Algebraic Attacks on Stream Ciphers with Linear Feedback

Nicolas T. Courtois[1] and Willi Meier[2]

[1] Cryptography Research, Schlumberger Smart Cards, 36-38 rue de la Princesse, BP 45, F-78430 Louveciennes Cedex, France, courtois@minrank.org
[2] FH Aargau, CH-5210 Windisch, Switzerland, meierw@fh-argau.ch

Abstract. A classical construction of stream ciphers is to combine several LFSRs and a highly non-linear Boolean function f. Their security is usually analysed in terms of correlation attacks, that can be seen as solving a system of multivariate linear equations, true with some probability. At ICISC'02 this approach is extended to systems of higher-degree multivariate equations, and gives an attack in 2^{92} for Toyocrypt, a Cryptrec submission. In this attack the key is found by solving an overdefined system of algebraic equations. In this paper we show how to substantially lower the degree of these equations by multiplying them by well-chosen multivariate polynomials. Thus we are able to break Toyocrypt in 2^{49} CPU clocks, with only 20 Kbytes of keystream, the fastest attack proposed so far. We also successfully attack the Nessie submission LILI-128, within 2^{57} CPU clocks (not the fastest attack known). In general, we show that if the Boolean function uses only a small subset (e.g. 10) of state/LFSR bits, the cipher can be broken, whatever is the Boolean function used (worst case). Our new general algebraic attack breaks stream ciphers satisfying all the previously known design criteria in at most the square root of the complexity of the previously known generic attack.

Key Words: Algebraic attacks on stream ciphers, pseudo-random generators, nonlinear filtering, Boolean functions, factoring multivariate polynomials, multivariate equations, overdefined problems, XL algorithm, ciphertext-only attacks, Toyocrypt, Cryptrec, LILI-128, Nessie.

1 Introduction

In this paper we study stream ciphers with linear feedback and a nonlinear combiner that produces the output, given the state of the linear part. The security of such stream ciphers received much attention. In [13], Golic gives a set of criteria that should be satisfied in order to resist to the known attacks on stream ciphers. For example, a stream cipher should resist to the fast correlation attack [15], the conditional correlation attack [1] and the inversion attack [13]. In order to resist different types of correlation attacks, many authors focused on proposing Boolean functions that will have no good linear approximation and that will be correlation immune with regard to a subset of several input bits, see for example [5]. Recently the scope of application of the correlation attacks have been extended. In [10], the author exploits rather correlation properties with regard to a non-linear low degree multivariate function that uses all of the variables, or in other words, non-linear low degree approximations. This kind of correlations is not new, see for example in [14]. However their application to cryptographic attacks did not receive sufficient attention, probably because only recently people became aware of the existence of efficient algorithms for solving systems of nonlinear multivariate equations of low degree [21, 8–10].

Following [10], stream ciphers with linear feedback are potentially very vulnerable to such algebraic attacks. If for one state we are able, by some method, to deduce from the output, a multivariate equation of low degree in the state bits, then it is also of low degree in the initial state bits. Then the same can (probably) be done for many other states, and given many keystream bits, we inevitably obtain a very overdefined system of equations (i.e. many equations). Such systems can be solved efficiently by techniques such as XL [21, 8], adapted for this purpose in [10] or the simple linearization [21].

In [10], the equations of low degree are obtained by approximating the non-linear component f of the cipher by a function of low degree. If the probability that the approximation holds is close to 1, then it can be used simultaneously for many equations, and we obtain efficient attacks with XL method. For example in [10], an attack in 2^{92} against Toyocrypt[3] is proposed, that requires only some 2^{19} bits of the keystream. With more keystream, and if at least some 32 bits are consecutive, a better attack is possible, due to Mihaljevic and Imai [18].

In this paper we show that algebraic attacks on stream ciphers will apply even if there is no good low degree approximation. We propose a new method of generating low degree equations, basically by multiplying the initial equations by well-chosen multivariate polynomials. This method allows to cryptanalyse a large class of stream ciphers, satisfying all the previously known design criteria. For example, all very traditional designs using only a small subset of the state bits, are shown to be insecure, whatever is the Boolean function used.

The paper is organized as follows: in Section 2 we give a general view of algebraic attacks on stream ciphers. The main component of our new attack on stream ciphers is described in Section 2.3. In Section 3 we overview Toyocrypt and previously known attacks, then in Section 3.1 we apply our new attack for Toyocrypt. In Sections 4 and 5 we will study LILI-128 and apply our attack. Then in Section 6 we develop our general attack on stream ciphers using a small subset of state bits. Finally we present our conclusions about the design of stream ciphers.

[3] Toyocrypt has been accepted to the second evaluation phase of the Japanese Cryptrec call for primitives, and (apparently) rejected later.

2 Algebraic Attacks Against Stream Ciphers

In this part we overview and substantially extend the general strategy given in [10], that reduces an attack on a stream cipher, to solving a system of multivariate equations.

2.1 The Stream Ciphers that May be Attacked

We consider only synchronous stream ciphers, in which each state is generated from the previous state independently of the plaintext, see for example [17] for precise definitions. In principle also, we consider only regularly clocked stream ciphers, and also (it makes no difference) stream ciphers that are clocked in a known way. However this condition can sometimes be relaxed, cf. attacks on LILI-128 described in Sections 4-5.

For simplicity we restrict to binary stream ciphers in which the state and keystream are composed of a sequence of bits and that generate one bit at a time. We also restrict to the case when the "connection function" that computes the next state is linear over $GF(2)$. We call L this "connection function", and assume that L is public, and only the state is secret. We also assume that the function f that computes the output bit from the state is public and does not depend on the secret key of the cipher. This function f is called "a nonlinear filter". The ciphers described here include the very popular filter generator, in which the state of a single LFSR[4] is transformed by a Boolean function, and also not less popular nonlinear function generators, in which outputs of several LFSRs are combined by a Boolean function [17].

The problem of cryptanalysis of such a stream cipher can be described as follows. Let $(k_0, \ldots, k_{n-1})$ be the initial state, then the output of the cipher (i.e. the keystream) is given by:

$$\begin{cases} b_0 = f \quad\ (k_0, \ldots, k_{n-1}) \\ b_1 = f\left(L\ (k_0, \ldots, k_{n-1})\right) \\ b_2 = f\left(L^2(k_0, \ldots, k_{n-1})\right) \\ \quad\quad \vdots \end{cases}$$

Our problem is to recover the key $k = (k_0, \ldots, k_{n-1})$ from some subset of keystream bits b_i. We are going to perform a partially known plaintext attack, i.e. we know some bits of the plaintext, and the corresponding ciphertext bits. The bits don't need to be consecutive. For example if the plaintext is written with latin alphabet, and does not use too many special characters, it is very likely that all the characters have their most significant bit equal to 0. This will be enough for us, if the text is sufficiently long. This would be (almost) a ciphertext-only attack.

In our attacks we just assume that we have some m bits of the keystream b_i at some known positions.

[4] A Linear Feedback Shift Register, see e.g. [17]. It is also possible to use a MLFSR, equivalent in theory [18] but having faster diffusion, as used in Toyocrypt cipher that we study later.

2.2 The Summary of the Attack

For easier reading we give here a brief summary of the attack developed later. At the time t, the current keystream bit gives an equation $f(s) = b_t$ with s being the current state. The main new idea consists of multiplying $f(s)$, that is usually of high degree, by a well chosen multivariate polynomial $g(s)$, such that fg is of substantially lower degree, denoted by d. Then, for example if $b_t = 0$, we get an equation of low degree $f(s)g(s) = 0$. This in turn, gives a multivariate equation of low degree d on the initial state bits k_i. If we get one such equation for each of sufficiently many keystream bits, we obtain a very overdefined system of multivariate equations that can be solved efficiently.

2.3 Design Criteria on *f* and Known Attacks

Let f be the Boolean function that is used to combine the outputs of the linear part of the cipher. For example the inputs to the function are some bits of the state of some LFSRs. The usual requirements on such functions can be summarised as follows. First, f should be balanced and have high algebraic degree. To prevent correlation attacks, f should be highly non-linear[5], and correlation immune at high order, see [5].

An additional criterion on f is given in [10]: f should also not have a very good correlation with respect to low-degree non-linear multivariate functions, otherwise efficient attacks are possible, for example for Toyocrypt [10]. They are possible when:

S1 either the Boolean function f has a low algebraic degree D (classical criterion),

S2 or f can be approximated by such a function with a probability close to 1 (new criterion [10]). This probability is usually denoted $1-\varepsilon$ for some small ε.

The practical attacks on Toyocrypt presented in [10] use the second case S2. In this paper we use equations true with probability 1 (as in the first case) except that we relax the degree condition: it is no longer necessary that f has a low algebraic degree d:

2.4 New Assumptions on *f* and New Attack Scenarios

S3 The multivariate polynomial f has some multiple fg of low degree d, with g being some non-zero multivariate polynomial.

S4 It is also possible to imagine attacks in which f has some multiple fg, such that fg can be approximated by a function of low degree with some probability $(1-\varepsilon)$.

[5] But maybe not a perfectly non-linear (bent) function, see Section 4 in [10].

How to Use Low Degree Multiples

In scenarios S1 and S2, for each known keystream bit at position t: b_t, we obtain a concrete value of $b_t = f(s)$, and thus we get the equation $b_t = f\left(L^t(k_0, \ldots, k_{n-1})\right)$. For this, f has to be of low degree. In scenarios S3 and S4, for each known keystream bit $b_t = f(s)$ at position t, we get:

$$f(s) \cdot g(s) = b_t \cdot g(s),$$

and, since the state is at time t is $s = L^t(k_0, \ldots, k_{n-1})$, it boils down to:

$$f\left(L^t(k_0, \ldots, k_{n-1})\right) \cdot g\left(L^t(k_0, \ldots, k_{n-1})\right) = b_t \cdot g\left(L^t(k_0, \ldots, k_{n-1})\right).$$

We get one multivariate equation for each keystream bit. This equation may be of very low degree, without f being of low degree, and without f having an approximation of low degree. In the basic version of this attack S3, we also require that g is of low degree. There are other possibilities that are studied in the extended version of this paper in which three basic versions of the attack S3a, S3b and S3c are studied.

The important question is now, given a cipher, whether such polynomials g exist, and how to find them. For this, see Section 5 and 6.

Remark 1: It can be seen that the scenarios S1 (respectively S2) are a special case of the scenarios S3 (respectively S4). Indeed, if we put $g = 1$, the equation used in scenarios S3/S4 becomes the usual equation $f(s) = b_t$ of the previous scenarios S1/S2.

2.5 Solving Overdefined Systems of Multivariate Equations

In our attack, given m keystream bits, let R be the number of multivariate equations of degree d, and with n variables k_i, that we obtain. With one equation we have $R = m$, but we may also combine several different g for the same f, and get, for example $R = 14 \cdot m$. We solve them as follows.

Linearization Method: There are about $T \approx \binom{n}{d}$ monomials of degree $\leq d$ in the n variables k_i (assuming $d \leq n/2$). We consider each of these monomials as a new variable V_j. Given $R \geq \binom{n}{d}$ equations, we get a system of $R \geq T$ linear equations with $T = \binom{n}{d}$ variables V_i that can be easily solved by Gaussian elimination on a linear system of size T.

XL Method: When as many as the required $m = \mathcal{O}(\binom{n}{d})$ keystream bits, are **not** available, it is still possible to use XL algorithm or Gröbner bases algorithms to solve the system, with less keystream bits, but with more computations, see [10] or the extended version of this paper.

About the Complexity of Gaussian Reduction Let ω be the exponent of the Gaussian reduction. In theory it is at most $\omega \leq 2.376$, see [6]. However the (neglected) constant factor in this algorithm is expected to be very big. The fastest practical algorithm we are aware of, is Strassen's algorithm [25] that requires about $7 \cdot T^{log_2 7}$ operations. Since our basic operations are over $GF(2)$, we expect that a careful bitslice implementation of this algorithm on a modern

CPU can handle 64 such operations in one single CPU clock. To summarize, we evaluate the complexity of the Gaussian reduction to be $7 \cdot T^{log_2 7}/64$ CPU clocks.

3 Cryptanalysis of Toyocrypt

We look at the stream cipher called Toyocrypt, a submission to the Japanese government Cryptrec call for cryptographic primitives. It was, at the time of the design, believed to resist to all known attacks on stream ciphers. In Toyocrypt, we have one 128-bit LFSR, and thus $n = 128$. The Boolean function is of the form:

$$f(s_0, .., s_{127}) = s_{127} + \sum_{i=0}^{62} s_i s_{\alpha_i} + s_{10}s_{23}s_{32}s_{42} +$$
$$+ s_1 s_2 s_9 s_{12} s_{18} s_{20} s_{23} s_{25} s_{26} s_{28} s_{33} s_{38} s_{41} s_{42} s_{51} s_{53} s_{59} + \prod_{i=0}^{62} s_i.$$

with $\{\alpha_0, \ldots, \alpha_{62}\}$ being some permutation of the set $\{63, \ldots, 125\}$. This system is quite vulnerable to an attack using low-order approximations: there is only one monomial of degree 17, and one of degree 63. The higher-order monomials are almost always zero. From this in [10] an attack following the scenario S2 is described. In this attack, f is approximated by a multivariate function of degree 4 with probability $1 - 2^{-17}$. The attack runs in 2^{92} CPU clocks. For more details see [10].

3.1 New Algebraic Attack on Toyocrypt

In our attack we need to find g such that fg is of low degree (or is such with high probability), following the assumption S3 (or S4) from Section 2.3. How do we find a function g such that fg is of low degree? One method we present in this paper, is by factoring multivariate polynomials. We consider the terms of high degree in $f(s)$ (regardless the lower degree terms) and look if they are divisible by a common low degree factor $g'(s)$. Then (for polynomials over $GF(2)$) we observe that $f(s) \cdot g(s)$ with $g(s) = g'(s) - 1$ is of low degree. Later in Sections 5 and 6, we will describe a different method to find polynomials g satisfying our requirements.

In the case of Toyocrypt, we observe that the combination of the parts of degree 4, 17 and 63, is divisible by a common factor $s_{23}s_{42}$. For each keystream bit b_t, we start from the equation $f(s) = b_t$, and multiply both sides of it by $g(s) = (s_{23} - 1)$. Then we get $f(s)s_{23} - f(s) = b_t(s_{23} - 1)$. The monomials divisible by s_{23} in f will cancel out, and what remains is an equation of degree 3 true with probability 1. We repeat the same trick for s_{42}, i.e. we put $g(s) = (s_{42} - 1)$. From this, we have a simple linearization attack following the scenario S3. For each keystream bit, we obtain 2 equations of degree 3 in the s_i, and thus 2 equations of degree 3 in the k_i. The linearization will work as soon as $R > T$, and we have $T \approx \binom{128}{3} = 2^{18.4}$ monomials. We have $R = 2m$, and having

$m = T/2 = 2^{17.4}$ keystream bits will be sufficient. Our new attack on Toyocrypt is in $7/64 \cdot T^{log_2 7} = \mathbf{2^{49}}$ CPU clocks, requires 16 Gigabytes of memory and only about 20 kilobytes of (non-consecutive) keystream.

We programmed this attack to see if it works. Our simulations show that the number of linearly dependent equations is negligible and therefore the attack will be able to recover the key. Details are given in the extended version of this paper.

4 Background on LILI-128 and Simple Attacks

In principle our algebraic attacks are designed only for regularly clocked stream ciphers (or ciphers clocked in a known way). However in some cases, this difficulty can be removed. This is the case for LILI-128, a submission to Nessie European call for cryptographic primitives.

4.1 Eliminating the First Component

LILI-128 is a stream cipher composed of two LFSR-based filter generators, the first being used to clock the second. There are two basic strategies to by-pass this.

A. Since the key length of the first component is only 39 bits, we may guess these 39 bits and attack the second component alone. In LILI-128, the first component advances the clock of the second component by 1, 2, 3 or 4. Given the state of the first component, we have access to some number of non-consecutive keystream bits of the second component, at known positions. This is sufficient for our attacks, and the complexity of the attack is multiplied by 2^{39}.

B. Given more keystream bits, it is possible to avoid repeating the whole attack 2^{39} times. For this, we use the Lemma 1 from [23]: after clocking the first LFSR of LILI-128 $2^{39}-1$ times, the second LFSR advances exactly $\Delta_d = 5 \cdot 2^{38} - 1$ times. Thus, we may, instead of guessing the state of the clock control subsystem, clock it $2^{39} - 1$ at a time, and apply any of the XL attacks exactly as if the first generator did not exist.

In both cases, the bits the attacker has access to, are in some known places of the keystream of the second component (but not in chosen places). It is perfectly sufficient to apply directly, to the second component, all the algebraic attacks S1-S4 described in Section 2.3: the second component is attacked exactly as if it was a stand-alone filter generator.

Intermediate Solutions A-B. The period of the first component of LILI-128 is not a prime, and we have $2^{39} - 1 = 7 \cdot 79 \cdot 121369 \cdot 8191$. This suggests that one should be able to design a decimation attack, in which by clocking the generator t clocks at a time, for a suitable t, one could simulate a smaller LFSR, see [12] for more details. It could give a version of our later attacks, intermediate between A and B, in which both the keystream requirements and the attack complexity would be multiplied by some quantities, being both smaller than 2^{39}.

The Boolean Function Used in LILI-128

We call f the output function of LILI-128 (called f_d in [22]). It is a highly non-linear Boolean function of degree 6, built following [20] and specified in [22] or the extended version of this paper. It uses a subset of 10 variables:

$$(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}) \stackrel{def}{=} (s_0, s_1, s_3, s_7, s_{12}, s_{20}, s_{30}, s_{44}, s_{65}, s_{80}) .$$

4.2 First Attacks on LILI-128 (Scenarios S1 and S2)

First, it is possible to apply to LILI-128 our scenario S1. We have $d = 6$, $\varepsilon = 0$ and $R = m$. Then $T \approx \binom{89}{6} \approx 2^{29.2}$ monomials, and in order to have $R > T$ we will need $m \approx 2^{29.2}$. It gives an attack in[6] about:

$$2^{39} \cdot 7/64 \cdot T^{\log_2 7} \approx 2^{118} < 2^{128}.$$

Given the Boolean function used, it is not useful to extend the attack to the scenario S2: there is no good approximation of degree $d < 6$, as it gives $\varepsilon > 2^{-6}$ which is by far too big. However we may improve the attack following the method B from Section 4.1: instead of guessing the state of the clock control subsystem we clock it $2^{39} - 1$ at a time, and apply the above simple linearization S1-type attack with $\varepsilon = 0$, exactly as if the first generator did not exist. Now the complexity is only $\frac{7}{64} T^{\log_2 7} \approx 2^{79}$ but this version requires much more keystream bits: 2^{68} instead of 2^{29}.

5 Better Attacks on LILI-128

First we try to apply to LILI-128 the idea of factoring multivariate polynomials from Section 3.1. For this, we consider the part of degree 5 and 6 of f. It can be factored as follows:

$$x_7x_9\,(x_3x_8x_{10}+x_4x_6x_8+x_4x_6x_{10}+x_4x_8x_{10}+x_5x_6x_8+x_5x_6x_{10}+x_4x_6x_8x_{10}+x_5x_6x_8x_{10})$$

It means that when we multiply f by either $(x_7 + 1)$ or $(x_9 + 1)$, the degree collapses from $6 + 1 = 7$ to $4 + 1 = 5$. We consider then the factoring of the part of degree 5 and 4 of respectively $f(x) \cdot (x_7 + 1)$ and $f(x) \cdot (x_9 + 1)$. Only the second function can still be factored and it gives:

$$x_{10}\,(x_3x_7x_8x_9+x_5x_7x_8x_9+x_3x_7x_8+x_3x_8x_9+x_4x_7x_9+x_4x_8x_9+x_5x_7x_8+x_5x_7x_9+x_6x_7x_9)$$

From this we deduce the remarkable fact that $f(x)(x_9 + 1)(x_{10} + 1)$ is of degree **4**, instead of 8.

[6] This simple attack has already been described by Steve Babbage in [3].

The function	LILI-128 f_d						Random Boolean					
Degree of g	10	1	**2**	3	4	10	10	1	2	3	4	10
Degree of fg	3	4	4	4	4	4	3	4	4	4	4	4
Nb. of g	0	0	**4**	8	**14**	14	0	0	0	0	0	0

Table 1. Simulations on the number of linearly independent g such that fg is of low degree

5.1 Finding More Low Degree Multiples of f for LILI-128

We are looking for the number of linearly independent polynomials g, such that fg is of low degree. In order to find them, we are looking for linear dependencies in the following set of polynomials (stopped at maximum degree for g and some maximum degree for fg).

$$\{f(x), f(x) \cdot x_1, f(x) \cdot x_2, \ldots, f(x) \cdot x_1x_2, \ldots; \quad 1, x_1, x_2, \ldots, x_1x_2, \ldots\}$$

We note that the maximum degrees cannot be higher than 10, as there are only 10 variables. Here are our results with $fg \neq 0$, compared to a random Boolean function of the same size:

We have computed and tested all these solutions. For example, one can verify that:

$$f(x) \cdot x_8x_{10} = x_8x_{10}\left(x_2x_9 + x_3x_7 + x_4x_7 + x_5x_9 + x_1 + x_4 + x_5 + x_6\right).$$

We see that the function f of LILI-128, behaves much worse than a random Boolean function. This shows that the design of LILI-128 is far from being optimal against algebraic attacks.

Note: In the extended version of this paper we also study equations such that $fg = 0$ and show that they also exist with $d = 4$ for LILI-128. Moreover when $d = 5$, simulations show that they also exist for any randomly chosen Boolean function of the same size.

Consequences for LILI-128

Following the results of Section 5.1, given m keystream bits, we obtain $14 \cdot m$ multivariate equations of degree 4 in the key bits k_i of LILI-128. We will have to solve an overdefined system of multivariate equations of degree 4, true with probability 1. This is done by linearization. Following Section 4.1 there are two versions of the attack.

A In the first version (A) the state of the first generator is guessed and the complexity is multiplied by 2^{39}. For each keystream bit we obtain 14 equations of degree 4 in the k_i. For linearization we have $T = \binom{89}{4} = 2^{21}$ monomials, and we will need $m = T/14 = 2^{18}$ keystream bits in order to have $R > T$.

Our best new attack on LILI-128 requires then $2^{39} \cdot 7 \cdot T^{log_2 7}/64 \approx 2^{96}$ CPU clocks. This first attack version works given access to some m stream bits, being at some known positions.

B In the second version (B), the first component is clocked $2^{39}-1$ clocks at a time (see Section 4.1) and thus the number of keystream bits is multiplied by 2^{39}. We have the same $T = \binom{89}{4} = 2^{21.2}$, and the complexity is now $\mathbf{2^{57}}$ CPU clocks. The attack requires 762 Gigabytes of memory and 2^{57} bits of consecutive keystream, or, it can be seen that we only need 2^{18} keystream bits that are on some positions of the form $\alpha + \beta \cdot (2^{39}-1)$, for a fixed α. Once the state at the time α of the second LFSR is recovered, the state of the first LFSR can be found very quickly within 2^{39} tries.

Remark: This is not the the best attack known for LILI-128. In [23] it is shown that LILI 128 can be broken with 2^{46} bits of keystream, a lookup table of 2^{45} 89-bit words and computational effort which is roughly equivalent to 2^{48} DES operations. Our attack has however much more general consequences.

6 General Attack on Stream Ciphers Using a Subset of LFSR bits

In this section we show that the case of LILI-128 is not isolated. We will show that all very traditional stream ciphers, with linear feedback and a highly non-linear (stateless) filtering function are insecure, for example[7] when they use only a small subset of state bits.

We consider a stream cipher with n state bits, and using only a small subset of k state bits to derive the keystream bit. Thus we have:

$$\{x_1, x_2, \dots, x_k\} \subset \{s_0, s_1, \dots, s_{n-1}\}.$$

We assume that k is a small constant and n is the security parameter. For example for the second component of LILI-128 $k = 10, n = 89$.

We would like to mount an attack following the scenario S3 and for this, we are looking for low-degree polynomials $g \neq 0$, such that fg is also of low degree. In order to find them, similarly as in Section 5.1, we check for linear dependencies in the set of polynomials $C = A \cup B$ defined below as follows. First, we consider all possible monomials up to some maximum degree d (this part will later compose fg).

$$A = \{1, x_1, x_2, \dots, x_1 x_2, \dots\}$$

Then we consider all multiples of f, multiplied by monomials of the degree up to d (this degree corresponds to the degree of g):

$$B = \{f(x), f(x) \cdot x_1, f(x) \cdot x_2, \dots, f(x) \cdot x_1 x_2, \dots\}$$

[7] Though Toyocrypt does not satisfy this assumption, it is still broken by our attack, that will also work in many other interesting cases, see Section 7.

Let $C = A \cup B$. All elements of A,B and C, can be seen as multivariate polynomials in the x_i: for this we need to substitute f with its expression in the x_i. A set of multivariate polynomials with k variables cannot have a dimension greater than 2^k. If there are more than 2^k elements in our set, linear dependencies will exist. Such combinations allow to find a function g such that $f \cdot g$ is of substantially lower degree than f. More precisely we have the following theorem:

Theorem 6.0.1 (Low Degree Relations).
Let f be any Boolean function $f : GF(2)^k \to GF(2)$. Then there is a Boolean function $g \neq 0$ of degree at most $\lceil k/2 \rceil$ such that: $f(x) \cdot g(x)$ is of degree at most $\lceil (k+1)/2 \rceil$.

Proof: If we include in A all the monomials with degrees up to $\lceil k/2 \rceil$, then

$$|A| = \sum_{i=0}^{\lceil k/2 \rceil} \binom{k}{i} \geq \frac{1}{2} 2^k.$$

Similarly for B, we multiply f by all the monomials with degrees up to $\lceil (k+1)/2 \rceil$, then

$$|B| = \sum_{i=0}^{\lceil (k+1)/2 \rceil} \binom{k}{i} > \frac{1}{2} 2^k$$

The rank of $C = A \cup B$ cannot exceed 2^k. Since $|C| = |A| + |B| > 2^k$, some linear dependencies must exist. Moreover there are no linear dependencies in the part A of our set C, and therefore $g \neq 0$. This ends the proof.

Consequences of the Theorem

We see that for any stream cipher with linear feedback, for which the non-linear filter uses k variables, it is possible to generate at least one equation of degree $\approx k/2$ in the n keystream bits. These equations will be solved, as usual, by linearization. We will need at most $\sum_{i=0}^{k/2} \binom{n}{i} \approx \binom{n}{k/2}$ keystream bits in order to obtain a complete saturated system solvable by linearization. Moreover, if for a given f, there are several linear dependencies in C, we will be able to use several linearly independent g, and for each keystream bit will obtain several equations. Then the keystream requirements will be divided accordingly. For example in Section 5.1 they are divided by 14. To summarise, we get the following general attack for any Boolean function f with k inputs.

d	$k/2$
ε	0
Data	$\binom{n}{k/2}$
Memory	$\binom{n}{k/2}^2$
Complexity	$\binom{n}{k/2}^\omega$

Remark. This attack deals with the worst case. For specific functions the cipher may be much less secure. For example in LILI-128, $k = 10$ and, with a strict application of Theorem 6.0.1 we obtain the worst case complexity $\mathcal{O}(n^{6\omega})$ ($\forall$ Boolean function). In the extended version of this paper we show that, on average (for a random Boolean function) the complexity is $\mathcal{O}(n^{5\omega})$. Finally, for the specific function used in LILI-128, our attack from Section 5.1 is in $\mathcal{O}(n^{4\omega})$.

The Complexity of the Attack. This attack is polynomial when k is fixed. This attack will only be exponential in n, if $k = \mathcal{O}(n)$. The number of bits used in a non-linear filter can**not** be small.

In practice, talking about polynomial (or not-polynomial) time is misleading and should always be confronted with concrete results. Knowing that the maximum degree of the filtering function cannot exceed k, it can be seen that in the scenario S1, any stream cipher with linear feedback can be broken in about $\binom{n}{k}^{\omega}$, given $\binom{n}{k}$ keystream bits, by simple linearization. This simple attack is already polynomial when k is fixed, and well known, see for example [13] or [3]. Here precisely is the problem: many stream ciphers, some of them unpublished and proprietary, have been designed in such a way that, one has for example $\binom{n}{k}^{\omega} \approx 2^{80}$. In practice, since our complexity $\binom{n}{k/2}^{\omega}$ is, very roughly the square root of the previous one, we can break all these ciphers in roughly 2^{40}.

Resistance Criteria Against This Attack: By inspection we verify that the requirement to achieve the best possible resistance against our attack is the following. We need to make sure that no linear dependencies exist when both sets A and B are generated up to any degree, strictly smaller than the degree for which dependencies must exist, due to the theorem. It can be seen that it boils down to assuring that:

Optimal Resistance Criterion: When both A and B are generated with degrees in the x_i up to $\lfloor k/2 \rfloor$ then the rank of the set $C = A \cup B$ is maximal. It is easy to show that this criterion implies that the degree of f will be sufficiently large to prevent the attack S1. However, it cannot guarantee that attacks of type S2 (as in [10]) or S4, will not exist.

7 Consequences for the Design of Stream Ciphers

There are many interesting cases in which the attacks described in this paper will work. For example, it can be seen that they will work for any regularly clocked stream cipher with linear feedback and any Boolean function f such that:

1. either f uses a small subset of state bits, (e.g. 10), as in LILI-128, see Section 6,
2. or is very, very sparse, as in Toyocrypt, see [10],
3. or can be factored with a low degree factor, as in Toyocrypt, see Section 3.1.
4. or can be approximated by one of the above (see e.g. our Scenario S4),
5. or its part of high degree is one of the above, for example in Section 5 it has low degree factors.

We conclude that, in a stream cipher with linear feedback, the filtering function should use many state bits, for example at least[8] 32, and should not be too sparse, so it has also many terms of very high degree (for example at least 32). Moreover the part of high degree, should not have a low degree factor, and should itself also use many state bits. Then, no approximation of the part of high degree (for example obtained by removing or adding a few very high degree terms) should have a low degree factor, or should use a small number of state bits. Finally, we see in the example of LILI-128, that specific functions may behave worse than a random Boolean function of the same size, for no apparent reason.

Our Algebraic Security Criterion on f. It can be seen that the attacks described in the present paper are possible when there exist h, g such that $fg + h = 0$, with either h is of low degree, or $h = 0$ and g is of low degree. We observe that in both cases, the degree of fg is smaller than expected: it is less than the sum of the degrees of f and g. Hence, ideally, to resist algebraic attacks, given the function f, for every function g of "reasonable" size the degree of fg should always be equal[9] to $deg(f) + deg(g)$. This is the new design criterion we propose for Boolean functions used in stream ciphers.

General Algebraic Attacks on Stream Ciphers. More generally, if there are several filtering functions f_i in a stream cipher, there should be no algebraic combination of the f_i and of "reasonable" size, that would have an unusually low degree. By extension, this criterion also applies to stream ciphers that have only one filtering function. Indeed a cipher having only one filtering function f, can be seen as using several functions defined as: $f, f \circ L, f \circ L^2, \ldots$. It can be seen that, in all cases, our security criterion can be re-formulated as: **there should be no non-trivial multivariate relations of low degree that relate the key bits and one or many output bits of the cipher.** Otherwise, if only one such multivariate relation exists (for any reason), an algebraic attack as described in this paper will be possible. It is important to see that this attack scenario (that could be called S5) applies potentially to all ciphers with linear feedback, even for filters with memory, see [2, 11], and not only to ciphers using stateless Boolean functions.

We obtain a design criterion, that is basically identical to the notion of non-trivial equations defined in Section 2 of [7]. It is also very similar to the design criterion given in [9] for the S-boxes of block ciphers.

8 Conclusion

In this paper we studied algebraic attacks on stream ciphers with linear feedback (e.g. using LFSRs), such that the only non-linear component is a filtering function. We reduce their cryptanalysis to solving a system of algebraic equations, namely an overdefined system of multivariate binary equations of low degree. We

[8] However, a too large number of state bits used in the filter function may conflict with certain design criteria introduced in [13] to render inversion attacks infeasible.

[9] In practice, obviously, it is sufficient that the degree of fg does not become too small. For example if it is always at least 10, all the attacks described in the present paper become impractical.

present a method to decrease the degree of the equations, by multiplying them by well chosen multivariate polynomials. Thus, we are able to cryptanalyse two well known stream ciphers.

For Toyocrypt, a Cryptrec submission, our best attack requires only 2^{49} CPU clocks and some 20 Kbytes of keystream, for a 128-bit cipher. Compared to the best known attack on Toyocrypt so far by Mihaljevic and Imai [18], our new attack has simultaneously a much lower complexity, much lower memory, and much looser requirements on the keystream. We also attacked LILI-128, a Nessie submission, and obtained an attack in 2^{57} CPU clocks for a 128-bit cipher, unfortunately far from being practical (requires 2^{57} keystream bits).

The main contribution of the present paper is the following. If the non-linear function of a cipher with linear feed-back uses only a small subset of state bits, the cipher will be insecure, though satisfying all the previously known design criteria. If only k keystream bits are used out of n, it is widely known that an attack in $\binom{n}{k}^{\omega}$ exists, whatever is the Boolean function, see [13] or [3]. Thus many stream ciphers, have been designed in such a way that, one has for example $\binom{n}{k}^{\omega} \approx 2^{80}$. In practice, since the worst-case complexity of our attack is $\binom{n}{k/2}^{\omega}$, roughly the square root of the previous one, we can break these ciphers in about 2^{40}. This attack works for any Boolean function (the worst-case). The examples of Toyocrypt and LILI-128 show that for specific ciphers, the resistance against algebraic attacks may be substantially worse.

It has long been known that for stateless Boolean functions used in stream ciphers one is confronted with design criteria that may conflict each other. Our attacks impose even stronger restrictions on the choice of such functions. Extrapolating from our attack, we proposed a very general security criterion for stream ciphers: the non-existence of multivariate relations of low degree relating the key bits and the output bits. It turns out to be basically identical to the security criterion defined in Section 2 of [7] for multivariate trapdoor functions, and also to the requirements advocated in [9] for S-boxes of block ciphers.

Important Note: The attacks described in the present paper work given any subset of keystream bits. In [11] it is shown that if the keystream bits are consecutive, the attack complexity can be substantially reduced.

Acknowledgments: Many thanks to Philip Hawkes, Josef Pieprzyk and the anonymous referees of Eurocrypt for their helpful comments.

References

1. Ross Anderson: *Searching for the Optimum Correlation Attack,* FSE'94, LNCS 1008, Springer, pp 137-143, 1994.
2. Frederik Armknecht: *A Linearization Attack on the Bluetooth Key Stream Generator,* Available on http://eprint.iacr.org/2002/191/. 13 December 2002
3. Steve Babbage: *Cryptanalysis of LILI-128,* Nessie project internal report, available at https://www.cosic.esat.kuleuven.ac.be/nessie/reports/, 22 January 2001.
4. Eli Biham: *A Fast New DES Implementation in Software,* FSE'97, Springer, LNCS 1267, pp. 260-272, 1997.
5. Paul Camion, Claude Carlet, Pascale Charpin and Nicolas Sendrier: *On Correlation-immune Functions,* In Crypto'91, LNCS 576, Springer, pp. 86-100, 1992.

6. Don Coppersmith, Shmuel Winograd: *Matrix multiplication via arithmetic progressions,* J. Symbolic Computation (1990), 9, pp. 251-280, March 1990.
7. Nicolas Courtois: *The security of Hidden Field Equations (HFE)*, Cryptographers' Track Rsa Conference 2001, San Francisco 8-12 April 2001, LNCS2020, Springer, pp. 266-281, 2001.
8. Nicolas Courtois and Jacques Patarin: *About the XL Algorithm over* $GF(2)$, Cryptographers' Track RSA 2003, San Francisco, April 13-17 2003, LNCS, Springer.
9. Nicolas Courtois and Josef Pieprzyk: *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations,* Asiacrypt 2002, LNCS 2501, Springer, 2002. A preprint with a different version of the attack is available at http://eprint.iacr.org/2002/044/.
10. Nicolas Courtois: *Higher Order Correlation Attacks, XL algorithm and Cryptanalysis of Toyocrypt,* ICISC 2002, November 2002, Seoul, Korea, LNCS 2587, Springer, 2002. An updated version is available at http://eprint.iacr.org/2002/087/.
11. Nicolas Courtois: *Fast Algebraic Attacks on Stream Ciphers with Linear Feedback,* Preprint, January 2003, available from the author.
12. Eric Filiol: *Decimation Attack of Stream Ciphers,* Indocrypt 2000, LNCS 1977, pp. 31-42, 2000. Available on eprint.iacr.org/2000/040.
13. Jovan Dj. Golic: *On the Security of Nonlinear Filter Generators,* FSE'96, LNCS 1039, Springer, pp. 173-188.
14. Jovan Dj. Golic: *Fast low order approximation of cryptographic functions,* Eurocrypt'96, LNCS 1070, Springer, pp. 268-282, 1996.
15. Willi Meier and Othmar Staffelbach: *Fast correlation attacks on certain stream ciphers,* Journal of Cryptology, 1(3):159-176, 1989.
16. Willi Meier and Othmar Staffelbach: *Nonlinearity Criteria for Cryptographic Functions,* Eurocrypt'89, LNCS 434, Springer, pp.549-562, 1990.
17. Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone: *Handbook of Applied Cryptography*, CRC Press, 1997.
18. M. Mihaljevic, H. Imai: *Cryptanalysis of Toyocrypt-HS1 stream cipher*, IEICE Transactions on Fundamentals, vol. E85-A, pp. 66-73, Jan. 2002. Available at http://www.csl.sony.co.jp/ATL/papers/IEICEjan02.pdf. Eurocrypt'88,
19. Rainer A. Rueppel: *Analysis and Design of Stream Ciphers*, Springer, New York, 1986.
20. Palash Sarkar, Subhamoy Maitra: *Nonlinearity Bounds and Constructions of Resilient Boolean Functions,* In Crypto 2000, LNCS 1880, Springer, pp. 515-532, 2000.
21. Adi Shamir, Jacques Patarin, Nicolas Courtois and Alexander Klimov: *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations*, Eurocrypt'2000, LNCS 1807, Springer, pp. 392-407, 2000.
22. L. Simpson, E. Dawson, J. Golic and W. Millan: *LILI Keystream Generator,* SAC'2000, LNCS 2012, Springer, pp. 248-261, 2000. Available at www.isrc.qut.edu.au/lili/.
23. Markku-Juhani Olavi Saarinen: *A Time-Memory Tradeoff Attack Against LILI-128,* FSE 2002, LNCS 2365, Springer, pp. 231-236, 2002. Available at http://eprint.iacr.org/2001/077/.
24. Claude Elwood Shannon: *Communication theory of secrecy systems,* Bell System Technical Journal 28 (1949), see in patricular page 704.
25. Volker Strassen: *Gaussian Elimination is Not Optimal,* Numerische Mathematik, vol 13, pp 354-356, 1969.