Phase-shift Fault Analysis of Grain-128

Hridya P R and Jimmy Jose

Department of Computer Science and Engineering,
National Institute of Technology Calicut, India
{hridya_p170025cs, jimmy}@nitc.ac.in

Abstract. Phase-shift fault attack is a type of fault attack used for cryptanalysis of stream ciphers. It involves clocking a cipher's feedback shift registers out of phase, in order to generate faulted keystream. Grain-128 cipher [14] is a 128-bit modification of the Grain cipher [15] which is one of the finalists in the eSTREAM project [2]. In this work, we propose a phase-shift fault attack against Grain-128 loaded with key-IV pairs that result in an all-zero LFSR after initialisation. We frame equations in terms of the input and output bits of the cipher and solve them using a SAT solver. By correctly guessing 40 innerstate bits, we are able to recover the entire 128-bit key with just 2 phase-shift faults for keystreams of length 200 bits.

Keywords: Grain-128 · Stream ciphers · Fault analysis · Hardware .

1 Introduction

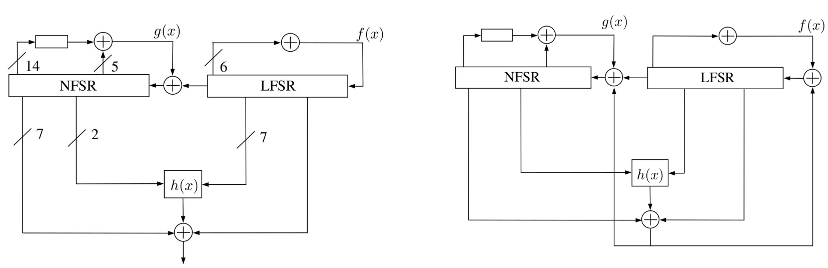
Grain [15] is a hardware oriented stream cipher which is suitable for hardware applications with restricted resources such as limited storage, gate count, or power consumption. Stream ciphers generate pseudorandom keystreams from a short random key. This keystream is simply xored with the plaintext to get the ciphertext. If an attacker could recover the short key, the keystream can be generated by running the key generation algorithm which is public. By the property of xor, the attacker can get the original plaintext by xoring the ciphertext with the keystream. Therefore, it should be difficult to recover the short key of a stream cipher.

Grain-128 [14] is an extension of the Grain cipher. Grain-128 was introduced to overcome attacks that take advantage of short keys. For example, time-memory-data trade-off attack [9] of complexity $O(2^{\frac{k}{2}})$ where k is the key size of a cipher, can attack Grain-v1 [15] of key size 80 with complexity $O(2^{40})$. There are many works in the literature that mounted fault attacks against Grain-128 either by targeting its LFSR [11] or NFSR [19] or both [21,7,6,5,13]. These fault attacks require inducing faults in the bits of registers (bit-flipping faults). This requires additional effort for finding the location of the fault since inducing faults at an intended location may not be practical. There is another type of fault attack called phase-shift fault attack which was suggested in [16] and was applied against Trivium [10] in [18] and Grain-v1 in [17].

The number of faults required for an attack signifies the number of times the device has to be reset and the number of keystream bits required signifies the data complexity required for the attack. Therefore, lesser the number of faults and number of keystream bits required, the better the attack. The phase-shift fault analysis of Trivium in [18] required only 2 phase-shifts and needed to produce only 120 bits per keystream instead of 2 bit-flips and 420 bits per keystream in bit-flipping fault analysis [20]. By introducing bit-flipping faults in Grain-128, minimum 4 faults and 256 bits per keystram were required to recover key [21]. In this work, phase-shift attack is mounted against Grain-128. Only two phase-shifts and 200 bits per keystream is sufficient for recovering the key in case of Grain-128 loaded with a weak key-IV pair. A key-IV pair that results in an all-zero LFSR after initialisation is termed as weak key-IV pair. About 40 innerstate bits are guessed to make the computation feasible. The term 'innerstate' refers to the state of LFSR and NFSR at a particular point of time.

2 Design of Grain-128

Grain-128 is made up of two 128-bit registers, an LFSR (linear feedback shift register) and an NFSR (nonlinear feedback shift register) as illustrated in Figure 1a. The NFSR is initialised with the key and the first 96 bits of LFSR is initialised with the initialisation vector (IV). The rest of the bits in LFSR is initialised with ones. The shifts of both these registers are synchronised using clocks. At each clock/step the register bits are shifted by one position. One bit gets discarded at one end and one bit is added at the other end of the register. The bit to be added is determined by the feedback function of the register. For one register, the feedback function is linear and for the other register, the function is nonlinear and hence they are named LFSR and NFSR respectively. Moreover before generating keystream, the cipher is clocked 256 times in the initialisation phase as shown in Figure 1b. During this phase, the output bit produced is fedback and xored with the input of both LFSR and NFSR.



(a) An overview of the cipher [14] (b) Initialisation [14]

Fig. 1: Design of Grain-128

3 Related Works

As mentioned in Section 1, fault attacks against Grain-128 was carried out by targeting either LFSR or NFSR or both. A fault attack by targeting LFSR was proposed in [11] . The differences between original keystream and faulted keystream was used to formulate linear equations on the LFSR state bits. After solving LFSR completely using the above equations, linear equations in NFSR could be obtained. Such equations were solved using Gröbner basis [4]. The average number of faults to be injected in LFSR for completely recovering the innerstate was calculated as 24. The highlights of this work are fault position determination and determination of innerstate at a particular time and computation of previous innerstates from that innerstate.

Similarly a fault attack by targeting NFSR was proposed in [19], assuming that the attacker can repeat Grain-128 algorithm using different IVs with the same key. Fault traces were precomputed and was used for deriving values of NFSR bits involved in feedback and output keystreams. On an average, 3.40 bits of NFSR were obtained from one fault. When NFSR bits were completely known, LFSR bits were calculated from the output differential. For this, equations involving LFSR bits were solved using Gaussian elimination. The attack require at most 312 faults.

Efforts were made in [7,6,5] for improving differential fault attack against the Grain family of ciphers and a refined attack of this type can be found in [21]. In this work, the attack assumes very less capabilities for the attacker and perform differential fault analysis. The injection of faults and their timing are very difficult to control. Majority of the works dealing with fault attacks assume that the attacker can inject faults in either LFSR or in the NFSR. This work uses four signature vectors to determine the fault location and can determine the location regardless of the register where the faults are injected. This means that the fault can be injected in either of these feedback registers. Moreover, the faults need not be injected immediately before the cipher starts generating keystream. The attacker can inject faults at any time after the initialisation phase. This attack requires a minimum of 4 faults. Another attack in [13] attempted to improve upon the attack in [21] considering an observation that the faults affected the 128th bit of the NFSR most of the time.Table 1 summarises bit flipping fault attacks against Grain-128.

Phase-shift analysis of Trivium cipher was carried out in [18]. The work compared phase-shifting attack and bit-flipping attack combined with phase-shift. At least two phase-shifts were required for a successfull attack of Trivium. Even with one bit-flip, one phase-shift could not successfully attack Trivium. The authors used SAT solvers and ElimLin algorithm [12] with Gröbner bases to solve equations in state bits.

Another interesting work along the same direction is phase-shift analysis of Grain-v1 in [17]. This work discussed four different ways in which phase-shift can be achieved and compared their results. Stopping the LFSR for one clock and producing the output was found to be the best case. The authors also experimented applying all types of phase-shifts in a single experiment. Some of the LFSR bits after initialisation (i.e., at least 28 bits) were assumed to be known in order to find out the rest of the bits.

Table 1: Comparison of Related Works

attack	targeted register	no. of faults	no. of bits per keystream	no. of equations	solved using
bit flipping fault[11]	LFSR	24	not specified	not specified	Gröbner basis
bit flipping fault[19]	NFSR	256	256	128	Gaussian elimination
bit flipping fault[21]	LFSR and NFSR	4	256	3840	SAT solver

4 Methodology

Phase-shift fault analysis involves clocking one of the shift registers ahead of the other and observing the changes in the output keystream. In this work, we choose LFSR to be phase-shifted, i.e., we clock LFSR without clocking NFSR, after the initialisation phase. Thereafter we clock both shift registers together for generating keystreams. LFSR is chosen because its feedback is not affected by the state of any of the bits of NFSR. On the other hand, NFSR feedback is dependent on the first bit of LFSR. Phase-shift attack can be carried out only if the following assumptions are true.
1. The attacker should be able to generate keystreams of any length.
2. The attacker should be able to clock LFSR without clocking NFSR when needed.
3. The attacker should be able to reset the device.

We represent the inner state after initialisation phase of LFSR and NFSR as $(x_1, x_2, ..., x_{128})$ and $(y_1, y_2, ..., y_{128})$ respectively. The output keystream is represented by $(o_1, o_2, ..., o_n)$ where n is the number of keystream bits to be produced. The feedback function of LFSR is

$$x_{i+128} = x_i + x_{i+7} + x_{i+38} + x_{i+70} + x_{i+81} + x_{i+96} \tag{1}$$

and that of NFSR is

$$\begin{aligned} y_{i+128} = x_i + y_i + y_{i+26} + y_{i+56} + y_{i+91} + y_{i+96} + y_{i+3}y_{i+67} + y_{i+11}y_{i+13} \\ + y_{i+17}y_{i+18} + y_{i+27}y_{i+59} + y_{i+40}y_{i+48} + y_{i+61}y_{i+65} + y_{i+68}y_{i+84} \end{aligned} \tag{2}$$

The output keystream is calculated using a Boolean function h(x) defined as

$$h(x) = y_{i+12}x_{i+8} + x_{i+13}x_{i+20} + y_{i+95}x_{i+42} + x_{i+60}x_{i+79} + y_{i+12}y_{i+95}x_{i+95} \tag{3}$$

The output bit is calculated as follows:

$$o_i = y_{i+2} + y_{i+15} + y_{i+36} + y_{i+45} + y_{i+64} + y_{i+73} + y_{i+89} + h(x) + x_{i+93} \quad (4)$$

In all the above equations, '+' refers to addition in GF(2).

During phase-shift, if we are clocking LFSR alone for one time, i.e., one phase-shift, the corresponding keystream is represented by $({o_1}^1, {o_2}^1, ..., {o_n}^1)$ and the NFSR feedback and the output bit, changes as shown by the following equations:

$$\begin{aligned} {y_{i+128}}^1 = {x_{i+1}}^1 + {y_i}^1 + {y_{i+26}}^1 + {y_{i+56}}^1 + {y_{i+91}}^1 + {y_{i+96}}^1 + {y_{i+3}}^1{y_{i+67}}^1 \\ +{y_{i+11}}^1{y_{i+13}}^1 + {y_{i+17}}^1{y_{i+18}}^1 + {y_{i+27}}^1{y_{i+59}}^1 + {y_{i+40}}^1{y_{i+48}}^1 + {y_{i+61}}^1{y_{i+65}}^1 \\ +{y_{i+68}}^1{y_{i+84}}^1 \end{aligned} \quad (5)$$

$$\begin{aligned} {h(x)}^1 = {y_{i+12}}^1{x_{i+9}}^1 + {x_{i+14}}^1{x_{i+21}}^1 + {y_{i+95}}^1{x_{i+43}}^1 + {x_{i+61}}^1{x_{i+80}}^1 \\ +{y_{i+12}}^1{y_{i+95}}^1{x_{i+96}}^1 \end{aligned} \quad (6)$$

$$\begin{aligned} {o_i}^1 = {y_{i+2}}^1 + {y_{i+15}}^1 + {y_{i+36}}^1 + {y_{i+45}}^1 + {y_{i+64}}^1 + {y_{i+73}}^1 + {y_{i+89}}^1 + {h(x)}^1 \\ +{x_{i+94}}^1 \end{aligned} \quad (7)$$

If we are doing two phase-shifts, the corresponding keystream is represented by $({o_1}^2, {o_2}^2, ..., {o_n}^2)$ and the NFSR feedback and the output bit, changes as shown by the following equations:

$$\begin{aligned} {y_{i+128}}^2 = {x_{i+2}}^2 + {y_i}^2 + {y_{i+26}}^2 + {y_{i+56}}^2 + {y_{i+91}}^2 + {y_{i+96}}^2 + {y_{i+3}}^2{y_{i+67}}^2 \\ +{y_{i+11}}^2{y_{i+13}}^2 + {y_{i+17}}^2{y_{i+18}}^2 + {y_{i+27}}^2{y_{i+59}}^2 + {y_{i+40}}^2{y_{i+48}}^2 \\ +{y_{i+61}}^2{y_{i+65}}^2 + {y_{i+68}}^2{y_{i+84}}^2 \end{aligned} \quad (8)$$

$$\begin{aligned} {h(x)}^2 = {y_{i+12}}^2{x_{i+10}}^2 + {x_{i+15}}^2{x_{i+22}}^2 + {y_{i+95}}^2{x_{i+44}}^2 + {x_{i+62}}^2{x_{i+81}}^2 \\ +{y_{i+12}}^2{y_{i+95}}^2{x_{i+97}}^2 \end{aligned} \quad (9)$$

$$\begin{aligned} {o_i}^2 = {y_{i+2}}^2 + {y_{i+15}}^2 + {y_{i+36}}^2 + {y_{i+45}}^2 + {y_{i+64}}^2 + {y_{i+73}}^2 + {y_{i+89}}^2 + {h(x)}^2 \\ +{x_{i+95}}^2 \end{aligned} \quad (10)$$

Algorithm 1 Generate Equations

```
Require: LFSR and NFSR after initialisation
                                    ▷ set n as number of keystream bits required
                                    ▷ equations for Grain-128
for i ← 1, n do
    Generate normal output equation
    Generate feed back equation of LFSR and left shift LFSR
    Generate feed back equation of NFSR and left shift NFSR
end for
                                    ▷ equations for Grain-128 with one phase-shift
Generate normal output equation
Generate feed back equation of LFSR and left shift LFSR
for i ← 2, n do
    Generate output equation
    Generate feed back equation of LFSR and left shift LFSR
    Generate feed back equation of NFSR and left shift NFSR
end for
                                    ▷ equations for Grain-128 with two phase-shifts
Generate normal output equation
Generate feed back equation of LFSR and left shift LFSR
Generate feed back equation of LFSR and left shift LFSR
for i ← 2, n do
    Generate output equation
    Generate feed back equation of LFSR and left shift LFSR
    Generate feed back equation of NFSR and left shift NFSR
end for
```

Algorithm 1 shows how the equations can be generated for normal and phase-shifted Grain-128. Equations upto five iterations of Grain-128 in each of these modes are listed in Appendix. The values of output bits denoted by (o_1, o_2, ..., o_n) , (${o_1}^1$, ${o_2}^1$, ..., ${o_n}^1$) and (${o_1}^2$, ${o_2}^2$, ..., ${o_n}^2$) are obtained by running Grain-128 algorithm and its one phase-shift and two phase-shifts modes. These values form the right-hand side of the output equations.

Since the device is reset between the normal and phase-shifted modes of operation of the cipher, $x_i = {x_i}^1 = {x_i}^2$ and $y_i = {y_i}^1 = {y_i}^2$ holds for i in the range from 1 to 128. These are the innerstate bits after initialisation. It is after this state that the phase-shifts are done.

As only LFSR is phase-shifted, the difference in the NFSR feedback of the three modes is an outcome of the single LFSR bit that contributes to the NFSR feedback. This thought motivated us to formulate equations involving NFSR bits of the three modes. For example, the following equation is obtained by combining equations 2 and 5.

$$ {y_{i+128}}^1 = {x_{i+1}}^1 + y_{i+128} + x_i \tag{11} $$

Similarly we can combine equations 5 and 8 to get,

$$ {y_{i+128}}^2 = {x_{i+2}}^2 + {y_{i+128}}^1 + {x_{i+1}}^1. \tag{12} $$

The equations 11 and 12 holds for i in the range from 1 to 32 only, as all other terms in the pair of equations 2 and 5 and the pair 5 and 8 cancels out only in this range. Remember, $x_i = {x_i}^1 = {x_i}^2$ and $y_i = {y_i}^1 = {y_i}^2$ holds for i in the range from 1 to 128.

Our intention is to find out the innerstate of Grain-128 immediately after the initialisation using the keystreams generated by the normal and phase-shifted Grain-128. From a particular innerstate, we can compute the previous innerstate using the output bit generated from that state. By repeating this computation we can reverse the initialisation process to recover the initial state and find the key and IV loaded in NFSR and LFSR. Section VIII of [11] explains how this can be achieved.

An example for previous state computation is given below:
We have already represented the inner state after initialisation phase of LFSR and NFSR as $(x_1, x_2, ..., x_{128})$ and $(y_1, y_2, ..., y_{128})$ respectively. Now the innerstate at the last step of initialisation can be represented by $(x_0, x_1, ..., x_{127})$ and $(y_0, y_1, ..., y_{127})$ respectively. The only unknowns to find are x_0 and y_0. During initialisation phase, the output bit is xored with the inputs, both to the LFSR and to the NFSR. Let the output bit at the last step be o_0, then,

$$o_0 = y_2 + y_{15} + y_{36} + y_{45} + y_{64} + y_{73} + y_{89} + y_{12}x_8 + x_{13}x_{20} + y_{95}x_{42} + x_{60}x_{79} + y_{12}y_{95}x_{95} + x_{93}$$

From equation 1,

$$x_{128} = x_0 + x_7 + x_{38} + x_{70} + x_{81} + x_{96}$$

Rearranging we get,

$$x_0 = x_7 + x_{38} + x_{70} + x_{81} + x_{96} + x_{128}$$

Similarly from equation 2,

$$y_{128} = x_0 + y_0 + y_{26} + y_{56} + y_{91} + y_{96} + y_3y_{67} + y_{11}y_{13} + y_{17}y_{18} + y_{27}y_{59} + y_{40}y_{48} + y_{61}y_{65} + y_{68}y_{84}$$

we get,

$$y_0 = x_0 + y_{26} + y_{56} + y_{91} + y_{96} + y_3y_{67} + y_{11}y_{13} + y_{17}y_{18} + y_{27}y_{59} + y_{40}y_{48} + y_{61}y_{65} + y_{68}y_{84} + y_{128}$$

Thus the previous state is obtained.

Equations involving the innerstate bits and the bits of the keystream generated from the innerstate are generated using Algorithm 1 discussed in Section 4. There are 256 variables representing the inner state after initialization phase. At each clock cycle, 2 new variables and 3 new equations are added to the system. Let n be the number of keystream bits produced, then in normal mode there are $3n$ equations and $256 + 2n$ variables and in one phase-shift mode, there are

$3(n-1)+2$ equations (only two equations in the first step since NFSR is not clocked) and $256+2(n-1)+1$ variables (there is only LFSR feedback bit in the first step). For two phase-shift mode there are $3(n-1)+3$ equations (two feedback equations of the LFSR and the output equation in the first step) and $256+2(n-1)+2$ variables (there are 2 LFSR feedback bits in the first step). For instance, if the cipher is executed in three modes by producing 100 bits in each keystream (i.e., $n=100$), there will be 899 equations involving 855 variables in the system of equations.

This system of equations is to be solved in order to find out the innerstate bits. Solving this system of nonlinear equations involving large number of variables is a complex process. So the best method is to use SAT solvers for solving. In this work, the Grain-128 algorithm is implemented in C programming language. Lingeling solver [8] is used for solving the system of nonlinear equations.

5 Results

Equations are converted to DIMACS [1] CNF format using SAGE mathematical software system[22]. The DIMACS CNF format is widely accepted as the standard format for Boolean formulas in conjunctive normal form. SAT solvers require their input to be in CNF (Conjunctive normal form) or XCNF (Extended conjunctive normal form) format.

Lingeling solver is used to solve the equations in DIMACS format. Lingeling executed without producing any result in feasible time because solving such high degree equations involving such large number of variables is beyond the limit of the available resources. So we used a weak key-IV pair as defined in [23] and conducted experiments to find out whether we are able to recover the key from the output keystream bits. A key-IV pair that results in an all-zero LFSR after initialisation is termed as weak key-IV. Since LFSR is all zeros, the problem of recovering the innerstate after initialisation is reduced to recovering the 128-bit NFSR. There are 2^{96} such key-IV pairs. A procedure for finding out such a key-IV pair is available in [23]. Assuming that the innerstate after initialization has all zeros in the LFSR, NFSR bits are randomly selected. From this innerstate, the execution of cipher is reversed (as explained in section 4) until we get the corresponding initial state. If all the bits in the last 32 bits of the LFSR are ones in the obtained initial state, then it is a valid initial state of Grain-128 and the remaining contents of NFSR and LFSR gives the weak key-IV pair. If the obtained initial state is not a valid inner state, repeat the procedure by randomly selecting NFSR bits again.

We simulated Grain-128 in normal, one phase-shift and two phase-shift modes and generated keystream bits in each mode. Without guessing any of the innerstate bits, Lingeling is unable to solve the system of equations in feasible time. All experiments are performed on a DELL PowerEdge T620 Tower Server (CPU ES-2690v2@3.00 GHz x 40, 251.9 GB RAM). By guessing 40 bits, Lingeling could give the solution containing 256 innerstate bits (including 88 unknown NFSR bits). Then, the key could be recovered by reversing the cipher execution from the obtained innerstate as discussed in section 4. Time taken for solving the system of equations when the number of guessed bits is varied, is illustrated in Table 2.

When three phase-shift mode is also included, the larger number of variables involved and number of equations in the system cause Lingeling to take more time to solve the system. So we restrict the number of phase-shift faults to two. The experiment is repeated by varying the number of keystream bits produced (n) from 100 bits to 200 bits. Though for 100 bits of keystream, time taken for solving the system of equation is lower than for 200 bits, 100 bits is not sufficient for recovering 88 bits of NFSR which is the highest number of unknown NFSR bits that we could recover.

Table 2: Time Taken for Solving Equations

	no. of bits guessed	unknown bits of NFSR	Time taken (in seconds)
100 keystream bits			
	above 80	below 48	nearly 0
	80	48	0.280
	72	56	0.601
	64	64	1.351
	56	72	14.189
	48	80	1025.292
	40	88	didn't terminate in feasible time
200 keystream bits			
	above 80	below 48	nearly 0
	80	48	1.904
	72	56	3.842
	64	64	3.100
	56	72	9.630
	48	80	6663.131
	40	88	739407.952

In [3], algebraic cryptanalysis of Grain-128 succeeded in recovering upto 64 unknown innerstate bits. The rest 192 innerstate bits were guessed. To compare [3] with this work, we can consider the case of Grain-128 loaded with weak key-IV pair. Even then, 64 bits are to be guessed for successfully mounting attack described in [3] whereas this work requires only 40 bits to be guessed.

6 Conclusion

A less researched attack model, phase-shift fault analysis, is mounted against Grain-128. In case of Grain-128 loaded with weak key-IV, we are able to recover the key by calculating the initial state from a particular innerstate of which unknown bits can be recovered by guessing the other 40 bits. The attack requires two phase-shift faults and a keystream of 200 bits in each execution of the cipher. This work underlines the fact that although phase-shift fault attack is practically difficult to mount, the designers should consider the possibility of such an attack while designing ciphers.

References

1. DIMACS. http://dimacs.rutgers.edu/, accessed: 2021-07-17
2. The ECRYPT Stream Cipher Project eSTREAM. http://www.ecrypt.eu.org/, accessed: 2021-07-17
3. Afzal, M., Masood, A.: Algebraic cryptanalysis of a nlfsr based stream cipher. In: 2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications. pp. 1–6 (2008). https://doi.org/10.1109/ICTTA.2008.4530286
4. Ajwa, I.A., Liu, Z., Wang, P.S.: Gröbner bases algorithm. Tech. rep., ICM Technical Reports Series (ICM-199502-00 (1995)
5. Banik, S., Maitra, S., Sarkar, S.: A Differential Fault Attack on Grain-128a using MACs. In: Security, Privacy, and Applied Cryptography Engineering - Second International Conference, SPACE 2012, Chennai, India, November 3-4, 2012. Proceedings. pp. 111–125 (2012). https://doi.org/10.1007/978-3-642-34416-9_8, https://doi.org/10.1007/978-3-642-34416-9_8
6. Banik, S., Maitra, S., Sarkar, S.: A Differential Fault Attack on the Grain Family of Stream Ciphers. In: Cryptographic Hardware and Embedded Systems - CHES 2012 - 14th International Workshop, Leuven, Belgium, September 9-12, 2012. Proceedings. pp. 122–139 (2012). https://doi.org/10.1007/978-3-642-33027-8_8, https://doi.org/10.1007/978-3-642-33027-8_8
7. Banik, S., Maitra, S., Sarkar, S.: A Differential Fault Attack on the Grain Family under Reasonable Assumptions. In: Progress in Cryptology - INDOCRYPT 2012, 13th International Conference on Cryptology in India, Kolkata, India, December 9-12, 2012. Proceedings. pp. 191–208 (2012). https://doi.org/10.1007/978-3-642-34931-7_12, https://doi.org/10.1007/978-3-642-34931-7_12
8. Biere, A.: Lingeling SAT Solver. http://fmv.jku.at/lingeling/, accessed: 2021-07-17
9. Biryukov, A., Shamir, A.: Cryptanalytic time/memory/data tradeoffs for stream ciphers. In: Okamoto, T. (ed.) Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1976, pp. 1–13. Springer (2000). https://doi.org/10.1007/3-540-44448-3_1, https://doi.org/10.1007/3-540-44448-3_1
10. Cannière, C.D., Preneel, B.: Trivium. In: New Stream Cipher Designs - The eSTREAM Finalists, pp. 244–266 (2008). https://doi.org/10.1007/978-3-540-68351-3_18, https://doi.org/10.1007/978-3-540-68351-3_18
11. Castagnos, G., Berzati, A., Canovas, C., Debraize, B., Goubin, L., Gouget, A., Paillier, P., Salgado, S.: Fault Analysis of Grain-128. In: IEEE International Workshop on Hardware-Oriented Security and Trust, HOST 2009, San Francisco, CA, USA, July 27, 2009. Proceedings. pp. 7–14 (2009). https://doi.org/10.1109/HST.2009.5225030, https://doi.org/10.1109/HST.2009.5225030

12. Courtois, N.T., Sepehrdad, P., Susil, P., Vaudenay, S.: Elimlin algorithm revisited. In: Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers. pp. 306–325 (2012). https://doi.org/10.1007/978-3-642-34047-5_18, https://doi.org/10.1007/978-3-642-34047-5_18
13. Dey, P., Chakraborty, A., Adhikari, A., Mukhopadhyay, D.: Improved Practical Differential Fault Analysis of Grain-128. In: Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE 2015, Grenoble, France, March 9-13, 2015. pp. 459–464 (2015), http://dl.acm.org/citation.cfm?id=2755858
14. Hell, M., Johansson, T., Maximov, A., Meier, W.: A Stream Cipher Proposal: Grain-128. In: Proceedings 2006 IEEE International Symposium on Information Theory, ISIT 2006, The Westin Seattle, Seattle, Washington, USA, July 9-14, 2006. pp. 1614–1618 (2006). https://doi.org/10.1109/ISIT.2006.261549, https://doi.org/10.1109/ISIT.2006.261549
15. Hell, M., Johansson, T., Meier, W.: Grain: A Stream Cipher for Constrained Environments. IJWMC **2**(1), 86–93 (2007). https://doi.org/10.1504/IJWMC.2007.013798, https://doi.org/10.1504/IJWMC.2007.013798
16. Hoch, J.J., Shamir, A.: Fault Analysis of Stream Ciphers. In: Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings. pp. 240–253 (2004). https://doi.org/10.1007/978-3-540-28632-5_18, https://doi.org/10.1007/978-3-540-28632-5_18
17. Hromada, V., Petho, T.: Phase-shift Fault Analysis of Grain v1. International Journal of Electronics and Telecommunications **64**(2), 131–136 (2018)
18. Hromada, V., Varga, J.: Phase-shift Fault Analysis of Trivium. Studia Scientiarum Mathematicarum Hungarica **52**(2), 205–220 (2015)
19. Karmakar, S., Chowdhury, D.R.: Fault Analysis of Grain-128 by targeting NFSR. In: Progress in Cryptology - AFRICACRYPT 2011 - 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings. pp. 298–315 (2011). https://doi.org/10.1007/978-3-642-21969-6_19, https://doi.org/10.1007/978-3-642-21969-6_19
20. Mohamed, M.S.E., Bulygin, S., Buchmann, J.A.: Using SAT solving to improve differential fault analysis of trivium. In: Kim, T., Adeli, H., Robles, R.J., Balitanas, M.O. (eds.) Information Security and Assurance - International Conference, ISA 2011, Brno, Czech Republic, August 15-17, 2011. Proceedings. Communications in Computer and Information Science, vol. 200, pp. 62–71. Springer (2011). https://doi.org/10.1007/978-3-642-23141-4_7, https://doi.org/10.1007/978-3-642-23141-4_7
21. Sarkar, S., Banik, S., Maitra, S.: Differential Fault Attack against Grain Family with Very Few Faults and Minimal Assumptions. IEEE Trans. Computers **64**(6), 1647–1657 (2015). https://doi.org/10.1109/TC.2014.2339854, https://doi.org/10.1109/TC.2014.2339854
22. Stein, W.: Sage Mathematics Software System. http://www.sagemath.org/, accessed: 2021-07-17
23. Zhang, H., Wang, X.: Cryptanalysis of Stream Cipher Grain Family. IACR Cryptology ePrint Archive **2009**, 109 (2009), http://eprint.iacr.org/2009/109

Appendix

The feedback equations and output equations up to five iterations in the keystream generation phase of Grain-128 in different modes are given below. The symbol '+' refers to addition in GF(2) in all the equations given below.

Equations of Grain-128

The inner state after initialisation phase of LFSR and NFSR are represented as $(x_1, x_2, ..., x_{128})$ and $(y_1, y_2, ..., y_{128})$ respectively. The output keystream is represented by $(o_1, o_2, ..., o_n)$.

$x_{129} = x_1 + x_8 + x_{39} + x_{71} + x_{82} + x_{97}$
$y_{129} = x_1 + y_1 + y_{27} + y_{57} + y_{92} + y_{97} + y_4y_{68} + y_{12}y_{14} + y_{18}y_{19} +$
$y_{28}y_{60} + y_{41}y_{49} + y_{62}y_{66} + y_{69}y_{85}$
$o_1 = y_3 + y_{16} + y_{37} + y_{46} + y_{65} + y_{74} + y_{90} + y_{13}x_9 + x_{14}x_{21} + y_{96}x_{43} + x_{61}x_{80} +$
$y_{13}y_{96}x_{96} + x_{94}$

$x_{130} = x_2 + x_9 + x_{40} + x_{72} + x_{83} + x_{98}$
$y_{130} = x_2 + y_2 + y_{28} + y_{58} + y_{93} + y_{98} + y_5y_{69} + y_{13}y_{15} + y_{19}y_{20} +$
$y_{29}y_{61} + y_{42}y_{50} + y_{63}y_{67} + y_{70}y_{86}$
$o_2 = y_4 + y_{17} + y_{38} + y_{47} + y_{66} + y_{75} + y_{91} + y_{14}x_{10} + x_{15}x_{22} + y_{97}x_{44} + x_{62}x_{81} +$
$y_{14}y_{97}x_{97} + x_{95}$

$x_{131} = x_3 + x_{10} + x_{41} + x_{73} + x_{84} + x_{99}$
$y_{131} = x_3 + y_3 + y_{29} + y_{59} + y_{94} + y_{99} + y_6y_{70} + y_{14}y_{16} + y_{20}y_{21} +$
$y_{30}y_{62} + y_{43}y_{51} + y_{64}y_{68} + y_{71}y_{87}$
$o_3 = y_5 + y_{18} + y_{39} + y_{48} + y_{67} + y_{76} + y_{92} + y_{15}x_{11} + x_{16}x_{23} + y_{98}x_{45} + x_{63}x_{82} +$
$y_{15}y_{98}x_{98} + x_{96}$

$x_{132} = x_4 + x_{11} + x_{42} + x_{74} + x_{85} + x_{100}$
$y_{132} = x_4 + y_4 + y_{30} + y_{60} + y_{95} + y_{100} + y_7y_{71} + y_{15}y_{17} + y_{21}y_{22} +$
$y_{31}y_{63} + y_{44}y_{52} + y_{65}y_{69} + y_{72}y_{88}$
$o_4 = y_6 + y_{19} + y_{40} + y_{49} + y_{68} + y_{77} + y_{93} + y_{16}x_{12} + x_{17}x_{24} + y_{99}x_{46} + x_{64}x_{83} +$
$y_{16}y_{99}x_{99} + x_{97}$

$x_{133} = x_5 + x_{12} + x_{43} + x_{75} + x_{86} + x_{101}$
$y_{133} = x_5 + y_5 + y_{31} + y_{61} + y_{96} + y_{101} + y_8y_{72} + y_{16}y_{18} + y_{22}y_{23} +$
$y_{32}y_{64} + y_{45}y_{53} + y_{66}y_{70} + y_{73}y_{89}$
$o_5 = y_7 + y_{20} + y_{41} + y_{50} + y_{69} + y_{78} + y_{94} + y_{17}x_{13} + x_{18}x_{25} + y_{100}x_{47} + x_{65}x_{84} +$
$y_{17}y_{100}x_{100} + x_{98}$

Equations of Grain-128 with one phase-shift

The inner state after initialisation phase of LFSR and NFSR are represented as $(x_1{}^1, x_2{}^1, ..., x_{128}{}^1)$ and $(y_1{}^1, y_2{}^1, ..., y_{128}{}^1)$ respectively. The output keystream is represented by $(o_1{}^1, o_2{}^1, ..., o_n{}^1)$.

$x_{129}{}^1 = x_1{}^1 + x_8{}^1 + x_{39}{}^1 + x_{71}{}^1 + x_{82}{}^1 + x_{97}{}^1$
$x_{130}{}^1 = x_2{}^1 + x_9{}^1 + x_{40}{}^1 + x_{72}{}^1 + x_{83}{}^1 + x_{98}{}^1$
$y_{129}{}^1 = x_2{}^1 + y_1{}^1 + y_{27}{}^1 + y_{57}{}^1 + y_{92}{}^1 + y_{97}{}^1 + y_4{}^1 y_{68}{}^1 + y_{12}{}^1 y_{14}{}^1 + y_{18}{}^1 y_{19}{}^1 + y_{28}{}^1 y_{60}{}^1 + y_{41}{}^1 y_{49}{}^1 + y_{62}{}^1 y_{66}{}^1 + y_{69}{}^1 y_{85}{}^1$
$o_1{}^1 = y_3{}^1 + y_{16}{}^1 + y_{37}{}^1 + y_{46}{}^1 + y_{65}{}^1 + y_{74}{}^1 + y_{90}{}^1 + y_{13}{}^1 x_{10}{}^1 + x_{15}{}^1 x_{22}{}^1 + y_{96}{}^1 x_{44}{}^1 + x_{62}{}^1 x_{81}{}^1 + y_{13}{}^1 y_{96}{}^1 x_{97}{}^1 + x_{95}{}^1$

$x_{131}{}^1 = x_3{}^1 + x_{10}{}^1 + x_{41}{}^1 + x_{73}{}^1 + x_{84}{}^1 + x_{99}{}^1$
$y_{130}{}^1 = x_3{}^1 + y_2{}^1 + y_{28}{}^1 + y_{58}{}^1 + y_{93}{}^1 + y_{98}{}^1 + y_5{}^1 y_{69}{}^1 + y_{13}{}^1 y_{15}{}^1 + y_{19}{}^1 y_{20}{}^1 + y_{29}{}^1 y_{61}{}^1 + y_{42}{}^1 y_{50}{}^1 + y_{63}{}^1 y_{67}{}^1 + y_{70}{}^1 y_{86}{}^1$
$o_2{}^1 = y_4{}^1 + y_{17}{}^1 + y_{38}{}^1 + y_{47}{}^1 + y_{66}{}^1 + y_{75}{}^1 + y_{91}{}^1 + y_{14}{}^1 x_{11}{}^1 + x_{16}{}^1 x_{23}{}^1 + y_{97}{}^1 x_{45}{}^1 + x_{63}{}^1 x_{82}{}^1 + y_{14}{}^1 y_{97}{}^1 x_{98}{}^1 + x_{96}{}^1$

$x_{132}{}^1 = x_4{}^1 + x_{11}{}^1 + x_{42}{}^1 + x_{74}{}^1 + x_{85}{}^1 + x_{100}{}^1$
$y_{131}{}^1 = x_4{}^1 + y_3{}^1 + y_{29}{}^1 + y_{59}{}^1 + y_{94}{}^1 + y_{99}{}^1 + y_6{}^1 y_{70}{}^1 + y_{14}{}^1 y_{16}{}^1 + y_{20}{}^1 y_{21}{}^1 + y_{30}{}^1 y_{62}{}^1 + y_{43}{}^1 y_{51}{}^1 + y_{64}{}^1 y_{68}{}^1 + y_{71}{}^1 y_{87}{}^1$
$o_3{}^1 = y_5{}^1 + y_{18}{}^1 + y_{39}{}^1 + y_{48}{}^1 + y_{67}{}^1 + y_{76}{}^1 + y_{92}{}^1 + y_{15}{}^1 x_{12}{}^1 + x_{17}{}^1 x_{24}{}^1 + y_{98}{}^1 x_{46}{}^1 + x_{64}{}^1 x_{83}{}^1 + y_{15}{}^1 y_{98}{}^1 x_{99}{}^1 + x_{97}{}^1$

$x_{133}{}^1 = x_5{}^1 + x_{12}{}^1 + x_{43}{}^1 + x_{75}{}^1 + x_{86}{}^1 + x_{101}{}^1$
$y_{132}{}^1 = x_5{}^1 + y_4{}^1 + y_{30}{}^1 + y_{60}{}^1 + y_{95}{}^1 + y_{100}{}^1 + y_7{}^1 y_{71}{}^1 + y_{15}{}^1 y_{17}{}^1 + y_{21}{}^1 y_{22}{}^1 + y_{31}{}^1 y_{63}{}^1 + y_{44}{}^1 y_{52}{}^1 + y_{65}{}^1 y_{69}{}^1 + y_{72}{}^1 y_{88}{}^1$
$o_4{}^1 = y_6{}^1 + y_{19}{}^1 + y_{40}{}^1 + y_{49}{}^1 + y_{68}{}^1 + y_{77}{}^1 + y_{93}{}^1 + y_{16}{}^1 x_{13}{}^1 + x_{18}{}^1 x_{25}{}^1 + y_{99}{}^1 x_{47}{}^1 + x_{65}{}^1 x_{84}{}^1 + y_{16}{}^1 y_{99}{}^1 x_{100}{}^1 + x_{98}{}^1$

$x_{134}{}^1 = x_6{}^1 + x_{13}{}^1 + x_{44}{}^1 + x_{76}{}^1 + x_{87}{}^1 + x_{102}{}^1$
$y_{133}{}^1 = x_6{}^1 + y_5{}^1 + y_{31}{}^1 + y_{61}{}^1 + y_{96}{}^1 + y_{101}{}^1 + y_8{}^1 y_{72}{}^1 + y_{16}{}^1 y_{18}{}^1 + y_{22}{}^1 y_{23}{}^1 + y_{32}{}^1 y_{64}{}^1 + y_{45}{}^1 y_{53}{}^1 + y_{66}{}^1 y_{70}{}^1 + y_{73}{}^1 y_{89}{}^1$
$o_5{}^1 = y_7{}^1 + y_{20}{}^1 + y_{41}{}^1 + y_{50}{}^1 + y_{69}{}^1 + y_{78}{}^1 + y_{94}{}^1 + y_{17}{}^1 x_{14}{}^1 + x_{19}{}^1 x_{26}{}^1 + y_{100}{}^1 x_{48}{}^1 + x_{66}{}^1 x_{85}{}^1 + y_{17}{}^1 y_{100}{}^1 x_{101}{}^1 + x_{99}{}^1$

Equations of Grain-128 with two phase-shifts

The inner state after initialisation phase of LFSR and NFSR are represented as $(x_1{}^2, x_2{}^2, ..., x_{128}{}^2)$ and $(y_1{}^2, y_2{}^2, ..., y_{128}{}^2)$ respectively. The output keystream is represented by $(o_1{}^2, o_2{}^2, ..., o_n{}^2)$.

$x_{129}{}^2 = x_1{}^2 + x_8{}^2 + x_{39}{}^2 + x_{71}{}^2 + x_{82}{}^2 + x_{97}{}^2$

$x_{130}{}^2 = x_2{}^2 + x_9{}^2 + x_{40}{}^2 + x_{72}{}^2 + x_{83}{}^2 + x_{98}{}^2$
$x_{131}{}^2 = x_3{}^2 + x_{10}{}^2 + x_{41}{}^2 + x_{73}{}^2 + x_{84}{}^2 + x_{99}{}^2$
$y_{129}{}^2 = x_3{}^2 + y_1{}^2 + y_{27}{}^2 + y_{57}{}^2 + y_{92}{}^2 + y_{97}{}^2 + y_4{}^2 y_{68}{}^2 + y_{12}{}^2 y_{14}{}^2 + y_{18}{}^2 y_{19}{}^2 + y_{28}{}^2 y_{60}{}^2 + y_{41}{}^2 y_{49}{}^2 + y_{62}{}^2 y_{66}{}^2 + y_{69}{}^2 y_{85}{}^2$
$o_1{}^2 = y_3{}^2 + y_{16}{}^2 + y_{37}{}^2 + y_{46}{}^2 + y_{65}{}^2 + y_{74}{}^2 + y_{90}{}^2 + y_{13}{}^2 x_{11}{}^2 + x_{16}{}^2 x_{23}{}^2 + y_{96}{}^2 x_{45}{}^2 + x_{63}{}^2 x_{82}{}^2 + y_{13}{}^2 y_{96}{}^2 x_{98}{}^2 + x_{96}{}^2$

$x_{132}{}^2 = x_4{}^2 + x_{11}{}^2 + x_{42}{}^2 + x_{74}{}^2 + x_{85}{}^2 + x_{100}{}^2$
$y_{130}{}^2 = x_4{}^2 + y_2{}^2 + y_{28}{}^2 + y_{58}{}^2 + y_{93}{}^2 + y_{98}{}^2 + y_5{}^2 y_{69}{}^2 + y_{13}{}^2 y_{15}{}^2 + y_{19}{}^2 y_{20}{}^2 + y_{29}{}^2 y_{61}{}^2 + y_{42}{}^2 y_{50}{}^2 + y_{63}{}^2 y_{67}{}^2 + y_{70}{}^2 y_{86}{}^2$
$o_2{}^2 = y_4{}^2 + y_{17}{}^2 + y_{38}{}^2 + y_{47}{}^2 + y_{66}{}^2 + y_{75}{}^2 + y_{91}{}^2 + y_{14}{}^2 x_{12}{}^2 + x_{17}{}^2 x_{24}{}^2 + y_{97}{}^2 x_{46}{}^2 + x_{64}{}^2 x_{83}{}^2 + y_{14}{}^2 y_{97}{}^2 x_{99}{}^2 + x_{97}{}^2$

$x_{133}{}^2 = x_5{}^2 + x_{12}{}^2 + x_{43}{}^2 + x_{75}{}^2 + x_{86}{}^2 + x_{101}{}^2$
$y_{131}{}^2 = x_5{}^2 + y_3{}^2 + y_{29}{}^2 + y_{59}{}^2 + y_{94}{}^2 + y_{99}{}^2 + y_6{}^2 y_{70}{}^2 + y_{14}{}^2 y_{16}{}^2 + y_{20}{}^2 y_{21}{}^2 + y_{30}{}^2 y_{62}{}^2 + y_{43}{}^2 y_{51}{}^2 + y_{64}{}^2 y_{68}{}^2 + y_{71}{}^2 y_{87}{}^2$
$o_3{}^2 = y_5{}^2 + y_{18}{}^2 + y_{39}{}^2 + y_{48}{}^2 + y_{67}{}^2 + y_{76}{}^2 + y_{92}{}^2 + y_{15}{}^2 x_{13}{}^2 + x_{18}{}^2 x_{25}{}^2 + y_{98}{}^2 x_{47}{}^2 + x_{65}{}^2 x_{84}{}^2 + y_{15}{}^2 y_{98}{}^2 x_{100}{}^2 + x_{98}{}^2$

$x_{134}{}^2 = x_6{}^2 + x_{13}{}^2 + x_{44}{}^2 + x_{76}{}^2 + x_{87}{}^2 + x_{102}{}^2$
$y_{132}{}^2 = x_6{}^2 + y_4{}^2 + y_{30}{}^2 + y_{60}{}^2 + y_{95}{}^2 + y_{100}{}^2 + y_7{}^2 y_{71}{}^2 + y_{15}{}^2 y_{17}{}^2 + y_{21}{}^2 y_{22}{}^2 + y_{31}{}^2 y_{63}{}^2 + y_{44}{}^2 y_{52}{}^2 + y_{65}{}^2 y_{69}{}^2 + y_{72}{}^2 y_{88}{}^2$
$o_4{}^2 = y_6{}^2 + y_{19}{}^2 + y_{40}{}^2 + y_{49}{}^2 + y_{68}{}^2 + y_{77}{}^2 + y_{93}{}^2 + y_{16}{}^2 x_{14}{}^2 + x_{19}{}^2 x_{26}{}^2 + y_{99}{}^2 x_{48}{}^2 + x_{66}{}^2 x_{85}{}^2 + y_{16}{}^2 y_{99}{}^2 x_{101}{}^2 + x_{99}{}^2$

$x_{135}{}^2 = x_7{}^2 + x_{14}{}^2 + x_{45}{}^2 + x_{77}{}^2 + x_{88}{}^2 + x_{103}{}^2$
$y_{133}{}^2 = x_7{}^2 + y_5{}^2 + y_{31}{}^2 + y_{61}{}^2 + y_{96}{}^2 + y_{101}{}^2 + y_8{}^2 y_{72}{}^2 + y_{16}{}^2 y_{18}{}^2 + y_{22}{}^2 y_{23}{}^2 + y_{32}{}^2 y_{64}{}^2 + y_{45}{}^2 y_{53}{}^2 + y_{66}{}^2 y_{70}{}^2 + y_{73}{}^2 y_{89}{}^2$
$o_5{}^2 = y_7{}^2 + y_{20}{}^2 + y_{41}{}^2 + y_{50}{}^2 + y_{69}{}^2 + y_{78}{}^2 + y_{94}{}^2 + y_{17}{}^2 x_{15}{}^2 + x_{20}{}^2 x_{27}{}^2 + y_{100}{}^2 x_{49}{}^2 + x_{67}{}^2 x_{86}{}^2 + y_{17}{}^2 y_{100}{}^2 x_{102}{}^2 + x_{100}{}^2$