jet INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2018, VOL. 64, NO. 2, PP. 151–156
Manuscript received September 29, 2017; revised March, 2018. DOI: 10.24425/119360

Phase-shift Fault Analysis of Grain v1

Viliam Hromada, and Tibor Pethö

***Abstract*—This paper deals with the phase-shift fault analysis of stream cipher Grain v1. We assume that the attacker is able to desynchronize the linear and nonlinear registers of the cipher during the keystream generation phase by either forcing one of the registers to clock one more time, while the other register is not clocked, or by preventing one of the registers from clocking, while the other register is clocked. Using this technique, we are able to obtain the full inner state of the cipher in reasonable time (under 12 hours on a single PC) by using 150 bits of unfaulted keystream, 600 bits of faulted keystreams and by correctly guessing 28 bits of the linear register.**

***Keywords*—stream ciphers, Grain v1, fault analysis, phase-shift attack, desynchronization attack**

I. Introduction

FAULT analysis is a relatively new approach in the cryptanalysis of stream ciphers. It is based on actively tampering with the device and inducing faults in the computation which results in keystreams different from the unfaulted one. This method of analysis was introduced in paper [6] by Hoch and Shamir. In their paper, authors proposed bit-flipping and phase-shift as two default fault analysis methods. Bit-flipping is changing the value of a single bit of the device's register, while phase-shift is the clocking of a certain device's register while the other registers are not clocked. This type of an attack is sometimes referred to as a desynchronization attack. The security of several stream ciphers has been investigated regarding this type of an attack, e.g. Trivium [7], A5/1 [4] or irregularly decimated generators [8].

In 2008, a stream cipher Grain v1 [5] was selected as one of the finalists of the eSTREAM project. The goal of the project was to find a set of stream ciphers suitable for different cryptographic applications. Grain was chosen as one of the most suitable stream ciphers for hardware implementation thus making it interesting for fault analysis, since its design is based on two feedback shift registers that are clocked at the same time and the output is a nonlinear combination of registers' bits.

Several fault attacks on Grain v1 have been published, e.g. [1], [2], [3]. To our knowledge, all published fault attacks on Grain v1 use the bit-flipping technique. We therefore focused on the phase-shift fault analysis of the cipher in combination with a method of algebraic cryptanalysis - SAT solvers. In the attack, we desynchronize the cipher's inner state by either forcing one extra clock in the linear or non-linear register, or by stopping one of the two registers for one clock. This desynchronization is done in the moment directly after the initialization of the device. We investigate the mathematical model of the attack by representing the output bits as a set of non-linear equations over GF(2) in variables that represent the bits of the inner state of the cipher and then try to solve this system of equations by using SAT-solvers to find some inner state of the cipher. The attack was simulated (implemented) in the mathematical package SAGE.

It stems from our experiments that guessing correctly at least 28 bits of the linear register of Grain v1 is sufficient to solve the system of equations under 12 hours on a single PC, while inducing all four possible clocking faults and generating 150 bits per keystream. Our further experiments suggest that the most important fault for the success of the attack is the stopping of the linear register for one clock, since omitting the corresponding equations led to the most significant increase in the time needed to solve the system of equations and if we attack the cipher by using only this fault, the time needed to solve the system is lower than in other three cases.

This paper is organized as follows. Section 2 describes the cipher Grain v1 and its used representation. Section 3 contains the description of our attack. In Section 4 we present the results of the experiments and we conclude the paper in Section 5.

II. Stream cipher Grain v1

Grain v1 [5] is a hardware-oriented synchronous stream cipher. It generates a sequence of keystream bits from an 80-bit secret key and an 64-bit initialization vector. Its inner state consists of 160 bits stored in two feedback shift registers, one linear (LFSR) and one non-linear (NFSR), as depicted on Figure 1.

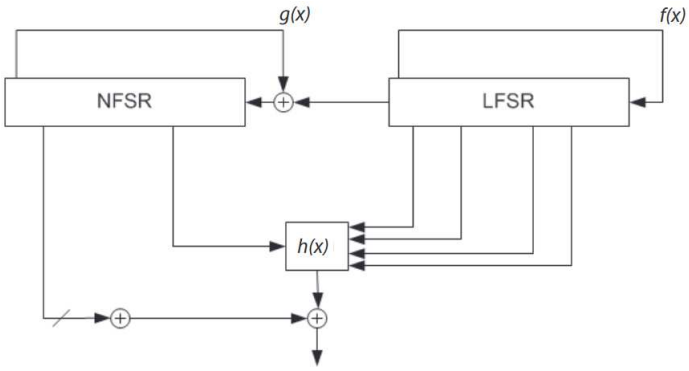

Fig. 1. Cipher Grain v1 [5]

The content of the LFSR is denoted $s_i, s_{i+1}, ..., s_{i+79}$. The feedback polynomial of the LFSR, denoted $f(x)$, is a primitive polynomial defined as:

$$1 + x^{18} + x^{29} + x^{42} + x^{57} + x^{67} + x^{80}$$

This work was supported by the grant VEGA No. 1/0159/17.

V. Hromada is with Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Bratislava, Slovakia (e-mail: viliam.hromada@stuba.sk).

T. Pethö is a graduate of Slovak University of Technology in Bratislava, Bratislava, Slovakia (e-mail: 8847@is.stuba.sk)

i.e. the update function of the LFSR is

$$s_{i+80} = s_{i+62} + s_{i+51} + s_{i+38} + s_{i+23} + s_{i+13} + s_i \quad (1)$$

The content of the NFSR is denoted $b_i, b_{i+1}, ..., b_{i+79}$. The feedback polynomial of the LFSR, denoted $g(x)$, is a polynomial defined as:

$$\begin{aligned} g(x) =& 1 + x^{17} + x^{20} + x^{28} + x^{35} + x^{43} + x^{47} + x^{52} + x^{59} \\ &+ x^{65} + x^{71} + x^{80} + x^{17}x^{20} + x^{43}x^{47} + x^{65}x^{71} \\ &+ x^{20}x^{28}x^{35} + x^{47}x^{52}x^{59} + x^{17}x^{35}x^{52}x^{71} \\ &+ x^{20}x^{28}x^{43}x^{47} + x^{17}x^{20}x^{59}x^{65} + x^{17}x^{20}x^{28}x^{35}x^{43} \\ &+ x^{47}x^{52}x^{59}x^{65}x^{71} + x^{28}x^{35}x^{43}x^{47}x^{52}x^{59} \end{aligned}$$

and the update function of the NFSR is

$$\begin{aligned} b_{i+80} =& s_i + b_{i+62} + b_{i+60} + b_{i+52} + b_{i+45} + b_{i+37} \\ &+ b_{i+33} + b_{i+28} + b_{i+21} + b_{i+14} + b_{i+9} + b_i \\ &+ b_{i+63}b_{i+60} + b_{i+37}b_{i+33} + b_{i+15}b_{i+9} \\ &+ b_{i+60}b_{i+52}b_{i+45} + b_{i+35}b_{i+28}b_{i+21} \\ &+ b_{i+63}b_{i+45}b_{i+28}b_{i+9} + b_{i+60}b_{i+52}b_{i+37}b_{i+33} \\ &+ b_{i+63}b_{i+60}b_{i+21}b_{i+15} + b_{i+63}b_{i+60}b_{i+52}b_{i+45}b_{i+37} \\ &+ b_{i+33}b_{i+28}b_{i+21}b_{i+15}b_{i+9} \\ &+ b_{i+52}b_{i+45}b_{i+37}b_{i+33}b_{i+28}b_{i+21} \end{aligned} \quad (2)$$

At a given time i the inner state of the cipher, denoted IS_i, is:

$$IS_i = (b_i, b_{i+1}, ..., b_{79+i}, s_i, s_{i+1}, ..., s_{79+i}), i \geq 0.$$

During the keystream generating phase, 5 of these variables are taken as an input to nonlinear filtering function $h(x)$ defined as:

$$\begin{aligned} h(x) =& x_1 + x_4 + x_0x_3 + x_2x_3 + x_3x_4 + x_0x_1x_2 + x_0x_2x_3 \\ &+ x_0x_2x_4 + x_1x_2x_4 + x_2x_3x_4 \end{aligned}$$

where the variables x_0, x_1, x_2, x_3, x_4 correspond to variables $s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}, b_{i+63}$. The output function of the cipher is taken as

$$z_i = \sum_{k \in \mathcal{A}} b_{i+k} + h(s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}, b_{i+63}) \quad (3)$$

where $\mathcal{A} = \{1, 2, 4, 10, 31, 43, 56\}$.

The initial state of the cipher is:

$$IS_0 = (b_0, b_1, ..., b_{79}, s_0, s_1, ..., s_{79}).$$

The initial state is initialized with the secret key $K = (k_0, k_1, ..., k_{79})$ and the initialization vector $IV = (u_0, u_1, ..., u_{63})$ as follows:

$$\begin{aligned} b_i &= k_i, i \in \{0, 1, ..., 79\} \\ s_i &= u_i, i \in \{0, 1, ..., 63\} \\ s_i &= 1, i \in \{64, 65, ..., 79\}. \end{aligned}$$

After the initialization, the cipher is clocked 160 times without generating any keystream bits. Instead, the output function is xored back with the inputs of the registers. After that Grain v1 generates one output bit per each clock. All the above equations are over the finite field $GF(2)$. If the inner state of Grain v1 is known at an arbitrary time, it is not difficult to clock the cipher backwards. Therefore, if we are able to find out the inner state IS_{t_j} at a time t_j, then we are able to clock the cipher backwards to the initial state and directly get the 80-bit secret key K.

III. Phase-shifting in Grain v1

The idea of phase-shifting is to desynchronize the registers of the cipher. If an encryption device consists of more registers, basically they are either clocked together, or are clock-controlled, meaning that the output of one register controls the clocking of another register. The de-synchronization of the device is then performed by tampering with the clocking of one of the registers, e.g. one of the registers stops for one clock, or is clocked one time more beforehand. Several papers were published that use this technique to successfully attack different stream ciphers [4], [7], [8]. Our model of this attack assumes that the following conditions are met:

1) The cryptographic device that implements Grain v1 is in the possession of the attacker.
2) The attacker can use the device to generate keystreams of an arbitrary length.
3) The attacker can tamper with the clocking of the registers by either clocking one of the registers one clock ahead, or by stopping one of the registers for one clock.
4) The attacker has perfect control over the moment of phase-shift.
5) The attacker is able to reset the device to its original proper unfaulted state.

Point 4 means that the attacker is able to run the initialization phase of Grain v1 and only after that desynchronizes the registers. He then proceeds to generate the faulted keystream.

In the beginning of the attack, at the time moment t_0 the inner state IS_{t_0} can be represented by 2 sequences: $\{b_i\}_{i=0}^{79}, \{s_i\}_{i=0}^{79}$. For simplicity, at time moment t_0 we set $i = 0$.

We distinguish four different situations, depending on the affected register and on type of the fault.

A. Stopping the LFSR for one clock

Let us now assume that we stop the clocking of LFSR for one clock, i.e. we clock only the NFSR and produce the output. The inner state of Grain v1 changes as follows:

$$\begin{gathered} (b_0, b_1, ..., b_{79}, s_0, s_1, ..., s_{79}) \\ \downarrow \\ (b_1, b_2, ..., b_{80}^A, s_0, s_1, ..., s_{79}) \end{gathered}$$

We use the exponent ^{A} to distinguish between the original unfaulted bits and bits affected by stopping the linear register. In this case, the new feedback bit b_{80}^A is equal to b_{80} and the affected output bit z_0^A is equal to z_0. However, in the next clock, with LFSR clocking properly,

$$\begin{gathered} (b_1, b_2, ..., b_{80}^A, s_0, s_1, ..., s_{79} \\ \downarrow \\ (b_2, b_3, ..., b_{81}^A, s_1, s_2, ..., s_{80}^A) \end{gathered}$$

the new feedback bit b_{81}^A is

$$\begin{aligned} b_{81}^A =& s_0 + b_{63} + b_{61} + b_{53} + b_{46} + b_{38} + b_{34} + b_{29} + b_{22} \\ &+ b_{15} + b_{10} + b_1 + b_{64}b_{61} + b_{38}b_{34} + b_{16}b_{10} \\ &+ b_{61}b_{53}b_{46} + b_{36}b_{29}b_{22} + b_{64}b_{46}b_{29}b_{10} + b_{61}b_{53}b_{38}b_{34} \\ &+ b_{64}b_{61}b_{22}b_{16} + b_{64}b_{61}b_{53}b_{46}b_{38} \\ &+ b_{34}b_{29}b_{22}b_{16}b_{10} + b_{53}b_{46}b_{38}b_{34}b_{29}b_{22} \end{aligned}$$

and $b_{81}^A + b_{81} = s_0 + s_1$, i.e. they differ. Also the output bit z_1^A is

$$z_1^A = \sum_{k \in K} b_{1+k} + h(s_3, s_{25}, s_{46}, s_{64}, b_{64})$$

The feedback bits of LFSR are unaffected by this fault, since they depend only on bits of LFSR, so $s_{80+i}^A = s_{80+i}$.

In general, if the linear register is stopped for one clock, then the equations are as follows:

$$b_i^A = b_i, 0 \leq i \leq 80 \tag{4}$$

$$z_0^A = z_0 \tag{5}$$

$$s_i^A = s_i, i \geq 0 \tag{6}$$

$$\begin{aligned} b_{i+80}^A =& s_{i-1}^A + b_{i+62}^A + b_{i+60}^A + b_{i+52}^A + b_{i+45}^A + b_{i+37}^A \\ &+ b_{i+33}^A + b_{i+28}^A + b_{i+21}^A + b_{i+14}^A + b_{i+9}^A + b_i^A \\ &+ b_{i+63}^A b_{i+60}^A + b_{i+37}^A b_{i+33}^A + b_{i+15}^A b_{i+9}^A \\ &+ b_{i+60}^A b_{i+52}^A b_{i+45}^A + b_{i+35}^A b_{i+28}^A b_{i+21}^A \\ &+ b_{i+63}^A b_{i+45}^A b_{i+28}^A b_{i+9}^A + b_{i+60}^A b_{i+52}^A b_{i+37}^A b_{i+33}^A \\ &+ b_{i+63}^A b_{i+60}^A b_{i+21}^A b_{i+15}^A + b_{i+63}^A b_{i+60}^A b_{i+52}^A b_{i+45}^A b_{i+37}^A \\ &+ b_{i+33}^A b_{i+28}^A b_{i+21}^A b_{i+15}^A b_{i+9}^A \\ &+ b_{i+52}^A b_{i+45}^A b_{i+37}^A b_{i+33}^A b_{i+28}^A b_{i+21}^A, i \geq 1 \end{aligned} \tag{7}$$

$$z_i^A = \sum_{k \in \mathcal{A}} b_{i+k}^A + h(s_{i+2}^A, s_{i+24}^A, s_{i+45}^A, s_{i+63}^A, b_{i+63}^A), i \geq 1 \tag{8}$$

All the above equations are over $GF(2)$.

In this case, the number of variables increases after each clock of the cipher. In the beginning at time t_0 we have 160 variables representing the state IS_{t_0}. During the first clock, when the LFSR is not clocking, we get one new variable - the new bit of NFSR. Afterwards, after each clock we get 2 new variables. Therefore, after N clocks of Grain v1 we have $2N - 1 + 160$ variables in our system.

The number of equations also depends on the number of clocks. After the first faulted clock, we get equations (4) and (5). Afterwards, after each clock we get one linear equation of type (6) two non-linear equations of type (7) and (8), i.e. after N clocks we get $3N - 1$ equations.

B. Forcing one extra clock in the LFSR

Let us now assume that we force the LFSR to clock one extra time, while the NFSR does not clock and also the cipher does not generate an output bit. The inner state of Grain v1 changes as follows:

$$\begin{gathered} (b_0, b_1, ..., b_{79}, s_0, s_1, ..., s_{79}) \\ \downarrow \\ (b_0, b_1, ..., b_{79}, s_1, s_2, ..., s_{80}^B) \end{gathered}$$

We use the exponent ^{B} to distinguish between the original unfaulted bits and bits affected by forcing the linear register to clock. The feedback bit s_{80}^B is equal to s_{80}, the NFSR does not clock and also the cipher does not generate an output bit. In the first clock after the fault induction

$$\begin{gathered} (b_0, b_1, ..., b_{79}, s_1, s_2, ..., s_{80}^B) \\ \downarrow \\ (b_1, b_2, ..., b_{80}^B, s_2, s_3, ..., s_{81}^B) \end{gathered}$$

the new feedback bit b_{80}^B is

$$\begin{aligned} b_{80}^B =& s_1 + b_{62} + b_{60} + b_{52} + b_{45} + b_{37} + b_{33} + b_{28} + b_{21} + b_{14} + \\ & b_9 + b_i + b_{63}b_{60} + b_{37}b_{33} + b_{15}b_9 + b_{60}b_{52}b_{45} + b_{35}b_{28}b_{21} + \\ & b_{63}b_{45}b_{28}b_9 + b_{60}b_{52}b_{37}b_{33} + b_{63}b_{60}b_{21}b_{15} + b_{63}b_{60}b_{52}b_{45}b_{37} + \\ & b_{33}b_{28}b_{21}b_{15}b_9 + b_{52}b_{45}b_{37}b_{33}b_{28}b_{21} \end{aligned}$$

and $b_{80}^B + b_{80} = s_0 + s_1$, i.e. they differ. Also the output bit z_0^B is

$$z_0^B = \sum_{k \in K} b_k + h(s_4, s_{26}, s_{47}, s_{65}, b_{63})$$

The feedback bits of LFSR are unaffected by this fault, since they depend only on bits of LFSR, so $s_{80+i}^B = s_{80+i}$.

In general, if the linear register is forced to clock one more time, then the equations are as follows:

$$b_i^B = b_i, 0 \leq i \leq 79 \tag{9}$$

$$s_{i+80}^B = s_{i+80}, i \geq 0 \tag{10}$$

$$\begin{aligned} b_{i+80}^B =& s_{i+1}^B + b_{i+62}^B + b_{i+60}^B + b_{i+52}^B + b_{i+45}^B + b_{i+37}^B \\ &+ b_{i+33}^B + b_{i+28}^B + b_{i+21}^B + b_{i+14}^B + b_{i+9}^B + b_i^B \\ &+ b_{i+63}^B b_{i+60}^B + b_{i+37}^B b_{i+33}^B + b_{i+15}^B b_{i+9}^B \\ &+ b_{i+60}^B b_{i+52}^B b_{i+45}^B + b_{i+35}^B b_{i+28}^B b_{i+21}^B \\ &+ b_{i+63}^B b_{i+45}^B b_{i+28}^B b_{i+9}^B + b_{i+60}^B b_{i+52}^B b_{i+37}^B b_{i+33}^B \\ &+ b_{i+63}^B b_{i+60}^B b_{i+21}^B b_{i+15}^B + b_{i+63}^B b_{i+60}^B b_{i+52}^B b_{i+45}^B b_{i+37}^B \\ &+ b_{i+33}^B b_{i+28}^B b_{i+21}^B b_{i+15}^B b_{i+9}^B \\ &+ b_{i+52}^B b_{i+45}^B b_{i+37}^B b_{i+33}^B b_{i+28}^B b_{i+21}^B, i \geq 0 \end{aligned} \tag{11}$$

$$z_i^B = \sum_{k \in \mathcal{A}} b_{i+k}^B + h(s_{i+4}^B, s_{i+26}^B, s_{i+47}, s_{i+65}^B, b_{i+63}^B), i \geq 0 \tag{12}$$

All the above equations are over $GF(2)$.

In this fault, the number of variables increases after each clock of the cipher. In the beginning at time t_0 we have 160 variables representing the state IS_{t_0}. During the first extra clock of LFSR, we get one new variable - the new bit of LFSR.

Afterwards, after each clock we get 2 new variables. Therefore, after N regular clocks of Grain v1 we have $2N + 1 + 160$ variables in our system.

The number of equations also depends on the number of clocks. After the first extra clock, we get equation (10) for $i = 0$. Afterwards, after each clock we get one linear equation of type (10) and two non-linear equations of type (11) and (12), i.e. after N regular clocks we get $3N + 1$ equations.

C. Stopping the NFSR for one clock

This fault is an analogy to the stopping of the linear register, so we will just state the corresponding equations, while using the exponent ^{C} to distinguish between the original unfaulted bits and bits affected by stopping the non-linear register for one clock:

$$b_i^C = b_i, 0 \leq i \leq 79 \tag{13}$$

$$z_0^C = z_0 \tag{14}$$

$$s_i^C = s_i, i \geq 0 \tag{15}$$

$$\begin{aligned} b_{i+80}^C =& s_{i+1}^C + b_{i+62}^C + b_{i+60}^C + b_{i+52}^C + b_{i+45}^C + b_{i+37}^C \\ &+ b_{i+33}^C + b_{i+28}^C + b_{i+21}^C + b_{i+14}^C + b_{i+9}^C + b_i^C \\ &+ b_{i+63}^C b_{i+60}^C + b_{i+37}^C b_{i+33}^C + b_{i+15}^C b_{i+9}^C \\ &+ b_{i+60}^C b_{i+52}^C b_{i+45}^C + b_{i+35}^C b_{i+28}^C b_{i+21}^C \\ &+ b_{i+63}^C b_{i+45}^C b_{i+28}^C b_{i+9}^C + b_{i+60}^C b_{i+52}^C b_{i+37}^C b_{i+33}^C \\ &+ b_{i+63}^C b_{i+60}^C b_{i+21}^C b_{i+15}^C + b_{i+63}^C b_{i+60}^C b_{i+52}^C b_{i+45}^C b_{i+37}^C \\ &+ b_{i+33}^C b_{i+28}^C b_{i+21}^C b_{i+15}^C b_{i+9}^C \\ &+ b_{i+52}^C b_{i+45}^C b_{i+37}^C b_{i+33}^C b_{i+28}^C b_{i+21}^C, i \geq 0 \end{aligned} \tag{16}$$

$$z_i^C = \sum_{k \in \mathcal{A}} b_{(i-1)+k}^C + h(s_{i+3}^C, s_{i+25}^C, s_{i+46}^C, s_{i+64}^C, b_{i+62}^C), i \geq 1 \tag{17}$$

All the above equations are over $GF(2)$.

Similarly to the case of stopping the LFSR, after N clocks of Grain v1 we have $2N - 1 + 160$ variables in a system of $3N - 1$ equations.

D. Forcing one extra clock in the NFSR

This fault is an analogy to the forcing of one extra clock in the linear register, so we will just state the corresponding equations, while using the exponent ^{D} to distinguish between the original unfaulted bits and bits affected by forcing the non-linear register to one extra clock:

$$b_i^D = b_i, 0 \leq i \leq 80 \tag{18}$$

$$s_{i+80}^D = s_{i+80}, i \geq 0 \tag{19}$$

$$\begin{aligned} b_{i+80}^D =& s_{i-1}^D + b_{i+62}^D + b_{i+60}^D + b_{i+52}^D + b_{i+45}^D + b_{i+37}^D \\ &+ b_{i+33}^D + b_{i+28}^D + b_{i+21}^D + b_{i+14}^D + b_{i+9}^D + b_i^D \\ &+ b_{i+63}^D b_{i+60}^D + b_{i+37}^D b_{i+33}^D + b_{i+15}^D b_{i+9}^D \\ &+ b_{i+60}^D b_{i+52}^D b_{i+45}^D + b_{i+35}^D b_{i+28}^D b_{i+21}^D \\ &+ b_{i+63}^D b_{i+45}^D b_{i+28}^D b_{i+9}^D + b_{i+60}^D b_{i+52}^D b_{i+37}^D b_{i+33}^D \\ &+ b_{i+63}^D b_{i+60}^D b_{i+21}^D b_{i+15}^D + b_{i+63}^D b_{i+60}^D b_{i+52}^D b_{i+45}^D b_{i+37}^D \\ &+ b_{i+33}^D b_{i+28}^D b_{i+21}^D b_{i+15}^D b_{i+9}^D \\ &+ b_{i+52}^D b_{i+45}^D b_{i+37}^D b_{i+33}^D b_{i+28}^D b_{i+21}^D, i \geq 1 \end{aligned} \tag{20}$$

$$z_i^D = \sum_{k \in \mathcal{A}} b_{i+1+k}^D + h(s_{i+3}^D, s_{i+25}^D, s_{i+46}^D, s_{i+64}^D, b_{i+64}^D), i \geq 0 \tag{21}$$

All the above equations are over $GF(2)$.

Similarly to the case of forcing the LFSR to go one extra clock, after N clocks of Grain v1 we have $2N + 1 + 160$ variables in a system of $3N + 1$ equations.

E. Attack outline

The goal of our attack is to find the inner state IS_{t_0} of the cipher at a time moment t_0 after the initialization phase of Grain v1. At first, we generate N bits of keystream and add the corresponding equations (1), (2), (3), into a system of equations S. Then we reset the device to the state IS_{t_0}, induce a clocking fault into one of the registers and again generate N bits of keystream and add the corresponding equations for faulted keystream, LFSR and NFSR bits (equations (4) - (21)) into the system S. If necessary, we reset the device again and phase-shift a different register and repeat the process. The number of different phase-shifts/clocking-faults in our attack is denoted m. We then try to solve the system of non-linear equations over $GF(2)$ by using SAT-solvers. If a solution is found, i.e. we know the inner state IS_{t_0}, we can clock the cipher backwards into the initial state and directly read the secret key, which is the initial value of NFSR's bits. The algorithm is presented in Algorithm 1.

IV. Results

We used the free open-source mathematics software system SAGE[1] for implementation of our experiments. SAGE provides support for polynomials over $GF(2)$ and an implementation of a SAT-solver - Cryptominisat 2.9.6. All experiments were performed on a standard PC (OS Cent OS 6 (Linux), CPU Intel i5-4200M @ 3.00GHz, 3 GB RAM).

In each experiment we tried to find the inner state of the cipher IS_{t_0} in time moment t_0 in which the faults were induced, by successfully solving a system of non-linear equations describing the inner states of the cipher. We were interested in fast attacks where the solution of the system of equations could be found in under 12 hours in a standard PC, i.e. if the SAT-solver did not find the solution in 12 hours, we terminated the solver and regarded the attack as not successful.

[1]Available at http://www.sagemath.org

Algorithm 1: Phase-shift Attack on Grain v1

Input: The number of phase-shifts m, the number of keystream bits N

Output: Secret K or empty set if K was not found.

1: Set the system of equations $S \leftarrow \emptyset$
2: Get the device into an unknown inner state IS_{t_0}
3: Generate N bits of keystream from the inner state IS_{t_0}
4: Insert equations (1),(2),(3) into the system S.
5: **for** i = 1 **to** m **do**
6: Reset the device to the state IS_{t_0}.
7: Induce a clocking fault into one of the registers.
8: Generate N bits of faulted keystream.
9: Insert corresponding equations from (4) - (21) into the system S.
10: **end for**
11: Solution $\leftarrow$ SAT_solver(S)
12: **if** Solution $\neq \emptyset$ **then**
13: Clock the device backwards from state IS_{t_0} until you reach the initial state.
14: Read secret key K from the first register.
15: **return** K.
16: **else**
17: **return** $\emptyset$.
18: **end if**

A. All four desynchronizations, no known bits

As a first experiment we induced four different phase-shift faults, i.e. forcing the LFSR to clock one extra time, forcing the NFSR to clock one extra time, stopping the LFSR for one clock, stopping the NFSR for one clock. In all four cases and also in the unfaulted case we generated 150 bits per keystream and tried to solve the corresponding system of equations.

Unfortunately, in this case, we were unable to obtain a solution of the system in under 12 hours, so we terminated the computation and regarded the attack unsuccessful. As a next approach, we opted for trying to solve a system of equations if some of the bits of LFSR at the time moment t_0 are known.

B. All four desynchronizations, known bits of LFSR

In this experiment we again induced four different synchronization faults, generated 150 bits per keystream, but also assumed that we know part of the bits of the LFSR, starting from s_0 to s_L (so $s_0, s_1, s_2, ..., s_L$). We constructed the corresponding equation system and again applied Cryptominisat to solve it. It stems from our experiments that if we know at least 28 bits of the linear register then we are able to find the inner state IS_{t_0} in under 12 hours.

The resulting times it took our computer to find the inner state are presented in Table I and Figure 2. The case $L = 79$ means that we basically know the content of the LFSR beforehand, in the case $L = 27$ we know the bits $s_0, s_1, ..., s_{27}$. We repeated the experiment 10 times and averaged the results.

The memory complexity of the attack is $M = 15N$ equations in $10N+160$ variables for N bits of keystream. The data complexity is $D = 5N$ keystream bits. In our case that is 2250 equations in 1660 variables from 750 bits of keystream.

TABLE I
AVERAGE SOLUTION TIMES FOR DIFFERENT NO. OF KNOWN BITS

L	Avg time [s]	L	Avg time [s]	L	Avg time [s]
27	42059	45	41	63	5
29	13840	47	21	65	5
31	9820	49	12	67	4
33	935	51	7	69	4
35	1055	53	6	71	4
37	360	55	6	73	5
39	157	57	5	75	2
41	102	59	5	77	2
43	56	61	5	79	2

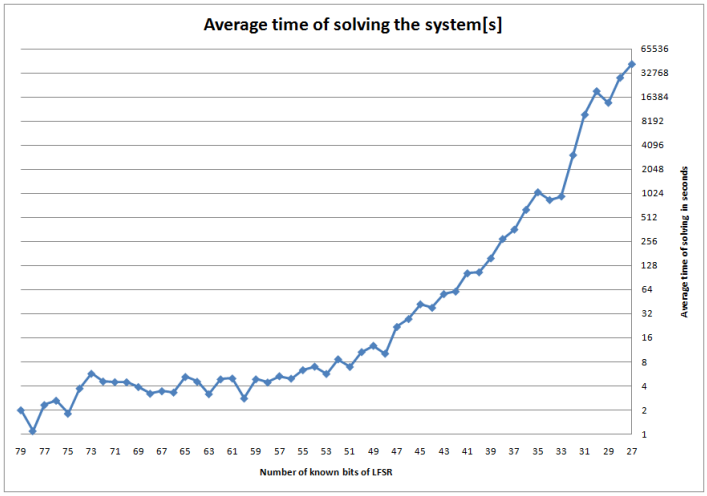

Fig. 2. Average solution times for different number of known bits

C. Three desynchronizations, known bits of LFSR

In this experiment, we induced only three different faults, i.e. we performed four different variants of the attack from the previous section, where we always omitted a different fault. Our goal was to experimentally determine, which fault had the largest impact on the attack - omitting which fault would lead to the largest increase in the time needed to solve the system of equations. We generated 150 bits per keystream and averaged 100 measurements.

The results are presented in Table II and Figure 3, all times are measured in seconds. As before, L denotes the number of known bits of LFSR. Columns denoted A, B, C, and D are denoted accordingly to the fault, which was omitted in the experiment from all possible four types of desynchronizations - the letters correspond to subsections of section 3 of this paper.

We see that the times are fairly similar, so it is rather difficult to estimate if omitting a certain type of a fault leads to a longer solution time than for a different type of a fault. For 50 known bits ($L = 49$), we can see that the longest time was needed in the case when we did not consider the stopping of the LFSR from clocking (column A).

D. One desynchronizations, known bits of LFSR

In the last experiment, we experimented with inducing only one fault and investigated what is the time needed to successfully solve the system of equations. We again used 150

TABLE II
AVERAGE SOLUTION TIMES IN SECONDS FOR DIFFERENT NO. OF KNOWN BITS FOR DIFFERENT OMITTED FAULTS

L	A	B	C	D
79	2.75	2.78	2.99	3.16
76	3.05	3.48	3.22	3.57
73	3.85	4.10	3.52	4.15
70	3.82	4.50	4.03	4.69
67	4.14	4.61	3.87	4.90
64	4.12	4.88	4.49	3.96
61	4.44	4.42	4.74	4.50
58	5.04	5.03	4.82	4.93
55	6.09	6.32	5.65	6.25
52	7.69	7.67	7.45	8.13
49	13.10	11.22	12.21	11.60

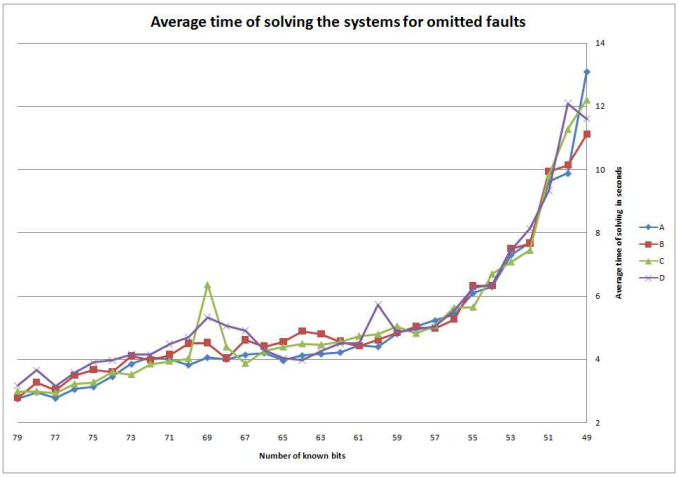

Fig. 3. Average solution times for different omitted faults

bits per keystream. This time, we only investigated the case when all the bits of LFSR are known ($L = 79$). We averaged the times from 100 experiments. In the table, columns are labeled A, B, C, and D, according to the type of the fault which was induced into the inner state of the cipher. The resulting times are presented in Table III.

We see that the solution was found in the shortest time when we stopped the linear feedback shift register for one clock (column A). This is in accordance with the results of the previous experiment, therefore we assume that this type of a desynchronization fault has the largest impact on the success of the attack, i.e. on the time the SAT-solver needs to find the inner state of the cipher from the set of equations.

V. CONCLUSION

In this paper we presented the results of our simulation of the phase-shift fault analysis of stream cipher Grain v1 combined with methods of algebraic cryptanalysis - SAT solvers. Our experiments show that the induction of 4 possible desynchronization faults: stopping the LFSR for one clock, forcing the LFSR to clock one more time, stopping the NFSR for one clock and forcing the NFSR to clock one more time can lead to an attack that requires 150 bits of keystream per each generated sequence and 28 known bits of LFSR to successfully find the inner state of the cipher IS_{t_0}, which can then be clocked backwards to find the secret key K.

We also investigated which type of desynchronization fault might be the most useful from the attacker's point of view. It stems from our experiments that the stopping of the linear feedback shift register for one clock is the best choice, since if we considered only this fault and solved the system of corresponding equations, the overall time was the lowest and if we omitted only this fault from the set of all possible four, the overall time of solving the system was the highest from the ones considered.

Of course, the physical implementation of phase-shifting is an issue and so far no implementation of this type of an attack has been published in the open literature. In [4] the authors argue that such an attack could be mounted on a device in "test" mode, where the tester is usually able to run single components of the device alone, or that such an attack could be implemented as a "backdoor" of the device by the manufacturing company.

If the device with its wirings and clocking is securely implemented, then this fault attack is of little to no concern to the user. However, if the attacker is able to induce this type of a fault and may manipulate with the clocking of the individual components, then it can lead to an attack with low data and memory complexity.

TABLE III
AVERAGE SOLUTION TIMES IN SECONDS FOR DIFFERENT INDUCED FAULTS

L	A	B	C	D
79	166.12	213.14	112.79	191.77

ACKNOWLEDGMENT

The authors would like to thank to anonymous reviewers for helpful suggestions and remarks how to improve the paper.

REFERENCES

[1] BANIK, S. - MAITRA, S. - SARKAR, S. A Differential Fault Attack on the Grain Family of Stream Ciphers. In *CHES*. 2012. p. 122-139.
[2] BANIK, S. - MAITRA, S. - SARKAR, S. A Differential Fault Attack on the Grain Family under Reasonable Assumptions. In *Indocrypt*. 2012. p. 191-208.
[3] BANIK, S. - MAITRA, S. - SARKAR, S. Differential Fault Attack against Grain family with very few faults and minimal assumptions. In *IEEE Transactions on Computers*, 2015, 64.6: 1647-1657.
[4] GOMUŁKIEWICZ, M., et al. Synchronization Fault Cryptanalysis for Breaking A5/1. In *Experimental and Efficient Algorithms.* Springer Berlin Heidelberg, 2005, p. 415-427.
[5] HELL, M. - JOHANSSON, T. - MEIER, W.. Grain: a stream cipher for constrained environments. In *International Journal of Wireless and Mobile Computing*, 2007, 2.1: 86-93.
[6] HOCH, J. - SHAMIR, A.. Fault analysis of stream ciphers. In: *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer Berlin Heidelberg, 2004. p. 240-253.
[7] HROMADA, V. - VARGA, J. Phase-shift Fault Analysis of Trivium. In *Studia Scientiarum Mathematicarum Hungarica*, 2015, 52.2: 205-220.
[8] LOE, C. W. - KHOO, K. Side Channel Attacks on Irregularly Decimated Generators. In *Information Security and Cryptology - ICISC 2007*, Springer Berlin Heidelberg, 2007, p. 116-130.