A Differential Fault Attack on Grain-128a using MACs

Subhadeep Banik, Subhamoy Maitra and Santanu Sarkar

Applied Statistics Unit, Indian Statistical Institute,
203 B T Road, Kolkata 700 108, India.
s.banik_r@isical.ac.in,subho@isical.ac.in,sarkar.santanu.bir@gmail.com

Abstract. The 32-bit MAC of Grain-128a is a linear combination of the first 64 and then the alternative keystream bits. In this paper we describe a successful differential fault attack on Grain-128a, in which we recover the secret key by observing the correct and faulty MACs of certain chosen messages. The attack works due to certain properties of the Boolean functions and corresponding choices of the taps from the LFSR. We present methods to identify the fault locations and then construct set of linear equations to obtain the contents of the LFSR and the NFSR. Our attack requires less than 2^{11} fault injections and invocations of less than 2^{12} MAC generation routines.

Keywords: Grain v1, Grain-128, Grain-128a, LFSR, MAC, NFSR, Stream Cipher.

1 Introduction

The Grain-128a authentication scheme was proposed in SKEW 2011 by Ågren et al. Any message in $\{0,1\}^*$ can be mapped to a 32-bit tag using this authenticated-encryption scheme. Grain-128a is essentially part of the Grain family which was first proposed by Hell, Johansson and Meier in 2005 [13] as a part of the eStream project. The physical structure of the Grain family is simple as well as elegant and has been designed so as to require low hardware complexity. In response to cryptanalysis against the initial design of the cipher, the modified versions Grain v1 [13], Grain-128 [14] and Grain-128a [2] were proposed in due course. Analysis of this cipher is an area of recent interest as evident from number of cryptanalytic results [3–12, 16–19, 22, 23].

Fault attacks are known to be very efficient against stream ciphers in general, and have received attention in recent cryptographic literature [15]. For differential fault attack scenario in stream ciphers, the attacker is allowed to inject faults in the internal state, and then by analyzing the difference in the faulty and the fault-free keystreams, one should be able to deduce the complete or partial information about the internal state/secret key. The most common method of injecting faults is by using laser shots or clock glitches [20, 21]. Though the fault attacks usually rely on optimistic assumptions and study the cipher in a model that is weaker than the original version, they are not unrealistic as evident from literature. In this paper too, the model we study is a follow up of existing state-of-the-art literature [4, 6, 16]. A detailed justification of the feasibility of such fault model is presented in [6, Section IIIB].

Grain-128 has been successfully cryptanalyzed by employing fault attacks [6, 16]. In this case, the attacker has the advantage of accessing and analyzing the entire fault-free and faulty keystreams. In Grain-128a, this is not the case as it accommodates authentication too. The scheme does not make the first 64 keystream bits available to the attacker. Thereafter the keystream bits are used for encryption and authentication alternatively. The scheme outputs 32-bit MAC of any message and this can be used by the attacker. In our work, we have described an approach to find the secret key used in the authentication scheme by observing the correct and faulty MACs of certain specific messages.

We proceed with the description of the Grain family, and in particular Grain-128a, in this section. The implementation of the attack on Grain-128a along with the fault location identification routine is explained in Section 2.

1.1 Brief description of Grain family

The exact structure of the Grain family is explained in this section. It consists of an n-bit LFSR and an n-bit NFSR. Certain bits of both the shift registers are taken as inputs to a combining Boolean function, whence the keystream is produced. The update function of the LFSR is given by the equation $y_{t+n} = f(Y_t)$, where $Y_t = [y_t, y_{t+1}, \ldots, y_{t+n-1}]$ is an n-bit vector that denotes the LFSR state at the t^{th} clock interval and ϕ is a linear function on the LFSR state bits obtained from a primitive polynomial in $GF(2)$ of degree n.

We abuse the + notation for Boolean XOR, i.e., GF(2) addition as well as standard arithmetic addition. However, that will be clear from the context.

The NFSR state is updated as $x_{t+n} = y_t + g(X_t)$. Here, $X_t = [x_t, x_{t+1}, \ldots, x_{t+n-1}]$ is an n-bit vector that denotes the NFSR state at the t^{th} clock interval and g is a non-linear function of the NFSR state bits.

The output keystream is produced by combining the LFSR and NFSR bits as $z_t = h'(X_t, Y_t) = \bigoplus_{a \in A} x_{t+a} + h(X_t, Y_t)$, where A is some fixed subset of $\{0, 1, 2, \ldots, n-1\}$.

The Grain family uses an n-bit key K, and an m-bit initialization vector IV, with $m < n$. The key is loaded in the NFSR and the IV is loaded in the 0^{th} to the $(m-1)^{th}$ bits of the LFSR. The remaining m^{th} to $(n-1)^{th}$ bits of the LFSR are loaded with some fixed pad $P \in \{0,1\}^{n-m}$. Hence at this stage, the $2n$ bit initial state is of the form $K||IV||P$. Then, for the first $2n$ clocks, the keystream produced at the output point of the function h' is XOR-ed to both the LFSR and NFSR update functions, i.e., during the first $2n$ clock intervals, the LFSR and the NFSR bits are updated as $y_{t+n} = z_t + f(Y_t)$, $x_{t+n} = y_t + z_t + g(X_t)$. This is the Key Scheduling Algorithm (KSA).

After the completion of the KSA, z_t is no longer XOR-ed to the LFSR and the NFSR but it is used as the Pseudo-Random keystream bit. This is the Pseudo-Random Generation Algorithm (PRGA). Therefore during this phase, the LFSR and NFSR are updated as $y_{t+n} = f(Y_t), x_{t+n} = y_t + g(X_t)$.

For Grain-128a authenticated encryption scheme the exact parameters are as follows. The size of Key $n = 128$ bits and the IV is of size $m = 96$ bits. The value of pad used is $P = 0xFFFF\ FFFE$. The LFSR update rule is given by

$$y_{t+128} \stackrel{\Delta}{=} f(Y_t) = y_{t+96} + y_{t+81} + y_{t+70} + y_{t+38} + y_{t+7} + y_t.$$

The NFSR state is updated as follows

$$\begin{aligned} x_{t+128} = \ & y_t + g(x_{t+96}, x_{t+95}, x_{t+93}, x_{t+92}, x_{t+91}, x_{t+88}, x_{t+84}, x_{t+82}, x_{t+78}, x_{t+70}, x_{t+68}, x_{t+67}, x_{t+65}, \\ & x_{t+61}, x_{t+59}, x_{t+48}, x_{t+40}, x_{t+27}, x_{t+26}, x_{t+25}, x_{t+24}, x_{t+22}, x_{t+13}, x_{t+11}, x_{t+3}, x_t), \end{aligned}$$

where $g(x_{t+96}, x_{t+95}, \ldots, x_t)$

$$\begin{aligned} \stackrel{\Delta}{=} \quad g(X_t) \ = \ & x_t + x_{t+26} + x_{t+56} + x_{t+91} + x_{t+96} + \\ & x_{t+3}x_{t+67} + x_{t+11}x_{t+13} + x_{t+17}x_{t+18} + x_{t+27}x_{t+59} + x_{t+40}x_{t+48} + x_{t+61}x_{t+65} + \\ & x_{t+68}x_{t+84} + x_{t+88}x_{t+92}x_{t+93}x_{t+95} + x_{t+22}x_{t+24}x_{t+25} + x_{t+70}x_{t+78}x_{t+82}. \end{aligned}$$

The pre-output function z_t is defined as

$$z_t = \bigoplus_{j \in A} x_{t+j} + y_{t+93} + h(x_{t+12}, y_{t+8}, y_{t+13}, y_{t+20}, x_{t+95}, y_{t+42}, y_{t+60}, y_{t+79}, y_{t+94})$$

where $A = \{2, 15, 36, 45, 64, 73, 89\}$ and $h(s_0, \ldots, s_8) = s_0s_1 + s_2s_3 + s_4s_5 + s_6s_7 + s_0s_4s_8$. The output function is defined as $y_t = z_{64+2t}$.

Authentication Assume that we have a message of length L defined by the bits $m_0, \ldots, m_{L-1}$. Set $m_L = 1$ as padding. To provide authentication, two registers, called accumulator and shift register

of size 32 bits each, are used. The content of accumulator and shift register at time t is denoted by $a_t^0, \ldots, a_t^{31}$ and $r_t, \ldots, r_{t+31}$. The accumulator is initialized through $a_0^t = z_t, 0 \leq t \leq 31$ and the shift register is initialized through $r_t = z_{32+t}, 0 \leq t \leq 31$. The shift register is updated as $r_{t+32} = z_{64+2t+1}$. The accumulator is updated as $a_{t+1}^j = a_t^j + m_t r_{t+j}$ for $0 \leq j \leq 31$ and $0 \leq t \leq L$. The final content of accumulator, $a_{L+1}^0, \ldots, a_{L+1}^{31}$ is used for authentication.

2 Differential Fault Analysis on Grain-128a

We like to point out that to the best of our knowledge there is no existing fault attack on Grain-128a available in literature. Moreover, our attack strategy works using the MAC of certain messages instead of exploiting the keystream bits directly. Before proceeding further, let us now formalize the fault model.

1. The attacker is able to reset the system with the original Key-IV (as in [4, 6]) or the original Key and different IVs (as in [16]) and start the cipher operations again.
2. The attacker can inject a fault at any one random bit location of the LFSR or NFSR. As a result of the fault injection, the binary value in the bit-location (where the fault has been injected) is toggled. The attacker is not allowed to choose the location where he wants to inject the fault. However, as assumed in both [4, 6, 16] the fault in any bit may be reproduced at any later stage of operation, once injected.
3. Similar to [4, 6], the attacker can inject faults in the LFSR only, whereas the NFSR has been used for fault injection in [16].
4. The attacker has full control over the timing of fault injection, i.e., it is possible to inject the fault precisely at any stage of the cipher operation.
5. The attacker can obtain the MAC of any message of his choice including the empty message.

2.1 Obtaining the Location of the Fault

Our attack model assumes that the attacker is allowed to toggle the value at exactly one random location of the LFSR. The attacker, however can not explicitly choose the location where the fault is to be injected. In order for the attack to succeed, it is very important that it will be possible to identify the location of the LFSR where the fault has been induced.

Let $S_0 \in \{0,1\}^{256}$ be the initial state of the Grain-128a PRGA, and S_{0,Δ_ϕ} be the initial state resulting after injecting fault in LFSR location $\phi \in [0, 127]$. Let $Z = [z_0, z_1, \ldots, z_{65}]$ and $Z^\phi = [z_0^\phi, z_1^\phi, \ldots, z_{65}^\phi]$ be the first 66 keystream bits produced by S_0 and S_{0,Δ_ϕ} respectively. Then as per the authentication scheme the MAC $\sigma(\emptyset)$ of the empty message $\emptyset$ is given by the vector $[z_0 + z_{32}, z_1 + z_{33}, \ldots, z_{31} + z_{63}]$ and similarly the MAC for the singular message bit 0 will be given by $\sigma(0) = [z_0 + z_{33}, z_1 + z_{34}, \ldots, z_{30} + z_{63}, z_{31} + z_{65}]$. The corresponding faulty MACs are $\sigma^\phi(\emptyset) = [z_0^\phi + z_{32}^\phi, z_1^\phi + z_{33}^\phi, \ldots, z_{31}^\phi + z_{63}^\phi]$ and $\sigma^\phi(0) = [z_0^\phi + z_{33}^\phi, z_1^\phi + z_{34}^\phi, \ldots, z_{30}^\phi + z_{63}^\phi, z_{31}^\phi + z_{65}^\phi]$.

The task for the fault location identification routine is to determine the fault location ϕ by analyzing the difference between $[\sigma(\emptyset), \sigma(0)]$ and $[\sigma^\phi(\emptyset), \sigma^\phi(0)]$.

Definition 1. *We define a 64-bit vector E_ϕ over GF(2) defined as follows. Let E_ϕ^1 be the bitwise logical XNOR (complement of XOR) of the MACs of $\sigma(\emptyset)$ and $\sigma^\phi(\emptyset)$, i.e., $E_\phi^1 = 1 + \sigma(\emptyset) + \sigma^\phi(\emptyset)$, (here + should be interpreted as $\oplus$) and similarly $E_\phi^2 = 1 + \sigma(0) + \sigma^\phi(0)$. Then $E_\phi = E_\phi^1 || E_\phi^2$.*

Since S_0 can have 2^{224} values (each arising from a different combination of the 128 bit key and 96 bit IV, rest 32 padding bits are fixed), each of these choices of S_0 may lead to different patterns of E_ϕ. The bitwise logical AND of all such vectors E_ϕ is denoted as the Signature vector Sgn_ϕ for the fault location ϕ.

Since it is computationally infeasible to generate 2^{224} patterns and AND them, below we present a clever idea to achieve this efficiently. Whenever $Sgn_\phi(i)$ is 1 for $0 \leq i \leq 31$, this implies that the i^{th} MAC bit produced by S_0 and S_{0,Δ_ϕ} for the empty message is equal for all choices of S_0. Similarly if $Sgn_\phi(i)$ is 1 for $32 \leq i \leq 63$ this implies that the $(i-32)^{th}$ MAC bit produced by S_0 and S_{0,Δ_ϕ} for the zero message is equal.

For Grain-128a, two initial states of the PRGA $S_0, S_{0,\Delta_{127}} \in \{0,1\}^{256}$ which differ only in the 127^{th} position of the LFSR, produce identical output bits in 62 specific positions among the initial 66 keystream bits produced during the PRGA. If an input differential is introduced in the 127^{th} LFSR position, then at all rounds numbered $k \in [0,65] \setminus \{33,34,48,65\}$, the difference exists in positions that do not provide input to the Boolean function h and hence at these clocks the keystream bit produced by the two states are essentially the same. At all other clock rounds the difference appears at positions which provide input to h. Hence the keystream produced at these clocks may be different. Since

$$\sigma(\emptyset) = [z_0 + z_{32}, z_1 + z_{33}, \ldots, z_{31} + z_{63}] \text{ and } \sigma^\phi(\emptyset) = [z_0^\phi + z_{32}^\phi, z_1^\phi + z_{33}^\phi, \ldots, z_{31}^\phi + z_{63}^\phi],$$

this implies that all bits of $\sigma(\emptyset)$ and $\sigma^{127}(\emptyset)$ are equal except for the bits indexed by $1, 2, 16$. Also since

$$\sigma(0) = [z_0 + z_{33}, z_1 + z_{34}, \ldots, z_{30} + z_{63}, z_{31} + z_{65}] \text{ and } \sigma^\phi(0) = [z_0^\phi + z_{33}^\phi, z_1^\phi + z_{34}^\phi, \ldots, z_{30}^\phi + z_{63}^\phi, z_{31}^\phi + z_{65}^\phi],$$

we can say that all bits of $\sigma(0)$ and $\sigma^{127}(0)$ are equal except for the bits indexed by $0, 1, 15, 31$. Following the explanation given above, we can write Sgn_{127} in hexadecimal notation, Sgn_{79} = `9FFF 7FFF 3FFE FFFE`, which has $64 - 3 - 4 = 57$ many 1's and rest 0's.

Generalizing the above idea, for two PRGA initial states $S_0, S_{0,\Delta_\phi} \in \{0,1\}^{256}$ which differ only in the ϕ^{th} LFSR location, an analysis of the differential trails shows that out of the first 66 keystream bits produced by them, the bits at a certain fixed rounds are guaranteed to be equal. Thus by performing the above analysis for all fault locations ϕ ($0 \leq \phi \leq 127$), it is possible to calculate all the Signature vectors. Table 1 presents the vectors for each fault location ϕ, where the Fault Signature Vectors Sgn_ϕ for $0 \leq \phi \leq 127$ are written in hexadecimal notation.

Steps for location Identification As mentioned above, the task for the fault identification routine is to determine the value of ϕ given the vector E_ϕ, i.e., obtaining a unique Sgn_ϕ. For any l-bit vector V, let $B_V = \{i : 0 \leq i < l,\ V(i) = 1\}$. Now define a relation $\preceq$ in $\{0,1\}^l$ such that for 2 elements $V_1, V_2 \in \{0,1\}^l$, we will have $V_1 \preceq V_2$ if $B_{V_1} \subseteq B_{V_2}$.

So we start with a Key-IV pair K, IV_0 and record the MACs of the empty and zero messages. We then reset the cipher with K, IV_0 and apply a fault at some location ϕ (that is selected randomly and not known at this point) at the beginning of the PRGA, and obtain the corresponding faulty MACs of the empty and zero message. Using these we compute the E_ϕ vector as given in Definition 1. The entire process requires 4 invocations of the MAC routine. Now we check the elements in B_{E_ϕ}. By the definition of Signature vector proposed above, we know that for the correct value of ϕ, $B_{Sgn_\phi} \subseteq B_{E_\phi}$ and hence $Sgn_\phi \preceq E_\phi$. So our strategy would be to search all the Signature vectors and formulate the candidate set $\Psi_0 = \{\psi : 0 \leq \psi \leq 127,\ Sgn_\psi \preceq E_\phi\}$. If $|\Psi_0|$ is 1, then the single element in Ψ_0 will give us the fault location ϕ. However, this may not necessarily be the case always. If $|\Psi_0| > 1$, we will be unable to decide conclusively at this stage.

In such a scenario we reset the cipher with K, IV_1 (IV_1 different from IV_0) and record the fault-free MAC of the empty and zero messages. We then reset the cipher with K, IV_1 again and apply the fault at the location ϕ (our fault model considers that the fault can be applied at the same location without knowing it) at the beginning of the PRGA round and record the corresponding faulty MACs. Now we recalculate the vector E_ϕ as defined previously. We now search over the Signature vectors in the candidate set Ψ_0 and narrow down the set of possible candidates to $\Psi_1 = \{\psi : \psi \in \Psi_0,\ Sgn_\psi \preceq E_\phi\}$. Clearly, $|\Psi_1| \leq |\Psi_0|$, and so if $|\Psi_1| = 1$ then the fault location ϕ is the single element in Ψ_1. If not, we

ϕ	Sgn_ϕ	ϕ	Sgn_ϕ	ϕ	Sgn_ϕ	ϕ	Sgn_ϕ
0	8EFF BEFF 1DFF 7DFE	32	FFF7 EF67 FFF7 EF4E	64	F7F7 ED73 F7EF DCE7	96	D7FF 9DF3 8FFF 3BE7
1	C77F DF7F 8EFF BEFE	33	FFFB F7B3 FFFB F7A7	65	FBFB F6B9 FBF7 EE73	97	EBFF CEF9 C7FF 9DF2
2	E3BF EFBF C77F DF7F	34	FFFD FBD9 FFFD FBD3	66	FDFD FB5C FDFB F739	98	F5FF E77C E3FF CEF9
3	F1DF F7DF E3BF EFBF	35	FFFE FDEC FFFE FDE9	67	FEFE FDAE FEFD FB9D	99	FAFF F3BE F1FF E77D
4	F8EF FBEF F1DF F7DF	36	FFFF 7EF6 FFFF 7EF5	68	FF7F 7ED7 FF7E FDCE	100	FD7F F9DF F8FF F3BE
5	FC77 FDF7 F8EF FBEF	37	FFFF BF7B FFFF BF7A	69	FFBF BF6B FFBF 7EE6	101	FEBF FCEF FC7F F9DE
6	FE3B FEFB FC77 FDF7	38	CFFF 9FBD 9FFF 5FBC	70	CFDF 9FB5 9FDF 3F73	102	FF5F FE77 FE3F FCEF
7	CF1D BF7D 9E3B 7EFB	39	E7FF CFDE CFFF AFDE	71	E7EF CFDA CFEF 9FB9	103	FFAF FF3B FF1F FE77
8	678E DFBE 4F1D BF7D	40	73FF E7EF E7FF D7EF	72	F3F7 E7ED E7F7 CFDD	104	FFD7 FF9D FF8F FF3B
9	B3C7 6FDF A78E DFBF	41	B9FF F3F7 73FF EBF7	73	F9FB F3F6 F3FB E7EE	105	FFEB FFCE FFC7 FF9D
10	D9E3 B7EF D3C7 6FDE	42	5CFF F9FB 39FF F5FB	74	7CFD F9FB F9FD F3F7	106	FFF5 FFE7 FFE3 FFCF
11	ECF1 DBF7 E9E3 B7EF	43	AE7F FCFD 9CFF FAFD	75	BE7E FCFD 7CFE F9FB	107	FFFA FFF3 FFF1 FFE6
12	F678 EDFB F4F1 DBF7	44	D73F FE7E CE7F FD7E	76	DF3F 7E7E BE7F 7CFD	108	FFFD 7FF9 FFF8 FFF3
13	7B3C 76FD 7A78 EDFB	45	6B9F FF3F E73F FEBF	77	EF9F BF3F DF3F BE7F	109	FFFE BFFC FFFC 7FF9
14	BD9E 3B7E BD3C 76FD	46	B5CF FF9F 739F FF5F	78	F7CF DF9F EF9F DF3E	110	FFFF 5FFE FFFE 3FFC
15	DECF 1DBF DE9E 3B7F	47	DAE7 FFCF B9CF FFAF	79	7BE7 EFCF 77CF EF9F	111	7FFF AFFF FFFF 1FFE
16	EF67 8EDF EF4F 1DBE	48	ED73 FFE7 DCE7 FFD7	80	BDF3 F7E7 BBE7 F7CF	112	BFFF D7FF 7FFF 8FFE
17	F7B3 C76F F7A7 8EDF	49	F6B9 FFF3 EE73 FFEB	81	CEF9 BBF3 9DF3 7BE7	113	DFFF EBFF BFFF C7FF
18	FBD9 E3B7 FBD3 C76F	50	FB5C FFF9 F739 FFF5	82	E77C DDF9 CEF9 BDF3	114	EFFF F5FF DFFF E3FF
19	FDEC F1DB FDE9 E3B7	51	FDAE 7FFC FB9C FFFA	83	F3BE 6EFC E77C DEF9	115	F7FF FAFF EFFF F1FF
20	7EF6 78ED 7EF4 F1DB	52	7ED7 3FFE FDCE 7FFD	84	F9DF 377E F3BE 6F7D	116	FBFF FD7F F7FF F8FF
21	BF7B 3C76 BF7A 78ED	53	BF6B 9FFF 7EE7 3FFF	85	FCEF 9BBF F9DF 37BE	117	FDFF FEBF FBFF FC7F
22	DFBD 9E3B DFBD 3C77	54	DFB5 CFFF BF73 9FFE	86	FE77 CDDF FCEF 9BDE	118	FEFF FF5F FDFF FE3F
23	EFDE CF1D EFDE 9E3A	55	EFDA E7FF DFB9 CFFF	87	FF3B E6EF FE77 CDEF	119	FF7F FFAF FEFF FF1F
24	F7EF 678E F7EF 4F1D	56	F7ED 73FF EFDC E7FF	88	FF9D F377 FF3B E6F7	120	FFBF FFD7 FF7F FF8F
25	FBF7 B3C7 FBF7 A78F	57	FBF6 B9FF F7EE 73FF	89	FFCE F9BB FF9D F37B	121	FFDF FFEB FFBF FFC7
26	FDFB D9E3 FDFB D3C6	58	FDFB 5CFF FBF7 39FF	90	FFE7 7CDD FFCE F9BD	122	FFEF FFF5 FFDF FFE3
27	FEFD ECF1 FEFD E9E3	59	FEFD AE7F FDFB 9CFF	91	FFF3 BE6E FFE7 7CDE	123	FFF7 FFFA FFEF FFF1
28	FF7E F678 FF7E F4F1	60	7F7E D73F 7EFD CE7F	92	7FF9 DF37 FFF3 BE6F	124	FFFB FFFD FFF7 FFF8
29	FFBF 7B3C FFBF 7A79	61	BFBF 6B9F BF7E E73F	93	3FFC EF9B 7FF9 DF37	125	7FFD FFFE FFFB FFFC
30	FFDF BD9E FFDF BD3C	62	DFDF B5CF DFBF 739F	94	1FFE 77CD 3FFC EF9B	126	3FFE FFFF 7FFD FFFE
31	FFEF DECF FFEF DE9E	63	EFEF DAE7 EFDF B9CF	95	8FFF 3BE6 9FFE 77CD	127	9FFF 7FFF 3FFE FFFE

Table 1. Signature Vectors for different fault locations.

repeat the above process for another round for a different Key-IV pair K, IV_2. If after k rounds of this process, $|\Psi_{k-1}| = 1$, then the single element in Ψ_{k-1} gives us the desired location ϕ.

With detailed experiments taking an average over 2^{20} uniformly randomly chosen Key-IV pairs, it we found that the average value of k is 1.31 to uniquely identify a fault location in the LFSR. Since we are working with the MAC of empty and zero message, thus, for each location we need to inject $\mu = 2 \cdot 1.31 = 2.62$ faults.

Now let us argue that the LFSR fault location can be uniquely identified by the signature scheme proposed here. The signature scheme is based on both the empty and the zero message. Now a simple exhaustive search, through the Signature vectors for all fault locations, will show that $Sgn_{\phi_1} \not\preceq Sgn_{\phi_2}$ for any two fault locations $0 \leq \phi_1 \neq \phi_2 \leq 127$. This implies that for any value of the fault location $\phi \in [0, 127]$ the fault identification scheme will eventually narrow down the candidate set Ψ_{k-1} to just one element for some value of k.

One may wonder if the Signature vector were to be based on the difference of MAC of just the empty or the 0 message, whether a location identification scheme could have been proposed. The answer is no. Take the signature scheme based on the MAC difference of just the empty message in which $l = 32$. Studying the Signature vectors, one can check that the first 32 bits of Sgn_{21} = `BF7B 3C76` and Sgn_{36} = `FFFF 7EF6`. Note that, for all locations $i \in [0, 31]$ such that $Sgn_{21}(i) = 1$, the value of $Sgn_{36}(i)$ is also 1. This implies that $Sgn_{21} \preceq Sgn_{36}$. Now consider the case with the fault location $\phi = 36$. Then by the definition of the signature vector we have $Sgn_{36} \preceq E_\phi$. Since $\preceq$ is a partial order on $\{0, 1\}^l$, this implies that $Sgn_{21} \preceq E_\phi$ and so whenever $\phi = 36$ the fault location identification routine will never be able to narrow down the set of possible candidates Ψ_k to only $\{36\}$ for any value

of k. So the signature scheme can not be based on the MAC difference of the empty message only. If we were to base the signature scheme on the MAC difference of the 0 message bit, then too a look at the signature tables will show us that $Sgn_{16} \preceq Sgn_{111}$ and the scheme would fail by the above argument. It will be very interesting to find out a message for which the signature scheme will work just looking at the fault-free and faulty MACs on it.

2.2 Determining the LFSR State

Towards this, let us present a few more notations at this point.

1. $S_t = [x_0^t, x_1^t, \ldots, x_{127}^t \quad y_0^t, y_1^t, \ldots, y_{127}^t]$ is used to denote the internal state of the cipher at the beginning of round t of the PRGA when initialized with the Key-IV pair K, IV_0. Thus x_i^t (y_i^t) denotes the i^{th} NFSR (LFSR) bit at the start of round t of the PRGA. When $t = 0$, we use $S_0 = [x_0, x_1, \ldots, x_{127} \quad y_0, y_1, \ldots, y_{127}]$ to denote the internal state for convenience.
2. S_t^{ϕ} is used to denote the internal state of the cipher at the beginning of round t of the PRGA when initialized with the Key-IV pair K, IV_0, when a fault has been injected in LFSR location ϕ at the beginning of the PRGA round.
3. z_i^{ϕ} denotes the keystream bit produced in the i^{th} PRGA round, after faults have been injected in LFSR location ϕ at the beginning of the PRGA round. z_i is the fault-free i^{th} keystream bit.

We start by making the following observations about the output Boolean function h in Grain-128a:

$$h(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + h(s_0, s_1, 1 + s_2, s_3, s_4, s_5, s_6, s_7, s_8) = s_3 \quad (1)$$

$$h(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + h(s_0, s_1, s_2, 1 + s_3, s_4, s_5, s_6, s_7, s_8) = s_2 \quad (2)$$

$$h(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + h(s_0, s_1, s_2, s_3, s_4, s_5, 1 + s_6, s_7, s_8) = s_7 \quad (3)$$

$$h(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + h(s_0, s_1, s_2, s_3, s_4, s_5, s_6, 1 + s_7, s_8) = s_6 \quad (4)$$

Let us now explain in detail how we can obtain the bit-value at a specific location of the LFSR, say for example y_{108}. Note that s_0, s_4 correspond to the NFSR locations $12, 95$ respectively and $s_1, s_2, s_3, s_5, s_6, s_7, s_8$ correspond to the LFSR locations $8, 13, 20, 42, 60, 79, 94$ respectively. Now look at (1) above and note that s_2 corresponds to the LFSR location 13. If two internal states S and S_{Δ} be such that they differ in the LFSR location 13 (and in no other tap locations that contribute to the keystream bit generation), then the difference of the keystream bit produced by them will be equal to the value in LFSR location 20. Similar analysis can be done corresponding to (2), (3), (4).

Assume that the attacker has injected a fault at location 127 of the LFSR at the beginning of the PRGA. Then at round 48 of the PRGA the input differential travels to location 79 of the LFSR, i.e., at round 48 the original state S_{48} and the faulty state S_{48}^{127} differ in location 79 of the LFSR and in no other location that contributes inputs to the output keystream bit at round 48. Then by equation (4), the sum of the corresponding fault-free and faulty bits produced at round 48 is given by $z_{48} + z_{48}^{127} = y_{60}^{48} = y_{108}$.

At round 16 of the PRGA, the differential does not sit on any LFSR location that contributes input to the output keystream bit at that round. Hence $z_{16} = z_{16}^{127}$.

Now consider the fault-free and faulty MAC (due to the fault at $\phi = 127$ at the beginning of the PRGA) of the empty message $\sigma(\emptyset)$ and $\sigma^{127}(\emptyset)$. From the definition of the MAC of empty message, it can be deduced that the bit number 16 of $\sigma(\emptyset) \oplus \sigma^{127}(\emptyset)$ is given by $z_{16} + z_{48} + z_{16}^{127} + z_{48}^{127} = y_{108}$.

Hence by looking at the difference in the correct and faulty MACs of the empty messages one can deduce the LFSR state bit y_{108} at the beginning of the PRGA.

In Table 2 we give a list of 115 LFSR state bits y_i that can be recovered by observing the difference of the faulty and correct d^{th} ($0 \leq d \leq 31$) MAC bit of the empty message for different values of the fault location ϕ. There are 174 (more than 115) entries in the table and this is due to the multiple

ϕ	d	State bit y_i	ϕ	d	State bit y_i	ϕ	d	State bit y_i	ϕ	d	State bit y_i	ϕ	d	State bit y_i	ϕ	d	State bit y_i
0	17	y_{109}	21	31	y_{123}	65	5	y_{84}	75	15	y_{94}	83	23	y_{102}	97	5	y_{116}
1	18	y_{110}	38	17	y_{109}	65	20	y_{72}	75	22	y_{114}	83	30	y_{122}	98	6	y_{117}
2	19	y_{111}	39	18	y_{110}	66	6	y_{85}	75	23	y_{68}	83	31	y_{76}	99	7	y_{118}
3	20	y_{112}	40	19	y_{111}	66	21	y_{73}	75	30	y_{82}	84	20	y_{112}	100	8	y_{119}
4	21	y_{113}	41	20	y_{112}	67	7	y_{86}	76	16	y_{95}	84	24	y_{103}	101	9	y_{120}
5	22	y_{114}	42	21	y_{113}	67	22	y_{74}	76	23	y_{115}	84	31	y_{123}	102	10	y_{121}
6	23	y_{115}	43	22	y_{114}	68	8	y_{87}	76	24	y_{69}	85	21	y_{113}	103	11	y_{122}
7	17	y_{109}	44	23	y_{115}	68	23	y_{75}	76	31	y_{83}	85	25	y_{104}	104	12	y_{123}
7	24	y_{116}	45	24	y_{116}	69	9	y_{88}	77	17	y_{96}	86	22	y_{114}	105	13	y_{124}
8	18	y_{110}	46	25	y_{117}	69	24	y_{76}	77	24	y_{116}	86	26	y_{105}	106	14	y_{125}
8	25	y_{117}	47	26	y_{118}	70	10	y_{89}	77	25	y_{70}	87	23	y_{115}	107	15	y_{126}
9	19	y_{111}	48	27	y_{119}	70	17	y_{109}	78	18	y_{97}	87	27	y_{106}	108	16	y_{127}
9	26	y_{118}	49	28	y_{120}	70	25	y_{77}	78	25	y_{117}	88	24	y_{116}	111	0	y_{92}
10	20	y_{112}	50	29	y_{121}	71	11	y_{90}	78	26	y_{71}	88	28	y_{107}	112	1	y_{93}
10	27	y_{119}	51	30	y_{122}	71	18	y_{110}	79	19	y_{98}	89	25	y_{117}	113	2	y_{94}
11	21	y_{113}	52	31	y_{123}	71	19	y_{64}	79	26	y_{118}	89	29	y_{108}	114	3	y_{95}
11	28	y_{120}	57	12	y_{64}	71	26	y_{78}	79	27	y_{72}	90	26	y_{118}	115	4	y_{96}
12	22	y_{114}	58	13	y_{65}	72	12	y_{91}	80	20	y_{99}	90	30	y_{109}	116	5	y_{97}
12	29	y_{121}	59	14	y_{66}	72	19	y_{111}	80	27	y_{119}	91	27	y_{119}	117	6	y_{98}
13	23	y_{115}	60	0	y_{79}	72	20	y_{65}	80	28	y_{73}	91	31	y_{110}	118	7	y_{99}
13	30	y_{122}	60	15	y_{67}	72	27	y_{79}	81	17	y_{109}	92	0	y_{111}	119	8	y_{100}
14	24	y_{116}	61	1	y_{80}	73	13	y_{92}	81	21	y_{100}	92	28	y_{120}	120	9	y_{101}
14	31	y_{123}	61	16	y_{68}	73	20	y_{112}	81	28	y_{120}	93	1	y_{112}	121	10	y_{102}
15	25	y_{117}	62	2	y_{81}	73	21	y_{66}	81	29	y_{74}	93	29	y_{121}	122	11	y_{103}
16	26	y_{118}	62	17	y_{69}	73	28	y_{80}	82	18	y_{110}	94	2	y_{113}	123	12	y_{104}
17	27	y_{119}	63	3	y_{82}	74	14	y_{93}	82	22	y_{101}	94	30	y_{122}	124	13	y_{105}
18	28	y_{120}	63	18	y_{70}	74	21	y_{113}	82	29	y_{121}	95	3	y_{114}	125	14	y_{106}
19	29	y_{121}	64	4	y_{83}	74	22	y_{67}	82	30	y_{75}	95	31	y_{123}	126	15	y_{107}
20	30	y_{122}	64	19	y_{71}	74	29	y_{81}	83	19	y_{111}	96	4	y_{115}	127	16	y_{108}

Table 2. LFSR state bits recovered

fault options for identifying some of the LFSR bits. The LFSR state bits not present in Table 2 are $y_0, y_1, \ldots, y_{12}$. However it can be verified that $\forall i \in [0, 12]$, by applying a fault at location $\phi = 109 + i$ the $(17 + i)^{th}$ bit in difference of $\sigma(\emptyset)$ and $\sigma^{109+i}(\emptyset)$ is equal to the state bit y_{127}^{1+i}. Since y_{127}^{1+i} is a linear function of $y_0, y_1, \ldots, y_{127}$, we can derive y_0 to y_{12} as follows. By the LFSR update rule of Grain-128a, we have the following 13 equations

$$y_{127}^{1+i} = y_{96+i} + y_{81+i} + y_{70+i} + y_{38+i} + y_{7+i} + y_i, \quad \forall i \in [0, 12].$$

In the last equation y_{12} is the only unknown and its value can be calculated easily. Similarly y_{11} is the only unknown in the previous equation. Solving the equations in this manner one can obtain the entire LFSR state at the beginning of the PRGA.

2.3 Determining the NFSR State

Once the LFSR internal state of the initial PRGA round is known, one can then proceed to determine the NFSR internal state. In [5] it was shown, that this could have been done efficiently for the initial version of the cipher i.e. Grain v0. After the attack in [5] was reported, the designers made the necessary changes to Grain v1, Grain-128 and Grain-128a so that for these new ciphers, determining the NFSR state form the knowledge of the LFSR state was no longer straightforward. In order to determine the NFSR bits, we look into the decomposition of the Boolean function h in more detail.

One may note that for Grain-128a, $h(\mathbf{s}) = s_0 \cdot u(\mathbf{s}) + v(\mathbf{s})$, where $u(\mathbf{s}) = s_1 + s_4 s_8$, and $v(\mathbf{s}) = s_2 s_3 + s_4 s_5 + s_6 s_7$. Thus we note that

$$(i)\ u(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + u(s_0, 1 + s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) = 1,$$

$$(ii)\ v(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + v(s_0, 1 + s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) = 0.$$

Also h can be written as $h(\mathbf{s}) = s_4 \cdot U(\mathbf{s}) + V(\mathbf{s})$, where $U(\mathbf{s}) = s_5 + s_0 s_8$, and $V(\mathbf{s}) = s_2 s_3 + s_4 s_5 + s_6 s_7$. We also have

$$(i)\ U(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + U(s_0, s_1, s_2, s_3, s_4, 1 + s_5, s_6, s_7, s_8) = 1,$$

$$(ii)\ V(s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) + V(s_0, s_1, s_2, s_3, s_4, 1 + s_5, s_6, s_7, s_8) = 0.$$

Since s_0, s_4 correspond to NFSR variables, h satisfies all the properties listed above.

As before, assume the scenario in which the attacker has injected a fault at location 8 of the LFSR at the beginning of the PRGA. Then at this round of the PRGA the input differential travels sits on location 8 of the LFSR i.e. at round 0 of the PRGA the original state S_0 and the faulty state S_0^8 differ in location 8 of the LFSR and in no other location that contributes inputs to the output keystream bit at round 0. Then by the previous relation equation, the sum of the corresponding fault-free bits produced at round 0 is given by $z_0 + z_0^8 = x_{12}^0 \cdot 1 + 0 = x_{12}$.

Also note that at round 32 of the PRGA the differential does not sit on any LFSR location that contributes input to the output keystream bit at that round. Hence $z_{32} = z_{32}^8$.

Now consider the fault-free and faulty MAC (due to fault at $\phi = 8$ at the beginning of the PRGA) of the empty message $\sigma(\emptyset)$ and $\sigma^8(\emptyset)$. From the definition of MAC of empty message it can be deduced that the bit number 16 of $\sigma(\emptyset) \oplus \sigma^8(\emptyset)$ is given by $z_0 + z_{32} + z_0^8 + z_{32}^8 = x_{12}$.

Hence by looking at the difference in the correct and faulty MACs of the empty messages one is able to deduce the NFSR state bit x_{12} at the beginning of the PRGA. In Table 3 we give an exhaustive list of the NFSR state bits x_i that can be recovered by observing the difference of the faulty and correct d^{th} MAC bit of the empty message for different values of the fault location ϕ.

ϕ	d	State bit x_i	ϕ	d	State bit x_i	ϕ	d	State bit x_i	ϕ	d	State bit x_i	ϕ	d	State bit x_i	ϕ	d	State bit x_i
8	0	x_{12}	24	16	x_{28}	40	0	x_{44}	49	7	x_{102}	57	15	x_{110}	65	23	x_{118}
9	1	x_{13}	25	17	x_{29}	41	1	x_{45}	49	9	x_{53}	57	17	x_{61}	65	25	x_{69}
10	2	x_{14}	26	18	x_{30}	42	0	x_{95}	50	8	x_{103}	58	16	x_{111}	66	24	x_{119}
11	3	x_{15}	27	19	x_{31}	42	2	x_{46}	50	10	x_{54}	58	18	x_{62}	66	26	x_{70}
12	4	x_{16}	28	20	x_{32}	43	1	x_{96}	51	9	x_{104}	59	17	x_{112}	67	25	x_{120}
13	5	x_{17}	29	21	x_{33}	43	3	x_{47}	51	11	x_{55}	59	19	x_{63}	67	27	x_{71}
14	6	x_{18}	30	22	x_{34}	44	2	x_{97}	52	10	x_{105}	60	18	x_{113}	68	26	x_{121}
15	7	x_{19}	31	23	x_{35}	44	4	x_{48}	52	12	x_{56}	60	20	x_{64}	68	28	x_{72}
16	8	x_{20}	32	24	x_{36}	45	3	x_{98}	53	11	x_{106}	61	19	x_{114}	69	27	x_{122}
17	9	x_{21}	33	25	x_{37}	45	5	x_{49}	53	13	x_{57}	61	21	x_{65}	69	29	x_{73}
18	10	x_{22}	34	26	x_{38}	46	4	x_{99}	54	12	x_{107}	62	20	x_{115}	70	28	x_{123}
19	11	x_{23}	35	27	x_{39}	46	6	x_{50}	54	14	x_{58}	62	22	x_{66}	70	30	x_{74}
20	12	x_{24}	36	28	x_{40}	47	5	x_{100}	55	13	x_{108}	63	21	x_{116}	71	29	x_{124}
21	13	x_{25}	37	29	x_{41}	47	7	x_{51}	55	15	x_{59}	63	23	x_{67}	71	31	x_{75}
22	14	x_{26}	38	30	x_{42}	48	6	x_{101}	56	14	x_{109}	64	22	x_{117}	72	30	x_{125}
23	15	x_{27}	39	31	x_{43}	48	8	x_{52}	56	16	x_{60}	64	24	x_{68}	73	31	x_{126}
															74	0	x_{127}

Table 3. NFSR state bits recovered

Finding the Remaining Bits From the table given above all state bits of the NFSR can be found except $x_0, x_1, \ldots, x_{11}$ and $x_{76}, x_{77}, \ldots, x_{94}$. These bits may be found as follows. It can be verified that $\forall i \in [0, 8]$, by applying a fault at location $\phi = 73 + 2i$ at the beginning of the PRGA, the difference travels to the LFSR location 8 at round $65 + 2i$. It can also be checked that at this PRGA round the differential does not affect any other location that contributes to the output bit i.e. the states S_{65+2i}

and S_{65+21}^{73+2i} differ in only the LFSR location 8 and no other location that affects the output bit at this round. Then by the previous relation

$$z_{65+2i} + z_{65+2i}^{73+2i} = x_{12}^{65+2i} \cdot 1 + 0 = x_{77+2i}, \quad \forall i \in [0, 8].$$

It can also be verified that as a result of applying the fault at $73 + 2i, \forall i \in [0, 8]$ at round 31 of the PRGA, the differential does not affect any location that provides inputs to the output bit. Hence, $z_{31} = z_{31}^{73+2i}$. Now consider the fault-free and faulty MAC of the message 0^{i+1} (string of $i + 1$ zeros) obtained by faulting LFSR location $73 + 2i$ at the beginning of the PRGA. From definition

$$\sigma(0^{i+1}) = [z_0 + z_{33+i}, z_1 + z_{34+i}, \ldots, z_{30-i} + z_{63}, z_{31-i} + z_{65}, z_{32-i} + z_{67}, \ldots, z_{31} + z_{65+2i}], \quad \forall i \in [0, 8]$$

Hence the last bit in difference of $\sigma(0^{i+1})$ and $\sigma^{73+2i}(0^{i+1})$ is equal to

$$z_{31} + z_{65+2i} + z_{31}^{73+2i} + z_{65+2i}^{73+2i} = x_{77+2i}, \ \forall i \in [0, 8].$$

Thus far we have recovered 106 of the 128 NFSR state bits. Consider the 0^{th} bit of $\sigma(\emptyset)$ given by

$$z_0 + z_{32} = \bigoplus_{t \in B} x_t + y_{93} + y_{125} + h(x_{12}, y_8, y_{13}, y_{20}, x_{95}, y_{42}, y_{60}, y_{79}, y_{94}) + h(x_{44}, y_{40}, y_{55}, y_{75}, x_{127}, y_{74}, y_{92}, y_{111}, y_{126})$$

Here $B = \{2, 15, 36, 45, 64, 73, 89, 34, 47, 68, 77, 96, 105, 121\}$. Note that x_2 is the only unknown linear term in the above equation, and so its value can be calculated immediately.

Define $x_{127+i} = x_{127}^i, y_{127+i} = y_{127}^i$ for all $i \geq 1$. Then by using a similar analysis, it can be verified that the i^{th} bit of

$$\sigma(\emptyset) + \sigma^{74+i}(\emptyset) = x_{127+i}, \ \forall i \in [0, 31]. \tag{5}$$

Again consider the 0^{th} bit of $\sigma(0^{2j+1})$ for $0 \leq j \leq 8$, given by $z_0 + z_{33+2j}$

$$z_0 + z_{33+2j} = \bigoplus_{t \in B_j} x_t + y_{93} + y_{126+2j} + h(x_{12}, y_8, y_{13}, y_{20}, x_{95}, y_{42}, y_{60}, y_{79}, y_{94}) + h(x_{45+2j}, y_{41+2j}, y_{56+2j}, y_{76+2j}, x_{128+2j}, y_{75+2j}, y_{93+2j}, y_{112+2j}, y_{127+2j}), \ \forall j \in [0, 8].$$

Here $B_j = \{2, 15, 36, 45, 64, 73, 89, 35 + 2j, 48 + 2j, 69 + 2j, 78 + 2j, 97 + 2j, 106 + 2j, 122 + 2j\}$. In the above set of equations any x_k with $k > 127$ may be calculated from (5). Any y_k with $k > 127$ is a linear function of $y_0, \ldots, y_{127}$ which are already known. Hence x_{78+2j} with $0 \leq j \leq 8$ are the only unknown linear terms in each of these equations and their values are also immediately determined. At this point the only unknown state bits are $x_0, x_1, x_3, \ldots, x_{11}, x_{76}$. Consider the p^{th} bit of $\sigma(\emptyset)$ given by $z_p + z_{32+p}$ for $p \in [1, 9] \setminus \{3\}$.

$$z_p + z_{32+p} = \bigoplus_{t \in B} x_{t+p} + y_{93+p} + y_{125+p} + h(x_{12+p}, y_{8+p}, y_{13+p}, y_{20+p}, x_{95+p}, y_{42+p}, y_{60+p}, y_{79+p}, y_{94+p}) + h(x_{44+p}, y_{40+p}, y_{55+p}, y_{75+p}, x_{127+p}, y_{74+p}, y_{92+p}, y_{111+p}, y_{126+p}), \ \forall p \in [1, 9] \setminus \{3\}$$

In all these equations x_{2+p} is the only unknown linear term and its value can also be determined immediately (the strategy does not work for $p = 3$ as x_{76} is still unknown). We are left with x_0, x_1, x_5, x_{76}. The values of x_0, x_1, x_5 can be obtained from the expansion of $x_{128}, x_{129}, x_{133}$ from the NFSR update rule. Now, x_{76} occurs as a linear term in bit number 3 of $\sigma(\emptyset)$ and its value too is calculated immediately. Thus we have calculated all of S_0.

2.4 Finding the Secret Key and Complexity of the Attack

It is known that the KSA and PRGA routines in the Grain family are invertible. Once we have all the bits of S_0, by running the inverse KSA routine one can recover the secret key.

First we need to hit each of the locations of the LFSR. We inject the fault randomly in the LFSR locations and thus, we need $\tau = 128 \cdot \sum_{i=1}^{128} \frac{1}{i} \approx 695.4$ expected number of fault injections. For each of these injected faults, we need to identify the fault locations. Taking the value of μ from Section 2.1 that is the required number of expected faults for each LFSR location, the total number of faults to be injected $= \tau\mu = 695.4 \cdot 2.62 \approx 1822$. Additionally, as described in Section 2.3, 9 more fault injections are required for the locations $\phi = 73, 75, \ldots, 89$ to recover certain NFSR bits. Therefore, the expected number of faults that our attack needs is $1822 + 9 = 1831 < 2^{11}$.

For each fault during the location identification stage, two MAC invocations are required, that amounts to $1822 \cdot 2 = 3644$. Additionally, 20 more invocations are required during some cases of NFSR bit recovery. Thus the total number of invocations is less than 2^{12}.

References

1. The ECRYPT Stream Cipher Project. eSTREAM Portfolio of Stream Ciphers. Revised on September 8, 2008.
2. M. Ågren, M. Hell, T. Johansson and W. Meier. A New Version of Grain-128 with Authentication. Symmetric Key Encryption Workshop 2011, DTU, Denmark, February 2011.
3. J. P. Aumasson, I. Dinur, L. Henzen, W. Meier, and A. Shamir. Efficient FPGA Implementations of High-Dimensional Cube Testers on the Stream Cipher Grain-128. In SHARCS - Special-purpose Hardware for Attacking Cryptographic Systems, 2009.
4. S. Banik, S. Maitra and S. Sarkar. A Differential Fault Attack on the Grain Family of Stream Ciphers. To be presented in CHES 2012.
5. C. Berbain, H. Gilbert and A. Maximov. Cryptanalysis of Grain. In FSE 2006, LNCS, Vol. 4047, pp. 15–29, 2006.
6. A. Berzati, C. Canovas, G. Castagnos, B. Debraize, L. Goubin, A. Gouget, P. Paillier, S. Salgado. Fault Analysis of Grain-128. In: IEEE International Workshop on Hardware-Oriented Security and Trust, pp. 7–14, 2009.
7. T. E. Bjørstad. Cryptanalysis of Grain using Time/Memory/Data tradeoffs (v1.0 / 2008-02-25). Available at `http://www.ecrypt.eu.org/stream`.
8. C. De Cannière, O. Küçük and B. Preneel. Analysis of Grain's Initialization Algorithm. In AFRICACRYPT 2008, LNCS, Vol. 5023, pp. 276–289, 2008.
9. I. Dinur, T. Güneysu, C. Paar, A. Shamir, R. Zimmermann. An Experimentally Verified Attack on Full Grain-128 Using Dedicated Reconfigurable Hardware. In ASIACRYPT 2011, LNCS Vol. 7073, pp. 327–343, 2011.
10. I. Dinur, A. Shamir. Breaking Grain-128 with Dynamic Cube Attacks. In FSE 2011, LNCS, Vol. 6733, pp. 167–187, 2011.
11. H. Englund, T. Johansson, and M. S. Turan. A framework for chosen IV statistical analysis of stream ciphers. In INDOCRYPT 2007, LNCS, Vol. 4859, pp. 268–281, 2007.
12. S. Fischer, S. Khazaei, and W. Meier. Chosen IV statistical analysis for key recovery attacks on stream ciphers. In AFRICACRYPT 2008, LNCS, Vol. 5023, pp. 236–245, 2008.
13. M. Hell, T. Johansson and W. Meier. Grain - A Stream Cipher for Constrained Environments. ECRYPT Stream Cipher Project Report 2005/001, 2005. Available at `http://www.ecrypt.eu.org/stream`.
14. M. Hell, T. Johansson and W. Meier. A Stream Cipher Proposal: Grain-128. In IEEE International Symposium on Information Theory (ISIT 2006), 2006.
15. J. J. Hoch, A. Shamir. Fault Analysis of Stream Ciphers. In CHES 2004, LNCS, Vol. 3156, pp. 1–20, 2004.
16. S. Karmakar and D. Roy Chowdhury. Fault analysis of Grain-128 by targeting NFSR. In AFRICACRYPT 2011, LNCS, Vol. 6737, pp. 298–315, 2011.
17. S. Khazaei, M. Hassanzadeh and M. Kiaei. Distinguishing Attack on Grain. ECRYPT Stream Cipher Project Report 2005/071, 2005. Available at `http://www.ecrypt.eu.org/stream`
18. S. Knellwolf, W. Meier and M. Naya-Plasencia. Conditional Differential Cryptanalysis of NLFSR-based Cryptosystems. In ASIACRYPT 2010, LNCS, Vol. 6477, pp. 130–145, 2010.
19. Y. Lee, K. Jeong, J. Sung and S. Hong. Related-Key Chosen IV Attacks on Grain-v1 and Grain-128. In ACISP 2008, LNCS, Vol. 5107, pp. 321–335, 2008.
20. S. P. Skorobogatov. Optically Enhanced Position-Locked Power Analysis. In CHES 2006, LNCS, Vol. 4249, pp. 61–75, 2006.
21. S. P. Skorobogatov, R. J. Anderson. Optical Fault Induction Attacks. In CHES 2002, LNCS, Vol. 2523, pp. 2–12, 2003.

22. P. Stankovski. Greedy Distinguishers and Nonrandomness Detectors. In INDOCRYPT 2010, LNCS, Vol. 6498, pp. 210–226, 2010.
23. H. Zhang and X. Wang. Cryptanalysis of Stream Cipher Grain Family. IACR Cryptology ePrint Archive 2009: 109. Available at `http://eprint.iacr.org/2009/109`.