Fast Correlation Attack Algorithm with List Decoding and an Application

Miodrag J. Mihaljević[1,2], Marc P.C. Fossorier[3], and Hideki Imai[4]

[1] Mathematical Institute, Serbian Academy of Sciences and Arts,
Kneza Mihaila 35, 11001 Belgrade, Yugoslavia
miodragm@turing.mi.sanu.ac.yu
[2] SONY Computer Science Laboratories
Takanawa Muse Bld., 3-14-13, Higashi-Gotanda, Shinagawa-ku,
Tokyo, 141-0022 Japan
[3] Department of Electrical Engineering, University of Hawaii,
2540 Dole St., Holmes Hall 483, Honolulu, HI 96822, USA
marc@spectra.eng.hawaii.edu
[4] University of Tokyo, Institute of Industrial Science,
4-6-1, Komaba, Meguro-ku, Tokyo, 153-8505 Japan
imai@iis.u-tokyo.ac.jp

Abstract. An improved method for the fast correlation attack on certain stream ciphers is presented. The proposed algorithm employs the following decoding approaches: list decoding in which a candidate is assigned to the list based on the most reliable information sets, and minimum distance decoding based on Hamming distance. Performance and complexity of the proposed algorithm are considered. A desirable characteristic of the proposed algorithm is its theoretical analyzibility, so that its performance can also be estimated in cases where corresponding experiments are not feasible due to the current technological limitations. The algorithm is compared with relevant recently reported algorithms, and its advantages are pointed out. Finally, the proposed algorithm is considered in a security evaluation context of a proposal (NESSIE) for stream ciphers.

Keywords. Stream ciphers, keystream generators, linear feedback shift registers, nonlinear combiner, nonlinear filter, cryptanalysis.

1 Introduction

An important method for attack or security examination of certain stream ciphers based on the nonlinear combination keystream generators and the nonlinear filter keystream generators (see [13], for example) is the fast correlation attack proposed in [14] and [22] as a significant improvement of the basic correlation attack [20]. Further extensions and refinements of fast correlation attack, as well as its analysis are presented in a number of papers including the following most recent ones: [8], [9], [7], [3], [15], [2], [16], [17]-[18], and [10]. The basic ideas

M. Matsui (Ed.): FSE 2001, LNCS 2355, pp. 196–210, 2002.

of the reported fast correlation attacks explicitly or implicitly include the following two main steps: (i) transform the cryptographic problem into a suitable decoding one; and (ii) apply (devise) an appropriate decoding algorithm. The decoding step employs either one-step or iterative decoding techniques.

Recently two techniques for fast correlation attack based on one-step decoding have been proposed in [3] and [16] demonstrating low-complexity and high-performance, and very recently another one-step based method for the fast correlation attack through reconstruction of linear polynomials has been reported in [10], showing powerful performance and low complexity. Also, a high-performance iterative decoding technique for fast correlation attack have been reported in [2] and [17]. All these approaches do not depend on the LFSR feedback polynomial weight.

Having in mind these recent achievements, the main objective of this paper is to propose an improved noniterative decoding approach for the fast correlation attack and to compare it with the recently reported results. The proposed approach employs list and minimum distance decoding, and it is based on the most reliable information sets.

The paper is organized as follows. Section 2 summarizes the decoding approach for fast correlation attack. Section 3 points out the main underlying ideas for the construction of an improved algorithm for the cryptanalysis. The complete algorithm for the cryptanalysis is proposed in Section 4. The main characteristics of the algorithm are discussed in Sections 5 and 6. An application of the algorithm for the security evaluation of a proposal for stream ciphers (NESSIE: LILI-128) is given in Section 7. Finally, the results of this paper are summarized in Section 8.

2 Background

2.1 Decoding Model of the Cryptanalysis

A nonlinear combination keystream generator combines the outputs of several linear feedback shift registers (LFSRs) by a nonlinear Boolean function. A nonlinear filter keystream generator consists of a single LFSR and a nonlinear Boolean function whose inputs are taken from some LFSR stages to produce the output. Both schemes may become vulnerable due to the correlation between an LFSR and the generator output.

The term correlation means that the mod 2 sum of the corresponding outputs of an LFSR and a generator can be considered as a realization of a binary random variable which takes value 0 and 1 with the probabilities $1-p$ and p, respectively, $p \neq 0.5$.

The fast correlation attack on a particular LFSR, with primitive feedback polynomial, in a nonlinear combining generator given the segment of the generator output can be considered as follows (a similar consideration is valid for the nonlinear filter, as well):

- the n-bit segment of the output sequence from the length-k LSFR is a codeword of an (n, k) punctured simplex code;
- the corresponding n-bit segment of the nonlinear combination generator output is the corresponding noisy codeword obtained through a BSC with crossover probability p;
- the problem of the LFSR initial state reconstruction, assuming known characteristic polynomial, is equivalent to the problem of decoding after transmission over a BSC with crossover probability p.

In the following, x_n, $n = 1, 2, ..., N$, denotes an LFSR output sequence which is a codeword $\mathbf{x}$ of a binary (N, L) punctured simplex code $\mathbf{C}$ where N is codeword length and L is number of information bits. $\mathbf{x}_0 = [x_1, x_2, ..., x_L]$ is the vector of information bits identical to the LFSR initial state; $\{z_n\}$ denotes the degraded sequence $\{x_n\}$ after transmission over a BSC with crossover probability p. Accordingly, $z_n = x_n \oplus e_n$, $n = 1, 2, ..., N$, where the effect of the BSC with crossover error probability p is modeled by an N-dimensional binary random variable $\mathbf{E}$ defined over $\{0, 1\}^N$ with independent coordinates E_n such that $\Pr(E_n = 1) = p$, $n = 1, 2, \ldots, N$, and e_n is a realization of E_n. Applying a codeword $\mathbf{x} = [x_n]_{n=1}^N \in \mathbf{C}$, to the input of the BSC, we obtain the random variable $\mathbf{Z} = \mathbf{E} \oplus \mathbf{x}$ as a received codeword at its output. Let $\mathbf{z} = [z_n]_{n=1}^N$ and $\mathbf{e} = [e_n]_{n=1}^N$ denote particular values of the random vector variables $\mathbf{Z}$ and $\mathbf{E}$, respectively.

2.2 List Decoding

List decoding was introduced in [4] and [21] and random coding arguments were used to explore its average decoding error probability for block codes of low rates for the BSC (for more details see [5] and [6], for example).

In list decoding of a block code, the decoder gives as its output not one codeword but a list of possible candidate codewords. The list decoder is in error only if the correct codeword is not on the list.

When the list is composed of ℓ candidates the decoding is denoted as list-of-ℓ decoding.

3 Underlying Ideas for Cryptanalysis

The developed algorithm is based on a pre-processing stage and a processing stage performed as follows.

3.1 Pre-processing Stage

The central issue in the pre-processing stage is the construction of parity-check equations which follows the approach proposed in [15].

An LFSR can be considered as a linear finite state machine, and accordingly, a state of a length-L LFSR after t clocks is given by the following matrix-vector product over GF(2):

$$\mathbf{x}_t = \mathbf{A}^t \mathbf{x}_0 \ , \ t = 1, 2, ... \ , \tag{1}$$

where $\mathbf{x}_t$ is an L dimensional binary vector representing the LFSR state after t clocks, $\mathbf{x}_0$ is an L dimensional binary vector representing the initial LFSR state, and $\mathbf{A}^t$ is the t-th power over GF(2) of the $L \times L$ state transition binary matrix $\mathbf{A}$. Assuming the LFSR characteristic polynomial $f(u) = 1 + \sum_{i=1}^{L} b_i u^i$, the matrix $\mathbf{A}$ is given by:

$$\mathbf{A} = \begin{bmatrix} b_1 & b_2 & b_3 & ... & b_L \\ 1 & 0 & 0 & ... & 0 \\ 0 & 1 & 0 & ... & . \\ . & . & . & ... & . \\ 0 & & & ... 1 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ . \\ \mathbf{A}_L \end{bmatrix} ,$$

where each $\mathbf{A}_i$, $i = 1, 2, ..., L$, represents a $1 \times L$ binary matrix (a row-vector).

The powers of the matrix $\mathbf{A}$ determine algebraic replica of the LFSR initial state bits, i.e., linear equations satisfied by the bits of the codewords from the dual code. Consequently, assuming that the first B information bits are known due to employment of a partial exhaustive search, for the i-th codeword bit, $i = B+1, B+2, ..., D$, where D is number of codeword bits under consideration, the novel algorithm employs the parity-check equations constructed according to the following definition.

Definition 1. The set Ω_i of parity-check equations associated with bit-i is composed of:

- For $i = B + 1, B + 2, ..., L$:
  All parity-check equations obtained as the *mod*2 sum of two distinct basic parity-check equations,

  $$(z_m \oplus \mathbf{A}_1^m \mathbf{z}_0) \oplus (z_n \oplus \mathbf{A}_1^n \mathbf{z}_0) ,$$

  where m and n have arbitrary values providing that the vector sum $\mathbf{A}_1^m \oplus \mathbf{A}_1^n$ has arbitrary values in the first B coordinates, value one at the i-th coordinate, and value zero in the all other coordinates.
- For $i = L + 1, L + 2, ..., D$:
  All parity-check equations obtained as the *mod*2 sum of three distinct basic parity-check equations,

  $$(z_i \oplus \mathbf{A}_1^i \mathbf{z}_0) \oplus (z_m \oplus \mathbf{A}_1^m \mathbf{z}_0) \oplus (z_n \oplus \mathbf{A}_1^n \mathbf{z}_0) ,$$

  where m and n have arbitrary values providing that the vector sum $\mathbf{A}_1^i \oplus \mathbf{A}_1^m \oplus \mathbf{A}_1^n$ has arbitrary values in the first B coordinates, and value zero in the all other coordinates.

Note that the previous definition differs from these given in [15], and also that for given parameters L, B and D, the sets Ω_i, $i = B + 1, B + 2, ..., D$, can be constructed in advance through a search procedure in a preprocessing phase, and later used for any particular application with these given parameters.

According to [15] and [17], in any set Ω_i specified by Definition 1, $i = B + 1, B + 2, ..., D$, an approximation on the expected number $\mid \bar{\Omega} \mid$ of parity-checks in Ω_i is given by:

$$\mid \bar{\Omega} \mid \approx 2^{B-L}\binom{N-L}{2} \text{ or } 2^{B-L}\binom{N-D}{2} \approx 2^{B-L}\binom{N}{2}. \quad (2)$$

3.2 Processing Stage

The processing stage employs list decoding to form a list of candidates and the minimum distance decoding (MDD) to select the true candidate. The list of candidates is formed based on a directed search and most reliable information sets. Accordingly, the processing consists of the following steps:
– Setting the hypothesis;
– Parity-checks evaluation;
– Selection of the most reliable information sets and assigning the list decoding candidate;
– MDD: correlation check.

4 Algorithm for Fast Correlation Attack Based on List and Minimum Distance Decoding

INPUT:
– the generator output $\{z_i\}_i^N$;
– the LFSR feedback polynomial;
– the correlation noise probability p, the missing event probability P_m and the false alarm probability P_f;
– the algorithm parameters: B and D.

PRE-PROCESSING

- *Setting of the algorithm parameters M and T*:
  for given p, P_m and P_f determine (according to [20]) the required number M of bits for MDD (i.e. the correlation check), and the MDD threshold T.
- *Determination of the parity-check equations*:
  for each codeword bit i, $i = B + 1, B + 2, ..., D$, construct the set Ω_i of corresponding parity-check equations based on Definition 1.

PROCESSING: List-of-2^{B+1} decoding and MDD employing the most reliable information sets.

1. *Setting the hypothesis*
   - Set a new hypothesis on the first B coordinates of the LFSR initial state, i.e. previously not considered pattern $\hat{x}_1, \hat{x}_2, ..., \hat{x}_B$, for the first B information bits, based on z_i, $i = 1, 2, ..., B$ and a new, previously unconsidered, most likely error-pattern.
   - If no one new pattern is possible go to Output (b).

2. *Parity-checks evaluation*
   For each codeword bit position i, $i = B+1, B+2, ..., D$, calculate the parity-check values employing the parity check equations in the set Ω_i.
3. *Selection of the most reliable information sets*
   Based on the evaluation of the parity-check equations on the codeword bit position i, $i = B+1, B+2, ..., D$, form two most reliable estimations of all information bits according to the following.
   - select $L-B$ positions corresponding to the bits with the most **satisfied** parity checks, and assuming that all of them are correct recalculate the information bits on coordinates $i = B+1, B+2, ..., L$.
   - select $L-B$ positions corresponding to the bits with the most **unsatisfied** parity checks, and assuming that all of them are in error, complement them and recalculate the information bits on coordinates $i = B+1, B+2, ..., L$.
4. *MDD - Correlation check*
   For each of two estimations obtained from Step 3, check if the current estimation of the information bits $\hat{\mathbf{x}}_0 = [\hat{x}_1, \hat{x}_2, ..., \hat{x}_L]$, is the true one, according to the following:
   For a given $\hat{\mathbf{x}}_0$, generate the corresponding sequence $\hat{x}_1, \hat{x}_2, ..., \hat{x}_M$, and calculate
   $S = \sum_{n=1}^{M} \hat{x}_n \oplus z_n$.
   If $S \leq T$ go to the Output (a), otherwise go to Step 1.

OUTPUT:
(a) The considered vector $\hat{\mathbf{x}}_0$ of information bits accepted as the true one;
(b) The true vector of information bits is not found.

5 Performance and Complexity Analyses

5.1 Performance Analysis

For each i, $i = B+1, B+2, \ldots, D$, let $q_i(\mathbf{z})$ or, simply, q_i denote the ratio of posterior error probabilities defined by

$$q_i = \frac{\Pr(E_i = 1 \mid \mathbf{z})}{1 - \Pr(E_i = 1 \mid \mathbf{z})} \quad , \tag{3}$$

evaluated based on a set of parity-check equations Ω_i, where E_i denotes a binary random variable which realization is the noisy bit on the ith codeword bit. In the particular case where all the parity-check equations are orthogonal and involve exactly two unknown bits beside the considered one, and s among $\mid \Omega \mid$ parity-check equations are satisfied, we have

$$q_i = q = \frac{p}{1-p} \left(\frac{1 + (1-2p)^2}{1 - (1-2p)^2} \right)^{|\Omega| - 2s} \quad , \tag{4}$$

recalling that p denotes the crossover probability of the BSC.

The same expression may also be used as an appropriate approximation for nonorthogonal, but linearly independent parity-checks.

The next statements yield the probability that the correct codeword (i.e. the key) appears in the decoding list.

Theorem 1. Let $p \leq 0.5$. The expected probability P_α that the $L-B$ bits with the largest numbers of satisfied parity-checks from a sequence of $D-B$ bits, are all error-free is given by

$$P_\alpha = [1 - \frac{1}{\Sigma_\alpha^{(Pr)}} \sum_{s=|\Omega|-C_\alpha}^{|\Omega|} \Pr(S=s)\ \frac{q}{q+1}]^{L-B}\ , \tag{5}$$

where

$$\Pr(S=s) = p\binom{|\ \Omega\ |}{s} p_w^s (1-p_w)^{|\Omega|-s}\ + (1-p)\binom{|\ \Omega\ |}{s}(1-p_w)^s p_w^{|\Omega|-s}\ , \tag{6}$$

$$p_w = \frac{1-(1-2p)^2}{2}\ , \tag{7}$$

$$\Sigma_\alpha^{(Pr)} = \sum_{s=|\Omega|-C_\alpha}^{|\Omega|} \Pr(S=s)\ , \tag{8}$$

and C_α is the smallest integer such that

$$(D-B)\Sigma_\alpha^{(Pr)} \geq L-B\ . \tag{9}$$

Theorem 2. Let $p \leq 0.5$. The expected probability P_β that the $L-B$ bits with the smallest numbers of satisfied parity-checks from a sequence of $D-B$ bits, are all in error is given by

$$P_\beta = [1 - \frac{1}{\Sigma_\beta^{(Pr)}} \sum_{s=0}^{C_\beta} \Pr(S=s)\ \frac{1}{q+1}]^{L-B}\ , \tag{10}$$

where $\Pr(S=s)$ is given by (6)-(7),

$$\Sigma_\beta^{(Pr)} = \sum_{s=0}^{C_\beta} \Pr(S=s)\ , \tag{11}$$

and C_β is the smallest integer such that

$$(D-B)\Sigma_\beta^{(Pr)} \geq L-B\ . \tag{12}$$

The proofs of Theorem 1 and Theorem 2 follow from the general results given in the Appendix. Note that C_α and C_β are chosen such that the corresponding check sum values occur with sufficiently high probability.

Theorems 1 and 2 imply the following corollary.

Corollary 1. The probability that the correct candidate is on the decoding list is equal to $P_\alpha + P_\beta - P_\alpha P_\beta$, and accordingly it is upper bounded by $\min\{P_\alpha + P_\beta;\ 1\}$ where P_α and P_β are given by Theorems 1 and 2, respectively.

The upper bound of Corollary 1 is expected to be tight when the probability that the correct codeword belongs to both lists remains small compared with both P_α and P_β.

5.2 Complexity Analysis

The search for the desired parity-checks can be done employing a time-memory trade-off as pointed out in [14], [3], [2] and [10].

According to the algorithm structure it can be directly shown that the following two statements hold.

Corollary 2. The time complexity of the pre-processing phase is proportional to $D\binom{N-L}{2}$, without employment of time-memory complexity trade-off, and it is proportional to $D(N-L)$ assuming employment of a sorting based approach and a memory proportional to $(N-L)$.

Corollary 3. Assuming that $|\Omega|$ denotes the average cardinality of the parity-check sets $|\Omega_i|$, and that w denotes the weight of the LFSR characteristic polynomial, the worst case time complexity of the processing phase is proportional to $2^B[\ (D-B)|\Omega| \ +\ (M-L)w\ ]\ mod 2$ additions.

5.3 Illustrative Examples

Tables 1 - 3 illustrate the performance and complexity of the proposed algorithm. Since the dominant operations in the algorithm are $mod 2$ additions, the overall algorithm complexity is proportional to a number of $mod 2$ additions. Also note that $L = 89$ corresponds to a longer LFSR than the longest one ($L = 60$) considered in [10].

6 Comparison

This section gives a comparison of the proposed algorithm and three recently reported ones, [15], [2] and [10], which address the same problem. The main differences in the underlying ideas are pointed out and illustrative examples related to the performance and complexity are presented.

6.1 Comparison of the Approaches

All the fast correlation attacks under comparison are based on certain decoding techniques associated with parity-checks.

The underlying decoding approaches of the algorithms under comparison can be summarized as follows:

Table 1. Proposed algorithm: Noise limit and processing complexity for which the algorithm yields, with probability close to 1, correct reconstruction of the initial LFSR state, when the LFSR characteristic polynomial is $1 + u + u^3 + u^5 + u^9 + u^{11} + u^{12} + u^{17} + u^{19} + u^{21} + u^{25} + u^{27} + u^{29} + u^{32} + u^{33} + u^{38} + u^{40}$, and the available sample is 400000 bits.

Noise Limit	Complexity
0.457	$\sim 2^{38}$ $mod2$ additions
0.460	$\sim 2^{39}$ $mod2$ additions
0.464	$\sim 2^{40}$ $mod2$ additions
0.467	$\sim 2^{41}$ $mod2$ additions
0.469	$\sim 2^{42}$ $mod2$ additions

Table 2. Proposed algorithm: Noise limit and processing complexity for which the algorithm yields, with probability close to 1, correct reconstruction of the initial LFSR state, when the LFSR characteristic polynomial is $1 + u + u^3 + u^5 + u^9 + u^{11} + u^{12} + u^{17} + u^{19} + u^{21} + u^{25} + u^{27} + u^{29} + u^{32} + u^{33} + u^{38} + u^{40}$, and the available sample is 360000 bits.

Noise Limit	Complexity
0.488	$\sim 2^{52}$ $mod2$ additions
0.489	$\sim 2^{53}$ $mod2$ additions
0.490	$\sim 2^{55}$ $mod2$ additions

Table 3. Proposed algorithm - Theoretical analysis: Noise limit, processing complexity and required ample size for which the algorithm yields, with probability close to 1, correct reconstruction of the initial LFSR state, when LFSR length is 89 with an arbitrary primitive characteristic polynomial.

Noise Limit	Complexity	Required Sample
0.469	$\sim 2^{52}$ $mod2$ add.	$\sim 0.25 \cdot 10^{12}$
0.478	$\sim 2^{52}$ $mod2$ add.	$\sim 10^{12}$
0.480	$\sim 2^{52}$ $mod2$ add.	$\sim 4 \cdot 10^{12}$

- one step decoding algorithm (OSDA) [15] is a combination of the threshold decoding [12], and the MDD based on Hamming distance,
- the algorithm [2] is a variant of iterative probabilistic decoding,
- the basic algorithm of [10] is a variant of MDD based on the squared distance measure,
- the proposed algorithm employs list decoding based on the most reliable information sets and MDD with Hamming distance.

The algorithms [15], [10] and the proposed one mainly use parity checks which virtually include only three unknown bits based on employment of exhaustive search, while algorithm [2] uses parity-checks of weight 4 or 5 without employment of exhaustive search in the processing phase.

It appears that list decoding based on the most reliable information sets is more powerful than threshold decoding in conjunction with certain exhaustive search, which explains the gain obtained with the proposed algorithm in comparison with the OSDA algorithms [15]-[16].

The algorithm [2] is based on an iterative probabilistic decoding approach which is a powerful one but complex as well (it requires real number operations as the dominant ones). According to [17], it seems that the iterative decoding techniques can be appropriate for the fast correlation attack when very limited samples are available for processing. So, for a given complexity and a large processing sample, certain noniterative decoding techniques can yield better performance-complexity trade-off.

The approach [10] employs the squared distance measure to avoid an exhaustive search over certain 2^n hypotheses where n is a parameter of the algorithm (see [10] for details) in order to reduce the required exhaustive search. Nevertheless, the use of this squared distance measure rather than Hamming distance becomes suboptimal for MDD over a BSC (see [11], p. 9-10, for example).

The previous discussion points out certain underlying arguments for justifying a better performance-complexity trade-off of the proposed algorithm in comparison with previously reported ones.

6.2 Comparison of Performance and Complexity

The following characteristics of the algorithms are under consideration: performance (noise limit for successful decoding); complexity of the processing; complexity of the preprocessing; required input.

In the considered comparison context, the time complexity of OSDA [15] is proportional to $2^B[\ (L-B)2^{L-B}\binom{N-L}{2}\ +\ (M-L)w\ ]\ mod 2$ additions, where M is length of the sequence for the correlation check, and w denotes the weight of the LFSR characteristic polynomial.

Iterative decoding [2] has expected implementation time complexity proportional to $10Nm(d)$ real number operations assuming at most 10 iterations, where N is the required sample length as well as total number of bits under processing, d is the number of bits involved in a parity-check, and $m(d)$ is the expected number of parity-checks per bit which is approximately equal to $2^{-L}N^{d-1}/(d-1)!$ (see [2]). Also, recall that the algorithm [2] employs *log*-domain processing.

The basic algorithm [10] has time complexity proportional to $2^{2k-L}n\binom{N}{t}$, where k, n, t are the algorithm parameters (see [10]). Also, the following statement is valid: In general, increasing t improves the performance at the cost of an increased precomputation time and increased memory requirement in precomputation, as well as an increased time complexity of the attack itself.

Processing complexity of the proposed algorithm is given by Corollary 3.

Finally, note that the statements on the pre-processing complexity given in [15]-[16], [2] and [10], as well as Definition 1 directly imply that the preprocessing complexity is proportional to $N^\alpha/(\alpha-1)!$ where N is length of the sequence under processing (i.e. required length of the input sequence), and, α is equal to 2 for the algorithms [15]-[17], to t for the algorithm [10], and to the weight

minus one of the employed parity-checks for the algorithm [2]. The previous discussion assumes a straightforward pre-processing without employment of the time-memory trade-off methods.

Tables 4 and 5 present an illustrative comparison of the proposed algorithm and three relevant recently reported algorithms of [15], [2] and [10]. These tables show that the proposed algorithm yields better performance assuming the same input and the same or lower complexity.

Table 4. Comparison of the algorithms, assuming the same inputs: Noise limit and processing complexity for which the algorithms yield, with probability close to 1, correct reconstruction of the initial LFSR state, when the LFSR characteristic polynomial is $1+u+u^3+u^5+u^9+u^{11}+u^{12}+u^{17}+u^{19}+u^{21}+u^{25}+u^{27}+u^{29}+u^{32}+u^{33}+u^{38}+u^{40}$, and the available sample is 360000 bits.

ALGORITHM	Noise Limit	Complexity
FSE2000 [15]	**0.476**	$\sim 2^{52}$
EUROC2000 [2]	**0.482**	$\sim 2^{52}$
Proposed	**0.488**	$\sim 2^{52}$

Table 5. Comparison of the algorithms, assuming the same inputs: Noise limit and processing complexity for which the algorithms yield, with probability close to 1, correct reconstruction of the initial LFSR state, when the LFSR characteristic polynomial is $1+u+u^3+u^5+u^9+u^{11}+u^{12}+u^{17}+u^{19}+u^{21}+u^{25}+u^{27}+u^{29}+u^{32}+u^{33}+u^{38}+u^{40}$, and the available sample is 400000 bits.

ALGORITHM	Noise Limit	Complexity
FSE2000 [15]	**0.420**	$\sim 2^{38}$
CRYPTO2000 [10]	**0.450**	$\sim 2^{38}$
Proposed	**0.457**	$\sim 2^{38}$

Finally, note that an actual implementation of the algorithms strongly depends on the environment and its optimization. Consequently it intentionally will not be discussed here, recalling that as an illustration of this issue, the results reported in [2] and [10] can be considered.

7 Security Consideration of a NESSIE Proposal

This section shows an application of the proposed algorithm for security consideration of a submission to the New European Schemes for Signatures, Integrity and Encryption (NESSIE) Project: a proposal for synchronous stream cipher LILI-128, [19].

The LILI-128 keystream generator is a keystream generator that uses two binary LFSRs, $LFSR_c$ and $LFSR_d$ of lengths 39 and 89, respectively, and two functions f_c and f_d to generate a pseudorandom binary keystream sequence. The components of the keystream generator can be grouped into two subsystems based on the functions they perform; clock control and data generation. The LFSR for the clock-control sub-system is regularly clocked. The output of this subsystem is an integer sequence which controls the clocking of the LFSR within the data-generation subsystem. If regularly clocked, the data-generation subsystem is a simple nonlinearly filtered LFSR (nonlinear filter generator). Accordingly, the LILI-128 generator may be viewed as a clock-controlled nonlinear filter generator with 128 bits secret key.

The state of LILI-128 is defined to be the contents of the two LFSRs. The functions f_c and f_d are evaluated from the current state data, and the feedback bits are calculated. Then the LFSRs are clocked and the keystream bit is output. At initialization, the 128 bit key is used directly to form the initial values of the two shift registers, from left to right, the first 39 bits in $LFSR_c$, then the remaining 89 bits in $LFSR_d$.

Note the following: Assuming a search over 2^{39} hypotheses, cryptanalysis of LILI-128 can be reduced to the problem of LFSR initial state reconstruction with length equal to 89 and output available through a noisy channel.

The underlying assumptions for the security examination of LILI-128 are the following:
- according to the **author's claims** (from the proposal) the noise in the equivalent BSC model corresponds to $p = 0.46875$;
- the available output sequence (keystream) from LILI-128 corresponds to **one hour work**, so that at least $60^3 \cdot 4.8 \cdot 10^6 \sim 10^{12}$ bits are available.

Accordingly, the security evaluation is summarized in Table 6.

Table 6. Theoretical estimations of LILI-128 security level.

ESTIMATION	Complexity of divide and conquer attack
LILI-128 author's claim	$\sim 2^{112}$ operations
estimation based on the proposed algorithm	$\sim 2^{91}$ *mod*2 additions

Note that Table 6 implies that the security margin is reduced by 21 bits in comparison with the claim from the LILI-128 proposal [19].

Finally, it is interesting to compare the developed attack on LILI-128 with its cryptanalysis using the time-memory-data trade-off technique recently reported in [1]. Recall that the trade-off technique [1] is applicable to any keystream generator, and that it is an alternative to the exhaustive key search.

According to [1], an appropriate time-memory-data trade-off when the secret key consists of 128 bits have the following characteristics: (i) required data: $\sim 2^{43}$; (ii) pre-processing complexity: memory $\sim 2^{43}$ and time $\sim 2^{86}$; (iii) processing complexity: memory $\sim 2^{43}$ and time $\sim 2^{86}$ (*recalculation* + *disk read*), where the *recalculation* assumes all operations required for generation 128 LILI-128 output bits. Accordingly, complexity of 2^{86} *recalculations* is proportional to $4 \times 128 \times 2^{86} = 2^{97}$ $mod2$ *additions* (note that factor 4 is due to the employed clock-control operation in LILI-128).

Recall that the cost associated with the disk read operations has to be taken into account, as pointed-out and discussed in [1].

The characteristics of the attacking technique proposed in this paper are the following: (i) required data: $\sim 2^{38}$; (ii) pre-processing complexity: memory $\sim 2^{38}$ and time $\sim 2^{38}$; (iii) processing complexity: memory $\sim 2^{27}$ and time $\sim 2^{91}$ $mod2$ *additions*.

Accordingly, the proposed technique has the following advantages over the time-memory-data trade-off technique [1] in the case of LILI-128 cryptanalysis:
- significantly smaller processing memory;
- significantly smaller overall processing time complexity, particularly due the fact that the proposed approach does not require the expensive disk read operations as a consequence of the required size of processing memory;
- significantly smaller pre-processing complexity;
- shorter data sequence.

8 Conclusions

An improved method for the fast correlation attack on certain stream ciphers has been presented. The proposed algorithm employs two optimal decoding approaches: the list decoding where a candidate is assigned to the list based on the most reliable information sets, and MDD with Hamming distance.

Comparisons show that the proposed algorithm outperforms recently reported algorithms in several scenarios.

A desirable characteristic of the proposed algorithm is its theoretical analyzibility, so that its performance can also be estimated in the cases where the considered experiments are not currently realizable due to technological limitations.

Finally the proposed algorithm can be employed for security examination of the NESSIE proposal for stream cipher LILI-128.

9 Appendix

Elements for the proofs of Theorem 1 and Theorem 2. Assuming that a given number s of satisfied parity-check equations is a realization of a random integer variable S, the Bayes probability of decision error for each bit-i is given by

$$P_B(s) = \min\{\Pr(E = 1 \mid \mathbf{z}), 1 - \Pr(E = 1 \mid \mathbf{z})\}$$

$$= \min\{\Pr(E = 1 \mid S = s), 1 - \Pr(E = 1 \mid S = s)\} = \begin{cases} \frac{q}{1+q} \text{ if } q \leq 1 \ , \\ \frac{1}{1+q} \text{ if } q > 1 \ , \end{cases}$$

and the average Bayes probability of error under the condition Ψ that the random variable S takes values from the set $\{s_0, s_1, ..., s_n\}$ is given by

$$P_B(\Psi) = \sum_s P_B(s) \ \Pr(S = s|\Psi) \ ,$$

where,

$$\Pr(S = s|\Psi) = \frac{\Pr(\Psi|S = s) \Pr(S = s)}{\Pr(\Psi)} \ ,$$

$$\Pr(\Psi|S = s) = \begin{cases} 1, & \text{if } \ S \in \{s_0, s_1, ..., s_n\} \ , \\ 0, & \text{otherwise} \ , \end{cases}$$

$$\Pr(S = s) = p \binom{|\ \Omega\ |}{s} p_w^s (1 - p_w)^{|\Omega|-s} \ + (1 - p) \binom{|\ \Omega\ |}{s} (1 - p_w)^s p_w^{|\Omega|-s} \ ,$$

with $p_w = (1 - (1 - 2p)^2)/2$, and

$$\Pr(\Psi) = \sum_{s:s_0,s_1,...,s_n} \Pr(S = s) \ .$$

Accordingly, after some algebra we obtain each theorem statement.

References

1. A. Biryukov and A. Shamir, "Cryptanalytic Time/ Memory/ Data Tradeoffs for Stream Ciphers", Advances in Cryptology - ASIACRYPT2000, *Lecture Notes in Computer Science*, vol. 1976, pp. 1-13, 2000.
2. A. Canteaut and M. Trabbia, "Improved fast correlation attacks using parity-check equations of weight 4 and 5," Advances in Cryptology - EUROCRYPT'2000, *Lecture Notes in Computer Science*, vol. 1807, pp. 573-588, 2000.
3. V. V. Chepyzhov, T. Johansson and B. Smeets, "A simple algorithm for fast correlation attacks on stream ciphers," Fast Software Encryption - FSE2000, *Lecture Notes in Computer Science*, vol. 1978, pp. 180-195, 2001.
4. P. Elias, "List decoding for noisy channels," *Wescon Convention Record*, Part 2, Institute of Radio Engineers (now IEEE), pp. 94-104, 1957.
5. P. Elias, "Zero error capacity under list decoding," *IEEE Trans. Inform. Theory*, vol 34, pp. 1070-1074, Sept. 1988.
6. P. Elias, "Error-correcting codes for list decoding," *IEEE Trans. Inform. Theory*, vol 37, pp. 5-12, Jan. 1991.
7. M. P. C. Fossorier, M. J. Mihaljević and H. Imai, "Critical noise for convergence of iterative probabilistic decoding with belief propagation in cryptographic applications," Applied Algebra, Algebraic Algorithms and Error Correcting Codes - AAECC 13, *Lecture Notes in Computer Science*, vol. 1719, pp. 282-293, 1999.

8. T. Johansson and F. Jonsson, "Improved fast correlation attacks on stream ciphers via convolutional codes," Advances in Cryptology - EUROCRYPT'99, *Lecture Notes in Computer Science*, vol. 1592, pp. 347-362, 1999.
9. T. Johansson and F. Jonsson, "Fast correlation attacks based on turbo code techniques," Advances in Cryptology - CRYPTO'99, *Lecture Notes in Computer Science*, vol. 1666, pp. 181-197, 1999.
10. T. Johansson and F. Jonsson, "Fast correlation attacks through reconstruction of linear polynomials," Advances in Cryptology - CRYPTO2000, *Lecture Notes in Computer Science*, vol. 1880, pp. 300-315, 2000.
11. S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice Hall, 1983.
12. J. L. Massey, *Threshold Decoding*. Cambridge, MA: MIT Press, 1963.
13. A. Menezes, P.C. van Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography*. Boca Roton: CRC Press, 1997.
14. W. Meier and O. Staffelbach, "Fast correlation attacks on certain stream ciphers," *Journal of Cryptology*, vol. 1, pp. 159-176, 1989.
15. M. J. Mihaljević, M. P. C. Fossorier and H. Imai, "A low-complexity and high-performance algorithm for the fast correlation attack," Fast Software Encryption - FSE2000, *Lecture Notes in Computer Science*, vol. 1978, pp. 196-212, 2001.
16. M. J. Mihaljević, M. P. C. Fossorier and H. Imai, "An algorithm for cryptanalysis of certain keystream generators suitable for high-speed software and hardware implementations," *IEICE Trans. Fundamentals*, vol. E84-A, pp. 311-318, Jan. 2001.
17. M. J. Mihaljević, M. P. C. Fossorier and H. Imai, "On decoding techniques for cryptanalysis of certain encryption algorithms," *IEICE Trans. Fundamentals*, vol. E84-A, pp. 919-930, Apr. 2001.
18. M. J. Mihaljević and J. Dj. Golić, "A method for convergence analysis of iterative probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. 46, pp. 2206-2211, Sept. 2000.
19. NESSIE list of Accepted Submissions: Proposal for Synchronous Stream Cipher LILI-128, Nov. 2000, http://www.cosic.esat.kuleuven.ac.be/nessie.
20. T. Siegenthaler, "Decrypting a class of stream ciphers using ciphertext only," *IEEE Trans. Comput.*, vol. C-34, pp. 81-85, 1985.
21. J. M. Wozencraft,"List decoding," *Quarterly Progress Report*, vol. 48, pp. 90-95, Research Laboratory of Electronics, MIT, Jan. 15, 1958.
22. K. Zeng and M. Huang, "On the linear syndrome method in cryptanalysis," Advances in Cryptology - CRYPTO '88, *Lecture Notes in Computer Science*, vol. 403, pp. 469-478, 1990.