Correlation attacks using a new class of weak feedback polynomials

Håkan Englund, Martin Hell and Thomas Johansson

Dept. of Information Techonolgy, Lund University,
P.O. Box 118, 221 00 Lund, Sweden

Abstract. In 1985 Siegenthaler introduced the concept of correlation attacks on LFSR based stream ciphers. A few years later Meier and Staffelbach demonstrated a special technique, usually referred to as fast correlation attacks, that is very effective if the feedback polynomial has a special form, namely, if its weight is very low. Due to this seminal result, it is a well known fact that one avoids low weight feedback polynomials in the design of LFSR based stream ciphers.
This paper identifies a new class of such weak feedback polynomials, polynomials of the form $f(x) = g_1(x) + g_2(x)x^{M_1} + \ldots + g_t(x)x^{M_{t-1}}$, where $g_1, g_2, \ldots, g_t$ are all polynomials of low degree. For such feedback polynomials, we identify an efficient correlation attack in the form of a distinguishing attack.

1 Introduction

Stream cipher design and cryptanalysis are topics that have received lots of attention recently, due to new interesting designs that are very fast in software, see e.g. [3,8,10,13,20] and many others. One way to design a stream cipher is to use a Linear Feedback Shift Register (LFSR) sequence as input to a nonlinear function. The shift register is initialized using a short random string and the output from the cipher is a much longer string that has many random like properties. But the LFSR output is linear and in some way the linearity of the sequence must be destroyed through some nonlinear process.

Different attacks can be used to attack the nonlinear part of a cipher. A popular topic lately has been algebraic attacks [5,6]. These attacks can be mounted if the nonlinear function gives rise to a large system of equations containing equations of low degree. A different attack is to use the correlation between the input and the output of the nonlinear function.

In 1985 Siegenthaler introduced the concept of correlation attacks on LFSR based stream ciphers. His basic idea was to perform a divide-and-conquer attack by exploring the correlation between the output sequence of the generator and the output sequence of one individual LFSR (assuming a nonlinear combination generator).

A few years later Meier and Staffelbach demonstrated a special technique, usually referred to as fast correlation attacks [17]. This attack is very effective if the feedback polynomial has a special form, namely, if its weight is very low.

The ideas resemble a lot so-called iterative decoding of error correcting codes, for example low density parity check codes. Since then, many ideas concerning fast correlation attacks have been presented, see e.g. [1, 2, 11, 14, 15, 19]. Due to this seminal result, it is a well known fact that one avoids low weight feedback polynomials in the design of LFSR based stream ciphers.

This paper identifies a new class of such weak feedback polynomials, namely, polynomials of the form

$$f(x) = g_1(x) + g_2(x)x^{M_1} + \ldots + g_t(x)x^{M_{t-1}},$$

where $g_1, g_2, \ldots, g_l$ are all polynomials of low degree. For such feedback polynomials, we identify an efficient correlation attack in the form of a distinguishing attack.

In a distinguishing attack the key is not recovered, instead one tries to distinguish an observed keystream from a truly random stream. This attack is not as powerful as the key recovery attack, in which one finds the key that corresponds to the plaintext and the ciphertext. Whereas most previous work on correlation attacks have been focused on key recovery attacks, more recent work in cryptanalysis of stream ciphers have, to a large extent, been concerned with distinguishing attacks, see e.g. [4, 9].

It should also be noted that a distinguishing attack can sometimes be turned into a key recovery attack, in a similar way as for block ciphers.

The results of the paper are as follows. For the new class of such weak feedback polynomials, given above, we present an algorithm for launching an efficient fast correlation attack. The applicability of the algorithm is twofold.

Firstly, if the feedback polynomial is of the above form with a moderate number of polynomials ,g_i, the new algorithm will be much more powerful than applying any previously known (like Meier-Staffelbach) algorithm. This could be interpreted as feedback polynomials of the above kind should be avoided in designing new LFSR based stream ciphers.

Secondly, for an arbitrary feedback polynomial, a standard approach is to search for low weight multiples of the feedback polynomial and then to apply the Meier-Staffelbach approach to fast correlation attacks using a low weight multiple. We can do the same and search for multiples of the feedback polynomial that have the above form. It turns out that this approach is in general less efficient than searching for low weight polynomial, but we can always find specific instances of feedback polynomials where we do get an improvement.

The remaining parts of the paper are presented as follows. In Section 2 we give the basic preliminaries for the attack. In Section 3 we discuss how a basic distinguishing attack is mounted and in Section 4 we expand this attack by using vectors with noise variables. In Section 5 we give the consequences when tweaking the different parameters of the attack. Section 6 discusses the problem of finding a multiple of the characteristic polynomial. In Section 7 we compare our attack to the basic attack and in Section 8 we give our conclusions.

2 Preliminaries

The model used for the attack is the standard model for a correlation attack, illustrated in Figure 1. For a more detailed description of this model we refer to, for example, [17]. The target stream cipher uses two different components, one

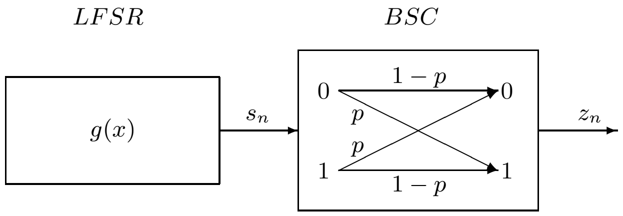

Fig. 1. Model for a correlation attack.

linear and one nonlinear. The linear part is a LFSR and the nonlinear part can be modeled as a black box. One often illustrates the correlation attack scenario through the class of nonlinear combining generators, although it is applicable to the more general case described above. See for example [18].

It is common to view the problem of cryptanalysis as a decoding problem. The nonlinear function can then, through a linear approximation, be seen as a binary symmetric channel (BSC) with crossover probability p (correlation probability $1-p$) with $p \neq 0.5$.

The output bits from the LFSR are denoted s_n, $n = 1, 2, \ldots$, and the keystream bits are denoted z_n, $n = 1, 2, \ldots$. From the BSC it follows immediately that $P(s_n = z_n) = 1-p \neq 0.5$. Assuming a known plaintext attack, the problem of ours is the following. Given the observed keystream sequence $\mathbf{z} = z_1, z_2 \ldots z_n$ we want to distinguish the keystream from a truly random process. If P_0 is the distribution induced by the cipher and P_1 is the uniform distribution, we try to determine if the underlying distribution D for the samples that we observe (z_n, $n = 1, 2, \ldots$) is more likely to be P_0 or P_1.

Example: We will here give an example in which such distinguishing attacks can be useful. Assume that the people in a small country are going to vote in a referendum. Alice is going to send her vote as a ciphertext $\mathbf{c} = \mathbf{m} + \mathbf{z}$, where $\mathbf{m}$ is the vote and $\mathbf{z}$ is the keystream. Now assume that there are two possible ways to vote, either $\mathbf{m} = \mathbf{m}^{(1)}$ or $\mathbf{m} = \mathbf{m}^{(2)}$, and that both of them include some large amount of data (e.g they are both pictures).

The attacker Eve, who can listen to the channel, receives the ciphertext $\mathbf{c}$. He makes the guess that Alice voted $\mathbf{m}^{(1)}$. Eve then adds $\mathbf{m}^{(1)}$ to the received ciphertext and depending on whether he made the right or wrong guess he gets

$$\hat{\mathbf{z}} = \mathbf{c} + \mathbf{m}^{(1)} = \begin{cases} \mathbf{z} & \text{correct guess} \\ \mathbf{m}^{(1)} + \mathbf{m}^{(2)} + \mathbf{z} & \text{wrong guess} \end{cases}.$$

If the guess was incorrect the result is a random sequence, assuming that $\mathbf{m}^{(1)} + \mathbf{m}^{(2)}$ is random. Hence if we apply a distinguisher to the vector $\hat{\mathbf{z}}$, Eve can determine how Alice voted.

2.1 Hypothesis testing

In this section we give a brief introduction to binary hypothesis testing. The task of a binary hypothesis test is to decide which of two hypotheses H_0 or H_1 is the explanation for an observed measurement. Statistics provides methods to determine how many output symbols that are needed to make a correct decision and also how to carry out the actual hypothesis test. These two parts will be explained in the following.

Assume that we have a sequence of m independent and identically distributed (i.i.d.) random variables $X_1, X_2, \ldots, X_m$ over an alphabet $\mathcal{X}$. Its distribution is denoted $D(x) = Pr(X_i = x)$, $1 \leq i \leq m$ and the sample values obtained in an experiment are denoted $\mathbf{x} = x_1, x_2, \ldots, x_m$. We have the two hypotheses $H_0 : D = P_0$ and $H_1 : D = P_1$, where P_0 and P_1 are two different distributions. To distinguish between the two hypotheses, one defines a *decision function*, $\phi : \mathcal{X}^m \to \{0,1\}$. $\phi(\mathbf{x}) = 0$ implies that H_0 is accepted and $\phi(\mathbf{x}) = 1$ implies that H_1 is accepted.

Two probabilities of error are associated with the decision function,

$$\begin{aligned} \alpha &= P(\phi(\mathbf{x}) = 1 | H_0 \textit{ is true}) \\ \beta &= P(\phi(\mathbf{x}) = 0 | H_1 \textit{ is true}). \end{aligned} \tag{1}$$

Let H_0 be the hypothesis that the distribution D is induced by the cipher and let H_1 be the hypothesis that D is uniform.

The overall probability of error, P_e, can be written as a weighted sum over α and β, i.e., $P_e = \pi_0 \alpha + \pi_1 \beta$, where π_0 and π_1 are the *a priori* probabilities of the two hypotheses. An important asymptotic result is the following,

$$P_e \approx 2^{-mC(P_0,P_1)}, \tag{2}$$

when m is large. The variable $C(P_0, P_1)$ is the *Chernoff information* between distributions P_0 and P_1. The Chernoff information is obtained through

$$C(P_0, P_1) = - \min_{0 \leq \lambda \leq 1} \log_2 \left(\sum_{x \in \mathcal{X}} (P_0(x))^{\lambda} (P_1(x))^{1-\lambda} \right). \tag{3}$$

It can be difficult to determine the exact value of λ but by picking just any value, e.g. $\lambda = 0.5$, it is possible to obtain a lower bound of the *Chernoff information* and hence, an upper bound of P_e. By using (2), the number of samples needed to distinguish P_0 and P_1 can be calculated for any error probability.

We also need to know how to perform the hypothesis test. The Neyman-Pearson lemma tells us how to carry out the actual test when we have a sequence of samples.

Lemma 1. *(Neyman-Pearson lemma) Let $X_1, X_2, \ldots, X_m$ be drawn i.i.d. according to mass function D. Consider the decision problem corresponding to the hypotheses $D = P_0$ vs. $D = P_1$. For $T \geq 0$ define a region*

$$A_m(T) = \left\{ \frac{P_0(x_1, x_2, \ldots, x_m)}{P_1(x_1, x_2, \ldots, x_m)} > T \right\} .$$

Let $\alpha_m = P_0^m(\mathcal{A}_m^c(T))$ and $\beta_m = P_1^m(\mathcal{A}_m(T))$ be the error probabilities corresponding to the decision region $\mathcal{A}_m$. Let $\mathcal{B}_m$ be any other decision region with associated error probabilities α^* and β^*. If $\alpha^* \leq \alpha$, then $\beta^* \geq \beta$.

This tells us that the region $\mathcal{A}_m(T)$ that is determined by $\frac{P_0(\mathbf{x})}{P_1(\mathbf{x})} > T$, is the one that jointly minimizes α and β. In the hypothesis test we want α and β to be equal and hence $T = 1$. With the assumption that all x_n are independent we can rewrite the Newman-Pearson test using 2-logarithms. This gives us the test

$$\frac{P_0(x_1, x_2, \ldots, x_m)}{P_1(x_1, x_2, \ldots, x_m)} > 1 \quad \Rightarrow \quad \sum_{n=1}^{m} \left(\log_2 \frac{P_0(x_n)}{P_1(x_n)} \right) > 0 \; . \tag{4}$$

The ratio in (4) is called a log-likelihood ratio, and the test is thus called a log-likelihood test.

This was a very brief overview of some tools that will be useful for us. For a more thorough treatment of hypothesis testing, we refer to any textbook on the subject, e.g. [7].

3 A basic distinguishing attack from a low weight feedback polynomial

We start our investigation by simplifying the Meier-Staffelbach approach and turn their original ideas into a distinguishing attack.

Referring to the assumed model (Figure 1), the observed keystream output is considered as a noisy version of the sequence from the LFSR,

$$z_n = s_n + e_n, \tag{5}$$

where e_n, $n = 1, 2, \ldots$, are variables representing the noise introduced by the approximation. The noise has a biased distribution

$$P(e_n = 0) = p = 1/2 + \varepsilon,$$

where ε is usually rather small. The recursive computation of s_n is linear, and for a LFSR the computation of s_n will depend on the characteristic polynomial of the LFSR. The recurrence can be written as $s_n = \sum_{j=1}^{L} c_j s_{n-j}$ where L is the LFSR length and c_j, $j = 1, 2, \ldots$, are some known constants. By introducing $c_0 = 1$ the recurrence above can be put into the form

$$\sum_{j=0}^{L} c_j s_{n-j} = 0, \quad n \geq j.$$

By adding the corresponding positions in $\mathbf{z}$ we can get all s_n canceling out. What remains is just a sum of independent noise variables. In more detail, let us introduce

$$x_n = \sum_{j=0}^{L} c_j z_{n-j}.$$

Then $x_n = \sum_{j=0}^{L} c_j z_{n-j} = \sum_{j=0}^{L} c_j s_{n-j} + \sum_{j=0}^{L} c_j e_{n-j} = \sum_{j=0}^{L} c_j e_{n-j}$. Since the distribution of e_n is nonuniform it is possible to distinguish the sample sequence x_n, $n = 1, 2, \ldots$, from a truly random sequence. If we assume the binary case (all variables are binary), the sum of the noise will have a bias which according to the piling-up lemma [16] can be expressed as follows,

$$P(\sum_{j=0}^{L} c_j e_{n-j} = 0) = 1/2 + 2^{w-1}\varepsilon^w, \tag{6}$$

where w is the weight of $(c_0, c_1, \ldots, c_L)$. We also know that the required number of samples is in the order of $1/(2^{w-1}\varepsilon^w)^2$.

In this paper we choose to use, however, the Chernoff information as a measure of the distance between two distributions, as described in Section 2.1. If we consider the binary case above, the expression for the Chernoff information in equation (3) becomes

$$C(P_0, P_1) \geq -\log_2 \left(\sqrt{\left(\frac{1}{2} + 2^{w-1}\varepsilon^w\right)\frac{1}{2}} + \sqrt{\left(\frac{1}{2} - 2^{w-1}\varepsilon^w\right)\frac{1}{2}} \right).$$

In Table 1 in appendix we give the number of samples needed for some different w values. In the table we use $\varepsilon = 0.1$ and $\varepsilon = 0.01$.

Note that the ideas behind this simple attack has appeared in many attack scenarios before, even if it might not have been described exactly in this context before. We see that the weight of the characteristic polynomial is directly connected to the success of the attack.

4 A more general distinguisher with correlated vectors

As we have seen in the previous section, low weight polynomials are easily attacked, but when the weight grows so does the required length and complexity (exponentially). At some point we argue that the attack is no longer realistic, or it might require more than an exhaustive key search. In this section we now describe a similar but more general approach that can be applied to another set of characteristic polynomials.

Consider a length L LFSR with characteristic polynomial

$$f(x) = f_0 + f_1 x + f_2 x^2 + f_3 x^3 + \ldots f_L x^L,$$

where $f_i \in \mathbb{F}_2$. We try to find a multiple, $a(x)$, of the characteristic polynomial so that this polynomial can be written as

$$a(x) = f(x)h(x) = g_1(x) + x^M g_2(x), \tag{7}$$

where $g_1(x)$ and $g_2(x)$ are polynomials of some small degree $\leq k$. It is possible that $f(x)$ is already on the form (7), then $h(x) = 1$. In the sequel we assume that such a polynomial $a(x)$ of the above form is given.

This will correspond to a shift register for which the taps are concentrated to two regions far away from each other. The linear recurrence relation can then be written as the two sums

$$\sum_{i=0}^{k} s_{n+i}a_i + \sum_{i=0}^{k} s_{n+M+i}a_{M+i} = 0, \tag{8}$$

where s_n is the nth output bit from the LFSR and $a_i,\ i = 0, 1, \ldots$, are the coefficients in the characteristic polynomial $a(x)$. We now consider the standard model for a correlation attack where the output of the cipher is considered as a noisy version of the LFSR sequence $z_n = s_n + e_n$. The noise variables e_n is introduced by the approximation of the nonlinear part of the cipher. Furthermore, the biased noise has distribution $P(e_n = 0) = 1/2 + \varepsilon$, and the variables are pairwise independent, i.e., $P(e_i, e_j) = P(e_i)P(e_j),\ \forall i \neq j$.

Let us introduce the notation Q_n to be the sum

$$Q_n = \sum_{i=0}^{k} z_{n+i}a_i + \sum_{i=0}^{k} z_{n+M+i}a_{M+i} = \sum_{i=0}^{k} e_{n+i}a_i + \sum_{i=0}^{k} e_{n+M+i}a_{M+i}. \tag{9}$$

This can also be written as

$$\begin{aligned}
Q_0 &= e[0,k] \cdot \underline{g_1} + e[M, M+k] \cdot \underline{g_2}, \\
Q_1 &= e[1,k+1] \cdot \underline{g_1} + e[M+1, M+k+1] \cdot \underline{g_2}, \\
&\vdots \\
Q_{N-1} &= e[N-1, N+k-1] \cdot \underline{g_1} + e[M+N-1, M+N+k-1] \cdot \underline{g_2},
\end{aligned}$$

if we introduce $e[i,j] = (e_i, \ldots, e_j)$ for $i \leq j$ and $\underline{g_1} = (g_{1,0}, g_{1,1}, \ldots, g_{1,k})^T$ where $g_{1,j},\ j = 0, 1, \ldots, k$ are the coefficients of the $g_1(x)$ polynomial. A corresponding notation is assumed for $\underline{g_2}$.

The noise variables $(e_n,\ n = 1, 2, \ldots)$ are independent but Q_i values that are close to each other will not be independent in general. This is because of the fact that several Q_i will contain common noise variables. We can take advantage of this fact by moving to a vector representing the noise as follows.

Introduce the vectorial noise vector E_n of length N as

$$E_n = (Q_{N \cdot n}, \ldots, Q_{N(n+1)-1}). \tag{10}$$

Alternatively, E_n can be expressed as

$$E_n = (e_{N\cdot n}, \ldots, e_{N(n+1)+k-1}) \cdot G_1 + (e_{N\cdot n+M}, \ldots, e_{N(n+1)+M+k-1}) \cdot G_2, \quad (11)$$

where G_1 and G_2 are the $(N+k) \times N$ matrices

$$G_1 = \begin{pmatrix} g_{1,0} & & \\ g_{1,1} & g_{1,0} & \\ \vdots & \vdots & g_{1,0} \\ g_{1,k} & g_{1,k-1} \cdots & g_{1,1} \\ & g_{1,k} & \vdots \\ & & g_{1,k} \end{pmatrix} \quad \text{and} \quad G_2 = \begin{pmatrix} g_{2,0} & & \\ g_{2,1} & g_{2,0} & \\ \vdots & \vdots & g_{2,0} \\ g_{2,k} & g_{2,k-1} \cdots & g_{2,1} \\ & g_{2,k} & \vdots \\ & & g_{2,k} \end{pmatrix}.$$

The remaining pieces to complete the distinguishing attack for the assumed polynomial are obvious. To prepare the attack, we derive the distribution of the E_n noise vector given in (11). This can be easily done for small polynomial degrees, since we know the distribution of the noice. This calculation results in a distribution which we denote by P_0. Our aim is to distinguish this distribution from the truly random case, so by P_1 we denote the uniform distribution.

When performing the attack, we collect a sample sequence

$$Q_0, Q_1, Q_2, \ldots, Q_B$$

by $Q_n = \sum_{i=0}^{k} z_{n+i} a_i + \sum_{i=0}^{k} z_{n+M+i} a_{M+i}$. This sample sequence is then transformed into a vectorial sample sequence $E_0, E_1, \ldots, E_{B'}$ by $E_n = (Q_{N\cdot n}, \ldots, Q_{N(n+1)-1})$.

The final step is to use optimal (Neyman-Pearson) hypothesis testing to decide whether $E_0, E_1, \ldots, E_{B'}$ is most likely to come from distribution P_0 or P_1. This proposed algorithm is summarized in Figure 2.

1. Find multiple.
2. For $t = 0 \ldots B$
   $Q_t = \sum_{i=0}^{k} z_{t+i} a_i + \sum_{i=0}^{k} z_{t+M+i} a_{M+i}$
   end for.
3. For $n = 0 \ldots B'$
   $E_n = (Q_{N\cdot n}, \ldots, Q_{N(n+1)-1})$
   end for.
4. Calculate $I = \sum_{n=0}^{B'} \left(\log_2 \frac{P_0(E_n)}{P_1(E_n)} \right)$.
If $I > 0$ then output **cipher**, else output **random**.

Fig. 2. Summary of proposed algorithm.

The performance of the algorithm depends on the polynomials, g_i, that are used. Figure 4 shows an example of how the number of vectors required for a successful attack depends on the vector length for a certain combination of two polynomials. $N = 1$ corresponds to the basic approach and we see that increasing the vector length will decrease the number of vectors needed. Note that $g_1(x)$ and $g_2(x)$ are just two examples of what the polynomials might look like, they do not represent a multiple of any specific primitive polynomial.

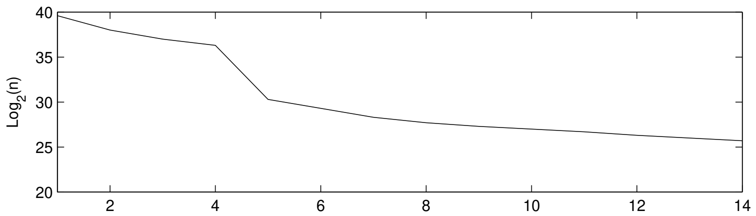

Fig. 3. The number of vectors needed as a function of the vector length N. In this example $g_1(x) = 1 + x + x^5 + x^6$ and $g_2(x) = 1 + x + x^7 + x^8$.

Finally, we note that we may generalize our reasoning with two groups to allow finding a multiple of arbitrarily many groups. Expression (7) for the multiple then becomes

$$a(x) = h(x)f(x) = g_1(x) + x^{M_1} g_2(x) + \ldots + x^{M_{t-1}} g_t(x), \qquad (12)$$

where $g_i(x)$ is a polynomial of some small degree $\leq k$, and $M_1 < M_2 < \ldots < M_{t-1}$. It is clear that when t grows it is easier to find multiples of the characteristic polynomial with the desired properties. But it is also clear that when t grows, the attack becomes weaker.

This distinguishing attack that we propose may be mounted on ciphers using a shift register where "good" multiples easily can be found. Ciphers using a polynomial where many taps are close together might be attacked directly without finding any multiple. This attack may be viewed as a new design criteria, one should avoid LFSRs where multiples of the form (7) are easily found.

5 Tweaking the parameters in the attack

We have described the new attack in the previous section. We now discuss how various algorithmic parameters effect the results.

5.1 How $g_i(x)$ effects the results

Since the vectors E_n are correlated it is intuitive that the forms of $g_i(x)$ will effect the strength of the attack. Some combinations of polynomials will turn out to be much better than others.

We have tested a number of different polynomials $g_i(x)$ in order to find a set of rules describing how the form of the polynomials effects the result. A definite rule to decide which polynomials are best suited for the attack is hard to find. Some basic properties that characterizes a good polynomial can be found. The parameters that determine how a polynomial will effect the distribution of the noise vectors are the following.

- *The weight of the polynomial.* A small weight means that the we have a small number of noise variables and hence, a large bias. And a large weight means many noise variables and therefore a more uniform noise distribution.
- *Arrangement of the terms in a polynomial.* This is the same as the arrangement of the taps in the LFSR. If there are many taps close together, the corresponding noise variables will occur more frequently in the noise vector. This will significantly effect the distribution of the noise vectors.

Since M_i is typically large, all polynomials $g_i(x)$ can be considered independent. Their properties will have the same influence on the total result. However, the polynomials are combined to form the distribution. This combination is just a variant of the piling-up lemma so it is obvious that the total distribution is more uniform than the distribution of the individual polynomials. Depending on the form of the individual polynomials, the resulting distribution becomes more or less uniform. Some combinations are "better" than others. An example of this can be found in Figure 4 in the appendix.

5.2 Vector length

The length of the vectors, denoted N impacts the effectiveness of the algorithm. The idea of using N larger than one is that we can get correlation between the vectors. Every time we increase N by 1 we will also increase the Chernoff information. Recalling (2), we see that increasing the Chernoff information means that we will decrease the number of vectors needed for our hypothesis test. The Chernoff information is however not a linear function of N. Depending on the form of the polynomials the increase can be much higher for some N than for others. The computational complexity is also higher when N gets higher since each vector have 2^N different values. The downside is that the complexity of the calculations increase with increasing N. So there is trade off between the number of samples needed and the complexity of the calculations. The largest gain in Chernoff information is usually achieved when going from $N = i$ to $N = i + 1$ for small i. This phenomenon can be seen in Figure 5 in the appendix.

5.3 Increase the number of groups

As we will show in Section 6.2, it is much easier to find a multiple if we allow the multiple to have more groups than just two. The drawback is that the distribution will become more uniform if more groups are used. This is similar to the binary case in which more taps in the multiple will cause a less biased distribution. The difference is that there is no equivalence to equation (6) for how much more uniform the distribution will be when having many groups, since this depends a lot on the polynomials used.

6 Finding multiples of the characteristic polynomial of a desired form

We have many times assumed that a multiple of the feedback polynomial has a certain form. Here we briefly look at the problem of finding multiples of a certain form.

6.1 Finding low weight multiples

According to the piling up lemma (6), the distribution becomes more uniform if the polynomial is of a high weight. Therefore, the first step in a correlation attack is to find a multiple that has a low weight. The multiple will produce the same sequence so the linear relation that describes the multiple will also satisfy the original LFSR sequence. There exist easy and efficient ways of finding a multiple of a given weight. The number of bits needed to actually start the attack depends on the degree of the multiple, which in turn depends the weight of the same. If we want to find a multiple of weight w of a polynomial that has degree L it can be shown [12] that the degree M of the multiple will be approximately:

$$M \geq 2^{\frac{L}{w-1}}. \tag{13}$$

In Table 2 in the appendix the result of this equation is listed for a LFSR of length 100 and a LFSR of length 1000.

6.2 Finding multiples with groups

Say that we want to find a multiple of the form

$$a(x) = h(x)f(x) = g_1(x) + x^M g_2(x),$$

where $f(x)$ is the characteristic polynomial of the cipher. The degree of $f(x)$ is L. This can be found by polynomial division. Assume that we have a $g_2(x)$ of degree smaller than k. We then multiply this polynomial with x^i. The result is divided by the original LFSR-polynomial. This gives us a quotient $q(x)$ and a remainder $r(x)$.

$$x^i g_2(x) = f(x) \cdot q(x) + r(x).$$

We have 2^k different $g_2(x)$-polynomials and i can be chosen in M different ways. The remainder $r(x)$ from the division is a polynomial with $0 \leq \deg(r(x)) < L$. If $r(x)$ has degree $\leq k$ we have found an acceptable $g_1(x)$. The probability of finding a polynomial of maximum degree k is $P(\deg(g_2(x)) \leq k) = 2^{k-L}$. If we would like it to be probable that we find at least one such polynomial, we need

$$2^k \cdot M \cdot \frac{2^k}{2^L} \geq 1 \quad \Rightarrow \quad M \geq 2^{L-2k}. \tag{14}$$

Examining the result in (14) we see that for modest values of k the length of the multiple will become quite large. Therefore we extend our reasoning to the case with arbitrarily many groups, then we have a multiple of the form

$$a(x) = h(x)f(x) = g_1(x) + x^{M_1}g_2(x) + \ldots + x^{M_{t-1}}g_t(x).$$

If we use the same reasoning as above we receive a new expression

$$2^k \cdot M_1 \cdot 2^k \cdot M_2 \cdot \ldots \cdot 2^k \cdot M_{t-1} \cdot \frac{2^k}{2^L} \geq 1 \quad \Rightarrow \quad M \geq 2^{\frac{L-tk}{t-1}}, \tag{15}$$

where it is assumed that $M_1, M_2, \ldots, M_{t-1} \leq M$. This gives us an upper bound on all M_i.

In (15) we see that by using larger values of t, i.e., more groups, we can lower the length of the multiple. One has to bear in mind though that a larger t, as stated in Section 5.3, will usually effect the Chernoff information in a negative way (from a cryptanalyst point of view). In Table 3 in the appendix we list some values on M needed to find a multiple, for some values of k and t.

7 Comparing the proposed attack with a basic distinguishing attack

The applicability of our algorithm is twofold. Firstly, if the characteristic polynomial is of the form $f(x) = g_1(x) + g_2(x)x^{M_1} + \ldots + g_t(x)x^{M_{t-1}}$. Applying the basic algorithm to those LFSRs without finding any multiple first will be equivalent to applying our algorithm with $N = 1$. Since our algorithm has the ability to have vectors with noise variables ($N > 1$), it will be a significant improvement over the basic algorithm. Using the basic algorithm without first finding a multiple is naive, but if the length L of the LFSR is large the degree of the low weight multiple will also be large, see (13). So if $f(x)$ is of high degree then our algorithm can be more effective.

Our algorithm can also be applied to arbitrary characteristic polynomials. Then the approach is to first find a multiple of the polynomial that is of the form $f(x) = g_1(x) + g_2(x)x^{M_1} + \ldots + g_t(x)x^{M_{t-1}}$ and then apply the algorithm. By comparing the two equations (13) and (15) we see that it is not much harder to find a polynomial of some weight w than it is to find a polynomial with the same number of groups. Tables showing the corresponding M can be found in the appendix. Although our algorithm takes advantage of the fact that the taps are close together, it is still not enough to compensate for the larger amount of noise variables. In this case the proposed attack will give improvements only for certain specific instances of characteristic polynomials, e.g., those having a surprisingly weak multiple of the form $f(x) = g_1(x) + g_2(x)x^{M_1}$ but no low weight multiples where the weight is surprisingly low.

8 Conclusion and future work

Through a new correlation attack, we have identified a new class of weak feedback polynomials, namely, polynomials of the form $f(x) = g_1(x) + g_2(x)x^{M_1} + \ldots +$

$g_t(x)x^{M_{t-1}}$, where $g_1, g_2, \ldots, g_t$ are all polynomials of low degree. The correlation attack has been described in the form of a distinguishing attack. This was done mainly for simplicity, since the theoretical performance is easily calculated and we can compare with the basic attack based on low weight polynomials.

The next step in this direction would be to examine the possibility of turning these ideas into a key recovery attack. This could be done in a similar manner as the Meier Staffelbach approach. For example, we could try to derive many different relations (multiples) and apply some iterative decoding approach in vector form. The theoretical part of such an approach will probably be much more complicated.

References

1. A. Canteaut, M. Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. *Advances in Cryptology-Eurocrypt'2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 573-588. Springer-Verlag, 2000.
2. V. Chepyzhov, T. Johansson, B. Smeets. A simple algorithm for fast correlation attacks on stream ciphers. *Advances in Cryptology-FSE'2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 181-195. Springer-Verlag, 2001.
3. D. Coppersmith, S. Halevi and C.S. Jutla. SCREAM: a software efficient stream cipher. In J. Daemen and V. Rijmen, editors, *Advances in Cryptology-FSE'2002*, volume 2365 of *Lecture Notes in Computer Science*, pages 195-210. Springer-Verlag, 2002.
4. D. Coppersmith, S. Halevi and C.S. Jutla. Cryptanalysis of stream ciphers with linear masking. In M. Young, editor. *Advances in Cryptology-Crypto'2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 515-532. Springer-Verlag, 2002.
5. N. Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. *Advances in Cryptology-Asiacrypt'2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267-287. Springer-Verlag, 2002.
6. N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. *Advances in Cryptology-Eurocrypt'2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345-359. Springer-Verlag, 2003.
7. T. Cover and J.A. Thomas. *Elements of information theory.* Wiley series in telecommunication. Wiley, 1991.
8. P. Ekdahl and T. Johansson. A new version of the stream cipher SNOW. In K. Nyberg and H. Heys, editors. *Selected Areas in Cryptography-SAC 2002*, volume 2595 of *Lecture Notes in Computer Science*, pages 47-61. Springer-Verlag, 2003.
9. P. Ekdahl and T. Johansson. Distinguishing attack on SOBER-t16 and SOBER-t32. In J. Daemen and V. Rijmen, editors. *Advances in Cryptology-FSE'2002*, volume 2365 of *Lecture Notes in Computer Science*, pages 210-224. Springer-Verlag, 2002.
10. N. Ferguson, D. Whiting, B. Schneider, J. Kelsey, S. Lucks and T. Kohno. Helix: Fast Encryption and Authentication in a Single Cryptographic Primitive. In T. Johansson, editor, *Advances in Cryptology-FSE'2003*, volume 2887 of *Lecture Notes in Computer Science*, pages 330-346. Springer-Verlag, 2003.
11. J.D. Golić. Intrinsic statistical weakness of keystream generators. *Advances in Cryptology-Asiacrypt'94*, volume 917 of *Lecture Notes in Computer Science*, pages 91-103. Springer-Verlag, 1995.

12. J.D. Golić. Computation of low-weight parity-check polynomials. *Electronic Letters*, 32(21):1981-1982, October 1996.
13. P. Hawkes and G. Rose. Primitive specification and supporting documentation for SOBER-t32 submission to NESSIE. In *Proceedings of First Open NESSIE Workshop, 2000.*
14. T. Johansson, F. Jönsson. Improved fast correlation attacks on stream ciphers via convolutional codes. *Advances in Cryptology-Eurocrypt'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 347-362. Springer-Verlag, 1999.
15. T. Johansson, F. Jönsson. Fast correlation attacks based on turbo code techniques. *Advances in Cryptology-Crypto'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 181-197. Springer-Verlag, 1999.
16. M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology-Eurocrypt'93*, volume 765 of *Lecture Notes in Computer Science*, pages 386-397. Springer-Verlag, 1994.
17. W. Meier and O. Staffelbach. Fast correlation attacks on stream ciphers.*Advances in Cryptology-Eurocrypt'88*,volume 330 of *Journal of Cryptology*,pages 310-314. Springer-Verlag, 1988.
18. A. Menezes, P. van Oorschot and S. Vanstone. *Handbook of Applied cryptography*, CRC Press, 1997.
19. M. Mihaljevic, M. Fossorier and H. Imai. A low-complexity and high-performance algorithm for the fast correlation attack. *Advances in Cryptology-FSE'2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 196-212. Springer-Verlag, 2001.
20. D. Watanabe, S. Furuya, H. Yoshida, K. Takaragi and B. Preneel. A new keystream generator MUGI. In J. Daemen and V. Rijmen, editors, *Fast Software Encryption 2002*, volume 2365 of *Lecture Notes in Computer Science*, pages 179-194. Springer-Verlag, 2002.

A Tables and figures

The appendix contains some tables and figures that can be used to compare some different parameters discussed in the paper. Table 1 shows the number of samples needed to distinguish a sequence from random in the basic binary case. The result is given for two different ε. Table 2 shows the expected degree of the multiple of $f(x)$ in the basic distinguishing attack, where w is the weight of the polynomial to be found and L is the length of the LFSR. Table 3 shows the corresponding values for our attack, where t denotes the number of groups and k the maximum number of taps allowed in each polynomial. The degree of the multiple is the same as the amount of plaintext needed before the actual attack can start.

M	w													
	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\varepsilon = 0.1$	$2^{16.4}$	2^{21}	$2^{25.7}$	$2^{30.3}$	2^{35}	$2^{39.6}$	$2^{44.3}$	$2^{48.9}$	$2^{53.6}$	$2^{58.2}$	$2^{62.8}$	$2^{67.5}$	$2^{72.1}$	$2^{76.8}$
$\varepsilon = 0.01$	$2^{36.3}$	$2^{47.6}$	$2^{58.9}$	$2^{70.2}$	$2^{81.5}$	$2^{92.8}$	2^{104}	2^{115}	2^{127}	2^{138}	2^{149}	2^{160}	2^{172}	2^{183}

Table 1. Number of samples needed for some different w in the binary case for $\varepsilon = 0.1$ and $\varepsilon = 0.01$. The number of vectors needed is calculated as $\frac{1}{C(P_0,P_1)}$.

M	w														
	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$L = 100$	2^{100}	2^{50}	2^{33}	2^{25}	2^{20}	2^{17}	2^{14}	2^{123}	2^{11}	2^{10}	2^{9}	2^{8}	2^{8}	2^{7}	2^{7}
$L = 1000$	2^{1000}	2^{500}	2^{333}	2^{250}	2^{200}	2^{167}	2^{143}	2^{125}	2^{111}	2^{100}	2^{91}	2^{83}	2^{77}	2^{71}	2^{67}

Table 2. The degree M of the multiple as a function of L and w.

M		t				
		2	3	4	5	6
	3	2^{94}	2^{47}	2^{31}	2^{24}	2^{19}
	4	2^{92}	2^{46}	2^{31}	2^{23}	2^{18}
	5	2^{90}	2^{45}	2^{30}	2^{23}	2^{18}
k	6	2^{88}	2^{44}	2^{29}	2^{22}	2^{18}
	7	2^{86}	2^{43}	2^{29}	2^{22}	2^{17}
	8	2^{84}	2^{42}	2^{28}	2^{21}	2^{17}

Table 3. The degree M of the multiple as a function of k and t for $L = 100$.

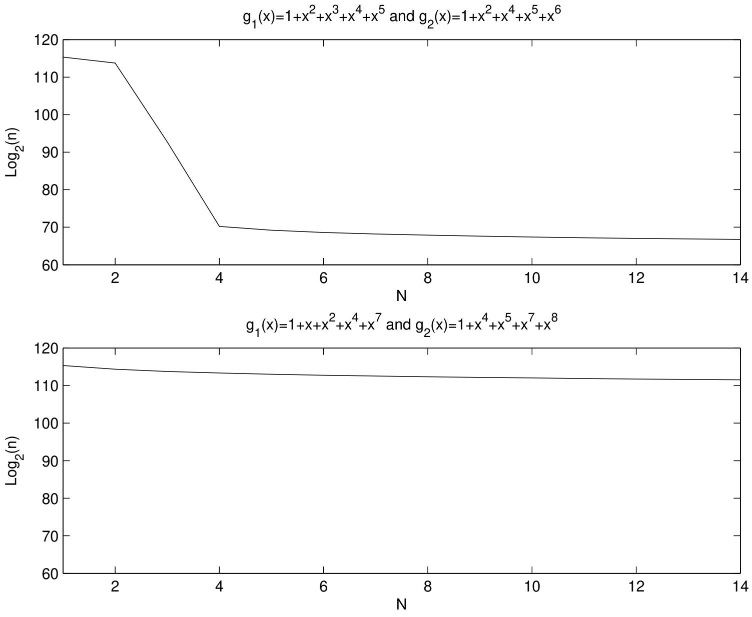

Fig. 4. The sequence length needed as a function of the vector length N (logarithmic).

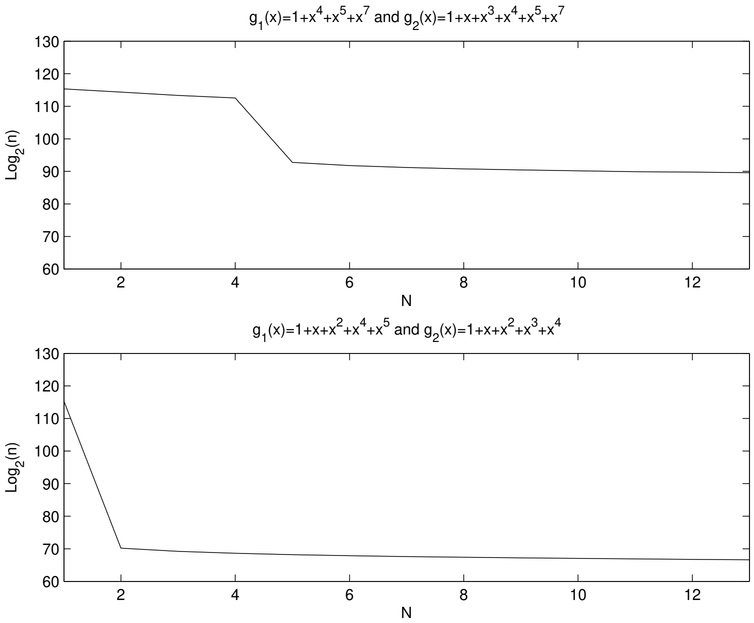

Fig. 5. The sequence length needed as a function of the vector length N (logarithmic).