg x_{11} \vee \neg x_{118}$. A *Conjunctive Normal Form* (CNF) is a conjunction of different clauses. An arbitrary mapping of the kind $\alpha : X \rightarrow \{0, 1\}$ defines an *assignment* of variables from X. For a Boolean formula C an assignment is called *satisfying* if substituting it into C (see e.g. [5]) results in evaluating C to 1 (True). If a satisfying assignment exists, then a formula is called *satisfiable*, otherwise it is called *unsatisfiable*. The Boolean satisfiability problem (SAT) for formula C (usually in CNF) is formulated as follows: to decide whether the formula C is satisfiable. SAT is one of the classic NP-complete problems [6].

By $\{0,1\}^n$ for an arbitrary $n \in \mathbb{N}^+$ denote the set of all possible binary words of length n. Hereinafter, by

$$f : \{0,1\}^n \to \{0,1\}^m \tag{1}$$

denote a *discrete function*, that transforms an arbitrary binary word of length n into a binary word of length m. Thus, f is defined everywhere on $\{0,1\}^n$. By $Rangef$, $Rangef \subseteq \{0,1\}^m$ denote the *range* of values of f. An *inversion problem* for f (or *preimage attack* on f) consists in the following: given an arbitrary $\gamma \in Rangef$ to find such an $\alpha \in \{0,1\}^n$ that $f(\alpha) = \gamma$. Clearly, many cryptographic attacks can be considered in the context of the described inversion problem.

Below we assume that function (1) is specified by some deterministic algorithm (e.g. by some Turing machine program). It is a well known fact that using a technique of propositional encoding of algorithms which is based on the Cook-Levin theorem [1], one can construct a Boolean circuit over some complete basis, such that this circuit specifies function f. Denote such a circtuit as S_f. Essentially, S_f is a directed acyclic graph that contains n vertices without parents (*input nodes*) and m vertices without children (*output nodes*). Apart from that, S_f has a certain number of inner vertices. With each vertex in S_f which is not an input node, an element of some *complete basis* [7] is associated. Let us refer to such nodes as *gates*. Further, we consider the complete Boolean basis $\{\wedge, \neg\}$, containing only the connectives $\wedge$ (conjunction or logical and) and $\neg$ (negation or inversion). The corresponding circuit over basis $\{\wedge, \neg\}$ is often referred to as an *And-Inverter Graph* (AIG) [8].

Associate with each node of circuit S_f some Boolean variable. We denote the set of variables associated with inputs of S_f as $X^{in} = \{x_1, \ldots, x_n\}$. The set of variables associated with outputs of S_f is denoted as $Y = \{y_1, \ldots, y_m\}$.

Let us construct for S_f a Boolean formula in CNF using Tseitin transformation [9]. Namely, let g be an arbitrary gate of circuit S_f and v be a Boolean variable linked with gate g. Let g be an Inverter-gate and u be the Boolean variable linked with the parent of g. Then we associate with g a CNF-representation of formula $v \equiv \neg u$, i.e. CNF formula $(v \vee u) \wedge (\neg v \vee \neg u)$. Now, let g be an And-gate and u, w be the variables linked with its parents. Then we associate with g the CNF-representation of formula $v \equiv u \wedge w$, i.e. CNF formula $(v \vee \neg u \vee \neg w) \wedge (\neg v \vee u) \wedge (\neg v \vee w)$. Let us denote the CNF formula associated with g as C_g and consider the following CNF formula (constructed using all gates in S_f):

$$C_f = \bigwedge_{g \in S_f} C_g \tag{2}$$

Below, similar to [10] let us employ the following notation: by x^σ denote literal x if $\sigma = 1$ and literal $\neg x$ if $\sigma = 0$.

Remind that the Unit Propagation rule (UP) [11], is the main rule used for Boolean constraint propagation in the modern SAT solvers [12]. Let C be some CNF formula, every clause of which contains at least two literals. A single application of UP to a formula of the kind $x^\sigma \wedge C$ consists in the following: first remove from formula C all occurrences of literal $x^{\neg\sigma}$. Next remove from C all clauses that contain literal x^σ. Denote the resulting CNF formula as C'. The formulas $x^\sigma \wedge C$ and $x^\sigma \wedge C'$ are equisatisfiable. Formula C' can contain clauses that consist of a single (unit) literal of the kind x^τ. In this case it is said that such literals are *derived* from $x^\sigma \wedge C$ w.r.t. the Unit Propagation rule.

Following [13] we will refer to CNF formula C_f in (2) as to *template CNF* formula for function f. Let us denote by X the set of all variables occurring in C_f. Template CNF formulas possess an important property. Consider an arbitrary $\alpha \in \{0,1\}^n$, $\alpha = (\alpha_1, \ldots, \alpha_n)$ and construct the following formula (remind, that $X^{in} = \{x_1, \ldots, x_n\}$):

$$x_1^{\alpha_1} \wedge \ldots \wedge x_n^{\alpha_n} \wedge C_f \tag{3}$$

The following fact is well-known (see e.g. [14], [15]).

Theorem 1. *By iteratively applying UP to (3) we derive (in form of literals) the values of all variables from X. For the variables from $Y = \{y_1, \ldots, y_m\}$ the derived values $y_1 = \gamma_1, \ldots, y_m = \gamma_m$ are such that $f(\alpha) = \gamma$, where $\alpha = (\alpha_1, \ldots, \alpha_n)$, $\gamma = (\gamma_1, \ldots, \gamma_m)$. The derived assignment of variables from X does not contain any inconsistencies in form of pairs of complementary literals (i.e. literals of the kind $x, \neg x$).*

Informally, the Theorem 1 means, that the iterative application of UP to (3) effectively simulates the computing of function f on an arbitrary input α, based on the circuit S_f.

Now let us consider the following CNF formula

$$C_f(\gamma) = C_f \wedge y_1^{\gamma_1} \wedge \ldots \wedge y_m^{\gamma_m} \tag{4}$$

where $\gamma = (\gamma_1, \ldots, \gamma_m)$, $\gamma \in Rangef$. From the properties of Tseitin transformations it follows that (4) is satisfiable (see [13]). However, finding its satisfying assignment can be a very hard problem, especially if f is a cryptographic function with good resistance to attacks. Nevertheless, if one manages to find such a satisfying assignment, then it is easy to extract from it such an $\alpha \in \{0,1\}^n$ that $f(\alpha) = \gamma$.

III. MQ-SYSTEMS OVER $GF(2)$: SUBSTITUTIONS AND INFERENCE

Once again, consider function (1) and the circuit S_f over the basis $\{\wedge, \neg\}$ that defines it. With each node of this circuit we associate a variable $\tilde{x}$ which takes the values from $GF(2)$ field. Thus, we obtain the set of variables which we denote as $\tilde{X}$. There is a natural one-to-one correspondence between X and $\tilde{X}$. Moreover, in all cases that we consider further, a variable $x \in X$ takes the value of 1 if and only if the corresponding variable $\tilde{x} \in \tilde{X}$ takes the value of 1. Taking this fact into account, let us abuse the notation by denoting the variables which take values in $GF(2)$ and the corresponding Boolean variables by the same letters, and explicitly mention whenever necessary what case is considered. Thus, in this section the sets X, X^{in}, Y have the same meaning as above, with the only difference that the variables from these sets take the values from $GF(2)$.

Similar to the method that we employed to construct template CNF formula C_f, let us traverse circuit S_f and associate with each gate an algebraic equation over $GF(2)$. Let g be an arbitrary gate and let v be a variable with values from $GF(2)$ associated with g. Assume that g is an Inverter gate and u is a variable associated with the parent of g. Then, we construct for g the equation $u \oplus v = 1$. Now, let us assume that g is an And-gate and u, w are the variables associated with its parents. In this case the equation corresponding to g is $u \wedge w \oplus v = 0$. Denote the system of obtained algebraic equations over $GF(2)$ as E_f. It is clear, that E_f is an MQ-system over $GF(2)$. We will refer to E_f as to *template MQ-system* for function f. In accordance with the above we denote by X the set of variables occurring in E_f.

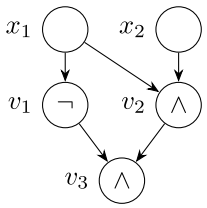

FIGURE 1. An example Boolean circuit with two inputs and 3 gates.

Consider an example at Fig. 1. It has two inputs associated with variables x_1 and x_2 (so, $X^{in} = \{x_1, x_2\}$) and three gates, with which the varables v_1, v_2, v_3 are associated. Let us construct an MQ-system over $GF(2)$ for this circuit. It will look as follows:

$$\begin{aligned} x_1 \oplus v_1 &= 1 \quad \text{negation-gate in the second layer;} \\ x_1 \wedge x_2 \oplus v_2 &= 0 \quad \text{and-gate in the second layer;} \\ v_1 \wedge v_2 \oplus v_3 &= 0 \quad \text{the output and-gate.} \end{aligned}$$

It is possible to substitute an assignment of values to variables in MQ-systems in a standard manner, i.e. by replacing the occurrences of a variable by the corresponding constant from $GF(2)$ and performing all possible elementary transformations. Sometimes, such substitutions can lead to derivations of equations of the kind $x = c$, where x is some variable and c is a constant from $GF(2)$. For example, consider the equation $u \wedge w \oplus v = 0$ and substitute to this equation the assignment $v = 1$. It is easy to see, that from this substitution we derive the equation $u \wedge w = 1$, from which we have two new equations $u = 1$, $w = 1$. In this case we can say that the values $u = 1$, $w = 1$ were *derived* from the considered MQ-system in accordance with the inference rule $\frac{u \wedge w \oplus v = 0, v = 1}{u = 1, w = 1}$. Hereinafter, we will use the standard formalism from mathematical logic (see e.g. [16]) locating the premise over the line and conclusion under the line.

Essentially, Theorem 1 makes it possible to use UP to derive from C_f the values of all variables in C_f by substituting to C_f the values of input variables. Similarly, we can define the set of inference rules, which will result in derivation of values of all variables in E_f when substituting the values of input variables. In particular, let us consider the following inference rules:

$$\frac{u \oplus v = 1, u = 0(1)}{v = 1(0)}; \quad \frac{u \oplus v = 0, u = 0(1)}{v = 0(1)}; \\ \frac{u \wedge w \oplus v = 0, u = 1}{w \oplus v = 0}; \quad \frac{u \wedge w \oplus v = 0, u = 0}{v = 0}. \qquad (5)$$

For a template system E_f we can define an analogue of (3) as a system produced from E_f by adding to it the equations of the kind $x_i = \alpha_i$, $i \in \{1, \ldots, n\}$ ($X^{in} = \{x_1, \ldots, x_n\}$). Denote the obtained system as $E_{f,\alpha}$, where $\alpha = (\alpha_1, \ldots, \alpha_n)$. The immediate analogue of Theorem 1 in the context of MQ-system is the following fact.

Theorem 2. *By iteratively applying the rules (5) to equations from the system $E_{f,\alpha}$ we derive (in form of equations of the kind $x = c$) the values of all variables from X. For the variables from $Y = \{y_1, \ldots, y_m\}$ the derived values $y_1 = \gamma_1, \ldots, y_m = \gamma_m$ are such that $f(\alpha) = \gamma$, where $\alpha = (\alpha_1, \ldots, \alpha_n)$, $\gamma = (\gamma_1, \ldots, \gamma_m)$. The derived assignment of variables from X does not contain any contradictions (e.g. pairs of equations of the kind $x = 0$, $x = 1$).*

Proof sketch: The basic idea of the proof is based on the notion of the *interpretation* of circuit S_f on a particular input $\alpha \in \{0,1\}^n$. Briefly, the interpretation is defined as consequent computation of values of gates, when with an input vertex number $i, i \in \{1, \ldots, n\}$ of circuit S_f there is associated a corresponding $\alpha_i \in \{0,1\}$. In this case by traversing through a circuit let us assign to each gate some value from $GF(2)$. We will refer to such value as to the value of the corresponding gate on the input word α. If e is an equation associated with gate g and the parents of this gate have some values, then the value of gate g is derived from e using one of the rules (5). As a result, by traversing the entire circuit S_f we obtain the values of all gates in form of equations $x = 0$ or $x = 1$, where x is a variable associated with a considered gate. Each gate will be assigned a single value, so there will be no contradictory equations in the obtained system. □

Thus, the substitution of an input $\alpha \in \{0,1\}^n$ in form of equations of the kind $x_i = \alpha_i$, $i \in \{1, \ldots, n\}$ and application of the rules (5) results in effective derivation of values of all the remaining variables in E_f. Substituting the values of a specific output $\gamma \in Range\, f$ in form of equations $y_j = \gamma_j, j \in \{1, \ldots, m\}$ results in a system that we denote as $E_f(\gamma)$. This system is consistent, but the problem of finding its solutions can be very hard for a cryptographically resistant function f. Nevertheless, if the solution of $E_f(\gamma)$ can be found, then one can effectively extract from it such an $\alpha \in \{0,1\}^n$ that $f(\alpha) = \gamma$.

IV. CRYPTOGRAPHIC ATTACKS BASED ON LINEARIZING SETS

The attacks described in the present section belong to the class of guess-and-determine attacks [2]. In such attacks there is some system of algebraic equations or a formula over variables from a set X. The goal is to find such a subset

$B \subseteq X$ that by substituting to a system or to a formula all $2^{|B|}$ assignments of variables from B and solving the resulting subproblems using some algorithm A we spend less resources than by attacking an original function using the brute-force method: in case of a function of the kind (1) we have a trivial brute force attack with $O(2^n)$ calls of function f. The set B is called a *guessed bits set*. One of the key points when constructing guess-and-determine attacks is how to choose an algorithm A. In form of A one can employ some polynomial algorithm or a combinatorial algorithm that is effective on a large number of practical problems of high dimension. The modern SAT solvers belong to the latter category and are employed in cryptography increasingly often (see Section VII). However, in the present paper we will assume that A is some polynomial algorithm for solving systems of linear equations over $GF(2)$ (for example, the well-known Gaussian Elimination algorithm [17]).

A. LINEARIZING SET NOTION

Hereinafter, by $\{0,1\}^{|B|}$ we denote the set of all possible assignments of all variables from set B. Informally, the basic idea of the attack described below in detail looks as follows. We aim to find such a guessed bits set B that substituting an assignment $\beta \in \{0,1\}^{|B|}$ to system $E_f(\gamma)$, $\gamma \in Range\, f$ with high probability results in a linear system. We measure the hardness of a corresponding attack using B in the number of the linear systems that need to be solved. Below let us formally describe the attacks of this kind.

So, consider a total function $f : \{0,1\}^n \to \{0,1\}^m$ and the problem of its inversion: for a known $\gamma \in Range\, f$ to find $\alpha \in \{0,1\}^n$ such that $f(\alpha) = \gamma$. Assume that f is defined by circuit S_f over the basis $\{\wedge, \neg\}$. Let us construct based on circuit S_f the corresponding template MQ-system E_f over the set of variables X associated with circuit S_f in a manner described above. Let B be an arbitrary subset of X : $B \subseteq X$. Denote by $E_f[\beta/B]$ the system over $GF(2)$ obtained by substituting into E_f the assignment $\beta \in \{0,1\}^{|B|}$ and applying to it the rules (5) as well as the rule

$$\frac{u \wedge w \oplus v = 0, v = 1}{u = 1, w = 1} \tag{6}$$

Let α be an arbitrary vector from $\{0,1\}^n$ which is considered as an assignment of variables from X^{in}. Associate with α the set of values Γ_α ($\Gamma_\alpha \in \{0,1\}^{|X|}$) of all variables from E_f derived (w.r.t Theorem Theorem 2) by substituting α into E_f. For an arbitrary $B \subseteq X$ we say that an assignment of variables from B in Γ_α is *induced* by α and denote such an assignment as β_α. Assume that $\gamma_\alpha = f(\alpha)$ is an assignment of variables from Y induced by α.

Define the probability space Σ in which the sample space is $\{0,1\}^n$, and assign to each $\alpha \in \{0,1\}^n$ the probability $\frac{1}{2^n}$, thus defining the uniform distribution over $\{0,1\}^n$. Consider an arbitrary $B \subseteq X$. Denote by "$E_f[\beta_\alpha/B, \gamma_\alpha/Y] \in Lin$" the fact that after substitution of assignments β_α and γ_α (of variables from B and Y, respectively), which are induced by $\alpha \in \{0,1\}^n$, into E_f, and subsequent application of all inference rules mentioned above, we obtain a linear system over $GF(2)$. Denote the corresponding event in the event algebra of space Σ (i.e. in the set $2^{\{0,1\}^n}$) by I_B. It is clear that the probability of I_B is defined as follows:

$$\Pr\{I_B\} = \frac{\#\left\{\alpha : E_f[\beta_\alpha/B, \gamma_\alpha/Y] \in Lin\right\}}{2^n}$$

Definition 1. *In application to an arbitrary set $B \subseteq X$ we will say that B linearizes the system E_f with probability $\rho_B = \Pr\{I_B\}$ or that B is a linearizing set for E_f with probability of linearization ρ_B.*

With an arbitrary set $B \subseteq X$ let us associate a Bernoulli random variable $\xi(B) : \{0,1\}^n \to \{0,1\}$ which is equal to 1 on $\alpha \in \{0,1\}^n$ if $\alpha \in I_B$ and equal to 0 on $\alpha \notin I_B$. It is obvious, that the success probability for $\xi(B)$ is ρ_B. It is not hard to imagine both situations: when $\rho_B = 0$ and when $\rho_B = 1$ (e.g. if $B = X^{in}$ then $\rho_B = 1$ due to Theorem 2).

B. GUESS-AND-DETERMINE ATTACK USING LINEARIZING SET AND ITS HARDNESS

If B is a linearizing set with probability ρ_B which is not extremely small, then we can use it to construct the following cryptographic attack on function (1). Assume that we observe the outputs $\gamma^1, \ldots, \gamma^k$ of function f obtained for inputs $\alpha^1, \ldots, \alpha^k$, chosen independently from $\{0,1\}^n$ w.r.t. a uniform distribution.

For an arbitrary $\gamma^j, j \in \{1, \ldots, k\}$ consider the set $\{0,1\}^{|B|}$ and for each assignment $\beta \in \{0,1\}^{|B|}$ check whether the system $E_f[\beta/B, \gamma^j/Y]$ is linear, and if yes then solve this system, otherwise (e.g. if $E_f[\beta/B, \gamma^j/Y] \notin Lin$) move to the next assignment β. Let $\alpha^j \in \{0,1\}^n$ be an assignment which induces γ^j and β^j, $\beta^j \in \{0,1\}^{|B|}$. Then the probability that $E_f[\beta^j/B, \gamma^j/Y] \in Lin$ is ρ_B. If as a result of processing all assignments from $\{0,1\}^{|B|}$ the event I_B was not observed for a fixed $\gamma^j, j \in \{1, \ldots, k-1\}$ then consider γ^{j+1} and repeat the described procedure. The probability that after analyzing k outputs of function f in this manner we will be able to invert at least one of them (under assumption that $\alpha^j, j \in \{1, \ldots, k\}$ are chosen independently) is

$$P_B = 1 - (1 - \rho_B)^k \tag{7}$$

Definition 2. *Let us say that the considered attack is successful if $P_B \geq 0.95$.*

Taking into account (7) it is easy to see that to fulfill the condition $P_B \geq 0.95$ it is sufficient to analyze $k = \left\lceil \frac{3}{\rho_B} \right\rceil$ of outputs $\gamma^1, \ldots, \gamma^k$. It follows from the fact that $(1-\rho_B)^{\left\lceil \frac{3}{\rho_B} \right\rceil} \leq (1-\rho_B)^{\frac{3}{\rho_B}}$ and the value in the right side of this inequality monotonically increases with the decrease of ρ_B and does not exceed $e^{-3} \approx 0.04978$.

From the above it follows that the *hardness* of a successful (in the aforementioned sense) guess-and-determine attack of the described type, when we use some linearizing set B as a guessed bits set, can be expressed by the following value

(number of systems which should be checked if they are linear and should be solved in the case the answer is "yes"):

$$2^{|B|} \cdot \left\lceil \frac{3}{\rho_B} \right\rceil \tag{8}$$

Thus, to estimate the hardness of the described attack we need to know the value ρ_B. The fact that ρ_B is the success probability of some Bernoulli random variable makes it possible to effectively estimate the value of ρ_B with any required precision using the algorithm which is traditionally referred to as the Monte Carlo method [18]. Fix $B \subseteq X$ and construct the inputs $\alpha^1, \ldots, \alpha^N$, picking them uniformly and independently from $\{0,1\}^n$. For each $j \in \{1, \ldots, N\}$ let us construct the assignments β^j and γ^j, induced by α^j (of variables from B and Y, respectively). Observing the value $\xi(B)$ in the experiment number j (we will denote this value by ξ^j) is defined as follows:

$$\xi^j = \begin{cases} 1, \text{ if } E_f[\beta^j/B, \gamma^j/Y] \in Lin \\ 0, \text{ if } E_f[\beta^j/B, \gamma^j/Y] \notin Lin \end{cases}$$

The estimation of probability ρ_B based on N independent observations of $\xi(B)$: $\xi^1, \ldots, \xi^N$ is the following value:

$$\frac{1}{N}\sum_{j=1}^{N} \xi^j \tag{9}$$

As it can be derived from the results of the paper [19], for any $\varepsilon \in (0,1)$ the following form of Chernoff's bound holds:

$$\Pr\left\{ \left| \rho_B - \frac{1}{N}\sum_{j=1}^{N} \xi^j \right| \leq \varepsilon \right\} \geq 1 - 2e^{-\frac{\varepsilon^2 N}{4}} \tag{10}$$

In (10) $\varepsilon \in (0,1)$ represents the *tolerance* (see [19]) of the estimation of probability ρ_B by value (9) and we can notice that ε can be fixed to arbitrarily small values and at the same time one can make the *confidence level* $1-\delta$ (see [19]), where $\delta = 2e^{-\frac{\varepsilon^2 N}{4}}$, to be as close to 1 by increasing the number of observations N of variable $\xi(B)$. In more detail, for any fixed $\varepsilon, \delta \in (0,1)$ we need to ensure that $N \geq \frac{4\ln(2/\delta)}{\varepsilon^2}$ to guarantee that (10) holds.

Therefore, we can move from (8) to the following *estimation of the hardness* of a cryptographic attack based on linearizing set B:

$$2^{|B|} \cdot \frac{3N}{\sum_{j=1}^{N} \xi^j}. \tag{11}$$

C. CONSTRUCTION OF GUESS-AND-DETERMINE ATTACK AS A METAHEURISTIC OPTIMIZATION PROBLEM

The next step consists in automatizing the search for such B that yields an attack with acceptable hardness estimation (11). Here we will use the approach described in a series of works [20], [21], [22], [23]. These papers considered guess-and-determine attacks which employed the SAT solvers to solve subproblems weakened by substituting into CNF formulas the values of variables from a guessed bits set. In these works, the problem of finding such a set with good estimation of attack hardness was reduced to the pseudo-Boolean optimization problem [24], to solve which the metaheuristic algorithms such as local search [25] or evolutionary algorithms [26] were then applied. We will use a similar approach to construct linearizing set-based attacks with relatively small hardness estimation.

First, let us define a fitness function such that the minimum of this function gives us a linearizing set-based attack with the smallest hardness. Let us define it in the following manner:

$$F_{E_f} : \{0,1\}^{|X|} \to \mathbb{R}^+ \tag{12}$$

The arguments of function (12) are the Boolean vectors from $\{0,1\}^{|X|}$ which define different sets B, $B \subseteq X$. For an arbitrary $\lambda_B \in \{0,1\}^{|X|}$ if the value of a coordinate number i, $i \in \{1, \ldots, |X|\}$ is 1 then it means that $x_i \in B$, otherwise if the value of a coordinate number i is 0 then $x_i \notin B$. Let us describe the process of computing the value of function (12) on an arbitrary $\lambda_B \in \{0,1\}^{|X|}$. First, construct set $B = \{x_1^B, \ldots, x_s^B\}$ corresponding to λ_B. Next, generate N randomly chosen inputs of function f: $\alpha^1, \ldots, \alpha^N$. For each $\alpha^j, j \in \{1, \ldots, N\}$ derive the assignments of all variables from X induced by it and based on them construct assignments β^j (of variables from B) and $\gamma^j = f(\alpha^j)$ (of variables from Y). Then, we substitute to E_f the assignments γ^j and β^j and apply inference rules described above for as long as they are applicable. If the resulting system of equations becomes linear, then we assume that $\xi^j = 1$, otherwise $\xi^j = 0$. Once we obtain the values $\xi^1, \ldots, \xi^N$ we compute the estimation (9) and then use formula (11) to compute the value of (12).

From the definition of function (12) it follows that it is a pseudo-Boolean function that is not defined analytically. That is why it is possible to view it as some black-box function and apply metaheuristic search algorithms to its minimization. As we mentioned above, in a number of earlier works there have been considered fitness functions that are somewhat similar to (12). They were minimized using both local search algorithms [20], [21], and evolutionary algorithms [22]. In our experiments the best results on minimizing function (12) were obtained using a variant of genetic algorithm described in [22] and [27]. In all experiments below we used exactly this algorithm. Let us briefly describe it.

The algorithm for minimizing (12) works with several vectors of the kind $\lambda_B \in \{0,1\}^{|X|}$. Denote these vectors by $\lambda^1, \ldots, \lambda^R$ and let us refer to the set $P = (\lambda^1, \ldots, \lambda^R)$ as to *population* of size R. With each individual of population P associate the value of fitness function (12) on the respective individual, and denote the corresponding values by $F_1, \ldots, F_R$. Also, let us associate with P the set $D = \{p_1, \ldots, p_R\}$, formed by numbers from $[0,1]$ defined as follows:

$$p_i = \frac{1/F_i}{\sum_{j=1}^{r} 1/F_j}, i = 1, \ldots, R$$

Thus, each λ_i is linked with some p_i, $i \in \{1, \ldots, R\}$, and, therefore, the set D can be viewed as the probability

distribution on the sample space P (because all Kolmogorov's axioms [28] hold). Next we use the standard mechanism for choosing the elements from P in accordance with distribution D. First, choose h pairs (each element of a pair is chosen independently from P), apply two-point crossover [26] to each pair and add the resulting pair to the new population P_{new} (which is initialized as empty). At this point, P_{new} has $H = 2h$ individuals. Next, choose from P (w.r.t. D) L individuals, and apply either standard (1+1) random mutation (see e.g. [29]) or a variant of this operator described in [30] to each of them. The resulting individuals are also put into P_{new}. Finally, we add to P_{new} E individuals from P with the best values of the fitness function. The parameters H, L, E are constrained as follows: $H + L + E = R$, which means that P_{new} consists of exactly R individuals. The particular values of R, H, L, E are picked for each considered cryptographic function individually in the process of computational experiments. The algorithm is terminated after a specified number of iterations and outputs the individual (i.e. a specific linearizing set B) with the best known value of the fitness function as the result.

V. USING SAT ORACLES TO GENERATE ADDITIONAL LINEAR CONSTRAINTS

In many cases it is possible to augment the system E_f by additional new linear equations, which in turn increase the probability of linearization. To construct new linear equations we invoke a SAT solver as an oracle. The technique described below is based on the fact that a template CNF formula of the kind C_f is not hard for modern CDCL SAT-solvers even if f is a strong cryptographic function. Moreover, the solvers are quite effective even when a small portion of output bits of function f or an assignment of a small part of internal gate variables of S_f is fixed in C_f.

Now let us in addition to systems E_f and $E_f(\gamma)$ consider CNF formulas C_f and $C_f(\gamma)$. As we noted above, there is a natural bijective mapping between the sets of variables in these systems of equations and formulas, which makes it possible to denote these variables by the same symbols. Assume that we want to determine the consistency of system E_f to which we substituted the values of variables corresponding to some inner gates of circuit S_f (not to its outputs). It is easy to see that the situations are possible when the resulting system is inconsistent. Indeed, let g be some AND-gate, u, w be the variables assigned to its inputs and v be the variable assigned to its output. Evidently, the situation when $u = w = 0$ and $v = 1$ is impossible since the value of v must coincide with the value of the logical connective ($\wedge$) corresponding to g on the considered input. In other words, in this case the value of v contradicts the semantics of g. Let us refer to such a contradiction as *explicit*. However, as we show below, the situation is possible when an assignment does not result in explicit contradiction with specific gates, but the system E_f is inconsistent with such an assignment. We will also see that the proof of inconsistency of E_f and an assignment of variables associated with some gates in some cases allows one to extend the system $E_f(\gamma)$ by new linear equations over $GF(2)$. The main question lies in how to find such assignments? We achieve this goal using a SAT solver.

As we have seen above, with an arbitrary assignment $\sigma = (\sigma_1, \ldots, \sigma_r)$ of some variables $u_1, \ldots, u_r$ in CNF formula C_f we can associate the set of literals $u_1^{\sigma_1}, \ldots u_r^{\sigma_r}$, and to substitute σ into C_f we apply the Unit Propagation rule to the formula $u_1^{\sigma_1} \wedge \ldots u_r^{\sigma_r} \wedge C_f$. Similarly, with the variables $u_1, \ldots, u_r$ in E_f and their assignment $\sigma = (\sigma_1, \ldots, \sigma_r)$ in $GF(2)$ we associate the following set of elementary equations: $u_1 = \sigma_1, \ldots, u_r = \sigma_r$. Adding these equations to E_f can evidently be viewed as a substitution of the corresponding values into E_f. Denote the resulting system as $E_f \cup \{u_j = \sigma_j\}_{j=1}^r$. The theoretical foundation of the results described below is based on the following theorem.

Theorem 3. *Consider the system E_f and the CNF formula C_f and let $u_1, \ldots, u_r$ be the variables associated with some gates in S_f. Then for an arbitrary Boolean vector $\sigma = (\sigma_1, \ldots, \sigma_r)$ the system $E_f \cup \{u_j = \sigma_j\}_{j=1}^r$ is inconsistent if and only if CNF formula $u_1^{\sigma_1} \wedge \ldots \wedge u_r^{\sigma_r} \wedge C_f$ is unsatisfiable.*

Proof sketch: The validity of the theorem follows from Theorem 1 and Theorem 2. Indeed, assume that $u_1^{\sigma_1} \wedge \ldots \wedge u_r^{\sigma_r} \wedge C_f$ is unsatisfiable. But it means that for any input $\alpha \in \{0,1\}^{|X^{in}|}$, $\alpha = (\alpha_1, \ldots, \alpha_n)$ the application of UP to CNF formula $x_1^{\alpha_1} \wedge \ldots \wedge x_n^{\alpha_n} \wedge C_f$, where $\{x_1, \ldots, x_n\} = X^{in}$ will derive literal $u_p^{\neg\sigma_p}$, for some p, $p \in \{1, \ldots, r\}$, which will contradict with the substituted assignment σ. However, in this case, as it follows from Theorem 2 the application of inference rules (5) to $E_{f,\alpha}$ will derive the equation of the kind $u_p = \neg\sigma_p$, $p \in \{1, \ldots, r\}$, which also contradicts the set of equations $\{u_j = \sigma_j\}_{j=1}^r$. Thus the system $E_f \cup \{u_j = \sigma_j\}_{j=1}^r$ becomes inconsistent. The proof of the complementary statement is constructed in exactly the same manner. □

Now let us describe the technique that makes it possible to extend the systems of the kind $E_f(\gamma)$, $\gamma \in Range\ f$ by new linear equations.

Assume that function $f : \{0,1\}^n \to \{0,1\}^m$ is specified by circuit S_f over the basis $\{\wedge, \neg\}$. Let us construct for S_f the MQ-system E_f and CNF formula C_f. Consider an arbitrary AND-gate $g \in S_f$ such that $v \in X$ is the variable associated with g and variables u, w are associated with the parents of g. Then, as it was noted above, there is an equation over $GF(2)$ that corresponds to g, let us denote it by $e_\wedge : u \wedge w \oplus v = 0$. Let $\chi_{e_\wedge}$ be the characteristic function of this equation, that takes the value of 1 only on the assignments of variables from $U = \{u, w, v\}$ which are the solutions of the equation $e_\wedge$ and takes the value of 0 on all the other assignments. This function is specified by Table 1 to which we will refer as to $T(\chi_{e_\wedge})$. It is clear that substituting to E_f any assignment of variables from U on which $\chi_{e_\wedge} = 0$ immediately results in an explicit contradiction. Similarly, the substitution of the corresponding values to C_f (in form of unit clauses) immediately derives the explicit contradiction by UP from the clauses which correspond to the AND-gate g. Therefore,

TABLE 1. Table $T(\chi_{e_\wedge})$.

u	w	v	$\chi_{e_\wedge}$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

we can safely exclude the corresponding rows of table $T(\chi_{e_\wedge})$ from consideration. Now let us focus on the part of table $T(\chi_{e_\wedge})$ formed by the rows on which $\chi_{e_\wedge} = 1$. Denote this table by $\tilde{T}(\chi_{e_\wedge})$ (Table 2).

TABLE 2. Table $\tilde{T}(\chi_{e_\wedge})$.

u	w	v	$\chi_{e_\wedge}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	1

Now let us establish the validity of the following fact.

Theorem 4. *Let $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ be an arbitrary assignment of variables u, w, v from table $\tilde{T}(\chi_{e_\wedge})$. Assume that CNF formula $u^{\sigma_1} \wedge w^{\sigma_2} \wedge v^{\sigma_3} \wedge C_f$ is unsatisfiable. Then to any MQ-system of the kind $E_f(\gamma)$, $\gamma \in Range\, f$ we can add a new linear equation over $GF(2)$ and the resulting MQ-system will have the same set of solutions as $E_f(\gamma)$.*

Proof sketch: Consider table $\tilde{T}(\chi_{e_\wedge})$ constructed for an arbitrary AND-gate g. Let us pick assignments $\sigma \in \{000, 010, 100, 111\}$, $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ one by one and assume that $u^{\sigma_1} \wedge w^{\sigma_2} \wedge v^{\sigma_3} \wedge C_f$ is unsatisfiable. After this we cross out the row σ from table $\tilde{T}(\chi_{e_\wedge})$ and observe that the remaining rows are the solutions of some linear equation over $GF(2)$. The equations corresponding (in the sense above) to the removed rows are outlined in Table 3

TABLE 3. Linear equations over $GF(2)$ that correspond to the removed rows of table $\tilde{T}(\chi_{e_\wedge})$.

u	w	v	
0	0	0	$u \oplus w \oplus v = 1$
0	1	0	$w \oplus v = 0$
1	0	0	$u \oplus v = 0$
1	1	1	$v = 0$

Let us illustrate this approach on the example of the first row. If $(\sigma_1, \sigma_2, \sigma_3) = (0, 0, 0)$ and formula $u^{\sigma_1} \wedge w^{\sigma_2} \wedge v^{\sigma_3} \wedge C_f$ is unsatisfiable, then from Theorem 3 any input $\alpha \in \{0, 1\}^n$ during the interpretation of circuit S_f on this input will result in the assignment of values of variables u, w, v from the set $\{010, 100, 111\}$ which is the subset of the set of solutions for the equation $u \oplus w \oplus v = 1$. Therefore, adding this equation to E_f will not actually change the set of solutions of E_f. Since this is valid for any input of f, it is also valid for the preimage of an arbitrary image $\gamma \in Range\, f$. □

In the computational experiments that we describe in the following section, we will show that even for some resistant cryptographic functions, a SAT solver, when viewed as an oracle, is able to prove the unsatisfiability of CNF formulas of the kind $u^{\sigma_1} \wedge w^{\sigma_2} \wedge v^{\sigma_3} \wedge C_f$ quite effectively. Moreover, the SAT oracles can be effective even in the situations when apart from literals u^{σ_1}, w^{σ_2} and v^{σ_3} the formula is extended with some other known information, e.g. a small number of known output bits.

VI. COMPUTATIONAL EXPERIMENTS

In this section we present the results of our experiments on the application of the proposed method for solving MQ-systems to a number of well-known stream ciphers.

A. CONSIDERED FUNCTIONS AND THEIR ENCODINGS

We considered several keystream generators that have been extensively studied in the literature. In particular, we analyzed the following stream ciphers: A5/1, Alternating Step Generator (ASG), Trivium and Bivium. We will present some historical perspective related to these ciphers in Section VII. In accordance with the formalism employed in the present paper in the case of the A5/1 generator we considered the problem of inversion of function $f_{A5/1} : \{0, 1\}^{64} \to \{0, 1\}^{64}$. ASG generator was studied in two variants earlier analyzed in papers [22], [31] using the SAT solvers: here we mean the functions $f_{ASG_{72}} : \{0, 1\}^{72} \to \{0, 1\}^{76}$ and $f_{ASG_{96}} : \{0, 1\}^{96} \to \{0, 1\}^{112}$. For the Bivium cipher we considered the state recovery attack in the same formulation as in papers [32], [33], [34] and some others. In particular, we studied the problem of inversion of function $f_{Biv} : \{0, 1\}^{177} \to \{0, 1\}^{200}$. The cryptanalysis of Trivium was considered as the problem of finding a secret key with a known initial value (IV) of length 80 and the known keystream fragment of length 300 bits, thus in the case of Trivium we analyzed the inversion problem for function $f_{Triv-M} : \{0, 1\}^{80} \to \{0, 1\}^{300}$, where M is the number of initialization steps (the corresponding parameter is a part of the specification of the Trivium algorithm [35]).

In all considered cases based on the algorithmic description of a corresponding function f we first constructed the circuit S_f over the basis $\{\wedge, \neg\}$, i.e. an And-Inverter Graph (AIG). For this purpose we employed the `Transalg` tool [13], [36]. Then, using the AIG produced by `Transalg` we constructed MQ-systems and CNF formulas of the kind E_f, $E_f(\gamma)$, C_f, $C_f(\gamma)$ using the procedures described above. We also implemented a program that performs substitution of assignments of variables into MQ-system and the application of inference rules (5)-(6). The problem of finding a linearizing set with minimal estimation of hardness of the corresponding attack was solved as a pseudo-Boolean optimization problem using the algorithms described in Section IV.

B. IMPLEMENTATION DETAILS

All computational experiments were run on the computing platform comprised of a single node of the Academician V.M. Matrosov computing cluster of Irkutsk supercomputer center [37], which is equipped with two 18-core Intel Xeon 2695 v4 CPUs and 128 Gb RAM. To optimize fitness function (12) we employed the EvoGuessAI framework that was developed specifically for solving pseudo-Boolean black-box optimization problems [38]. Each experiment took 24 hours, and as the result we chose the linearizing set with the best known value of function (12). To compute the values of fitness function we used random samples of size $N = 1000$.

Despite the fact that checking whether a system of the kind $E_f[\beta/B, \gamma^j/Y]$ is linear can be performed fast, to compute the value of function (12) in a single point of the search space, we need to make N checks of such kind, which leads to a significant slowdown of the search when N is large, while when N is small, we lack the justified guarantees of the accuracy of constructed estimations. That is why we employed the reduced search spaces. The simplest variant of such a reduction is based on the following fact which is a direct consequence of Theorem 2: the set X^{in} is a linearizing set with $\rho_B = 1$ for any system of the kind E_f and $E_f(\gamma)$. Taking all this into account, in all experiments we minimized function (12) only over the set $2^{X^{in}}$, representing different subsets of X^{in} by Boolean vectors of length $n = |X^{in}|$.

To construct additional linear equations (see Section V) we used in the role of SAT oracles the Glucose SAT solver, versions 3 and 4 [39].

C. LINEARIZING SET ATTACKS ON A5/1

The A5/1 keystream generator is one of the most well-known cryptographic functions (together with DES, AES, RSA, MD and SHA) since for a long time it was used to encrypt the data transmitted in GSM networks. This algorithm is considered completely compromised ever since the the construction of the rainbow tables, which made it possible to find a secret key of the generator (with some probability of success) in just several minutes on a usual PC [40]. We will mention other attacks on A5/1 in Section VII.

A5/1 uses three Linear Feedback Shift Registers (LFSR) [41], which are shifted asynchronously. We used the description of the algorithm presented in [42]. The A5/1 LFSRs are of the sizes 19, 22 and 23, thus the size of the secret key is 64 bits. One of the first attacks on A5/1 which is significantly more effective compared to brute force is the attack described by Ross Anderson but from the practical point of view the corrected version of this attack from [43] is of more interest since it was implemented using FPGA. In the context of this attack the variables corresponding to the 1st and the 3rd LFSRs are put into the guessed bits set together with 11 variables from the 2nd LFSR. As a result we have the guessed bits set B comprised of 53 variables. We will refer to this set as to "Anderson's set" and note that the details regarding its construction can be found in [43]. Also we refer to the corresponding guess-and-deremine attack as to "Anderson's attack". By analyzing the structure of this set it is easy to see that the Anderson's set B is actually a linearizing set with linearization probability $\rho_B = 1$. Thus, the hardness of this attack is 2^{53} of solved systems of linear equations over $GF(2)$.

TABLE 4. The results of linearizing set-based attacks on the A5/1 keystream generator.

Algorithm	$\|B\|$	ρ_B	Hardness
LS	44	0.072	$2^{49.4}$
LS+SAT	44	0.213	$2^{47.8}$

From the results of the paper [44] it follows that in the case of A5/1 to restore the secret key that leads to the generation of a keystream fragment of size ≥ 64 bits, it is sufficient to use the first 64 bits of this keystream. Thus, we considered the problem of cryptanalysis of A5/1 as the inversion problem for the function $f_{A5/1} : \{0,1\}^{64} \to \{0,1\}^{64}$. It means, that in a single inversion attempt we used a 64-bit fragment of keystream.

In the computational experiments conducted in accordance with the general scheme outlined above, we found linearizing sets comprised by fewer than 53 variables but with $\rho_B < 1$. The results of the best found attack (in the sense of the number of solved linear systems) are showed in Table 4. Hereinafter, to denote the attack in which we consider only template CNF E_f we use the notation LS, and by LS+SAT we denote the attack in which E_f is extended by additional linear equations constructed with the help of a SAT oracle using the technique described in Section V. Note, that for A5/1 the LS+SAT attack has better value of ρ_B and smaller hardness, compared to the LS attack.

We would like to specifically note, that the constructed attack has significantly smaller hardness compared to the Anderson's attack. Also, the algorithms for solving linear equations over $GF(2)$ allow effective GPU implementation thus making the constructed attack realistic.

D. LINEARIZING SET ATTACKS ON THE ALTERNATING STEP GENERATOR

The Alternating Step Generator (ASG) is a popular object of studies in cryptography, despite the fact that to the best of our knowledge, it was not employed in any practical security system. ASG was proposed in [45]. It uses 3 LFSRs, among which one is called control register and it produces the so-called control sequence. The latter defines which of the other two registers will be clocked at each moment. The keystream bit is formed by the sum of the output bits of two controlled registers modulo two.

The trivial guess-and-determine attack on ASG consists in choosing the variables encoding the contents of the control register in the role of the guessed bits set. In our experiments we launched the procedure for minimization of function (12) on the set X^{in} (formed by the variables encoding the initial

values of all three LFSRs). An interesting fact is that in almost all experiments the minimization algorithm constructed the set that coincides (with the exception of a single variable in some cases) with the set of variables corresponding to the control register. We would like to emphasize that the algorithm was not provided with any additional details about the control register, and still it managed to find this set in a completely automatic mode.

In Table 5 and Table 6 we show the results of the automatic construction of linearizing set attacks on variants of ASG with key lengths 72 and 96 bits (ASG_{72} and ASG_{96}, respectively). Recall, that we considered the inversion problems for functions $f_{ASG_{72}} : \{0,1\}^{72} \rightarrow \{0,1\}^{76}$ and $f_{ASG_{96}} : \{0,1\}^{96} \rightarrow \{0,1\}^{112}$. For ASG 72 and ASG 96 the keystream sizes were picked from [31].

TABLE 5. The results of linearizing set-based attacks on the ASG_{72} keystream generator.

Algorithm	$\|B\|$	ρ_B	Hardness
LS	23	1	$2^{24.6}$
LS+SAT	22	0.835	$2^{23.8}$

TABLE 6. The results of linearizing set-based attacks on the ASG_{96} keystream generator.

Algorithm	$\|B\|$	ρ_B	Hardness
LS	31	1	$2^{32.6}$
LS+SAT	31	1	$2^{32.6}$

To the best of our knowledge, the best attack on ASG was described in [46]. However, in the general case it requires a keystream fragment of larger size. As far as we are aware from the published papers on ASG cryptanalysis in the case when the keystream fragment size is comparable to the size of the secret key, there are no attacks that are significantly more effective than brute force of the control register (which essentially is a linearizing set attack with probability of linearization $\rho_B = 1$). In the case of ASG_{72} the control register consists of 23 cells, and in the case of ASG_{96} – of 31 cells. Again, they were found automatically by our method. In the case of ASG_{72} the use of SAT oracle made it possible to remove a single variable from this set and improve the fitness function value.

Thus, we can consider cryptanalysis of ASG as a reference problem for the proposed method: our algorithm automatically constructs the attack which is the best known (for relatively small keystream length).

E. LINEARIZING SET ATTACKS ON WEAKENED VARIANTS OF TRIVIUM

The Trivium cipher [35] is one of the winners of the eSTREAM project for lightweight stream ciphers. It has a simple architecture and, as a consequence, is easy to implement in hardware. The generator consists of 3 shift registers of a special kind (they are not LFSRs) of lengths 84, 93 and 111 bits (thus the cipher has the state of registers of size 288 bits). Trivium uses a secret key of length 80 bits and a non-secret Boolean vector called initial value (IV) of the same length. The secret key and IV are put into the first and the second registers, which are then padded by zeroes. The third register is filled by a known sequence presented in the specification of Trivium. After this the cipher works in the initialization mode in which the registers are clocked $4 \times 288 = 1152$ times without producing keystream. After the initialization phase, Trivium can generate keystream of an arbitrary length. We considered the problems of the kind $f_{Triv-M} : \{0,1\}^{80} \rightarrow \{0,1\}^{300}$, where by Trivium-M we denote a variant of Trivium with the number of initialization steps equal to M.

To the best of our knowledge, there are no known attacks on the original Trivium (with $M = 1152$) that are significantly more effective than the brute force algorithm that traverses through all possible variants of the 80-bit secret key (given the known value of IV and some keystream fragment). The hardness estimations smaller than 2^{80} can be constructed only when considering the Trivium variants that are weakened in the number of initialization steps (i.e. for $M < 1152$). The best known results in this context are obtained using the so-called cube attack [47], [48]. Note, however, that the attacks from these papers employ the scenario, in the context of which the encryption uses many different IVs.

The linearizing set-based attacks scenario described above implies that one uses a single IV. The similar scenario is implied in the SAT-based attacks which exploit the Inverse Backdoor Set (IBS) notion. This class of cryptographic attacks was introduced in [21]. IBS can be considered as a special case of the backdoor set notion which was proposed in [49]. The IBS-based attacks are aimed at functions' inversion problems, and the probabilistic reasoning that specifies them is quite similar to the one we used to define the probability of linearization.

TABLE 7. The results of linearizing set-based and IBS-based attacks on Trivium-M keystream generator (M is the number of initialization steps).

M	Algorithm	$\|B\|$	ρ_B	Hardness	Time estimation
160	LS	44	0.915	$2^{45.7}$	$2^{41.5}$
	LS+SAT	36	0.636	$2^{38.2}$	$2^{37.2}$
	IBS	26	-	–	$\mathbf{2^{30.8}}$
192	LS	58	0.370	$2^{61.0}$	$2^{58.6}$
	LS+SAT	45	0.789	$2^{46.9}$	$2^{45.7}$
	IBS	38	-	–	$\mathbf{2^{43.6}}$
288	LS	73	0.725	$2^{76.0}$	$2^{72.7}$
	LS+SAT	64	0.993	$2^{65.6}$	$\mathbf{2^{63.2}}$
	IBS	63	-	–	$2^{67.9}$
384	LS	78	0.971	$2^{79.6}$	$2^{74.3}$
	LS+SAT	75	0.977	$2^{76.6}$	$\mathbf{2^{73.8}}$
	IBS	75	-	–	$2^{79.6}$
416	LS	79	1	$2^{79.0}$	$\mathbf{2^{74.4}}$
	LS+SAT	76	0.826	$2^{77.9}$	$2^{75.5}$
	IBS	76	-	–	$2^{80.8}$

In Table 7 we show the results on the application of LS-based and IBS-based attacks on functions f_{Triv-M}, $M \in \{160, 192, 288, 384, 416\}$. Note, that for IBS attacks,

in accordance with [21] we show the runtime estimation in seconds for a CPU on which the estimation was computed. For each of LS attacks we show both the number of linear systems that need to be solved, as well as the runtime estimation on the same CPU as in the IBS case (single core of Intel Xeon 2695 v4). For this purpose we used a straightforward naive CPU-implementation of Gaussian Elimination.

From the analysis of the obtained data it can be seen that the IBS-based attacks are more effective for small M, however, with the increase of M the effectiveness of LS-based attacks also grows, and for $M \in \{384, 416\}$ we can see that LS and LS+SAT significantly outperform IBS. Also, we would like to again note that the performance of LS-based attacks can be substantially improved by implementing them on GPU.

F. LINEARIZING SET ATTACKS ON BIVIUM

The Bivium cipher is a weakened variant of the Trivium cipher, that employs only two out of three Trivium registers. Bivium is a popular object for different cryptographic attacks, some of which we will mention in Section VII. Overall, Bivium is not a cryptographically resistant cipher and it allows even effective state recovery attacks. In such attacks one needs to recover the state of cipher registers at the moment the keystream generation started, by analyzing the corresponding keystream fragment. In the case of Bivium the state is of the size $177 = 84 + 93$ bits, and the size of the analyzed keystream fragment is 200 bits (thus the inversion problem for function $f_{Biv} : \{0,1\}^{177} \to \{0,1\}^{200}$ was considered). The hardness estimations of the constructed linearizing set-based attacks are presented in Table 8.

TABLE 8. The results of linearizing set-based attacks on the Bivium keystream generator.

Algorithm	$\lvert B\rvert$	ρ_B	Hardness
LS	54	0.624	$2^{56.3}$
LS+SAT	42	0.635	$2^{44.2}$

We would like to specifically note that the attack corresponding to the LS+SAT scenario is apparently one of the most effective attacks on this cipher among the known state recovery attacks. Indeed, to the best of our knowledge the best state recovery attacks on Bivium are the ones presented in [20] and [34]. They both use the SAT solvers to solve weakened problems and their hardness is expressed in seconds. For example the attack from [34] has the hardness of $2^{36.5}$ seconds. Even naive and straightforward implementation of Gaussian Elimination (GE) on CPU for the found linearizing set gives us the hardness estimation of the corresponding attack in seconds equal to 2^{33}. Of course, this estimation can be significantly improved using a careful implementation of GE on GPU.

VII. RELATED WORK

The term "Algebraic cryptanalysis" became established since the publication of a monograph [2] by G. Bard, despite the fact that the term itself has been used earlier (e.g. in [50]). In accordance with [2] Algebraic cryptanalysis is an area, where a cryptanalysis problem is considered as the problem of solving a system of algebraic equations (usually, over some finite field). One can employ a variety of methods for solving such algebraic equations, a good review of which was presented in [2]. The approach that the monograph lists as one of the most promising consists in the use of SAT solvers. The interest to the latter as to computational tools for reducing the combinatorial complexity steadily increases in many different areas, see e.g. [51], [52], [53], [54], [55], and [56], etc. Note, that the SAT solvers are often used to construct cryptographic attacks, and sometimes such attacks can even be considered practical: speaking about practical attack we mean that, first, an attack is applicable to some cryptographic function which is used currently or has been used recently, and, second, such an attack can be implemented in reasonable time (see [34], [57], [58], etc.).

The main object of our interest is formed by the systems of multivariate quadratic equations (MQ-systems), which encode the considered cryptographic algorithms. To construct such an MQ-system we view the cryptographic function as a Boolean circuit which is a special case of a directed acyclic graph. In this context we can notice that using a graph to specify a discrete function is a natural concept with many examples from different areas (from classic works on random graphs theory [59] and computational biology [60] to relatively recent papers about specific graph connection properties [61], [62], [63]. In the present paper we exploit the following fact: if a total discrete function is specified by some polynomial time algorithm, then one can use this specification to effectively construct Boolean circuit defining the original function. This fact is the direct consequence of the fundamental Cook-Levin theorem [1], [64], [65]. We use the reductions of Boolean circuits to MQ-systems and SAT to construct guess-and determine attacks on several stream ciphers and note that some of these attack can be implemented in practice.

The A5/1 algorithm for a long time was used to encrypt traffic in GSM networks. It employs three Linear Feedback Shift Registers (LFSR) [41], that are cycled asynchronously, and this fact is responsible for the non-linearity of crypt-analysis equations. There are a number of attacks on A5/1. Probably, the most well-known attacks are the ones described in [40] and [42]. After the publication of the attack from [40] A5/1 is considered to be completely compromised. The first guess-and-determine attack on this algorithm was proposed by R. Anderson in 1993. This attack can be mounted in reasonable time if one uses specialized computing architectures allowing massive parallelism, such as FPGA or GPU. The corresponding results were presented in [43] and [66] for FPGA, and in [67] for GPU. One can view the Anderson's attack as an attack in which one has to solve simple systems of linear equations over $GF(2)$. Then the hardness of such an attack is equal to solving 2^{53} of such systems. A similar measure of hardness of an attack found

automatically by our method (see Section VI-C) is $\approx 2^{47.8}$ of linear systems over $GF(2)$.

The Alternating Step Generator (ASG) was proposed in [45]. It implements the idea of using a separate register for controlling the shifts of the generating register and in this sense it is similar to the Shrinking [68] and Self-Shrinking generators [69]. Apparently, the most effective known attack on ASG was proposed in [46], however, it requires to use a keystream fragment of relatively large size. When the keystream fragment size is small, then the best known attack is the one which consists in traversing over all possible assignments of variables corresponding to the initial values of the control register (it is easy to see that this set is a linearizing set with $\rho_B = 1$). Note that the method proposed in the present paper almost always automatically finds the set B which consists of all control register bits.

The Trivium stream cipher is one of the winners of the eSTREAM project.[1] It has a simple structure and is easy to implement both in software and in hardware. Today there are no known attacks on the original Trivium that are significantly more effective than bruteforce over the 80-bit secret key. Thus, the principles behind the Trivium architecture look quite appealing for constructing lightweight cryptographically resistant ciphers based on them. Apparently, the best known attacks on Trivium are the cube attacks, first introduced in [47]. That approach was further developed in several papers, see e.g. [48] for the state-of-the-art variant. However, it is necessary to note, that these attacks are constructed under a very specific scenario, in which the same secret key is used in combination with many different IVs.

The Bivium cipher [70] is the weakened version of Trivium in the sense that Bivium employs two registers of Trivium out of three. There are several known attacks on Bivium, e.g. [33], [34], and [71], and the ones employing SAT solvers often yield the results which are among the best. Here we mean the state recovery attacks, when one needs to recover the initial values of all generator registers at the moment the keystream generation starts (once it was done, the key can be effectively constructed by inverting the initialization phase). In Bivium the size of the cipher state is 177 bits. The linearizing set attack constructed in the present paper uses the guessed bits set of only 42 bits. This fact implicitly reveals how insecure the Bivium cipher is. In seconds the hardness of the cryptanalysis of Bivium using the attack based on the found linearizing set is about 2^{33} (using a straightforward implementation of Gaussian elimination on CPU). This estimation is smaller than the estimation (in seconds) of the attack on Bivium described in [34] which, to the best of our knowledge, was the best known state recovery attack on Bivium so far.

As we already mentioned, the guess-and-determine (or guess-then-determine) attacks form one of the most numerous classes of cryptographic attacks and even the key papers in this area are too numerous to cite. The monograph [2] can be considered a good starting point for better understanding the main concepts behind such attacks. It is hard to pinpoint the exact moment when the automatic construction of guess-and-determine attacks was proposed for the first time, but apparently the paper [72] was among the first works in this direction. In that paper, the authors associated with the multivariate equations for Trivium a special pseudo-Boolean function which was later minimized using metaheuristic algorithms. However, the fitness function used in [72] (to which they refer as to "cost function") was defined in quite peculiar manner: namely, without using any basic algorithm for solving systems of multivariate equations. The idea to express hardness of a guess-and-determine attack via the mean value of runtime of some combinatorial algorithm used to solve weakened cryptanalysis equation systems, was proposed in [73] (a SAT solver was used for this purpose). The general approach consisting in the combination of the Monte Carlo estimation of hardness of a guess-and-determine attack w.r.t. specific guessed bits set B and specific algorithm A with the ability to move between different guessed bits sets using metaheuristic optimization was apparently first proposed in [20] and [74]. Later, a number of guess-and-determine attacks were constructed using metaheuristic pseudo-Boolean optimization in [22], [23], and [31]. We are aware of several other works that proposed the approaches to automatic construction of guess-and-determine attacks but they are based on other principles compared to the ones used in our method (e.g. [75], [76])

It should be noted that the algorithmic essence of guess-and-determine attacks using SAT solvers agrees well with the Cube-and-Conquer concept [77], since specific assignments of variables from a guessed bits set B can be viewed as the so-called *cubes*. It is also easy to see, that the set of all possible assignments of variables from B provides some *SAT partitioning* of formula $C_f(\gamma)$ in the sense of [78]. In recent years, there appeared a number of papers on SAT-based cryptographic attacks which employ the ideas of cube-and-conquer and SAT partitioning, see e.g. [23], [79], [80], and [81], etc.

In [21] it was shown that a guessed bit set in any guess-and-determine attack can be considered as a special variant of the so-called "Backdoor set", the notion of which was first introduced in [49]. Backdoors quickly became the concept that is widely used in structural and parameterized complexity [82]. Also note that a fitness function used in [21] essentially expresses the SAT immunity (see [73]) of a considered cipher w.r.t. a specific guessed bits set.

In [83] we proposed the concept of probabilistic backdoors. The notion of linearizing sets introduced in the present paper can be viewed as a combination of the notions of probabilistic backdoor set (or ρ-Backdoor) from [83] and Inverse Backdoor Set (IBS) from [21]. In fact, when we are working with linearizing sets we need to estimate the probability that a weakened problem lies in a polynomial Schaefer's class [84] comprised by formulas, for which SAT is solvable via system of linear equations over $GF(2)$.

[1] https://www.ecrypt.eu.org/stream/portfolio_revision1.pdf

Also we would like to note the papers [85], [86] in which for several keystream generators they described the attacks which are essentially linearizing set-based ones with probability of linearization equal to 1.

VIII. DISCUSSION AND CONCLUSION

In the present paper we described a new class of cryptographic guess-and-determine attacks, which are based on the idea to search for guessed bits sets, such that by assigning some values to variables from this set one can transform the considered MQ-system over $GF(2)$ into a linear one with some probability. We showed that it is possible to construct a special pseudo-Boolean function, the value of which for a particular guessed bits set estimates the hardness of the linearizing set-based attack which uses this set. The value of such function represents the estimation of the number of systems of equations, for which one needs to check whether a system is linear and if yes to solve it, in order to find a secret key with high probability of success ($>$ 95%). Any guessed bits set in the context of the considered scenario is referred to as a linearizing set (with corresponding probability of linearization). To search for linearizing sets with non-trivial hardness estimations we use metaheuristic optimization algorithms. A number of attacks constructed using our method can be considered practical, since they can be implemented in reasonable time. In particular, the state recovery attack on Bivium presented in this paper is apparently the best known attack among state recovery attacks on this cipher.

From our point of view, the general concept of constructing guess-and-determine attacks by combining Monte Carlo estimations with metaheuristic optimization, which was described in the paper, looks quite promising. It can be used in combination with combinatorial algorithms different from SAT, such as e.g. Buchberger's algorithm (Gröbner basis method), or with polynomial subsolvers different from the ones considered above.

The technique of using SAT oracles to generate additional linear equations presented in Section V, to the best of our knowledge, have not been described before. We plan to develop this technique in the nearest future. In this context it looks promising to design new heuristics for forming sets of variables over which we then attempt to construct additional linear equations by using a SAT oracle.

One of the main problems for implementing SAT-based algebraic attacks in practice consists in that it is currently impossible to realize complete state-of-the-art SAT solving algorithms on modern GPUs. The linearizing set-based attacks have an advantage in this context, because they can be mounted on modern GPUs using the implementations of the algorithms for solving systems of linear equations over GF(2). Surprisingly, the vast majority of publicly available implementations of this kind are designed to work with floating point numbers, and are not suitable for implementing the techniques proposed in the present paper. We plan to fix this situation in the nearest future.

REFERENCES

[1] O. Goldreich, *Computational Complexity: A Conceptual Perspective*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[2] G. V. Bard, *Algebraic Cryptanalysis*. New York, NY, USA: Springer, 2009.

[3] N. T. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations," in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), vol. 1807. Berlin, Germany: Springer, 2000, pp. 392–407.

[4] N. T. Courtois and J. Pieprzyk, "Cryptanalysis of block ciphers with overdefined systems of equations," in *Advances in Cryptology—ASIACRYPT* (Lecture Notes in Computer Science), vol. 2501. Berlin, Germany: Springer, 2002, pp. 267–287.

[5] C.-L. Chang and R. C.-T. Lee, *Symbolic Logic and Mechanical Theorem Proving* (Computer Science Classics). New York, NY, USA: Academic Press, 1973.

[6] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*. New York, NY, USA: W. H. Freeman, 1979.

[7] I. Wegener, *The Complexity of Boolean Functions*. Hoboken, NJ, USA: Wiley, 1987.

[8] A. Biere, K. Heljanko, and S. Wieringa, "AIGER 1.9 and beyond," Inst. Formal Models Verification, Johannes Kepler Univ., Linz, Austria, Tech. Rep. 11/2, 2011.

[9] G. S. Tseitin, "On the complexity of derivation in propositional calculus," in *Studies in Constructive Mathematics and Mathematical Logic*. New York, NY, USA: Springer, 1970, pp. 115–125.

[10] S. Szeider, "Backdoor sets for DLL subsolvers," *J. Automated Reasoning*, vol. 35, nos. 1–3, pp. 73–88, Oct. 2006.

[11] W. F. Dowling and J. H. Gallier, "Linear-time algorithms for testing the satisfiability of propositional horn formulae," *J. Log. Program.*, vol. 1, no. 3, pp. 267–284, Oct. 1984.

[12] J. Marques-Silva, I. Lynce, and S. Malik, "Conflict-driven clause learning SAT solvers," in *Handbook Satisfiability* (Frontiers in Artificial Intelligence and Applications), vol. 336, 2nd ed. Amsterdam, The Netherlands: IOS Press, 2021, pp. 133–182.

[13] A. Semenov, I. Otpuschennikov, I. Gribanova, O. Zaikin, and S. Kochemazov, "Translation of algorithmic descriptions of discrete functions to SAT with applications to cryptanalysis problems," *Log. Methods Comput. Sci.*, vol. 16, no. 1, pp. 29:1–29:42, 2020.

[14] C. Bessiere, G. Katsirelos, N. Narodytska, and T. Walsh, "Circuit complexity and decompositions of global constraints," in *Proc. IJCAI*, 2009, pp. 412–418.

[15] R. Drechsler, T. A. Junttila, and I. Niemelä, "Non-clausal SAT and ATPG," in *Handbook Satisfiability* (Frontiers in Artificial Intelligence and Applications), vol. 336, 2nd ed. Amsterdam, The Netherlands: IOS Press, 2021, pp. 1047–1086.

[16] S. A. Cook and R. A. Reckhow, "The relative efficiency of propositional proof systems," *J. Symbolic Log.*, vol. 44, no. 1, pp. 36–50, Mar. 1979.

[17] Roger A. Horn and Charles R. Johnson," *Matrix Analysis*, Cambridge, U.K.: Cambridge Univ. Press, 1985.

[18] N. Metropolis and S. Ulam, "The Monte Carlo method," *J. Amer. Stat. Assoc.*, vol. 44, no. 247, pp. 335–341, 1949.

[19] R. M. Karp, M. Luby, and N. Madras, "Monte-Carlo approximation algorithms for enumeration problems," *J. Algorithms*, vol. 10, no. 3, pp. 429–448, Sep. 1989.

[20] A. Semenov and O. Zaikin, "Algorithm for finding partitionings of hard variants of Boolean satisfiability problem with application to inversion of some cryptographic functions," *SpringerPlus*, vol. 5, no. 1, pp. 1–16, Dec. 2016.

[21] A. Semenov, O. Zaikin, I. Otpuschennikov, S. Kochemazov, and A. Ignatiev, "On cryptographic attacks using backdoors for SAT," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2018, pp. 6641–6648.

[22] A. Pavlenko, A. Semenov, and V. Ulyantsev, "Evolutionary computation techniques for constructing SAT-based attacks in algebraic cryptanalysis," in *Applications of Evolutionary Computation* (Lecture Notes in Computer Science), vol. 11454. Cham, Switzerland: Springer, 2019, pp. 237–253.

[23] A. Semenov, O. Zaikin, and S. Kochemazov, "Finding effective SAT partitionings via black-box optimization," in *Black Box Optimization, Machine Learning, and No-Free Lunch Theorems* (Springer Optimization and Its Applications), vol. 170. Cham, Switzerland: Springer, 2021, pp. 319–355.

[24] E. Boros and P. L. Hammer, "Pseudo-Boolean optimization," *Discrete Appl. Math.*, vol. 123, nos. 1–3, pp. 155–225, Nov. 2002.

[25] E. K. Burke and G. Kendall, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, 2nd ed. New York, NY, USA: Springer, 2013.

[26] S. Luke, *Essentials of Metaheuristics*, 2nd ed. Morrisville, NC, USA: Lulu, 2013.

[27] A. Semenov, D. Chivilikhin, A. Pavlenko, I. Otpuschennikov, V. Ulyantsev, and A. Ignatiev, "Evaluating the hardness of SAT instances using evolutionary optimization algorithms," in *Proc. 27th Int. Conf. Princ. Pract. Constraint Program. (CP)*, in Leibniz International Proceedings in Informatics, vol. 210, 2021, pp. 47:1–47:18.

[28] W. Feller, *An Introduction to Probability Theory and its Applications*, vol. 2, 2nd ed. Hoboken, NJ, USA: Wiley, 1971.

[29] S. Droste, T. Jansen, and I. Wegener, "On the analysis of the (1+1) evolutionary algorithm," *Theor. Comput. Sci.*, vol. 276, nos. 1–2, pp. 51–81, Apr. 2002.

[30] B. Doerr, H. P. Le, R. Makhmara, and T. D. Nguyen, "Fast genetic algorithms," in *Proc. GECCO*, 2017, pp. 777–784.

[31] O. Zaikin and S. Kochemazov, "An improved SAT-based guess-and-determine attack on the alternating step generator," in *Information Security* (Lecture Notes in Computer Science), vol. 10599. Cham, Switzerland: Springer, 2017, pp. 21–38.

[32] A. Maximov and A. Biryukov, "Two trivial attacks on TRIVIUM," in *Selected Areas in Cryptography* (Lecture Notes in Computer Science), vol. 4876. Berlin, Germany: Springer, 2007, pp. 36–55.

[33] T. Eibach, E. Pilz, and G. Völkel, "Attacking Bivium using SAT solvers," in *Theory and Applications of Satisfiability Testing—SAT* (Lecture Notes in Computer Science), vol. 4996. Berlin, Germany: Springer, 2008, pp. 63–76.

[34] M. Soos, K. Nohl, and C. Castelluccia, "Extending SAT solvers to cryptographic problems," in *Theory and Applications of Satisfiability Testing—SAT* (Lecture Notes in Computer Science), vol. 5584. Berlin, Germany: Springer, 2009, pp. 244–257.

[35] C. De Cannière, "TRIVIUM: A stream cipher construction inspired by block cipher design principles," in *Information Security* (Lecture Notes in Computer Science), vol. 4176. Berlin, Germany: Springer, 2006, pp. 171–186.

[36] I. Otpuschennikov, A. Semenov, I. Gribanova, O. Zaikin, and S. Kochemazov, "Encoding cryptographic functions to SAT using TRANSALG system," in *Proc. ECAI* (Frontiers in Artificial Intelligence and Applications), vol. 285. Amsterdam, The Netherlands: IOS Press, 2016, pp. 1594–1595.

[37] *Irkutsk Supercomputer Center of SB RAS*. Accessed: Sep. 20, 2023. [Online]. Available: http://hpc.icc.ru

[38] *EvoGuessAI Library*. Accessed: Sep. 20, 2023. [Online]. Available: https://github.com/aimclub/evoguess-ai

[39] G. Audemard and L. Simon, "On the glucose SAT solver," *Int. J. Artif. Intell. Tools*, vol. 27, no. 1, 2018, Art. no. 1840001.

[40] K. Nohl, "Attacking phone privacy," in *Proc. BlackHat Lect. Notes*, Las-Vegas, NV, USA, 2010, pp. 1–6.

[41] A. J. Menezes, S. A. Vanstone, and P. C. Van Oorschot, *Handbook of Applied Cryptography*, 1st ed. Boca Raton, FL, USA: CRC Press, 1996.

[42] A. Biryukov, A. Shamir, and D. A. Wagner, "Real time cryptanalysis of A5/1 on a PC," in *Fast Software Encryption* (Lecture Notes in Computer Science), vol. 1978. New York, NY, USA: Springer, 2000, pp. 1–18.

[43] T. Gendrullis, M. Novotný, and A. Rupp, "A real-world attack breaking A5/1 within hours," in *Cryptographic Hardware and Embedded Systems—CHES* (Lecture Notes in Computer Science), vol. 5154. Berlin, Germany: Springer, 2008, pp. 266–282.

[44] J. D. Golic, "Cryptanalysis of alleged A5 stream cipher," in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), vol. 1233. Konstanz, Germany: Springer, 1997, pp. 239–255.

[45] C. G. Günther, "Alternating step generators controlled by De Bruijn sequences," in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), vol. 304. Berlin, Germany: Springer, 1987, pp. 5–14.

[46] S. Khazaei, S. Fischer, and W. Meier, "Reduced complexity attacks on the alternating step generator," in *Selected Areas in Cryptography* (Lecture Notes in Computer Science), vol. 4876. Berlin, Germany: Springer, 2007, pp. 1–16.

[47] I. Dinur and A. Shamir, "Cube attacks on tweakable black box polynomials," in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), vol. 5479. Berlin, Germany: Springer, 2009, pp. 278–299.

[48] X. Fu, X. Wang, X. Dong, and W. Meier, "A key-recovery attack on 855-round Trivium," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science), vol. 10992. Cham, Switzerland: Springer, 2018, pp. 160–184.

[49] R. Williams, C. P. Gomes, and B. Selman, "Backdoors to typical case complexity," in *Proc. IJCAI*, 2003, pp. 1173–1178.

[50] N. T. Courtois and G. V. Bard, "Algebraic cryptanalysis of the data encryption standard," in *Cryptography Coding* (Lecture Notes in Computer Science), vol. 4887. Berlin, Germany: Springer, 2007, pp. 152–169.

[51] A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu, "Symbolic model checking without BDDs," in *Tools and Algorithms for the Construction and Analysis of Systems* (Lecture Notes in Computer Science), vol. 1579. Berlin, Germany: Springer, 1999, pp. 193–207.

[52] D. Kroening, "Software verification," in *Handbook Satisfiability* (Frontiers in Artificial Intelligence and Applications), vol. 336, 2nd ed. Amsterdam, The Netherlands: IOS Press, 2021, pp. 791–818.

[53] M. J. H. Heule, O. Kullmann, and V. W. Marek, "Solving and verifying the Boolean Pythagorean triples problem via cube-and-conquer," in *Theory and Applications of Satisfiability Testing—SAT* (Lecture Notes in Computer Science), vol. 9710. Cham, Switzerland: Springer, 2016, pp. 228–245.

[54] N. Narodytska, S. Kasiviswanathan, L. Ryzhyk, M. Sagiv, and T. Walsh, "Verifying properties of binarized deep neural networks," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2018, pp. 1–10.

[55] A. Ignatiev, N. Narodytska, and J. Marques-Silva, "Abduction-based explanations for machine learning models," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2019, pp. 1511–1519.

[56] K. Chukharev and D. Chivilikhin, "fbSAT: Automatic inference of minimal finite-state models of function blocks using SAT solver," *IEEE Access*, vol. 10, pp. 131592–131610, 2022.

[57] I. Mironov and L. Zhang, "Applications of SAT solvers to cryptanalysis of hash functions," in *Theory and Applications of Satisfiability Testing—SAT* (Lecture Notes in Computer Science), vol. 4121. Berlin, Germany: Springer, 2006, pp. 102–115.

[58] A. Semenov, O. Zaikin, D. Bespalov, and M. Posypkin, "Parallel logical cryptanalysis of the generator A5/1 in BNB-grid system," in *Parallel Computing Technologies* (Lecture Notes in Computer Science), vol. 6873. Berlin, Germany: Springer, 2011, pp. 473–483.

[59] P. Erdös and A. Rényi, "On random graphs," *Pub. Math., Debrecen*, vol. 6, pp. 290–297, 1959.

[60] S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," *J. Theor. Biol.*, vol. 22, no. 3, pp. 437–467, Mar. 1969.

[61] G. Chartrand, G. L. Johns, K. A. McKeon, and P. Zhang, "Rainbow connection in graphs," *Mathematica Bohemica*, vol. 133, no. 1, pp. 85–98, 2008.

[62] Y. Shang, "A sharp threshold for rainbow connection in small-world networks," *Miskolc Math. Notes*, vol. 13, no. 2, pp. 493–497, 2012.

[63] Y. Shang, "Concentration of rainbow k-connectivity of a multiplex random graph," *Theor. Comput. Sci.*, vol. 951, Mar. 2023, Art. no. 113771.

[64] S. A. Cook, "The complexity of theorem-proving procedures," in *Proc. 3rd Annu. ACM Symp. Theory Comput.*, Shaker Heights, OH, USA, May 1971, pp. 151–158.

[65] L. A. Levin, "Universal sequential search problems," *Problemy Peredachi Inf.*, vol. 9, no. 3, pp. 115–116, 1973.

[66] T. Güneysu, T. Kasper, M. Novotný, C. Paar, and A. Rupp, "Cryptanalysis with COPACOBANA," *IEEE Trans. Comput.*, vol. 57, no. 11, pp. 1498–1513, Nov. 2008.

[67] V. Bulavintsev, A. Semenov, O. Zaikin, and S. Kochemazov, "A bitslice implementation of Anderson's attack on A5/1," *Open Eng.*, vol. 8, pp. 7–16, Mar. 2018.

[68] D. Coppersmith, H. Krawczyk, and Y. Mansour, "The shrinking generator," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science), vol. 773. Berlin, Germany: Springer, 1994, pp. 22–39.

[69] W. Meier and O. Staffelbach, "The self-shrinking generator," *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), vol. 950. Berlin, Germany: Springer, 1995, pp. 205–214.

[70] H. Raddum, "Cryptanalytic results on TRIVIUM," ECRYPT Stream Cipher Project, eSTREAM, Laguna Hills, CA, USA, Tech. Rep. 2006/039, 2006. [Online]. Available: https://www.ecrypt.eu.org/stream/papersdir/2006/039.ps

[71] C. Mcdonald, C. Charnes, and J. Pieprzyk, "Attacking Bivium with MiniSat," ECRYPT Stream Cipher Project, eSTREAM, Laguna Hills, CA, USA, Tech. Rep. 2007/040, 2007.
[72] J. Borghoff, L. R. Knudsen, and K. Matusiewicz, "Hill climbing algorithms and Trivium," in *Selected Areas in Cryptography* (Lecture Notes in Computer Science), vol. 6544. Berlin, Germany: Springer, 2011, pp. 57–73.
[73] N. T. Courtois, J. A. Gawinecki, and G. Song, "Contradiction immunity and guess-then-determine attacks on GOST," *Tatra Mountains Math. Publications*, vol. 53, no. 1, pp. 65–79, Dec. 2012.
[74] A. A. Semenov and O. Zaikin, "Using Monte Carlo method for searching partitionings of hard variants of Boolean satisfiability problem," in *Parallel Computing Technologies* (Lecture Notes in Computer Science), vol. 9251. Cham, Switzerland: Springer, 2015, pp. 222–230.
[75] C. Bouillaguet, P. Derbez, and P.-A. Fouque, "Automatic search of attacks on round-reduced AES and applications," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science), vol. 6841. Berlin, Germany: Springer, 2011, pp. 169–187.
[76] H. Hadipour and M. Eichlseder, "Autoguess: A tool for finding guess-and-determine attacks and key bridges," in *Applied Cryptography and Network Security* (Lecture Notes in Computer Science), vol. 13269. Cham, Switzerland: Springer, 2022, pp. 230–250, 2022.
[77] M. Heule, O. Kullmann, S. Wieringa, and A. Biere, "Cube and conquer: Guiding CDCL SAT solvers by lookaheads," in *Proc. Haifa Verification Conf. (HVC)*, in Lecture Notes in Computer Science, vol. 7261, pp. 50–65, 2011.
[78] A. E. J. Hyvärinen, "Grid based propositional satisfiability solving," Ph.D. thesis, Dept. Inf. Comput. Sci., Aalto Univ., Espoo, Finland, 2011.
[79] O. Zaikin and S. Kochemazov, "On black-box optimization in divide-and-conquer SAT solving," *Optim. Methods Softw.*, vol. 36, no. 4, pp. 672–696, 2019.
[80] I. Gribanova and A. Semenov, "Constructing a set of weak values for full-round MD4 hash function," in *Proc. 43rd Int. Conv. Inf., Commun. Electron. Technol. (MIPRO)*, Oct. 2020, pp. 1212–1217.
[81] O. Zaikin, "Inverting 43-step MD4 via cube-and-conquer," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 1894–1900.
[82] S. Gaspers and S. Szeider, "Backdoors to satisfaction," in *The Multivariate Algorithmic Revolution and Beyond* (Lecture Notes in Computer Science), vol. 7370. Berlin, Germany: Springer, 2012, pp. 287–317.
[83] A. A. Semenov, A. Pavlenko, D. Chivilikhin, and S. Kochemazov, "On probabilistic generalization of backdoors in Boolean satisfiability," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2022, pp. 10353–10361.
[84] T. J. Schaefer, "The complexity of satisfiability problems," in *Proc. 10th Annu. ACM Symp. Theory Comput. (STOC)*, 1978, pp. 216–226.
[85] G. P. Agibalov, "Logicheskye uravneniya v kriptoanalize generatorov kluchevogo potoka (Logical equations in cryptanalysis of keystream generators)," (in Russian), *Vestnik Tomskogo Gosudarstvennogo Universiteta, Prilozhenye, Bull. Tomsk State Univ.*, vol. 6, no. 6, pp. 31–41, Sep. 2003.
[86] N. E. Timoshevskaya, "Zadacha o kratchaishem linearizacionnom mnozhestve (The problem of finding the shortest linearizing set)," (in Russian), *Vestnik Tomskogo Gosudarstvennogo Universiteta, Prilozhenye, Bull. Tomsk State Univ.*, vol. 4, no. 14, pp. 79–83, Aug. 2005.

ALEXANDER SEMENOV received the specialist's degree in mathematics and the Ph.D. degree in computer science, in 1992 and 1998, respectively. From 2007 to 2021, he led the Laboratory of Discrete Analysis and Applied Logic, Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences (ISDCT SB RAS). He is currently a Senior Researcher with the Computer Technologies Laboratory, ITMO University. His research interests include computational complexity, cryptography, parallel computing, discrete optimization, combinatorial algorithms, SAT solving algorithms, and techniques.

KIRILL ANTONOV received the bachelor's degree in fundamental informatics and information technology from ISU, Irkutsk, Russia, in 2020, and the master's degree in information security from MEPhI University, Moscow, Russia, in 2022, where he is currently pursuing the Ph.D. degree.

He is also an Assistant Professor with the Department of Cryptography and Computer Systems Security, MEPhI University. His research interests include the analysis of symmetric cryptosystems and algebraic methods in cryptanalysis.

STEPAN KOCHEMAZOV received the specialist's degree in computer science from Irkutsk State University, in 2007.

Since 2010, he has been a Research Fellow with the Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences (ISDCT SB RAS), and since 2021, he has been with the Computer Technologies Laboratory, ITMO University. His research interests include SAT, SAT solvers, SAT encodings, and the applications of SAT solvers in practice.

ARTEM PAVLENKO received the bachelor's and master's degrees in applied mathematics and informatics from ITMO University, Saint Petersburg, Russia, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree with the Computer Technologies Laboratory.

He is also a Junior Research Associate with the Computer Technologies Laboratory, ITMO University. He studies software engineering, SAT solvers, metaheuristics and cryptanalysis, while working on the Ph.D. degree.