ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/220333274

Two Trivial Attacks on Trivium

Conference Paper · August 2007

DOI: 10.1007/978-3-540-77360-3_3 · Source: DBLP

CITATIONS
79

READS
374

2 authors:

Alexander Maximov
Ericsson
11 PUBLICATIONS **727** CITATIONS
SEE PROFILE

Alex Biryukov
University of Luxembourg
161 PUBLICATIONS **7,096** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project Functional Decomposition and S-Box Reverse-Engineering View project

Project Proofs of Work and Memory Hard functions View project

All content following this page was uploaded by Alexander Maximov on 20 May 2014.

The user has requested enhancement of the downloaded file.

Two Trivial Attacks on Trivium

Alexander Maximov and Alex Biryukov

Laboratory of Algorithmics, Cryptology and Security
University of Luxembourg
6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg
e-mail: movax@mail.ru, alex.cryptan@gmail.com

Abstract

Trivium is a stream cipher designed in 2005 by C. De Cannière and B. Preneel for the European project eSTREAM. It has successfully passed the first phase of the project and has been selected for a special focus in the second phase for the hardware portfolio of the project. Trivium has an internal state of size 288 bits and the key of length 80 bits. Although the design has a simple and elegant structure, no attack on it has been found yet.

In this paper we study a class of Trivium-like designs. We propose a set of techniques that one can apply in cryptanalysis of such constructions. The first group of methods is for recovering the internal state and the secret key of the cipher, given a piece of a known keystream. Our attack is more than 2^{30} faster than the best known attack. Another group of techniques allows to gather statistics on the keystream, and to build a distinguisher.

We study two designs: the original design of Trivium and a truncated version Bivium, which follows the same design principles as the original. We show that the internal state of the full Trivium can be recovered in time around $c \cdot 2^{83.5}$, and for Bivium this complexity is $c \cdot 2^{36.1}$. These are the best known results for these ciphers. Moreover, a distinguisher for Bivium with working time 2^{32} is presented, the correctness of which has been verified by simulations.

1 Introduction

Additive stream ciphers are an important class of data encryption primitives, in which the process of encryption simulates the one-time-pad. The core of any stream cipher is its *pseudo-random keystream generator* (PRKG). It is initialized with a *secret key* K, and an *initial value* (IV). Afterwards, it produces a long *pseudo-random sequence* called *keystream* $\mathbf{u}$. In the encryption procedure, the ciphertext $\mathbf{c}$ is then obtained by a bitwise xor of the message $\mathbf{m}$ and the keystream $\mathbf{u}$, i.e., $\mathbf{c} = \mathbf{m} \oplus \mathbf{u}$.

Many stream ciphers are currently used in various aspects of our life. To mention some of them, they are: RC4 [Sma03] (is used on the Internet), E_0 [Blu03] (in Bluetooth), A5/1 [BGW99] (in GSM communication), and others. However, it has been shown that these primitives are susceptible to various kinds of weaknesses and attacks [FM00, MS01, LV04, LMV05, BSW00, MJB04]. In 1999 the European project NESSIE was launched [NES99] and among other encryption and signature primitives it attempted to select stream ciphers for its final portfolio. However after a few rounds of evaluation and cryptanalysis, most of the proposals were broken[1]. As a result the board of the project NESSIE could not select any of the stream cipher proposals for its final portfolio.

The recent European project ECRYPT [ECR05] has started in 2004 within the Sixth Framework Programme (FP6). It announced a new call for stream cipher proposals, for its subproject eSTREAM. In the first phase 34 proposals were received, but only a few of them got the status of "focused" algorithms in the second phase. In the *hardware portfolio* only four new designs are in focus, they are: TRIVIUM [CP05], Grain [HJM05], Mickey [BD05], and Phelix [WSLM05].

In this paper we analyze one of these designs – TRIVIUM. The stream cipher TRIVIUM was proposed in 2005 for the project eSTREAM by C. De Canniére and B. Preneel [CP05]. It has an internal state of 288 bits and the key of 80 bits. Though the cipher was designed for hardware implementation it is also very fast in software, which makes it one of the most attractive candidates of the competition. The structure of the cipher is elegant and simple, and it follows clearly described design principles. After the design was announced many cryptographers tried to analyze it. However, only two results on TRIVIUM are known so far.

The first known result is actually given on the eSTREAM discussion forum [eDF05] where the complexity to recover the internal state from given keystream is argued to be 2^{135}. The second result is a paper from H. Raddum [Rad06], where a new algorithm for solving nonlinear systems of equations is proposed and applied on TRIVIUM. The attack complexity found was 2^{164}. Two reduced versions of this design, BIVIUM -A and -B, were proposed in that paper as well. The first reduced version was broken "in about one day", whereas the second version required time of around 2^{56} *seconds.*

In this paper we consider the design of TRIVIUM in general, and as examples we consider two instances: the original design of TRIVIUM and a reduced version BIVIUM, the one given in [Rad06] under the name BIVIUM-B. We propose a set of techniques to analyse this class of stream ciphers, and show how its internal state can be recovered given the keystream. The complexity of this attack determines the upper bound for the security level of the cipher. Its complexities for TRIVIUM and BIVIUM are found to be $c \cdot 2^{83.5}$ and $c \cdot 2^{36.1}$, respectively, where c is the complexity of solving a sparse system of linear equations (192 for TRIVIUM and 118 for BIVIUM). It means that, for example, the secret key

[1] There was a discussion at NESSIE on whether a distinguishing attack of very high complexity qualifies as a break of a cipher.

cannot be increased to 128 bits in a straightforward way unless the design in general is changed. This time complexity is much better than in [eDF05] and [Rad06], and is the best known result on Trivium so far.

In the second attack linear statistical methods are applied. We show how a distinguisher can be built, and propose a linear distinguishing attack on Bivium with less than 2^{32} operations in total. This attack was implemented and in practice works even slightly better than expected.

This paper is organized as follows. In Section 2 we define the structures of Trivium and Bivium. Afterwards, in Section 3, we give methods for a *state recovering attack*, and propose a set of attack scenarios for both Trivium and Bivium. A linear distinguishing attack is given in Section 4. The paper ends with the summary of our results and conclusions.

1.1 Notation

In this paper we accept the following notation. A single bit will commonly be denoted by $x_i^{(t)}$, where i is an index of a variable, and t is the time instance. Bold symbols $\mathbf{u}$ represent a stream or a vector of bit-oriented data $u_1, u_2, \ldots$. Let us also define *triple-clock* of a cipher as just three consecutive clocks of it.

2 Bivium and Trivium

In Figure 1 two classes of stream ciphers are shown, namely, Bivium and Trivium.

The number of basic components is two or three, respectively. Each basic component (a register) consist of *three blocks, each of size divisible by 3.* An instance of this class is a *specification vector* with the blocks' sizes specified, i.e.,

$$\begin{aligned} \textsc{Bivium} &\Rightarrow \textsc{Bi}(A_1, A_2, A_3; B_1, B_2, B_3), \\ \textsc{Trivium} &\Rightarrow \textsc{Tri}(A_1, A_2, A_3; B_1, B_2, B_3; C_1, C_2, C_3). \end{aligned} \tag{1}$$

Notation on the registers is summarized in Table 1.

Reg	total length	cells denoted	the AND gate	In:Out	Res
R_A	$A = A_1 + A_2 + A_3$	$a_0^{(t)}, \ldots, a_{A-1}^{(t)}$	$a_{A-3}^{(t)} \cdot a_{A-2}^{(t)}$	$p_t : q_t$	x_t
R_B	$B = B_1 + B_2 + B_3$	$b_0^{(t)}, \ldots, b_{B-1}^{(t)}$	$b_{B-3}^{(t)} \cdot b_{B-2}^{(t)}$	$q_t : p_t/r_t$	y_t
R_C	$C = C_1 + C_2 + C_3$	$c_0^{(t)}, \ldots, c_{C-1}^{(t)}$	$c_{C-3}^{(t)} \cdot c_{C-2}^{(t)}$	$r_t : p_t$	z_t

Table 1: The structure of the internal state's registers.

At any time t, the keystream bits of Bivium and Trivium are derived as $u_t = x_t + y_t$, and $v_t = x_t + y_t + z_t$, respectively. In this paper two examples from this class of stream ciphers are considered in detail, the specification of which

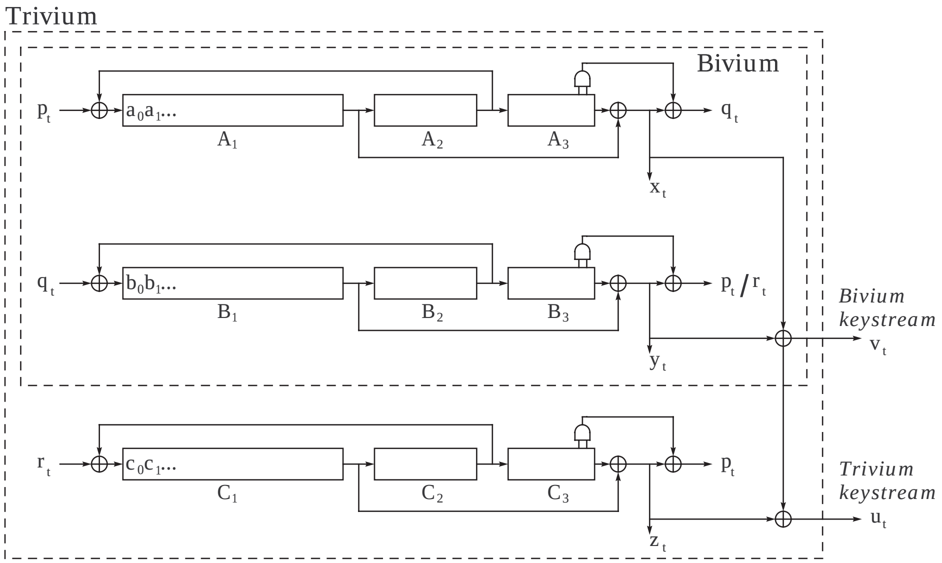

Figure 1: Bivium and Trivium classes of stream ciphers.

is given in Table 2. These correspond to Trivium and Bivium as described in [CP05, Rad06].

Description	Specification	$A : B : C$	Size, θ
Trivium [CP05]	Tri$(66, 3, 24; 69, 9, 6; 66, 21, 24)$	$93 : 84 : 111$	288
Bivium [Rad06]	Bi$(66, 3, 24; 69, 9, 6)$	$93 : 84 : -$	177

Table 2: Two instances' specifications, Trivium and Bivium.

For simplicity in further derivations let us introduce three subsets:

$$\begin{aligned} \mathcal{T}_0^{(t)} &= \{a_{3i+0}^{(t)}\} \cup \{b_{3j+0}^{(t)}\} \cup \{c_{3k+0}^{(t)}\} & & i = 0, 1, \ldots, A/3-1, \\ \mathcal{T}_1^{(t)} &= \{a_{3i+1}^{(t)}\} \cup \{b_{3j+1}^{(t)}\} \cup \{c_{3k+1}^{(t)}\} & \text{where} \quad & j = 0, 1, \ldots, B/3-1, \\ \mathcal{T}_2^{(t)} &= \{a_{3i+2}^{(t)}\} \cup \{b_{3j+2}^{(t)}\} \cup \{c_{3k+2}^{(t)}\} & & k = 0, 1, \ldots, C/3-1. \end{aligned} \tag{2}$$

3 The First Trivial Analysis: State Recovering

In this attack, given a keystream $\mathbf{u}$ of some length n an attacker wants to recover the internal state of the cipher. Since the cipher has invertible state-update function this also leads to a *key recovery attack*. A classical time-memory trade-off technique based on birthday paradox gives the upper bound for such an attack of $O(\sqrt{\theta})$ known keystream, and memory, where θ is the size of the internal state. The importance of the *state recovering analysis* is that it gives the upper bound for the length of the secret key K. When the design of TRIVIUM has appeared, several researchers raised the question: *Whether the secret key can be increased from 80 bits till, for example, 128 bits, thus, improving the security level?* In this section we will give the precise answer.

3.1 Guessing $\mathcal{T}_0^{(t)}$ at some time t

One of the main observations is that all blocks of the cipher are divisible by 3. Moreover, the transition of the internal state at time t to time $t+1$ is a linear transformation of the subset $\mathcal{T}_{t \bmod 3}^{(t)}$, plus a minor one bit disturbance from the adjacent two subsets. Therefore, the attack scenario can consist of the following phases.

PHASE I: Guess the state $\mathcal{T}_0^{(t)}$ at some time t,

PHASE II: Having the state $\mathcal{T}_0^{(t)}$ guessed correctly, recover the rest of the bits.

In the case of an exhaustive search of the sub state $\mathcal{T}_0^{(t)}$, the time complexity is $O(2^{\theta/3})$, and the keystream length is $O(1)$. Note also that the first $d = \min\{A_1, B_1, C_1\}/3$ forward triple-clocks d linear equations on the bits of $\mathcal{T}_0^{(t)}$ can be received. It means that the number of bits to be guessed can be reduced, and the total time complexity is then

$$O(2^{(\theta - \min\{A_1,B_1,C_1\})/3}).$$

For TRIVIUM and BIVIUM these complexities are 2^{74} and 2^{37}, respectively.

3.2 Guessing Outcomes for Specific AND Gates

To receive more linear equations for the phases I and II, we suggest to consider a set of specific AND gates:

$$\begin{array}{ll} a_{A-3}^{(t+3i)} \cdot a_{A-2}^{(t+3i)}, & i = 0, 1, \ldots, g_a - 1, \\ b_{B-3}^{(t+3j)} \cdot b_{B-2}^{(t+3j)}, & j = 0, 1, \ldots, g_b - 1, \\ c_{C-3}^{(t+3k)} \cdot c_{C-2}^{(t+3k)}, & k = 0, 1, \ldots, g_c - 1, \end{array} \tag{3}$$

where g_a, g_b, g_c are chosen parameters. If we guess these gates, then the number of linear equations that we can get for the phase I is

$$d' = \min\{g_a + \frac{B_1}{3}, g_b + \frac{C_1}{3}, g_c + \frac{A_1}{3}\}.$$

The most probable guess would be that all the gates are zeros, since $\Pr\{x \& y = 0\} = 0.75$. However, if we allow some of the gates to output ones, the length of the keystream can be reduced significantly. Let G gates out of $g_a + g_b + g_c$ `AND` gates produce zeros, and the remaining H gates produce ones. Then, the probability of such an event is

$$p_g = 0.75^G 0.25^H.$$

Note that we can allocate H ones among $G + H$ positions in $\binom{G+H}{H}$ ways. Therefore, the keystream is required to be of length $O\left(1/\left[p_g \cdot \binom{G+H}{H}\right]\right)$.

3.3 Guessing Sums of Specific `AND` Gates

The right guess of specific `AND` gates from the previous subsection allows us to increase the number of linear equations for the first phase till d'. However, the remaining bits of $\mathcal{T}_0^{(t)}$ should be guessed with probability 1/2. However, if the keystream can still be increased, then that probability can be reduced, in a trade-off with the keystream length.

After d' triple-clocks, we start receiving nonlinear equation, where the linear part consists of the bits from $\mathcal{T}_0^{(t)}$, and the nonlinear part is the sum of w `AND` gates, for some small w. Since the outcome of each of them is biased, then their sum is biased as well. Let p_w be the probability that the sum of w gates is zero, then we have:

$$p_w = \sum_{i=0}^{\lfloor w/2 \rfloor} \binom{w}{2i} 0.75^{w-2i} 0.25^{2i}, \tag{4}$$

or, its recursive formula is $p_{w+1} = 0.75p_w + 0.25(1 - p_w)$, with $p_0 = 1$. Let l_w be the number of nonlinear equations with the sum of w `AND` gates. Then, the time complexity to recover l_w bits is $p_w^{l_w}$, instead of 0.5^{l_w}, but the keystream length is increased by the ratio $p_w^{-l_w}$. The total probability of such an event is

$$q = \sum_{w=1}^{\infty} p_w^{l_w}.$$

It is rationale to use this approach for small ws, say for $w \in \{1, 2, 3, 4\}$, since for large ws the probability p_w is very close to 0.5, and it does not give a big gain in comparison with the truly random guess, but rather increases the length of the keystream rapidly.

3.4 Collecting System of Equations for Remaining Unknowns

Assume that the state of $\mathcal{T}_0^{(t)}$ and the outcomes of specific $G + H$ `AND` gates are guessed and derived correctly. To recover the remaining $2/3^{\text{rd}}$ of the state we need to collect a number of equations on $\mathcal{T}_1^{(t)}$ and $\mathcal{T}_2^{(t)}$, enough to derive the exact solution.

At any time t, if the values $a_{A-3}^{(t)}, b_{B-3}^{(t)}, c_{C-3}^{(t)}$ are known, then two consecutive clocks of the cipher are linear. Because of our specific guess, we have that d'

triple-clocks the system is linear. In one triple-clock we receive two linear equations on the remaining unknowns of the internal state. The first nonlinearity will not affect on the degree of receiving equations immediately, but rather with some delay. The first nonlinear equations will be of degree 2, and then of degree 3, and so on. Also note that each of H guesses also give us two equations of degree 1 of the form $x_i = 1$ and $x_{i+1} = 1$, and each of the G guesses give us one equation of degree 2 of the form $x_i x_j = 0$. The structure of this cipher is such that backward clocks increase the degree of equations rapidly footnote TRIVIUM is designed to maximize parallelism in forward direction. This allows hardware designers to choose trade-off between speed and chip-size. . Therefore, only few equations of low degree can be collected by backward clocking.

Let the number of equations of degrees 1 and 2 that we can collect be n_1 and n_2, respectively. Then, when all the parameters are fixed, a particular scenario can be described.

3.5 Attack Scenarios for TRIVIUM and BIVIUM

Let us accumulate techniques given in the previous sub sections, and propose a set of attack scenarios for TRIVIUM and BIVIUM in Table 3.

SCENARIO T0		Descr. = TRI			l_1:l_2:l_3:l_4 = 0:0:0:0			Ph.II unknowns=192	
g_a:g_b:g_c	G:H	r	d'	q	p_g	n_1	n_2	time	keystream
0:0:0	0:0	1	22	1	1	100	61	$c \cdot 2^{74.0}$	$O(1)$
SCENARIO T1		Descr. = TRI			l_1:l_2:l_3:l_4 = 5:5:4:1			Ph.II unknowns=192	
g_a:g_b:g_c	G:H	r	d'	q	p_g	n_1	n_2	time	keystream
46:37:42	125:0	1	59	$2^{-9.7}$	$2^{-51.9}$	192	178	$c \cdot 2^{83.5}$	$2^{61.5}$
SCENARIO T2									
42:33:38	113:4	$2^{22.6}$	55	$2^{-9.7}$	$2^{-53.2}$	192	162	$c \cdot 2^{88.9}$	$2^{40.3}$
SCENARIO T3		Descr. = TRI			l_1:l_2:l_3:l_4 = 0:0:5:4			Ph.II unknowns=192	
g_a:g_b:g_c	G:H	r	d'	q	p_g	n_1	n_2	time	keystream
29:30:30	89:0	1	52	$2^{-7.8}$	$2^{-36.9}$	158	152	$c \cdot 2^{79.7}$	$2^{44.7}$
SCENARIO B0		Descr. = BI			l_1:l_2:l_3:l_4 = 0:0:0:0			Ph.II unknowns=118	
g_a:g_b:g_c	G:H	r	d'	q	p_g	n_1	n_2	time	keystream
0:0:—	0:0	1	22	1	1	100	61	$c \cdot 2^{37.0}$	$O(1)$
SCENARIO B1									
9:5:—	14:0	1	27	1	$2^{-5.8}$	118	67	$c \cdot 2^{37.8}$	$2^{5.8}$

Table 3: Attack scenarios.

In all scenarios above the constant c is the time required for the second phase, where the remaining bits are recovered, and it is different for different scenarios.

T0 and B0 are trivial scenarios for TRIVIUM and BIVIUM, where no outcomes of any AND gates are guessed. However, the number of linear equations is not enough to recover the remaining bits using simple Gaussian elimination. Therefore, equations of a higher degree need to be collected and used. These scenarios have the least possible time and keystream complexities, and are the lower bounds.

In T1 and B1 we show optimal, on our view, choice of parameters such that the second phase has enough linear equations and the time complexity is minimal. However, along with linear equations we also have many equations of degree 2, which we are not using at all. Note that the attack complexities presented here are much lower than those given in [Rad06].

In T2 we show how the trade-off between the length of the keystream and time works. For a small increase of time we can reduce keystream significantly.

In T3 we receive a system of equations of degree ≤ 2 on 192 variables. This system is quite overdefined (more than 50%), and it might be possible to have an efficient algorithm for solving such a system.

However, the results given in these scenarios can be improved significantly if a *pre-* or/and a *post-* statistical tests can be applied efficiently. For these approaches see Appendices A and B.

3.6 System of Equations: Linear vs. Nonlinear

Another possibility to reduce the constant c can be done by efficient solving of a system of sparsed linear equations (in cases of T1, T2, B1), or by the use of equations of a higher degree (in cases of T3, B0).

Finding such an algorithm is a hard problem, and we leave it as an open question.

3.7 Conclusions: Our Results vs. Exhaustive Search

We have shown that Bivium can be broken for the time around $c \cdot 2^{37}$, which makes a really low bound for the security level. This example was taken to make a comparison with the paper [Rad06], where the best attack on this design has complexity around $c \cdot 2^{56}$ *seconds*.

Although the security level of Trivium is 2^{80}, an exhaustive search requires much more time, $\gamma 2^{80}$, where γ is the initialization time of the cipher, which includes 1152 clocks to be done before the first keystream bits are produced. Because of different implementation issues can be applied, including parallelism, an average time required for one clock of the cipher can vary. However, we can assume that the coefficient γ is around 2^{10}, and an exhaustive search would require around 2^{10+80} operations. This means that such scenarios as T1, T3 are competitive, and at least are very close to the exhaustive search, if not faster.

Obviously, in this particular design the security level cannot be improved by simple increasing the size of the key – our attack will definitely be faster than an exhaustive search. Therefore, in order to increase the security level the design of Trivium should be changed, for example, the size of the state could be increased. This would result in a longer initialization time and a larger hardware footpring.

4 The Second Trivial Analysis: Statistical Tests

Linear cryptanalysis is one of the most powerful analysis of stream ciphers. In this section we find a way of sampling from the keystream such that their distribution is biased. By this mean we build a linear distinguisher for the cipher.

4.1 Standard Approximation Technique

Let the variables of $\mathcal{T}_0$ be denoted as $\{w_0, w_1, \ldots, w_{95}\}$. Then, *assuming that all* `AND` *terms are zeros*, we receive a system of linear equations of rank 93 (instead of 96). It means that we can sample from the stream as follows

$$\sum_{i \in \mathcal{I}_k} w_i = N_k, \quad \forall k \in \{93, 94, 95, 96\}, \tag{5}$$

where

$$\begin{aligned}
\mathcal{I}_{93} = \{&0,1,4,6,8,9,12,13,14,17,19,20,23,25,27,30,31,34,35,38,39,41,43,44,\\
&67,68,70,72,73,76,77,80,81,84,85,88,89,92,93\};\\
\mathcal{I}_{94} = \{&0,2,4,5,6,7,8,10,12,15,17,18,19,21,23,24,25,26,27,28,30,32,34,36,\\
&38,40,41,42,43,45,67,69,70,71,72,74,76,78,80,82,84,86,88,90,92,94\};\\
\mathcal{I}_{95} = \{&0,3,4,5,7,11,12,14,16,17,18,22,23,24,26,28,29,30,33,34,37,38,42,\\
&46,67,71,75,76,79,80,83,84,87,88,91,92,95\};\\
\mathcal{I}_{96} = \{&0,5,9,14,15,18,20,24,29,41,44,47,67,70,73,96\}.
\end{aligned} \tag{6}$$

The noise variable N_k is a sum of a set of random `AND` gates. Therefore, the bias and the complexity of a distinguisher can be summarized in Table 4.

k	# of `AND` gates in N_k	bias ϵ	attack complexity
93	108	2^{-108}	2^{216}
94	126	2^{-126}	2^{252}
95	112	2^{-112}	2^{224}
96	72	2^{-72}	2^{144}

Table 4: Linear distinguishers for Trivium and its attack complexities.

Obviously, we could also mix these four equations to receive other 8 linear combinations that are different in principal from the found four. However, we could not achieve complexity lower than 2^{144}.

For Bivium, the rank appeared to be 57 (instead of 59), and similar resulting Table 5 is as follows.

I.e., Bivium can be distinguished from random in time complexity 2^{32}, which is much faster than all previously known attacks on it. Since the complexity of the attack is feasible, we could run the simulation of the attack on Bivium, which confirmed the found theoretical bias.

k	# ANDs	time	$\mathcal{I}_k$
57	49	2^{98}	$\{0,2,4,5,6,7,8,14,16,18,20,22,24,26,28,30,32,33,34,35,37,39,41,43,45,47,49,51,53,55,57\}$
58	49	2^{98}	$\{1,3,5,6,7,8,9,15,17,19,21,23,25,27,29,31,33,34,35,36,38,40,42,44,46,48,50,52,54,56,58\}$
59	16	2^{32}	$\{0,5,9,10,14,33,36,59\}$

Table 5: Linear distinguishers for Bivium and their attack complexities.

4.2 Another Way of Approximation

In the previous section all AND terms were approximated as zero. However, another sort of approximation is possible, such as

$$\texttt{AND}(x,y) = \tau_x x + \tau_y y + n,$$

where τ_x, τ_y are chosen coefficients, and n is the noise variable with the bias $\epsilon = 2^{-1}$. Whenever approximations for every AND gate are appropriately chosen, there must exist a biased linear equation on a shorter window of the keystream than that in the previous subsection. Our goal is to reduce the number of noise variables in the final expression for sampling. Unfortunately, the search for appropriate coefficients, which give us a strongly biased expression for sampling, is a hard task. Moreover, the probability that we can find an expression with the number of gates less than 72 is low. In our simulations we could find several biased equations on a shorter window, but the number of approximations were larger than 72. This issue is an interesting open problem.

4.3 Multidimensional Approximation

In Sub section 4.1 we gave a set of linear relations for a biased sampling from the keystream. The best equations for Trivium and Bivium require 72 and 16 approximations of AND gates, respectively. However, these samples are not independent, and some of the noises appear in several samples at different time instances. Therefore, the attack complexity can be improved by considering several samples jointly. I.e., we suggest to test a multidimensional approximation where one sample comes from a joint distribution.

Unfortunately, this did not give us a significant improvement. We considered three samples jointly, and the bias of that noise was $2^{-15.4}$, which is larger than 2^{-16}, but does not differ significantly.

5 Results and Conclusions

In this paper we have studied methods for analysis of Trivium-like stream ciphers. Below we give a comparison Table 6 of the known attacks on two instances, original Trivium and a reduced version called Bivium.

Case	Complexity	Exhaustive search	State Recovering Attack		Distinguishing Attack	
			previous	new attack	previous	new attack
TRIVIUM	time	$\gamma 2^{80}$ $\gamma \approx 2^{10}$	$\delta \cdot 2^{135}$ [eDF05] 2^{164} [Rad06]	$c \cdot 2^{83.5}$ $c \approx 2^{16}$	2^{144} [CP05]	—
	keystream	$O(1)$	$O(1)$	$2^{61.5}$	2^{144}	—
BIVIUM	time	$\gamma 2^{80}$	2^{56} sec. [Rad06]	$c \cdot 2^{36.1}$ $c \approx 2^{14}$	—	2^{32} verified
	keystream	$O(1)$	$O(1)$	$2^{11.7}$	—	2^{32}

Table 6: Resulting comparison of various attacks.

A brief summary for the algorithm of the state recovering attack on TRIVIUM is given in Table 7, and a distinguishing attack on BIVIUM is presented in Table 8.

Given: $\mathbf{u} = u_1, u_2$ – the keystream of TRIVIUM of length $2^{61.5}$

Attack Scenario T1*:*

1. For every $t = 0, 1, 2, \ldots, \lceil 2^{61.5} \rceil$ assume that $a_{90}^{(t+3i)} a_{91}^{(t+3i)} = 0, b_{81}^{(t+3j)} b_{82}^{(t+3j)} = 0, c_{108}^{(t+3k)} c_{109}^{(t+3k)} = 0$, for $i = [0 : 45], j = [0 : 36], k = [0 : 41]$.
2. Collect 59 linear equations on $\mathcal{T}_0$ with probability 1, and 15 more linear equations with the total probability $2^{-9.7}$, see Sub section 3.3.
3. For every guess of the remaining 22 bits from $\mathcal{T}_0$, derive the state of $\mathcal{T}_0$ using the linear equations collected in step 2.
4. Collect 192 linear equations on $\mathcal{T}_1$ and $\mathcal{T}_2$, clocking the cipher forward, under the assumption that the guess above was correct.
5. Recover the state of $\mathcal{T}_1$ and $\mathcal{T}_2$ by any linear technique (e.g., Gaussian elimination) in fixed time, and verify the solution in time $O(1)$.
6. Repeat the loops in steps 1 and 3 until the right internal state is found.

Table 7: Attack scenario T1 on TRIVIUM in brief.

With the key of 80 bits TRIVIUM seems to be secure. However, contrary to what one could expect from its almost 300 bit state, there is no security margin. This also means that one cannot use 128 bit keys and IVs with the current design. For this purpose, either the internal state has to be increased or some other re-design should take place.

```
 Given:   $\mathbf{v} = v_1, v_2$ – the keystream of Bivium of length $2^{32}$
  Init:   $P[2] = \mathbf{0}$ – a binary distribution, not normalized
 A linear distinguishing attack on Bivium:
1. For every $t = 1, 2, \ldots, 2^{32}$ calculate

     $s = v_t + v_{t+15} + v_{t+27} + v_{t+30} + v_{t+42} + v_{t+99} + v_{t+108} + v_{t+177}$,

   and attune the distribution as $P[s]++$.
2. After the loop is finished, calculate the distance

     $\xi = P[0]/2^{32} - 0.5$.

3. Make the final decision

     $\delta(\xi) = \begin{cases} \mathbf{v} \text{ is from Bivium}, & \text{if } \xi > 2^{-16}/2, \\ \mathbf{v} \text{ is Random}, & \text{if } \xi \leq 2^{-16}/2. \end{cases}$
```

Table 8: A linear distinguishing attack on Bivium in detail.

References

[BD05] S. Babbage and M. Dodd. Mickey-128, 2005. *http://www.ecrypt.eu.org/stream/ciphers/mickey128/mickey128.pdf.*

[BGW99] M. Briceno, I. Goldberg, and D. Wagner. A pedagogical implementation of A5/1. Available at *http://jya.com/a51-pi.htm* (accessed August 18, 2003), 1999.

[Blu03] SIG Bluetooth. Bluetooth specification. Available at *http://www.bluetooth.com* (accessed August 18, 2003), 2003.

[BSW00] A. Biryukov, A. Shamir, and D. Wagner. Real time cryptanalysis of A5/1 on a PC. In B. Schneier, editor, *Fast Software Encryption 2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, 2000.

[CP05] C. De Canniére and B. Preneel. Trivium – a stream cipher construction inspired by block cipher design principles. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/030 (2005-04-29), 2005. *http://www.ecrypt.eu.org/stream.*

[ECR05] eSTREAM: ECRYPT Stream Cipher Project, IST-2002-507932. Available at *http://www.ecrypt.eu.org/stream/* (accessed September 29, 2005), 2005.

[eDF05] eSTREAM Discussion Forum. A reformulation of TRIVIUM. created on 02/24/06 12:52PM, 2005. *http://www.ecrypt.eu.org/stream/phorum/read.php?1,448*.

[FM00] S. R. Fluhrer and D. A. McGrew. Statistical analysis of the alleged RC4 keystream generator. In B. Schneier, editor, *Fast Software Encryption 2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 19–30. Springer-Verlag, 2000.

[HJM05] M. Hell, T. Johansson, and W. Meier. Grain V.1. — a stream cipher for constrained environments, 2005. *http://www.it.lth.se/grain/grainV1.pdf*.

[LMV05] Y. Lu, W. Meier, and S. Vaudenay. The conditional correlation attack: A practical attack on Bluetooth encryption. In V. Shoup, editor, *Advances in Cryptology—CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 97–117. Springer-Verlag, 2005.

[LV04] Y. Lu and S. Vaudenay. Cryptanalysis of Bluetooth keystream generator two-level E_0. In *Advances in Cryptology—ASIACRYPT 2004*, Lecture Notes in Computer Science. Springer-Verlag, 2004.

[MJB04] A. Maximov, T. Johansson, and S. Babbage. An improved correlation attack on A5/1. In H. Handschuh and M. Anwar Hasan, editors, *Selected Areas in Cryptography—SAC 2004*, volume 3357 of *Lecture Notes in Computer Science*, pages 1–18. Springer-Verlag, 2004.

[MS01] I. Mantin and A. Shamir. Practical attack on broadcast RC4. In M. Matsui, editor, *Fast Software Encryption 2001*, volume 2355 of *Lecture Notes in Computer Science*, pages 152–164. Springer-Verlag, 2001.

[NES99] NESSIE: New European Schemes for Signatures, Integrity, and Encryption. Available at *http://www.cryptonessie.org* (accessed August 18, 2003), 1999.

[Rad06] H. Raddum. Cryptanalytic results on TRIVIUM. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/039, 2006. *http://www.ecrypt.eu.org/stream*.

[Sma03] N. Smart. *Cryptography: An Introduction.* McGraw-Hill Education, 2003. ISBN 0-077-09987-7.

[WSLM05] D. Whiting, B. Schneier, S. Lucks, and F. Muller. Phelix - fast encryption and authentication in a single cryptographic primitive. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/020 (2005-04-29), 2005. *http://www.ecrypt.eu.org/stream*.

Appendix A: Statistical Pre-Test for the Phase I

In the scenarios above the constant c within time complexity denotes the time needed for solving a system of equations in the second phase. Although the equations are sparse, this constant can still be large. When the number of variables is 192, we assume that this constant is approximately lower bounded as $c \approx 2^{16}$.

One idea to reduce the total time complexity is to consider only those "windows" in the stream where the probability for the guess of the AND gates is larger than in a random case.

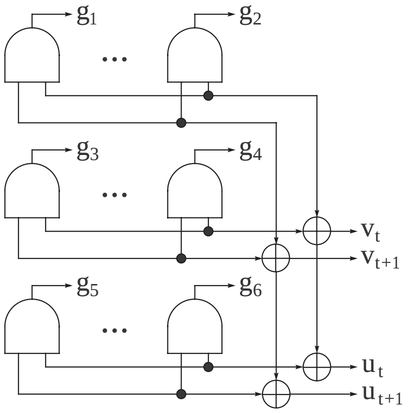

Figure 2: Statistical pre-test.

Let us observe an output pair (u_t, u_{t+1}) (or (v_t, v_{t+1})) at some time t and $t+1$, each component of which is the sum of 6 (respectively, 4) bits of the state from $\mathcal{T}_1^{(t)}$ and $\mathcal{T}_2^{(t)}$, as shown in Figure 2. The question here is: What is the probability that the sum of six (four) AND gates is zero, given the observed pair? We can use this criteria to cut undesired cases, since the sum of the gates must be zero when all of them are zeros as well. Below, in Table 9, we give these probabilities in accordance.

I.e., when the keystream in a specified "window" is a zero sequence, then the probability of our guess, a set of specific AND gates is zero, is larger than otherwise. However, this approach would require a much longer keystream, and the gain in time complexity is not significant. More complicated tests can also be developed.

(u_t, u_{t+1})	Pr{the sum of `AND` gates is zero}	
(v_t, v_{t+1})	in Trivium	in Bivium
$(0, 0)$	0.53125	0.625
$(0, 1)$	0.5	0.5
$(1, 0)$	0.5	0.5
$(1, 1)$	0.5	0.5

Table 9: Keystream influence for the pre-test technique.

Appendix B: Statistical Post-Test of the Phase I

Another approach is to make a test after the first 1/3rd of the state is guessed and derived. Let us introduce a decision rule for the test

$$\delta(\mathcal{T}_0^{(t)}) = \begin{cases} \text{Accept}, & \mathcal{T}_0^{(t)} \text{ passes the test,} \\ \text{Reject}, & \text{otherwise.} \end{cases} \tag{7}$$

Associated with the decision rule δ there are two error probabilities.

$$\begin{aligned} \alpha &= \Pr\{\delta(\mathcal{T}_0^{(t)}) = \text{Reject}|\text{the guess } \mathcal{T}_0^{(t)} \text{ is correct}\}, \\ \beta &= \Pr\{\delta(\mathcal{T}_0^{(t)}) = \text{Accept}|\text{the guess } \mathcal{T}_0^{(t)} \text{ is wrong}\}. \end{aligned} \tag{8}$$

Thus, the time complexity can be reduced from $c{\cdot}Q$ down to $\beta{\cdot}c{\cdot}Q$. However, the success of the attack will be $P_{\texttt{succ}} = 1 - \alpha$. If the test is statistically strong, then α and β are small, lowering the time complexity significantly.

One such a test could be as follows. At a time t the sequence of d' triple-clocks allows us to receive d' linear equations on the bits of $\mathcal{T}_0^{(t)}$. However, if we continue clocking, we will then receive a sequence of biased samples. The bias decreases rapidly as long as the number of random `AND` terms in the equation for the noise variable grows.

Unfortunately, for Trivium there is no valuable gain, but for Bivium the gain is more visible. Consider the scenario B1. After the first phase the following triple-clocks give us the following samples.

`AND` gates in the noise, $i =$	1	2	3	4	...
Number of samples, $l_i =$	5	4	1	13	∞

Let us denote the first 23 samples (24=5+4+1+13) as $\mathbf{s}^{23} = s_0, s_1, \ldots, s_{22}$, and the decision rule for our test be

$$\delta(\mathbf{s}^{23}) = \begin{cases} \text{Accept}, & \text{if } H_w(\mathbf{s}^{23}) \geq \sigma_0, \\ \text{Reject}, & \text{otherwise,} \end{cases}$$

where $0 \leq \sigma_0 \leq 23$ is some appropriately chosen *decision threshold*. The error probabilities are then as follows.

$$\begin{aligned} \alpha &= \sum_{\left\{\substack{\forall t_w \,:\, 0 \le t_w \le l_w,\, w = 1 \ldots 4 \\ t_1 + t_2 + t_3 + t_4 < \sigma_0}\right.} \prod_{w=1}^{4} \binom{l_w}{t_1} p_w^{t_w} (1 - p_w)^{l_w - t_w}, \\ \beta &= 2^{-23} \sum_{t=\sigma_0}^{23} \binom{23}{t}, \end{aligned} \tag{9}$$

where the probabilities p_w are calculated via (4). Additional information is extracted from the fact that the distribution of α is "shifted" with regard to the distribution of β, and, therefore, the gain can be achieved. In Table 10 these probabilities are given for several values of the threshold σ_0.

σ_0	0	7	11	12	14	18	23
α	~ 0	0.0038	0.1585	0.2964	0.6275	0.9839	~ 1
β	~ 1	0.9826	0.6612	0.5000	0.2024	0.0053	~ 0

Table 10: Error probabilities for the post-test technique.

I.e., if we choose $\sigma_0 = 18$ in B1, then the time complexity will be $c \cdot 2^{30.2}$, instead of $2^{37.8}$. The length of the keystream remains the same. However, the success probability of this attack is $P_{\mathtt{succ}} = 0.0161$, which is low.

The situation with the success rate can be improved if the attack will be repeated $1/P_{\mathtt{succ}}$ times. Thus, we have the overall time complexity around $2^{5.9} \cdot 2^{30.2} = 2^{36.1}$, but the keystream is also increased till $2^{11.7}$. We could trade-off a better time complexity with the length of the keystream, and the overall success probability is around 1.

Searching for a proper statistical test is a challenge and is not an easy task.

View publication stats