Novel Technique in Linear Cryptanalysis

Wen-Long Sun and Jie Guan

In this paper, we focus on a novel technique called the cube–linear attack, which is formed by combining cube attacks with linear attacks. It is designed to recover the secret information in a probabilistic polynomial and can reduce the data complexity required for a successful attack in specific circumstances. In addition to the different combination strategies of the two attacks, two cube–linear schemes are discussed. Applying our method of a cube–linear attack to a reduced-round Trivium, as an example, we get better linear cryptanalysis results. More importantly, we believe that the improved linear cryptanalysis technique introduced in this paper can be extended to other ciphers.

Keywords: Cryptanalysis, linear cryptanalysis, cube–linear attack, Trivium, stream cipher.

I. Introduction

Linear cryptanalysis is an effective known-plaintext attack (KPA) against block ciphers. At present, KPA has been adapted to stream ciphers [1]–[5]. M. Matsui and A. Yamagishi [6], in 1992, introduced the idea of linear cryptanalysis in an attack on FEAL [7]. The techniques used in this attack were refined by M. Matsui and had a dramatic effect on the Data Encryption Standard (DES). This eventually led to the first experimental cryptanalysis of the cipher being reported in the open community [8]–[9].

Subsequently, several refinements to the basic idea of linear cryptanalysis have been suggested to improve the efficiency of the attacks it envelops, either in specific circumstances or in all cases. In 1994, B.S. Kaliski and M.J.B. Robshaw [10] proposed an extension to linear cryptanalysis based on the use of multiple linear approximations. S.K. Langford and M.E. Hellman [11], in 1994, introduced the differential–linear attack, which is a mix of both differential cryptanalysis and linear cryptanalysis. In 1996, L.R. Kundsen and M.J.B. Robshaw [12] introduced the idea of extending M. Matsui's linear cryptanalytic techniques to more general cases in which non-linear relations are also considered. In [13], zero-correlation linear cryptanalysis, the counterpart of impossible differential cryptanalysis in the domain of linear cryptanalysis, was proposed by A. Bogdanov and V. Rijmen, resulting in a faster attack for some ciphers.

1. Motivation and Contribution

As introduced above, there have been several extensions to linear cryptanalysis at present. Nevertheless, is there any available information not yet exploited by previous linear cryptanalysis methods?

In this paper, we answer this question positively. Generally speaking, linear cryptanalysis exploits specific correlations

Manuscript received Nov. 26, 2013; revised Sept. 13, 2014; accepted Nov. 4, 2014.

This work was supported by the National Natural Science Foundation of China (Grant No. 61202491, 61272041, and 61272488).

Wen-Long Sun (corresponding author, swl_cipher@163.com) is with the Information Science and Technology Institute, Zhengzhou, Henan Province and also with Beijing Satellite Navigation Center, Beijing, China.

Jie Guan (guanjie007@163.com) is with the Information Science and Technology Institute, Zhengzhou, China.

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) in both the public variables and the secret variables. As we know, a very powerful tool to recover the secret variables is cube attacks, proposed by [14]. To derive the secret information, the attacker sums this bit over all possible values of a subset of the public variables. The summations are used to derive linear equations in the secret variables, which can be efficiently solved. When launching linear attacks, we can, in specific scenarios, obtain the explicit description of a multivariate polynomial. Since the polynomial's probability is less than one after one or several approximations in linear attacks, we have to adapt our use of cube attacks in such cases because they are used only on such polynomials when the polynomial's probability is equal to one.

Here, we combine cube attacks with linear attacks to propose a novel method called cube–linear attack, aiming to recover the secret variables of a probabilistic multivariate polynomial. It uses information that both cube attacks and linear attacks use. Once the secret variables have been recovered, the degree of non-linear monomials involving the recovered bits would decrease and may even fall to zero. This is the same result as desired by linear attacks. Subsequent cryptanalysis results prove our method to be efficient.

In addition to the two combination scenarios (using linear attacks during a cube attack and using cube attacks during a linear attack), two schemes, A and B, are discussed. The data complexity and success rate of each scheme can be calculated only when all the derived polynomials [14] defined by any determined cube index are independent. Otherwise, the situations are too complex to give any concrete computational formula using the existing theories. An introduction to the detailed theory of our method is given in Section III. Afterwards, an improved linear cryptanalysis is proposed using the cube–linear attack as a novel technique. 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 1,152). The data complexity is $2^{47.2}$ IVs with 97.8% success rate, improving upon previous linear cryptanalysis results. Although a few better cryptanalysis results on Trivium have been published using other attacks ([14], [15], and [16]), it's confirmed that our method is meaningful from the point of view of improving linear cryptanalysis.

For convenience, we particularly follow the relevant concepts and terminology of cube attacks. Although based on the same essence of higher-order differential cryptanalysis, there are some differences between cube attacks and our cube–linear attack. Firstly, cube attacks are applied to such a polynomial having probability one, while our method copes with probabilistic polynomials. Secondly, the primary cost of cube attacks lies in the searching of the appropriate cube indexes, while our method focuses on an explicit polynomial. Moreover, they are two different methods, and our cube–linear attack is proposed as an improved technique for linear cryptanalysis.

2. Organization

This paper is organized as follows. In Section II, we briefly review cube attacks and linear cryptanalysis. Afterwards, we describe the theory of our cube–linear attack and propose an improved linear cryptanalysis (Section III). In Section IV, we apply our method to a specific analysis on Trivium. Finally, we make a few concluding remarks in Section V.

II. Cube Attacks and Linear Cryptanalysis

1. Review of Cube Attacks

Cube attacks [14], first formalized by I. Dinur and A. Shamir at EUROCRYPT 2009, are a generic type of algebraic attack and can be applied to 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 almost any cryptographic scheme, each output bit, z_i, can be described by a multivariate master polynomial over GF(2) comprising public variables $v_1, v_2, \ldots, v_m$, which are either bits of the plaintext of a block cipher or bits of the initial vector of a stream cipher and that are dependent upon secret variables $x_1, x_2, \ldots, x_n$ — $z_i = p(v_1, \ldots, v_m, x_1, \ldots, x_n)$.

To simplify our notation, the distinction between public and private variables is now ignored. 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 rewritten as the sum of terms that 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 variables in common with t_I, and each term in $q(x_1, \ldots, x_n)$ does not contain at least one variable from I.

Any subset I of size s defines an 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 $p_I \overset{\Delta}{=} \sum p_{|v}$. A maxterm of p is a term t_I such that the superpoly $P_{S(I)}$ of I in p is a linear polynomial that is not a constant.

Theorem 1 [14]. For any polynomial p and subset of variables I, we have $p_I \equiv P_{S(I)} \mod 2$.

Cube attacks have two phases. In the first preprocessing phase, the task is to find as many maxterms and corresponding linear superpolys as possible. In the next phase, the online phase, the attacker solves the system of linear equations obtained and acquires some values related to the secret variables.

2. Review of Linear Cryptanalysis

The basic idea behind linear cryptanalysis is to find some linear approximation to the action of cryptographic primitives. In other words, what the attacker exploits are some statistical correlations between the input and the output. For a cryptosystem 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 task of the attacker is to find the index sets I, J, and L such that

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

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

For the block and stream ciphers, linear attacks are usually executed as follows. First, we look for the linear or nonlinear approximations of different rounds, and then we combine them. From this, we can obtain final linear approximations for the whole cryptosystem with probability calculated according to Lemma 1 (piling-up Lemma).

Lemma 1 [8]. For each value i, $1 \le i \le n$, let X_i be a random variable, independent of X_j for all $j \neq i$, such that $P(X_i = 0) = p_i$, $P(X_i = 1) = 1 - p_i$. Then

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

Given a 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 [8]. The core idea of Algorithm 1 is a maximum-likelihood.

Algorithm 1. 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

The computational formula of the success rate to recover the key $\sum_{i \in I} k_i$ is as follows [8] and is related to both the data complexity N (the number of plaintext/ciphertext pairs) and the bias ε :

$$\gamma = \frac{1}{\sqrt{2\pi}} \int_{-2\sqrt{N}\varepsilon}^{\infty} e^{-x^2/2} \, dx. \tag{3}$$

The main goal of linear cryptanalysis is to find an effective linear approximation. However, thus far, there has been no optimal algorithm to look for such a linear approximation for any cryptosystem. In this paper, we tentatively put forward a novel technique called cube–linear attack, contributing to linear cryptanalysis.

III. Improved Linear Cryptanalysis

1. Cube–Linear Attack

This subsection provides the basic theory of our cube–linear attack. Generally speaking, linear cryptanalysis exploits specific correlations between the input and output of cryptographic primitives. For almost any cryptographic scheme, each output bit can be described by a multivariate master polynomial over GF(2) in the public variables and the secret variables. When launching linear attacks, we can, in specific scenarios, obtain the explicit description of a multivariate polynomial $z_i = p(v, k)$ with probability p^*. Based on cube attacks, the polynomial $z_i = p(v, k)$ 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. We can determine the maxterm t_I leading to a linear expression $P_{S(I)}$, and then the secret variables in the resultant system of polynomial equations can be efficiently solved. Since the polynomial's probability, p^*, is less than one after one or several approximations in linear attacks and cube attacks are only used on such polynomials when their probability is equal to one, the actual use of cube attacks in this case means that they first need to be adapted prior to their use. Here, we combine cube attacks with linear attacks to propose a novel method called a cube–linear attack, aiming to recover the secret variables of a probabilistic multivariate polynomial. Based on the aforementioned different ways of combining cube attacks and linear attacks, two schemes, A and B, are discussed under the following attack condition: all derived polynomials defined by any determined maxterm are

independent.

A. Description of Scheme A

Scheme A can be considered as the "cube–linear–cube" attack, the name befittingly corresponding to the three steps that it incorporates. The following analysis embodies a more detailed introduction of scheme A. Based on the split polynomial $p(v,k) \equiv t_I \cdot P_{S(I)} + q(v,k)$, as mentioned above, we can easily determine the maxterm leading to an expression $P_{S(I)}$ of which the degree $d(P_{S(I)})$ is one. Correspondingly, I and C_I are also 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^* stands for the probability of $p_{|v}$ when C_I takes the value v on condition that the polynomial $z_i = p(v,k)$ holds with probability p^*). Let us denote $K_{|v}$ the XOR of all the monomials involving only the key information in the derived polynomial $p_{|v}$. Then $K_{|v}$ can be determined by Algorithm 1 since the derived polynomial $p_{|v}$ is a linear expression when considering $K_{|v}$ as a single variable. Similar to cube attacks, we can obtain a new linear equation when summing all the recovered $K_{|v}$.

Scheme A

Step 1. Determine the maxterm t_I.
Step 2. Recover $K_{|v}$ in each derived polynomial $p_{|v}$ by Algorithm 1.
Step 3. Sum all the recovered $K_{|v}$, and then the value $\sum_{v \in C_I} K_{|v}$ is known.

Theorem 2. The 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 s is the size of index subset I; N_v and γ_v are respectively referred to as the data complexity and the success rate to recover $K_{|v}$ using Algorithm 1.

Proof. Since the probability of the 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. Consequently, the total data complexity of scheme A is $N_A = \sum_{v=0}^{2^s-1} N_v = O\left(\sum_{v=0}^{2^s-1} \left(\left|p_v^* - 1/2\right|\right)^{-2}\right)$.

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" as the "right" and "wrong" values of $K_{|v}$, respectively; that is, $p\left(K_{|v} = 0\right) = \gamma_v$ and $p\left(K_{|v} = 1\right) = \gamma_v^*$, respectively. Based on the aforementioned attack condition, we deduce that all the recovered $K_{|v}$ are independent of each other. Therefore, 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 Lemma 1. ■

B. Description of Scheme B

Scheme B

Step 1. Determine the maxterm t_I.
Step 2. Sum all the derived polynomials $p_{|v}$, then we have $P_{S(I)} \overset{\Delta}{=} \sum_{v \in C_I} p_{|v}$.
Step 3. Recover the key information in $P_{S(I)}$ by Algorithm 1.

Corresponding to the above three steps, the first two steps of scheme B are actually the use of the cube attack, and then the key information in $P_{S(I)}$ can be determined with the help of Algorithm 1.

Theorem 3. 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 aforementioned attack condition, all the derived polynomials $p_{|v}$ defined by $v \in C_I$ hold independently 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 Lemma 1. By Algorithm 1, the data complexity N_B to recover the key information in $P_{S(I)}$ with a confident success rate γ_B is

$$O\left(\left(2^{2^s-1} \prod_{v=0}^{2^s-1} \left|p_v^* - 1/2\right|\right)^{-2}\right).$$ ■

C. Comments on Schemes A and B

It should be noted that the recovered key information using scheme A is the same as that of scheme B.

Theorem 4. When the probability of each derived polynomial p_v^* satisfies $\left|p_v^* - 1/2\right| \le 2^{-1.5}$, $N_A \le \frac{2^s}{2^{s-1}} N_B$ (equality holds iff $\left|p_v^* - 1/2\right| = 2^{-1.5}$), where s is the size of index subset I; and N_A and N_B represent respectively the data complexity of scheme A and scheme B.

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 $x_v = \left(2 \cdot \left|p_v^* - 1/2\right|\right)^{-2}, x_v \in (1, +\infty)$, then $N_A = 4 \sum_{v=0}^{2^s-1} x_v$

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, then it's deduced that $x_v = 2^{s/(2^s-1)}$.

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

$$\frac{\sum_{v=0}^{2^s-1} x_v}{\prod_{v=0}^{2^s-1} x_v} = \frac{x_0 + x_1 + \cdots + x_{2^s-1}}{x_0 x_1 \cdots x_{2^s-1}}$$
$$= \frac{1}{x_1 x_2 \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{i-1} x_{i+1} \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{2^s-2}}$$
$$\leq \frac{1}{2^s} + \cdots + \frac{1}{2^s} + \cdots + \frac{1}{2^s} = \frac{1}{2^s} \cdot 2^s = 1.$$

Specifically, when $x_v \geq 2$; that is, $\left|p_v^* - 1/2\right| \leq 2^{-1.5}$, we have

$$\frac{\sum_{v=0}^{2^s-1} x_v}{\prod_{v=0}^{2^s-1} x_v} = \frac{x_0 + x_1 + \cdots + x_{2^s-1}}{x_0 x_1 \cdots x_{2^s-1}}$$
$$= \frac{1}{x_1 x_2 \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{i-1} x_{i+1} \cdots x_{2^s-1}} + \cdots + \frac{1}{x_0 x_1 \cdots x_{2^s-2}}$$
$$\leq \frac{1}{2^{2^s-1}} + \cdots + \frac{1}{2^{2^s-1}} + \cdots + \frac{1}{2^{2^s-1}} = \frac{2^s}{2^{2^s-1}}.$$

That is, $N_A \leq \frac{2^s}{2^{2^s-1}} N_B$. ■

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

Table 1 provides some comparisons between schemes A and B at the same level of success rate. It shows that scheme A is in fact more superior in most scenarios because of the following reasons: First of all, since the probability of the derived polynomial p_v^* in most cases satisfies $\left|p_v^* - 1/2\right| << 2^{-1.5}$, it means that $N_A << N_B$ according to our calculations. Secondly, the success rate r_A doesn't rapidly drop due to $\gamma_A = 1/2 + 2^{2^s-1} \prod_{v=0}^{2^s-1} (\gamma_v - 1/2)$. Furthermore, scheme A could actually be enhanced by increasing a few plaintext/ciphertext pairs based on the theory of linear cryptanalysis. Taken together, when to choose scheme A or scheme B should depend on the particular situation.

Table 1. Comparison between schemes A and B.

s \ ε	$2^{-1.3}$		2^{-6}		2^{-9}		Success rate
	N_A	N_B	N_A	N_B	N_A	N_B	
1	2^4	$2^{3.2}$	$2^{13.4}$	2^{22}	$2^{19.4}$	2^{34}	97.8%
2	$2^{5.3}$	$2^{4.4}$	$2^{14.7}$	2^{42}	$2^{20.7}$	2^{66}	
3	$2^{6.5}$	$2^{6.8}$	$2^{15.9}$	2^{82}	$2^{21.9}$	2^{130}	

Remark 1. For both schemes, the first thing we need to do in practice after determining the maxterm t_I is to verify whether the attack condition 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 for the data complexity and success rate of the two schemes using the present theory.

2. Improved Linear Cryptanalysis

For a probabilistic polynomial, it is the secret variables recovered by our cube–linear attack that have never before been exposed by previous linear cryptanalysis. Consequently, it's confirmed that the cube–linear attack indeed provides us with a paradise of improvement in linear cryptanalysis. Overall, the improved linear cryptanalysis could be summarized as follows.

Improved linear cryptanalysis.

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

Step 1. Verify whether the attack condition comes into existence or not, if it is tenable, then go to the next step; Otherwise, our theory won't work.

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

Step 3. Combining the recovered key information, carry on looking for its linear approximation.

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

IV. Improved Linear Cryptanalysis on Trivium

1. Trivium Stream Cipher

Trivium [17], a hardware-oriented stream cipher, was

designed by C. De Cannière and B. Preneel and was selected for the final eSTREAM portfolio [18]. It takes an 80-bit key K and an 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. 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:

$$(s_1, s_2, \cdots, s_{93}) \leftarrow (k_1, \cdots, k_{80}, 0, \cdots, 0),$$
$$(s_{94}, \cdots, s_{177}) \leftarrow (\mathrm{iv}_1, \cdots, \mathrm{iv}_{80}, 0, \cdots, 0),$$
$$(s_{178}, \cdots, s_{288}) \leftarrow (0, \cdots, 0, 1, 1, 1).$$

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

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

No output is produced during the first 1,152 rounds. After this initialization phase, the value of z is output as the key stream at each round.

2. Description of Attack Process

For the 288-round Trivium, we have

$$\begin{aligned} z_1 &= s_{288}(66) + s_{288}(93) + s_{288}(162) \\ &\quad + s_{288}(177) + s_{288}(243) + s_{288}(288), \end{aligned} \tag{4}$$

where $s_t(i)$ is the ith internal state bit at time t. The output z_1 is the sum of bits $s_{288}(66)$, $s_{288}(93)$, $s_{288}(162)$, $s_{288}(177)$, $s_{288}(243)$, and $s_{288}(288)$. The ANF of z_1 is found exhaustively in terms of the internal state bit at $t = 144$ as [5]

$$\begin{aligned} z_1 &= s_{144}(6) + s_{144}(16) \cdot s_{144}(17) + s_{144}(31) \cdot s_{144}(32) \\ &\quad + s_{144}(33) + s_{144}(57) + s_{144}(82) \cdot s_{144}(83) + s_{144}(84) \\ &\quad + s_{144}(96) + s_{144}(97) \cdot s_{144}(98) + s_{144}(99) + s_{144}(111) \\ &\quad + s_{144}(129) + s_{144}(142) \cdot s_{144}(143) + s_{144}(144) + s_{144}(150) \\ &\quad + s_{144}(162) + s_{144}(163) \cdot s_{144}(164) + s_{144}(165) + s_{144}(186) \\ &\quad + s_{144}(192) + s_{144}(208) \cdot s_{144}(209) + s_{144}(210) + s_{144}(231) \\ &\quad + s_{144}(235) \cdot s_{144}(236) + s_{144}(237) + s_{144}(252), \end{aligned} \tag{5}$$

and its closest linear approximation [5] is

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

with bias 2^{-9}.

Since the purpose of the attack is to find a linear approximation in the key, IV and output bits, the linear approximation given above is to be rewritten in terms of $s_0(i)$, $i = 1, 2, \ldots, 80, 94, 95, \ldots, 173$. The remaining terms are omitted since they are assigned to constants during the initialization phrase. Now, we have the right equation in (7) (see Appendix A), whereas it should be noted that there is something wrong in the one given by M.S. Turan and O. Kara [5], which would affect the subsequent results.

Next, we show that there are still some improvements for our purpose when using a cube–linear attack. For the sake of analysis, we denote $K_{\mathrm{monomials}}$ and $\mathrm{IV}_{\mathrm{monomials}}$ the XOR of monomials involving only the key bits and the XOR of monomials involving only the IV bits, respectively and set the XOR of the remaining monomials as $(\mathrm{IV} \cdot K)_{\mathrm{monomials}}$. Then, (7) could be written as follows:

$$\begin{aligned} z_1 &= K_{\mathrm{monomials}} + \mathrm{IV}_{\mathrm{monomials}} + (\mathrm{IV} \cdot K)_{\mathrm{monomials}} \\ &= K_{\mathrm{monomials}} + \mathrm{IV}_{\mathrm{monomials}} + \mathrm{iv}_{25} \cdot k_{14} + \mathrm{iv}_{25} \cdot k_{41} + \mathrm{iv}_{25} \cdot k_{39} \cdot k_{40} \\ &\quad + \mathrm{iv}_{26} \cdot k_{13} + \mathrm{iv}_{26} \cdot k_{40} + \mathrm{iv}_{26} \cdot k_{38} \cdot k_{39} + \mathrm{iv}_{31} \cdot k_{20} + \mathrm{iv}_{31} \cdot k_{47} \\ &\quad + \mathrm{iv}_{31} \cdot k_{45} \cdot k_{46} + \mathrm{iv}_{32} \cdot k_{19} + \mathrm{iv}_{32} \cdot k_{46} + \mathrm{iv}_{32} \cdot k_{44} \cdot k_{45} + \mathrm{iv}_{49} \cdot k_{38} \\ &\quad + \mathrm{iv}_{49} \cdot k_{65} + \mathrm{iv}_{49} \cdot k_{63} \cdot k_{64} + \mathrm{iv}_{50} \cdot k_{37} + \mathrm{iv}_{50} \cdot k_{64} + \mathrm{iv}_{50} \cdot k_{62} \cdot k_{63} \\ &\quad + \mathrm{iv}_{70} \cdot k_{59} + \mathrm{iv}_{71} \cdot k_{58} + \mathrm{iv}_{76} \cdot k_{65} + \mathrm{iv}_{77} \cdot k_{64}. \end{aligned} \tag{8}$$

Observing (8), it is very 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 whether there is such an index subset I that leads to a linear expression $P_{S(I)}$ based on the basic idea of cube attacks. Actually, the four index subsets {70}, {71}, {76}, and {77} in (8) are available for recovering k_{58}, k_{59}, k_{64}, and k_{65}, respectively. Taking the index subset $I = \{77\}$ as a case in point, we explain how to recover k_{64} using our method.

Step 1. Determine the index subset $I = \{77\}$ and $C_I = \{\mathrm{iv}_{77}\}$ accordingly.

Step 2. According to cube attacks, the other public variables could be assigned any values except for those variables belonging to C_I. For simplicity, we set iv_{25}, iv_{26}, iv_{31}, iv_{32}, iv_{49}, and iv_{50} to zero. When $\mathrm{iv}_{77} = 0$ (that is, iv_{25}, iv_{26}, iv_{31}, iv_{32}, iv_{49}, iv_{50}, iv_{70}, iv_{71}, iv_{76}, and iv_{77} all equal zero), we get the following:

$$K_{\mathrm{monomials}} = z, \tag{9}$$

Table 2. Key information recovered.

(iv_{25}, iv_{26}, iv_{31}, iv_{32}, iv_{49}, iv_{50}, iv_{70}, iv_{71}, iv_{76}, iv_{77})	Key	Bias	Data complexity	Success rate (%)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0)	$K_{monimials}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 1, 0, 0, 0)	$K_{monimials}+k_{59}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 0, 1, 0, 0)	$K_{monimials}+k_{58}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 0, 0, 1, 0)	$K_{monimials}+k_{65}$	2^{-9}	2^{19}	99.77
(0, 0, 0, 0, 0, 0, 0, 0, 0, 1)	$K_{monimials}+k_{64}$	2^{-9}	2^{19}	99.77

which holds with a bias of 2^{-9}, where z stands for the XOR of all the known variables in (8) after assigning IV. When $iv_{77} = 1$ (that is, iv_{25}, iv_{26}, iv_{31}, iv_{32}, iv_{49}, iv_{50}, iv_{70}, iv_{71}, iv_{76} are all equal zero, and iv_{77} equals one), we get the following:

$$k_{64} + K_{\text{monomials}} = z', \tag{10}$$

which holds with a bias of 2^{-9}, where z' stands for the XOR of all the known variables in (8) after assigning IV.

Note that (9) and (10) are independent of each other since the key bit k_{64} could be considered as a random variable. According to Theorem 4 and Table 1, here, we can achieve a lower data complexity using scheme A compared to scheme B at the same level of success rate. Therefore, we can respectively recover $K_{\text{monomials}}$ and $k_{64} + K_{\text{monomials}}$ with a success rate of 99.77% and data complexity of 2^{19} IVs according to scheme A.

Step 3. Based on the above analysis, $k_{64} = (k_{64} + K_{\text{monomials}}) + (K_{\text{monomials}})$. The success rate to recover k_{64} is 99.54% 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 $(1–0.9977)^2$ can be ignored according to Theorem 2.

Similarly, k_{58}, k_{59}, and k_{65} can be recovered correspondingly for the other index subsets {70}, {71}, and {76}. Therefore, we can simultaneously recover the four key bits k_{58}, k_{59}, k_{64}, and k_{65} with success rate 98.17% (0.9977^8) and $2^{21.32}$ IVs ($5 \times 2^{19} \approx 2^{21.32}$). The above results are given in Table 2.

3. Search for Better Linear Approximations

The above attack provides us with a paradise of improvement in the search for better linear approximations. Since we aren't aware of the actual values of the four key bits recovered and they are dependent upon concrete scenarios, note that all 16 possible values of the four bits now have to be considered. In the following article, we take (k_{58}, k_{59}, k_{64}, k_{65}) = (1, 1, 1, 1) as an example to illustrate how to look for linear approximations with bigger bias.

Returning (k_{58}, k_{59}, k_{64}, k_{65}) = (1, 1, 1, 1) to (7), we have equation (11) (see Appendix A). Observing (11), there are 34 linear, 48 quadratic, and 18 cubic terms. The linear approximation to (11) could be naturally obtained as

$$\begin{aligned} z_1 = {} & 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} \\ & + k_{39} + k_{54} + k_{57} + k_{63} + k_{67} + k_{68} + k_{69} + k_{72} + iv_3 \\ & + iv_6 + iv_{21} + iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{49} \\ & + iv_{50} + iv_{51} + iv_{70} + iv_{71} + iv_{72} + iv_{76} + iv_{77} + iv_{78} \end{aligned} \tag{12}$$

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

According to (5)–(12), the bias that (12) holds could 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 in particular, it is 80 bits for Trivium). However, the bias is too small to be used to recover any information about the key. According to the method used in [5], the magnitude of the bias can be increased when allowed to choose some special key bits and IV bits. Unfortunately, the rules of selection weren't provided, and furthermore, the selected bits in [5] weren't also the best. So, here, Algorithm 2 (see Appendix B) is proposed to choose those special key bits or IV bits.

Denote $\Omega_K = \{k_i \mid k_i = 0,\ 1 \le i \le 80\}$ and $\Omega_{IV} = \{iv_i \mid iv_i = 0,\ 1 \le i \le 80\}$ as the key bit set chosen to zero and the IV bit set chosen to zero, respectively. Then, $|\Omega_K|$ and $|\Omega_{IV}|$ represent the size of Ω_K and Ω_{IV}, respectively, and n_1 and n_2 are referred to as the individual quantity of the remaining quadratic and cubic terms, respectively, after Ω_K and Ω_{IV} have been determined. The bias of a linear approximation to (11) 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 to choose Ω_K and Ω_{IV}. Algorithm 2 takes finding better linear approximations as a criterion, and the main ideas are to select such a bit that can eliminate the most nonlinear monomials when being set to zero. More detailed rules to cope with more complicated scenarios are showed in Algorithm 2. When $|\Omega_K| = 10$ and $|\Omega_{IV}| = 10$, for instance, Table 3 provides the two sets Ω_K and Ω_{IV} using Algorithm 2.

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

$$\begin{aligned} z_1 = {} & 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} + k_{39} \\ & + k_{54} + k_{57} + k_{63} + k_{67} + k_{68} + k_{69} + k_{72} + iv_3 + iv_6 \\ & + iv_{21} + iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{49} + iv_{50} \\ & + iv_{51} + iv_{70} + iv_{71} + iv_{72} + iv_{76} + iv_{77} + iv_{78}, \end{aligned} \tag{13}$$

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
$iv_{10}\ iv_{13}\ iv_{25}\ iv_{31}\ iv_{34}$ $iv_{37}\ iv_{40}\ iv_{50}\ iv_{54}\ iv_{55}$	$k_{13}\ k_{19}\ k_{23}\ k_{38}$ $k_{40}\ k_{45}\ k_{46}\ k_{50}\ k_{63}$	k_5	2^{-23}
		k_{67}	2^{-23}
		k_{68}	2^{-23}

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

$(k_{58},k_{59},k_{64},k_{65})$	$\lvert\Omega_{IV}\rvert = 10$	$\lvert\Omega_K\rvert = 10$		Bias
(0, 0, 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}
(0, 1, 0, 0)			k_{67}	2^{-23}
(1, 0, 0, 0)			k_{68}	2^{-23}
(1, 1, 0, 0)				
(0, 0, 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}
(0, 1, 1, 1)			k_{67}	2^{-23}
(1, 0, 1, 1)			k_{68}	2^{-23}
(1, 1, 1, 1)				
(0, 0, 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}
(0, 1, 0, 1)			k_{67}	2^{-23}
(1, 0, 0, 1)			k_{68}	2^{-23}
(1, 1, 0, 1)				
(0, 0, 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}
(0, 1, 1, 0)			k_{67}	2^{-24}
(1, 0, 1, 0)			k_{68}	2^{-24}
(1, 1, 1, 0)				

Table 5. Comparison with previous results.

	$(\lvert\Omega_K\rvert, \lvert\Omega_{IV}\rvert)$	Bias	Num.	Data complexity	Success rate (%)
Turan [5]	(10, 10)	2^{-31}	1	2^{62}	97.8
Ours	(10, 10)	$2^{-23.2}$	3	$2^{47.2}$	97.8
Jia [19]	(10, 13)	2^{-31}	2	2^{61}	97.8
Ours	(10, 13)	$2^{-20.2}$	3	$2^{41.2}$	97.8

$$\begin{aligned} z_1 = {} & 1 + k_3 + k_6 + k_{15} + k_{21} + k_{27} + k_{30} + k_{37} + k_{38} + k_{39} \\ & + k_{54} + k_{57} + k_{63} + k_{68} + k_{69} + k_{72} + iv_3 + iv_6 + iv_{21} \\ & + iv_{24} + iv_{30} + iv_{33} + iv_{39} + iv_{45} + iv_{49} + iv_{50} + iv_{51} \\ & + iv_{70} + iv_{71} + iv_{72} + iv_{76} + iv_{77} + iv_{78}, \end{aligned} \tag{14}$$

and

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

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

4. Comparison with Previous Results

For the reduced version of 288-round Trivium, M.S. Turan and O. Kara in [5] found a linear approximation with bias of 2^{-31} when $|\Omega_K| = 10$ and $|\Omega_{IV}| = 10$. In [19], Jia and others presented a multiple linear cryptanalysis on 288-round Trivium, resulting in another linear approximation with the same bias 2^{-31} when $|\Omega_K| = 10$ and $|\Omega_{IV}| = 13$. In this paper, we find some linear approximations with bigger bias. Moreover, the additional four key bits are recovered. These results are summarized in Table 5. Obviously, our attack is better than the previous.

V. Conclusion

In cryptography, linear cryptanalysis is the most basic cryptanalysis approach aiming to look for a linear relation between inputs and outputs. A variety of refinements to the attack have been suggested in the past. In this paper, a novel technique called cube–linear attack is proposed (for the first time) to take a probabilistic polynomial as the attack target and furthermore to mine the secret information that has yet to be exploited by previous linear cryptanalysis methods. 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 on Trivium, we get better linear cryptanalysis results. Although a few better cryptanalytic results on Trivium had been published earlier using other attacks, we believe that our method is meaningful from the point of view of improving linear cryptanalysis; furthermore, it could be extended to other ciphers. For example, we have improved the bias of the revised 288-round Trivium algorithms proposed in [20] and [21] by 2^{14} and 2^4, respectively. Therefore, our method is worth considering in launching linear attacks on a cryptosystem.

Appendix A

The two equations (7) and (11) appearing in this paper are as follows:

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

$$+\mathrm{iv}_{54}\cdot\mathrm{iv}_{55}+\mathrm{iv}_{58}\cdot\mathrm{iv}_{59}+\mathrm{iv}_{61}\cdot\mathrm{iv}_{62}+\mathrm{iv}_{67}\cdot\mathrm{iv}_{68}+\mathrm{iv}_{70}\cdot\mathrm{iv}_{71}$$
$$+\mathrm{iv}_{76}\cdot k_{65}+\mathrm{iv}_{73}\cdot\mathrm{iv}_{74}+k_{13}\cdot k_{39}\cdot k_{40}+k_{14}\cdot k_{38}\cdot k_{39}+k_{19}\cdot k_{45}\cdot k_{46}$$
$$+k_{20}\cdot k_{44}\cdot k_{45}+\ k_{37}\cdot k_{63}\cdot k_{64}+\mathrm{iv}_{70}\cdot k_{59}+k_{38}\cdot k_{39}\cdot k_{40}+k_{38}\cdot k_{39}\cdot k_{41}$$
$$+k_{38}\cdot k_{62}\cdot k_{63}+k_{44}\cdot k_{45}\cdot k_{46}+k_{44}\cdot k_{45}\cdot k_{47}+k_{62}\cdot k_{63}\cdot k_{64}$$
$$+\mathrm{iv}_{71}\cdot k_{58}+k_{62}\cdot k_{63}\cdot k_{65}+\mathrm{iv}_{40}\cdot\mathrm{iv}_{54}\cdot\mathrm{iv}_{55}+\mathrm{iv}_{41}\cdot\mathrm{iv}_{53}\cdot\mathrm{iv}_{54}$$
$$+\mathrm{iv}_{53}\cdot\mathrm{iv}_{54}\cdot\mathrm{iv}_{55}+\mathrm{iv}_{53}\cdot\mathrm{iv}_{54}\cdot\mathrm{iv}_{56}+\mathrm{iv}_{25}\cdot k_{14}+\mathrm{iv}_{25}\cdot k_{41}+\mathrm{iv}_{25}\cdot k_{39}\cdot k_{40}$$
$$+\mathrm{iv}_{26}\cdot k_{13}+\mathrm{iv}_{26}\cdot k_{40}+\mathrm{iv}_{26}\cdot k_{38}\cdot k_{39}+\mathrm{iv}_{31}\cdot k_{20}+\mathrm{iv}_{31}\cdot k_{47}+\mathrm{iv}_{31}\cdot k_{45}\cdot k_{46}$$
$$+\mathrm{iv}_{32}\cdot k_{46}+\mathrm{iv}_{32}\cdot k_{19}+\mathrm{iv}_{32}\cdot k_{44}\cdot k_{45}+\mathrm{iv}_{49}\cdot k_{38}+\mathrm{iv}_{49}\cdot k_{65}+\mathrm{iv}_{49}\cdot k_{63}\cdot k_{64}$$
$$+\mathrm{iv}_{50}\cdot k_{37}+\mathrm{iv}_{50}\cdot k_{64}+\mathrm{iv}_{50}\cdot k_{62}\cdot k_{63}. \quad (7)$$

$$z_1 = 1+k_3+k_6+k_{15}+k_{21}+k_{27}+k_{30}+k_{37}+k_{38}+k_{39}+k_{54}$$
$$+k_{57}+k_{63}+k_{67}+k_{68}+k_{69}+k_{72}+\mathrm{iv}_3+\mathrm{iv}_6+\mathrm{iv}_{21}+\mathrm{iv}_{24}$$
$$+\mathrm{iv}_{30}+\mathrm{iv}_{33}+\mathrm{iv}_{39}+\mathrm{iv}_{45}+\mathrm{iv}_{49}+\mathrm{iv}_{50}+\mathrm{iv}_{51}+\mathrm{iv}_{70}+\mathrm{iv}_{71}$$
$$+\mathrm{iv}_{72}+\mathrm{iv}_{76}+\mathrm{iv}_{77}+\mathrm{iv}_{78}+k_4\cdot k_5+k_{13}\cdot k_{14}+k_{13}\cdot k_{41}$$
$$+k_{14}\cdot k_{40}+k_{16}\cdot k_{17}+k_{19}\cdot k_{20}+k_{19}\cdot k_{47}+k_{22}\cdot k_{23}+k_{20}\cdot k_{46}$$
$$+k_{25}\cdot k_{26}+k_{28}\cdot k_{29}+k_{34}\cdot k_{35}+k_{37}\cdot k_{38}+k_{37}\cdot k_{63}+k_{39}\cdot k_{40}$$
$$+k_{43}\cdot k_{44}+k_{45}\cdot k_{46}+k_{49}\cdot k_{50}+k_{52}\cdot k_{53}+k_{61}\cdot k_{62}+k_{67}\cdot k_{68}$$
$$+k_{70}\cdot k_{71}+\mathrm{iv}_{10}\cdot\mathrm{iv}_{11}+k_{70}\cdot k_{71}+\mathrm{iv}_{10}\cdot\mathrm{iv}_{11}+\mathrm{iv}_{13}\cdot\mathrm{iv}_{14}+\mathrm{iv}_{25}\cdot\mathrm{iv}_{26}$$
$$+\mathrm{iv}_{31}\cdot\mathrm{iv}_{32}+\mathrm{iv}_{34}\cdot\mathrm{iv}_{35}+\mathrm{iv}_{37}\cdot\mathrm{iv}_{38}+\mathrm{iv}_{40}\cdot\mathrm{iv}_{56}+\mathrm{iv}_{41}\cdot\mathrm{iv}_{55}$$
$$+\mathrm{iv}_{49}\cdot\mathrm{iv}_{50}+\mathrm{iv}_{54}\cdot\mathrm{iv}_{55}+\mathrm{iv}_{58}\cdot\mathrm{iv}_{59}+\mathrm{iv}_{61}\cdot\mathrm{iv}_{62}+\mathrm{iv}_{67}\cdot\mathrm{iv}_{68}$$
$$+\mathrm{iv}_{70}\cdot\mathrm{iv}_{71}+\mathrm{iv}_{73}\cdot\mathrm{iv}_{74}+k_{13}\cdot k_{39}\cdot k_{40}+k_{14}\cdot k_{38}\cdot k_{39}+k_{19}\cdot k_{45}\cdot k_{46}$$
$$+k_{20}\cdot k_{44}\cdot k_{45}+k_{38}\cdot k_{39}\cdot k_{40}+k_{38}\cdot k_{39}\cdot k_{41}+k_{38}\cdot k_{62}\cdot k_{63}$$
$$+k_{44}\cdot k_{45}\cdot k_{46}+k_{44}\cdot k_{45}\cdot k_{47}+\mathrm{iv}_{40}\cdot\mathrm{iv}_{54}\cdot\mathrm{iv}_{55}+\mathrm{iv}_{41}\cdot\mathrm{iv}_{53}\cdot\mathrm{iv}_{54}$$
$$+\mathrm{iv}_{53}\cdot\mathrm{iv}_{54}\cdot\mathrm{iv}_{55}+\mathrm{iv}_{53}\cdot\mathrm{iv}_{54}\cdot\mathrm{iv}_{56}+\mathrm{iv}_{25}\cdot k_{14}+\mathrm{iv}_{25}\cdot k_{41}$$
$$+\mathrm{iv}_{25}\cdot k_{39}\cdot k_{40}+\mathrm{iv}_{26}\cdot k_{13}+\mathrm{iv}_{26}\cdot k_{40}+\mathrm{iv}_{26}\cdot k_{38}\cdot k_{39}+\mathrm{iv}_{31}\cdot k_{20}$$
$$+\mathrm{iv}_{31}\cdot k_{47}+\mathrm{iv}_{31}\cdot k_{45}\cdot k_{46}+\mathrm{iv}_{32}\cdot k_{46}+\mathrm{iv}_{32}\cdot k_{19}+\mathrm{iv}_{32}\cdot k_{44}\cdot k_{45}$$
$$+\mathrm{iv}_{49}\cdot k_{38}+\mathrm{iv}_{49}\cdot k_{63}+\mathrm{iv}_{50}\cdot k_{37}+\mathrm{iv}_{50}\cdot k_{62}\cdot k_{63}. \quad (11)$$

Appendix B

Algorithm 2. Algorithm to choose special key and IV bits.

Reset Ω_K and Ω_{IV};

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

Step 1: Reset Ω and Π, count the frequency of each bit in all nonlinear monomials, put the bits with the highest frequency in Ω;

Step 2:

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};

Step 3: Calculate the polynomial again, based on chosen Ω_K and Ω_{IV};

Step 4:

if $\left((|\Omega_K| < n_K)\ \&\&\ (|\Omega_{\mathrm{IV}}| < n_{\mathrm{IV}})\right)$ Return to Step 1;

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

Stop searching the key bits, return to Step 1, and continue to search the IV bits;

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

Stop searching the IV bits, return to Step 1, and continue to search the key bits;

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

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

End

References

[1] J.D. Golić, "Linear Cryptanalysis of Stream Ciphers," *Fast Softw. Encryption: Int. Workshop*, Leuven, Belgium, Dec. 14–16, 1994, pp. 154–169.

[2] J.D. Golić, V. Bagini, and G. Morgari, "Linear Cryptanalysis of Bluetooth Stream Cipher," *EUROCRYPT*, Amsterdam, Netherlands, Apr. 28–May 2, 2002, pp. 238–255.

[3] F. Muller and T. Peyrin, "Linear Cryptanalysis of the TSC Family of Stream Ciphers," *ASIACRYPT*, Chennai, India, Dec. 4–8, 2005, pp. 373–394.

[4] S. Khazaei and M. Hassanzadeh, "Linear Sequential Circuit Approximation of the Trivium Stream Ciphers," ECRYPT Stream Cipher Project, EU, Rep. 2005/063, Jan. 2006.

[5] M.S. Turan and O. Kara, "Linear Approximations for 2-Round Trivium," *Int. Conf. Security Inf. Netw.*, Gazimagusa, North Cyprus, May 8–10, 2007, pp. 96–105.

[6] M. Matsui and A. Yamagishi, "A New Method for Known Plaintext Attack of FEAL Cipher," *EUROCRYPT*, Balatonfüred, Hungary, May 24–28, 1992, pp. 81–91.

[7] A. Shimizu and S. Miyaguchi, "Fast Data Encipherment Algorithm FEAL," *EUROCRYPT*, Amsterdam, Netherlands, Apr. 13–15, 1987, pp. 267–278.

[8] M. Matsui, "Linear Cryptanalysis Method for DES Cipher," *EUROCRYPT*, Lofthus, Norway, May 23–27, 1993, pp. 386–397.

[9] M. Matsui, "The First Experimental Cryptanalysis of the Data Encryption Standard," *CRYPTO*, Santa Barbara, CA, USA, Aug. 21–25, 1994, pp. 1–11.

[10] B.S. Kaliski and M.J.B. Robshaw, "Linear Cryptanalysis Using Multiple Approximations," *CRYPTO*, Santa Barbara, CA, USA, Aug. 21–25, 1994, pp. 26–39.

[11] S.K. Langford and M.E. Hellman, "Differential-Linear Cryptanalysis," *CRYPTO*, Santa Barbara, CA, USA, Aug. 21–25, 1994, pp. 17–25.

[12] L.R. Knudsen and M.J.B. Robshaw, "Non-linear Approximations in Linear Cryptanalysis," *EUROCRYPT*, Saragossa, Spain, May 12–16, 1996, pp. 224–236.

[13] A. Bogdanov and V. Rijmen, *Linear Hulls with Correlation Zero and Linear Cryptanalysis of Block Ciphers*, Cryptology ePrint Archive, 2011. Accessed Aug. 5, 2013. http://eprint.iacr.org/2011/123

[14] I. Dinur and A. Shamir, "Cube Attacks on Tweakable Black Box

Polynomials," *EUROCRYPT*, Cologne, Germany, Apr. 26–30, 2009, pp. 278–299.

[15] M. Vielhaber, *Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack*, Cryptology ePrint Archive, 2007. Accessed Aug. 10, 2013. http://eprint.iacr.org/2007/413

[16] P.-A. Fouque and T. Vannet, "Improving Key Recovery to 784 and 799 Rounds of Trivium Using Optimized Cube Attacks," *Fast Softw. Encryption: Int. Workshop*, Singapore, Mar. 11–13, 2013, pp. 502–517.

[17] C. De Cannière and B. Preneel, "TRIVIUM: A Stream Cipher Construction Inspired by Block Cipher Design Principles," *Inf. Security*, Samos Island, Greece, Aug. 30–Sept. 2, 2006, pp. 171–186.

[18] eSTREAM, *The ECRYPT Stream Cipher Project*, 2008. Accessed Nov. 15, 2013. http://www.ecrypt.eu.org/stream

[19] Y. Jia et al., "Linear Cryptanalysis of 2-Round Trivium with Multiple Approximations," *J. Electron. Inf. Technol.*, vol. 33, no. 1, 2011, pp. 223–227.

[20] M. Afzal and A. Masood, *Modifications in the Design of Trivium to Increase its Security Level*, Cryptology ePrint Archive Report, 2009. Accessed Aug. 15, 2013. http://eprint.iacr.org/2009/250

[21] A.S. Raj and C. Srinivasan, "Analysis of Algebraic Attack on Trivium and Minute Modification to Trivium," *Conf. Netw. Security Appl.*, Chennai, India, July 15–17, 2011, pp. 35–42.

Wen-Long Sun received his BE and MSc degrees in cryptology from the Information Science and Technology Institute, Zhengzhou, China, in 2011 and 2014, respectively. Now, he is an assistant engineer at the Beijing Satellite Navigation Center, Beijing, China. His main research interests include cryptology and information security.

Jie Guan received her PhD degree in cryptology from the Information Science and Technology Institute, Zhengzhou, China, in 2004. She is currently a professor at the Information Science and Technology Institute. Her main subject interest is cryptology, and she teaches information systems, the theory of cryptography, and quantum computation.