IET Information Security

Research Article

Improved guess-and-determine attack on TRIVIUM

ISSN 1751-8709
Received on 9th July 2018
Revised 24th January 2019
Accepted on 21st February 2019
E-First on 25th March 2019
doi: 10.1049/iet-ifs.2018.5357
www.ietdl.org

Lin Jiao[1] ✉, *Yonglin Hao*[1], *Yongqiang Li*[2]

[1]*State Key Laboratory of Cryptology, Beijing 100878, People's Republic of China*

[2]*State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, People's Republic of China*

✉ *E-mail: jiaolin_jl@126.com*

Abstract: TRIVIUM is a stream cipher of the finalists by eSTREAM project and has been accepted as ISO standard. Although the design has a simple structure, no attack on its full cipher has been found yet. In this study, based on Maximov and Biryukov's attack, the authors present an improved guess-and-determine attack on TRIVIUM. Analysis details are provided corresponding to TRIVIUM specifications for better comprehension, and errors that may lead to higher attack complexity in the original attack are pointed and corrected. They further bring in some techniques like backward-clock equation collection, quadratic equations, linear transformation to improve the attack. In addition, they integrate with time-memory-data tradeoffs from the framework, based on the analysis of the coefficient matrices form of derived linear equation systems on the internal state. In this way, better use of the imposed quadratic conditions can be made, which leads to reduced attack complexity by filtering out the impossible keystreams before solving the equation systems. Their attack offers more parameter selections, and gives several borderline results compared with the key exhaustive search. The new attack behaves better in the original case. It also verifies the necessity of data requirement imposed on TRIVIUM, which is questioned in TRIVIUM specifications.

1 Introduction

Trivium is a hardware oriented synchronous stream cipher designed by De Cannière and Preneel [1]. It is one of the finalists by eSTREAM project and has been accepted as ISO standard on lightweight stream ciphers [2]. The structure of Trivium is elegant and simple, with only bit operations, which makes it applicable to source restricted applications, such as radio frequency identification tags. It also provides a reasonably efficient software implementation.

Trivium is based on three cascaded non-linear feedback shift registers of different lengths, which update by iterating three interconnected quadratic recurrence relations, i.e. at each round, a bit is shifted into each of the three registers using a combination of taps from that and another register. Trivium has an internal state of 288 bits, and accepts an 80-bit secret key and an 80-bit initialisation vector (IV). It is initialised with the secret key and the IV written into two of the registers, and some constants in a fixed pattern for the remaining bits. Moreover, 1152 blank rounds are iterated before producing the outputs. The keystream bit is generated by linearly XORing together some internal state bits per subsequent iteration. No taps appear on the first 65 bits of each register, so each novel state bit is not used until at least 65 rounds after it is generated. This is the key to Trivium's software performance and flexibility in hardware allowing for parallelisation.

There have been lots of cryptanalysis of Trivium since its submission, and yet no attack was discovered on its full version. For the analysis of initialisation stage, early results include the chosen IV statistical attack [3, 4], which recovered the key on Trivium reduced to 672 rounds, and the distinguishing attack on 961-round Trivium with practical complexity for weak keys, inspired by the message modification technique and the conditional differential tool [5]. Cube attacks are the major method for recent cryptanalysis results on reduced round Trivium [6–12], which exploit low degree properties of the output bit's ANF over IV bits, and the best published result to date is on 855-round Trivium [The result exhibits correctness controversy.]. The analysis for state recovery or distinguishing in keystream generation stage is few [13–15], and the corresponding complexity is high. In [16], two trivial attacks including a state recovering and a linear distinguishing on Trivium were proposed, and it shows that Trivium has a very thin safety margin with a state-recovery complexity claimed $c \cdot 2^{83.5}$, where c denotes the time needed for solving a linear equation system of 192 variables.

The guess-and-determine attack is a cryptanalysis method that has been applied to various stream ciphers [17–21], aiming at recovering the internal state with the knowledge of some guesses and known keystream bits. Sometimes, a series of conditions are imposed to make the algebraic structure of the cipher keep simple to be deduced, which can be identified as a common guess-and-determine technique. Comparing the keystream bits generated by the recovered internal state with those known in advance, we are able to identify the correctness. The complexity of such attack is dominated by the number of guesses that have to be made and the success probability of those imposed conditions. A good attacking scenario should extract more information from the keystream bits.

This paper gives an improved guess-and-determine attack on Trivium for state recovery, based on the first attack in [16]. The original attack works on the observations that all blocks of the internal state are divisible by 3, and the update of the internal state is a linear combination of one coresidual subset plus a minor AND bit disturbance from the adjacent two subsets. Thus it is valid to guess and determine one coresidual subset and then the other two subsets of the internal state sequentially given the keystream bits under several conditions.

In this paper, more analysis details omitted in [16] are provided for better comprehension. The attack is formulated corresponding with the description in Trivium specifications [1], and therefore the transformation of the internal state with time looks simple and clear. The internal state can be divided into three coresidual subsets module 3, and further, expressions of the output and update functions according to the partition definition can be presented. Next, we conduct the divide and conquer attack. Firstly, we aim to derive one subset, and for this, a systematical analysis about the components of the confined linear equation system is given, which corresponds with the time instants before the feedback bits running into the outputs, the conditions imposed on the AND gate in the

Table 1 Summary of analysis results of TRIVIUM

Type	Round	Complexity	Reference	Attack
key recovery	736	2^{30}	Eurocrypt 2009 [6]	cube
key recovery	767	2^{36}	Eurocrypt 2009 [6]	cube
key recovery	799	practical	FSE 2013 [8]	cube
key recovery	832	2^{77}	Crypto 2017 [9]	cube
key recovery	835	2^{75}	Eurocrypt 2018 [10]	cube
key recovery	839	2^{79}	Crypto 2018 [12]	cube
key recovery	855	2^{77}	Crypto 2018 [11]	cube
key recovery	1152	$2^{90.17}$	—	exhaustive search
state recovery	1152	2^{164}	eSTREAM 2006 [14]	guess and determine
state recovery	1152	$c \cdot 2^{135}$, $c \simeq 2^{21.8}$	eSTREAM 2006 [13]	guess and determine
distinguishing	1152	2^{144}	eSTREAM 2005 [15]	linear approximate
state recovery	1152	$c \cdot 2^{83.5}$, $c \simeq 2^{16}$ (claimed)	SAC 2007 [15]	guess and determine
state recovery	1152	$2^{88.63}$	Section 5	guess and determine

update functions, and the probabilistic linear equations collected with bias. We deduce the time slicing nodes, the number of imposed conditions, and the bias of the probability equations soundly in this paper. Secondly, we go deeper into the components of the linear equation system on the other two subsets, based on the known first part. We first consider collecting forward clocks, and it is important to note that [16] (see Table 5 in [16]) just included the forward-clock equations, which could only collect 190 rather than the claimed 192 linear equations under the given conditions. Thus the real complexity of the original attack must be higher. We add in the specific analysis of backward clocks, and derive two more linear equations. Furthermore, we introduce a trick to transform three quadratic imposed conditions into two linear equations. Based on these equations, the internal state is able to be recovered.

We explain how the attack process evolves, and analyse the form of the coefficient matrices of linear equation systems on the first and the other two subsets, respectively. We point out that it is too optimistic to estimate the complexity $c \simeq 2^{16}$ in [16]. To further reduce the attack complexity overall, we propose an improved scheme combining with the technique of time-memory-data tradeoffs. The new scheme takes the advantage of being able to filter out the keystreams that cannot lead to the correct internal state before solving the equation systems by the imposed conditions, which is not allowed in [16]. It makes better use of those quadratic conditions.

We provide several results in different cases, by controlling the parameter selections according to the security bounds of TRIVIUM. With the same measurement, our attack takes a complexity better than that in [16]. Moreover, compared with the key exhaustive search, our results are borderline. Virtually, we hope our attack may provide a wider range of ideas and perspectives on the analysis of TRIVIUM-like stream ciphers, although it cannot be regarded in a practical sense. The summary of the previous results and new result including state recovery attacks and key recovery attacks on the full and reduced round TRIVIUM are shown in Table 1.

This paper is organised as follows. We first give a brief introduction to TRIVIUM in Section 2. In Section 3, we present a concrete guess-and-determine attack on TRIVIUM. In Section 4, we provide an improved framework with time-memory-data tradeoffs technique. In Section 5, we analyse the parameter selection and complexity of the improved attack. Finally, some conclusions are provided in Section 6.

2 Description of TRIVIUM

TRIVIUM is a synchronous stream cipher designed to generate up to 2^{64} bits of keystream from an 80-bit secret key and an 80-bit IV. As for most stream ciphers, this process consists of two phases: the internal state initialisation and the keystream generation. It contains a 288-bit internal state denoted by $(s_1, \dots, s_{288})$, which consists of three basic registers, and each register consists of three blocks, each of size divisible by 3, shown in Table 2. The keystream generation consists of an iterative process that extracts the values of 15 specific state bits and uses them both to update 3 bits of the state, and to compute 1 bit of keystream z_i as follows. The state bits are then rotated and the process repeats itself until the requested N keystream bits have been generated.

```
for $i = 1$ to $N$ do the iteration for TRIVIUM
    $t_1 \leftarrow s_{66} + s_{93}$
    $t_2 \leftarrow s_{162} + s_{177}$
    $t_3 \leftarrow s_{243} + s_{288}$
    $z_i \leftarrow t_1 + t_2 + t_3$
    $t_1 \leftarrow t_1 + s_{91} \cdot s_{92} + s_{171}$
    $t_2 \leftarrow t_2 + s_{175} \cdot s_{176} + s_{264}$
    $t_3 \leftarrow t_3 + s_{286} \cdot s_{287} + s_{69}$
    $(s_1, s_2, \dots, s_{93}) \leftarrow (t_3, s_1, \dots, s_{92})$
    $(s_{94}, s_{95}, \dots, s_{177}) \leftarrow (t_1, s_{94}, \dots, s_{176})$
    $(s_{178}, s_{279}, \dots, s_{288}) \leftarrow (t_2, s_{178}, \dots, s_{287})$
end for
```

Note that here '+' and ' ·' stand for addition and multiplication over GF(2) (i.e. XOR and AND), respectively. The graphical representation is shown in Fig. 1.

The algorithm is initialised by loading an 80-bit key and an 80-bit IV into the 288-bit initial state, and setting all remaining bits to 0, except for s_{286}, s_{287}, and s_{288}, as

$$(s_1, s_2, \dots, s_{93}) \leftarrow (K_1, \dots, K_{80}, 0, \dots, 0),$$

$$(s_{94}, s_{95}, \dots, s_{177}) \leftarrow (IV_1, \dots, IV_{80}, 0, \dots, 0),$$

$$(s_{178}, s_{279}, \dots, s_{288}) \leftarrow (0, \dots, 0, 1, 1, 1).$$

Then, the state is rotated over four full cycles, i.e. 1152 iterations in the same way as above, but without generating keystream bits.

3 Improved divide and conquer attack on TRIVIUM

In this section, we present an improved guess-and-determine attack on TRIVIUM, based on the first attack given in [16]. Firstly, we redefine the internal state bit by attaching the time instant as $s_{i,t}$ ($i = 1, 2, \dots, 288$, $t = 0, 1, \dots$), where i is the index of the internal state, and t is the time instant. One internal state bit $s_{i,t}$ is generated by feedback if $t \geq i - j + 1$, where $j \in \{1, 94, 178\}$. The output function can be rewritten as

$$z_t = s_{66,t} + s_{93,t} + s_{162,t} + s_{177,t} + s_{243,t} + s_{288,t}, \quad (1)$$

and the update functions can be rewritten as

$$\begin{aligned} s_{i,t+1} &= s_{i-1,t}, \quad i \neq 1, 94, 178, \\ s_{1,t+1} &= s_{243,t} + s_{288,t} + s_{286,t} \cdot s_{287,t} + s_{69,t}, \\ s_{94,t+1} &= s_{66,t} + s_{93,t} + s_{91,t} \cdot s_{92,t} + s_{171,t}, \\ s_{178,t+1} &= s_{162,t} + s_{177,t} + s_{175,t} \cdot s_{176,t} + s_{264,t}. \end{aligned} \tag{2}$$

Based on the observation that the internal state blocks of Trivium are all divisible by 3 (shown in Table 2), the internal state can be divided into three subsets defined as

$$\begin{aligned} \Gamma_0 &= \{s_{i,t} \mid (i-t) \bmod 3 = 0\}, \\ \Gamma_1 &= \{s_{i,t} \mid (i-t) \bmod 3 = 1\}, \\ \Gamma_2 &= \{s_{i,t} \mid (i-t) \bmod 3 = 2\}. \end{aligned}$$

We rewrite the output function and update functions for further derivation, with the following abstract notations, which can be easily comprehended corresponding to (1) and (2)

$$\begin{aligned} z_{3k} &= \sum \Gamma_0, \\ z_{3k+1} &= \sum \Gamma_2, \\ z_{3k+2} &= \sum \Gamma_1, \end{aligned}$$

where $k = 0, 1, \ldots$. Here for example, $z_{3k} = \sum \Gamma_0$ means that z_{3k} is always generated by XORing some state bits from Γ_0. Let $\star \in \{1, 94, 178\}$, then we have

$$\begin{aligned} \Gamma_0 \ni s_{\star,3k+1} &= \sum \Gamma_0 + \Gamma_1 \cdot \Gamma_2 \\ \Gamma_2 \ni s_{\star,3k+2} &= \sum \Gamma_2 + \Gamma_0 \cdot \Gamma_1 \\ \Gamma_1 \ni s_{\star,3k} &= \sum \Gamma_1 + \Gamma_2 \cdot \Gamma_0 \end{aligned}$$

where $k = 0, 1, \ldots$. Here for example, $\Gamma_1 \cdot \Gamma_2$ means a disturbance generated by an AND of two internal bits from Γ_1 and Γ_2.

Now, let us conduct the divide and conquer attack on Trivium.

3.1 Recover Γ_0

It is important to recover s long enough time instant interval of Γ_0 at first. This part illustrates the components of the deduced linear equations on the initial state bits from Γ_0. Firstly, we consider the time node that the feedback bits begin to enter z_{3k}. Since $s_{i,t}$ is a feedback bit if $t \geq i - j + 1$, where $j \in \{1, 94, 178\}$, we have

$$\begin{aligned} s_{66,3k} &= s_{1,3k-65}, \quad \text{and} \quad 3k - 65 \geq 1 \Rightarrow k \geq 22, \\ s_{162,3k} &= s_{94,3k-68}, \quad \text{and} \quad 3k - 68 \geq 1 \Rightarrow k \geq 23, \\ s_{243,3k} &= s_{178,3k-65}, \quad \text{and} \quad 3k - 65 \geq 1 \Rightarrow k \geq 22. \end{aligned}$$

Thus we know that when $k \in [0, 21]$, there are no feedback bits entering into z_{3k}, where $[a, b]$ means the integer closed set of the time instant interval from a to b.

To make the equations derived from the output bits on Γ_0 subset remain linear for easy solving, we impose conditions on the quadratic terms of the feedback bits as

$$\Gamma_1 \cdot \Gamma_2 = 0,$$

i.e.

$$\begin{aligned} s_{286,3k} \cdot s_{287,3k} &= 0, \quad k = 22, \ldots, \\ s_{91,3k} \cdot s_{92,3k} &= 0, \quad k = 23, \ldots, \\ s_{175,3k} \cdot s_{176,3k} &= 0, \quad k = 22, \ldots. \end{aligned}$$

It results in linear update functions on Γ_0 subset, and the variables in these three update functions, in addition, the variables in the output functions for z_{3k} are still confined in Γ_0 subset. The founding probability of each of the above conditions is about 0.75, since $\Pr\{x \cdot y = 0\} = 0.75$ for the randomness of the keystream. Define the time instant interval of conditions corresponding to each update function of $s_{1,3k+1}, s_{94,3k+1}, s_{178,3k+1}$ as $k \in [0, x_1], [0, x_2], [0, x_3]$, respectively. Then

$$\min\{x_1 + 1, x + 2, x_3 + 1\}$$

linear equations on Γ_0 subset are further obtained since it is deduced from

$$\begin{cases} x_2 + 2 & \text{if } x_2 + 1 \leq x_1 \text{ and } x_2 + 1 \leq x_3 \\ \min\{x_1 + 1, x_3 + 1\} & \text{if } x_2 + 1 < x_1 \text{ or } x_2 + 1 < x_3 \end{cases}.$$

In the subsequent time instants, we start receiving non-linear equations that the linear part only consists of bits from Γ_0 subset, and the non-linear part is a sum of ω AND gates of internal state bits. The probability that the sum of ω AND gates equals 0 is calculated as

$$p_\omega = \sum_{i=0}^{\lfloor \omega/2 \rfloor} \binom{\omega}{2i} 0.75^{\omega - 2i} 0.25^{2i}, \tag{3}$$

via the recursive formula of $p_{\omega+1} = 0.75 p_\omega + 0.25(1 - p_\omega)$ and $p_0 = 1$. Then we are able to transform the subsequent non-linear equations into probabilistic linear equations. Let l_ω be the number of non-linear equations collected with the sum of ω AND gates. Let Δ be the total number of collected probabilistic linear equations and P_Δ be the corresponding probability. We have

$$P_\Delta = \prod p_\omega^{l_\omega}, \text{ and } \Delta = \sum l_\omega, \tag{4}$$

where ω is from 1, and the specified value range is analysed in Section 5. The remaining initial internal state bits in Γ_0 are guessed, and the number of guessing bits is denoted as g. All the linear equations collected above can be transformed into linear equations

Table 2 Structure of the internal state of Trivium

Registers	93	84	111
blocks	66,3,24	69,9,6	66,21,24

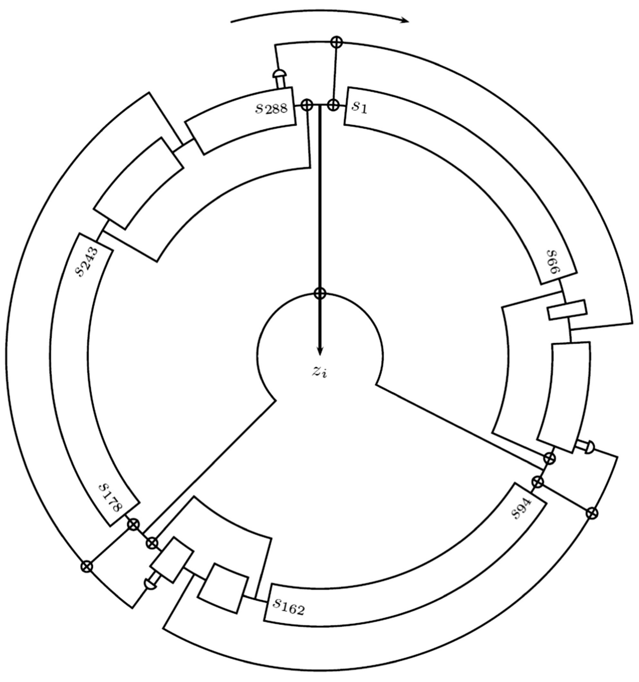

Fig. 1 *TRIVIUM*

on variables of the initial internal state bits in Γ_0, by the linear update functions based on the imposed conditions. It finally requires

$$22 + \min\{x_1 + 1, x + 2, x_3 + 1\} + \Delta + g \geq 96$$

to recover the initial internal state bits in Γ_0.

3.2 Recover Γ_1 *and* Γ_2

Based on the known time instant interval of Γ_0, we continue to recover state bits in Γ_1 and Γ_2. The components of the deduced equations on Γ_1 and Γ_2 are collected partly.

3.2.1 Forward-clock collection: Since variables in Γ_0 up to some time instant are derived, the update functions on Γ_1 and Γ_2, are shown as follows:

$$\begin{aligned}\Gamma_2 \ni s_{\star, 3k+2} &= \sum \Gamma_2 + \Gamma_0 \cdot \Gamma_1,\\ \Gamma_1 \ni s_{\star, 3k} &= \sum \Gamma_1 + \Gamma_2 \cdot \Gamma_0,\end{aligned}$$

are linear on variables both in Γ_1 and Γ_2 two subsets, where $\star \in \{1, 94, 178\}$. Thus here the solving of Γ_1 and Γ_2 cannot be further divided. Now we present the analysis of the time instant interval of derived state bits in Γ_0 in detail, which is missing in the original attack in [16]. For z_{3k+1}

$$\begin{aligned}&s_{66,3k+1} = s_{1,3k-64}, \text{ contains } s_{286,3k-65} \cdot s_{287,3k-65},\\ &\qquad \text{where } \Gamma_0 \ni s_{286,3k-65} = s_{178,3k-173},\\ &\qquad \text{and } 3k - 173 \geq 1 \Rightarrow k \geq 58,\\ &s_{162,3k+1} = s_{94,3k-67}, \text{ contains } s_{91,3k-68} \cdot s_{92,3k-68},\\ &\qquad \text{where } \Gamma_0 \ni s_{91,3k-68} = s_{1,3k-158},\\ &\qquad \text{and } 3k - 158 \geq 1 \Rightarrow k \geq 53,\\ &s_{243,3k+1} = s_{178,3k-64}, \text{ contains } s_{175,3k-65} \cdot s_{176,3k-65},\\ &\qquad \text{where } \Gamma_0 \ni s_{175,3k-65} = s_{94,3k-146},\\ &\qquad \text{and } 3k - 146 \geq 1 \Rightarrow k \geq 49.\end{aligned}$$

The analysis above presents the time instants that the feedback bits in Γ_0 start to enter z_{3k+1}. The analysis process is the same for z_{3k+2}, and the corresponding time instants are not changed. Based on the imposed conditions of the update functions on Γ_0, a subsequent portion of Γ_0 (feedback bits in Γ_0) are derived in Section 3.1, and the time instant interval of linear equations referring to z_{3k+1} and z_{3k+2} can be further extended. In order to collect as many as possible linear equations on Γ_1 and Γ_2, it has to make the time instant intervals arrive at the same time node, i.e.

$$58 + x_3 + 1 = 53 + x_1 + 1 = 49 + x_2 + 1,$$

which derives

$$\begin{cases} x_1 = x_3 + 5,\\ x_2 = x_3 + 9.\end{cases}$$

Moreover as shown in Section 3.1, the time instant interval of the second part of linear equations on Γ_0 subset is

$$\min\{x_1 + 1, x + 2, x_3 + 1\} = x_3 + 1.$$

The number of linear equations collected at forward clocks on Γ_1 and Γ_2 are totally

$$(58 + 1 + x_3) \times 2,$$

and similarly the above linear equations can be transformed into linear equations on variables of the initial internal state bits in Γ_1 and Γ_2 by their linear update functions.

The attack [16] just considered in the linear equations clocking the cipher forward, see Table 5 in [16], and it took $x_3 = 36$. Thus actually only 190 linear equations on Γ_1 and Γ_2 can be collected, rather than the 192 linear equations claimed in [16], which is a little error. The real complexity of the attack in [16] must be higher than that it presented.

3.2.2 Backward-clock collection: We add in a part of linear equations on Γ_1 and Γ_2, considering backward clocks from $t = 0$ [The initial internal state in the attack is not necessary to be the state just accomplished the initialisation stage of TRIVIUM. It can be any time instant for $t = 0$ at keystream generation stage, since the internal state transformation is bijective. Thus the keystream bits generated at the previous time instants are able to be obtained.]. For

$$\begin{aligned}z_{-1} = {} & s_{66,-1} + s_{93,-1}\\ & + s_{162,-1} + s_{177,-1} + s_{243,-1} + s_{288,-1},\end{aligned}$$

we substitute $s_{288,-1}$ with

$$\begin{aligned}s_{288,-1} &= s_{1,0} + s_{243,-1} + s_{286,-1} \cdot s_{287,-1} + s_{69,-1}\\ &= s_{1,0} + s_{244,0} + s_{287,0} \cdot s_{288,0} + s_{70,0},\end{aligned}$$

where $s_{288,0} \in \Gamma_0$ is known, and derive

$$\begin{aligned}z_{-1} = {} & s_{67,0} + s_{94,0} + s_{163,0} + s_{178,0} + s_{244,0}\\ & + (s_{1,0} + s_{244,0} + s_{287,0} \cdot s_{288,0} + s_{70,0}),\end{aligned}$$

which is linear on Γ_1 and Γ_2.

For

$$\begin{aligned}z_{-2} = {} & s_{66,-2} + s_{93,-2}\\ & + s_{162,-2} + s_{177,-2} + s_{243,-2} + s_{288,-2},\end{aligned}$$

we substitute $s_{288,-2}$ with

$$\begin{aligned}s_{288,-2} &= s_{1,-1} + s_{243,-2} + s_{286,-2} \cdot s_{287,-2} + s_{69,-2}\\ &= s_{2,0} + s_{245,0} + s_{288,0} \cdot s_{288,-1} + s_{71,0},\end{aligned}$$

where $s_{288,0} \in \Gamma_0$ is known, and derive

$$\begin{aligned}z_{-2} &= s_{68,0} + s_{95,0} + s_{164,0} + s_{179,0} + s_{245,0}\\ &\quad + (s_{2,0} + s_{245,0} + s_{288,0} \cdot s_{288,-1} + s_{71,0})\\ &= s_{68,0} + s_{95,0} + s_{164,0} + s_{179,0} + s_{2,0} + s_{71,0}\\ &\quad + s_{288,0} \cdot (s_{1,0} + s_{244,0} + s_{287,0} \cdot s_{288,0} + s_{70,0}),\end{aligned}$$

which is still linear on Γ_1 and Γ_2.

However, z_{-3} is no more linear on Γ_1 and Γ_2, since it contains $s_{288,-3}$, which consists of $s_{286,-3} \cdot s_{287,-3}$, i.e., $s_{288,-1} \cdot s_{288,-2}$. Also since then, so many quadratic terms on Γ_1 and Γ_2 begin to show up at backward clocks.

Thus we derive two more linear equations on Γ_1 and Γ_2 at backward clocks [The authors of [14] are well aware that a few linear equations can be derived from backward clocking, but do not give the details.].

3.2.3 Linear transformation of quadratic equations: We introduce a new trick in this part, to transform three imposed quadratic conditions into two linear equations on Γ_1 and Γ_2.

Assume that 190 linear equations on 192 variables of Γ_1 and Γ_2 are obtained. We have the quadratic condition

$$x_1 \cdot x_2 = 0,$$

where x_1 and x_2 are the initial internal state bits in Γ_1 and Γ_2, respectively. Then the remaining 190 variables of the initial internal state bits in Γ_1 and Γ_2 can be linearly expressed by x_1 and x_2.

Take two more quadratic conditions

$$y_1 \cdot y_2 = 0, \text{ and } y_1' \cdot y_2' = 0,$$

where y_1 and y_1', y_2 and y_2' are also the initial internal state bits in Γ_1 and Γ_2. We have

$$\begin{cases} y_1 = a_1x_1 + b_1x_2 + c_1, \\ y_2 = a_2x_1 + b_2x_2 + c_2, \end{cases}$$

$$\begin{cases} y'_1 = a'_1x_1 + b'_1x_2 + c'_1, \\ y'_2 = a'_2x_1 + b'_2x_2 + c'_2. \end{cases}$$

It derives that

$$\begin{cases} y_1 \cdot y_2 = k_0 + k_1x_1 + k_2x_2 + k_3x_1 \cdot x_2 \\ \quad = k_0 + k_1x_1 + k_2x_2 = 0, \\ y'_1 \cdot y'_2 = k'_0 + k'_1x_1 + k'_2x_2 + k'_3x_1 \cdot x_2 \\ \quad = k'_0 + k'_1x_1 + k'_2x_2 = 0, \end{cases}$$

where the notations of a, b, c, k are all the constant coefficients, and the values of k are derived from a, b, c. Thus we get two more linear equations on Γ_1 and Γ_2, transformed from three imposed quadratic conditions. Here it is possible to recover x_1 and x_2, and further to recover the initial internal state bits in Γ_1 and Γ_2.

It is finally required

$$(58 + 1 + x_3) \times 2 + 2 + 2 \geq 192$$

to recover the initial internal state bits in Γ_1 and Γ_2.

An attack against a toy cipher is presented in the Appendix for illustrating the process of the attack.

4 Attack process and framework optimisation

In this section, the attack's framework and its optimisation improved by the time-memory-data tradeoffs are given.

Firstly, we summarise the above divide and conquer attack process as follows:

1. Collect 22 linear equations on Γ_0 subset, given the keystream bits of TRIVIUM without the feedback.
2. Impose conditions

$$\begin{aligned} s_{286,3k} \cdot s_{287,3k} &= 0, \quad k = 22, \ldots, x_3 + 27, \\ s_{91,3k} \cdot s_{92,3k} &= 0, \quad k = 23, \ldots, x_3 + 32, \\ s_{175,3k} \cdot s_{176,3k} &= 0, \quad k = 22, \ldots, x_3 + 22, \end{aligned}$$

   to make the update functions on Γ_0 linear. Further, collect $x_3 + 1$ linear equations on Γ_0 subset given the keystream bits.
3. Collect Δ more linear equations on Γ_0 subset subsequently at forward clocks with the success probability P_Δ.
4. For every guess of the remaining g bits of the initial internal state in Γ_0, derive a time instant interval of the state bits in Γ_0 using the linear equations collected in Steps 1–3.

   (a) Collect $118 + 2x_3$ linear equations on Γ_1 and Γ_2 clocking the cipher forward with known state bits in Γ_0.
   (b) Collect 2 linear equations on Γ_1 and Γ_2 clocking the cipher backward.
   (c) Derive 2 linear equations on Γ_1 and Γ_2 by transforming three imposed quadratic conditions above.
   (d) Recover the internal state bits in Γ_1 and Γ_2 by any linear solving technique in some fixed time, and verify the solution in time O(1).
5. Repeat the loops in Steps 1–4 until the right internal state is found, given the sliding windows of the keystream.

It should be noted that the wrong guesses, the failure conditions and the probabilistic equations all may lead to the incorrect solutions.

We further analyse the coefficient matrices form of the linear equation systems on $\mathbf{\Gamma}_0$ and $\mathbf{\Gamma}_1, \mathbf{\Gamma}_2$. The linear equation system on $\mathbf{\Gamma}_0$ can be simplified in an abstract form of

$$\boldsymbol{C} \cdot \mathbf{\Gamma}_0 = \boldsymbol{Z}^0,$$

where $\boldsymbol{C}$ is a constant matrix derived from the linear equation system, and $\boldsymbol{Z}^0$ denotes a vector of keystream bits with length $(96 - g)$, that each component of $\boldsymbol{Z}^0$ is certain z_{3k}. Here, $\boldsymbol{G}_0$ denotes the initial internal state variables in $\mathbf{\Gamma}_0$. Then the initial internal state bits in $\mathbf{\Gamma}_0$ can be derived by Gaussian elimination as

$$\mathbf{\Gamma}_0 = \boldsymbol{C}^{-1} \cdot \boldsymbol{Z}^0,$$

where $\boldsymbol{C}^{-1}$ denotes the inverse matrix of $\boldsymbol{C}$ Given the keystream vectors $\boldsymbol{Z}^1$ and $\boldsymbol{Z}^2$, which are defined the same as $\boldsymbol{Z}^0$ with z_{3k+2} and z_{3k+1} of length $|\boldsymbol{Z}^1| + |\boldsymbol{Z}^2| = 190$ in total, the linear equation system on $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ is in the form of

$$(\boldsymbol{Z}^0)\begin{pmatrix} \mathbf{\Gamma}_1 \\ \mathbf{\Gamma}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{Z}^1 \\ \boldsymbol{Z}^2 \\ \mathbf{0} \end{pmatrix},$$

where $(\boldsymbol{Z}^0)$ denotes a matrix that each element may consist of a combination of certain bits in $\boldsymbol{Z}^0$, imported from the update functions on $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$, $\mathbf{0}$ denotes a two dimensional zero vector. Here, $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ denote the initial internal state variables in $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$. It is derived that

$$\begin{pmatrix} \mathbf{\Gamma}_1 \\ \mathbf{\Gamma}_2 \end{pmatrix} = (\boldsymbol{Z}^0)^{-1}\begin{pmatrix} \boldsymbol{Z}^1 \\ \boldsymbol{Z}^2 \\ \mathbf{0} \end{pmatrix},$$

where $(\boldsymbol{Z}^0)^{-1}$ denotes the inverse matrix of $(\boldsymbol{Z}^0)$. It is hard to derive $(\boldsymbol{Z}^0)^{-1}$ in formalisation of symbols on $\boldsymbol{Z}^0$. That is, $\mathbf{\Gamma}_0$ should be linearly solved at first with certain $\boldsymbol{Z}^0$, and then the matrix $(\boldsymbol{Z}^0)$ can be derived by substituting the variables of $\mathbf{\Gamma}_0$ in the update functions on $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ with the values of $\mathbf{\Gamma}_0$ for each guess. Thus the matrix $(\boldsymbol{Z}^0)$ is of specific values, and then the inverse matrix $(\boldsymbol{Z}^0)^{-1}$ can be derived with some linear technique. The coefficient matrix of linear equation systems on $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ is symbolic changing with the keystream, rather than numerical. Thus we doubt the correctness to estimate the complexity of solving equation system with $c = 2^{16}$ as claimed in [16], for $192^{2.808} = 2^{21.298}$ if the Strassen's algorithm is used for computing the solutions of the linear equation system, which is applied most often.

To further reduce the complexity, we optimise the attack with time-memory-date tradeoffs. For each guess of the internal state bits in $\mathbf{\Gamma}_0$, the coefficient matrix of the corresponding linear equation system on $\mathbf{\Gamma}_0$ subset can be derived, and further the expression of each variable in $\mathbf{\Gamma}_0$ with $\boldsymbol{Z}^0$ is able to be deduced. A random portion [The windows in the keystream where the founding probability of the imposed conditions is larger than the random should be considered at first, which can be obtained from the tests. For example, the probability of the condition that a set of specific AND gates is zero is larger than otherwise, when the keystream is a zero sequence. In this way, the time and data complexities can be both further reduced.] of $\{\boldsymbol{Z}^0\}$ with a size of 2^{96-d-g} is chosen to precompute the value of $\mathbf{\Gamma}_0$ and the coefficient matrices $(\boldsymbol{Z}^0)$ of the linear equation systems on $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$. With the Strassen's algorithm, the inverse matrix $(\boldsymbol{Z}^0)^{-1}$ can be deduced offline. Store these $(\boldsymbol{Z}^0)^{-1}$ in order to invoke them online directly. Meanwhile,

Table 3 Derivation of time instants for quadratic terms entering

Tap for z_t	Feedback position	k that feedback bits begin to enter z_{3k}	Instant interval of imposed conditions ($k \in$)	k that z_{3k} begins to contain quadratic terms
66	1	22	$[0, x_3 + 5]$	$x_3 + 28$
93	1	31	$[0, x_3 + 5]$	$x_3 + 37$
162	94	23	$[0, x_3 + 9]$	$x_3 + 33$
177	94	28	$[0, x_3 + 9]$	$x_3 + 38$
243	178	22	$[0, x_3]$	$x_3 + 23$
288	178	37	$[0, x_3]$	$x_3 + 38$

Table 4 The probability that the sum of ω AND gates equals 0

ω	1	2	3	4	6
p_ω	0.75	0.625	0.5625	0.53125	0.5078125

Table 5 Derivation for number of quadratic terms

Time instants ($x_3 +$)	Number of time instants	Taps contain quadratic terms	Number of quadratic terms
23,24,25,26,27	5	243	1
28,29,30,31,32	5	243,66	2
33,34,35,36	4	243,66,162	3
37	1	243,66,162,93	4
38, …	—	243,66,162,93,177,288	6

Table 6 Values of Δ and P_Δ

Δ	$-\log_2 P_\Delta$	Δ	$-\log_2 P_\Delta$	Δ	$-\log_2 P_\Delta$
1	0.415	8	4.109	15	9.697
2	0.830	9	4.787	16	10.675
3	1.245	10	5.465	17	11.652
4	1.66	11	6.295	18	12.629
5	2.075	12	7.125	19	13.606
6	2.753	13	7.955	—	—
7	3.431	14	8.785	—	—

linear expressions of $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ with $\mathbf{Z}^1$ and $\mathbf{Z}^2$ can be obtained. Thus a verification system consisting of the remaining $(3x_3 + 17 - 3)$ imposed quadratic conditions in the form of $\mathbf{\Gamma}_1 \cdot \mathbf{\Gamma}_2 = 0$ can be expressed with $\mathbf{Z}^1$ and $\mathbf{Z}^2$ immediately.

At the online stage, observe the keystream and search for $(96 - g)$-bit strings of z_{3k}s that matches with some precomputed $\mathbf{Z}^0$. For one such string, let the corresponding vectors of z_{3k+2}s and z_{3k+1}s in the keystream be $\mathbf{Z}^1$ and $\mathbf{Z}^2$. The vectors are directly checked by the above verification system on $\mathbf{Z}^1$ and $\mathbf{Z}^2$ in time O(1) for each guessing mode. If they pass the verification, we further do the operation of matrix multiplication as

$$(\mathbf{Z}^0)^{-1}\begin{pmatrix}\mathbf{Z}^1\\ \mathbf{Z}^2\end{pmatrix}$$

with the stored corresponding $(\mathbf{Z}^0)^{-1}$ to derive $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$, while the value of $\mathbf{\Gamma}_0$ is already precomputed. The advantage of the optimisation is that the imposed conditions are further used to filter out the keystream bits unsatisfied, before doing the dominant cost operations of matrix multiplication, while these imposed conditions are not used at all in [16], since they are quadratic and hard to be solved with low complexity.

5 Complexity analysis

We present specific attack parameters and the corresponding complexity of the improved guess-and-determine attack on Trivium in this section. In addition, comparisons with the complexities of different cases and the key exhaustive search on Trivium are given.

5.1 Parameter selection

We first provide a detailed analysis of parameter selection.

5.1.1 How to determine Δ: In order to determine the parameter Δ and the corresponding probability P_Δ, we should analyse the number of quadratic terms shown up in the subsequent keystream bits z_{3k} from $k \geq 23 + x_3$. We first derive the time instants that each tap in the output function begins to have quadratic terms. The derivation is shown in Table 3. Then we are able to derive the number of quadratic terms shown up in the subsequent keystream bits, and the corresponding time instants. The derivation is shown in Table 3.

Next, we focus on the corresponding probability. Based on the formula of p_ω via (3), we have seen in Tables 4 and 5.

Moreover, we derive P_Δ according to Δ with its formula (4), shown in Table 6.

5.1.2 Other parameters: To get enough linear equations for solving initial internal state variables in $\mathbf{\Gamma}_0$, we have

$$23 + x_3 + \Delta + g \geq 96.$$

Similarly, to get enough linear equations for recovering $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$, we have

$$2x_3 + 118 + 2 + 2 \geq 192.$$

That is

$$\begin{cases} x_3 + g + \Delta \geq 73, \\ x_3 \geq 35. \end{cases}$$

5.2 Complexity analysis

The data complexity of the attack consists of three parts. Since $(3x_3 + 17)$ conditions each with founding probability of 0.75 are imposed, it expects around $0.75^{-(3x_3+17)}$ length of the keystream to satisfy the conditions. For the Δ probabilistic linear equations, it requires to increase the length of the keystream by the ratio $1/P_\Delta$ to make the success of the equations establishment. A much longer keystream with a length of 2^d is required for a match with the precomputed $\mathbf{Z}^0$. Thus the data complexity is about

$$0.75^{-(3x_3+17)} \cdot P_\Delta^{-1} \cdot 2^d.$$

The time complexity is as

$$O\big(0.75^{-(3x_3+17)} \cdot P_\Delta^{-1} \cdot 2^g\big),$$

since the attack is conducted with the equation solving for each guess, and if it fails for all the 2^g guesses, a new window of keystream is chosen for another iteration, and the number of iterations is according with the success probability of the imposed conditions and probabilistic linear equations. For each guess, the verification of the corresponding keystream vectors $\mathbf{Z}^1$ and $\mathbf{Z}^2$ is done first, and the passing rate is 0.75^{3x_3+14}, since the verification system is composed by the remaining $(3x_3 + 14)$ quadratic conditions in the form of $x \cdot y = 0$, each with a founding probability of 0.75. This makes full use of all the conditions of degree 2, while they are not used at all in the original attack [16]. If the keystream vectors pass the verification, the linear equation system on $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ is solved by a matrix multiplication with the inverse matrix precomputed at the offline stage, and it calls for a complexity of $192^2 = 2^{15.17}$. Finally, the time complexity of the improved attack is

$$\begin{aligned} &0.75^{-(3x_3+17)} \cdot P_\Delta^{-1} \cdot 2^g \\ &\quad + 0.75^{-(3x_3+17)} \cdot P_\Delta^{-1} \cdot 2^g \cdot 0.75^{3x_3+14} \cdot 2^{15.17}, \end{aligned}$$

which consists of a complexity for filtering the keystreams and a complexity for linear equations solving with the selected keystreams. The complexity of precomputation is dominant by the calculations of the inverses to the coefficient matrices of the linear equation systems on $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$, according to a number of 2^{96-d-g} chosen $\mathbf{Z}^0$ s. It calls for

$$2^{96-d-g} \cdot 2^g \cdot 2^{21.298} = 2^{96-d} \cdot 2^{21.298}$$

operations referring to the 2^g guesses, since each inverse matrix computing costs around $192^{2.808} = 2^{21.298}$.

The memory complexity is dominant by the storage of the above precomputed inverse matrices, which calls for

$$2^{96-d} \cdot 2^{15.17-1}$$

an average. Since the matrices may be sparse, the memory complexity can be much lower.

5.3 Several results

The main idea is using the complexity bounds to control the parameter selection. Since TRIVIUM is designed to generate up to 2^{64} bits of keystream from a pair of an 80-bit secret key and an 80-bit IV, the condition for data complexity can be simplified as

$$D: (3x_3 + 17) \cdot (-\log_2 0.75) - \log_2 P_\Delta + d \leq 64,$$

Although the security level of TRIVIUM is 2^{80}, we believe that a key exhaustive search will require much more time, $\gamma \cdot 2^{80}$, where γ is the initialisation time of the cipher that includes 1152 iterations to be done before the first keystream bit is produced, and can be estimated as $O(1152) \simeq O(2^{10.17})$ operations for one key guess. Thus an exhaustive search on TRIVIUM would require around

$$O(2^{90.17})$$

operations, which can be identified as the time and memory bound of the attack. We simplify the conditions for precomputation complexity and memory complexity as follows

$$\begin{aligned} P&: 96 - d + 21.298 \leq 90.17, \\ M&: 96 - d + 14.17 \leq 90.17. \end{aligned}$$

For the time complexity, the condition is in the form of

$$\begin{aligned} T:\ \max \{&(3x_3 + 17) \cdot (-\log_2 0.75) - \log_2 P_\Delta + g, \\ &3(-\log_2 0.75) - \log_2 P_\Delta + g + 15.17\} \leq 90.17. \end{aligned}$$

We substitute $g = 73 - x_3 - \Delta$ into the condition for time complexity, and have

$$\begin{aligned} &(3x_3 + 17) \cdot (-\log_2 0.75) - \log_2 P_\Delta + g \\ &= 80.055 + 0.245x_3 - \log_2 P_\Delta - \Delta. \end{aligned}$$

It is obvious that the value of the above function is reducing with the decreasing of x_3, since $(-\log_2 P_\Delta - \Delta)$ is decreasing with x_3. Thus x_3 is taken the minimise value as

$$x_3 = 35.$$

In Case 1, the data complexity is bounded by 2^{64}, and the time-memory-data tradeoffs are not taken into consideration. We have

$$-\log_2 P_\Delta \leq 13.37$$

according to the data restriction. Thus it takes $-\log_2 P_\Delta = 12.629$ with $\Delta = 18$, and derives

$$\log_2 D = 63.259.$$

Then it deduces $g = 20$, and the time complexity is derived as

$$\log_2 O(T) = 83.259.$$

It is less than the complexity of $c \cdot 2^{83.572}$ claimed in [16], which must be larger actually for the pointed errors in Section 3.2. To add up with the complexity of solving equations, the complexity in [16] is around $2^{104.87}$, since it calls for at least $2^{21.298}$ to solve the equations online, rather than the claimed 2^{16}.

In Case 2, the data complexity is bounded by 2^{64}, and the time-memory-data tradeoffs are used. Here, P_Δ takes 0, so does the Δ. The maximum d is 13 according to the data restriction, and we have

$$\log_2 D = 63.63.$$

In this way, the precomputation and memory complexity are higher than the exhaustive search bound, respectively as

$$\log_2 P = 104.298, \text{ and} \log_2 M = 97.17.$$

Then it derives $g = 38$, and the time complexity is about

$$\log_2 T = 88.63.$$

Table 7 Comparison of the complexities in cases

Attack	Case 1	Case 2	Case 3
x_3	35	35	35
$(\Delta, -\log_2 P_\Delta)$	(18,12.629)	(0,0)	(0,0)
g	20	38	38
d	—	13	29
D	$2^{63.259}$	$2^{63.63}$	$2^{79.63}$
P	—	$2^{104.298}$	$2^{88.298}$
M	—	$2^{97.17}$	$2^{81.17}$
T	$2^{83.259+21.298}$	$2^{88.63}$	$2^{88.63}$

In Case 3, we relax the requirement on the data to the security level of Trivium, 2^{80}. Then a borderline result that all the complexities are under the exhaustive search bound can be obtained. It takes $P_\Delta = 0$, so does the $\Delta = 0$. It takes $d = 29$ for the relaxing data bound, and then $g = 38$. The data complexity is around

$$\log_2 D = 79.63.$$

The precomputation complexity is

$$\log_2 P = 88.298,$$

and the memory complexity is

$$\log_2 M = 81.17.$$

The time complexity is about

$$\log_2 T = 88.63.$$

The complexities are all under $2^{90.17}$. This result also shows the necessity of data requirement less than 2^{64} imposed on Trivium, which is questioned as an open problem in the Trivium specifications [1]. A comparison of the complexities in above cases are shown in Table 7.

6 Conclusion

In this paper, we present a guess-and-determine attack on Trivium. Compared with the original attack in [16], which is a basis, our result is improved. We provide more technique details for better comprehension, further consider in more tricks to improve the attack. In the view of the overall framework, we combine with the time-memory-data tradeoffs to make full use of the quadratic conditions, and give more attack scenarios. The parameter selection and complexity deduction are analysed at length. Our work provides foundation for further research against the keystream generation phase of Trivium, which is quite few up to now.

7 Acknowledgment

This work was supported by the programs of the National Natural Science Foundation of China (grant no. 61772517) and the Young Elite Scientists Sponsorship Program by CAST (2016QNRC001).

8 References

[1] De Canniere, C., Preneel, B.: 'Trivium specifications'. Report 2005/030, eSTREAM. ECRYPT Stream Cipher Project, 2005

[2] International Organization for Standardization (ISO): 'ISO/IEC 29192-3:2012, information technology – security techniques – lightweight cryptography – part 3: stream ciphers', 2012

[3] Englund, H., Johansson, T., Turan, M.S.: 'A framework for chosen IV statistical analysis of stream ciphers'. Progress in Cryptology – INDOCRYPT 2007, 8th Int. Conf. on Cryptology in India, Chennai, India, 9–13 December 2007, pp. 268–281, Proceedings

[4] Fischer, S., Khazaei, S., Meier, W.: 'Chosen IV statistical analysis for key recovery attacks on stream ciphers'. Progress in Cryptology – AFRICACRYPT 2008, Casablanca, Morocco, 2008

[5] Knellwolf, S., Meier, W., Naya-Plasencia, M.: 'Conditional differential cryptanalysis of Trivium and KATAN'. Int. Workshop on Selected Areas in Cryptography, Toronto, ON, Canada, 2011, pp. 200–212, 236–245

[6] Dinur, I., Shamir, A.: 'Cube attacks on tweakable black box polynomials'. Advances in Cryptology-EUROCRYPT2009, Cologne, Germany, 2009 (LNCS, **5479**), pp. 278–299

[7] Aumasson, J., Dinur, I., Meier, W., *et al.*: 'Cube testers and key recovery attacks on reduced-round MD6 and Trivium'. Fast Software Encryption, 16th Int. Workshop, FSE 2009, Leuven, Belgium, 2009, pp. 1–22

[8] Fouque, P., Vannet, T.: 'Improving key recovery to 784 and 799 rounds of Trivium using optimized cube attacks'. Fast Software Encryption-20th Int. Workshop, FSE 2013, Singapore, 11–13 March 2013, pp. 502–517. Revised Selected Papers

[9] Todo, Y., Isobe, T., Hao, Y., *et al.*: 'Cube attacks on non-blackbox polynomials based on division property'. Advances in Cryptology – CRYPTO 2017, Santa Barbara, CA, USA, 20–24 August 2017, pp. 250–279, Proceedings, Part III

[10] Liu, M., Yang, J., Wang, W., *et al.*: 'Correlation cube attacks: from weak-key distinguisher to key recovery'. Advances in Cryptology – EUROCRYPT 2018, Tel Aviv, Israel, 29 April – 3 May 2018, pp. 715–744. Proceedings, Part II

[11] Fu, X., Wang, X., Dong, X., *et al.*: 'A key-recovery attack on 855-round Trivium authors'. Advances in Cryptology – CRYPTO 2018, Santa Barbara, CA, USA, 2018, pp. 160–184

[12] Wang, Q., Hao, Y., Todo, Y., *et al.*: 'Improved division property based cube attacks exploiting algebraic properties of superpoly'. Advances in Cryptology – CRYPTO 2018, Santa Barbara, CA, USA, 2018, pp. 275–305

[13] eSTREAM Discussion Forum: 'A reformulation of Trivium created on 02/24/06 12:52PM', 2005

[14] Raddum, H.: 'Cryptanalytic results on Trivium'. Report 2006/039, eSTREAM, ECRYPT Stream Cipher Project, 2006

[15] Khazaei, S., Hasanzadeh, M.M., Kiaei, M.S.: 'Linear sequential circuit approximation of grain and Trivium stream ciphers'. Report 2005/063, eSTREAM, ECRYPT Stream Cipher Project, 2005

[16] Maximov, A., Biryukov, A.: 'Two trivial attacks on Trivium'. Selected Areas in Cryptography 2007, Ottawa, Canada, 2007, pp. 36–55

[17] Babbage, S., De Canniere, C., Lano, J.: 'Cryptanalysis of SOBER-t32'. FSE 2003. Heidelberg: Springer-Verlag, Lund, Sweden, 2003 (LNCS, **2887**), pp. 111–128

[18] Feng, X.T., Liu, J., Zhou, Z.C., *et al.*: 'A byte-based guess and determine attack on SOSEMANUK'. ASIACRYPT 2010, Singapore, 2010 (LNCS, **6477**), pp. 146–157

[19] Golić, J.: 'Cryptanalysis of alleged A5 stream cipher'. EUROCRYPT, 1997, Germany, 1997, (LNCS, **1233**), pp. 239–255

[20] Hawkes, P., Rose, G.G.: 'Guess-and-Determine attacks on SNOW'. SAC 2002, Newfoundland, Canada, 2002 (LNCS, **2595**), pp. 37–46

[21] Mattsson, J.: 'A guess-and-determine attack on the stream cipher polar bear'. eSTREAM report 2006/017, 2006, pp. 5–10

9 Appendix

In this part, we present some experimental results of the Trivium-like toy cipher to support our theoretical estimation.

The toy cipher contains a 72-bit internal state, and works as follows:

$$\begin{aligned} z_t &= s_{15,t} + s_{24,t} + s_{36,t} + s_{45,t} + s_{60,t} + s_{72,t}, \\ s_{i,t+1} &= s_{i-1,t}, i \neq 1, 25, 46, \\ s_{1,t+1} &= s_{60,t} + s_{72,t} + s_{70,t} \cdot s_{71,t} + s_{18,t}, \\ s_{25,t+1} &= s_{15,t} + s_{24,t} + s_{22,t} \cdot s_{23,t} + s_{39,t}, \\ s_{46,t+1} &= s_{36,t} + s_{45,t} + s_{43,t} \cdot s_{44,t} + s_{66,t}. \end{aligned}$$

To recover enough long time instant interval of Γ_0, we directly obtain four linear equations on $s_{3k,0}$ from z_0, z_3, z_6, z_9, like

$$z_0 = s_{15,0} + s_{24,0} + s_{36,0} + s_{45,0} + s_{60,0} + s_{72,0}.$$

From $t = 12$, the feedback bits begin to enter the outputs, such as

$$\begin{aligned} z_{12} &= s_{3,0} + s_{12,0} + s_{25,1} + s_{33,0} + s_{48,0} + s_{60,0} \\ &= s_{3,0} + s_{12,0} + (s_{15,0} + s_{24,0} + s_{22,0} \cdot s_{23,0} + s_{39,0}) \\ &\quad + s_{33,0} + s_{48,0} + s_{60,0}. \end{aligned}$$

Thus, we have to impose 31 conditions on the AND terms to remain the equations linear. They are

$$s_{70,0}\cdot s_{71,0}=0, s_{70,3}\cdot s_{71,3}=0,\ldots,s_{70,30}\cdot s_{71,30}=0,$$
$$s_{22,0}\cdot s_{23,0}=0, s_{22,3}\cdot s_{23,3}=0,\ldots,s_{22,30}\cdot s_{23,30}=0,$$
$$s_{43,0}\cdot s_{44,0}=0, s_{43,3}\cdot s_{44,3}=0,\ldots,s_{43,24}\cdot s_{44,24}=0.$$

Then we collect more linear equations on Γ_0 from $z_{12}, z_{15},\ldots,z_{39}$. For

$$\begin{aligned}z_{39} &= s_{1,25}+s_{1,16}+s_{25,28}+s_{25,19}+s_{46,25}+s_{46,13}\\ &= (s_{60,24}+s_{72,24}+s_{70,24}\cdot s_{71,24}+s_{18,24})\\ &\quad +(s_{60,15}+s_{72,15}+s_{70,15}\cdot s_{71,15}+s_{18,15})\\ &\quad +(s_{15,27}+s_{24,27}+s_{22,27}\cdot s_{23,27}+s_{39,27})\\ &\quad +(s_{15,18}+s_{24,18}+s_{22,18}\cdot s_{23,18}+s_{39,18})\\ &\quad +(s_{36,24}+s_{45,24}+s_{43,24}\cdot s_{44,24}+s_{66,24})\\ &\quad +(s_{36,12}+s_{45,12}+s_{43,12}\cdot s_{44,12}+s_{66,12})\end{aligned}$$

and it will contain quadratic term $s_{43,27}\cdot s_{44,27}$ in z_{42}. For clarity, we directly guess 10 bits of $s_{3,0}, s_{6,0},\ldots,s_{30,0}$ without considering the probabilistic equations and solve the above system of 14 linear equations to recover $s_{33,0}, s_{36,0},\ldots,s_{72,0}$, moreover, deduce $s_{1,1}, s_{1,4},\ldots,s_{1,31}, s_{25,1}, s_{25,4},\ldots,s_{25,31}$, and $s_{46,1}, s_{46,4},\ldots,s_{46,25}$.

Next, to recover the state bits from $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$, we run the cipher forwards, and directly get four linear equations like

$$z_1 = s_{15,1}+s_{24,1}+s_{36,1}+s_{45,1}+s_{60,1}+s_{72,1}.$$

Then, since $s_{21,0}$ is derived

$$\begin{aligned}z_{13} &= s_{2,0}+s_{11,0}+s_{25,2}+s_{32,0}+s_{47,0}+s_{59,0}\\ &= s_{2,0}+s_{11,0}+(s_{14,0}+s_{23,0}+s_{21,0}\cdot s_{22,0}+s_{38,0})\\ &\quad +s_{32,0}+s_{47,0}+s_{59,0}\end{aligned}$$

is still linear. For

$$\begin{aligned}z_{34} &= s_{1,20}+s_{1,11}+s_{25,23}+s_{25,14}+s_{46,20}+s_{46,8}\\ &= (s_{60,19}+s_{72,19}+s_{70,19}\cdot s_{71,19}+s_{18,19})\\ &\quad +(s_{60,10}+s_{72,10}+s_{70,10}\cdot s_{71,10}+s_{18,10})\\ &\quad +(s_{15,22}+s_{24,22}+s_{22,22}\cdot s_{23,22}+s_{39,22})\\ &\quad +(s_{15,13}+s_{24,13}+s_{22,13}\cdot s_{23,13}+s_{39,13})\\ &\quad +(s_{36,19}+s_{45,19}+s_{43,19}\cdot s_{44,19}+s_{66,19})\\ &\quad +(s_{36,7}+s_{45,7}+s_{43,7}\cdot s_{44,7}+s_{66,7})\\ &= (s_{46,5}+s_{53,0}+s_{51,0}\cdot s_{52,0}+s_{1,2})\\ &\quad +(s_{50,0}+s_{62,0}+s_{60,0}\cdot s_{61,0}+s_{8,0})\\ &\quad +(s_{1,8}+s_{2,0}+s_{1,1}\cdot s_{1,0}+s_{25,8})\\ &\quad +(s_{2,0}+s_{11,0}+s_{9,0}\cdot s_{10,0}+s_{26,0})\\ &\quad +(s_{25,8}+s_{26,0}+s_{25,1}\cdot s_{25,0}+s_{47,0})\\ &\quad +(s_{29,0}+s_{38,0}+s_{36,0}\cdot s_{37,0}+s_{59,0}),\end{aligned}$$

$s_{1,1}$ and $s_{25,1}$ are still derived, so it remains linear. Such linear equations can be collected until z_{64}, z_{65}, and we obtain 44 linear equations forwards. To run backwards, we get two linear equations from z_{-1}, and z_{-2}, like

$$z_{-1} = s_{16,0}+s_{25,0}+s_{37,0}+s_{46,0}+s_{1,0}+s_{71,0}\cdot s_{72,0}+s_{19,0}$$

where $s_{72,0}$ is known. Further, we derive two linear equations from three imposed conditions as illustrated formerly. We solve the system of 48 linear equations to recover the whole state of the toy cipher, and verify its correctness.

We simulate the attack process, and it nearly takes 2^{23} guess and determine iterations to derive the correct internal state, which supports our theoretical estimation.