Cryptogr. Commun.
DOI 10.1007/s12095-012-0069-3

Correlation attacks on combination generators

Anne Canteaut · María Naya-Plasencia

Received: 22 January 2012 / Accepted: 1 August 2012

Abstract The combination generator is a popular stream cipher construction. It consists of several independent devices working in parallel whose outputs are combined by a Boolean function. The output of this function is the keystream. The security of this generator has been extensively studied in the case where the devices are LFSRs. Some particular cases where the devices are nonlinear have also been studied, most notably the different versions of the eSTREAM proposal named Achterbahn. Several cryptanalysis techniques against these ciphers have been published, extending the classical correlation attack. But each of these attacks has been presented mainly in a very particular scenario. Therefore, this paper aims at generalising these methods to any combination generator in order to be able to compare their respective advantages and to determine the optimal attack for each particular generator. Generic formulas for the data-time-space complexities are then provided, which only depend on the number of devices, their periods and the number of their internal states and of the Boolean combining function. Some of the considered improvements can also be used in a much more general context, which includes linear attacks against some block ciphers.

Keywords Correlation attacks · Stream cipher · NLFSR · Parity-check

Mathematics Subject Classifications (2010) 68P25 · 94A60

A. Canteaut (✉)
INRIA project-team SECRET, B.P. 105, 78153 Le Chesnay cedex, France
e-mail: Anne.Canteaut@inria.fr

M. Naya-Plasencia
Laboratoire PRISM, Université de Versailles St-Quentin-en-Yvelines,
45 avenue des Etats-Unis, 78035 Versailles Cedex, France
e-mail: maria.naya.plasencia@gmail.com

Published online: 21 August 2012

1 Introduction

One of the most popular constructions of stream ciphers is the combination generator. In this model, several independent devices (e.g. feedback shift registers) work in parallel. Their outputs are taken as inputs by a Boolean combining function, and the output of this function provides the keystream bits. The case where the combination generator uses shift registers with a linear feedback function is a very old and well-studied model which has been shown to be vulnerable to several attacks, including correlation attacks [5–7, 25–27, 29, 33, 34, 38], algebraic attacks [9, 10] and distinguishing attacks [3, 22]. Meanwhile, the combination generators composed of LFSRs with unknown feedback polynomials or of shift registers with nonlinear feedbacks appear to be less vulnerable to this type of attacks.

Such a combination generator using nonlinear feedback shift registers (NLFSRs) was submitted in 2005 to the eSTREAM public competition [11], launched by the ECRYPT European network in order to recommend some secure stream ciphers. This keystream generator named Achterbahn was designed by Gammel et al. [13–16]. A version of this algorithm was selected for the second phase of the competition. It was afterwards eliminated due to several attacks presented on its successive versions [20, 21, 28, 35, 36].

However, since each of these attacks has been applied to a different scenario and described in a different paper, it is hard to compare their respective advantages, to determine precisely the scenarios where each variant applies and to decide which is the optimal attack in a given context. This paper then aims at reviewing and generalizing these attacks against combination generators with nonlinear constituent devices in order to include all these variants in a well-defined family and to provide generic formulas for the complexities in data, time and memory, depending on the parameters of the generator. Here, we also want to determine the parameters of the combination generator which make it resistant to the whole family of attacks. Indeed, these attacks only require the knowledge of the periods of the sequences produced by the constituent devices and of the Boolean combining function. The result is that, once we are given such a generator, we will be able to determine in an automatic way, the different trade-offs and then the complexities of the different attacks that can be applied. Therefore, it makes it possible to design such a cipher, with an a priori knowledge of its security level regarding correlation attacks (which are the best known applicable attacks up to date). We also show that some of the techniques used for attacking the combination generator apply to a more general problem which includes for instance the problem to be solved in linear attacks against iterated block ciphers with Matsui's Algorithm 2.

The paper is organised as follows. Section 2 presents the combination generator which will be analysed in detail, as well as a the basic principle of correlation attacks against this generator and the more general problem which must be solved for breaking this type of generator. Then, Section 3 describes a general method for speeding-up most correlation attacks as soon as the considered biased sequence can be decomposed as the sum of two sequences with independent initial states. Section 4 presents another technique introduced by Hell and Johansson [20] which leads to a better trade-off between time and data complexities for solving the general correlation problem. Section 5 then shows that, in a correlation attack against the combination generator, both previous improvements can be applied to detect the

correlation between parity-check relations derived from the generator. It finally compares the different trade-offs achieved by these attacks and discusses which are the best ones to be applied depending on the situation.

2 General model and notation

2.1 The general combination generator

The general keystream generator which is studied throughout the paper is a pseudo-random generator composed of independent binary devices, i.e., each of these devices is a finite-state automaton producing one bit at each time instant. A combination generator based on n such devices consists in combining the outputs of the devices by a Boolean function f of n variables. Then, the output of this Boolean function at each time instant provides the corresponding bit of keystream (Fig. 1).

In the paper, the keystream is denoted by $\boldsymbol{S} = (S(t))_{t\geq 0}$. Moreover, $\mathcal{R}_i$ denotes device i and L_i denotes the number of bits of its internal state. The sequence produced by $\mathcal{R}_i$ is denoted by $\boldsymbol{x}_i = (x_i(t))_{t\geq 0}$. The important and only fact that the attacker needs to known about this sequence is that it is a periodic sequence with period $T_i \leq 2^{L_i}$. In Section 5, for the sake of simplicity, we consider the periods of the n sequences to be coprime. The attacks are also valid if it is not the case, though their complexities could be easily reduced.

The internal states of the devices are usually initialised from the secret key and a public initialisation vector by an initialisation algorithm. In this paper, we will focus on state-recovery attacks only, i.e., we will aim at recovering the initial states of the devices just before starting generating the keystream sequence, leaving the key-recovery problem which highly depends on the properties of each initialisation algorithm. The fact that an attacker is able to recover the initial states of the devices is obviously considered as a major weakness of the cipher on its own, as it implies for instance that the keystream can be reproduced. Also, in all practical cases that we have studied, the state-recovery attack could always be turned into a key-recovery attack.

2.2 Principle of correlation attacks

As for all attacks against synchronous stream ciphers, we will consider the known-plaintext scenario which equivalently means that we assume that some keystream bits are known to the attacker.

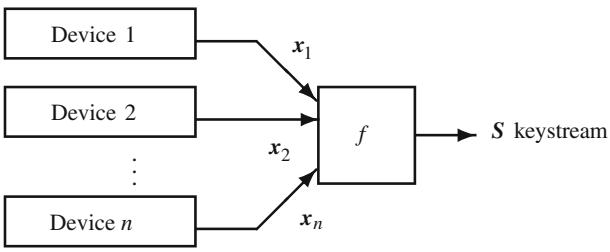

Fig. 1 Keystream generator composed of several independent devices combined by a Boolean function

A generic attack which always applies to this type of cipher is the exhaustive search for the initial states of the n devices. For each possible initial configuration, we can compute the generated keystream and deduce the correct initial state when the sequence produced by the combining function is the same as the observed keystream. Such a generic attack requires at least $2^{\sum_{i=1}^{n} L_i}$ trials. In a well-conceived stream cipher, this time complexity is too large and makes the attack infeasible.

In this context, correlation attacks introduced by Siegenthaler [38] are divide-and-conquer attacks in the sense that they aim at recovering the initial states of some of constituent devices only, independently from the other ones. In other words, they exploit the existence of a smaller generator, composed of a subset of the constituent devices combined by a Boolean function having fewer input variables than f, whose output $\boldsymbol{\sigma} = (\sigma(t))_{t\geq 0}$ is correlated to the keystream produced by the original generator.

If the n-bit vector corresponding to the outputs of the n devices at each time instant is uniformly distributed, the existence of such a small generator is equivalent to the existence of a biased approximation of f depending on fewer variables, in the sense of the following definition.

Definition 1 Let h be a Boolean function with n variables. Then, the bias of h is

$$\mathcal{E}(h) = 2\Pr[h(x) = 0] - 1 = \frac{1}{2^n}\big[\sharp\{x \in \mathbf{F}_2^n, h(x) = 0\} - \sharp\{x \in \mathbf{F}_2^n, h(x) = 1\}\big].$$

Sometimes, this quantity is called the imbalance of h, since the function is said to be balanced if $\mathcal{E}(h) = 0$. It also corresponds to the correlation between h and the all-zero function.

If f and g are two Boolean functions, the correlation between f and g, also named the bias of the approximation of f by g, equals $\mathcal{E}(f + g)$.

In a correlation attack, the lowest number of devices which must be considered simultaneously in the small generator is the lowest integer m such that there exists g, a biased approximation of the Boolean function f on n input variables, which depends on m input variables only. By definition, the smallest m is equal to $R + 1$ where R is the so-called *resiliency order* (aka correlation-immunity order when f is balanced) of the combining function [37]. It is worth noticing that R is upper-bounded by $(n - 1 - \deg f)$ [37], and that the algebraic degree of f cannot be too low for avoiding algebraic attacks for instance. Then, a precomputational step in the attack consists in finding an appropriate biased approximation g of f, which depends on m variables. For $m = R + 1$, which is usually the best choice for the attacker, it is known that the approximation of f with $m = R + 1$ variables which has the highest bias is the linear function corresponding to the sum of all involved variables (possibly with a nonzero constant term) [5]. In the case where it is suitable to consider more than $R + 1$ devices together, then the approximation with the highest bias may be nonlinear, but it can be easily computed by the technique described in [39, Theorem 1]. It is worth noticing that there is no need for maximizing the magnitude of $\mathcal{E}(f + g)$ instead of maximizing $\mathcal{E}(f + g)$ when there is no restriction on the value of $g(0)$. Indeed, for any approximation g, we have $\mathcal{E}(f + (g \oplus 1)) = -\mathcal{E}(f + g)$. It follows that we can assume that $\mathcal{E}(f + g)$ is always positive when g is a good approximation of f.

Once an appropriate approximation has been found, the attack consists in recovering the correct initialisations of the m targeted devices. The simplest method is the one originally proposed by Siegenthaler [38]: it performs an exhaustive search for the initial state of the small generator, i.e., for the initial states of the m targeted devices. For each initialisation, the attacker produces N bits of the output $\boldsymbol{\sigma}$ and computes the correlation between these N bits and the corresponding keystream bits, i.e.,

$$\sum_{t=0}^{N-1}(-1)^{S(t)+\sigma(t)} .$$

The correct initialisation is then expected to be the one maximizing the correlation, or to lead to a correlation higher than some appropriate threshold.

Therefore, this attack is a particular instance of the following more general problem, which will be referred to as the *general correlation problem*. Let $\boldsymbol{z} = (z(t))_{t\geq 0}$ be a binary sequence depending on a secret parameter K (e.g. a part of the initial state) which can take 2^k different values. Assume that, for any t,

$$\Pr[z(t) = 0] = \begin{cases} \frac{1}{2}(1+\varepsilon) & \text{if } K = K^\star \\ \frac{1}{2} & \text{otherwise} \end{cases}$$

where $K^\star$ is a given unknown value and $\varepsilon > 0$. The problem is then to recover $K^\star$, i.e., to find the initial state which maximizes

$$\sum_{t=0}^{N-1}(-1)^{z(t)} .$$

It is worth noticing that linear attacks against iterated block ciphers with Matsui's Algorithm 2 [32] are faced with the same problem where $\boldsymbol{z}$ consists of several evaluations of a linear relation between the plaintext and the ciphertext.

In the following, we present two techniques for solving the general correlation problem when the sequence $\boldsymbol{z}$ can be decomposed as a sum of several sequences with independent initial states and with periods less than N. Section 3 describes a general algorithm which reduces the time complexity of the attack with a similar data complexity. This algorithm has been used in the particular case of the attack against Achterbahn 80/128 proposed by Naya-Plasencia [35]. The second improvement described in Section 4 has been first proposed by Hell and Johansson [20]. It consists in using decimated sequences in order to achieve a better trade-off between time and data complexity. Finally, Section 5 shows that both improvements can be used for computing the correlation between so-called parity-check relations in the particular context of a correlation attack against a combination generator. The obtained attacks apply to any such generator once the periods of the n constituent sequences are known, as well as the (possibly strong) Boolean combining function f. In general, we will see that the main weaknesses of this generator might originate from two possible facts: the fact that the periods of the sequences produced by the devices are too small (even though the number of devices, n, is big), and a weak combining function.

3 Speeding-up the general correlation attack

3.1 Basic algorithm

The basic technique for solving the general correlation problem consists in performing an exhaustive search for the secret initial state of the sequence $\boldsymbol{z}$ and in applying a hypothesis test to recover the correct initialisation. The optimal hypothesis test is defined by the Neyman–Pearson lemma which compares the value of the correlation to an appropriate threshold. All initial state candidates can also be sorted as specified by the Neyman–Pearson lemma, and then they can be tested in order of probability [30]. It is known, for instance from [22, Section 4.1], that the number of samples needed to determine the correct initialisation out of 2^k possible values is then

$$N \simeq \frac{2k \ln 2}{\varepsilon^2},$$

where ε denotes the bias of the sequence $\boldsymbol{z}$. The time complexity is then $N2^k$.

In the particular case where $\boldsymbol{z}$ depends affinely on its k-bit initial state K, i.e., if there exists a sequence $(y(t))_{t\geq 0}$ independent from K such that

$$z(t) = \alpha_t \cdot K \oplus y(t), \ \forall t \geq 0 \ ,$$

the time complexity can be reduced by using a FFT technique as proposed for instance in [7, 31]. Indeed, for any possible initial state K, we have

$$\mathcal{Z}(K) = \sum_{t=0}^{N-1} (-1)^{z(t)} = \sum_{t=0}^{N-1} (-1)^{\alpha_t \cdot K \oplus y(t)} = \sum_{x \in \mathbf{F}_2^k} F(x)(-1)^{x \cdot K}$$

where F is a function from $\mathbf{F}_2^k$ into $\mathbb{Z}$ defined by

$$F(x) = \sum_{t<N,\ \alpha_t = x} (-1)^{y(t)} \ ,$$

where $F(x) = 0$ when x does not correspond to any $\alpha_t, 0 \leq t < N$. The set of all $\mathcal{Z}(K)$, $K \in \mathbf{F}_2^k$, can then be obtained by computing the fast Walsh–Hadamard transform of F with complexity $\mathcal{O}(k2^k)$.

This technique leads to an attack against the combination generator with m targeted devices, $\mathcal{R}_{i_1}, \ldots, \mathcal{R}_{i_m}$. Throughout the paper, we use the same notation as in Fig. 2: a combination generator with n constituent devices is considered. Its combining function is denoted by f and depends on n variables. The attack then uses an approximation g of f, which depends on m variables only. In this context, the data complexity of the attack is then

$$\frac{2 \ln 2 \sum_{j=1}^{m} L_{i_j}}{\varepsilon^2} \ ,$$

where $\varepsilon = \mathcal{E}(f + g)$ and the time complexity with the basic algorithm is

$$\frac{2 \ln 2 \sum_{j=1}^{m} L_{i_j}}{\varepsilon^2} \times 2^{\sum_{j=1}^{m} L_{i_j}} \ .$$

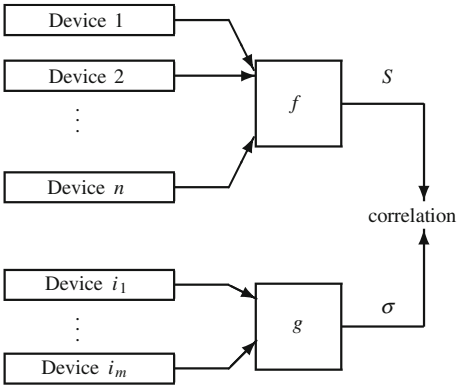

Fig. 2 Principle of correlation attacks against the combination generator

But, the FFT technique may apply, in particular when the targeted devices are LFSRs and when the best approximation g of f is linear, which is always the case when the small generator involves the smallest possible number of devices, $m = R + 1$, where R is the resiliency order of f. Then, the sequence $\boldsymbol{\sigma}$ produced by the small generator can be expressed as

$$\sigma(t) = \bigoplus_{j=1}^{m} \left(\alpha_{j,t} \cdot K_{i_j}\right) = (\alpha_{1,t}, \ldots, \alpha_{m,t}) \cdot (K_{i_1}, \ldots, K_{i_m}) \ ,$$

where K_{i_j} denotes the initial state of register $\mathcal{R}_{i_j}$. The time complexity for recovering the initial states of these m registers is then reduced to

$$\sum_{j=1}^{m} L_{i_j} 2^{\sum_{j=1}^{m} L_{i_j}} + \frac{2 \ln 2 \sum_{j=1}^{m} L_{i_j}}{\varepsilon^2} \ ,$$

where the first term corresponds to the FFT computation and the second one to the computation of the values of F. Therefore, the complexity of the general algorithm is divided by $\frac{2 \ln 2}{\varepsilon^2}$.

3.2 Speeding-up the exhaustive search in the general case

The previously described FFT technique only applies to the case where $\boldsymbol{z}$ depends affinely on its initial state, i.e., in the case of the combination generator, when the small generator has a linear next-state function. In particular, it cannot be used when the constituent devices are nonlinear devices as in the case of Achterbahn. However, we now present a method which leads to a similar speed-up when the combining function of the small generator can be decomposed as a sum of two functions with disjoint input variables, i.e., $g(x_{i_1}, \ldots, x_{i_m}) = g_u(x_{i_1}, \ldots, x_{i_{m'}}) + g_v(x_{i_{m'+1}}, \ldots, x_{i_m})$. This obviously occurs when the small generator corresponds to the sum of m devices, which is the practical situation when the combining function of the keystream generator has an appropriate biased linear approximation, in particular when the number of targeted devices is minimal. This technique has been introduced by [35] in the particular context of the cryptanalysis of Achterbahn, and here we provide a more general description.

More generally, our technique can be used for the general correlation problem, in order to determine among the 2^k initial states the one which maximizes

$$\sum_{t=0}^{N-1}(-1)^{z(t)} ,$$

as soon as this sequence can be expressed as the sum (modulo 2) of two sequences, $\boldsymbol{u}$ and $\boldsymbol{v}$ with independent initial states and such that the period T_u of $\boldsymbol{u}$ is known and smaller than N. The main idea of the algorithm is then to gather some computations which depend on $\boldsymbol{u}$ only, and then to exploit the fact that the period of $\boldsymbol{u}$ is smaller than N in order to compute those terms T_u times only, instead of N times.

In the case of an attack against the combination generator, the sequence $\boldsymbol{z} = \boldsymbol{S} \oplus \boldsymbol{\sigma}$ has such a decomposition if the approximation g of f can be decomposed as $g(x_{i_1}, \ldots, x_{i_m}) = g_u(x_{i_1}, \ldots, x_{i_{m'}}) + g_v(x_{i_{m'+1}}, \ldots, x_{i_m})$. Then, $\boldsymbol{u}$ corresponds to the combination by g_u of the outputs of m' devices among the m targeted devices, e.g.,

$$\boldsymbol{u} = g_u(x_{i_1}(t), \ldots, x_{i_{m'}}(t)) .$$

Indeed, it is known that

$$T_u = \prod_{j=1}^{m'} T_{i_j}$$

is a period of $\boldsymbol{u}$. In this case, we have

$$\boldsymbol{v} = \left(S(t) \oplus g_v(x_{i_{m'+1}}(t), \ldots, x_{i_m}(t))\right)_{t\geq 0} .$$

We now compute the correlation between $\boldsymbol{u}$ and $\boldsymbol{v}$. As previously explained, detecting this bias requires the knowledge of a number of samples N which exceeds

$$N_0 = \frac{2k \ln 2}{\varepsilon^2} . \tag{1}$$

Then, for each of the 2^k possible initial states of $\boldsymbol{z}$, we compute the sum $\sum_{t=0}^{N-1} z(t)$ on N bits where N is the smallest multiple of T_u which exceeds N_0, i.e.,

$$N = T_u \left\lceil \frac{N_0}{T_u} \right\rceil .$$

We denote $c = \frac{N}{T_u}$. Then, the previous sum can be rewritten in the following way:

$$\begin{aligned}
\sum_{t=0}^{N-1} z(t) &= \sum_{t=0}^{N-1} u(t) \oplus v(t) \\
&= \sum_{r=0}^{T_u-1} \sum_{q=0}^{c-1} u(qT_u + r) \oplus v(qT_u + r) \\
&= \sum_{r=0}^{T_u-1} \sum_{q=0}^{c-1} u(r) \oplus v(qT_u + r)
\end{aligned}$$

since T_u is a period of $\boldsymbol{u}$. Now, we use the fact that, for any pair (q, r),

$$u(r) \oplus v(qT_u + r) = \begin{cases} v(qT_u + r) & \text{if } u(r) = 0 \\ 1 - v(qT_u + r) & \text{if } u(r) = 1 \, . \end{cases}$$

Therefore, we deduce that

$$\sum_{t=0}^{N-1} z(t) = \sum_{r=0}^{T_u-1} (u(r) \oplus 1) \left(\sum_{q=0}^{c-1} v(qT_u + r) \right) + \sum_{r=0}^{T_u-1} u(r) \left(c - \sum_{q=0}^{c-1} v(qT_u + r) \right) \, .$$

For any r, $0 \leq r < T_u$, we set

$$V(r) = \sum_{q=0}^{c-1} v(qT_u + r) \, .$$

The vector $(V(0), \ldots, V(T_u - 1))$ consists of T_u integers which depend on the initial state of $\boldsymbol{v}$ only (i.e., the initial states of the $(m - m')$ corresponding devices in the context of an attack against the combination generator). Then, this vector can be computed independently from the initial state of $\boldsymbol{u}$. We now have

$$\begin{aligned} \sum_{t=0}^{N-1} z(t) &= \sum_{r=0}^{T_u-1} (u(r) \oplus 1) \, V(r) + \sum_{r=0}^{T_u-1} u(r) \, (c - V(r)) \\ &= \sum_{r=0}^{T_u-1} (-1)^{u(r)} \left(V(r) - \frac{c}{2} \right) + \frac{T_u c}{2} \, . \end{aligned}$$

It follows that

$$\sum_{t=0}^{N-1} (-1)^{z(t)} = N - 2 \sum_{t=0}^{N-1} z(t) = \sum_{r=0}^{T_u-1} (-1)^{u(r)} \, (c - 2V(r)) \, .$$

Now, we need to compute this sum for each initial state of $\boldsymbol{u}$ and find if there is one that provides the expected bias. However, starting from a particular initial state U_0, we can find the initial state U_τ of $\boldsymbol{u}$ which maximizes this sum within the same cycle as U_0. Indeed, let U_τ denote the internal state of the generator producing $\boldsymbol{u}$ at time τ starting from U_0, for $0 \leq \tau < T_u$. Then, the sequence generated from U_τ is exactly the sequence derived from the sequence generated from U_0 shifted by τ positions, i.e., $(u(t + \tau \bmod T_u))_{0 \leq t < T_u}$. Then, starting from a particular initial state U_0 of $\boldsymbol{u}$, we want to find the value of τ which maximizes the value of

$$\sum_{r=0}^{T_u-1} (-1)^{u(r+\tau \bmod T_u)} \left(V(r) - \frac{c}{2} \right) + \frac{T_u c}{2} \quad \text{for } 0 \leq \tau < T_u$$

which corresponds to the cross-correlation between two sequences, $\boldsymbol{u}$ and $(V(0), \ldots, V(T_u - 1))$, of length T_u. This can be done efficiently with a fast Fourier transform [2, pages 306–312] with complexity $\mathcal{O}(T_u \log T_u)$. Once the maximum of the cross-correlation has been computed for a given initial state of $\boldsymbol{u}$, we have to

repeat this procedure for another value U_0 in a different cycle of $\boldsymbol{u}$. Once all initial states for $\boldsymbol{u}$ have been examined, the same algorithm has to be repeated for another initial state of $\boldsymbol{v}$. The algorithm is described in Algorithm 1 for the particular case of a correlation attack against the combination generator where

$$u(t) = g_u(x_{i_1}(t), \ldots, x_{i_{m'}}(t)) \text{ and } v(t) = S(t) \oplus g_v(x_{i_{m'+1}}(t), \ldots, x_{i_m}(t)) \ .$$

Algorithm 1 Correlation attack against the general combination generator with a speed-up of the exhaustive search for m targeted devices.

for each initial state of the last $(m - m')$ devices **do**
  for r from 0 to $T_u - 1$ **do**
    $V(r) \leftarrow c - 2\sum_{q=0}^{c-1} v(qT_u + r)$
  end for
  repeat
    choose an initial state for the first m' devices which does not belong to a previously examined cycle
    for r from 0 to $T_u - 1$ **do**
      $u(r) \leftarrow g_u(x_{i_1}(r), \ldots, x_{i_{m'}})(r)$
    end for
    Compute the following cross-correlation with an FFT

$$\mathcal{S}(\tau) = \sum_{r=0}^{T_u-1} (-1)^{u(r+\tau \bmod T_u)} V(r), \quad 0 \leq \tau < T_u$$

    if $\mathcal{S}(\tau) >$ threshold for some τ **then**
      return the initial states of the last $(m - m')$ devices and the internal states of the first m' devices after τ clocks.
    end if
  until all initial states of the first m' devices have been examined
end for

Let 2^{k_u} denote the number of possible initial states for $\boldsymbol{u}$. In most practical situations, the number of cycles of $\boldsymbol{u}$ is roughly $2^{k_u}/T_u$ since it is expected that the constituent devices of the generator do not have any short cycles. For instance, in the case of LFSRs with primitive feedback polynomials, coprime periods and with nonzero initial states, the number of possible initial states for $\boldsymbol{u}$ is equal to the period $T_u = \prod_{j=1}^{m'} T_{i_j}$, since it is assumed that the all-zero initial state is avoided for any of the LFSRs. The situation is similar in Achterbahn because all constituent devices are NLFSRs with period $T_i = 2^{L_i} - 1$ for all i. Then, the overall time complexity of the attack is given by the following formula:

$$2^{k-k_u}\left[T_u c + \frac{2^{k_u}}{T_u}(T_u \log T_u)\right] \ .$$

We point out here that even if the cross-correlation is not computed with an FFT, but for all the possible values of τ, the final time complexity would still be reduced compared to the classical attack, as this changes the second term in the previous sum

to $2^{k_u} T_u$, which divides the complexity of the classical attack by a factor $c = N/T_u$. In the situations where the generator producing $\boldsymbol{u}$ has only a few cycles, T_u is close to 2^{k_u} as previously explained. The time complexity can then be expressed as

$$2^k \left[\frac{N_0}{T_u} + \log T_u \right] ,$$

where N_0 is the data complexity, i.e., the minimal number of samples needed for the correlation attack as expressed by (1). Then, the optimal choice for the decomposition of the small generator into $\boldsymbol{u} \oplus \boldsymbol{v}$ corresponds to the situation where T_u is close to $\frac{N_0}{\log N_0}$. In this case, the time complexity is proportional to

$$2^k \log N_0 .$$

This must be compared to $2^k N_0$, which is the time complexity of the classical correlation attack described in the previous section, while the data complexity is unchanged. Let us notice here that instead of considering N, which is a multiple of T_u, we could consider N_0 samples only, as done in [36] as N was limited. This case is similar but the previous sum has to be decomposed into two parts. For the sake of simplicity, we have presented here the case where N is a multiple of T_u, as it usually does not increase the complexity and as the extension is quite straightforward.

The memory requirement corresponds to the storage of both sequences $\boldsymbol{u}$ and $\boldsymbol{V}$ which are composed of T_u bits and of T_u integers respectively.

In the case of an attack against the combination generator as described by Algorithm 1, the time complexity can then be expressed as

$$2^{\sum_{j=m'+1}^{m} L_{i_j}} \left[T_u c + \frac{2^{\sum_{j=1}^{m'} L_{i_j}}}{T_u} (T_u \log T_u) \right] ,$$

where $T_u = \prod_{j=1}^{m'} T_{i_j}$.

4 Correlation attacks using decimation

4.1 Basic principle

Another trade-off between the data and time complexity in correlation attacks can be achieved with a method introduced by Hell and Johansson in [20]. The idea is to perform the correlation attack on a decimated version of the sequence in order to reduce the size of the initial state for which we have to perform an exhaustive search. First, we give a general description of the algorithm presented by Hell and Johansson. Then, we show in Section 4.2 how it can be improved. The technique proposed in [20] applies for determining, among 2^k different initial states of the sequence $\boldsymbol{z}$, the one which maximizes

$$\sum_{t=0}^{N-1} (-1)^{z(t)}$$

as soon as $\boldsymbol{z}$ can be expressed as the sum of two sequences $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$ with independent initial states and such that a period T_d of $\boldsymbol{\gamma}$ is known and smaller than N. Then, for any δ, we have

$$z(tT_d + \delta) = \alpha(tT_d + \delta) \oplus \gamma(tT_d + \delta) = \alpha(tT_d + \delta) \oplus \gamma(\delta) \ .$$

Then, we deduce that

$$\sum_{t=0}^{N-1} (-1)^{z(tT_d+\delta)} = (-1)^{\gamma(\delta)} \sum_{t=0}^{N-1} (-1)^{D_\delta(t)} \ ,$$

where $\boldsymbol{d}_\delta = (D_\delta(t))_{t\geq 0}$ denotes the decimated sequence $(\alpha(tT_d + \delta))_{t\geq 0}$. Therefore, if $\boldsymbol{d}_\delta$ can be computed by the attacker for some δ (e.g. $\delta = 0$), finding the maximal $\sum_t (-1)^{z(t)}$ for N well-chosen values of t boils down to finding the highest magnitude for $\sum_{t=0}^{N-1} (-1)^{D_\delta(t)}$. This can be done by an exhaustive search for the initial state of $\boldsymbol{d}_\delta$. Let 2^{k_d} denote the number of initial states for $\boldsymbol{\alpha}$. Then, $k_d < k$ as $\boldsymbol{\alpha}$ does not depend on the initialisation of $\boldsymbol{\gamma}$. The attack then requires

$$N = \frac{2k_d \ln 2}{\varepsilon^2}$$

evaluations of the decimated sequence, implying that the data complexity is now

$$\frac{2k_d \ln 2 T_d}{\varepsilon^2}$$

which is usually higher than the data complexity without decimation given by (1), but the time complexity is

$$2^{k_d} N = \frac{2k_d \ln 2}{\varepsilon^2} \times 2^{k_d}$$

which improves the usual algorithm in most cases.

This improvement applies in the context of a correlation attack against the combination generator if the approximation g can be decomposed as a sum of two functions with disjoint input variables:

$$g(x_{i_1}, \ldots, x_{i_m}) = g_d(x_{i_1}, \ldots, x_{i_\partial}) + g'(x_{i_{\partial+1}}, \ldots, x_{i_m}) \ .$$

We then decimate the sequence $\boldsymbol{z} = (\boldsymbol{S} \oplus \boldsymbol{\sigma})$ by the product of the periods of the first ∂ devices among $\mathcal{R}_{i_1}, \ldots, \mathcal{R}_{i_\partial}$:

$$T_d = \prod_{j=1}^{\partial} T_{i_j} \ .$$

Then, for any $t \geq 0$ and any δ, we have

$$\begin{aligned}
\sigma(tT_d + \delta) &= g'(x_{i_{\partial+1}}(tT_d + \delta), \ldots, x_{i_m}(tT_d + \delta)) \oplus g_d(x_{i_1}(tT_d + \delta), \ldots, x_{i_\partial}(tT_d + \delta)) \\
&= g'(x_{i_{\partial+1}}(tT_d + \delta), \ldots, x_{i_m}(tT_d + \delta)) \oplus g_d(x_{i_1}(\delta), \ldots, x_{i_\partial}(\delta)) \\
&= g'(x_{i_{\partial+1}}(tT_d + \delta), \ldots, x_{i_m}(tT_d + \delta)) \oplus \gamma(\delta) \ .
\end{aligned}$$

Since the last term in the sum is a constant, the decimated sequence $(\boldsymbol{S} \oplus \boldsymbol{\sigma})$ can be computed (up to a constant) from the initial states of $(m - \partial)$ devices only, instead of

m. The correlation attack is then similar as before, except that we compute the correlation between decimated versions of $\boldsymbol{S}$ and $\boldsymbol{\alpha} = (g'(x_{i_{\partial+1}}(t), \ldots, x_{i_m}(t)))_{t\geq 0}$. Then, the magnitude of the correlation, instead of its signed value, must be maximized.

The correct initial states of the last $(m - \partial)$ devices can then be recovered with

$$N \simeq \frac{2 \ln 2 \sum_{j=\partial+1}^{m} L_{i_j}}{\varepsilon^2} \text{ samples },$$

leading to the following data complexity

$$\frac{2 \ln 2 \sum_{j=\partial+1}^{m} L_{i_j} \prod_{j=1}^{\partial} T_{i_j}}{\varepsilon^2} . \tag{2}$$

The time complexity is now

$$N \times 2^{\sum_{j=\partial+1}^{m} L_{i_j}} = \frac{2 \ln 2 \sum_{j=\partial+1}^{m} L_{i_j}}{\varepsilon^2} \times 2^{\sum_{j=\partial+1}^{m} L_{i_j}} \tag{3}$$

for the basic algorithm. When $m - \partial > 1$, the technique presented in Section 3.2 for speeding-up the exhaustive search can be applied if g' can again be decomposed as the sum of two functions with disjoint variables, leading to the following time complexity

$$2^{\sum_{j=\partial+1}^{m} L_{i_j}} \left[\frac{2 \ln 2 \sum_{j=\partial+1}^{m} L_{i_j}}{T_u \varepsilon^2} + \log T_u \right]$$

where $T_u = \prod_{j=m-m'+1}^{m} T_{i_j}$ for some $m' \geq 0$.

4.2 Improving data complexity: using several parallel decimated sequences

Decimation usually increases the data complexity of the attack, and this might be a bottleneck, for instance when the number of keystream bits produced from a single initial state is limited. Moreover, it clearly appears that the attack does not exploit this high amount of keystream bits in an optimal way since the data complexity is usually much higher than the number of keystream bits used in the attack. Therefore, we may expect to find a different trade-off between data and time complexities when a decimated sequence is used.

The general problem is to determine the initial state which provides the highest value of

$$\sum_{t=0}^{N-1} (-1)^{z(tT_d+\delta)} = (-1)^{\gamma(\delta)} \sum_{t=0}^{N-1} (-1)^{D_\delta(t)} ,$$

by exploiting the fact that $\boldsymbol{d}_\delta$ depends on 2^{k_d} initial states only. As done in [36], instead of computing this sum for a single value of δ, we rather compute a vector of Δ integers,

$$(\mathcal{D}(\delta),\ 0 \leq \delta < \Delta) \text{ where } \mathcal{D}(\delta) = \sum_{t=0}^{N'-1} (-1)^{D_\delta(t)} ,$$

but for a smaller number N' of samples in each component of the vector. We are now faced with the same situation as in a linear attack using Matsui's algorithm 2 with

several independent approximations [1, 17, 18, 24]. The attacker computes 2^{k_d} vectors with independent components where each component follows the binomial distribution with parameters N' and $\frac{1}{2}(1 \pm \varepsilon)$ for the correct initial state, and with parameters N' and $\frac{1}{2}$ otherwise. Since the sign of the bias of each $\mathcal{D}(\delta)$ is unknown, we have to perform an exhaustive search for this sign and compare the empirical probability distribution for the vector $((-1)^{D_\delta(t)})_{0 \le \delta < \Delta}$ with the theoretical distribution

$$p_b(x) = 2^{-\Delta} \prod_{0 \le \delta < \Delta} \left(1 + (-1)^{b_\delta \oplus x_\delta} \varepsilon\right), \ x \in \mathbf{F}_2^\Delta$$

for all Δ-bit vectors b. Several statistical tests have been proposed for comparing both distributions [1, 23]. For instance, it has been proposed in [1] to sort the possible initial states depending on the value of

$$\min_{b \in \mathbf{F}_2^\Delta} \sum_{\delta=0}^{\Delta-1} \left(\mathcal{D}(\delta) - (-1)^{b_\delta} \varepsilon\right)^2 .$$

Once the values of $\mathcal{D}(\delta)$ have been computed, the additional time complexity of the algorithm is then $\Delta 2^\Delta$ for each of the 2^{k_d} initial states. The overall time complexity is then

$$2^{k_d}(\Delta N' + \Delta 2^\Delta) ,$$

implying that the overhead is usually negligible compared to the algorithm with a single decimated sequence. This complexity can be improved if the Δ correlations $\mathcal{D}(\delta)$ are computed with the faster algorithm described in Section 3.

Then, it has been proved in [17, Proposition 3.1] that, if $\Delta\varepsilon^2 \ll 1$, the number of samples N' required for determining the correct candidate with the same error probability as for the classical attack is

$$N' = \frac{2k_d \ln 2}{\Delta\varepsilon^2} .$$

In other words, the number of required samples is divided by the number of decimated sequences which are considered. The data complexity then decreases to

$$N'T_d + \Delta = \frac{2k_d \ln 2 T_d}{\Delta\varepsilon^2} + \Delta$$

while the time complexity, equal to

$$2^{k_d}(\Delta N' + \Delta 2^\Delta) = \left(\frac{2 \ln 2 k_d}{\varepsilon^2} + \Delta 2^\Delta\right) \times 2^{k_d} ,$$

has a negligible overhead. The general algorithm combining the decimation technique and the speeding-up method described in Section 3 then applies when $\boldsymbol{z} = \boldsymbol{u} \oplus \boldsymbol{v} \oplus \boldsymbol{\gamma}$ where these three sequences have independent initial states of respective sizes k_u, k_v and k_γ and where $\boldsymbol{u}$ and $\boldsymbol{\gamma}$ are periodic with respective periods T_u and T_d. The algorithm is then described in Algorithm 2 when N' is the number of needed samples for each decimated sequence, i.e.

$$N' = \frac{2(k_u + k_v) \ln 2}{\Delta\varepsilon^2} .$$

Algorithm 2 Correlation attack using several decimated sequences.

for each initial state of $\boldsymbol{v}$ **do**
  for δ from 0 to $\Delta - 1$ **do**
    for r from 0 to $T_u - 1$ **do**
      $V_\delta(r) \leftarrow \frac{N'}{T_u} - 2\sum_{q=0}^{N'/T_u-1} v((qT_u + r)T_d + \delta)$
    end for
  end for
  repeat
    choose an initial state of $\boldsymbol{u}$ which does not belong to a previously examined cycle
    for δ from 0 to $\Delta - 1$ **do**
      Compute the following cross-correlation with an FFT

$$\mathcal{D}_\delta(\tau) = \sum_{r=0}^{T_u-1} (-1)^{u((r+\tau \bmod T_u)T_d+\delta)} V_\delta(r), \quad 0 \le \tau < T_u$$

    end for
    for τ from 0 to $T_u - 1$ **do**
      Compute

$$\min_{b \in \mathbf{F}_2^\Delta} \sum_{\delta=0}^{\Delta-1} \left(\mathcal{D}_\delta(\tau) - (-1)^{b_\delta}\varepsilon\right)^2$$

      if $\mathcal{S}(\tau) >$ threshold **then**
        return the initial state of $\boldsymbol{v}$ and the internal state of $\boldsymbol{u}$ after τ clocks.
      end if
    end for
  until all initial states of $\boldsymbol{u}$ have been examined
end for

The corresponding data complexity is then

$$T_d N' + \Delta = \frac{2(k_u + k_v)T_d \ln 2}{\Delta\varepsilon^2} + \Delta$$

and the time complexity is

$$2^{k_u+k_v}\left(\frac{\Delta N'}{T_u} + \Delta \log T_u + \Delta 2^\Delta\right).$$

Example In the attack against Achterbahn-80 [14], we have to find the initial state of a sequence derived from the keystream, of the form

$$\boldsymbol{S} = f(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_6, \boldsymbol{x}_8, \boldsymbol{x}_9, \boldsymbol{x}_{10}, \boldsymbol{x}_{11})$$

where each $\boldsymbol{x}_i$ is the output of a nonlinear device of length $L_i = 21 + i$ and with period $2^{L_i} - 1$. In [36], Naya Plasencia used an approximation of $\boldsymbol{S}$ of the form $\boldsymbol{\sigma} = \boldsymbol{x}_1 \oplus \boldsymbol{x}_6 \oplus \boldsymbol{x}_{10}$ which satisfies $\mathcal{E}(\boldsymbol{S} \oplus \boldsymbol{\sigma}) = 2^{-24}$. Then, we need to find the initial state of $\boldsymbol{z} = \boldsymbol{S} \oplus \boldsymbol{\sigma}$, and we decompose it as $\boldsymbol{z} = \boldsymbol{u} \oplus \boldsymbol{v} \oplus \boldsymbol{\gamma}$ with $\boldsymbol{\gamma} = \boldsymbol{x}_1$ which has

period $T_d = 2^{22} - 1$, $\boldsymbol{u} = \boldsymbol{x}_6$ which has $2^{k_u} = 2^{27}$ initial states and period $T_u = 2^{27} - 1$, and $\boldsymbol{v} = \boldsymbol{S} \oplus \boldsymbol{x}_{10}$. Since the keystream length for a given initial state is limited to 2^{52}, we can apply the previous algorithm with $\Delta = 4$. From the previous formulae, we have that each of the four decimated sequences needs to be evaluated in $N' = 2^{28.3}$ positions, implying that the data complexity is $2^{50.3}$. The time complexity is 2^{65}.

5 Correlation attacks with parity-check relations

A correlation attack can be seen as a decoding problem where the initial state of $\boldsymbol{z}$ is recovered by an ML-decoding algorithm. Since the time complexity of ML-decoding is usually too high, a well-known strategy for decoding linear codes consists in exploiting parity-check relations, i.e., linear relations between some bits of the codewords, especially sparse parity-check relations which usually make the decoding much faster. The price to pay for this is that the decoding is less efficient in the sense that more redundancy is needed. In other words, the data complexity increases. This idea has been introduced by Meier and Staffelbach and is at the origin of the so-called *fast correlation attacks* against LFSR-based generators [34]. Actually, in the case of the combination generator based on LFSRs, when the small generator producing $\boldsymbol{\sigma}$ is linear, many sparse parity-check relations for $\boldsymbol{\sigma}$ can be derived from the LFSR feedback polynomials or from their sparse multiples. This high number of relations then allows the attacker to recover its initial state. Many variants of this attack have been proposed, e.g. [5–7, 25–27, 34]. When the constituent devices are nonlinear, the number of parity-check relations is much smaller, implying that this type of attack would require a huge data complexity. A small number of linear relations can nevertheless be exploited in a distinguishing attack, as proposed by [8, 12]. The following two sections describe how such an attack can be performed against the general combination generator, by using some relations derived from the periods of the devices as first proposed in [28]. It can also be combined with an exhaustive search for the initial state of some of the devices in order to eliminate the influence of a part of the constituent devices and then reduce the time complexity. Then, we show in Section 5.3 that combining those two attacks leads to key-recovery with a good trade-off between time and data complexities.

5.1 Parity-check relations

A *parity-check relation* for a binary sequence $\boldsymbol{z} = (z(t))_{t \geq 0}$ is a linear relation between some bits of $\boldsymbol{z}$ at different instants $(t + \tau)$ where τ varies in a fixed set $\mathcal{T}$ of integers, and t takes any value:

$$\bigoplus_{\tau \in \mathcal{T}} z(t + \tau) = 0, \quad \forall t \geq 0.$$

For instance, the indexes τ corresponding to the nonzero coefficients of the characteristic polynomial of a linear recurring sequence provide a parity-check relation. A two-term parity-check relation,

$$z(t) \oplus z(t + \tau) = 0, \quad \forall t \geq 0,$$

obviously means that τ is a period of the sequence.

In the case of the combination generator, if $\boldsymbol{\sigma}$ is produced by combining devices $i_1, \ldots, i_m$ by some function g, a two-term parity-check relation for $\boldsymbol{\sigma}$ is given by

$$\sigma(t) \oplus \sigma\left(t + \prod_{j=1}^{m} T_{i_j}\right) = 0, \quad \forall t \geq 0,$$

but it can only be used if the attacker has access to keystream bits at distance $\prod_{j=1}^{m} T_{i_j}$ from each other, which is usually impossible. Then, Johansson et al. [28] have suggested to reduce the degree of the relation, i.e., the highest distance between two involved positions, by increasing the number of terms, as shown by the following simple proposition.

Proposition 1 *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be n sequences with periods $T_1, \ldots, T_n$. We denote*

$$\mathcal{T} = \langle T_1, \ldots, T_n \rangle = \left\{ \sum_{i=1}^{n} c_i T_i, \quad c_i \in \{0, 1\} \right\} .$$

Then, the binary sequence $\boldsymbol{x}$ defined by

$$x(t) = \bigoplus_{i=1}^{n} x_i(t)$$

satisfies

$$\bigoplus_{\tau \in \mathcal{T}} x(t + \tau) = 0, \quad \forall t \geq 0.$$

Proof We can prove that the influence of each sequence $\boldsymbol{x}_j$, $1 \leq j \leq n$, in the sum vanishes. Actually, the set $\mathcal{T}$ can be decomposed into two halves,

$$\mathcal{T}_j = \left\{ \sum_{i \in \{1,\ldots,n\} \setminus \{j\}} c_i T_i, \quad c_i \in \{0, 1\} \right\} \text{ and } T_j + \mathcal{T}_j ,$$

such that $x_j(t + \tau) = x_j(t + \tau + T_j)$ for any t and any $\tau \in \mathcal{T}_j$. Therefore, for any j, $1 \leq j \leq n$, we have

$$\bigoplus_{\tau \in \mathcal{T}} x_j(t + \tau) = \bigoplus_{\tau \in \mathcal{T}_j} \left(x_j(t + \tau) \oplus x_j(t + \tau + T_j) \right) = 0 .$$

□

Proposition 1, combined with the fact that the product of the periods of two sequences is a period for their sum, provides several trade-offs between the degree and the number of terms of a parity-check relation for $\boldsymbol{\sigma}$. For instance, if we consider the sequence $\boldsymbol{\sigma}$ defined by

$$\sigma(t) = x_1(t) \oplus x_2(t) \oplus x_3(t) ,$$

then, the following three relations are examples of parity-check relations for $\boldsymbol{\sigma}$ with different numbers of terms:

$$\sigma(t) \oplus \sigma(t + T_1 T_2 T_3) = 0$$
$$\sigma(t) \oplus \sigma(t + T_1) \oplus \sigma(t + T_2 T_3) \oplus \sigma(t + T_1 + T_2 T_3) = 0$$
$$\sigma(t) \oplus \sigma(t + T_1) \oplus \sigma(t + T_2) \oplus \sigma(t + T_1 + T_2) \oplus$$
$$\sigma(t + T_3) \oplus \sigma(t + T_1 + T_3) \oplus \sigma(t + T_2 + T_3) \oplus \sigma(t + T_1 + T_2 + T_3) = 0 .$$

Now, if $\boldsymbol{\sigma}$ is correlated to the keystream $\boldsymbol{S}$, then any parity-check relation for $\boldsymbol{\sigma}$ provides a biased linear relation for the keystream. Actually, for any set $\mathcal{T}$ such that $\bigoplus_{\tau\in\mathcal{T}} \sigma(t+\tau) = 0$ for all $t \geq 0$, we have

$$\bigoplus_{\tau\in\mathcal{T}} S(t+\tau) = \bigoplus_{\tau\in\mathcal{T}} S(t+\tau) \oplus \bigoplus_{\tau\in\mathcal{T}} \sigma(t+\tau) = \bigoplus_{\tau\in\mathcal{T}} (S \oplus \sigma)(t+\tau) .$$

Since the sequence $(\boldsymbol{S} \oplus \boldsymbol{\sigma})$ is biased with bias $\mathcal{E}(f+g)$ where g is the combining function of the small generator producing $\boldsymbol{\sigma}$, then it can be proved that the corresponding parity-check relation applied to $(\boldsymbol{S} \oplus \boldsymbol{\sigma})$ is also biased but with a smaller bias. It is worth noticing that the bias of the parity-check relation cannot be directly derived from the piling-up lemma since the terms in the sum are not statistically independent [19, 21, 35]. Moreover, there might exist two different approximations g and g' of the combining function f such that, for the same $\mathcal{T}$, we have

$$\bigoplus_{\tau\in\mathcal{T}} g(x_{i_1}(t+\tau), \ldots, x_{i_m}(t+\tau)) = 0 \text{ and } \bigoplus_{\tau\in\mathcal{T}} g'(x_{j_1}(t+\tau), \ldots, x_{j_{m'}}(t+\tau)) = 0, \ \forall t \geq 0 .$$

In this case, the bias of the relation applied to the keystream,

$$\bigoplus_{\tau\in\mathcal{T}} f(x_1(t+\tau), \ldots, x_n(t+\tau)) ,$$

cannot be directly deduced from both biases $\mathcal{E}(f+g)$ and $\mathcal{E}(f+g')$. However, the following lower bound on the bias of the parity-check relation on the keystream has been exhibited in [4].

Theorem 1 [4, *Theorem* 5] *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ be n sequences with least periods $T_1, \ldots, T_n$, f a Boolean function of n variables and $\boldsymbol{S} = f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$. Let $\kappa_1, \ldots, \kappa_{s+1}$ be a strictly increasing sequence of integers with $\kappa_1 = 0$ and $\kappa_{s+1} = m$. Let*

$$\mathcal{T} = \langle M_1, \ldots, M_s \rangle$$

where $M_i = q_i \mathrm{lcm}(T_{\kappa_i+1}, \ldots, T_{\kappa_{i+1}})$ for some integer $q_i > 0$. Assume that each M_i is coprime with all T_j with $j \notin [\kappa_i + 1; \kappa_{i+1}]$. Let $\boldsymbol{PC}_{f,\mathcal{T}}$ be the sequence defined by

$$PC_{f,\mathcal{T}}(t) = \bigoplus_{\tau\in\mathcal{T}} s(t+\tau), \ \forall t \geq 0 .$$

Then, for any Boolean function g of m variables of the form

$$g(x_1, \ldots, x_m) = \bigoplus_{i=1}^{s} g_i(x_{\kappa_i+1}, \ldots, x_{\kappa_{i+1}})$$

where each g_i is a Boolean function of $(\kappa_{i+1} - \kappa_i)$ variables, we have

$$\mathcal{E}(\boldsymbol{PC}_{f,\mathcal{T}}) \geq \left[\mathcal{E}(f \oplus g)\right]^{2^s}.$$

In the following, we focus on sets $\mathcal{T}$ of the form

$$\mathcal{T} = \langle M_1, \ldots, M_s \rangle \tag{4}$$

where each M_i equals some T_{i_j} or the product of several T_{i_j} (possibly with a nonzero multiplicative factor) as defined in Theorem 1, and we will assume for the sake of simplicity that all T_{i_j} are coprime. If $\mathcal{T}$ involves all periods T_{i_j}, $1 \leq j \leq m$, then we have that

$$\mathcal{E}(\boldsymbol{PC}_{f,\mathcal{T}}) \geq \left[\mathcal{E}(f \oplus \ell)\right]^{2^s},$$

with $\ell = \bigoplus_{j=1}^{m} x_{i_j}$. Moreover, if $m = R + 1$ where R is the resiliency order of f, which is the usual case in practice, then this lower bound is tight [4, Theorem 12]:

$$\mathcal{E}(\boldsymbol{PC}_{f,\mathcal{T}}) = \left[\mathcal{E}(f \oplus \ell)\right]^{2^s}.$$

Therefore, this bias can be exploited for distinguishing the keystream from a random sequence.

5.2 Distinguishing attacks based on parity-check relations

The distinguishing attack consists in computing the biased sequence

$$PC_{f,\mathcal{T}}(t) = \bigoplus_{\tau \in \mathcal{T}} S(t + \tau), \ \forall t \geq 0$$

from the keystream, where $\mathcal{T}$ is defined as specified by (4). For instance, for $m = R + 1$, a natural choice for $\mathcal{T}$ is

$$\mathcal{T} = \langle T_{i_1}, \ldots, T_{i_m} \rangle .$$

Then, the attacker applies a hypothesis test in order to determine whether the computed sequence has the expected bias or not. The number of samples of the parity-check relation which are needed for detecting the bias is given by

$$N \simeq \frac{2 \ln 2}{\mathcal{E}(\boldsymbol{PC}_{f,\mathcal{T}})^2} \leq \frac{2 \ln 2}{\varepsilon^{2^{m+1}}} \tag{5}$$

where $\varepsilon = \mathcal{E}(f + \ell)$ with $\ell = \bigoplus_{j=1}^{m} x_{i_j}$. As previously discussed, this formula provides an upper bound in the general case, but it is tight for $m = R + 1$. It is worth noticing that the lower bound on $\mathcal{E}(\boldsymbol{PC}_{f,\mathcal{T}})$ implies that this bias is always positive. Therefore, the statistical test aims at maximizing the value of

$$\sum_{t=0}^{N-1} (-1)^{PC_{f,\mathcal{T}}(t)}$$

or equivalently, at minimizing $\sum_{t=0}^{N-1} PC_{f,\mathcal{T}}(t)$.

When $\mathcal{T} = \langle T_{i_1}, \ldots, T_{i_m} \rangle$, the number of keystream bits needed for the distinguishing attack is equal to

$$N + \sum_{j=1}^{m} T_{i_j} \leq \frac{2 \ln 2}{\varepsilon^{2^{m+1}}} + \sum_{j=1}^{m} T_{i_j} \, .$$

The corresponding time complexity is then

$$2^m N \leq \frac{2^{m+1} \ln 2}{\varepsilon^{2^{m+1}}}$$

where equality holds in both formulae when $m = R + 1$. The attack may then be faster than the classical correlation attack, but it has a higher data complexity.

Moreover, it does not allow the initial state of the keystream generator to be recovered.

5.3 Combining both techniques

Much more appropriate trade-offs between time and data complexity can therefore be obtained by combining both attacks. Let us consider m_1 constituent devices, namely $\mathcal{R}_{i_1}, \ldots, \mathcal{R}_{i_{m_1}}$, whose influences will be cancelled by the computation of a parity-check relation. Let ℓ denote the linear function $\ell = \bigoplus_{j=1}^{m_1} x_{i_j}$. Then, this set of m_1 devices must be chosen such that there exists a biased approximation g of $(f + \ell)$, depending only on the $(m - m_1)$ input variables with indexes $i_{m_1+1}, \ldots, i_m$. The most appropriate set of parameters in many situations is given by $m = R + 1$ and $g = \bigoplus_{j=m_1+1}^{m} x_{i_j}$.

The first step of the attack consists in computing the following parity-check relation on the keystream sequence:

$$PC_{f,\mathcal{T}}(t) = \bigoplus_{\tau \in \mathcal{T}} S(t + \tau), \ \forall t \geq 0$$

with $\mathcal{T} = \langle T_{i_1}, \ldots, T_{i_{m_1}} \rangle$.

Then, for each possible initial state of the $(m - m_1)$ devices $\mathcal{R}_{i_{m_1+1}}, \ldots, \mathcal{R}_{i_m}$, a sequence $\boldsymbol{\sigma}$ is computed by

$$\sigma(t) = g(x_{i_{m_1+1}}(t), \ldots, x_{i_m}(t)) \, .$$

The parity-check relation

$$PC_{g,\mathcal{T}}(t) = \bigoplus_{\tau \in \mathcal{T}} \sigma(t + \tau)$$

is then evaluated. If the guessed initial state is correct, then the sequences $\boldsymbol{PC_{f,\mathcal{T}}}$ and $\boldsymbol{PC_{g,\mathcal{T}}}$ are correlated. Actually, we have

$$PC_{f,\mathcal{T}}(t) \oplus PC_{g,\mathcal{T}}(t) = PC_{f,\mathcal{T}}(t) \oplus PC_{g,\mathcal{T}}(t) \oplus PC_{\ell,\mathcal{T}}(t) = PC_{f+g+\ell,\mathcal{T}}(t) \, .$$

The corresponding bias is $\mathcal{E}(\boldsymbol{PC_{f+g+\ell,\mathcal{T}}})$ which is greater than or equal to $\varepsilon^{2^{m_1}}$ with $\varepsilon = \mathcal{E}(f + g + \ell)$. Then, a correlation attack can be performed in order to detect a correlation between $\boldsymbol{PC_{f,\mathcal{T}}}$, which is derived from the keystream, and $\boldsymbol{PC_{g,\mathcal{T}}}$ which is computed for each possible initial state of the $(m - m_1)$ targeted devices.

Recovering the correct initial state among the $(2^{\sum_{j=m_1+1}^{m} L_{i_j}} - 1)$ sequences then requires

$$N \simeq \frac{2 \ln 2 \sum_{j=m_1+1}^{m} L_{i_j}}{\varepsilon^{2^{m_1+1}}} \text{ samples },$$

leading to the following data complexity

$$\frac{2 \ln 2 \sum_{j=m_1+1}^{m} L_{i_j}}{\varepsilon^{2^{m_1+1}}} + \sum_{j=1}^{m_1} T_{i_j} . \tag{6}$$

The time complexity is now

$$2^{m_1} N \times 2^{\sum_{j=m_1+1}^{m} L_{i_j}} = \frac{2^{m_1+1} \ln 2 \sum_{j=m_1+1}^{m} L_{i_j}}{\varepsilon^{2^{m_1+1}}} \times 2^{\sum_{j=m_1+1}^{m} L_{i_j}} \tag{7}$$

for the basic algorithm described by Algorithm 3. It must be noticed that the time complexity is independent from the periods and the lengths of devices $\mathcal{R}_{i_1}, \ldots, \mathcal{R}_{i_{m_1}}$. Obviously, for a given value of m, increasing the number $(m - m_1)$ of devices for which we perform an exhaustive search allows the data complexity to be reduced. It may increase the time complexity, but this is not always the case since the expression (7) for the time complexity consists of the product of two terms, one increasing with $(m - m_1)$ and the second one depending on N, which decreases when m_1 decreases. Therefore, the optimal choice for the parameters highly depends on the size of the devices and on the bias of the approximation. Finding the best trade-off between both terms is then an important task.

Algorithm 3 Correlation attack combining exhaustive search and parity-check relations.

for each t from 0 to $(N-1)$ **do**
  $PC_{f,\mathcal{T}}(t) \leftarrow \bigoplus_{\tau \in \mathcal{T}} S(t+\tau)$
end for
for each initial state of the devices $i_{m_1+1}, \ldots, i_m$ **do**
  $c \leftarrow 0$
  for each t from 0 to $(N-1)$ **do**
    $PC_{g,\mathcal{T}}(t) \leftarrow \bigoplus_{\tau \in \mathcal{T}} g(x_{i_{m_1}+1}(t+\tau), \ldots, x_{i_m}(t+\tau))$
    $c \leftarrow c + (PC_{f,\mathcal{T}}(t) \oplus PC_{g,\mathcal{T}}(t))$
  end for
  if $c >$ threshold **then**
    return the initial states of the $(m - m_1)$ targeted devices.
  end if
end for

Obviously, when $m - m_1 > 1$, the highest value of the correlation between $\boldsymbol{PC_{f,\mathcal{T}}}$ and $\boldsymbol{PC_{g,\mathcal{T}}}$ can be identified faster with Algorithm 2.

The general technique then consists in identifying m_1 devices for building the parity-check relations. Then, we search for an approximation g of $f + \ell$ with bias ε where ℓ is the sum of the m_1 variables involved in the parity-check relations. We decompose g into three functions with disjoint input variables:

$$g(x_{i_{m_1}+1}, \ldots, x_{i_m}) = g_d(x_{i_{m_1+1}}, \ldots, x_{i_{m_1+\partial}}) + g_u(x_{i_{m_1+\partial+1}}, \ldots, x_{i_{m'}}) + g_v(x_{i_{m'+1}}, \ldots, x_{i_m}) . \tag{8}$$

Table 1 Data and time complexities of all variants of the correlation attack.

	m_1	∂	m'	g	T_d	T_u	k	Data complexity	Time complexity
Basic	0	0	0	approx. of f of m var.	1	1	$\sum_{j=1}^{m} L_{i_j}$	$\frac{2k\ln 2}{\varepsilon^2}$	$2^k \times \frac{2k\ln 2}{\varepsilon^2}$
Algo 1	0	0	m'	approx. of f of m var.	1	$\prod_{j=1}^{m'} T_{i_j}$	$\sum_{j=1}^{m} L_{i_j}$	$\frac{2k\ln 2}{\varepsilon^2}$	$2^k \times \left(\frac{2k\ln 2}{T_u\varepsilon^2} + \log T_u\right)$
Algo 2	0	∂	m'	approx. of f of m var.	$\prod_{j=1}^{\partial} T_{i_j}$	$\prod_{j=\partial+1}^{m'} T_{i_j}$	$\sum_{j=\partial+1}^{m} L_{i_j}$	$\frac{2kT_d\ln 2}{\Delta\varepsilon^2} + \Delta$	$2^k \times \left(\frac{2k\ln 2}{T_u\varepsilon^2} + \Delta\log T_u + \Delta 2^{\Delta}\right)$
Section 5.2	m	0	0	$\sum_{j=1}^{m} x_{i_j}$	1	1	0	$\frac{2\ln 2}{\varepsilon^{2^{m+1}}} + \sum_{j=1}^{m} T_{i_j}$	$2^m \times \frac{2\ln 2}{\varepsilon^{2^{m+1}}}$
Algo 3	m_1	0	0	approx. of $\left(f \oplus \sum_{j=1}^{m_1} x_{i_j}\right)$ of $(m-m_1)$ var.	1	1	$\sum_{j=m_1+1}^{m} L_{i_j}$	$\frac{2k\ln 2}{\varepsilon^{2^{m_1+1}}} + \sum_{j=1}^{m_1} T_{i_j}$	$2^{k+m_1} \times \frac{2k\ln 2}{\varepsilon^{2^{m_1+1}}}$
Algos 3 + 1	m_1	0	m'	approx. of $\left(f \oplus \sum_{j=1}^{m_1} x_{i_j}\right)$ of $(m-m_1)$ var.	1	$\prod_{j=m_1+1}^{m'} T_{i_j}$	$\sum_{j=m_1+1}^{m} L_{i_j}$	$\frac{2k\ln 2}{\varepsilon^{2^{m_1+1}}} + \sum_{j=1}^{m_1} T_{i_j}$	$2^{k+m_1} \times \left(\frac{2k\ln 2}{T_u\varepsilon^{2^{m_1+1}}} + \log T_u\right)$
General	m_1	∂	m'	approx. of $\left(f \oplus \sum_{j=1}^{m_1} x_{i_j}\right)$ of $(m-m_1)$ var.	$\prod_{j=m_1+1}^{m_1+\partial} T_{i_j}$	$\prod_{j=m_1+\partial+1}^{m'} T_{i_j}$	$\sum_{j=m_1+\partial+1}^{m} L_{i_j}$	$\frac{2kT_d\ln 2}{\Delta\varepsilon^{2^{m_1+1}}} + \sum_{j=1}^{m_1} T_{i_j} + \Delta$	$2^{k+m_1} \times \left(\frac{2k\ln 2}{T_u\varepsilon^{2^{m_1+1}}} + \Delta\log T_u + \Delta 2^{\Delta}\right)$

Let

$$T_d = \prod_{j=m_1+1}^{m_1+\partial} T_{i_j}, \quad T_u = \prod_{j=m_1+\partial+1}^{m'} T_{i_j} \text{ and } k = \sum_{j=m_1+\partial+1}^{m} L_{i_j},$$

where 2^k is the number of initial states for the devices involved in both approximations g_u and g_v. Then, we need to evaluate the correlation for each of the Δ decimated sequences from

$$N' = \frac{2k \ln 2}{\Delta \varepsilon^{2^{m_1+1}}} \text{ samples.}$$

The corresponding data complexity is then

$$T_d N' + \sum_{j=1}^{m_1} T_{i_j} + \Delta = \frac{2T_d k \ln 2}{\Delta \varepsilon^{2^{m_1+1}}} + \sum_{j=1}^{m_1} T_{i_j} + \Delta \text{ keystream bits.}$$

The time complexity is

$$2^k 2^{m_1} \left(\frac{\Delta N'}{T_u} + \Delta \log T_u + \Delta 2^{\Delta} \right).$$

As extremal cases, we recover the time and data complexities of the correlation attacks presented in the previous sections. More precisely, Table 1 describes all variants of the attack. The number of variables m can take any value between 1 and $(n-1)$, while the only requirement on (m_1, ∂, m') is that the involved approximation g can be decomposed as (8).

6 Conclusions

In this paper we have successfully generalised the five correlation attacks [20, 21, 28, 35, 36] presented to analyse the successive versions of the combination generator based on NLFSRs, Achterbahn. We have also showed that some of these improvements apply to a more general problem which is encountered in some other contexts in cryptography. In the context of the general combination generator, we have defined a whole family of correlation attacks using several additional ideas against this type of cipher that provides different time-data-memory trade-offs. These are the best known attacks for the considered construction. We have provided general formulas for computing accurate complexity estimates in each case. This allows to find the optimal attack in each particular case. We hope that this work will help future designers to know a priori how the parameters of the ciphers need to be chosen for being resistant to such attacks, as well as will permit the cryptanalysts to apply in an automatic way these attacks. We believe that this generalisation of the attacks proposed against Achterbahn will provide a better understanding, which is very important for some other possible uses and for finding potential future improvements.

References

1. Biryukov, A., De Cannière, C., Quisquater, M.: On multiple linear approximations. In: Advances in Cryptology—CRYPTO 2004. Lecture Notes in Computer Science, vol. 3152, pp. 1–22. Springer, Heidelberg (2004)
2. Blahut, R.E.: Fast Algorithms for Digital Signal Processing. Addison Wesley (1985)
3. Canteaut, A., Filiol, E.: Ciphertext only reconstruction of stream ciphers based on combination generators. In: Fast Software Encryption—FSE 2000. Lecture Notes in Computer Science, vol. 1978, pp. 165–180. Springer-Verlag (2001)
4. Canteaut, A., Naya-Plasencia, M.: Parity-check relations on combination generators. IEEE Trans. Inf. Theory **58**(6), 3900–3911 (2012)
5. Canteaut, A., Trabbia, M.: Improved fast correlation attacks using parity-check equations of weight 4 and 5. In: Advances in Cryptology—EUROCRYPT 2000. Lecture Notes in Computer Science, vol. 1807, pp. 573–588. Springer-Verlag (2000)
6. Chepyshov, V., Johansson, T., Smeets, B.: A simple algorithm for fast correlation attacks on stream ciphers. In: Fast Software Encryption—FSE 2000, Lecture Notes in Computer Science, vol. 1978, pp. 124–135. Springer-Verlag (2000)
7. Chose, P., Joux, A., Mitton, M.: Fast correlation attacks: an algorithmic point of view. In: Advances in Cryptology—EUROCRYPT 2002. Lecture Notes in Computer Science, vol. 2332, pp. 209–221. Springer-Verlag (2002)
8. Coppersmith, D., Halevi, S., Jutla, C.: Cryptanalysis of stream ciphers with linear masking. In: Advances in Cryptology—CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442. Springer-Verlag (2002)
9. Courtois, N.: Fast algebraic attacks on stream ciphers with linear feedback. In: Advances in Cryptology—CRYPTO 2003. Lecture Notes in Computer Science, vol. 2729, pp. 176–194. Springer-Verlag (2003)
10. Courtois, N., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In: Advances in Cryptology—EUROCRYPT 2003. Lecture Notes in Computer Science, vol. 2656, pp. 345–359. Springer-Verlag (2003)
11. ECRYPT—European Network of Excellence in Cryptology: The eSTREAM Stream Cipher Project. http://www.ecrypt.eu.org/stream/ (2004)
12. Ekdahl, P., Johansson, T.: Distinguishing attacks on SOBER-t16 and t32. In: Fast Software Encryption—FSE 2002. LNCS, vol. 2365, pp. 210–224. Springer (2002)
13. Gammel, B., Göttfert, R., Kniffler, O.: The Achterbahn stream cipher. Submission to eSTREAM. http://www.ecrypt.eu.org/stream/ (2005)
14. Gammel, B., Göttfert, R., Kniffler, O.: Achterbahn-128/80. Submission to eSTREAM. http://www.ecrypt.eu.org/stream/ (2006)
15. Gammel, B., Göttfert, R., Kniffler, O.: Status of Achterbahn and Tweaks. In: Proceedings of SASC 2006—Stream Ciphers Revisited. http://www.ecrypt.eu.org/stream/papersdir/2006/027.pdf (2006)
16. Gammel, B., Göttfert, R., Kniffler, O.: Achterbahn-128/80: design and analysis. In: Proceedings of SASC 2007—Stream Ciphers Revisited. http://www.ecrypt.eu.org/stream/papersdir/2007/020.pdf (2007)
17. Gérard, B., Tillich, J.P.: On linear cryptanalysis with many linear approximations. In: IMA International Conference, Cryptography and Coding. Lecture Notes in Computer Science, vol. 5921, pp. 112–132. Springer (2009)
18. Gérard, B., Tillich, J.P.: Advanced Linear Cryptanalysis of Block and Stream Ciphers, vol. 7, chap. Using Tools from Error Correcting Theory in Linear Cryptanalysis, pp. 87–114. IOS Press (2011)
19. Göttfert, R., Gammel, B.: On the frame length of Achterbahn-128/80. In: Proceedings of the 2007 IEEE Information Theory Workshop on Information Theory for Wireless Networks, pp. 1–5. IEEE (2007)
20. Hell, M., Johansson, T.: Cryptanalysis of Achterbahn-Version 2. In: Selected Areas in Cryptography—SAC 2006. Lecture Notes in Computer Science, vol. 4356, pp. 45–55. Springer (2006)
21. Hell, M., Johansson, T.: Cryptanalysis of Achterbahn-128/80. IET Inf. Secur. **1**(2), 47–52 (2007)
22. Hell, M., Johansson, T., Brynielsson, L.: An overview of distinguishing attacks on stream ciphers. Cryptogr. Commun. **1**(1), 71–94 (2009)

23. Hermelin, M., Cho, J., Nyberg, K.: Multidimensional extension of Matsui's Algorithm 2. In: Fast Software Encryption—FSE 2009. Lecture Notes in Computer Science, vol. 5665, pp. 209–227. Springer (2009)
24. Hermelin, M., Nyberg, K.: Advanced Linear Cryptanalysis of Block and Stream Ciphers, vol. 7, chap. Linear Cryptanalysis Using Multiple Linear Approximations, pp. 25–54. IOS Press (2011)
25. Johansson, T., Jönsson, F.: Fast correlation attacks based on turbo code techniques. In: Advances in Cryptology—CRYPTO'99. Lecture Notes in Computer Science, vol. 1666, pp. 181–197. Springer-Verlag (1999)
26. Johansson, T., Jönsson, F.: Improved fast correlation attack on stream ciphers via convolutional codes. In: Advances in Cryptology—EUROCRYPT'99. Lecture Notes in Computer Science, vol. 1592, pp. 347–362. Springer-Verlag (1999)
27. Johansson, T., Jönsson, F.: Fast correlation attacks through reconstruction of linear polynomials. In: Advances in Cryptology—CRYPTO'00. Lecture Notes in Computer Science, vol. 1880, pp. 300–315. Springer-Verlag (2000)
28. Johansson, T., Meier, W., Muller, F.: Cryptanalysis of Achterbahn. In: Fast Software Encryption—FSE 2006, Lecture Notes in Computer Science, vol. 4047, pp. 1–14. Springer (2006)
29. Joux, A.: Algorithmic Cryptanalysis. Chapman & Hall/CRC (2009)
30. Junod, P., Vaudenay, S.: Optimal key ranking procedures in a statistical cryptanalysis. In: Fast Software Encryption—FSE 2003. Lecture Notes in Computer Science, vol. 2887, pp. 235–246. Springer-Verlag (2003)
31. Lu, Y., Vaudenay, S.: Faster correlation attack on Bluetooth keystream generator E0. In: Advances in Cryptology—CRYPTO 2004. Lecture Notes in Computer Science, vol. 3152, pp. 407–425. Springer-Verlag (2004)
32. Matsui, M.: The first experimental cryptanalysis of the data encryption standard. In: Advances in Cryptology—CRYPTO'94. Lecture Notes in Computer Science, vol. 839. Springer-Verlag (1995)
33. Meier, W., Staffelbach, O.: Fast correlation attacks on stream ciphers. In: Advances in Cryptology—EUROCRYPT'88. Lecture Notes in Computer Science, vol. 330, pp. 301–314. Springer-Verlag (1988)
34. Meier, W., Staffelbach, O.: Fast correlation attack on certain stream ciphers. J. Cryptol. **1**(3), 159–176 (1989)
35. Naya-Plasencia, M.: Cryptanalysis of Achterbahn-128/80. In: Fast Software Encryption—FSE 2007. Lecture Notes in Computer Science, vol. 4593, pp. 73–86. Springer (2007)
36. Naya-Plasencia, M.: Cryptanalysis of Achterbahn-128/80 with a new keystream limitation. In: WEWoRC 2007—Second Western European Workshop in Research in Cryptology. Lecture Notes in Computer Science, vol. 4945, pp. 142–152. Springer (2008)
37. Siegenthaler, T.: Correlation-immunity of nonlinear combining functions for cryptographic applications. IEEE Trans. Inf. Theory **30**(5), 776–780 (1984)
38. Siegenthaler, T.: Decrypting a class of stream ciphers using ciphertext only. IEEE Trans. Comput. **C-34**(1), 81–84 (1985)
39. Zhang, M.: Maximum correlation analysis of nonlinear combining functions in stream ciphers. J. Cryptol. **13**(3), 301–313 (2000)