On the (Im)Possibility of Practical and Secure Nonlinear Filters and Combiners*

An Braeken and Joseph Lano

Katholieke Universiteit Leuven, Dept. Elect. Eng.-ESAT/SCD-COSIC,
Kasteelpark Arenberg 10, 3001 Heverlee, Belgium
{an.braeken, joseph.lano}@esat.kuleuven.be

Abstract. A vast amount of literature on stream ciphers is directed to the cryptanalysis of LFSR-based filters and combiners, resulting in various cryptanalytic attacks. In this paper, we present a unified framework for the security of a design against these attacks based on the properties of the LFSR(s) and the Boolean function used. It is explained why building nonlinear filters seems more practical than building nonlinear combiners. We also investigate concrete building blocks that offer a good trade-off in their resistance against these various attacks, and can at the same time be used to build a low-cost synchronous stream cipher for hardware applications.

Keywords: Combination and filter generator, distinguishing attack, correlation attack, algebraic attack, hardware complexity.

1 Introduction

For efficient encryption of data, cryptography mainly uses two types of symmetric algorithms, block ciphers and stream ciphers. In the past decades, block ciphers have become the most widely used technology. However, as block ciphers are often used in a stream cipher mode such as CTR and OFB, stream ciphers may offer equivalent security at a lower cost.

Designing a secure stream cipher appears to be a hard task. In the NESSIE competition, flaws have been found in all candidates. In fact, unexpected biases are often detected in designs, especially if the design is based on relatively new concepts or uses large vectorial Boolean functions of which it is impossible to calculate all biases and correlations beforehand.

By far the most studied designs are the nonlinear filter and combiner generators, which are based on LFSRs in conjunction with a Boolean function. So far, new developments were mostly restricted to the development of separate attacks which were then applied to a design that is particularly vulnerable to the

* This work was supported in part by the Concerted Research Action (GOA) Ambiorics 2005/11 of the Flemish Government and by the European Commission through the IST Programme under Contract IST2002507932 ECRYPT. An Braeken is a F.W.O. Research Assistant, sponsored by the Fund for Scientific Research - Flanders (Belgium), Joseph Lano is financed by a Ph.D. grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

B. Preneel and S. Tavares (Eds.): SAC 2005, LNCS 3897, pp. 159–174, 2006.

new attack. Little attention has been given to the cryptographic design requirements that result from these attacks combined. The recent ECRYPT eSTREAM project motivates us to investigate this further.

Although at first sight the vast amount of research on nonlinear filters and combiners seems to put these designs at a disadvantage compared to newer design principles, we believe that they can also benefit from the accumulated knowledge of their underlying mathematics. In fact, if one can develop a stream cipher that is resistant to all these attacks combined and is still easily implementable in hardware, confidence in this classical and transparent design can be higher (of course thorough public evaluation will still be necessary) and hence actual application can be faster.

In this paper, we study the cryptographic design requirements for filters and combiners. In Sect. 3, we study the impact of the most important attacks (distinguishing attacks, correlation attacks and algebraic attacks) on the building blocks. By analyzing building blocks that offer optimal resistance against certain attacks, we establish minimal requirements for the internal state size, the LFSR polynomials and the properties of the Boolean function (number of inputs, Walsh transform, degree, nonlinearity, algebraic immunity, ...). This analysis allows to establish design criteria for filters and combiners respectively. We then study, in Sect. 4, some Boolean functions such as power functions and symmetric functions, which can be interesting design choices as they have some desirable properties and are easy to implement in hardware.

Not all our results could be presented in this extended abstract. For an extended version of this paper containing more explanations, tables, and proofs, please refer to [5].

2 Preliminaries

The two basic models of key stream generators are the combination and the filter generator. A *combination generator* uses several LFSRs in parallel and combines their outputs in a nonlinear way (by means of the combining function). If the output is computed by a nonlinear function (filter function) of some taps of one LFSR, a *filter generator* is obtained. In this section, we list some properties of the two building blocks that are used in these generators, namely LFSRs and Boolean functions.

2.1 Linear Feedback Shift Registers

Definition 1. *A* Linear Feedback Shift Register (LFSR) *of length L is a collection of L 1-bit memory elements $s_t^0, s_t^1, \ldots, s_t^{L-1}$. At each time t the memory is updated as follows:*

$$\begin{cases} s_t^i = s_{t-1}^{i+1} \text{ for } i = 0, \ldots, L-2 \\ s_t^{L-1} = \bigoplus_{i=1}^{L} c_i \cdot s_{t-1}^{L-i}. \end{cases} \tag{1}$$

where the c_i are fixed binary coefficients that define the feedback equation of the LFSR. The LFSR stream $(s_t)_{t \geq 0}$ consists of the successive values in the memory element s_0.

Associated with an L-bit LFSR is its *feedback polynomial P(X)* of degree d, $P(X) = 1 + \sum_{i=1}^{L} c_i \cdot X^i$. The *weight* of the feedback polynomial is equal to its number of nonzero terms. In practical designs, a feedback polynomial is chosen to be *primitive*. This implies that every nonzero initial state produces an output sequence with maximal period $2^L - 1$, which is also called a *pn*-sequence.

For many cryptanalytic attacks, it is useful to search *low-weight multiples* of the feedback polynomial $P(x)$, see [6, 19]. The number $m(D, w)$ of multiples $Q(X) = 1 + \sum_{i=1}^{D} c_i \cdot X^i$ of the polynomial $P(X)$, with degree less than or equal to D and with weight w, can be approximated by [6]:

$$m(D, w) \approx \frac{D^{w-1}}{(w-1)! \cdot 2^L} \,. \tag{2}$$

It is interesting to know from which D_{min} we can expect a first multiple $Q(X)$ of weight w to start appearing. It follows from (2) that:

$$D_{min}(w) \approx (2^L \cdot (w-1)!)^{\frac{1}{w-1}} \,. \tag{3}$$

The most efficient approach, known to date, to search for these low-weight multiples is a birthday-like approach, see [19]. The *precomputation complexity* P needed to find all multiples $Q(X)$ of weight w and degree at most D can be approximated by:

$$P(D, w) \approx \frac{D^{\lceil \frac{w-1}{2} \rceil}}{\lceil \frac{w-1}{2} \rceil !} \,. \tag{4}$$

2.2 Boolean Functions

A *Boolean function* f is a mapping from $\mathbb{F}_2^\varphi$ into $\mathbb{F}_2$. The *support* of f is defined as $\sup(f) = \{\overline{x} \in \mathbb{F}_2^\varphi : f(\overline{x}) = 1\}$. The cardinality of $\sup(f)$ represents the *weight* $\mathrm{wt}(f)$ of the function.

A Boolean function can be uniquely represented by means of its *algebraic normal form (ANF)*:

$$f(\overline{x}) = f(x_0, \ldots x_{\varphi-1}) = \bigoplus_{(a_0,\ldots,a_{\varphi-1}) \in \mathbb{F}_2^\varphi} h(a_0, \ldots, a_{\varphi-1})\, x_0^{a_0} \ldots x_{\varphi-1}^{a_{\varphi-1}} \,, \tag{5}$$

where f and h are Boolean functions on $\mathbb{F}_2^\varphi$. The *algebraic degree* of f, denoted by $\deg(f)$, is defined as the highest number of variables in the terms $x_0^{a_0} \ldots x_{\varphi-1}^{a_{\varphi-1}}$ in the ANF of f.

Alternatively, a Boolean function can be represented by its *Walsh spectrum*:

$$W_f(\overline{\omega}) = \sum_{\overline{x} \in \mathbb{F}_2^\varphi} (-1)^{f(\overline{x}) \oplus \overline{x} \cdot \overline{\omega}} = 2^{\varphi-1} - 2wt(f \oplus \overline{x} \cdot \overline{\omega}) \,, \tag{6}$$

where $\overline{x} \cdot \overline{\omega} = x_0\omega_0 \oplus x_1\omega_1 \oplus \cdots \oplus x_{\varphi-1}\omega_{\varphi-1}$ is the *dot product* of $\overline{x}$ and $\overline{\omega}$. We will use the following two well-known formulae for the Walsh values:

$$\begin{cases} \sum_{\overline{\omega} \in \mathbb{F}_2^\varphi} W_f(\overline{\omega}) = \pm 2^\varphi \\ \sum_{\overline{\omega} \in \mathbb{F}_2^\varphi} W_f^2(\overline{\omega}) = 2^{2 \cdot \varphi} \,, \end{cases} \tag{7}$$

where the second equality is known as Parseval's theorem.

Several properties are of importance for Boolean functions from a cryptographic viewpoint. A function is said to be *balanced* if $\mathrm{wt}(f) = 2^{\varphi-1}$ and thus $W_f(\overline{0}) = 0$. The *nonlinearity* N_f of the function f is defined as the minimum distance between f and any affine function; it can be calculated as $N_f = 2^{\varphi-1} - \frac{1}{2}\max_{\overline{\omega}\in\mathbb{F}_2^n} |W_f(\overline{\omega})|$. The *best affine approximation* $l(\overline{x})$ is associated with this notion. We will say that f has *bias* ϵ if it has the same output as its best affine approximation with probability $0.5 + \epsilon$. It is easy to see that $\epsilon = N_f/2^{\varphi} - 0.5 = \frac{\max_{\overline{\omega}\in\mathbb{F}_2^n} |W_f(\overline{\omega})|}{2^{\varphi+1}}$. A function f is said to be *correlation-immune* [34] of order ρ, $CI(\rho)$, if and only if its Walsh transform W_f satisfies $W_f(\overline{\omega}) = 0$, for $1 \leq wt(\overline{\omega}) \leq \rho$. If the function is also balanced, then the function is called ρ-*resilient*. Two important bounds hold for the bias ϵ:

$$\begin{cases} \epsilon \geq 2^{-\varphi/2-1} \\ \epsilon \geq 2^{\rho+1-\varphi}, \end{cases} \tag{8}$$

where the first bound is due to Parseval's theorem and equality holds only for *bent* functions; the second bound reflects the trade-off between resiliency and nonlinearity.

The lowest degree of the function g from $\mathbb{F}_2^{\varphi}$ into $\mathbb{F}_2$ for which $f \cdot g = \overline{0}$ or $(f \oplus \overline{1}) \cdot g = \overline{0}$ is called the *algebraic immunity (AI)* of the function f [24]. The function g is said to be an *annihilator* of f if $f \cdot g = \overline{0}$. It has been shown [11] that any function f with φ inputs has algebraic immunity at most $\lceil \frac{\varphi}{2} \rceil$.

A *vectorial Boolean function* F from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$, also called (n, m) S-box, can be represented by an m-tuple $(f_1, \ldots, f_m)$ of Boolean functions f_i on $\mathbb{F}_2^n$ (corresponding to the output bits).

3 Security Analysis

During the last two decades, several classes of attacks have been proposed on the filter and combination generator. In this section, we will thoroughly investigate these different attacks and will derive minimal requirements that the LFSRs and Boolean functions should satisfy. Our goal is to investigate whether it is possible to construct practical and secure filter or combination generators with 80-bit key security and low hardware cost, which implies that we should keep it close to the edge of the minimal requirements while at the same time keeping a reasonable security margin.

For most attacks, our analysis reflects the currently known attacks described in the literature, but we now relate these attacks directly to the mathematical properties of the concrete building blocks used. Our treatment of distinguishing attacks combines and extends the ideas of the recent work done in [25, 17] to concrete distinguishing attacks on all filter and combination generators. It follows that distinguishing attacks are very similar to correlation attacks but are often stronger, as they can use many linear approximations simultaneously and do not need a decoding algorithm. Note also that resynchronization mechanisms are not discussed in this paper. A secure resynchronization mechanism of the scheme is also necessary. We refer to [12, 2] for more details concerning resynchronization attacks.

3.1 Tradeoff Attacks

Time-Memory-Data Tradeoff attacks [3] are generic attacks against stream ciphers. To prevent these attacks, the internal state should be at least twice the key size. Consequently, with an 80-bit key, the LFSR has at least a length of 160 bits. In the following, we will investigate the security of filter and combination generator with an internal state of 256 bits, and thus taking a sufficient security margin. This allows us to quantify our analysis, but of course it is easy to adapt the framework to other security parameters.

3.2 Berlekamp-Massey Attacks

The linear complexity of a bit stream $(s_t)_{t\geq 0}$ is equal to the length of the shortest LFSR generating that stream. For a Boolean function of degree d, the linear complexity LC of the resulting key stream generated by a filter generator is upper bounded by $\sum_{i=0}^{d} \binom{L}{i}$. Moreover, it is very likely that the LC of the key stream is lower bounded by $\binom{L}{d}$ and that its period remains equal to 2^L-1. If the constituent LFSRs of the combination generator have distinct degrees greater than 2 and initial state different from 0, then the LC of the key stream generated by a combination is equal to $f(L_1,\ldots,L_n)$, where the ANF of f is evaluated over the integers. We refer to [33, 22] for more details.

The Berlekamp-Massey attack requires $2\cdot LC$ data and has complexity of LC^2. For a key stream generator with internal size equal to 256 and a Boolean function of sufficiently high degree, this attack is clearly of no concern. A degree-7 function will be sufficient for a nonlinear filter. For combiners the calculation is more complex as it depends on the size of the LFSRs and on the ANF of the Boolean functions, but also here we start having resistance against the Berlekamp-Massey attack from degree 7.

3.3 Distinguishing Attacks

The distinguishing attack we describe here is based on the framework developed in [17] combined with the mathematical results from [25]. We extend the framework for the filter generator and also develop the attack for the combiner generator.

Filter Generator. The idea of the attack is the following. The LFSR stream has very good statistical properties, but of course there are linear relations, which are determined by the feedback polynomial $P(X)$ and its multiples $Q(X)$, where the cryptanalyst first needs to find the latter in a precomputation step.

Given a linear relation $Q(X)$ of weight w in the LFSR stream, the same relation for the corresponding bits of the key stream will hold with some probability different from one half, because the Boolean function f does not destroy all linear properties. This probability is determined by the combined action of all affine approximations of the Boolean function. This interaction is governed by the piling-up lemma and can be expressed as [25]:

$$\varepsilon' = \frac{\sum_{\overline{\omega}=0}^{2^{\varphi}-1} (W_f(\overline{\omega}))^w}{2^{\varphi\cdot w+1}}\,. \tag{9}$$

To distinguish the key stream from random, the number of samples needed is in the order of $\frac{1}{\varepsilon'^2}$, which is also the time complexity of the attack. The data complexity is the sum of the degree of the relation $Q(X)$ and the number of samples needed. It is possible to decrease the data complexity to some extent by using multiple relations simultaneously.

We now study the impact of this attack on the bent functions, as these offer the best resistance against this distinguishing attack. We assume our LFSR has an internal state of 256 bits and investigate the data complexity of the attack as a function of the number of inputs. By combining (7), (8) and (9), we can calculate that the bias for bent functions can be written as:

$$\varepsilon' = 2^{-(\lceil w/2 \rceil - 1)\cdot\varphi - 1}. \tag{10}$$

A cryptanalyst is interested in finding the weight w for which the attack complexity is minimal. For very low weight w, the degree of $Q(X)$ will be prohibitively high as shown by (3). For very high weight w, the bias (10) will be too small. We hence expect an optimal tradeoff somewhere in between.

We have calculated these numbers, and it turns out that no 256-bit LFSR with a Boolean function with less than 20 inputs can be made secure against this distinguishing attack! However, we have to make two important observations:

- The precomputation time to find the necessary multiples $Q(X)$ is very high (in the order of 2^{150}, governed by (3) and (4). A discussion on the amount of precomputation we can allow is an important topic of discussion. Note that this also applies to other attacks such as trade-off attacks, correlation attacks and algebraic attacks.
- There has been some debate on the relevance of distinguishing attacks requiring long key streams during NESSIE [32]. Whereas time complexity is only a matter of resources at the attacker's side, available data depends on what has been encrypted by the sending party. Hence, we propose to limit the maximum amount of key stream generated from a single key/iv pair to 2^{40} bits (practical assumption), after which the scheme should resynchronize. This measure prevents the appearance of low-weight multiples: from (3) it follows that normally no multiples of weight less than 8 exist with degree less than 2^{40}. Now, we can recalculate the best attack complexities, by adding the extra constraint $\log_2(D_{min}) < 40$. It follows that, under this practical restriction, nonlinear filters can be built which are secure against distinguishing attacks starting from 14 inputs. Note that the restriction of a single key stream to 2^{40} bits also provides some protection against other cryptanalytic attacks, as explained below.

Combination Generator. For combination generators, the attack can be improved by using a divide and conquer approach. Let us assume we have a combiner with φ LFSRs and that the Boolean function has resiliency ρ. The average length of one LFSR is thus $\frac{256}{\varphi}$. The attacker now mounts the same distinguishing attack, restricting himself to r LFSRs, where r must be of course strictly

greater than the order of resiliency ρ. Again, we first search for a low-weight multiple of this equivalent $\frac{256 \cdot r}{\varphi}$-length LFSR, and then try to detect the bias that remains of this relation after the Boolean function. From the piling-up lemma, it follows that this bias is as follows:

$$\varepsilon' = \frac{\sum_{\overline{\omega} \in S} (W_f(\overline{\omega}))^w}{2^{\varphi \cdot w + 1}}, \tag{11}$$

where the set S is defined as:

$$S = \{\overline{\omega} | wt(\overline{\omega}) > \rho \text{ and } \overline{\omega} < 2^r\}, \tag{12}$$

assuming, without loss of generality, that the attacked LFSRs are numbered $0, 1, \ldots r-1$. This divide and conquer approach gives the attacker some advantages compared to the case of the nonlinear filter:

- If the resiliency of the Boolean function is low, the length of the attacked equivalent LFSR can be low. First, this will allow the attacker to perform the precomputation step governed by (3) and (4) much faster. Second, the length of the low-weight multiples will be much lower, making the data complexities feasible for much lower weights, where the detectable biases will be much larger as shown by (11). Note that when $\frac{256 \cdot r}{\varphi}$ is very small, we will even be able to mount a powerful weight-2 attack without precomputation step: this will just correspond to the period of the small equivalent LFSR. As shown in [17], such an attack can be easily turned into a key recovery attack.
- If the resiliency of the Boolean function is high, we will still have the advantages explained in the above point, but to a lesser extent. But here the tradeoff between resiliency and nonlinearity (8) will come into play, resulting in higher Walsh values in (11) and hence a higher bias.

It follows that it is much harder to make the combiner model resistant to this distinguishing attack. We have again calculated the best resistance that can be achieved, and it turns out that 36 inputs and an optimally chosen Boolean function would be needed to resist the attack.

Again, we propose to limit the maximum amount of key stream obtained from a single key/iv pair to 2^{40} bits. After recalculating the complexities with this restriction, we now see that, under ideal assumptions for the cryptographer, he can make a combiner which is secure starting from 18 inputs.

It is important to note that these lower bounds will be very hard to approach in reality due to the difficulty of finding practical functions that have properties close to the optimal case described here, and due to the fact that our lower bounds consider equal LFSR lengths. In reality all LFSR lengths need to be distinct, which will allow the attack to improve his attack significantly by attacking the shortest LFSRs. The larger the number LFSRs, the more serious this problem becomes. As a result of this, we believe it is not possible to design a practical nonlinear combiner, with 256 bits internal state, that can resist to this distinguishing attack. This in comparison to the case of the filter generator, where we can succeed to get close to the bounds with actually implementable functions, as evidenced by the power functions described in Sect. 4. We will now develop a similar reasoning for the case of fast correlation attacks.

3.4 Fast Correlation Attacks

Correlation and fast correlation attacks exploit the correlation between the LFSR stream s_t and the key stream z_t. These attacks can be seen as a decoding problem, since the key stream z_t can be considered as the transmitted LFSR stream s_t through a binary symmetric channel with error probability $p = P_{t\geq 0}(z_t \neq s_t)$. For the nonlinear filter generator, p is determined by $0.5+\epsilon$. For the combination generator, a divide-and-conquer attack as in the case of a distinguishing attack is possible. Consequently, it is sufficient to restrict the attack to a subset of LFSRs $\{i_0, \ldots, i_\rho\}$, such that the following holds:

$$P(f(\overline{x}) \neq 0 | x_{i_0} = a_0, \ldots, x_{i_\rho} = a_\rho, \forall (a_0, \ldots, a_\rho) \in \mathbb{F}_2^{\rho+1}) \neq 1/2\,. \tag{13}$$

Here, as defined above, the parameter ρ corresponds with the order of resiliency of the Boolean function.

Then, the attack consists of a fast decoding method for any LFSR code C of length N (the amount of available key stream) and dimension L (the length of the LFSR), where the length N of the code is lower bounded by Shannon's channel coding theorem:

$$N \geq \frac{L}{C(p)} = \frac{L}{1 + p\log_2 p + (1-p)\log_2(1-p)} \approx \frac{\ln(2)L}{2\epsilon^2}\,. \tag{14}$$

If we want to achieve perfect security against this attack for a 256-bit LFSR and allowing at most 2^{40} bits in a single key stream, the bias ϵ should be less than or equal to 2^{-17}. To achieve this, we would need a highly nonlinear Boolean functions with more than 34 inputs. This can never be implemented efficiently in hardware. However, the above criterion is far too stringent as actual decoding algorithms are not able to do this decoding with a good time complexity. We now look at the current complexities of these decoding algorithms. Besides the maximum-likelihood (ML) decoding, which has very high complexity of $L \cdot 2^L$, mainly two different approaches have been proposed in the literature. In the first approach, the existence of sparse parity check equations for the LFSR code are exploited. These parity check equations correspond with the low weight multiples of the connection polynomial. In this way, the LFSR code can be seen as a low-density parity-check (LDPC) code and has several efficient iterative decoding algorithms. In the second approach, a smaller linear $[n, l]$ code with $l < L$ and $n > N$ is associated to the LFSR on which ML decoding is performed. The complexity of both approaches highly depends on the existence of low degree multiples. As shown above, the precomputation time for finding these low degree multiples is very high. Moreover, the approach requires long key streams and suffers from a high decoding complexity.

Note the very strong resemblance between this classical framework for the correlation attacks and the framework we developed in the previous subsection for distinguishing attacks. The main difference between the two is that in the correlation attacks we need a decoding method, whereas in the distinguishing attacks we just need to apply a simple distinguisher. Analysis of the above formulae learns that the complexity of the decoding procedure is much higher than

for the distinguishing procedure. Even with huge improvements of the existing decoding procedures, we do not expect this situation to change. Hence we can conclude that a choice of parameters that makes the design resistant against distinguishing attacks (explained above), will also make it resistant against correlation attacks.

3.5 Algebraic Attacks

In *algebraic attacks* [11], a system of nonlinear equations between input and output is constructed and subsequently solved. The complexity of solving this system of equations highly depends on the degree of these equations. In the usual algebraic attack, equations between one bit of the output of the filter or combination generator and the initial state of the LFSR are searched. These equations are then solved by linearization. The lowest possible degree d of these equations, also called the *Algebraic Immunity (AI)*, is obtained by the annihilators of the filter or combination function and its complement. The total complexity $C(L, d)$ of the algebraic attack on a stream cipher with a linear state of L bits and equations of degree d is then determined by

$$C(L,d) = \left(\sum_{i=0}^{d} \binom{L}{i}\right)^{\omega} = D^{\omega}\,, \tag{15}$$

where ω corresponds to the coefficient of the most efficient solution method for the linear system. We use here Strassen's exponent [35] which is $\omega = \log_2(7) \approx 2.807$. Clearly, the number of required key stream bits is equal to D. Note that in the complexity analysis, the linearization method is used for solving the equations. It is an open question if other algorithms like the Buchberger algorithm, F4 or F5 [18] can significantly improve this complexity. Also, the total number of terms of degree less than or equal to d is considered in the complexity, while in general nothing is known about the proportion of monomials of degree d that appear in the system of equations. Therefore, a sufficient security margin should be taken into account.

It can be calculated that an AI of 5 is currently sufficient to withstand standard algebraic attacks in our framework, for an LFSR of length 256 and aiming at 80-bit security.

The implications for Boolean functions are the following. It has been shown [11] that the AI of a Boolean function with φ inputs can be at most $\lceil \frac{\varphi}{2} \rceil$. Our Boolean function hence needs to have at least 9 inputs. We will of course need to check that the AI of the Boolean function is large enough. The complexity of the algorithm to check if there are equations of degree less than or equal to d (corresponding to AI equal to d) is slightly better than $\binom{\varphi}{d}^{\omega}$ [24], which is feasible for most practical functions of interest here.

Fast algebraic attacks can be much more efficient than the usual algebraic attacks. In the fast algebraic attacks [9], the attacker tries to decrease the degree d of the system of equations even further by searching for relations between the initial state of the LFSR and several bits of the output function simultaneously.

The equations where the highest degree terms do not involve the key stream bits are considered. This is done in an additional precomputation step (complexity $D \cdot \log^2 D$ [20]): linear combinations of the equations are searched which cancel out the highest degree terms in order to obtain equations of degree e. The complexity for solving these equations is now $C(L, e)$. As shown in [20], we have to take into account another complexity as well, namely that of substituting the key stream into the system of equations. This complexity can be approximated by $2 \cdot E \cdot D \cdot \log D$. The required key stream length is equal to $D + E - 1 \approx D$ because $D >> E$. If we have found C such relations of degree e, we can further reduce the data complexity by a factor C, but at the cost of increasing the substitution complexity by this same factor C because we have to repeat the substitution step C times.

The problem with fast algebraic attacks is that, for the moment, very little is known about whether these reductions are possible or not for some Boolean function. The only way we can be sure is by searching for the existence of these relations, but the complexity of this step may become prohibitively high.

Another approach to achieve resistance against fast algebraic attack, without needing this precomputation, is by choosing a very high AI. In this case either the complexity of the substitution step or the complexity of the solving step will become prohibitively high. It can be calculated that we achieve such security starting from an AI of 12. When restricting the amount of key stream bits to 2^{40}, the AI of the function should be greater than or equal to 9.

It should be stressed that not much is known about the strength and the implementation of fast algebraic attacks, and it is hence probable that the algorithms can be improved. The further investigation of fast algebraic attacks is an interesting research topic, and designers are advised to take a reasonable security margin, as evidenced by [10].

4 Boolean Functions for the Filter Generator

From the analysis in the previous section, it follows that the two most important properties a good Boolean function should have are high algebraic immunity and high nonlinearity. Besides, to be used in practice, they should be implementable in hardware at a very low cost. We will now present two classes of functions that have a low-cost hardware implementation.

4.1 Symmetric Functions

Symmetric functions [7] are functions with the very interesting property that their hardware complexity is linear in the number of variables. A symmetric function is a function of which the output is completely determined by the weight of the input vector. Therefore, the truth table $v_f = (v_0, \ldots, v_\varphi)$, also called *value vector*, of the symmetric function f on $\mathbb{F}_2^\varphi$ reduces to a vector of length $\varphi + 1$, corresponding with the function values v_i of the vectors of weight i with $0 \leq i \leq \varphi$. We have identified the following class of symmetric functions with maximum AI:

Theorem 2. *The symmetric function in $\mathbb{F}_2^\varphi$ with value vector*

$$v_f(i) = \begin{cases} 0 \text{ for } i < \lceil \frac{\varphi}{2} \rceil \\ 1 \text{ else} \end{cases} \tag{16}$$

has maximum AI. Let us denote this function by F_k where k is equal to the threshold $\lceil \frac{\varphi}{2} \rceil$.

By Proposition 2 and Proposition 4 of [7], the degree of these functions are determined as follows.

Theorem 3. *The degree of the symmetric function $F_{\lceil \frac{\varphi}{2} \rceil}$ on $\mathbb{F}_2^\varphi$ is equal to $2^{\lfloor \log_2 \varphi \rfloor}$.*

If φ is odd, these functions are trivially balanced because $v_f(i) = v_f(\varphi - i)$ for $0 \leq i \leq \lfloor \frac{\varphi}{2} \rfloor$. As shown in Proposition 5 of [7], trivially balanced functions satisfy the following properties: The derivative $D_{\overline{1}}f$ with respect to $\overline{1}$ is the constant function, and $W_f(\overline{x}) = 0$ for all $\overline{x}$ with $\mathrm{wt}(\overline{x})$ even. For φ even, the functions are not balanced. But by XORing with an extra input variable from the LFSR, this property is immediately obtained.

Another problem is that the nonlinearity of these functions is not high. In particular, $\max_{\overline{w} \in \mathbb{F}_2^\varphi} |W_f(\overline{w})| = 2\binom{\varphi-1}{\frac{\varphi-1}{2}}$ for odd φ and equal to $\binom{\varphi}{\frac{\varphi}{2}}$ for even φ. Therefore, $\epsilon \approx 2^{-3.15}, 2^{-3.26}, 2^{-3.348}$ for $\varphi = 13, 15, 17$ respectively. Note that the nonlinearity increases very slowly with the number of inputs φ: even for 255 input bits, we only get a bias of $2^{-5.33}$.

Remark 1. The nonlinearity of this class of symmetric functions corresponds to the nonlinearity of the functions with maximum AI that are obtained by means of the construction described in [13]. This construction has the best nonlinearity with respect to other constructions that have a provable lower bound on the AI presented in literature so far. An extensive study on the AI and nonlinearity of symmetric Boolean functions is performed in [4]. It has been shown that there exists for some even dimensions n other classes of symmetric functions with maximum AI which have slightly better nonlinearity, but still far too small to be resistant against the distinguishing attack. Also, no symmetric function with better bias in nonlinearity compared with the AI has been found in [4].

4.2 Power Functions

The idea of using a filter function f derived from a power function P on $\mathbb{F}_2^\varphi$ is as follows: we consider the φ input bits to the function P as a word $\overline{x}$ in $\mathbb{F}_2^\varphi$. We then compute the p-th power, $\overline{y} = \overline{x}^p$, of this word. The output of our Boolean function f is then one bit y_i for $i \in \{0, \ldots, \varphi - 1\}$ of this output word $\overline{y} = (y_0, \ldots, y_{\varphi-1})$. Note that all these functions for $i \in \{0, \ldots, \varphi-1\}$ are linearly equivalent to the trace of the power function P. We now discuss the nonlinearity, algebraic immunity and implementation complexity of some interesting power functions.

Nonlinearity. We will investigate Boolean functions derived from highly nonlinear bijective power functions. These functions have bias $\epsilon = 2^{-\frac{\varphi}{2}}$ for φ even and $2^{-\frac{\varphi}{2}-\frac{1}{2}}$ for φ odd, which is very close to the ideal case, the bent functions. The known classes of such highly nonlinear power functions are the inverse [28], Kasami [21], Dobbertin [14], Niho 1 [16] and Niho 2 [15] classes.

Algebraic Immunity. In [8], the AI of the Boolean functions derived from the highly nonlinear power functions is computed up to dimension less than or equal to 14. These results together with our simulations for higher dimensions indicate that most of the highly nonlinear bijective power functions we study do not achieve the optimal AI, but they do reasonably well on this criterion. For instance, the AI of the Boolean function derived from the inverse power function on $\mathbb{F}_{2^{16}}$ is equal to 6 (where 8 would be the maximum attainable). However, as shown by Courtois [10], fast algebraic attacks can be efficiently applied on this function. In particular, there exist 4 annihilators of degree 6 which reduce to degree 4 and there exist 32 annihilators of degree 7 which reduce to degree 3.

Implementation. We now study implementation complexity of some concrete functions and give the nonlinearity and AI of these practical functions. Efficient implementations of the *inverse function* in the field $\mathbb{F}_{2^i}$ for $i \geq 3$ has been studied by several authors, due to the fact that this function is used in the Advanced Encryption Standard. An efficient approach can be obtained by working in composite fields as described in [30]. Based on recursion, the inverse function is decomposed into operations in the field $\mathbb{F}_{2^2}$. A minimal hardware implementation for $\varphi = 16$ requires 398 XOR gates and 75 AND gates, and thus consists of about 1107.5 NAND gates. It is also possible to increase the clock frequency if necessary by pipelining the design.

Hardware implementation of *exponentiation with general exponents* in $\mathbb{F}_{2^\varphi}$ has been well studied [1]. However, it turns out that all classes of bijective highly nonlinear power functions with degree greater than or equal to 6 have a very regular pattern in their exponent, which can be exploited for a more efficient implementation: All exponents e satisfy the property that the vector $\overline{e} = (e_0, \ldots, e_{\varphi-1})$ defining its binary representation, *i.e.*, $e = \sum_{i=0}^{\varphi-1} e_i 2^i$, contains a regular sequence consisting of ones together with at most one bit which is separated of this sequence. The distance between two consecutive elements in the sequence is equal to one except in the case of the Dobbertin functions for $k > 1$. If the weight of this sequence is equal to a power of 2, than this property can be exploited leading to a more efficient implementation.

We will demonstrate this improved implementation on the power function X^{511} in $\mathbb{F}_2^{16}$. The exponent 511 has binary representation 111111111_2. Consequently, it contains a sequence of weight 9, or also a sequence of weight 8 with one extra digit. First, consider the normal basis $\{\alpha, \alpha^2, \alpha^4, \cdots, \alpha^{2^{\varphi-1}}\}$ of $\mathbb{F}_2^\varphi$ for $\varphi = 16$ (a normal basis exists for every $\varphi \geq 1$, see [29]). Computing the power function in this basis will not change the properties of nonlinearity, degree, AI and Walsh spectrum of the output functions, since power functions in different bases are linearly equivalent. Squaring in this basis represents simply a cyclic

shift of the vector representation of that element. Consequently, computing the power 511 of an element $\overline{x} \in \mathbb{F}_2^{16}$, can be computed as follows:

$$\begin{aligned} \overline{x}^{511} &= (\overline{x} \cdot \overline{x}^2) \cdot (\overline{x}^4 \cdot \overline{x}^8) \cdot (\overline{x}^{16} \cdot \overline{x}^{32}) \cdot (\overline{x}^{64} \cdot \overline{x}^{128}) \cdot \overline{x}^{256} \\ &= (\overline{y} \cdot \overline{y}^4) \cdot (\overline{y}^{16} \cdot \overline{y}^{64}) \cdot \overline{x}^{256} \text{ with } \overline{y} = \overline{x} \cdot \overline{x}^2 \\ &= \overline{z} \cdot \overline{z}^{16} \cdot \overline{x}^{256} \text{ with } \overline{z} = \overline{y} \cdot \overline{y}^4 . \end{aligned} \tag{17}$$

Therefore, we only need to perform some shifts together with 4 multiplications in the normal basis of $\mathbb{F}_2^{16}$. These multiplications correspond with $(\overline{x} \cdot \overline{x}^2)$, $(\overline{y} \cdot \overline{y}^4)$, and $(\overline{z} \cdot \overline{z}^{16} \cdot \overline{x}^{256})$. The hardware complexity of such multiplication depends on the basis used to represent the field elements, or more precisely, on the number of ones C_φ in the multiplication matrix. It is known that $C_\varphi \geq 2\varphi - 1$ with equality if and only if the normal basis is optimal [26]. Note that optimal bases do not exist for any dimension φ. The overall gate count of the multiplication is lower bounded by $\varphi C_\varphi \geq 2\varphi^2 - \varphi$ AND gates and $(\varphi - 1)C_\varphi \geq 2\varphi^2 - 3\varphi + 1$ XOR gates [23]. Other implementations may provide even better complexity. Also several algorithms exist for performing normal basis multiplication in software efficiently [27, 31]. Therefore, the number of NAND gates for a multiplication in normal basis is lower bounded by 1906.5 for $\varphi = 16$, 2161.5 for $\varphi = 17$, and 2719.5 for $\varphi = 19$ respectively. If the vector containing the binary representation of the exponent consists of a regular sequence with weight 2^i, then the number of multiplications is equal to i, or $i + 1$ if there is an additional digit defining the complete exponent.

We can conclude that the power functions described here offer a very good tradeoff between ease of implementation and cryptanalytic strength. Moreover, we believe that the implementation of a complete S-box in the design has several advantages. In the first place, we can increase the throughput of the generator by outputting more bits m instead of outputting 1 bit. Therefore, a careful study on the best bias in the affine approximation and the AI of all linear and nonlinear combinations of m output bits need to be performed. Another possibility, which makes the analysis harder but may increase the security, is to destroy the linearity of the state. We could consider a filter generator with memory by simply feeding some remaining bits from the S-box into a nonlinear memory. Another possibility is to feedback bits of the S-box into the LFSR during key stream generation. In both cases, it seems that the added nonlinearity may allow us to increase the throughput. Finally, resynchronization can be performed faster by using all bits of the S-box to destroy as rapidly as possible the linear relations between the bits.

5 Conclusion

In this paper, we have presented a framework for the security of the classical LFSR-based nonlinear filter and combiner generators. We have related the resistance to the most important cryptanalytic attacks (distinguishing attacks, (fast) correlation attacks and (fast) algebraic attacks) to the mathematical properties of the LFSRs and the Boolean function. From our analysis, we are inclined to

prefer the nonlinear filter generator, with a Boolean function having as most important properties high nonlinearity and high algebraic immunity.

These classical and very transparent designs are the only stream cipher building blocks for which a complete analysis of the linear biases, correlations and nonlinear relations is possible. A design that has been thoroughly analyzed with respect to the presented framework could hence be more trustworthy than a design that is based on a new, little studied design strategy.

We have also presented two classes of Boolean functions, the symmetric functions and the power functions, that should be further analyzed as they possess some desired properties and are at the same time easy to implement in hardware.

Further investigation of such LFSR-based schemes remains a necessity. Notably, the understanding of the existence of lower degree equations in fast algebraic attacks is missing. The aim of this paper is to be a step in the direction of the unification of this interesting research field, in which until now too much attention has been given to ad hoc attacks on some designs and not to the relations between the mathematical properties and the attacks.

References

1. Gordon Agnew, Thomas Beth, Ronald Mullin, and Scott Vanstone. Arithmetic operations in GF(2^m). *Journal of Cryptology*, 6(1):3–13, 1993.
2. Frederik Armknecht, Joseph Lano, and Bart Preneel. Extending the resynchronization attack. In Helena Handschuh and Anwar Hasan, editors, *Selected Areas in Cryptography, SAC 2004*, number 3357 in Lecture Notes in Computer Science, pages 19–38. Springer-Verlag, 2004.
3. Steve Babbage. Space/time trade-off in exhaustive search attacks on stream ciphers. Eurocrypt Rump session, 1996.
4. An Braeken. On the algebraic immunity of symmetric boolean functions. Technical report, K.U. Leuven, 2005.
5. An Braeken and Joseph Lano. On the (im)possibility of practical and secure nonlinear filters and combiners (extended version). COSIC technical report, 2005. `https://www.cosic.esat.kuleuven.be/publications/`.
6. Anne Canteaut and Michael Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In B. Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000*, number 1807 in Lecture Notes in Computer Science, pages 573–588. Springer-Verlag, 2000.
7. Anne Canteaut and Marion Videau. Symmetric Boolean functions. *IEEE Trans. Inform. Theory*, 2004. Regular paper. To appear.
8. Claude Carlet and Philippe Gaborit. On the construction of balanced Boolean functions with a good algebraic immunity. Proceedings of First Workshop on Boolean Functions : Cryptography and Applications, Mars 2005, Rouen, 2005.
9. Nicolas Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, number 2729 in Lecture Notes in Computer Science, pages 176–194. Springer-Verlag, 2003.
10. Nicolas Courtois. Cryptanalysis of sfinks. ECRYPT Stream Cipher Project, 2005.
11. Nicolas Courtois and Willi Meier. Algebraic attacks on stream ciphers with linear feedback. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003*, number 2656 in Lecture Notes in Computer Science, pages 345–359. Springer-Verlag, 2003. extended version on eprint.

12. Joan Daemen, Rene Govaerts, and Joos Vandewalle. Resynchronization weaknesses in synchronous stream ciphers. In T. Helleseth, editor, *Advances in Cryptology - EUROCRYPT 1993*, number 765 in Lecture Notes in Computer Science, pages 159–167. Springer-Verlag, 1993.
13. Deepak Dalai, Kishan Gupta, and Subhamoy Maitra. Cryptographically significant Boolean functions: Construction and analysis in terms of algebraic immunity. In H. Gilbert and H. Handschuh, editors, *Fast Software Encryption, FSE 2005*, Lecture Notes in Computer Science. Springer-Verlag, 2005.
14. Hans Dobbertin. One-to-one highly nonlinear power functions on GF(2^n). *Applicable Algebra in Engineering, Communication, and Computation*, 9:139–152, 1998.
15. Hans Dobbertin. Almost perfect nonlinear power functions on $gf(2^n)$: The Niho case. *Information and Computation*, 151(1-2):57–72, 1999.
16. Hans Dobbertin, Thor Helleseth, Vijay Kumar, and Halvard Martinsen. Ternary m-sequences with three-valued crosscorrelation: New decimations of Welch and Niho type. *IEEE Transactions on Information Theory*, IT-47:1473–1481, November 2001.
17. Hakan Englund and Thomas Johansson. A new simple technique to attack filter generators and related ciphers. In Helena Handschuh and Anwar Hasan, editors, *Selected Areas in Cryptography, SAC 2004*, number 3357 in LNCS, pages 39–53. Springer, 2004.
18. Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *International Symposium on Symbolic and Algebraic Computation — ISSAC 2002*, pages 75–83. ACM Press, 2002.
19. Jovan Golic. Computation of low-weight parity-check polynomials. *Electronics Letters*, 32(21):1981–1982, 1996.
20. Philip Hawkes and Gregory Rose. Rewriting variables: The complexity of fast algebraic attacks on stream ciphers. In Matthew Franklin, editor, *Advances in Cryptology - CRYPTO 2004*, number 3152 in Lecture Notes in Computer Science, pages 390–406. Springer-Verlag, 2004.
21. Tadao Kasami. The weight enumerators for several classes of subcodes of the second order binary Reed-Muller codes. *Information and Control*, 18:369–394, 1971.
22. Edwin Key. An analysis of the structure and complexity of nonlinear binary sequence generators. *IEEE Transactions on Information Theory*, 22:732–736, 1976.
23. James Massey and Jimmy Omura. Computational method and apparatus for finite field arithmetic. *US Patent No. 4, 587,627*, 1986.
24. Willi Meier, Enes Pasalic, and Claude Carlet. Algebraic attacks and decomposition of boolean functions. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, number 3027 in Lecture Notes in Computer Science, pages 474–491. Springer-Verlag, 2004.
25. Havard Molland and Thor Helleseth. An improved correlation attack against irregular clocked and filtered keystream generators. In Matthew Franklin, editor, *Advances in Cryptology - CRYPTO 2004*, number 3152 in Lecture Notes in Computer Science, pages 373–389. Springer-Verlag, 2004.
26. Ronald Mullin, I. Onyszchuk, and Scott Vanstone. Optimal normal bases in GF(p^n). *Discrete Applied Mathematics*, 22:149–161, 1989.
27. Peng Ning and Yiqun Lisa Yin. Efficient software implementation for finite field multiplication in normal basis. In Sihan Qing, Tatsuaki Okamoto, and Jianying Zhou, editors, *Third International Conference on Information and Communications Security ICICS 2001*, number 2229 in Lecture Notes in Computer Science, pages 177–188. Springer-Verlag, 2001.

28. Kaisa Nyberg. Differentially uniform mappings for cryptography. In T. Helleseth, editor, *Eurocrypt 1993*, volume 950 of *Lecture Notes in Computer Science*, pages 55–64. Springer-Verlag, 1993.
29. Oystein Ore. On a special class of polynomials. *Trans. Amer. Math.Soc.*, 35:559–584, 1933.
30. Christopher Paar. *Efficient VLSI Architectures for Bit-Parallel Computation in Galois Fields*. Doctoral dissertation, Institute for Experimental Mathematics, University of Essen, Germany, 1994.
31. Arash Reyhani-Masoleh and Anwar Hasan. Fast normal basis multiplication using general purpose processors. *IEEE Transaction on Computers*, 52(3):1379–1390, 2003.
32. Greg Rose and Philip Hawkes. On the applicability of distinguishing attacks against stream ciphers. In *Proceedings of the 3rd NESSIE Workshop*, page 6, 2002.
33. Rainer Rueppel. Stream ciphers. In G. Simmons, editor, *Contemporary Cryptology. The Science of Information Integrity*, pages 65–134. IEEE Press, 1991.
34. Thomas Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE Transactions on Information Theory*, IT-30(5):776–780, 1984.
35. Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13:354–356, 1969.