Breaking One.Fivium by AIDA an Algebraic IV Differential Attack

Michael Vielhaber, Instituto de Matemáticas, Universidad Austral de Chile
Casilla 567, Valdivia, Chile, vielhaber@gmail.com October 28, 2007

Abstract We show, how to break Trivium with a setup of 576 (instead of 1152) clock cycles, with an effort of 2^6 chosen IV resynchronisations up to cycle 625 for each of the 47 recovered key bits.

Keywords: Trivium, estream, Algebraic IV differential attack, AIDA.

1. The Trivium and One.Fivium stream ciphers

The eStream cipher proposal Trivium [1] has a 288 bit register, initially filled with the (unknown) key at bits 1–80, and with the (supposedly choosable) IV at bits 94–173. Bits 286–288 are set to one, all others to zero.

The setup phase then consists in $4 \cdot 288$ cycles without revealing output. We assume a setup of just $2 \cdot 288 = 576$ cycles, half the length, hence our name 1.5-vium instead of 3-vium for this (acknowledgedly weaker) algorithm.

2. Definitions

(*i*) Let $s_1, \ldots s_{288}$ be the register bits of Trivium as in [1].

(*ii*) Let $K_1, \ldots, K_{80}$ be the key bits. Initially, $K_1 = s_1, \ldots, K_{80} = s_{80}$. Occasionally, we will write $K(i)$ for K_i.

Note: In this paper, we keep with the original enumeration, whereby K_1 goes into s_1, IV_1 goes into s_{94}. We see nothing "more natural" (cf. [3]) in using the other way round, and certainly any change likely amounts to confusion – more severe in cryptanalysis than mere excess or lack of "naturality".

(*iii*) Let $IV_1, \ldots, IV_{80}$ be the key bits. Initially, $IV_1 = s_{94}, \ldots, IV_{80} = s_{173}$.

(*iv*) Let I be a subset of the index set $\{1, 2, \ldots, 80\}$. Then

$$IV_I := \wedge_{i \in I} IV_i,$$

e.g. $IV_{\{7,11\}} = IV_7 \wedge IV_{11}$ and $IV_\emptyset = 1$ (always true for empty index set).

(*v*) Let $OUT(t)$ be the output at time step t, where $t = 1$ corresponds to the clock cycle *immediately following* filling the register. Hence, Trivium according to specification [1] will reveal output bits from $OUT(1153)$ on, while $OUT(1), \ldots, OUT(1152)$ are hidden during the setup phase. One.Fivium shows $OUT(577), OUT(578), \ldots$.

(*vi*) Let OUT(t):IV be the output for a given fixed IV (still depending on the unknown key).

(*vii*) For $0 \leq n \leq 80$; $i_k \in \{1, \ldots, 80\}$, $1 \leq k \leq n$; $t \in \mathbb{N}$, let

$$t :< i_1, \ldots, i_n >:= \bigoplus_{0 \leq IV_{i_1}, \ldots, IV_{i_n} \leq 1; IV_j = 0, j \neq i_k, \forall k} OUT(t) : IV$$

be the sum modulo 2 of $OUT(t)$ from all 2^n runs of TRIVIUM where the specified IV bits assume all possible combinations and the other $80 - n$ IV bits are held zero.

3. Exclusive Normal Form

All further manipulations after the initial fill are based on the two functions $\oplus$ (XOR) and $\wedge$ (AND) over $\mathbb{F}_2$. We can thus describe every register bit as well as the output by what we call ENF, Exclusive Normal Form. Using definition (*iv*), the ENF of some Boolean expression in the $IV_1 \ldots, IV_{80}, K_1, \ldots, K_{80}$ is:

$$\bigoplus_{I=\emptyset}^{\{1,2,\ldots,80\}} IV_I[\oplus_{k=1}^{n_I} \wedge K_{\{I,k\}}],$$

where I runs through all the 2^{80} combinations of the IV-Bits, n_I is the number of minterms including a certain IV combination IV_I, and $K_{\{I,k\}}$ is the *and*-ing of the corresponding key bits (minterms).

Example:

The output at time $t = 1$ (invisible) is given by $K_{66} \oplus IV_{69} \oplus 1$, hence in ENF $\mathrm{OUT}(1) = IV_\emptyset[K_{66} \oplus 1] \oplus IV_{\{69\}}[1]$

Proposition 1

$\oplus, \wedge, 1$ *is a complete system,i.e. every Boolean function can be represented by some ENF* (*where* $IV_\emptyset = 1$ *is always true*).

Proof. As we know from Boolean algebra, $\vee, \wedge, \neg$ is a complete system. Since $\neg a = a \oplus 1$ and $a \vee b = a \oplus b \oplus (a \wedge b)$, we are done. □

Proposition 2 Distributive Law

$$(\oplus_{i \in I} A_i) \ \wedge \ (\oplus_{j \in J} B_j) = \oplus_{(i,j) \in I \times J} (A_i \wedge B_j)$$

Proof. We show $(a \oplus b) \ \wedge \ (c \oplus d) = ac \oplus ad \oplus bc \oplus bd$ by a truth table. The whole formula then follows by induction over the quantities $|I|$ and $|J|$.

a	b	c	d	a⊕ b	c⊕ d	∧ = ⊕	ac	ad	bc	bd
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	0
0	1	0	1	1	1	1	0	0	0	1
0	1	1	0	1	1	1	0	0	1	0
0	1	1	1	1	0	0	0	0	1	1
1	0	0	0	1	0	0	0	0	0	0
1	0	0	1	1	1	1	0	1	0	0
1	0	1	0	1	1	1	1	0	0	0
1	0	1	1	1	0	0	1	1	0	0
1	1	0	0	0	0	0	0	0	0	0
1	1	0	1	0	1	0	0	1	0	1
1	1	1	0	0	1	0	1	0	1	0
1	1	1	1	0	0	0	1	1	1	1

□

In a similar way, every bit of Trivium at every timestep can in principle be represented by its ENF, ordered by the IV contents of the respective conjunction.

Hence, the feedback function $s_{178} = s_{162} \oplus s_{177} \oplus s_{264} \oplus s_{176} \cdot s_{175}$ (and similarly for the sites s_1 and s_{94}) can be expressed in terms of the respective ENF's as:

$$ENF(178) = ENF(162) \oplus ENF(177) \oplus ENF(264) \oplus (ENF(175) \wedge ENF(176)),$$

where by Proposition 2

$$(\oplus_{i=1}^{m} IVa_i \wedge Ka_i) \wedge (\oplus_{j=1}^{n} IVb_j \wedge Kb_j) := \bigoplus_{(i,j)=(1,1)}^{(n,m)} (IVa_i \wedge IVb_j) \wedge (Ka_i \wedge Kb_j).$$

Here we can appreciate that generally, the ENF complexity (number of terms) will *square* due to the $\wedge$ operation (an explosion somewhere between exponentiation and Ackermann's function). Fortunately (for the attacker), squaring 0 or 1 is the typical effect up to ca. 300 clock cycles, before take-off.

4. Algebraic IV Differential Attack: Inclusion-Exclusion-Principle

These ENF's thus grow fast and become huge. However, it is possible to rip apart the ENF of a certain output position by choosing an appropriate mix of IV's:

Let the IV be allzero. Then only the part with $I = \emptyset$ is valid, since all other values for I do not correspond to an allzero IV. Similarly, an IV of $0x0\_05$ that is bits 1 and 3 set, will result in the sum over $I = \emptyset, \{1\}, \{3\}, \{1, 3\}$ that is four terms being present

(and in general 2^k terms, if the IV has Hamming weight k). Apparently, there is no IV directly responsable for $I = \{1, 3\}$ and *only* $I = \{1, 3\}$.

However, by the inclusion-exclusion-principle, we obtain:

IV	$I = \emptyset$	$I = \{1\}$	$I = \{3\}$	$I = \{1, 3\}$
IV = 0x0_05	$\surd$	$\surd$	$\surd$	$\surd$
IV = 0x0_04	$\surd$		$\surd$	
IV = 0x0_01	$\surd$	$\surd$		
IV = 0x0_00	$\surd$			

and thus (now comes the inclusion-exclusion-principle): $\bigoplus_{IV \subset 0x0\_05} = [I = \{1, 3\}]$.

Proposition 3

Let $OUT(t) = (IV_{i_1} \wedge IV_{i_2} \wedge \cdots \wedge IV_{i_n}) \wedge K_k \bigoplus Z$, *where the ENF* Z *does not include terms of the form* $(IV_{i_1} \wedge IV_{i_2} \wedge \cdots \wedge IV_{i_n}) \wedge (K_{j_1} \wedge K_{j_2} \wedge \cdots \wedge K_{j_m})$, *for any* $m \geq 0, 1 \leq j_1, \ldots, j_m \leq 80$.

Then $K_k = \bigoplus_{I \subset \{i_1, i_2, \ldots, i_n\}} OUT(t) : IV_I$

Proof: By the inclusion–exclusion–principle. □

Since there are 80 IV bits, there are 2^{80} $I-$sets to be chosen from (where the full set $I = \{1, 2, \ldots, 80\}$ would require (by the inclusion–exclusion–principle) to check *all* 2^{80} IV combinations, as brute-forzish as just checking all key combinations).

The attack on Trivium and similar ciphers now splits into two phases:

A) A once-and-for-all precomputation, to determine a set I at a sufficiently distant time t (as to be *after* the setup phase, we do not achieve this yet) with an ENF entry of the form $IV_I[\oplus_{k=1}^{n} X_k]$, where the X_k should be a single bit of K or at most double terms $K_{k_1} \wedge K_{k_2}$. This precomputation is done only once per cipher. It has *not* to be repeated for different key values.

B) To attack the cipher it then suffices to run the chosen IV's according to the pattern in I ($2^{\#I}$ IVs, where $\#I$ is the Hamming weight). This phase B should be essentially trivial.

5. Results obtained for a setup of 576 cycles

We have, *e.g.*, $K(62) = 624 :< 2, 7, 8, 12, 27, 78 >$, that is after a setup of 623 or less cycles (instead of 1152) key bit 62 is visible as a linear combination of a certain output bit of $2^6 = 64$ runs with chosen IV, since $OUT(624) = IV_2 IV_7 IV_8 IV_{12} IV_{27} IV_{78} K_{62} \bigoplus \ldots$, where $\ldots$ is some (large) ENF without the IV–Kombination $IV_2 IV_7 IV_8 IV_{12} IV_{27} IV_{78}$.

The list of key bits (47 bits, the rest being an easy 2^{33} exercise in brute force) and corresponding xor sums of output bits is given here, please note that all timesteps lie between 577 and 625, incl., and no combination needs more than 6 IV bits to be set.

Key Bit(s)	Clock Cycle	IV–Bits used in combinations	Key Bit(s)	Clock Cycle	IV–Bits used in combinations
1	597	$< 4, 7, 12, 15, 2, 56 >$	61	587	$< 4, 7, 12, 15, 40, 74 >$
$2 \oplus 65$	580	$< 4, 7, 12, 15, 8, 33 >$	62	604	$< 4, 7, 12, 15, 23, 75 >$
$3 \oplus 66$	580	$< 4, 7, 12, 15, 14, 32 >$	63	604	$< 4, 7, 12, 15, 23, 74 >$
4	579	$< 4, 7, 12, 15, 6, 47 >$	64	597	$< 4, 7, 12, 15, 3, 30 >$
5	577	$< -, 7, 12, 15, 1, 79 >$	65	580	$< 4, 7, 12, 15, 2, 33 >$
6	611	$< 4, 7, 12, 15, 41, 51 >$	66	580	$< 4, 7, 12, 15, 16, 34 >$
8	589	$< 4, 7, 12, 15, 23, 54 >$	67	596	$< 4, 7, 12, 15, 40, 65 >$
9	589	$< 4, 7, 12, 15, 36, 63 >$	68	596	$< 4, 7, 12, 15, 40, 64 >$
11	595	$< 4, 7, 12, 15, 24, 41 >$	15	581	$< 4, 28, 31, 79, 3, 47 >$
14	604	$< 4, 7, 12, 15, 21, 32 >$	18	600	$< 4, 28, 31, 79, 1, 69 >$
16	578	$< 4, 7, 12, 15, 77, 79 >$	20	598	$< 4, 28, 31, 79, 3, 50 >$
17	588	$< 4, 7, 12, 15, 20, 79 >$	23	625	$< 4, 28, 31, 79, 8, 12 >$
19	587	$< 4, 7, 12, 15, 23, 40 >$	30	606	$< 4, 28, 31, 79, 12, 46 >$
25	580	$< 4, -, 12, 15, 23, 49 >$	32	606	$< 4, 28, 31, 79, 1, 17 >$
26	580	$< 4, -, 12, 15, 22, 49 >$	33	591	$< -, 28, 31, 79, 2, 37 >$
27	579	$< 4, 7, 12, -, 23, 48 >$	35	589	$< 4, 28, 31, 79, 14, 51 >$
36	583	$< 4, 7, 12, -, 34, 44 >$	58	588	$< 4, 28, 31, 79, 35, 38 >$
38	580	$< -, 7, 12, 15, 49, 55 >$	21	583	$< 2, 7, 8, 12, 19, 45 >$
39	578	$< -, 7, 12, 15, 52, 79 >$	22	583	$< 2, 7, 8, 12, 20, 56 >$
55	598	$< 4, 7, 12, 15, 51, 58 >$	10	583	$< 2, 8, -, 80, 19, 43 >$
56	578	$< 4, 7, 12, 15, 26, 50 >$	12	582	$< 2, 8, 12, 80, 19, 44 >$
$57 \oplus 63$	588	$< 4, 7, 12, -, 14, 24 >$	58	607	$< 2, 8, 12, 80, 19, 71 >$
$59 \oplus 65$	612	$< 4, 7, 12, 15, 10, 41 >$	69	579	$< 2, 8, 12, 80, 14, 49 >$
$60 \oplus 66$	589	$< 4, -, 12, 15, 38, 48 >$			

The second line should be read as $K_2 \oplus K_{65} = \oplus_{I \subset \{4,7,12,15,8,33\}} OUT(580) : IV_I$.

How did we obtain this table: By semi-exhaustive searching *many* IV combinations. So phase A is done (and involved *far* more than $80 \times 640 \times 2^6$ steps of the Trivium algorithm, the effort phase B now takes).

How can you verify it: Take the official Trivium implementation. Tweak the setup length from 1152 to 576 (comment out the second line UPDATE(); ROTATE();, l. 254 in the reference implementation [3]). Now key stream starts after 576 setup cycles. Run it any number you wish with random key bits and the IV's as described by the table. Compare the output xor with the (random!) key bit. Observe equality. Every time. If you believe in probabilistically checking primality, you are fine here as well.

6. Comparison with Turan and Kara's work

Another attack on Trivium with reduced setup length is described in [2]. What Turan and Kara call a "2-Round-Trivium" consists in a setup of only 288 clock

cycles, before access to output stream is allowed. They obtain a 2^{-31} approximation to certain key bits.

In contrast, our approach allows (at least) 576 clock cycles and gives *direct* equations for the key bits. So, in Turan and Kara's terminology, this is a 4-Round attack with a bias of 2^{-0}! (using the piling-up lemma with $\varepsilon = p(1) - p(0)$, to be able to directly multiply biases of *xor*-ed streams).

7. Further Work

The apparent goal is to push the allowed setup length from the achieved 576 all the way up to 1152. This will require larger sets of IV's, not just 6, and thus a whole lot more precomputation.

Furthermore, we obtained directly key bits as K part in the ENF. We could of course also use sums of key bits, or terms with 2 (or some few) key bits *and*–ed together, requiring some more computational effort in phase B.

This work is under way. A sneak preview:
$K(23) = 640{:}< 3, 4, 7, 12, 15, 25, 29, 79 >$, covering 5/9 of the setup time.

Conclusion

We developed a new kind of attack against low-complexity algorithms, the AIDA, Algebraic IV Differential Attack.

While the Trivium inventors state in [1, Sect. 4.5] that "*each state bit depends on each key and IV bit in nonlinear way after two full cycles (i.e.,* $2 \cdot 288$ *iterations). We expect that two more cycles will suffice to protect the cipher against resynchronization attacks*", we have shown that at least the nonlinear dependance gives no safety at all and the margin against attacks will (if at all) only start within the next 576 iterations.

References

[1] Christophe de Cannière, Bart Preneel, "Trivium Specifications"
`http://www.ecrypt.eu.org/stream/p3ciphers/trivium/trivium_p3.pdf`

[2] M. S. Turan, O. Kara,"Linear Approximations for 2-round Trivium"
`http://www.ecrypt.eu.org/stream/papersdir/2007/008.pdf`

[3] `http://www.ecrypt.eu.org/stream/p3ciphers/trivium/trivium_p3source.zip::trivium.c`