A Novel Technique in Linear Cryptanalysis

Wen-Long Sun
(Information Science and Technology Institute, Zhengzhou, China
swl_cipher@163.com)
Jie Guan
(Information Science and Technology Institute, Zhengzhou, China
guanjie007@163.com)
Lin Ding
(Information Science and Technology Institute, Zhengzhou, China
dinglin_cipher@163.com)

Abstract: In this paper, we focus on a novel technique called cube-linear attack, which is obtained by combining the cube and linear attacks together, is first proposed to deal with the probabilistic polynomial, aiming to furthermore mine the available secret information. Based on different combination ways of the two attacks, moreover, two cube-linear schemes are discussed. Naturally, we can use cube-linear attack as an unordinary trick in linear cryptanalysis, which has never been considered by the previous linear cryptanalysis yet. As a new contribution to linear cryptanalysis, it is beneficial to allow for a reduction in the amount of data required for a successful attack in specific circumstances. Applying our method to a reduced-round Trivium, as an example, we get better linear cryptanalysis results. More importantly, we believe that the novel linear cryptanalysis technique introduced in this paper can be extended to other ciphers. In other words, it is worth considering for our method in linear cryptanalysis.

Keywords: Cryptanalysis, Linear Cryptanalysis, Cube-Linear Attack, Trivium, Stream Cipher

1. Introduction

Linear cryptanalysis is an effective known-plaintext attack against block ciphers. At present, the attack has been adapted to stream ciphers [Golić, 94, Golić, 02, Muller, 05, Shahram, 05, Turan, 07]. Matsui and Yamagishi [Matsui, 92] in 1992 introduced the idea of linear cryptanalysis in an attack on FEAL [Shimizu, 87]. The techniques used in this attack were refined by Matsui and used dramatic effect on Data Encryption Standard (DES), eventually leading to the first experimental cryptanalysis of the cipher reported in the open community [Matsui, 93, Matsui, 94].

Subsequently, several refinements to the basic idea of linear cryptanalysis have been suggested to improve the efficiency of the attacks, either in specific circumstances or in all cases. In 1994, Kaliski and Robshaw [Kaliski, 94] proposed an extension based on the use of multiple linear approximations. Martin Hellman and Susan K. Langford [Langford, 94] in 1994 introduced the differential-linear attack which is a mix of both differential cryptanalysis and linear cryptanalysis. In 1996, Kundsen and Robshaw [Knudsen, 96] introduced the idea of extending Matsui's linear cryptanalytic techniques to the more general case in which non-linear relations are also considered. In [Bogdanov, 11], zero-correlation linear cryptanalysis, the counterpart of impossible differential cryptanalysis in the domain of linear cryptanalysis, was proposed by Bogdanov and Rijmen, resulting in a faster attack for some ciphers.

1.1 Motivation and Contribution

As introduced above, there have been several extensions to linear cryptanalysis at present. Nevertheless, is there any other method able to be exploited as a new contribution to linear cryptanalysis? Namely, is there any available information unexposed by the previous linear cryptanalysis?

In this paper, we answer this question positively. Generally speaking, linear cryptanalysis exploits specific correlations between the input and the output of cryptographic primitives. For almost any cryptographic scheme, each output bit can be described by a multivariate polynomial over $GF(2)$ of public variables and secret variables. When launching a linear attack, we can in specific scenarios

obtain an explicit description of multivariate polynomial. For the tweakable polynomial, it is cube attack proposed in [Dinur, 09] that is a very powerful tool to recover the key information. Once recovering the key information contained in the polynomial, the degrees of non-linear monomials involving these recovered bits will decrease, even fall to zero. It's just the desired purposes. As we know, the cube attack technique is used to deal with such a polynomial with probability one. Due to the above polynomial usually obtained with probability less than one after some approximations, however, the actual use of cube attack has to be adapted for the lower overheads. Here, we combine the cube and linear attacks together to recover the available key information, called cube-linear attack. Subsequent analysis also proves that our method works well. Since there is still some key information able to be recovered by cube-linear attack, which has never been considered by the previous linear cryptanalysis yet, we deeply confirm that there must be more or less improvements for the subsequent linear cryptanalysis.

Particularly based on different combination ways of cube and linear attacks, two schemes A and B are given in order to achieve a lower data complexity required for a successful attack. The data complexity and the success rate of each scheme are discussed only when all the derived polynomials defined by any determined cube index are independent. Otherwise, the situations are too complicated to obtain any concrete results about the data complexity and the success rate of the two schemes, if using the existing theory (See Section 3). A detailed description of our theory appears in Section 3.

Afterwards, an improved linear cryptanalysis technique is proposed using the cube-linear attack. As an application, we cryptanalyze the security of the eSTREAM finalist Trivium against linear cryptanalysis. Three linear approximations with the same average bias $2^{-23.2}$ are found and four key bits are recovered for the reduced version of Trivium with the initialization of 288 rounds (out of 1152). The data complexity $2^{47.2}$ IVs is enough to achieve this result with 97.8% success rate, improving the previous linear cryptanalysis results. Although a few better cryptanalytic results on Trivium had been published several years earlier using other attacks [Vielhaber, 07, Dinur, 09], however, we believe that our methods are meaningful from the point of view of improving linear cryptanalysis.

Notably, there are some differences between cube attack and our method, although based on the same essence of higher order differential cryptanalysis. For convenient to introduce, we follow the relevant concepts and terminology of cube attack. In order not to cause misunderstanding, however, it is necessary to explain the differences as follows. Firstly, as stated previously, the cube attack is applied to a polynomial with the probability one, while what our method copes with is the probabilistic polynomial. Secondly, the primary costs of cube attack lie in searching for the appropriate cube indexes, while our method focuses on dealing with an explicit polynomial. More importantly, we argue that it is not appropriate to make comparisons between cube attack and linear cryptanalysis, since they are two different methods. In terms of the cryptanalysis, one in fact should try various kinds of methods and what we do in this paper is to put forward a trick called cube-linear attack which could exactly result in more efficient linear cryptanalysis. In other words, it is worth considering for our method in launching linear attacks on other ciphers.

1.2 Organization

This paper is organized as follows. In Section 2 we briefly review cube attack and linear cryptanalysis. Afterwards, we describe the cube-linear attack and propose an improved linear cryptanalysis in Section 3. In Sections 4, we apply our method to a specific analysis of Trivium. Finally, we make a few concluding remarks in Section 5.

2. Cube Attack and Linear Cryptanalysis

2.1 A Review of Cube Attack

Cube attack [Dinur, 09], first formalized by Itai Dinur and Adi Shamir at EUROCRYPT 2009, is a generic type of algebraic attack that may be applied against any cryptosystem, provided that the

attacker has access to a bit of information that can be represented by a "low-degree" multivariate polynomial over *GF*(2). In essential, the cube attack, similar to AIDA [Vielhaber, 07], is closely related to the previously known attack—the higher order differential attacks [Lai, 94].

In almost any cryptographic scheme, each output bit z_i can be described by a multivariate master polynomial over *GF*(2) of public variables $v_1, v_2, \cdots, v_m$ being bits of the plaintext for block cipher or bits of the initial vector for stream cipher and depending on secret variables $x_1, x_2, \cdots, x_n$ being bits of the key: $z_i = p(v_1, \ldots, v_m, x_1, \ldots, x_n)$.

To simplify our notation, we now ignore the distinction between public and private variables. Given a multivariate master polynomial with *n* variables $p(x_1, \ldots, x_n)$ over *GF*(2) in algebraic normal form (ANF), and a term t_I containing variables from an index subset I that are multiplied together, the polynomial can be written as the sum of terms which are supersets of I and terms that miss at least one variable from I:

$$p(x_1, \ldots, x_n) \equiv t_I \cdot P_{S(I)} + q(x_1, \ldots, x_n)$$

where $P_{S(I)}$ is called the superpoly of I in p. Note that the superpoly of I in p is a polynomial that does not contain any common variable with t_I, and each term in $q(x_1, \ldots, x_n)$ does not contain at least one variable form I.

Any subset I of size *s* defines a *s*-dimensional Boolean cube of 2^s vectors C_I in which we assign all the possible combinations of 0/1 values to variables in I, and leave all the other variables undetermined. Any vector $v \in C_I$ defines a new derived polynomial $p_{|v}$ with $n - s$ variables (whose degree may be the same or lower than the degree of the original polynomial). Summing these derived polynomials over all the 2^s possible vectors in C_I, we end up with a new polynomial, which is denoted by $p_I \stackrel{\Delta}{=} \sum p_{|v}$.

Theorem1. [Dinur, 09] For any polynomial *p* and subset of variables I, $p_I \equiv p_{S(I)}$ modulo 2.

The attack is a known plaintext one and it has two phases. In the first preprocessing phase the task is to find as many maxterms t_I and corresponding linear superpolys $P_{S(I)}$ as possible. In the next online phase, the attacker solves the system of linear equations obtained and gets some values about the secret variables.

2.2 A Review of Linear Cryptanalysis

The basic idea behind linear cryptanalysis is to find some linear approximations to the action of cryptographic primitives. In other words, the attack exploits some statistical correlations between input and output bits. For a cryptographic primitive with *k*-bit key $(k_1, k_2, \ldots, k_k)$, *n*-bit plaintext $(p_1, p_2, \ldots, p_n)$ and ciphertext $(c_1, c_2, \ldots, c_n)$, the aim of the attack is to find the index sets I, J, L such that

$$\sum_{j \in J} p_j \oplus \sum_{l \in L} c_l = \sum_{i \in I} k_i \qquad (1)$$

holds with probability $p = 1/2 + \varepsilon, \varepsilon \neq 0$.

For an iterated block cipher or stream cipher, linear cryptanalysis is usually executed as follows. By looking for and combining the linear or nonlinear approximations of different rounds, the final linear approximations of the whole cipher can be found with probability calculated according to Lemma 1 (Piling-Up Lemma).

Lemma1. [Matsui, 93] For each value $(1 \le i \le n)$, let X_i be a random variable, independent of X_i for all $j \ne i$, such that $P(X_i = 0) = p_i$, $P(X_i = 1) = 1 - p_i$. Then

$$P(X_1 \oplus X_2 \oplus \ldots \oplus X_n = 0) = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} (p_i - 1/2)$$

Algorithm1: Determination of Key Information

T : = # of plaintexts (out of N) such that the left side of (1) is equal to 0.
IF $T > N/2$
THEN guess $\sum k_i = 0$ (when $p > 1/2$) or 1 (otherwise)
ELSE guess $\sum k_i = 1$ (when $p > 1/2$) or 0 (otherwise)
END

Given an effective linear approximation, it is possible to determine one bit of information about the key $\sum_{i \in I} k_i$ with the help of Algorithm 1 [Matsui, 93], the indices I being fixed by linear expression; its core is a maximum-likelihood.

The success rate to recover the key $\sum_{i \in I} k_i$ is as follows [Matsui, 93], which is related to both data complexity N (the number of plaintext/ciphertext pairs) and bias ε :

$$\gamma = \frac{1}{\sqrt{2\pi}} \int_{-2\sqrt{N}\varepsilon}^{\infty} e^{-x^2/2} \, dx$$

The most important goal of linear cryptanalysis is to find the best linear approximation, i.e. the linear expression which holds with bigger bias. However, there have thus far no optimal algorithms to find the best linear approximations. In this paper, we tentatively put forward a novel technique as a contribution to linear cryptanalysis, namely cube-linear attack to be introduced next.

3. An Improved Linear Cryptanalysis

3.1 Cube-Linear Attack

This subsection provides the basic idea of our cube-linear attack to deal with the probabilistic polynomial usually emerging in linear cryptantlysis. Generally speaking, linear cryptanalysis exploits specific correlations between the input and the output of cryptographic primitives. For almost any cryptographic scheme, each output bit can be described by a multivariate master polynomial over $GF(2)$ of public variables and secret variables. During the linear attack, we can in specific scenarios obtain such an explicit description of polynomial $z_i = p(v,k)$ with probability p^* which is easily split into the form $p(x_1,\ldots,x_n) \equiv t_I \cdot P_{S(I)} + q(x_1,\ldots,x_n)$ for any term t_I based on cube attack. Naturally, we can determine the maxterm t_I leading to an expression $P_{S(I)}$ of which the degree $d\left(P_{S(I)}\right)$ is one. Furthermore, the key information contained in $P_{S(I)}$ can be recovered. As we know, the cube attack is usually used to deal with such a polynomial with probability one. Due to the above polynomial usually holding with probability less than one after some approximations, however, the actual use of cube attack has to be adapted. Here, we combine the cube and linear attacks together in order to achieve a lower data complexity required for a successful attack. Based on the different combination ways of the two approaches, we give two schemes, Scheme A and Scheme B, to recover the information unexposed by the previous linear cryptanalysis techniques under the following assumption.

Assumption1. All the derived polynomials defined by any determined maxterm t_I are independent.

3.1.1 Description of Scheme A

Scheme A, the adapted cube attack, can also be considered as "cube-linear-cube attack", corresponding to the three steps of Scheme A. The below analysis show more detailed explanations about Scheme A. Based on the split polynomial $p(v,k) \equiv t_I \cdot P_{S(I)} + q(v,k)$ as above, we can easily determine the maxterm t_I leading to an expression $P_{S(I)}$ of which the degree $d\left(P_{S(I)}\right)$ is one. Correspondingly, I and C_I are all distinct. Running all the possible values of C_I, any vector $v \in C_I$ can define a derived polynomial $p_{|v}$ with probability p_v^* (p_v^* represents the probability of $p_{|v}$ when C_I takes the value v in condition that the polynomial $z_i = p(v,k)$ holds with probability p*). Denote $K_{|v}$ the XOR of all the monomials involving only the key in the derived polynomial $p_{|v}$. Then $K_{|v}$ can be determined with the help of Algorithm 1, since the derived polynomial $p_{|v}$ is a linear approximation when considering $K_{|v}$ as a single variable. Similar to cube attack, we obtain a new linear equation when summing all the recovered $K_{|v}$.

SCHEME A
Step1. Determine the maxterm t_I.
Step2. Recover $K_{|v}$ in each derived polynomial $p_{|v}$ by Algorithm 1.
Step3. Sum all the recovered $K_{|v}$, then we know the value $\sum_{v \in C_I} K_{|v}$.

Theorem2. Data complexity of Scheme A is $N_A = \sum_{v=0}^{2^s-1} N_v = O\left(\sum_{v=0}^{2^s-1} \left(\left|p_v^* - 1/2\right|\right)^{-2}\right)$, and the success rate of Scheme A is $\gamma_A = 1/2 + 2^{2^s-1} \prod_{v=0}^{2^s-1} (\gamma_v - 1/2)$, where N_v and γ_v are respectively referred to as the data complexity and success rate to recover $K_{|v}$ using Algorithm 1, and s is the size of index subset I.

Proof. Since the probability of each derived polynomial $p_{|v}$ is p_v^*, the data complexity to recover $K_{|v}$ is easily calculated as $N_v = O\left(\left|p_v^* - 1/2\right|^{-2}\right)$ with a confident success rate γ_v according to Algorithm 1. The total data complexity of Scheme A is obviously $N_A = \sum_{v=0}^{2^s-1} N_v = O\left(\sum_{v=0}^{2^s-1} \left(\left|p_v^* - 1/2\right|\right)^{-2}\right)$.

Let γ_v^* be the failure rate to recover $K_{|v}$, then $\gamma_v^* = 1 - \gamma_v$. For simplicity, we may as well denote "0" and "1" the right value of $K_{|v}$, and the wrong value of $K_{|v}$ respectively. Then the probability to recover the right value of $K_{|v}$ and the wrong value of $K_{|v}$ using Algorithm 1 are $p\left(K_{|v} = 0\right) = \gamma_v$ and $p\left(K_{|v} = 1\right) = \gamma_v^*$. Based on the Assumption 1, we deduce that all the recovered $K_{|v}$ independently hold each other. So the success rate of Scheme A is $\gamma_A = p\left(\bigoplus_{v=0}^{2^s-1} K_{|v} = 0\right) = 1/2 + 2^{2^s-1} \prod_{v=0}^{2^s-1} (\gamma_v - 1/2)$ by Piling-Up Lemma. □

3.1.2 Description of Scheme B

SCHEME B
Step1. Determine the maxterm t_I.
Step2. Sum all the derived polynomials $p_{\|v}$, then we have $p_{S(I)} \overset{\Delta}{=} \sum_{v \in C_I} p_{\|v}$.
Stpe3. Recover the key information contained in $p_{S(I)}$ by Algorithm 1.

Corresponding to the above three steps, Scheme B can be considered as "cube-linear attack". The first two steps are actually the use of cube attack, then the key information contained in $p_{S(I)}$ can be determined with the help of Algorithm 1.

Theorem3. The data complexity of Scheme B is $N_B = O\left(\left(2^{2^s-1}\prod_{v=0}^{2^s-1}\left|p_v^* - 1/2\right|\right)^{-2}\right)$ with a confident success rate γ_B.

Proof. Under the above Assumption 1, all the derived polynomials $p_{\|v}$ defined by $v \in C_I$ independently hold with probability p_v^*. So the bias that $p_{S(I)} = \sum_{v \in C_I} p_{\|v}$ holds is calculated as $2^{2^s-1}\prod_{v=0}^{2^s-1}\left|p_v^* - 1/2\right|$ due to Piling-up Lemma. Using Algorithm 1, the data complexity N_B to recover the key information contained in $p_{S(I)}$ with a confident success rate γ_B is $O\left(\left(2^{2^s-1}\prod_{v=0}^{2^s-1}\left|p_v^* - 1/2\right|\right)^{-2}\right)$. □

3.1.3 Comments on Scheme A and Scheme B

It should be noted that the recovered information using Scheme A is the same as Scheme B.

Theorem4. When the probability of each derived polynomial p_v^* satisfies $\left|p_v^* - 1/2\right| \le 2^{-1.5}$, then $N_A \le \frac{2^s}{2^s - 1} N_B$ (the equality sign holds when $\left|p_v^* - 1/2\right| = 2^{-1.5}$), where N_A and N_B respectively represent the data complexity of Scheme A and Scheme B, and s is the size of index subset I.

Proof. According to Theorem 2 and Theorem 3, $N_A = O\left(\sum_{v=0}^{2^s-1}\left(\left|p_v^* - 1/2\right|\right)^{-2}\right)$ and $N_B = O\left(\left(2^{2^s-1}\prod_{v=0}^{2^s-1}\left|p_v^* - 1/2\right|\right)^{-2}\right)$. Let $x_v = \left(2 \cdot \left|p_v^* - 1/2\right|\right)^{-2}$, then $x_v \in (1, +\infty)$, $N_A = 4\sum_{v=0}^{2^s-1}(x_v)$ and $N_B = 4 \cdot \prod_{v=0}^{2^s-1}(x_v)$. If $N_A = N_B$ and all x_v share the same value, we deduce $x_v = 2^{s/(2^s-1)}$.

When $x_v \ge 2^{s/(2^s-1)}$, we have the following

$$\begin{aligned}\frac{\sum_{v=0}^{2^s-1} x_v}{\prod_{v=0}^{2^s-1} x_v} &= \frac{x_0 + x_1 + \cdots + x_{2^s-1}}{x_0 x_1 \cdots x_{2^s-1}} \\ &= \frac{1}{x_1 x_2 \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{i-1} x_{i+1} \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{2^s-2}} \\ &\le \frac{1}{2^s} + \cdots + \frac{1}{2^s} + \cdots + \frac{1}{2^s} = \frac{1}{2^s} \cdot 2^s = 1\end{aligned}$$

Specially, when $x_v \ge 2$, i.e. $\left|p_v^* - 1/2\right| \le 2^{-1.5}$, the following

$$\frac{\sum_{v=0}^{2^s-1} x_v}{\prod_{v=0}^{2^s-1} x_v} = \frac{x_0 + x_1 + \cdots + x_{2^s-1}}{x_0 x_1 \cdots x_{2^s-1}}$$

$$= \frac{1}{x_1 x_2 \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{i-1} x_{i+1} \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{2^s-2}}$$

$$\le \frac{1}{2^{2^s-1}} + \cdots + \frac{1}{2^{2^s-1}} + \cdots + \frac{1}{2^{2^s-1}} = \frac{2^s}{2^{2^s-1}}$$

holds, i.e. $N_A \le \frac{2^s}{2^{2^s-1}} N_B$. □

We have proved that $N_A \le \frac{2^s}{2^{2^s-1}} N_B \le N_B$ only when the probability p_v^* of each derived polynomial satisfies $\left|p_v^* - 1/2\right| \le 2^{-1.5}$. The results haven't yet been given for the remaining scenarios. With regard to the success rate of the two schemes, $\gamma_A = 1/2 + 2^{2^s-1} \prod_{v=0}^{2^s-1} (\gamma_v - 1/2)$, whereas r_B is equal to r_v depending on the relationship between the data complexity and the bias. If no more plaintext/ciphertext pairs, the success rate of Scheme A will be lower than Scheme B. **Table 1** provides some comparisons between Scheme A and Scheme B in the same level of success rate.

Table 1. Comparison Between Scheme A and Scheme B

$\left\|p_v^* - 1/2\right\|$ / s	$2^{-1.3}$		2^{-3}		2^{-6}		2^{-9}		Success Rate/%
	N_A	N_B	N_A	N_B	N_A	N_B	N_A	N_B	
1	2^4	$2^{3.2}$	$2^{7.4}$	2^{10}	$2^{13.4}$	2^{22}	$2^{19.4}$	2^{34}	97.7
2	$2^{5.3}$	$2^{4.4}$	$2^{8.7}$	2^{18}	$2^{14.7}$	2^{42}	$2^{20.7}$	2^{66}	
3	$2^{6.5}$	$2^{6.8}$	$2^{9.9}$	2^{28}	$2^{15.9}$	2^{82}	$2^{21.9}$	2^{130}	

It shows that Scheme A is in fact more superior in the major scenarios because of the following reasons. First of all, since the probability of the derived polynomial p_v^* in the common cases satisfies $\left|p_v^* - 1/2\right| << 2^{-1.5}$, this means that $N_A << N_B$ according to our calculations. Secondly, the success rate r_A wouldn't quickly decrease since $\gamma_A = 1/2 + 2^{2^s-1} \prod_{v=0}^{2^s-1} (\gamma_v - 1/2)$. Meanwhile, it can be enhanced through increasing a few plaintext/ciphertext pairs by the theory of linear cryptanalysis. Taken together, however, this should depend on the particular situation that when to choose Scheme A or Scheme B.

REMARK1. For both Scheme A and Scheme B, the first thing in practice we need to do after determining the maxterm t_I is to verify that whether Assumption 1 comes into existence or not. If it is tenable, then we have the above theorems. Otherwise, the situations are too complicated to obtain any concrete results about the data complexity and success rate of the two schemes using the present theory.

3.2 An Improved Linear Cryptanalysis

In the probabilistic polynomial, it is the key information recovered by our cube-linear attack that has never been exposed by the previous linear cryptanalysis, thus we confirm that the cube-linear attack indeed provides us with a paradise of improvement for the linear cryptanalysis. Overall, the improved linear cryptanalysis technique can be summarized as follows.

AN IMPROVED LINEAR CRYPTANALYSIS

When obtaining a polynomial $z_i = p(v,k)$ with probability p^*

Step1. Verify that whether Assumption 1 comes into existence or not, if it is tenable, go on next; Otherwise, our theory won't work.

Step2. Based on Theorem 2, 3 and 4, choose a better scheme (Scheme A or Scheme B) to recover the key information.

Step3. Combining the recovered key information, carry on linear cryptanalysis and look for its linear approximation ρ.

Denote *E* the event that a recovery of as many key information as possible through calling Scheme A or Scheme B. N_E and γ_E are referred to as the data complexity and the success rate that the event *E* holds. Let N_ρ and γ_ρ respectively denote the data complexity and the success rate to determine the combined key information in the linear approximation ρ using Algorithm 1.

Theorem5. For the improved linear cryptanalysis as above, the data complexity N is $N_E + N_\rho$ and the success rate γ is $\gamma_E \cdot \gamma_\rho$ at least.

Proof. With regard to the improved linear cryptanalysis, the data complexity N sticks out a mile to be $N_E + N_\rho$. As to the success rate γ, it consists of the two parts as follows. If the event *E* holds, then the final linear approximation ρ must be right. Otherwise, the final linear approximation ρ must be wrong, but it is also possible that we can successfully determine the combined key information in the linear approximation ρ using Algorithm 1. For example, we have the following

$$z = iv_m k_n + \sum_{j \in J} iv_j + \sum_{i \in I} k_i$$

where k_n can be recovered by adopting cube-linear attack, $n \notin I$. We suppose the right and the wrong value of k_n are "0" and "1" respectively. If the value of k_n recovered by us is 1, we obtain a wrong linear approximation $z = iv_m + \sum_{j \in J} iv_j + \sum_{i \in I} k_i$. Fortunately, when all the data used by Algorithm 1 satisfies the condition that $iv_m = 0$, we can also successfully determine the information $\sum_{i \in I} k_i$. For the former situation, the success rate is $\gamma_E \cdot \gamma_\rho$; While for the latter one, when we ensure that the event *E* hold with high level of success rate, it can be omitted. Therefore, the success rate is at least $\gamma_E \cdot \gamma_\rho$. □

As an application, we will cryptanalyze the security of the eSTREAM finalist Trivium against linear cryptanalysis in the next section.

4. Improved Linear Cryptanalysis of Reduced Trivium

4.1 Trivium Stream Cipher

Trivium [Cannière, 05], a hardware-oriented stream cipher, was designed by De Cannière and Preneel and was selected for the final eSTREAM portfolio [eSTREAM, 04]. It takes a 80-bit key *K* and a 80-bit initial value IV as input. The internal state consists of 288 bits which are aligned in three non-linear feedback shift registers of lengths 93, 84 and 111, respectively. It is claimed to be suitable to generate up to 2^{64} bits of keystream from a pair of key and IV. They are initialized as follows:

$$(s_1, s_2, \cdots, s_{93}) \leftarrow (k_1, \cdots, k_{80}, 0, \cdots, 0)$$
$$(s_{94}, \cdots, s_{177}) \leftarrow (iv_1, \cdots, iv_{80}, 0, \cdots, 0)$$
$$(s_{178}, \cdots, s_{288}) \leftarrow (0, \cdots, 0, 1, 1, 1)$$

The state is then updated iteratively by the following round transformation:

$$t_1 \leftarrow s_{66} + s_{93}$$

$$
\begin{aligned}
t_2 &\leftarrow s_{162} + s_{177} \\
t_3 &\leftarrow s_{243} + s_{288} \\
z &\leftarrow t_1 + t_2 + t_3 \\
t_1 &\leftarrow s_{66} + s_{91} \cdot s_{92} + s_{93} + s_{171} \\
t_2 &\leftarrow s_{162} + s_{175} \cdot s_{176} + s_{177} + s_{264} \\
t_3 &\leftarrow s_{243} + s_{286} \cdot s_{287} + s_{288} + s_{69} \\
(s_1, s_2, \cdots, s_{93}) &\leftarrow (t_3, s_1, \cdots, s_{92}) \\
(s_{94}, s_{95}, \cdots, s_{177}) &\leftarrow (t_1, s_{94}, \cdots, s_{176}) \\
(s_{178}, s_{179}, \cdots, s_{288}) &\leftarrow (t_2, s_{178}, \cdots, s_{287})
\end{aligned}
$$

No output is produced during the first 1152 rounds. After this initialization phase the value of z is output as the key stream at each round.

4.2 Description of Attack Process

For 288-round Trivium, the following

$$z_1 = s_{288}(66) + s_{288}(93) + s_{288}(162) + s_{288}(177) + s_{288}(243) + s_{288}(288) \tag{2}$$

holds, where $s_t(i)$ is the i^{th} internal state bit at time t.

The output z_1 is the sum bits of bits $s_{288}(66), s_{288}(93), s_{288}(162), s_{288}(177), s_{288}(243)$ and $s_{288}(288)$. The algebraic normal form of z_1 is found exhaustively in terms of the internal state bit at $t = 144$ as [Turan, 07]:

$$
\begin{aligned}
z_1 = {} & s_{144}(6) + s_{144}(16) \cdot s_{144}(17) + s_{144}(31) \cdot s_{144}(32) + s_{144}(33) + s_{144}(57) + s_{144}(82) \cdot s_{144}(83) + \\
& s_{144}(84) + s_{144}(96) + s_{144}(97) \cdot s_{144}(98) + s_{144}(99) + s_{144}(111) + s_{144}(129) + s_{144}(142) \cdot s_{144}(143) + \\
& s_{144}(144) + s_{144}(150) + s_{144}(162) + s_{144}(163) \cdot s_{144}(164) + s_{144}(165) + s_{144}(186) + s_{144}(192) + \\
& s_{144}(208) \cdot s_{144}(209) + s_{144}(210) + s_{144}(231) + s_{144}(235) \cdot s_{144}(236) + s_{144}(237) + s_{144}(252)
\end{aligned} \tag{3}
$$

and its closest linear approximation [Turan, 07] is

$$
\begin{aligned}
z_1 = {} & s_{144}(6) + s_{144}(33) + s_{144}(57) + s_{144}(84) + s_{144}(96) + s_{144}(99) + s_{144}(111) + \\
& s_{144}(129) + s_{144}(144) + s_{144}(150) + s_{144}(162) + s_{144}(165) + s_{144}(186) + \\
& s_{144}(192) + s_{144}(210) + s_{144}(231) + s_{144}(237) + s_{144}(252)
\end{aligned} \tag{4}
$$

with bias 2^{-9}.

Since the aim of the attack is to obtain an linear approximation based on key, IV and output bits, the linear approximation given above is to be rewritten in terms of $s_0(i)$, $i = 1, 2, \cdots, 80$ and $i = 94, 95, \cdots, 173$. The remaining terms are omitted since they are assigned to constants during the initialization. Then we have the right equation (5) (See Appendix) [Sun, 12], whereas it should be noted that there is something wrong, which will affect the subsequent results, in the one given by Turan and Kara [Turan, 07].

Next, we show that there are still some improvements for our purpose when using cube-linear attack. For the sake of analysis, denote $K_{monomials}$ and $IV_{monomials}$ the XOR of monomials involving only key bits, and the XOR of monomials involving only IV bits respectively. Set the XOR of the remaining monomials as $(IV \cdot K)_{monomials}$. Then, Eq.(5) is written as follows.

$$
\begin{aligned}
z_1 = K_{monomials} + IV_{monomials} + (IV \cdot K)_{monomials} = K_{monomials} + IV_{monomials} + \\
iv_{25} \cdot k_{14} + iv_{25} \cdot k_{41} + iv_{25} \cdot k_{39} \cdot k_{40} + iv_{26} \cdot k_{13} + iv_{26} \cdot k_{40} + iv_{26} \cdot k_{38} \cdot k_{39} + \\
iv_{31} \cdot k_{20} + iv_{31} \cdot k_{47} + iv_{31} \cdot k_{45} \cdot k_{46} + iv_{32} \cdot k_{19} + iv_{32} \cdot k_{46} + iv_{32} \cdot k_{44} \cdot k_{45} + \\
iv_{49} \cdot k_{38} + iv_{49} \cdot k_{65} + iv_{49} \cdot k_{63} \cdot k_{64} + iv_{50} \cdot k_{37} + iv_{50} \cdot k_{64} + iv_{50} \cdot k_{62} \cdot k_{63} + \\
iv_{70} \cdot k_{59} + iv_{71} \cdot k_{58} + iv_{76} \cdot k_{65} + iv_{77} \cdot k_{64}
\end{aligned} \tag{6}
$$

Observing Eq.(6), it is easy to split it into the form $p(x_1,\ldots,x_n) = t_I \cdot P_{S(I)} + q(x_1,\ldots,x_n)$ and to determine that whether there is such an index subset I which leads to a linear expression $P_{S(I)}$ or not, based on the basic idea of cube attack. Actually, the four index subsets {70}, {71}, {76}, and {77} in Eq.(6) are available to separately recover k_{58}, k_{59}, k_{64}, and k_{65}. Taking the index subset I={77} as a case in point , we will explain how to recover k_{64} utilizing our method.

Step 1: Determine the index subset I = {77}, and correspondingly, C_I = {iv_{77}}.

Step 2: According to cube attack, we can assign the other public variables any values except the variables belonging to C_I. For simplicity, set these IV bits iv_{25}, iv_{26}, iv_{31}, iv_{32}, iv_{49} and iv_{50} zeros, and the other IV bits any binary values.

a) When iv_{77}=0, i.e. (iv_{25}, iv_{26}, iv_{31}, iv_{32}, iv_{49}, iv_{50}, iv_{70}, iv_{71}, iv_{76}, iv_{77})=(0, 0, 0, 0, 0, 0, 0, 0, 0, 0), the following

$$K_{monomials} = z \tag{7}$$

holds with bias 2^{-9}, where z represents the XOR of all the known variables in Eq.(6) after assigning IV.

b) When iv_{77}=1, i.e. (iv_{25}, iv_{26}, iv_{31}, iv_{32}, iv_{49}, iv_{50}, iv_{70}, iv_{71}, iv_{76}, iv_{77})=(0, 0, 0, 0, 0, 0, 0, 0, 0, 1), the following

$$k_{64} + K_{monomials} = z' \tag{8}$$

holds with bias of 2^{-9}, where z' represents the XOR of all the known variables in Eq.(6) after assigning IV.

Note that Eq.(7) and Eq.(8) are independent, since the key bit k_{64} can be considered as a random variable. According to Theorem 4 and Table 1, Scheme A can here achieve a lower data complexity in the same level of success rate compared to Scheme B. Therefore, we can separately recover $K_{monomials}$ and $k_{64} + K_{monomials}$ with 99.77% success rate according to Scheme A, requiring the same data complexity 2^{19} IVs.

Step 3: From the above analysis, the following

$$k_{64} = (k_{64} + K_{monomials}) + (K_{monomials})$$

holds with 99.54% success rate, requiring 2^{20} IVs. Here, the total success rate 99.54% is approximately calculated as 0.9977^2 since the failure rate (1-0.9977) is so small that $(1\text{-}0.9977)^2$ can be ignored according to Theorem 2.

Similarly, when we choose the other index subsets {70}, {71} and {76}, k_{58}, k_{59}, and k_{65} can be recovered correspondingly. Therefore, we can simultaneously recover four key bits k_{58}, k_{59}, k_{64}, and k_{65} with 98.17% (0.9977^8) success rate, requiring about $2^{21.32}$ IVs ($5\times 2^{19} \approx 2^{21.32}$). **Table 2** gives these results.

Table2. The Key Information Recovered

(iv_{25},iv_{26},iv_{31},iv_{32},iv_{49},iv_{50},iv_{70},iv_{71},iv_{76},iv_{77})	Information Recovered	Bias	Data Complexity	Success Rate /%
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0)	$K_{monomials}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 1, 0, 0, 0)	$K_{monomials} + k_{59}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 0, 1, 0, 0)	$K_{monomials} + k_{58}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 0, 0, 1, 0)	$K_{monomials} + k_{65}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 0, 0, 0, 1)	$K_{monomials} + k_{64}$	2^{-9}	2^{19}	99.77

4.3 Search Better Linear Approximations

The above analysis provides us with a paradise of improvement for searching better linear approximations. Note that, we aren't aware of the exact values of these recovered four key bits depending on the concrete scenarios. Therefore, we now have to consider about all 16 possible values of these four bits. In the following, we take (k_{58}, k_{59}, k_{64}, k_{65}) = (1, 1, 1, 1) for an example to illustrate how to find linear approximations with bigger bias.
Return (k_{58}, k_{59}, k_{64}, k_{65}) = (1, 1, 1, 1) to Eq.(5), we have the equation (9) (See Appendix).

Observing Eq.(9), 34 linear, 48 quadratic terms and 18 cubic terms are included in it. The linear approximation of Eq.(9) can be naturally found as

$$\begin{aligned} z_1 = 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} + k_{39} + k_{54} + k_{57} + k_{63} + k_{67} + k_{68} + k_{69} + k_{72} + iv_3 + \\ iv_6 + iv_{21} + iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{49} + iv_{50} + iv_{51} + iv_{70} + iv_{71} + iv_{72} + iv_{76} + iv_{77} + iv_{78} \end{aligned} \quad (10)$$

with bias $2^{65} \cdot (0.25)^{48} \cdot (0.375)^{18} = 2^{-56.56}$, assuming all nonlinear terms are independent.

According to Eq.(3) ~ Eq.(10), the bias that Eq.(10) holds can be calculated as $\varepsilon = 2 \cdot 2^{-9} \cdot 2^{-56.56} = 2^{-64.56} < 2^{-|IV|/2} = 2^{-40}$ ($|IV|$ is referred to as the size of IV, and especially it is 80 bits for Trivium). The bias is too little to be used to recover any information about the key. Nevertheless, we can increase the magnitude of the bias when allowed to choose some special key bits and IV bits.

Denote $\Omega_K = \{k_i \mid k_i = 0, 1 \le i \le 80\}$ and $\Omega_{IV} = \{iv_i \mid iv_i = 0, 1 \le i \le 80\}$ the set of the key bits chosen to zeros, and the set of those IV bits chosen to zeros respectively. $|\Omega_K|$ and $|\Omega_{IV}|$ separately represent the size of Ω_K and Ω_{IV}, and n_1 and n_2 are referred to as the individual quantity of the remaining quadratic terms and the cubic terms after choosing Ω_K and Ω_{IV}. The bias of a linear approximation of Eq.(9) is calculated as follows:

$$\varepsilon = 2 \cdot 2^{-9} \cdot 2^{(n_1+n_2)-1} \cdot (0.25)^{n_1} \cdot (0.375)^{n_2} = (0.5)^{n_1+9} \cdot (0.75)^{n_2}$$

When given $|\Omega_K|$ and $|\Omega_{IV}|$, we can make use of Algorithm 2 [Sun, 12] (See Appendix) to choose Ω_K and Ω_{IV}. Algorithm 2 takes finding better linear approximations as a criterion and the main idea is to select such a bit which can eliminate the most amount of nonlinear monomials when it's set zero. There are also more detailed rules to deal with more complicated scenarios. In short, Algorithm 2 has proved to be the best approach to look for linear approximations when given $|\Omega_K|$ and $|\Omega_{IV}|$. When $|\Omega_K| = 10$ and $|\Omega_{IV}| = 10$, **Table 3** provides the chosen Ω_K and Ω_{IV} using Algorithm 2.

Table 3. (Ω_K, Ω_{IV}) and Corresponding Bias ε

$(\Omega_{IV}, \lvert\Omega_{IV}\rvert = 10)$	$(\Omega_K, \lvert\Omega_K\rvert = 10)$		Bias ε
iv_{10} iv_{13} iv_{25} iv_{31} iv_{34} iv_{37} iv_{40} iv_{50} iv_{54} iv_{55}	k_{13} k_{19} k_{23} k_{38} k_{40} k_{45} k_{46} k_{50} k_{63}	k_5	2^{-23}
		k_{67}	2^{-23}
		k_{68}	2^{-23}

According to the different Ω_K of **Table 3**, three linear approximations with the same bias 2^{-23} are found:

$$\begin{aligned} z_1 = 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} + k_{39} + k_{54} + k_{57} + \\ k_{63} + k_{67} + k_{68} + k_{69} + k_{72} + iv_3 + iv_6 + iv_{21} + iv_{24} + iv_{30} + iv_{33} + \\ iv_{39} + iv_{45} + iv_{49} + iv_{50} + iv_{51} + iv_{70} + iv_{71} + iv_{72} + iv_{76} + iv_{77} + iv_{78} \end{aligned}$$

$$\begin{aligned} z_1 = 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} + k_{39} + k_{54} + k_{57} + \\ k_{63} + k_{68} + k_{69} + k_{72} + iv_3 + iv_6 + iv_{21} + iv_{24} + iv_{30} + iv_{33} + iv_{39} + \\ iv_{45} + iv_{49} + iv_{50} + iv_{51} + iv_{70} + iv_{71} + iv_{72} + iv_{76} + iv_{77} + iv_{78} \end{aligned}$$

$$z_1 = 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} + k_{39} + k_{54} + k_{57} + k_{63} + k_{67} + k_{69} + k_{72} + iv_3 + iv_6 + iv_{21} + iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{49} + iv_{50} + iv_{51} + iv_{70} + iv_{71} + iv_{72} + iv_{76} + iv_{77} + iv_{78}$$

Similarly, when (k_{58}, k_{59}, k_{64}, k_{65}) takes the other 15 possible values, linear approximations can also be found as the same method. The corresponding results are concluded in **Table 4**.

Table 4. Different $(k_{58}, k_{59}, k_{64}, k_{65})$ and Corresponding (Ω_K, Ω_{IV})

$(k_{58},k_{59},k_{64},k_{65})$	$(\Omega_{IV}, \lvert\Omega_{IV}\rvert = 10)$	$(\Omega_K, \lvert\Omega_K\rvert = 10)$		Bias
(0, 0, 0, 0)(0, 1, 0, 0) (1, 0, 0, 0)(1, 1, 0, 0)	iv_{10} iv_{13} iv_{25} iv_{31} iv_{34} iv_{37} iv_{40} iv_{50} iv_{54} iv_{55}	k_{13} k_{19} k_{23} k_{38} k_{39} k_{40} k_{45} k_{46} k_{50}	k_5	2^{-23}
			k_{67}	2^{-23}
			k_{68}	2^{-23}
(0, 0, 1, 1)(0, 1, 1, 1) (1, 0, 1, 1)(1, 1, 1, 1)	iv_{10} iv_{13} iv_{25} iv_{31} iv_{34} iv_{37} iv_{40} iv_{50} iv_{54} iv_{55}	k_{13} k_{19} k_{23} k_{38} k_{40} k_{45} k_{46} k_{50} k_{63}	k_5	2^{-23}
			k_{67}	2^{-23}
			k_{68}	2^{-23}
(0, 0, 0, 1)(0, 1, 0, 1) (1, 0, 0, 1)(1, 1, 0, 1)	iv_{10} iv_{13} iv_{25} iv_{31} iv_{34} iv_{37} iv_{40} iv_{50} iv_{54} iv_{55}	k_{13} k_{19} k_{38} k_{39} k_{40} k_{45} k_{46} k_{50} k_{62}	k_5	2^{-23}
			k_{67}	2^{-23}
			k_{68}	2^{-23}
(0, 0, 1, 0)(0, 1, 1, 0) (1, 0, 1, 0)(1, 1, 1, 0)	iv_{10} iv_{13} iv_{25} iv_{31} iv_{34} iv_{37} iv_{40} iv_{50} iv_{54} iv_{55}	k_{13} k_{19} k_{38} k_{39} k_{40} k_{45} k_{46} k_{50} k_{63}	k_5	2^{-24}
			k_{67}	2^{-24}
			k_{68}	2^{-24}

From **Table 4**, we observe that three linear approximations with the same bias can be found for each (k_{58}, k_{59}, k_{64}, k_{65}), whereas 2^{-23} and 2^{-24} are the biggest and the smallest bias of all the linear approximations found. Note that for the sake of comparison with the previous results, the average bias $2^{-23.2}$ is used to describe our results due to the "undetermined" (k_{58}, k_{59}, k_{64}, k_{65}).

4.4 Comparison with Previous Results

For the reduced version of 288-round Trivium, Turan et al. in [Turan, 07] found a linear approximation with bias of 2^{-31} when $|\Omega_K| = 10$ and $|\Omega_{IV}| = 10$. In [Jia, 11], Jia et al. presented a multiple linear attack on 288-round Trivium, resulting in another linear approximation with the same bias 2^{-31} when $|\Omega_K| = 10$ and $|\Omega_{IV}| = 13$. In our [Sun, 12], Algorithm 2 was applied to look for linear approximations. As a result, three linear approximations with the same bias 2^{-25} were found when $|\Omega_K| = 10$ and $|\Omega_{IV}| = 10$, and three ones with the same bias 2^{-22} were found when $|\Omega_K| = 10$ and $|\Omega_{IV}| = 13$. While in this paper, we find linear approximations with bigger bias. What's more, the additional four key bits are recovered. These results are summarized in **Table 5**. Obviously, our attack is better than the others.

Table 5. Comparison with Previous Results

	$(|\Omega_K|, |\Omega_{IV}|)$	Bias	Number of Approximations	Data Complexity	Success Rate /%
Turan et al. [Turan, 07]	(10,10)	2^{-31}	1	2^{62}	97.7
Sun et al. [Sun, 12]	(10,10)	2^{-25}	3	2^{50}	97.7
This Paper	(10,10)	$2^{-23.2}$	3	$2^{47.2}$	97.8
Jia et al. [Jia, 11]	(10,13)	2^{-31}	2	2^{61}	97.7
Sun et al. [Sun, 12]	(10,13)	2^{-22}	3	2^{44}	97.7
This Paper	(10,13)	$2^{-20.2}$	3	$2^{41.2}$	97.8

5. Conclusions

In cryptography, linear cryptanalysis is the most basic cryptanalysis approach aiming to find the linear relation between outputs and inputs. A variety of refinements to the attack have been suggested in the past. In this paper, a novel technique called cube-linear attack is first proposed to deal with a probabilistic polynomial and furthermore to mine the available information unexposed by the previous linear cryptanalysis. As a new contribution to linear cryptanalysis, it is beneficial to allow for a reduction in the amount of data required for a successful attack in specific circumstances. Applying our method to a specific analysis of Trivium, we get better linear cryptanalysis results so far. Although a few better cryptanalytic results on Trivium had been published several years earlier using other attacks, however, we believe that our methods are meaningful from the point of view of improving linear cryptanalysis and can be extended to other ciphers. In other words, it is worth considering for our method in launching linear attack on a cryptographic primitive.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 61202491).

References

[Golić, 94] J. Dj. Golić. Linear Cryptanalysis of Stream Ciphers. Fast Software Encryption, pp. 154-169, December 14-16, 1994.
[Golić, 02] J. Dj. Golić, Bagini. V., Morgari, G. Linear Cryptanalysis of Bluetooth Stream Cipher. EUROCRYPT, pp. 238-255, April 28 –May 2, 2002.
[Muller, 05] Muller, F., Peyrin, T. Linear Cryptanalysis of TSC Family of Stream Ciphers. ASIACRYPT, pp. 373-394, December 4-8, 2005.
[Shahram, 05] Shahram Khazaei, Mehdi Hassanzadeh. Linear Sequential Circuit Approximation of the Trivium Stream Ciphers. eSTREAM, ECRYPT Stream Cipher Project Report 2005/063 http://www.ecrypt.eu.org/stream/. 2005
[Turan, 07] Turan M S and Kara O. Linear Approximations for 2-round Trivium. Workshop on The State of the Art of Stream Cipher (SASC2007), pp. 22-31, January 31 - February 1, 2007.
[Matsui, 92] M. Matsui, A. Yamagishi. A New Method for Known Plaintext Attack of FEAL Cipher. EUROCRYPT, pp. 81-91, May 24-28, 1992.
[Shimizu, 87] A. Shimizu and S. Miyaguchi. Fast Data Encipherment Algorithm FEAL. EUROCRYPT, pp. 267-280, April 13-15, 1987.
[Matsui, 93] M. Matsui, Linear Cryptanalysis Method for DES Cipher. EUROCRYPT, pp. 386-397, May 23-27, 1993.
[Matsui, 94] M. Matsui. The First Experimental Cryptanalysis of the Data Encryption Standard. CRYPTO, pp. 1-11, August 21-25, 1994.
[Kaliski, 94] Kaliski, B.S., Robshaw, M.J.B. Linear Cryptanalysis using Multiple Approximations. CRYPTO, pp. 26-39, August 21-25, 1994.
[Langford, 94] S. Langford, M. Hellman. Differential-Linear Cryptanalysis. CRYPTO 1994, pp. 17-26, August 21-25, 1994.
[Knudsen, 96] Knudsen, L.R., Robshaw. M.J.B. Non-linear Approximations in Linear Cryptanalysis. EUROCRYPT. pp. 224-236, May 12-16, 1996.
[Bogdanov, 11] Bogdanov, A., Rijmen, V. Linear Hulls with Correlation Zero and Linear Cryptanalysis of Block Ciphers. Cryptology ePrint Archive: Report 2011/123. 2011
[Dinur, 09] Itai Dinur and Adi Shamir. Cube Attacks on Tweakable Black Box Polynomials. EUROCRYPT. pp. 278-299, April 26-30, 2009.
[Vielhaber, 07] Vielhaber, M. Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack. Cryptology ePrint Archive Report 2007/413. 2007.
[Lai, 94] Lai, X. Higher Order Derivatives and Differential Cryptanalysis. Communications and Cryptography. Vol.276, pp. 227-233,1994.
[Cannière, 05] Ch. De Cannière and B. Preneel. Trivium. TRIVIUM-Specifications. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/030 (2005) http://www.ecrypt.eu.org/stream.
[eSTREAM, 04] eSTREAM, The ECRYPT Stream Cipher Project. IST-2002-507932[EB/OL], Available at

http://www.ecrypt.eu.org/stream, 2004.
[Sun, 12] Sun Wen-long, Guan Jie, Liu Jiandong. Linear Cryptanalysis of Simplified Trivium. Chinese Journal of Computers. vol.35, No.9, pp. 1890-1896, 2012. (in Chinese)
[Jia, 11] Jia Yan-yan, Hu Yu-pu, Yang Wen-feng, Gao Jun-tao. Linear Cryptanalysis of 2-round Trivium with Multiple Approximations. Journal of Electronics & Information Technology, Vol.33, No.1, pp. 223-227, 2011. (in Chinese)
[Maximov, 07] Maximov A, Biryukov A. Two Trivial Attacks on Trivium. Workshop on The State of the Art of Stream Ciphers (SASC2007), pp. 1-16, January 31 - February 1, 2007.
[Raddum, 06] Raddum H. Cryptanalytic Results on Trivium. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/039, 2006.

Appendix

A. The equation (5) and equation (9) appearing in this paper are as follows.

Equation (5) is

$$\begin{aligned} z_1 = {} & 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{39} + k_{54} + k_{57} + k_{67} + k_{68} + k_{69} + k_{72} + iv_3 + iv_6 + iv_{21} + \\ & iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{51} + iv_{72} + iv_{78} + k_4 \cdot k_5 + k_{13} \cdot k_{14} + k_{13} \cdot k_{41} + k_{14} \cdot k_{40} + k_{16} \cdot \\ & k_{17} + k_{19} \cdot k_{20} + k_{19} \cdot k_{47} + k_{22} \cdot k_{23} + k_{20} \cdot k_{46} + k_{25} \cdot k_{26} + k_{28} \cdot k_{29} + k_{34} \cdot k_{35} + k_{37} \cdot k_{38} + k_{37} \cdot \\ & k_{65} + k_{38} \cdot k_{64} + k_{39} \cdot k_{40} + k_{43} \cdot k_{44} + k_{45} \cdot k_{46} + k_{49} \cdot k_{50} + k_{52} \cdot k_{53} + k_{58} \cdot k_{59} + k_{61} \cdot k_{62} + k_{63} \cdot \\ & k_{64} + iv_{77} \cdot k_{64} + k_{64} \cdot k_{65} + k_{67} \cdot k_{68} + k_{70} \cdot k_{71} + iv_{10} \cdot iv_{11} + iv_{13} \cdot iv_{14} + iv_{25} \cdot iv_{26} + iv_{31} \cdot iv_{32} + \\ & iv_{34} \cdot iv_{35} + iv_{37} \cdot iv_{38} + iv_{40} \cdot iv_{56} + iv_{41} \cdot iv_{55} + iv_{49} \cdot iv_{50} + iv_{54} \cdot iv_{55} + iv_{58} \cdot iv_{59} + iv_{61} \cdot iv_{62} + \\ & iv_{67} \cdot iv_{68} + iv_{70} \cdot iv_{71} + iv_{76} \cdot k_{65} + iv_{73} \cdot iv_{74} + k_{13} \cdot k_{39} \cdot k_{40} + k_{14} \cdot k_{38} \cdot k_{39} + k_{19} \cdot k_{45} \cdot k_{46} + k_{20} \cdot \\ & k_{44} \cdot k_{45} + k_{37} \cdot k_{63} \cdot k_{64} + iv_{70} \cdot k_{59} + k_{38} \cdot k_{39} \cdot k_{40} + k_{38} \cdot k_{39} \cdot k_{41} + k_{38} \cdot k_{62} \cdot k_{63} + k_{44} \cdot k_{45} \cdot k_{46} + \\ & k_{44} \cdot k_{45} \cdot k_{47} + k_{62} \cdot k_{63} \cdot k_{64} + iv_{71} \cdot k_{58} + k_{62} \cdot k_{63} \cdot k_{65} + iv_{40} \cdot iv_{54} \cdot iv_{55} + iv_{41} \cdot iv_{53} \cdot iv_{54} + iv_{53} \cdot \\ & iv_{54} \cdot iv_{55} + iv_{53} \cdot iv_{54} \cdot iv_{56} + iv_{25} \cdot k_{14} + iv_{25} \cdot k_{41} + iv_{25} \cdot k_{39} \cdot k_{40} + iv_{26} \cdot k_{13} + iv_{26} \cdot k_{40} + iv_{26} \cdot k_{38} \cdot \\ & k_{39} + iv_{31} \cdot k_{20} + iv_{31} \cdot k_{47} + iv_{31} \cdot k_{45} \cdot k_{46} + iv_{32} \cdot k_{46} + iv_{32} \cdot k_{19} + iv_{32} \cdot k_{44} \cdot k_{45} + iv_{49} \cdot k_{38} + iv_{49} \cdot \\ & k_{65} + iv_{49} \cdot k_{63} \cdot k_{64} + iv_{50} \cdot k_{37} + iv_{50} \cdot k_{64} + iv_{50} \cdot k_{62} \cdot k_{63} \end{aligned} \tag{5}$$

Equation (9) is

$$\begin{aligned} z_1 = {} & 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} + k_{39} + k_{54} + k_{57} + k_{63} + k_{67} + k_{68} + k_{69} + k_{72} + \\ & iv_3 + iv_6 + iv_{21} + iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{49} + iv_{50} + iv_{51} + iv_{70} + iv_{71} + iv_{72} + iv_{76} + \\ & iv_{77} + iv_{78} + k_4 \cdot k_5 + k_{13} \cdot k_{14} + k_{13} \cdot k_{41} + k_{14} \cdot k_{40} + k_{16} \cdot k_{17} + k_{19} \cdot k_{20} + k_{19} \cdot k_{47} + k_{22} \cdot k_{23} + \\ & k_{20} \cdot k_{46} + k_{25} \cdot k_{26} + k_{28} \cdot k_{29} + k_{34} \cdot k_{35} + k_{37} \cdot k_{38} + k_{37} \cdot k_{63} + k_{39} \cdot k_{40} + k_{43} \cdot k_{44} + k_{45} \cdot k_{46} + \\ & k_{49} \cdot k_{50} + k_{52} \cdot k_{53} + k_{61} \cdot k_{62} + k_{67} \cdot k_{68} + k_{70} \cdot k_{71} + iv_{10} \cdot iv_{11} + k_{70} \cdot k_{71} + iv_{10} \cdot iv_{11} + iv_{13} \cdot iv_{14} + \\ & iv_{25} \cdot iv_{26} + iv_{31} \cdot iv_{32} + iv_{34} \cdot iv_{35} + iv_{37} \cdot iv_{38} + iv_{40} \cdot iv_{56} + iv_{41} \cdot iv_{55} + iv_{49} \cdot iv_{50} + iv_{54} \cdot iv_{55} + \\ & iv_{58} \cdot iv_{59} + iv_{61} \cdot iv_{62} + iv_{67} \cdot iv_{68} + iv_{70} \cdot iv_{71} + iv_{73} \cdot iv_{74} + k_{13} \cdot k_{39} \cdot k_{40} + k_{14} \cdot k_{38} \cdot k_{39} + k_{19} \cdot \\ & k_{45} \cdot k_{46} + k_{20} \cdot k_{44} \cdot k_{45} + k_{38} \cdot k_{39} \cdot k_{40} + k_{38} \cdot k_{39} \cdot k_{41} + k_{38} \cdot k_{62} \cdot k_{63} + k_{44} \cdot k_{45} \cdot k_{46} + k_{44} \cdot k_{45} \cdot \\ & k_{47} + iv_{40} \cdot iv_{54} \cdot iv_{55} + iv_{41} \cdot iv_{53} \cdot iv_{54} + iv_{53} \cdot iv_{54} \cdot iv_{55} + iv_{53} \cdot iv_{54} \cdot iv_{56} + iv_{25} \cdot k_{14} + iv_{25} \cdot k_{41} + \\ & iv_{25} \cdot k_{39} \cdot k_{40} + iv_{26} \cdot k_{13} + iv_{26} \cdot k_{40} + iv_{26} \cdot k_{38} \cdot k_{39} + iv_{31} \cdot k_{20} + iv_{31} \cdot k_{47} + iv_{31} \cdot k_{45} \cdot k_{46} + iv_{32} \cdot \\ & k_{46} + iv_{32} \cdot k_{19} + iv_{32} \cdot k_{44} \cdot k_{45} + iv_{49} \cdot k_{38} + iv_{49} \cdot k_{63} + iv_{50} \cdot k_{37} + iv_{50} \cdot k_{62} \cdot k_{63} \end{aligned} \tag{9}$$

B.

Algorithm 2 [Sun, 12]：Algorithm to Search Best Linear Approximations

Reset Ω_K and Ω_{IV};

Input : n_K and n_{IV} (sizes of Ω_K and Ω_{IV} to choose respectively)

Step1: Reset Ω and Π, count the frequency of each bit in all non-linear monomials, put the bits with the highest frequency in Ω;

Step2:

1.When $|\Omega| = 1$, determine the type of the bit in Ω, if it is key bit, then put it to Ω_K, otherwise Ω_{IV};

2.When $|\Omega| \geq 2$,count the frequency of quadratic term of each bit in Ω, put the bits with the highest frequency in Π:

2.2.1 If $|\Pi| = 1$, determine type of the bit in Π, if it is key bit, then put it to Ω_K, otherwise Ω_{IV};

2.2.2 If $|\Pi| \geq 2$, choose arbitrarily one bit in Π, and determine type, if it is key bit, put it in Ω_K, otherwise Ω_{IV};

Step3: Calculate the polynomial again, based on the chosen Ω_K and Ω_{IV};

Step4:

if $\left((|\Omega_K| < n_K) \&\&(|\Omega_{IV}| < n_{IV})\right)$

Return to Step1;

else if $\left((|\Omega_K| = n_K) \&\&(|\Omega_{IV}| \neq n_{IV})\right)$

Stop searching the key bits, return to Step1, and continue to search the IV bits;

else if $\left((|\Omega_K| \neq n_K) \&\&(|\Omega_{IV}| = n_{IV})\right)$

Stop searching the IV bits, return to Step1, and continue to search the key bits;

else if $\left((|\Omega_K| = n_K) \&\&(|\Omega_{IV}| = n_{IV})\right)$

Output: $\left((\Omega_K, \Omega_{IV}), \varepsilon\right)$

End