

Distinguishing Stream Ciphers with Convolutional Filters

Joan Daemen and Gilles Van Assche
STMicroelectronics – Smart Cards ICs Division
Excelsiorlaan 44–46, 1930 Zaventem, Belgium

February 15, 2005

Abstract

This paper presents a new type of distinguisher for the shrinking generator and the alternating-step generator with known feedback polynomial and for the multiplexor generator. For the former the distinguisher is more efficient than existing ones and for the latter it results in a complete breakdown of security. The distinguisher is conceptually very simple and lends itself to theoretical analysis leading to reliable predictions of its probability of success.

1 Introduction

In this paper we present efficient distinguishers for a class of stream ciphers. This class can be characterized as irregularly sampled linear feedback shift registers (LFSR). These stream ciphers have the following in common:

- A set of *source* registers, each of which generates a source sequence.
- The source sequences are sampled in an irregular fashion to form an *output* sequence. In most cases, the sampling is governed by an independent sequence generator, typically just another LFSR. The latter is called the *sampling* sequence or sampling LFSR.

Examples of this type of stream ciphers are the shrinking generator [3, 16], the alternating-step generator [12, 16] and some variants of the multiplexor generator [13, 16]. The bits in an LFSR source sequence satisfy a linear recurrence that can be very easily detected. Clearly, as each bit in the output sequence corresponds to a bit in a source sequence, the bits in the output sequence may also satisfy a linear recurrence. The irregularity of the sampling process is supposed to make this hard to exploit. This paper now presents distinguishers for the shrinking and alternating-step generators exploiting all remainders of linear recurrence in the output sequence. To build such a distinguisher requires knowledge of the feedback polynomial of the source sequence. It also presents very powerful distinguishers for the multiplexor generator exploiting the weakness that a single bit in the source stream may appear multiple times in the output stream. We call the distinguishers presented in this paper *convolutional filters* as they make use of convolution as their main operation.

Correlation attacks on the shrinking generator were already described in [4], analyzed in [17] and later improved in [14]. Detectable statistical weaknesses in the output stream were shown in [5] and [6] if the feedback polynomial has very low weight or moderate degree. More recent work includes another correlation analysis of the shrinking generator in [8] and

of the alternating-step generator in [9] and [11]. A draft paper [10] saw the light describing a statistical distinguisher for the shrinking generator.

The work that led to this paper was triggered by an efficient attack on the shrinking generator described in [2] and can be considered as an improvement and extension of the latter. It improves the attacks in [2] in that convolutional filters require less output stream for the same probability of success. As opposed to the distinguisher proposed in [2], convolutional filters are conceptually very simple: they return a real number and require no decision rules (hard or soft) in the computation thereof. As such, they require no parameter trade-offs or fine-tuning, their probability of success is easy to compute analytically and there is no discrepancy between the theory and the simulation results. Finally, while [2] describe attacks for the shrinking generator only, this paper presents distinguishers for the shrinking generator, the alternating-step generator and the multiplexor generator. In the rest of this paper we first provide a number of definitions, then the distinguishers for the shrinking and alternating-step generators, the distinguishers for multiplexor generators and finally a description of our simulation results.

2 Definitions

2.1 Sequences

We denote sequences by lowercase letters such as a and b and their individual components with notation a_t and b_t , where the indices start from 1. We define the product of two sequences $c = a \times b$ as the sequence with $c_t = a_t b_t$ and the convolution of a sequence a with a function f , $c = a \otimes f$ as the sequence with $c_t = \sum_i f(i) a_{t-i}$.

2.2 Linear feedback shift registers

Linear feedback shift registers (LFSR) come in two types. In the *Fibonacci configuration* the feedback is from a number of stages to the first stage while in a *Galois configuration*, the feedback is from the last stages to a number of stages. Both configurations are governed by a feedback polynomial that determines the positions of the stages involved in the feedback. The output bits of a linear feedback shift register (LFSR) satisfy a recurrence relation determined by its feedback polynomial:

$$a_i \oplus a_{i-G_1} \oplus a_{i-G_2} \oplus \cdots \oplus a_{i-G_{w-1}} = 0. \quad (1)$$

We call w the weight of the feedback polynomial and define the $w - 1$ *gaps* as $g_1 = G_1$, $g_2 = G_2 - G_1, \dots, g_{w-1} = G_{w-1} - G_{w-2}$. The output bits of an LFSR satisfy many recurrence relations, one for every multiple of the feedback polynomial. For some distinguishers, the efficiency tends to decrease with the weight of the polynomial and it is advantageous to find multiples of the feedback polynomial with low weight. Techniques for doing so are described in [1, 7, 15, 18].

2.3 Index maps

We define the index map $S(j)$ associated with a (sampling) sequence s as:

$$S(j) = \min\{k \mid \sum_{i=1}^k s_i = j\}, \quad (2)$$

where the bits s_i are interpreted as integers 0 and 1. Having $S(j) = k$ requires that $s_k = 1$ and that the interval $[s_1 \dots s_{k-1}]$ contains $j - 1$ ones. Clearly, $S(j)$ is an increasing function. Given a random binary sequence s , $S(j)$ is a stochastic variable with probability distribution:

$$\Pr[S(j) = k] = 2^{-k} \binom{k-1}{j-1} \text{ if } k \geq j \text{ and } 0 \text{ otherwise.} \quad (3)$$

More generally, having $S(j+h) - S(j) = g$ requires the interval $[s_{S(j)} \dots s_{S(j)+g-1}]$ to contain $h - 1$ ones and $s_{S(j+h)} = 1$:

$$\Pr[S(j+h) - S(j) = g] = 2^{-g} \binom{g-1}{h-1} \text{ if } g \geq h \text{ and } 0 \text{ otherwise.} \quad (4)$$

We denote $\Pr[S(j+h) - S(j) = g]$ by $\mathcal{S}(g, h)$ as it is independent of j . This function satisfies:

$$\sum_{g=h}^{\infty} \mathcal{S}(g, h) = 1 \text{ and } \sum_{h=1}^g \mathcal{S}(g, h) = 1/2. \quad (5)$$

For a given g , $\mathcal{S}(g, h)$ has a maximum in $h = (g+1)/2$ for g odd and in $h = (g+1 \pm 1)/2$ for g even. We denote the mean value of a stochastic variable x by $\langle x \rangle$ and its variance by σ^2 . Unless g is very small or h is very far from $g/2$, $\mathcal{S}(g, h)$ is closely approximated by a (scaled) normal distribution:

$$\mathcal{S}(g, h) \approx \frac{1}{2} \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(h-\langle h \rangle)^2}{2\sigma^2}}, \quad (6)$$

with $\langle h \rangle = (g+1)/2$ and $\sigma^2 = (g-1)/4$. For a given h , the shape of $\mathcal{S}(g, h)$ is slightly skewed with respect to a normal distribution. It reaches its maximum value in both $g = 2h - 2$ and $g = 2h - 1$ and has $\langle g \rangle = 2h$ and $\sigma^2 = 2h$.

2.4 The shrinking generator

A shrinking generator (SG) is a stream cipher with a single source LFSR and a sampling LFSR. During an iteration both registers are clocked. If the sampling bit is 1, the source bit is presented at the output of the generator. Otherwise, no output bit is generated. On the average the SG requires two iterations per bit generated. Its output bits satisfy:

$$z_i = a_{S(i)}, \quad (7)$$

with a the source sequence and $S(i)$ the index map of the sampling sequence.

2.5 The alternating-step generator

An alternating-step generator (ASG) is a stream cipher with two source LFSRs generating sequences a and b and a sampling LFSR generating sequence s . During an iteration the sampling LFSR and only one of the two source LFSRs is clocked. Which one of the two source LFSRs is clocked depends on the output bit of the sampling LFSR. The output bit y of the ASG is the XOR of the two output bits of the source LFSR. The difference of two subsequent output bits $z_t = y_t \oplus y_{t-1}$ of an ASG is either the XOR of two output bits $a_i \oplus a_{i-1}$ of one source LFSR or $b_j \oplus b_{j-1}$ of the other source LFSR. Note that if a sequence a satisfies a recurrence relation, this is also the case for a sequence c with $c_i = a_i \oplus a_{i-1}$. In the following,

we will deal with the sequences z , c and d and not the sequences y , a and b . The bits of c map to bits in z by

$$c_i = z_{S(i)}. \quad (8)$$

The bits of d with $d_i = b_i \oplus b_{i-1}$ map to bits in z in a similar way:

$$d_i = z_{S'(i)}, \text{ with } S'(i) = \min\{k \mid \sum_{j=1}^k (1 - s_j) = i\}. \quad (9)$$

2.6 Multiplexor generators

We consider multiplexor generators with a single source LFSR and a single sampling LFSR. During an iteration both registers are clocked. A multiplexor taking as input a number n of stages in the sampling LFSR selects a stage in the source LFSR whose contents is presented as output bit. The input to the multiplexor can be modeled as a sampling sequence S of integers in the range $[0, 2^n - 1]$ and the stages selected as a function of S_t as an array M with 2^n stage positions. We call M the *selection position table*.

If the source LFSR has a Fibonacci configuration, the multiplexor generator can be modeled as a binary source sequence a sampled by a sampling sequence S in the following way:

$$z_t = a_{t+M[S_t]}. \quad (10)$$

If the source LFSR has a Galois configuration, this model does not apply.

3 A basic distinguisher for SG and ASG

If we select a number w output bits, they may correspond to w source bits that satisfy the recurrence relation. We denote the selected output bits by $z_t, z_{t-H_1}, z_{t-H_2}, \dots, z_{t-H_{w-1}}$ and the *gaps* of this selection as $H = (h_1, h_2, \dots, h_{w-1})$. Given H and the gaps $G = (g_1, g_2, \dots, g_{w-1})$ of the recurrence relation, we can compute the probability that the selected output bits correspond to source bits that satisfy the recurrence relation. This probability is independent of t and only depends on G and H . We denote it by $P(G|H)$. Given a sequence z , we define x_t as:

$$x_t = (-1)^{z_t \oplus z_{t+H_1} \oplus \dots \oplus z_{t+H_{w-1}}}. \quad (11)$$

Using the convention $\bar{z}_i = (-1)^{z_i}$, this becomes $x_t = \bar{z}_t \bar{z}_{t+H_1} \dots \bar{z}_{t+H_{w-1}}$. We can model the probability distribution of x_t as the combination of two distributions:

- If the output bits correspond to source bits that satisfy the linear recurrence, the distribution of x_t has a peak equal to 1 at 1. For an SG or ASG, this happens with probability $P(G|H)$.
- Otherwise, the distribution of x_t has equal peaks of value $1/2$ both at positions 1 and -1 . For an SG or ASG, this happens with probability $1 - P(G|H)$.

Hence, for an SG or ASG, x_t has a distribution with mean $\langle x_t \rangle = P(G|H)$ and variance is $1 - P(G|H)^2 \approx 1$. For a truly random sequence z , x_t has mean 0 and variance 1.

The basic distinguisher now consists of the following. Given a stream z , compute x_t for a large range of t values and take the average value X :

$$X = \frac{1}{L} \sum_{t=1}^L x_t. \quad (12)$$

If we consider the different x_t as independent, X is the average of a large number of independent stochastic variables all with variance 1 and so has a normal distribution with standard deviation $1/\sqrt{L}$. If z is the output of an SG or ASG, $\langle X \rangle = P(G|H)$ and if z is a random sequence $\langle X \rangle = P(G|H)$.

If $X > P(G|H)/2$ we decide z is the output of an SG (or ASG). If $L = P(G|H)^{-2}$, the probability of error is about 31%. To obtain a probability of error below 1%, we must take $L \approx 22P(G|H)^{-2}$. For a given probability of success, the amount $P(G|H)^{-2}$ determines the length of the output sequence required with a given distinguisher. We denote $P(G|H)^{-2}$ by L_d .

3.1 The shrinking generator

The probability that a gap h in the output sequence maps to a gap g in the input sequence is given by $\Pr[S(j+h) - S(j) = g] = \mathcal{S}(g, h)$. The $w-1$ gaps of H are mapped to $w-1$ gaps in G in an independent way. Therefore it follows that:

$$P_s(G|H) = \prod_{i=1}^{w-1} \mathcal{S}(g_i, h_i). \quad (13)$$

Choosing the gaps $h_i = (g_i + 1)/2$ such that $P_s(G|H)$ is maximized and using the Gaussian approximation yields:

$$P_s(G|H) \approx \prod_{i=1}^{w-1} \frac{1}{\sqrt{2\pi(g_i - 1)}} \text{ and } L_d = (2\pi)^{w-1} \prod_{i=1}^{w-1} (g_i - 1). \quad (14)$$

3.2 The alternating-step generator

The probability that a gap h in the output sequence maps to a gap g in the source sequences c is given by $\Pr[S(j+g) - S(j) = h] = \mathcal{S}(h, g)$. The $w-1$ gaps of H are mapped to $w-1$ gaps in G in an independent way. However, we require that the bits come from source sequence c and not d , which happens with probability $1/2$. Therefore it follows that:

$$P_a(G|H) = \frac{1}{2} \prod_{i=1}^{w-1} \mathcal{S}(h_i, g_i). \quad (15)$$

Choosing the gaps $h_i = 2g_i - 1$ such that $P_a(G|H)$ is maximized and using the Gaussian approximation yields:

$$P_a(G|H) \approx \frac{1}{2} \prod_{i=1}^{w-1} \frac{1}{2\sqrt{\pi(g_i - 1)}} \text{ and } L_d = 4(4\pi)^{w-1} \prod_{i=1}^{w-1} (g_i - 1). \quad (16)$$

4 A convolutional filter for SG and ASG

Instead of just considering combinations of bits of the output sequence for which $P(G|H)$ is optimum, we introduce a more sophisticated distinguisher that considers *all* combinations of w bits of z for which $P(G|H)$ is different from 0. We compute a function Y as:

$$Y = \frac{1}{L} \sum_t y_t \text{ with } y_t = \sum_H C_H \bar{z}_t \bar{z}_{t+H_1} \cdots \bar{z}_{t+H_{w-1}}. \quad (17)$$

Here the C_H are weighing factors as the optimum result is not necessarily obtained by just adding all combinations. Each y_t is the sum of a number of independent expressions $C_H \bar{z}_t \bar{z}_{t+H_1} \cdots \bar{z}_{t+H_{w-1}}$. Such an expression has variance C_H^2 and mean $C_H P(G|H)$. If we want y_t to have a variance equal to 1, we must choose the C_H values such that $\sum_H C_H^2 = 1$. In other words, the C_H values can be seen as the coordinates of a vector of length 1. The mean value of y_t is:

$$\langle y_t \rangle = \sum_H C_H P(G|H). \quad (18)$$

The latter can be seen as the inner product between two vectors, the C -vector and the $P(G|H)$ -vector. The mean value $\langle y_t \rangle$ for a truly random sequence being zero, we wish to maximize this inner product so as to best distinguish SG or ASG from other generators. We must thus choose the vector C equal to the vector $P(G|H)$ divided by its norm, hence:

$$C_H = \frac{P(G|H)}{\sqrt{\sum_H P(G|H)^2}}. \quad (19)$$

For this choice of C_H , we obtain:

$$\langle y_t \rangle = \sqrt{\sum_H P(G|H)^2} \text{ and } L_d = \frac{1}{\sum_H P(G|H)^2}. \quad (20)$$

4.1 The shrinking generator

We can now compute the value of L_d for an SG given the feedback polynomial G of its source register:

$$L_d^{-1} = \sum_{h_1} \sum_{h_2} \cdots \sum_{h_{w-1}} \prod_{i=1}^{w-1} \mathcal{S}(g_i, h_i)^2 \quad (21)$$

$$= \sum_{h_1} \mathcal{S}(g_1, h_1)^2 \sum_{h_2} \mathcal{S}(g_2, h_2)^2 \cdots \sum_{h_{w-1}} \mathcal{S}(g_{w-1}, h_{w-1})^2. \quad (22)$$

Introducing following notation

$$\rho_g(h) = \frac{\mathcal{S}(g, h)}{\sqrt{\sum_h \mathcal{S}(g, h)^2}} \quad (23)$$

results in:

$$L_d^{-1} = \prod_{i=1}^{w-1} \sum_{h_i} \mathcal{S}(g_i, h_i)^2 \text{ and } C_H = \prod_{i=1}^{w-1} \rho_{g_i}(h_i). \quad (24)$$

Using the Gaussian approximation yields $\sum_h \mathcal{S}(g, h)^2 \approx 1/4\sqrt{\pi(g-1)}$ and

$$L_d = (4\sqrt{\pi})^{w-1} \sqrt{\prod_i (g_i - 1)}. \quad (25)$$

In the following theorem we prove that the stream y can be computed iteratively by taking $w-1$ convolutions and $w-1$ stream multiplications.

Theorem 1 *The computation of $y_t = v_t^{(0)}$ for an SG is given as*

$$\begin{aligned} v^{(w-2)} &= \bar{z} \times (\bar{z} \otimes \rho_{g_{w-1}}), \dots \\ v^{(i)} &= \bar{z} \times (v^{(i+1)} \otimes \rho_{g_{i+1}}), \dots \\ v^{(0)} &= \bar{z} \times (v^{(1)} \otimes \rho_{g_1}). \end{aligned}$$

Proof: We have:

$$\begin{aligned} v_t^{(0)} &= \sum_{h_1} \rho_{g_1}(h_1) \bar{z}_t v_{t-h_1}^{(1)} \\ &= \sum_{h_1} \rho_{g_1}(h_1) \left(\sum_{h_2} \bar{z}_{t-h_1} \rho_{g_2}(h_2) v_{t-(h_1+h_2)}^{(2)} \right) \bar{z}_t \\ &= \sum_{h_1} \rho_{g_1}(h_1) \sum_{h_2} \rho_{g_2}(h_2) v_{t-H_2}^{(2)} \bar{z}_{t-H_1} \bar{z}_t \\ &= \sum_{h_1} \sum_{h_2} \rho_{g_1}(h_1) \rho_{g_2}(h_2) v_{t-H_2}^{(2)} \bar{z}_t \bar{z}_{t-H_1} \\ &= \dots \\ &= \sum_{h_1} \dots \sum_{h_{w-1}} \prod_{i=1}^{w-1} \rho_{g_i}(h_i) \bar{z}_t \bar{z}_{t-H_1} \dots \bar{z}_{t-H_{w-1}} \\ &= \sum_H \left(\prod_{i=1}^{w-1} \rho_{g_i}(h_i) \right) \bar{z}_t \bar{z}_{t-H_1} \dots \bar{z}_{t-H_{w-1}}. \end{aligned}$$

We thus correctly obtain

$$y_t = v_t^{(0)} = \sum_H C_H \bar{z}_t \bar{z}_{t-H_1} \dots \bar{z}_{t-H_{w-1}} \text{ with } C_H = \prod_{i=1}^{w-1} \rho_{g_i}(h_i). \quad (26)$$

□

4.2 The alternating-step generator

We can now compute the values of L_d and C_H for an ASG given the gaps G of a feedback polynomial of one of its source registers:

$$L_d^{-1} = \sum_{h_1} \sum_{h_2} \dots \sum_{h_{w-1}} \frac{1}{4} \prod_{i=1}^{w-1} \mathcal{S}(h_i, g_i)^2 \quad (27)$$

$$= \frac{1}{4} \sum_{h_1} \mathcal{S}(h_1, g_1)^2 \sum_{h_2} \mathcal{S}(h_2, g_2)^2 \dots \sum_{h_{w-1}} \mathcal{S}(h_{w-1}, g_{w-1})^2. \quad (28)$$

Introducing following notation:

$$\mu_g(h) = \frac{\mathcal{S}(h, g)}{\sqrt{\sum_h \mathcal{S}(h, g)^2}} \quad (29)$$

results in:

$$L_d = \frac{4}{\prod_{i=1}^{w-1} \sum_{h_i} \mathcal{S}(h_i, g_i)^2} \text{ and } C_H = \prod_{i=1}^{w-1} \mu_{g_i}(h_i). \quad (30)$$

The expression $\sum_h \mathcal{S}(h, g)^2$ appears to be very closely approximated by $1/2\sqrt{2\pi(g-1)}$, yielding

$$L_d = 4(2\sqrt{2\pi})^{w-1} \sqrt{\prod_i (g_i - 1)}. \quad (31)$$

Theorem 2 *The computation of $y_t = v_t^{(0)}$ for an ASG is given as*

$$\begin{aligned} v^{(w-2)} &= \bar{z} \times (\bar{z} \otimes \mu_{g_{w-1}}), \dots \\ v^{(i)} &= \bar{z} \times (v^{(i+1)} \otimes \mu_{g_{i+1}}), \dots \\ y &= \bar{z} \times (v^{(1)} \otimes \mu_{g_1}). \end{aligned}$$

The proof is very similar to that of Theorem 1.

4.3 Usage of multiple recursion relations

In the ASG, we can conduct the same attack using the feedback polynomial of source sequence b . Moreover, we can conduct the attack for any polynomial that is a multiple of a feedback polynomial of (one of) the source registers. In general, given any number of independent distinguishers with mean $\langle y^{(i)} \rangle$, the optimum distinguisher is formed by

$$y_t = \frac{\sum_i \langle y^{(i)} \rangle y_t^{(i)}}{\sqrt{\sum_i \langle y^{(i)} \rangle^2}} \text{ yielding } L_d = \frac{1}{\sum_i \langle y^{(i)} \rangle^2} = \frac{1}{\sum_i \frac{1}{L_d^{(i)}}}. \quad (32)$$

5 Distinguishers for multiplexor generators

We construct distinguishers exploiting the fact that source stream bits may appear multiple times in the output stream. Whereas the distinguishers for the SG and ASG reveal weaknesses in the source sequences, here the distinguishers work independently from the nature of the source stream and reveals weaknesses due to the sampling process itself.

5.1 Fibonacci configuration

The probability that two bits in the output stream separated by a gap h originate from the same bit in the source stream is:

$$\Pr[t + h + M[S_{t-h}] = t + M[S_t]] = \Pr[M[s] - M[s'] = h], \quad (33)$$

for independent random variables s and s' following the same distribution as S . We define the distribution function of the selection position table, $P_M(i)$, as

$$P_M(i) = \Pr[M[a] = i] = 2^{-n} \sum_j \delta_{iM[j]} , \quad (34)$$

with δ the Kronecker delta. If we now define Q_M as the convolution of P_M with itself, $Q_M = P_M \otimes P_M$, we have:

$$\Pr(M[s] - M[s'] = h) = Q_M(h). \quad (35)$$

So $Q_M(h)$ gives the probability that two output bits separated by a gap h originate from the same bit in the source stream.

A basic distinguisher consists of the following. Find the gap h_m for which $Q_M(h)$ is maximum and compute $X = \sum_t x_t / \sqrt{L}$ with $x_t = \bar{z}_t \bar{z}_{t-h_m}$. Clearly $[x_t] = Q_M(h_m)$, yielding $L_d = Q_M(h_m)^{-2}$. We can build a convolutional filter that exploits the probabilities $Q_M(h)$ for all gaps h :

$$Y = \frac{1}{L} \sum_{t,h>0} C_h \bar{z}_t \bar{z}_{t-h} \text{ or } Y = \frac{1}{L} \sum_t y_t \text{ with } y_t = \bar{z}_t \sum_{h>0} C_h \bar{z}_{t-h} . \quad (36)$$

The restriction $h > 0$ is there to ensure that every expression of type $y_t y_{t+h}$ appears only once. With a similar argument as in Section 4, the optimum values for C_h are given by $C_h = q_M(h)$ with

$$q_M(h) = \frac{Q_M(h)}{\sqrt{\sum_{h>0} Q_M(h)^2}} \text{ if } h > 0 \text{ and } 0 \text{ otherwise.} \quad (37)$$

This yields:

$$L_d = \frac{1}{\sum_{h>0} Q_M(h)^2} \text{ and } C_h = q_M(h), \quad (38)$$

resulting in:

$$y = \bar{z} \times (\bar{z} \otimes q_M). \quad (39)$$

Hence the distinguisher takes one convolution and one stream multiplication. Both for the basic distinguisher and the convolutional filter L_d depends strongly on Q_M determined by the table M . In the worst case (for the attacker), M has been chosen such that for any gap h , $Q_M(h) = 2^{-2n}$ or zero. For example if $M = (0, 1, 3, 7)$, the resulting Q_M is 2^{-4} for $h \in \{1, 2, 3, 4, 6, 7\}$. For this kind of M , the simple distinguisher has $L_d = 2^{4n}$ and the convolutional filter has

$$L_d = 1/2^{n-1}(2^n - 1)2^{-4n} \approx 2^{2n+1} . \quad (40)$$

As there are $(2^n - 1)(2^n - 2)/2$ differences among 2^n entries, the choice of such an M is only possible if the length of the source register is in the order of 2^{2n-1} , so L_d is only a factor 4 longer than the source register. For example for a multiplexor choosing from 64 positions the source register must have a length in the order of 2000 bits and L_d is only 8000.

Another interesting case is when $M[i] = i$, i.e., the selection positions are subsequent. For $h \neq 0$ we have $Q_M(h) = (2^n - h)2^{-2n}$. The best simple distinguisher has $h_m = 1$ and yields $L_d = 2^{4n}/(2^n - 1)^2 \approx 2^{2n}$. For the convolutional filter this gives:

$$L_d^{-1} = \sum_{i=1}^{2^n-1} \frac{i^2}{2^{4n}} = 2^{-4n} \sum_{i=1}^{2^n-1} i^2, \quad (41)$$

resulting in

$$L_d = 2^{4n} \frac{6}{(2^n - 1)(2(2^{2n-1} + 1)(2^n))} \approx 3 \cdot 2^n. \quad (42)$$

Hence for a multiplexor selecting from 64 positions this yields $L_d \approx 192$, again only a small factor larger than the minimum size of the source register. Both cases show that for a multiplexor generator with a Fibonacci source register we can construct distinguishers with L_d in the same order of magnitude as the source register.

5.2 Galois configuration

If the source register has a Galois configuration the output sequence cannot be modeled as a simple sampling of the source sequence and the analysis above does not apply. The value of L_d depends on interaction between the selection position table and the feedback polynomial of the source LFSR. Given the weight of the feedback polynomial and characteristics of the selection position table upper bounds for L_d can be formulated. For example, for a 64-bit multiplexor and a source register with an LFSR of weight 17 a distinguisher similar to the one in Equation (39) can be built with L_d below $2^{15}/7 \approx 4700$.

6 Simulation results

We have experimentally verified the correctness of the values L_d for all distinguishers presented in this paper. As $L_d = \langle Y \rangle^{-2}$, it suffices to apply the distinguisher to sequences with length much larger than L_d and see whether Y converges to $\langle y_t \rangle$ in case it matches the generator and to 0 in case of a random sequence. All our experimentally obtained data confirmed the theoretical values.

The function $\rho_g(h)$ used for the simulation is based on the Gaussian approximation for $\mathcal{S}(g, h)$. To avoid the infinite domain of Gaussian variables, it is truncated beyond 5 times the standard deviation below and above the average $\langle h \rangle$. The same truncation is done for $\mathcal{S}(h, g)$, which is used to compute $\mu_g(h)$. With the roles of g and h reversed for $\mu_g(h)$, this results in an asymmetry in the truncation below and above $\langle h \rangle$ in order to preserve the actual shape of $\mu_g(h)$. For the multiplexor generators the exact expression of $Q_M(h)$ is used.

We illustrate our simulation results for a convolutional filter adapted to an SG with a source LFSR governed by the polynomial $p(x) = x^{300} + x^{219} + x^{131} + x^{73} + 1$ taken from [2]. For this polynomial, [2] gives a theoretical estimation of a parameter $N = 2^{30.3}$ where N plays the same role as L_d with $N = 4L_d$. It reports an experiment with 48 successes out of 50 sequences each of length 2^{29} , i.e. a failure rate of 4 %. Our convolutional filter has $\langle Y \rangle \approx 0.000272$ resulting in $L_d \approx 1.35 \times 10^7$. This is a factor 24 smaller than the equivalent in [2]. For sequences of length 2^{29} the expected failure rate of our convolutional filter is below 0.1 %.

For a sequence of L the standard deviation of Y is equal to $1/\sqrt{L}$. Relating that to $\langle Y \rangle$ of the convolutional filter yields $\sigma(Y) = \sqrt{L_d/L} \langle Y \rangle$. Figure 1 shows the convergence of Y to $\langle Y \rangle$ for an output sequence of the target SG and to 0 for the output of a pseudo-random generator based on SHA-1. Figure 2 compares the distribution of the value of Y over a set of 100 sequences of length L_d of the SG with the distributions predicted by the theory.

The experimental implementation uses explicit convolutions and component-wise multiplications. The complexity of the attack is dominated by the convolutions. For the SG and

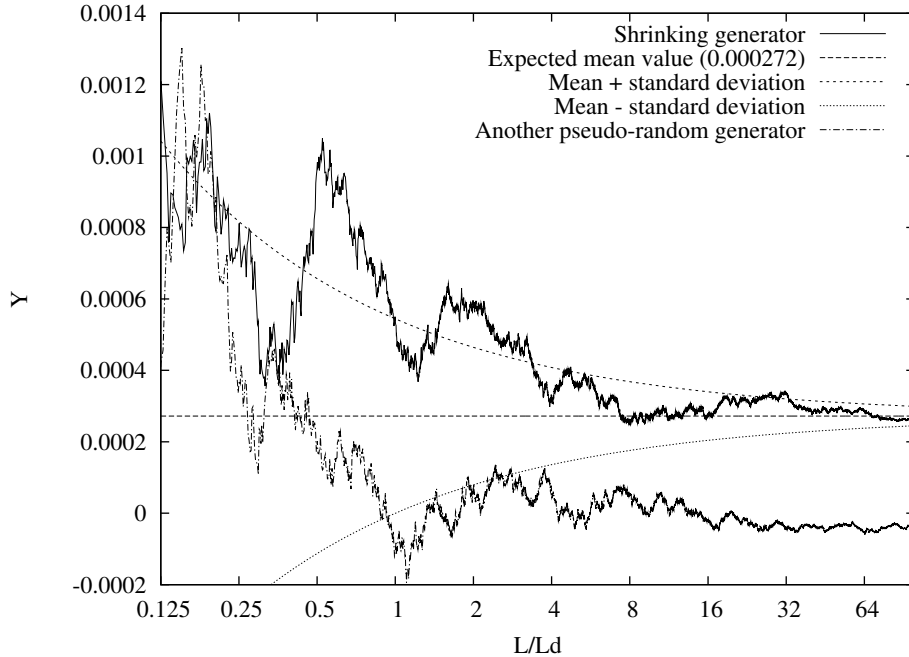


Figure 1: Convergence of Y as a function of L .

ASG, the convolution kernels have width of the order of $\sqrt{g_i}$. The complexity of the attack is thus $O(L_d \times \sum_i \sqrt{g_i})$. For the multiplexor generator with $M[i] = i$, the width of $Q_M(h)$ is of order 2^n . The complexity of this attack is thus $O(L_d^2)$. In this last case, we could make the convolution in the frequency domain using a fast Fourier transform (FFT), decreasing the complexity down to $O(L_d \log L_d)$.

7 Conclusions

Convolutional filters are a new type of distinguisher applicable to shrinking generators, alternating-step generators and multiplexor generators. They are more powerful than existing distinguishers for those generators and their conceptual simplicity allows to predict their probability of success accurately.

References

- [1] A. Canteaut, M. Trabbia, “Improved Fast Correlation Attacks Using Parity-Check Equations of Weight 4 and 5”, *Advances in Cryptology – Eurocrypt 2000*, LNCS 1807, Springer-Verlag, 2000, pp. 573-588.
- [2] P. Ekdahl, W. Meier, T. Johansson, “Predicting the Shrinking Generator with Fixed Connections”, *Advances in Cryptology – Eurocrypt 2003*, LNCS 2656, Springer-Verlag, 2003, pp. 330-344.

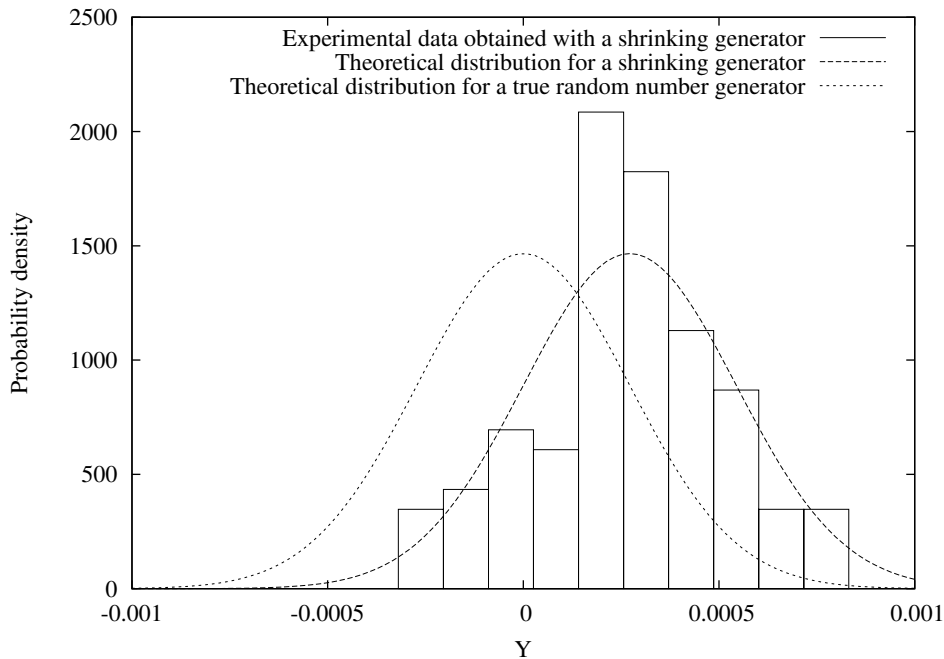


Figure 2: Distribution of Y for 100 sequences of length L_d .

- [3] D. Coppersmith, H. Krawczyk, Y. Mansour, “The Shrinking Generator”, *Advances in Cryptology – Crypto ’93*, LNCS 773, Springer-Verlag, 1994, pp. 22-39.
- [4] J. Dj. Golić, L. O’Connor, “Embedding and probabilistic correlation attacks on clock-controlled shift registers”, *Advances in Cryptology – Eurocrypt ’94*, LNCS 950, Springer-Verlag, 1995, pp. 230-243.
- [5] J. Dj. Golić, “Towards Fast Correlation Attacks on Irregularly Clocked Shift Registers”, *Advances in Cryptology – Eurocrypt ’95*, LNCS 921, Springer-Verlag, 1995, pp. 248-262.
- [6] J. Dj. Golić, “Linear Models for Keystream Generators”, *IEEE Trans. on Computers*, Vol. 45, No 1 January, IEEE Press, 1996, pp. 41-49.
- [7] J. Dj. Golić, “Computation of low-weight parity-check polynomials”, *Electronic Letters*, Vol. 32, No 21 October, 1996.
- [8] J. Dj. Golić, “Correlation analysis of the Shrinking Generator”, *Advances in Cryptology – CRYPTO 2001*, LNCS 2139, Springer-Verlag, 2001, pp. 440-457.
- [9] J. Dj. Golić, “On the Success of the Embedding Attack on the Alternating Step Generator”, *Selected Areas in Cryptography 2003*, 2003, pp. 262-274.
- [10] J. Dj. Golić, R. Menicocci, “A New Statistical Distinguisher for the Shrinking Generator”, *Cryptology ePrint Archive: Report 2003/041*, <http://eprint.iacr.org/2003/041/>
- [11] J. Dj. Golić, R. Menicocci, “Correlation Analysis of the Alternating Step Generator”, *Designs, Codes and Cryptography* 31(1), 2004, pp. 51-74.

- [12] C. G. Günther, “Alternating step generators controlled by de Bruijn sequences”, *Advances in Cryptology – Eurocrypt ’87*, LNCS 304, 1988, pp. 5-14
- [13] S. Jennings, “Multiplexed sequences: Some properties of the minimum polynomial”, *Cryptography – Proceedings of the Workshop on Cryptography, Burg Feuerstein*, LNCS 149, Springer-Verlag, 1983, pp. 189-206.
- [14] T. Johansson, “Reduced complexity correlation attacks on two clock-controlled generators”, *Advances in Cryptology – Asiacrypt ’98*, LNCS 1514, Springer-Verlag, 1998, pp. 342-357.
- [15] T. Johansson, F. Jönsson, “Fast Correlation Attacks Through Reconstruction of Linear Polynomials”, *Advances in Cryptology – CRYPTO 2000*, LNCS 1880, Springer-Verlag, 2000, pp. 300-315.
- [16] A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.
- [17] L. Simpson, J. Dj. Golić, E. Dawson, “A probabilistic correlation attack on the shrinking generator”, *Information Security and Privacy ’98 - Brisbane*, LNCS 1438, Springer-Verlag, 1998, pp. 147-158.
- [18] D. Wagner, “A Generalized Birthday Problem”, *Advances in Cryptology – CRYPTO 2002*, LNCS vol 2442, Springer-Verlag, 2002, pp. 288-303.