Cryptanalysis of Grain

Côme Berbain[1], Henri Gilbert[1], and Alexander Maximov[2]

[1] France Telecom Research and Development
38-40 rue du Général Leclerc, 92794 Issy-les-Moulineaux, France
[2] Dept. of Information Technology, Lund University, Sweden
P.O. Box 118, 221 00 Lund, Sweden
{come.berbain, henri.gilbert}@francetelecom.com
movax@it.lth.se

Abstract. Grain [11] is a lightweight stream cipher proposed by M. Hell, T. Johansson, and W. Meier to the eSTREAM call for stream cipher proposals of the European project ECRYPT [5]. Its 160-bit internal state is divided into a LFSR and an NFSR of length 80 bits each. A filtering boolean function is used to derive each keystream bit from the internal state. By combining linear approximations of the feedback function of the NFSR and of the filtering function, it is possible to derive linear approximation equations involving the keystream and the LFSR initial state. We present a key recovery attack against Grain which requires 2^{43} computations and 2^{38} keystream bits to determine the 80-bit key.

Keywords: Stream cipher, Correlation attack, Walsh transform

1 Introduction

Stream ciphers are symmetric encryption algorithms based on the concept of pseudorandom keystream generator. In the typical case of a binary additive stream cipher, the key and an additional parameter named initialization vector (IV) are used to generate a binary sequence called keystream which is bitwise combined with the plaintext to provide the ciphertext. Although it seems rather difficult to construct a very fast and secure stream cipher, some efforts to achieve this have recently been deployed. The NESSIE project [24] launched in 1999 by the European Union did not succeed in selecting a secure enough stream cipher. Recently, the European Network of Excellence in Cryptology ECRYPT launched a call for stream cipher proposals named eSTREAM [5]. The candidate stream ciphers were submitted in May 2005. Those candidates are divided into software oriented and hardware oriented ciphers.

[0] The work described in this paper has been supported in part by Grant VR 621-2001-2149, in part by the French Ministry of Research RNRT X-CRYPT project and in part by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT.

Hardware oriented stream ciphers are specially designed so that their implementation requires a very small number of gates. Such ciphers are useful in mobile systems, e.g. mobile phones or RFID, where minimizing the number of gates and power consumption is more important than very high speed.

One of the new hardware candidates submitted to eSTREAM is a stream cipher named Grain [11] which was developed by M. Hell, T. Johansson, and W. Meier[3] as an alternative to stream ciphers like GSM A5/1 or Bluetooth E_0. It uses a 80-bit key and a 64-bit initialization vector to fill in an internal state of size 160 bits divided into a *nonlinear feedback shift register* (NFSR) and a *linear feedback shift register* (LFSR) of length 80 bits each. At each clock pulse, one keystream bit is produced by selecting some bits of the LFSR and of the NFSR and applying a boolean function. It is well known that LFSR sequences satisfy several statistical properties one would expect from a random sequence, but do not offer any security. Their combination with NFSR sequences is expected to improve the security. However, NFSR based constructions have not yet been as well studied as LFSR based constructions. The claimed security level of Grain is 2^{80}, and it was conjectured by the authors of Grain that there exists no attack significantly faster than exhaustive search.

In this paper, we describe two key recovery attacks against Grain. The proposed attacks exploit linear approximations of the output function. The first one requires 2^{55} operations, 2^{49} bits of memory, and 2^{51} keystream bits, and the second one requires 2^{43} operations, 2^{42} bits of memory, and 2^{38} keystream bits.

This paper is organized as follows. We first describe the Grain stream cipher (Section 2) and we derive some linear approximations involving the LFSR and the keystream (Section 3). We then present two techniques for recovering the initial state of the LFSR (Section 4). Finally, we present a technique allowing to recover the initial state of the NFSR once we know the LFSR initial state (Section 5).

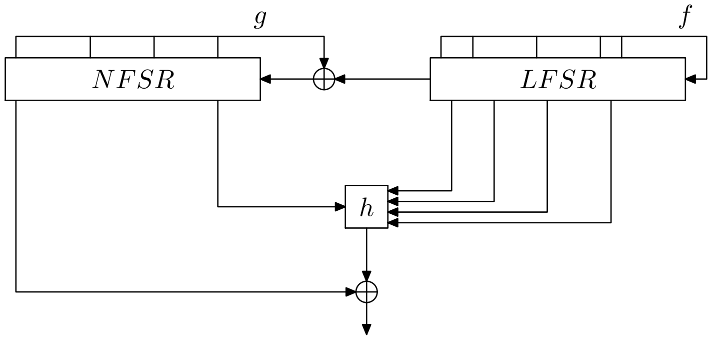
2 Description of Grain

Grain [11] is based upon three main building blocks: an 80-bit linear feedback shift register, an 80-bit nonlinear feedback shift register, and a nonlinear filtering function. Grain is initialized with the 80-bit key K and the 64-bit initialization value IV. The cipher output is an L-bit keystream sequence $(z_t)_{t=0,\dots,L-1}$.

The current LFSR content is denoted by $Y^t = (y_t, y_{t+1}, \dots, y_{t+79})$. The LFSR is governed by the linear recurrence:

$$y_{t+80} = y_{t+62} \oplus y_{t+51} \oplus y_{t+38} \oplus y_{t+23} \oplus y_{t+13} \oplus y_t.$$

[3] The design of Grain was also submitted and recently accepted for publication in *the International Journal of Wireless and Mobile Computing, Special Issue on Security of Computer Network and Mobile Systems.*



The current NFSR content is denoted by $X^t = (x_t, x_{t+1}, \ldots, x_{t+79})$. The NFSR feedback is disturbed by the output of the LFSR, so that the NFSR content is governed by the recurrence:

$$x_{t+80} = y_t \oplus g(x_t, x_{t+1}, \ldots, x_{t+79}),$$

where the expression of nonlinear feedback function g is given by

$$\begin{aligned} g(x_t, x_{t+1}, \ldots, x_{t+79}) = {} & x_{t+63} \oplus x_{t+60} \oplus x_{t+52} \oplus x_{t+45} \oplus x_{t+37} \oplus x_{t+33} \oplus x_{t+28} \\ & \oplus x_{t+21} \oplus x_{t+15} \oplus x_{t+9} \oplus x_t \oplus x_{t+63}x_{t+60} \oplus x_{t+37}x_{t+33} \\ & \oplus x_{t+15}x_{t+9} \oplus x_{t+60}x_{t+52}x_{t+45} \oplus x_{t+33}x_{t+28}x_{t+21} \\ & \oplus x_{t+63}x_{t+45}x_{t+28}x_{t+9} \oplus x_{t+60}x_{t+52}x_{t+37}x_{t+33} \\ & \oplus x_{t+63}x_{t+60}x_{t+21}x_{t+15} \oplus x_{t+63}x_{t+60}x_{t+52}x_{t+45}x_{t+37} \\ & \oplus x_{t+33}x_{t+28}x_{t+21}x_{t+15}x_{t+9} \\ & \oplus x_{t+52}x_{t+45}x_{t+37}x_{t+33}x_{t+28}x_{t+21}. \end{aligned}$$

The cipher output bit z_t is derived from the current LFSR and NFSR states as the exclusive or of the masking bit x_t and a nonlinear filtering function h as follows:

$$\begin{aligned} z_t &= x_t \oplus h(y_{t+3}, y_{t+25}, y_{t+46}, y_{t+64}, x_{t+63}) \\ &= h'(y_{t+3}, y_{t+25}, y_{t+46}, y_{t+64}, x_t, x_{t+63}) \\ &= x_t \oplus x_{t+63}p_t \oplus q_t, \end{aligned}$$

where p_t and q_t are the functions of $y_{t+3}, y_{t+25}, y_{t+46}, y_{t+64}$ given by:

$$\begin{aligned} p_t &= 1 \oplus y_{t+64} \oplus y_{t+46}(y_{t+3} \oplus y_{t+25} \oplus y_{t+64}), \\ q_t &= y_{t+25} \oplus y_{t+3}y_{t+46}(y_{t+25} \oplus y_{t+64}) \oplus y_{t+64}(y_{t+3} \oplus y_{t+46}). \end{aligned}$$

The boolean function h is correlation immune of the first order. As noticed in [11], "this does not preclude that there are correlations of the output of $h(x)$ to sums of inputs", but the designers of Grain appear to have expected the NFSR masking bit x_t to make it impractical to exploit such correlations.

The key and IV setup consists of loading the key bits in the NFSR, loading the 64-bit IV followed by 16 ones in the LFSR, and clocking the cipher 160 times in a special mode where the output bit is fed back into the LFSR and the NFSR. Once the key and IV have been loaded, the keystream generation mode described above is activated and the keystream sequence (z_t) is produced.

3 Deriving Linear Approximations of the LFSR Bits

3.1 Linear Approximations Used to Derive the LFSR Bits

The purpose of the attack is, based on a keystream sequence $(z_t)_{t=0\ldots L-1}$ corresponding to an unknown key K and a known IV value, to recover the key K. The initial step of the attack is to derive a sufficient number N of linear approximation equations involving the 80 bits of the initial LFSR state $Y^0 = (y_0, \ldots, y_{79})$ (or equivalently a sufficient number N of linear approximation equations involving bits of the sequence (y_t)) to recover the value of Y^0. Hereafter, as will be shown in Section 5, the initial NFSR state X^0 and the key K can then be easily recovered.

The starting point for the attack consists in noticing that though the NFSR feedback function g is balanced, the function g' given by $g'(X^t) = g(X^t) \oplus x_t$ is unbalanced. We have:

$$Pr\{g'(X^t) = 1\} = \frac{522}{1024} = \frac{1}{2} + \epsilon_{g'},$$

where $\epsilon_{g'} = \frac{5}{512}$. It is useful to notice that the restriction of g' to input values X^t such that $x_{t+63} = 0$ is totally balanced and that the imbalance of the function g' is exclusively due to the imbalance of the restriction of g' to input values X^t such that $x_{t+63} = 1$.

If one considers one single output bit z_t, the involvement of the masking bit x_t in the expression of z_t makes it impossible to write any useful approximate relation involving only the Y^t bits. But if one considers the sum $z_t \oplus z_{t+80}$ of two keystream bits output at a time interval equal to the NFSR length 80, the $x_t \oplus x_{t+80}$ contribution of the corresponding masking bits is equal to $g'(X^t) \oplus y_t$, and is therefore equal to y_t with probability $\frac{1}{2} + \epsilon_{g'}$. As for the other terms of $z_t \oplus z_{t+80}$, they can be approximated by linear functions of the bits of the sequence (y_t). In more details:

$$\begin{aligned} z_t \oplus z_{t+80} = g'(X^t) &\oplus y_t \oplus h(y_{t+3}, y_{t+25}, y_{t+46}, y_{t+64}, x_{t+63}) \\ &\oplus h(y_{t+83}, y_{t+105}, y_{t+126}, y_{t+144}, x_{t+143}). \end{aligned}$$

To find linear approximations of the term $h(y_{t+3}, y_{t+25}, y_{t+46}, y_{t+64}, x_{t+63})$, we can restrict our search, since the restriction of $g'(X^t)$ to input values such that $x_{t+63} = 0$ is balanced, to input values such that $x_{t+63} = 1$, which amounts to finding linear approximations of $p_t \oplus q_t$.

We found a set of two best linear approximations for this function, namely:

$$L_1 = \{y_{t+3} \oplus y_{t+25} \oplus y_{t+64} \oplus 1; y_{t+25} \oplus y_{t+46} \oplus y_{t+64} \oplus 1\}.$$

Each of the approximations of L_1 is valid with a probability $\frac{1}{2}+\epsilon_1$, where $\epsilon_1 = \frac{1}{4}$.

Now the term $h(y_{t+83}, y_{t+105}, y_{t+126}, y_{t+144}, x_{t+143})$ is equal to either q_{t+80} or $p_{t+80} \oplus q_{t+80}$, with a probability $\frac{1}{2}$ for both expressions. We found a set of 8 best simultaneous linear approximations for these two expressions, namely:

$$\begin{aligned} L_2 = \{ \; & y_{t+83} \oplus y_{t+144} \oplus 1; && y_{t+83} \oplus y_{t+105} \oplus y_{t+126} \oplus y_{t+144} \oplus 1; \\ & y_{t+83} \oplus y_{t+126} \oplus y_{t+144}; && y_{t+83} \oplus y_{t+105} \oplus y_{t+126}; \\ & y_{t+83} \oplus y_{t+105}; && y_{t+83} \oplus y_{t+105} \oplus y_{t+144} \oplus 1; \\ & y_{t+105} \oplus y_{t+144}; && y_{t+105} \oplus y_{t+126} \oplus y_{t+144} \oplus 1; \}. \end{aligned}$$

Each of the 8 approximations of L_2 has an average probability $\epsilon_2 = \frac{1}{8}$ of being valid.

Thus, we have found 16 linear approximations of $z_t \oplus z_{t+80}$, namely all the linear expressions of the form

$$y_t \oplus l_1(y_{t+3}, y_{t+25}, y_{t+46}, y_{t+64}) \oplus l_2(y_{t+83}, y_{t+105}, y_{t+126}, y_{t+144}),$$

where $l_1 \in L_1$ and $l_2 \in L_2$. Each of these approximations is valid with a probability $\frac{1}{2} + \epsilon$, where ϵ is derived from $\epsilon_{g'}$, ϵ_1, and ϵ_2 using the Piling-up Lemma:

$$\epsilon = \frac{1}{2} \cdot 2^2 \cdot \epsilon_{g'} \cdot \epsilon_1 \cdot \epsilon_2 = \frac{5}{4096} \simeq 2^{-9.67}.$$

The extra multiplicative factor of $\frac{1}{2}$ takes into account the fact that the considered approximations are only valid when $x_{t+63} = 1$. The LFSR derivation attacks of Section 4 exploit these 16 linear approximations.

3.2 Generalisation of the Attack Method

In this Section, we try to generalise the previous approximation method. The purpose is not to find better approximations than those identified in Section 3.1, but to derive some design criteria on the boolean functions g and h'. However in the previous approximation, we used the fact that the bias of g depends on the value of x_{t+63}, so that the approximations of g and h' are not correct independently. We do not take this phenomenon into account in this Section. Therefore, we only provide a simplified picture of potential generalised attacks.

The function $g(X^t, Y^t)$ operates on $w(g) = w_L(g) + w_N(g)$ variables taken from the LFSR and the NFSR, where $w_L(g)$ is the number of variables taken from the LFSR and $w_N(g)$ the number of variables taken from the NFSR. Let the function $A_g(X^t, Y^t)$ be a linear approximation of the function g, i.e.

$$A_g(X^t, Y^t) = \bigoplus_{i=0}^{w_N(g)-1} d_i x_{t+\phi_g(i)} \oplus \bigoplus_{j=0}^{w_L(g)-1} c_j y_{t+\psi_g(j)}, \quad c_j, d_i \in \mathbb{F}_2, \qquad (1)$$

such that the distance between $g(\cdot)$ and $A_g(\cdot)$ defined by:

$$d_g = \sharp\{x \in \mathbb{F}_2^{w(g)} : A_g(x) \neq g(x)\} > 0,$$

is strictly larger than zero. Then, we have

$$\Pr\{A_g(x) \neq g(x)\} = \frac{1}{2^{w(g)}} d_g,$$

i.e.

$$\Pr\{A_g(x) + g(x) = 0\} = 1/2 + \epsilon_g,$$

where the bias is:

$$\epsilon_g = 1/2 - 2^{-w(g)} d_g.$$

Similarly, the function $h'(X^t, Y^t)$ can also be approximated by some linear expressions of the form:

$$A_{h'}(X^t, Y^t) = \bigoplus_{i=0}^{w_N(h')-1} k_i x_{t+\phi_{h'}(i)} \oplus \bigoplus_{j=0}^{w_L(h')-1} l_j y_{t+\psi_{h'}(j)}, \quad k_j, l_i \in \mathbb{F}_2. \tag{2}$$

Recall, $z_t \stackrel{p}{=} A_{h'}(\cdot)_t$ with some probability p. Knowing the expressions (1) and (2), one can sum up together $w_N(A_g(\cdot))$ expressions of $A_{h'}(\cdot)$ at different times t, in such a way that all terms X^t will be eliminated (just because the terms X^t will be cancelled due to the parity check function $A_g(\cdot)$, leaving the terms Y^t and noise variables only). Note also that any linear combination of $A_{h'}(\cdot)$ is a linear combination of the keystream bits z_t.

The sum of $w_N(A_g(\cdot))$ approximations $A_{h'}(\cdot)$ will introduce $w_N(A_g(\cdot))$ independent noise variables due to the approximation at different time instances. Moreover, the cancellation of the terms X^t in the sum will be done by the parity check property of the approximation $A_g(\cdot)$. If the function $A_{h'}(\cdot)$ contains $w_N(A_{h'})$ terms from X^t, then the parity cancellation expression $A_g(\cdot)$ will be applied $w_N(A_{h'})$ times. Each application of the cancellation expression $A_g(\cdot)$ will introduce another noise variable due to the approximation $N_g : g(\cdot) \to A_g(\cdot)$. Therefore, the application of the expression $A_g(\cdot)$ $w_N(A_{h'})$ times will introduce $w_N(A_{h'})$ additional noise variables N_g. Accumulating all above and following the Piling-up Lemma, the final correlation of such a sum (of the linear expression on Y^t) is given by the following Theorem.

Theorem 1. *There always exists a linear relation in terms of bits from the state of the LFSR and the keystream, which have the bias:*

$$\epsilon = 2^{(w_N(A_{h'}) + w_N(A_g) - 1)} \cdot \epsilon_g^{w_N(A_{h'})} \cdot \epsilon_{h'}^{w_N(A_g)},$$

where $A_g(\cdot)$ and $A_{h'}(\cdot)$ are linear approximations of the functions $g(\cdot)$ and $h'(\cdot)$, respectively, and:

$$\Pr\{A_g(\cdot) = g(\cdot)\} = 1/2 + \epsilon_g, \qquad \Pr\{A_{h'}(\cdot) = h'(\cdot)\} = 1/2 + \epsilon_{h'}.$$

This theorem gives us a criteria for a proper choice of the functions $g(\cdot)$ and $h'(\cdot)$. The biases ϵ_g and $\epsilon_{h'}$ are related to the *nonlinearity* of these boolean functions, and the values $w_N(A_g)$ and $w_N(A_{h'})$ are related to the *correlation immunity* property; however, there is a well-known trade-off between these two properties [27]. Unfortunately, in the case of Grain the functions $g(\cdot)$ and $h'(\cdot)$ were improperly chosen.

4 Deriving the LFSR Initial State

In the former Section, we have shown how to derive an arbitrary number N of linear approximation equations in the $n = 80$ initial LFSR bits, of bias $\epsilon \simeq 2^{-9.67}$ each, from a sufficient number of keystream bits. Let us denote these equations by:

$$\bigoplus_{i=0}^{n-1} \alpha_i^j \cdot y_i = b^j, j = 1, \ldots, N.$$

In this Section we show how to use these relations to derive the initial LFSR state Y^0. This can be seen as a decoding problem, up to the fact that the code length is not fixed in advance and one has to find an optimal trade-off between the complexities of deriving a codeword (i.e. collecting an appropriate number of linear approximation equations) and decoding this codeword.

An estimate of the number N of linear approximation equations needed for the right value of the unknown to maximize the indicator

$$I = \sharp \left\{ j \in \{1, \ldots, N\} \;\middle|\; \bigoplus_{i=0}^{n-1} \alpha_i^j \cdot y_i = b^j \right\},$$

or at least to be very likely to provide say one of the two or three highest values of I, can be determined as follows.

Under the heuristic assumption that for the correct (respectively incorrect) value of Y^0, I is the sum of N independent binary variables x_i distributed according to the Bernoulli law of parameters $p = Pr\{x_i = 1\} = \frac{1}{2} - \epsilon$ and $q = Pr\{x_i = 0\} = \frac{1}{2} + \epsilon$ (resp. the Bernoulli law of parameters $Pr\{x_i = 1\} = \frac{1}{2}$ and $Pr\{x_i = 0\} = \frac{1}{2}$, mean value $\mu = \frac{1}{2}$, and standard deviation $\sigma = \frac{1}{2}$), N can be derived by introducing a threshold of say $T = N(\frac{1}{2} + \frac{3\epsilon}{4})$ for I and requiring: *(i)* that the probability that I is larger than T for an incorrect value of Y^0 is less than a suitably chosen false alarm probability p_{fa}; *(ii)* that the probability that I is lower than T for the correct value is less than a non detection probability p_{nd} of say 1%. For practical values of p_{fa}, the first condition is by far the most demanding. Setting the false alarm rate to $p_{fa} = 2^{-n}$ ensures that the number of false alarms is less than 1 in average.

Due to the Central Limit Theorem, $\frac{\sum x_i - N\mu}{\sqrt{N}\sigma}$ is distributed according to the normal law, so that:

$$Pr\left\{ \frac{1}{N} \sum x_i - \mu > \frac{3\epsilon}{4} \right\} = Pr\left\{ \frac{\sum x_i - N\mu}{\sqrt{N}\sigma} > \frac{3\sqrt{N}\epsilon}{4\sigma} \right\} \tag{3}$$

can be approximated by $\frac{1}{\sqrt{2\pi}} \int_\lambda^{+\infty} e^{-\frac{t^2}{2}} dt$, where $\lambda = \frac{3\sqrt{N}\epsilon}{2}$. Consequently, if N is selected in such a way that $\frac{3\sqrt{N}\epsilon}{2} = \lambda$, i.e.

$$N = \left(\frac{2\lambda}{3\epsilon} \right)^2,$$

where λ is given by:

$$\frac{1}{\sqrt{2\pi}}\int_{\lambda}^{+\infty} e^{-\frac{t^2}{2}} dt = p_{fa} = 2^{-n},$$

then inequality 3 is satisfied.

A naive LFSR derivation method would consist of collecting N approximate equations, computing the indicator I independently for each of the 2^n possible values of Y^0 and retaining those Y^0 candidates leading to a value of I larger than the $N(\frac{1}{2}+\frac{3\epsilon}{4})$ threshold. This method would require a low number of keystream bits (say $\frac{N+80}{16}$) but the resulting complexity $N \cdot 2^{80}$ would be larger than the one of exhaustive key search.

In the rest of this Section, we show that much lower complexities can be obtained by using the fast Walsh transform algorithm and a few extra filtering techniques in order to speed up computations of correlation indicators. Former examples of applications of similar Fast Fourier Transform techniques in order to significantly decrease the total complexity of correlation attacks can be found in [4] [9] [16].

4.1 Use of the Fast Walsh Transform to Speed up Correlation Computations

Basic Method. Let us consider the following problem. Given a sufficient number M of linear approximation equations of bias ϵ involving m binary variables y_0 to y_{m-1}, how to efficiently determine these m variables? Let us denote these M equations by $\sum_{i=0}^{m-1} \alpha_i^j \cdot y_j = b^j, j = 1, \ldots, M$. For a sufficiently large value of M, one can expect the right value of $(y_0, \ldots, y_{m-1})$ to be the one maximizing the indicator:

$$\begin{aligned} I(y_0, \ldots, y_{m-1}) &= \sharp\left\{ j \in \{1, \ldots, M\} \;\middle|\; \sum_{i=0}^{m-1} \alpha_i^j \cdot y_j = b^j \right\} \\ &= \frac{M}{2} + \frac{1}{2} \cdot S(y_0, \ldots, y_{m-1}), \end{aligned}$$

where:

$$\begin{aligned} S(y_0, \ldots, y_{m-1}) = {} & \sharp\left\{ j \in \{1, \ldots, M\} \;\middle|\; \sum_{i=0}^{m-1} \alpha_i^j \cdot y_i = b^j \right\} \\ & - \sharp\left\{ j \in \{1, \ldots, M\} \;\middle|\; \sum_{i=0}^{m-1} \alpha_i^j \cdot y_i \neq b^j \right\}. \end{aligned}$$

Equivalently one can expect $(y_0, \ldots, y_{m-1})$ to be the value which maximizes the indicator $S(y_0, \ldots, y_{m-1})$. Instead of computing all of 2^m values of $S(y_0, \ldots, y_{m-1})$ independently, one can derive these values in a combined way using fast Walsh transform computations in order to save time.

Let us recall the definition of the Walsh transform. Given a real function of m binary variables $f(x_1, \ldots, x_{m-1})$, the Walsh transform of f is the real function of m binary variables $F = W(f)$ defined by:

$$F(u_0, \ldots, u_{m-1}) = \sum_{x_0, \ldots, x_{m-1} \in \{0,1\}^m} f(x_0, \ldots, x_{m-1})(-1)^{u_0 x_0 + \ldots + u_{m-1} x_{m-1}}.$$

Let us define the function $s(\alpha_0, \ldots, \alpha_{m-1})$ by:

$$\begin{aligned} &\sharp\{j \in \{1, \ldots, M\} \mid (\alpha_0^j, \ldots, \alpha_{m-1}^j) = (\alpha_0, \ldots, \alpha_{m-1}) \wedge b^j = 1\} \\ -\, &\sharp\{j \in \{1, \ldots, M\} \mid (\alpha_0^j, \ldots, \alpha_{m-1}^j) = (\alpha_0, \ldots, \alpha_{m-1}) \wedge b^j = 0\}. \end{aligned}$$

The function s can be computed in M steps. Moreover, it is easy to check that the Walsh transform of s is S, i.e.

$$\forall (y_0, \ldots, y_{m-1}) \in \{0,1\}^m, W(s)(y_0, \ldots, y_{m-1}) = S((y_0, \ldots, y_{m-1})).$$

Therefore, the computational cost of the estimation of all the 2^m values of S using fast Walsh transform computations is $M + m \cdot 2^m$; the required memory is 2^m.

Improved Hybrid Method. More generally, if $m_1 < m$, one can use the following hybrid method between exhaustive search and Walsh transform in order to save space.

For each of the 2^{m-m_1} values of $(y_{m_1}, \ldots, y_{m-1})$, define the associated restriction S' of S as the m_1 bit boolean function given by:

$$\begin{aligned} S'(y_0, \ldots, y_{m_1-1}) = &\sharp\left\{j \in \{1, \ldots, M\} \;\middle|\; \sum_{i=0}^{m_1-1} \alpha_i^j \cdot y_i = \sum_{i=m_1}^{m} \alpha_i^j \cdot y_i \oplus b^j \right\} \\ &- \sharp\left\{j \in \{1, \ldots, M\} \;\middle|\; \sum_{i=0}^{m_1-1} \alpha_i^j \cdot y_i \neq \sum_{i=m_1}^{m} \alpha_i^j \cdot y_i \oplus b^j \right\}. \end{aligned}$$

It is easy to see that if we define $s'(\alpha_0, \ldots, \alpha_{m_1-1})$ as

$$\begin{aligned} &\sharp\left\{j \in \{1, \ldots, M\} \;\middle|\; (\alpha_0^j, \ldots, \alpha_{m_1-1}^j) = (\alpha_0, \ldots, \alpha_{m_1-1}) \wedge \sum_{i=m_1}^{m} \alpha_i^j \cdot y_i \oplus b^j = 1 \right\} \\ -&\sharp\left\{j \in \{1, \ldots, M\} \;\middle|\; (\alpha_0^j, \ldots, \alpha_{m_1-1}^j) = (\alpha_0, \ldots, \alpha_{m_1-1}) \wedge \sum_{i=m_1}^{m} \alpha_i^j \cdot y_i \oplus b^j = 0 \right\}, \end{aligned}$$

then S' is the Walsh transform of s'.

Therefore, the computational cost of the estimation of all the 2^m values of S using this method is $2^{m-m_1}(M + m_1 \cdot 2^{m_1})$. If we compare this with the former basic Walsh transform method, we see that the required memory decreases from 2^m to 2^{m_1}, whereas the time complexity increase remains negligible as long as $m_1 << log_2(M)$.

4.2 First LFSR Derivation Technique

In order to reduce the LFSR derivation complexity when compared with the naive method of complexity $N \cdot 2^n$, we can exploit more keystream to produce more linear approximation equations in the unknowns y_0 to y_{n-1}, and retain only those equations involving the $m < n$ variables y_0 to y_{m-1}, i.e. which coefficients in the $n - m$ variables y_m to y_{n-1} are equal to 0.

Thus a fraction of about 2^{m-n} of the relations are retained and we have to collect about $N2^{n-m}$ approximate relations to retain N relations. This requires a number of keystream bits of:

$$\frac{N2^{n-m} + 80}{16}.$$

As seen in the former Section, once the relations have been filtered, the computational cost of the derivation of the values of these m variables using fast Walsh transform computations is about $m2^m$ for the basic method, and more generally $2^{m-m_1}(N + m_1 2^{m_1})$ if fast Walsh transform computations are applied to a restricted set of $m_1 < m$ variables.

Thus, the overall time complexity of this method is:

$$N2^{n-m} + m2^m,$$

and more generally:

$$N2^{n-m} + 2^{m-m_1}(N + m_1 2^{m_1}).$$

Once the m variables y_0 to y_{m-1} have been recovered, one can either reiterate the same technique for other choices of the m unknown variables, which increases the complexity by a factor of less than 2 if $m \geq \frac{n}{2}$, or test each of the 2^{n-m} candidates in the next step of the attack (NFSR and key derivation).

An estimate of the number N of equations needed is given by

$$N = \left(\frac{2\lambda}{3\epsilon}\right)^2,$$

where λ is determined by the condition $\frac{1}{\sqrt{2\pi}} \int_{\lambda}^{+\infty} e^{-\frac{t^2}{2}} dt = 2^{-m}$. This condition ensures that the expected number of false alarm is less than 1.

The minimal complexity is obtained for $m = 49$. For this parameter value, we have $\lambda = 7.87$ and $N = 2^{24}$. The attack complexity is about 2^{55}, the number of keystream bits needed is around 2^{51}, and the memory needed is about 2^{49}.

4.3 Second LFSR Derivation Technique

An alternative method is to derive new linear approximation equations (of lower bias) involving $m < n$ unknown variables y_0 to y_{m-1} by combining the R available approximate equations of bias ϵ pairwise, and retaining only those pairs of relations for which the $n-m$ last coefficients collide. One obtains in this way about $N' = R^2 \cdot 2^{m-n-1}$ new affine equations in y_0 to y_{m-1}, of bias $\epsilon' = 2\epsilon^2$. The allocation of the m variables maximizing the number of satisfied equations can be found by fast Walsh computations as explained in the former Section.

The number N' of relations needed is about $\left(\frac{2\lambda}{3\epsilon'}\right)^2$, where λ is determined by the condition $\frac{1}{\sqrt{2\pi}}\int_{\lambda}^{+\infty} e^{-\frac{t^2}{2}}dt = 2^{-m}$. The required number R of relations of bias ϵ is therefore $R = (N'2^{n-m+1})^{\frac{1}{2}}$, and the number of keystream bits needed is about $\frac{R+80}{16}$. The complexity of the derivation of the N' relations is $\max(R, N') = \max((N'2^{n-m+1})^{\frac{1}{2}}, N')$.

Once the N' relations have been derived, the computational cost of the derivation of the values of these m variables using fast Walsh transform computations is about $m \cdot 2^m$ for the basic method, and more generally if fast Walsh transform computations are applied to a restricted set of $m_1 < m$ variables it costs $2^{m-m_1}(N' + m_1 \cdot 2^{m_1})$.

Thus the total complexity of the derivation of the m LFSR bits is:

$$\max((N'2^{n-m+1})^{\frac{1}{2}}, N') + m2^m,$$

and more generally:

$$\max((N'2^{n-m+1})^{\frac{1}{2}}, N') + 2^{m-m_1}(N' + m_1 2^{m_1}).$$

The minimal complexity is obtained for $m = 36$. For this parameter value, we have $\lambda = 6.65$ and $N' = 2^{41}$. The attack complexity is about 2^{43}, the number of keystream bits needed is about 2^{38} and the memory required is about 2^{42}.

5 Recovering the NFSR Initial State and the Key

Once the initial state of the LFSR has been recovered, we want to recover the initial state $(x_0, \ldots, x_{79})$ of the NFSR. Fortunately, the knowledge of the LFSR removes the nonlinearity of the output function and we can express each keystream bit z_i by one of the following four equations depending on the initial state of the LFSR:

$$\begin{aligned} z_i &= x_i, & z_i &= x_i \oplus 1, \\ z_i &= x_i \oplus x_{63+i}, & z_i &= x_i \oplus x_{63+i} \oplus 1. \end{aligned}$$

Since functions p and q underlying h are balanced, each equation has the same occurrence probability. We are going to use the non linearity of the output function to recover the initial state of the NFSR by writing the equations corresponding to the first keystream bits.

The 16 first equations are linear equations involving only bits of the initial state of the NFSR because $63 + i$ is lower than 80.

To recover all the bits of the initial state, we introduce a technique which consists of building chains of keystream bits. The equations for keystream bits z_{17} to z_{79} involve either one bit of the NFSR ($z_i = x_i$ or $z_i = x_i \oplus 1$) or two bits ($z_i = x_i \oplus x_{63+i}$ or $z_i = x_i \oplus x_{63+i} \oplus 1$). An equation involving only one bit allows us to instantly recover the value of the corresponding bit of the initial state. This can be considered as a chain of length 0. On the other hand, an equation involving two bits does not allow this because we do not know the value of x_{63+i} (for $i > 16$).

However, by considering not only the equations for z_i but also all the equation for $z_{k\cdot 63+i}$ for $k \geq 1$, we can cancel the bits we do not know and retrieve the value of x_i. With probability $\frac{1}{2}$, the equation for z_{63+i} involves one single unknown bit. Then it provides the value of x_{63+i} and consequently the value of x_i. Here the chain is of length 1, since we have to consider one extra equation to retrieve x_i. The equation for z_{63+i} can also involve two bits with probability $\frac{1}{2}$. Then we have to consider the equation of $z_{2\cdot 63+i}$, which can also either involve only one bit (we have a chain of length 2) or two bits and we have to consider more equations to solve. Each equation has a probability $\frac{1}{2}$ to involve 1 or 2 bits. Consequently the probability that a chain is of length n is $\frac{1}{2^{n+1}}$ and the probability that a chain is of length strictly larger than n is $\frac{1}{2^{n+1}}$.

We want to recover the values of $x_{17}, \ldots, x_{79}$. We have to build 64 different chains. Let us consider $L = 63 \cdot n$ bits of keystream. The probability that one of the chains is of length larger than n is less than $= 64 \cdot 2^{-n-1}$ and therefore less than 2^{-n+5}. If we want this probability to be bounded by 2^{-10}, then $n > 15$ and $L > 945$ suffices. Consequently a few thousands of keystream bits are required to retrieve the initial state of the NFSR and the complexity of the operation is bounded by $64 \cdot n$.

Since the internal state transition function associated to the special key and IV setup mode is one to one, the key can be efficiently derived from the NFSR and LFSR states at the beginning of the keystream generation by running this function backward.

6 Simulations and Results

To confirm that our cryptanalysis is correct, we ran several experiments. First we checked the bias ϵ of Section 3.1 by running the cipher with a known initial state of both the LFSR and the NFSR, computing the linear approximations, and counting the number of fulfilled relations for a very large number of relations. For instance we found that one linear approximation is satisfied 19579367 times out of 39060639, which gives an experimental bias of $2^{-9.63}$, to be compared with the theoretical bias $\epsilon = 2^{-9.67}$.

To check the two proposed LFSR reconstruction methods of Section 4, we considered a reduced version of Grain in order to reduce the memory and time required by the attack on a single computer: we shortened the LFSR by a factor of 2. We used an LFSR of size 40 with a primitive feedback polynomial and we reduced by two the distances for the tap entries of function h: we selected taps number 3, 14, 24, and 33, instead of 3, 25, 46, and 64 for Grain.

The complexity of the first technique for the actual Grain is 2^{55} which is out of reach of a single PC. For our reduced version, the complexity given by the formula of Section 4.2 is only 2^{35}. We exploited the 16 linear approximations to derive relations colliding on the first 11 bits. Consequently the table of the Walsh transform is only of size 2^{29}. We used $15612260 \simeq 2^{23}$ relations, which corresponds to a false alarm probability of 2^{-29}. Our implementation needed around one hour to recover the correct value of the LFSR internal state on a computer with a Intel Xeon processor running at 2.5 GHz with 3 GB of memory. The Walsh transform computation took only a few minutes.

For the actual Grain, the second technique requires only 2^{43} operations which is achievable by a single PC. However it also requires 2^{42} of memory which corresponds to 350 GB of memory. We do not have such an amount of memory but for the reduced version the required memory is only 2^{29}. Since the complexity given by the formula of Section 4.3 is dominated by the required number of relations to detect the bias, our simulation has a complexity close to 2^{43}. In practice, we obtained a result after 4 days of computation on the same computer as above and $2.5 \cdot 10^{12} \simeq 2^{41}$ relations where considered and allowed to recover the correct LFSR initial state.

Finally, we implemented the method of Section 5 to recover the NFSR. Given the correct initial state of the LFSR, and the first thousand keystream bits, our program recovers the initialization of the NFSR in a few seconds for a large number of different initializations of both the known LFSR and unknown NFSR. We also confirmed the failure probability assessed in Section 5 for this method (which corresponds to the occurrence probability of at least one chain of length larger than 15).

7 Conclusion

We have presented a key-recovery attack against Grain which requires 2^{43} computations, 2^{42} bits of memory, and 2^{38} keystream bits. This attack suggests that the following slight modifications of some of the Grain features might improve its strength:

- Introduce several additional masking variables from the NFSR in the keystream bit computation.
- Replace the nonlinear feedback function g in such a way that the associated function g' be balanced (e.g. replace g by a 2-resilient function). However this is not necessarily sufficient to thwart all similar attacks.
- Modify the filtering function h in order to make it more difficult to approximate.
- Modify the function g and h to increase the number of inputs.

Following recent cryptanalysis of Grain including the key recovery attack reported here and distinguishing attacks based on the same kind of linear approximations as those presented in Section 3 [19] [26], the authors of Grain proposed a tweaked version of their algorithm [12], where the functions g and h' have been modified. This novel version of Grain appears to be much stronger and is immune against the statistical attacks presented in this paper.

We would like to thank Matt Robshaw and Olivier Billet for helpful comments.

References

1. M. Briceno, I. Goldberg, and D. Wagner. A pedagogical implementation of A5/1. Available at `http://jya.com/a51-pi.htm`, Accessed August 18, 2003, 1999.
2. A. Canteaut and M. Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In B. Preneel, editor, *Advances in Cryptology—EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 573–588. Springer-Verlag, 2000.
3. V. Chepyzhov and B. Smeets. On a fast correlation attack on certain stream ciphers. In D. W. Davies, editor, *Advances in Cryptology—EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 176–185. Springer-Verlag, 1991.
4. M. W. Dodd. *Applications of the Discrete Fourier Transform in Information Theory and Cryptology.* PhD thesis, University of London, 2003.
5. ECRYPT. eSTREAM: ECRYPT Stream Cipher Project, IST-2002-507932. Available at `http://www.ecrypt.eu.org/stream/`, Accessed September 29, 2005, 2005.
6. P. Ekdahl and T. Johansson. Another attack on A5/1. In *Proceedings of International Symposium on Information Theory*, page 160. IEEE, 2001.
7. P. Ekdahl and T. Johansson. Another attack on A5/1. *IEEE Transactions on Information Theory*, 49(1):284–289, January 2003.
8. H. Englund and T. Johansson. A new simple technique to attack filter generators and related ciphers. In *Selected Areas in Cryptography*, pages 39–53, 2004.
9. H. Gilbert and P. Audoux. Improved fast correlation attacks on stream ciphers using FFT techniques. personnal communication, 2000.
10. J.D. Golić. Cryptanalysis of alleged A5 stream cipher. In W. Fumy, editor, *Advances in Cryptology—EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 239–255. Springer-Verlag, 1997.
11. M. Hell, T. Johansson, and W. Meier. Grain - A Stream Cipher for Constrained Environments. ECRYPT Stream Cipher Project Report 2005/001, 2005. `http://www.ecrypt.eu.org/stream`.
12. M. Hell, T. Johansson, and W. Meier. Grain - A Stream Cipher for Constrained Environments, 2005. `http://www.it.lth.se/grain`.
13. T. Johansson and F. Jönsson. Fast correlation attacks based on turbo code techniques. In *Advances in Cryptology—CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 181–197. Springer-Verlag, 1999.
14. T. Johansson and F. Jönsson. Improved fast correlation attacks on stream ciphers via convolutional codes. In *Advances in Cryptology—EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 347–362. Springer-Verlag, 1999.
15. F. Jönsson. *Some Results on Fast Correlation Attacks.* PhD thesis, Lund University, Department of Information Technology, P.O. Box 118, SE–221 00, Lund, Sweden, 2002.
16. A. Joux, P. Chose, and M. Mitton. Fast Correlation Attacks: An Algorithmic Point of View. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 209–221. Springer-Verlag, 2002.
17. B. S. Jr. Kaliski and M. J. B. Robshaw. Linear Cryptanalysis Using Multiple Approximations. In Yvo G. Desmedt, editor, *Advances in Cryptology – CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 26–39. Springer-Verlag, 1994.

18. M. Matsui. Linear cryptanalysis method for DES cipher. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer-Verlag, 1993.
19. A. Maximov. Cryptanalysis of the "Grain" family of stream ciphers. In *ACM Transactions on Information and System Security (TISSEC)*, 2006.
20. W. Meier and O. Staffelbach. Fast correlation attacks on stream ciphers. In C.G. Günter, editor, *Advances in Cryptology—EUROCRYPT'88*, volume 330 of *Lecture Notes in Computer Science*, pages 301–316. Springer-Verlag, 1988.
21. W. Meier and O. Staffelbach. Fast correlation attacks on certain stream ciphers. *Journal of Cryptology*, 1(3):159–176, 1989.
22. W. Meier and O. Staffelbach. The self-shrinking generator. In A. De Santis, editor, *Advances in Cryptology—EUROCRYPT'94*, volume 905 of *Lecture Notes in Computer Science*, pages 205–214. Springer-Verlag, 1994.
23. M. Mihaljevic and J.D. Golić. A fast iterative algorithm for a shift register initial state reconstruction given the noisy output sequence. In J. Seberry and J. Pieprzyk, editors, *Advances in Cryptology—AUSCRYPT'90*, volume 453 of *Lecture Notes in Computer Science*, pages 165–175. Springer-Verlag, 1990.
24. NESSIE. New European Schemes for Signatures, Integrity, and Encryption. Available at `http://www.cryptonessie.org`, Accessed August 18, 2003, 1999.
25. W.T. Penzhorn and G.J. Kühn. Computation of low-weight parity checks for correlation attacks on stream ciphers. In C. Boyd, editor, *Cryptography and Coding - 5th IMA Conference*, volume 1025 of *Lecture Notes in Computer Science*, pages 74–83. Springer-Verlag, 1995.
26. M. Hassanzadeh S. Khazaei and M. Kiaei. Distinguishing Attack on Grain. ECRYPT Stream Cipher Project Report 2005/001, 2005. `http://www.ecrypt.eu.org/stream`.
27. T. Siegenthaler. Correlation-immunity of non-linear combining functions for cryptographic applications. *IEEE Transactions on Information Theory*, 30:776–780, 1984.
28. T. Siegenthaler. Decrypting a class of stream ciphers using ciphertext only. *IEEE Transactions on Computers*, 34:81–85, 1985.