A Differential Fault Attack on the Grain Family of Stream Ciphers

Subhadeep Banik, Subhamoy Maitra, Santanu Sarkar

Indian Statistical Institute Kolkata

September 10, 2012

CHES 2012, Leuven Belgium

GRAIN family of Stream Ciphers

Grain Family

- Proposed by Hell et al in 2005
- Part of E-stream's hardware portfolio
- Bit-oriented, Synchronous stream cipher
- The first version (v0) of the cipher was cryptanalyzed
  1. A Distinguishing attack by Kiaei et. al (Ecrypt : 071).
  2. A State Recovery attack by Berbain et.al (FSE 2006).
- After this, the versions Grain v1, Grain 128, Grain 128a were proposed.

Motivation

- No fault analysis of Grain v1 has been reported.
- Existing works (Berzati et. al. HOST 09, Karmakar et. al. AFRICACYPT 11) are on Grain-128.
- Grain-128 has a relatively uncomplicated output function

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

- Hence, fault analysis is relatively simpler.

General Structure of the Grain Family

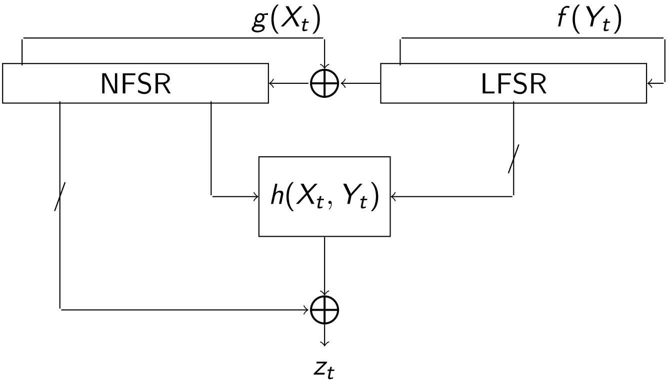


Figure: Structure of Grain v1

Grain v1

In Grain v1 the size of Key $n = 80$ bits and the IV is of size $m = 64$ bits. The value of pad used in the KLA is $P = 0xFFFF$. The LFSR update rule is given by

$$y_{t+80} \triangleq f(Y_t) = y_{t+62} + y_{t+51} + y_{t+38} + y_{t+23} + y_{t+13} + y_t$$

The NFSR state is updated as follows

$$x_{t+80} = y_t + g(X_t)$$

where $g(X_t) =$

$$x_{t+62} + x_{t+60} + x_{t+52} + x_{t+45} + x_{t+37} + x_{t+33} + x_{t+28} + x_{t+21} +$$
$$x_{t+14} + x_{t+9} + x_t + x_{t+63}x_{t+60} + x_{t+37}x_{t+33} + x_{t+15}x_{t+9} +$$
$$x_{t+60}x_{t+52}x_{t+45} + x_{t+33}x_{t+28}x_{t+21} + x_{t+63}x_{t+45}x_{t+28}x_{t+9} +$$
$$x_{t+60}x_{t+52}x_{t+37}x_{t+33} + x_{t+63}x_{t+60}x_{t+21}x_{t+15} +$$
$$x_{t+63}x_{t+60}x_{t+52}x_{t+45}x_{t+37} + x_{t+33}x_{t+28}x_{t+21}x_{t+15}x_{t+9} +$$
$$x_{t+52}x_{t+45}x_{t+37}x_{t+33}x_{t+28}x_{t+21}$$

Grain v1

The output keystream is produced by combining the LFSR and NFSR bits as follows

$$z_t = \bigoplus_{a \in A} x_{t+a} + h(y_{t+3}, y_{t+25}, y_{t+46}, y_{t+64}, x_{t+63}) \triangleq \bigoplus_{a \in A} x_{t+a} + h(X_t, Y_t)$$

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

$$h(s_0, s_1, s_2, s_3, s_4) = s_1 + s_4 + s_0 s_3 + s_2 s_3 + s_3 s_4 + s_0 s_1 s_2 + s_0 s_2 s_3 + s_0 s_2 s_4 + s_1 s_2 s_4 + s_2 s_3 s_4.$$

Keystream generating routines

- **Key Loading Algorithm (KLA)**
  - n-bit key $K \rightarrow$ NFSR
  - m-bit $(m < n)$ $IV \rightarrow$ LFSR[0]...LFSR[m-1]
  - $p = n - m$ bit pad $P \rightarrow$ LFSR[m]...LFSR[n-1]
- **Key Schedule Algorithm (KSA)**
  - For $2n$ clocks, output of h' is XOR-ed to the LFSR and NFSR update functions
  - $y_{t+n} = f(Y_t) + z_t$ and $x_{t+n} = y_t + z_t + g(X_t)$
- **Pseudo Random bitstream Generation Algorithm (PRGA)**
  - The feedback is discontinued
  - $y_{t+n} = f(Y_t)$ and $x_{t+n} = y_t + g(X_t)$
  - $z_t = h'(X^t, Y^t)$

Differential Fault Attack

Fault Model

- The attacker is able to reset the system with the original Key-IV and start the cipher operations again.
- The attacker can inject a fault at any one random bit location of the LFSR or NFSR.
- The fault in any bit may be reproduced at any later stage of operation, once injected.(Berzati et. al. $\mathrm{HOST}\ 09$)
- 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.

Identifying Fault Location

Location Identification

- Apply a fault at a random LFSR location: imperative to determine fault location before proceeding.
- This is done by comparing the fault-free and faulty Key-streams.
- More than one fault at same location may be required to conclusively identify the location.

The Idea

- Consider 2 initial states $S_0, S_{0,\Delta_{79}}$ such that $S_0 \oplus S_{0,\Delta_{79}} = s_{79}$

In all rounds $k \in [0, 79] \setminus \{15, 33, 44, 51, 54, 57, 62, 69, 72, 73, 75, 76\}$, the difference does not affect output keystream bit.
At all these rounds output of $S_0, S_{0,\Delta_{79}}$ guaranteed to be equal. Hence formulate signature vector Sgn_{79}= `FFFE FFFF BFF7 EDBD FB27`.

- Idea is to match the sum of faultless and faulty keystream bits with all Sgn_ϕ for $\phi \in [0, 79]$

Notations

- S_0 is the initial state of the Grain v1 PRGA.
- S_{0,Δ_ϕ} is the initial state after faulting LFSR location $\phi \in [0, 79]$
- $Z = [z_0, z_1, \ldots, z_l] \Rightarrow$ first l fault-less keystream bits.
- $Z^\phi = [z_0^\phi, z_1^\phi, \ldots, z_l^\phi] \Rightarrow$ first l faulty keystream bits.

Define l bit vectors $E_\phi, Sgn_\phi \Rightarrow E_\phi(i) = 1 + z_i + z_i^\phi$

$$\Rightarrow Sgn_\phi(i) = \bigodot_{S_0} E_\phi(i)$$

More Definitions

For any element $V \in \{0,1\}^l$

- Define support of $V \rightarrow B_V = \{i : 0 \leq i < l,\ V(i) = 1\}$
- Define a relation $\preceq$ in $\{0,1\}^l$ s.t. $\forall V_1, V_2 \in \{0,1\}^l$,

$$V_1 \preceq V_2 \text{ if } B_{V_1} \subseteq B_{V_2}$$

1. $\preceq$ is a partial order in $\{0,1\}^l$

The Task

- Given E_ϕ : Find ϕ
- Elements in $B_{E_\phi} \rightarrow$ PRGA rounds i during which $z_i = z_i^\phi$.
- For the correct value of ϕ :

$$B_{Sgn_\phi} \subseteq B_{E_\phi} \Rightarrow Sgn_\phi \preceq E_\phi$$

- Strategy : Formulate the candidate set

$$\Psi_0 = \{\psi : 0 \leq \psi \leq 79,\ Sgn_\psi \preceq E_\phi\}$$

- If $|\Psi_0| = 1$ then the element in Ψ_0 is surely ϕ.

If $|\Psi_0| \neq 1$

- Reset the cipher. Go to PRGA round l and fault the same location ϕ.
- Recalculate E_ϕ. Re-employ strategy

$$\Psi_1 = \{\psi : \psi \in \Psi_0,\ Sgn_\psi \preceq E_\phi\}$$

- If $|\Psi_1| = 1$, then the single element in this set is surely ϕ.
- Else Re-employ previous strategy for PRGA rounds $2l, 3l, \ldots$

Optimizations on l

- If $l \leq 44$, the scheme trivially fails.
  - $Sgn_{40} \preceq Sgn_{79} \rightarrow$ if $\phi = 79$ conclusive identification impossible.
- If $l > 44$, the scheme works.
  - $Sgn_{i_1} \npreceq Sgn_{i_2}\ \forall i_1, i_2 \in [0, 79]$
- Smaller value of l implies more faults for identification.
- Computer simulations over 2^{20} random Key-IV pairs : $l = 80$ is optimal.

Average no of faults vs l

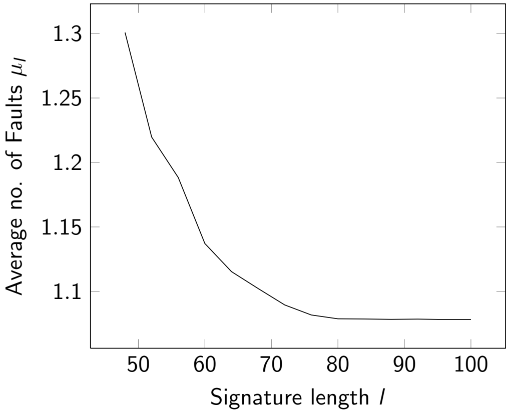


Figure: Average number of faults vs Length of Signature.

Beginning the Attack

More Notations

- $S_t = [x_0^t, x_1^t, \ldots, x_{79}^t \;\; y_0^t, y_1^t, \ldots, y_{79}^t]$ state at round t of the PRGA. x_i^t $(y_i^t) \rightarrow i^{th}$ NFSR (LFSR) bit at t^{th} round of the PRGA.
- When $t = 0$, $S_0 = [x_0, x_1, \ldots, x_{79} \;\; y_0, y_1, \ldots, y_{79}]$ for convenience.
- $S_t^{\phi}(t_1, t_2)$ state round t of the PRGA, when a fault at LFSR location ϕ at $t = t_1, t_2$.
- $z_t^{\phi}(t_1, t_2)$ t^{th} faulty keystream bit, when a fault at LFSR location ϕ at $t = t_1, t_2$.
- z_t is the fault-free t^{th} keystream bit.

Affine Differential Resistance

Definition

Consider a q-variable Boolean function F. A non-zero vector $\alpha \in \{0,1\}^q$ is said to be an affine differential of the function F if $F(\mathbf{x}) + F(\mathbf{x} + \alpha)$ is an affine function. A Boolean function is said to be affine differential resistant if it does not have any affine differential.

In Grain v1

$$h(s_0, s_1, s_2, s_3, s_4) + h(1 + s_0, 1 + s_1, s_2, s_3, 1 + s_4) = s_2$$

Therefore h is **not affine differential resistant.**

Fault attack on Grain v1: Getting the LFSR

Lemma

Fault in LFSR location $38 + r \ \forall r \in [0, 41]$*, at rounds* λ *and* $\lambda + 20$ *for* $\lambda = 0, 1, \ldots$

$\Rightarrow$ *In Round* $t = 55 + \lambda + r$*,* $S^{38+r}_{55+\lambda+r}(\lambda, \lambda + 20) \oplus S_{55+\lambda+r} = [y_3, y_{25}, x_{63}]^{55+\lambda+r}$

No difference in other 9 *locations that contributes to the output keystream bit.*

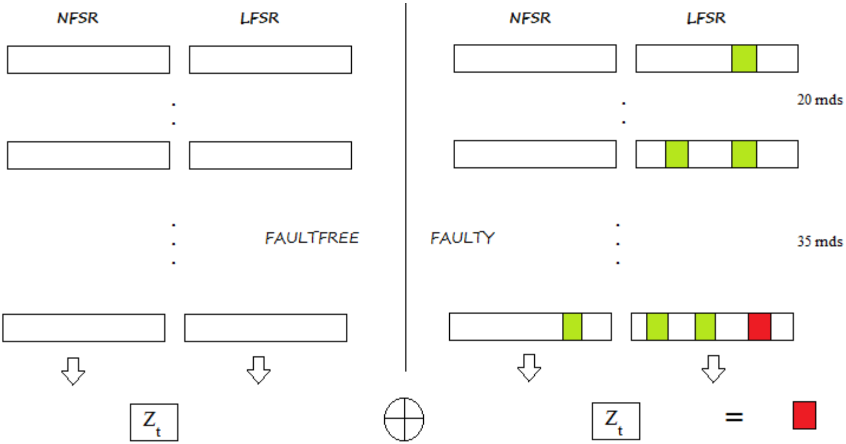
Therefore $z_t + z^{\phi}_t(\lambda, \lambda + 20) = y^t_{46}$ where $t = 55 + \lambda + r$

$\Rightarrow$ y^t_{46} is a linear function in $[y_0, y_1, \ldots, y_{79}]$ i.e. the LFSR bits of S_0.

$\Rightarrow$ Gives one linear equation in initial LFSR bits.

$\Rightarrow$ Use this to get 80 linearly independent equations and solve to get all LFSR bits of S_0.

Fault attack on Grain v1: LFSR recovery



Fault attack on Grain v1: Getting the NFSR

In Grain v1 we have

$$h = s_4 \cdot u(s_0, s_1, s_2, s_3) + v(s_0, s_1, s_2, s_3)$$

$$u(s_0, s_1, s_2, s_3) + u(s_0, 1 + s_1, s_2, 1 + s_3) = 1$$

Lemma

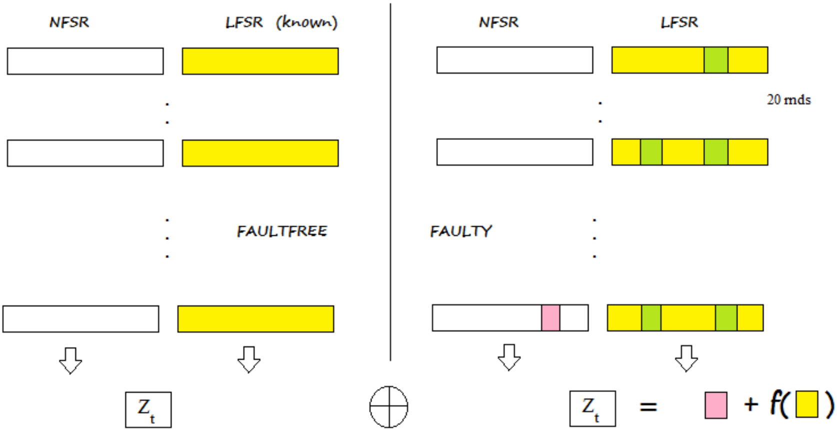
Fault in LFSR location ϕ at $0, 20$ PRGA rounds, then at round t

$$S_t + S_t^{\phi}(0, 20) = [y_{25}, y_{64}]^t$$

(i) $\phi = 51 + r,\ t = 91 + r$ *for* $0 \leq r \leq 28$*,*
(ii) $\phi = 62 + r,\ t = 55 + r$ *for* $0 \leq r \leq 17$*,*
(iii) $\phi = 62 + r,\ t = 75 + r$ *for* $0 \leq r \leq 15$*.*

$\Rightarrow z_t + z_t^{\phi}(0, 20) = x_{63}^t + v([y_3, y_{25}, y_{46}, y_{64}]^t) + v([y_3, 1 + y_{25}, y_{46}, 1 + y_{64}]^t)$

Fault attack on Grain v1: NFSR recovery



Getting the NFSR

- Using above technique 63 NFSR bits of S_{103} are recovered.
- LFSR bits of S_{103} already known(during PRGA LFSR evolution is autonomous).
- Not recovered $\Rightarrow [x_0, x_1, \ldots, x_{14}, x_{33}, x_{34}]^{103}$
- Solve the following equations to find the remaining bits

$$z_{102+\gamma} = x_{0+\gamma}^{103} + x_{1+\gamma}^{103} + x_{3+\gamma}^{103} + x_{9+\gamma}^{103} + x_{30+\gamma}^{103} + x_{42+\gamma}^{103} + x_{55+\gamma}^{103} + u_{102+\gamma} x_{62+\gamma}^{103} + v_{102+\gamma}$$

for $\gamma = 0, 1, \ldots, 14$.

Given $u_i = u(y_3^i, y_{25}^i, y_{46}^i, y_{64}^i)$ and $v_i = v(y_3^i, y_{25}^i, y_{46}^i, y_{64}^i)$.

- KSA and PRGA operations are easily invertible in Grain.

$$S_{103} \overset{PRGA^{-1}}{\rightarrow}{}_{103\ times}\ S_0 \overset{KSA^{-1}}{\rightarrow} SecretKey$$

Countermeasure

$F(s_0, s_1, s_2, s_3, s_4) = s_0 s_1 + s_1 s_2 + s_2 s_3 + s_3 s_4 + s_4 s_0 + s_0 s_2 + s_1 s_3 + s_2 s_4 + s_3 s_0 + s_4 s_1 + s_0 s_1 s_3 + s_1 s_2 s_4 + s_2 s_3 s_0 + s_3 s_4 s_1 + s_4 s_0 s_2.$

- F is affine differential resistant.
- F is a $(5, 3, 1, 12)$ function $\Rightarrow$ same as h.
- A realization of F in hardware takes just 8 more gates than h.

Discussion

- Fault attack was possible because h is not affine differential resistant.
- However, the assumptions in the attack are quite strong.
- Can Grain be fault-attacked under relaxed assumptions?

DFA on Grain with relaxed assumptions

- We assume that fault can be reproduced at a single location multiple number of times: optimistic and expensive.
- We have performed a differential fault attack on the Grain family by relaxing this assumption.
- No longer necessary to fault any location more than once.
- For more please visit **INDOCRYPT 2012**.

Another Follow up work on Grain-128a

- Grain-128a was proposed in SKEW 2011 by Ågren et. al.
- Outputs 32 bit MAC of any message and encrypts it as well.
- Using the same idea and by querying the device for faulty MACs of the empty message: Secret Key can be recovered.
- To be presented at **SPACE 2012**.

THANK YOU