On Meier-Staffelbach's Fast Correlation Attack

Esen Akkemik, Orhun Kara, Ayşegül Kurşunlu

***Abstract*—In this paper, we study the experimental results of the algorithm given in Meier-Staffelbach's "Fast Correlation Attacks on Certain Stream Ciphers" paper. We study the speed of the attack in terms of the steps executed under fixed keystream lengths and correlation probabilities for Algorithm B. We also give a slightly modified version of Algorithm B which can find the linear feedback shift register outputs from keystreams where the original algorithm fails. We compare the performances of both algorithms in some specific cases. Experimental results show that for these specific cases the new algorithm is superior to the original algorithm. We present the correlation probability between the linear feedback shift register sequence and the resulting sequence at each step under various probabilities and keystream lengths. We show that the number of steps decreases by using longer keystream sequence or higher correlation probability as it is expected for the correlation attacks.**

***Index Terms*—Correlation attack, Fast correlation attack, Linear feedback shift register (LFSR), Stream cipher.**

I. Introduction

THE one-time pad (Vernam Cipher) is an encryption algorithm where a completely random key is xored with a plaintext to create a ciphertext. Since the key is random an attacker cannot recover the plaintext from the ciphertext unless he knows the key. Such an encryption algorithm is called perfectly secure. Unfortunately, it is not practical to generate completely random key streams of arbitrary lengths. Thus, pseudo-random sequences are used instead. One way to generate pseudo-random sequences is to use the output of a stream cipher. Linear feedback shift registers (LFSR) are the basic components of most stream ciphers, since they have efficient hardware implementations and produce sequences with good statistical properties.

Combination generators are one of the most common keystream generators based on LFSRs. Figure 1 shows a combination generator which has m linear feedback shift registers (LFSR) whose outputs are combined by a nonlinear Boolean function F with the desired properties [3]. Any keystream generator having LFSRs as a component is vulnerable to correlation attacks. This cryptanalytic technique introduced by Siegenthaler [2] is an example of "divide-and-conquer" methods. In the attack the correlation between the LFSR sequence a and the output sequence z of F, also known as keystream, is exploited. It is shown in [2] that if the keystream is correlated with at least one of the LFSR outputs, then a correlation attack against this (these) LFSR(s) will significantly reduce the complexity of exhaustive search. For example, Geffe generator with initial correlation probability $p = 0.75$, Pless generator with $p = 0.75$ and Brüer generator with p up to 0.75 have been broken with this attack if the LFSR length is smaller than 50 [2]. This attack is indeed one of the most severe analysis for LFSR-based generators.

Correlation attacks are the most common and effective divide-and-conquer type attacks mounted on keystream generators. There are several known stream ciphers, which are shown to be vulnerable to correlation attacks, such as A5/1 of GSM protocol, E0 of Bluetooth, LILI-128 and Grain.

In 2001 in [7], Ekdahl and Johansson introduced a correlation attack on A5/1. In 2004, Maximov et. al. introduced another attack on A5/1 [8] which is an improvement of the attack in [7]. This is a ciphertext-only attack which uses the redundancy during silence to get some known outputs from the cipher. Finally, in 2006 Barkan and Biham mounted correlation attack on A5/1 using conditional estimators [14]. A fast correlation attack on LILI-128 stream cipher is given in [9]. This attack is an application of the techniques given in [16] to LILI-128. A fast correlation attack on Bluetooth algorithm E0 is given In [12]. The authors apply convolution to the analysis of the distinguisher based on all correlations and propose an efficient distinguisher resulting from linear dependency of the largest correlations. In [13], a correlation based attack applied on the stream cipher Grain is given.

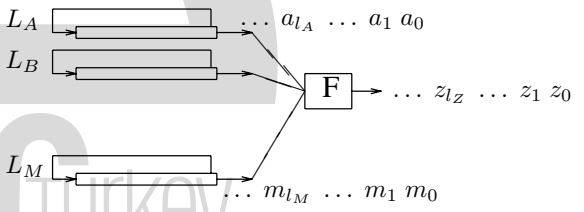

Fig. 1. A pseudo-random keystream generator

The fast correlation attack is firstly proposed by Meier and Staffelbach in [1]. In this attack, the keystream z is considered as a noisy version of the LFSR sequence a. It is assumed that N bits of the sequence z and the correlation probability between a and z are known. A necessary condition for this attack is that the number of tabs of L_A is small. In order to obtain a from z, firstly linear equations are obtained by using the feedback polynomial. Later, the bits of z are used in these equations. A new correlation probability is assigned to each bit by checking whether the bits of z satisfy those equations or not. If this new probability is high enough, then it is likely that this particular bit of z is equal to the corresponding bit of a. Several papers [10], [11], [15] improve the results of [1]. These algorithms are effective only if the feedback polynomial is of low weight. In [15], a new method for the computation of low weight parity checks based on the

E. Akkemik, O. Kara and A. Kurşunlu are with TÜBİTAK UEKAE, Gebze, Kocaeli, TURKEY email: {esena, orhun, akursunlu}@uekae.tubitak.gov.tr
E. Akkemik is also with Institute of Applied Mathematics, METU Ankara, TURKEY
Manuscript received September 21, 2007; revised November 23, 2007.

theory of error decoding codes is given. The LFSR sequence is considered as a codeword of a maximal-length block code. By applying the decoding of cyclic block codes Penzhorn develop an efficient algorithm for the computation of low-weight parity check equations that will be used in Meier-Staffelbach algorithm. In [10], Mihaljević and Golić proposed a new algorithm for reconstruction of LFSR sequence given the keystream. They use the finite state matrix representation of an LFSR and iterative error correction. This new algorithm works for large number of feedback taps and for smaller correlation probability under the condition that large number of keystream bits is observed. In [11], the authors propose another technique for fast correlation attack where they use the keystream bits efficiently by using different iterative decoding methods.

After the introduction of correlation attacks by using low-weight parity check codes by Meier and Staffelbach, several variations of correlation attacks emerged which used different codes and decoding algorithms. In 1999, Johansson and Jönsson suggested to use convolutional codes to increase the performance of fast correlation attacks [4]. This new version of fast correlation attack can be applied to LFSRs with any feedback polynomials. They show that a low weight convolutional code can be found by the code generated from the LFSR sequences. This convolutional code is decoded by using a low complexity decoding algorithm. A correctly decoded sequence will give the output sequence of the LFSR. In [5], the same authors study the theoretical background of this algorithm. They find the relationship between the correlation probability, the length of the keystream, LFSR length and the code parameters.

After the introduction of convolutional codes in fast correlation attacks, Chepyzhov et. al. introduced another method in [11]. In this paper, they find another linear code associated with the LFSR sequence. They calculate the proper parity checks for the new code. The keystream bits are combined according to the parity checks, and the probability of each codeword in the code is calculated. This process corresponds to ML-decoding. The decoding of the code leads to the initial state of the LFSR.

Johansson and Jönsson later introduced new algorithms for fast correlation attacks based on turbo codes in [6]. These algorithms are based on the iterative decoding techniques with the embedded convolutional codes. The authors identify parallel embedded convolutional codes by considering the permuted versions of the code generated by the LFSR sequence. These codes have the same information sequence, but have different parity checks. Later, the keystream is used to construct the sequences that will serve as the received sequences for the codes determined in the previous stage. These sequences are then used to find the correct information sequence by an iterative decoding technique.

In this paper we study the experimental results of Algorithm B given in [1]. We investigate the speed of the algorithm in terms of the steps executed during the whole process. We study the speed under different correlation probabilities and keystream lengths. We observe that the original setting of Algorithm B cannot deduce the LFSR sequence a for some specific cases. We propose a new algorithm, a modified version of Algorithm B, which can deduce a in these cases.In addition, we show several experimental results regarding to the performances of both algorithms.

This paper is organized as follows. In Section II the statistical model of the attack is given. In Section III, we give the details of the two algorithms given in [1]. Also, a modified version of Algorithm B is given in this section. In Section IV the experimental results of the original and the modified versions of Algorithm B are given. The comparison of the performances of the original and the new algorithms are given in Section V. We give the summary of the results in VI.

II. Statistical Model

Assume that LFSR A (L_A) with a length of k and number of tabs t are given. Let the output sequence of L_A be the sequence a. Recall that a is given by a linear relationship of the form

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k} \tag{1}$$

where $c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_k x^k$ is called the feedback polynomial of the relation. The sequence z can be seen as a perturbation of the LFSR sequence a by a binary symmetric channel with a probability p of flipping each bit. The attack is based on obtaining posterior probabilities for each bit by the a priori probability (p). These probabilities indicate the probability that a certain bit of z is equal to the corresponding bit of a. The bits of z whose probabilities are less than a given threshold are flipped.

In order to recover the LFSR sequence a from z, the bits of the sequence z are replaced in the linear equations derived from the feedback polynomial of LFSR A (L_A). Linear equations of a for a fixed bit a_n are given as follows:

$$\begin{aligned} L_1 &= a_n + b_1 = 0 \\ L_2 &= a_n + b_2 = 0 \\ &\vdots \\ L_m &= a_n + b_m = 0, \end{aligned} \tag{2}$$

where each b_i $(i = 1, \ldots, m)$ is a sum of t different bits of a. Thus, a number of linear equations are obtained for each bit z_n $(n = 1, \ldots, N)$ of z as

$$\begin{aligned} L_1 &= z_n + y_1 = 0 \\ L_2 &= z_n + y_2 = 0 \\ &\vdots \\ L_m &= z_n + y_m = 0, \end{aligned} \tag{3}$$

where each y_i $(i = 1, \ldots, m)$ is a sum of t different bits of z. The equations are satisfied if $z_n = a_n$ and $y_i = b_i$ or $z_n \neq a_n$ and $y_i \neq b_i$. We know that $\Pr(z_n = a_n) = p$. Let s be the probability that $y_i = b_i$ and can be calculated by the following recursive formula:

$$\begin{aligned} s(p,t) &= ps(p,t-1) + (1-p)(1-s(p,t-1)) \\ s(p,1) &= p. \end{aligned} \tag{4}$$

For each bit z_n of z, the number of the equations satisfied in Eq.3 is determined. We denote this number by h. Then, we calculate the conditional probability of $z_n = a_n$ given that h equations are satisfied by

$$P(z_n = a_n | h \text{ eqs. hold}) = \frac{ps^h(1-s)^{m-h}}{ps^h(1-s)^{m-h} + (1-p)(1-s)^h s^{m-h}}, \quad (5)$$

The basic idea behind the algorithms given in [1] is to use posterior probabilities given in Eq.5 to obtain a from z.

III. Fast Correlation Attack Algorithms

Two algorithms given in [1] and a slightly modified version of Algorithm B are given in this section.

A. Algorithm A

In Algorithm A, the correct bits satisfying the linear equations are searched. This process is done by selecting the bits that satisfy sufficient equations. In this way, an estimate of the sequence a is obtained at the corresponding bit positions. Thus, small modifications on the estimated sequence will lead to the original sequence a.

The steps of Algorithm A is as follows:

1. Determine the expected number of the equations m by using the formula
$$m(N,k,t) \approx log(\frac{N}{2k})(t+1). \quad (6)$$
2. Find the maximum value of h such that $Q \cdot N \geq k$, where Q is the probability that at least h of m equations are satisfied and computed by
$$Q = \sum_{i=h}^{m} \binom{m}{i} \left(ps^i(1-s)^{m-i} + (1-p)(1-s)^i s^{m-i}\right), \quad (7)$$
where s is given in 4.
3. Compute the new probabilities p^* according to Eq.5 for the bits of z and choose the bits having the highest new probabilities as a reference guess I_0 of a at the corresponding bit positions.
4. Find the correct guess by making modifications of I_0 by changing $0, 1, 2, \ldots$ bits and testing the correlation of the resulting LFSR sequence with z.

B. Algorithm B

In Algorithm B, the bits of z are considered together with the probability of being correct. It is known that the correlation probability between z and a is p. Each bit z_n of z is assigned a new probability p^*, which is the probability that $z_n = a_n$ and this probability depends on the number of equations satisfied or not satisfied by z_n. This conditional probability is computed by Eq.5. The conditional probability for $z_n \neq a_n$ can be computed similarly.

Calculating the conditional probabilities is repeated α times with a new p^* at each step. In [1], it is mentioned that $\alpha = 5$ is a suitable choice for many cases. After α times re-computations of p^*, all bits having the probability p^* lower than a certain threshold are complemented. At the end of each step (item 3-7 below), a new z sequence is obtained. It is expected that the number of wrong bits in z will decrease at each step. These steps are repeated several times with a new z sequence and it is expected that this algorithm will end up with the correct LFSR sequence a.

The steps of Algorithm B is as follows:

1. Determine the expected number m of the equations using Eq. 6 .
2. Find the value of h which maximizes I and call it h_{max}. I is the difference between the probability that at most h equations are satisfied when $z_n \neq a_n$ and the probability that at most h equations are satisfied when $z_n = a_n$. I can be obtained by the formula
$$I = \sum_{i=0}^{h} \binom{m}{i} \left((1-p)(1-s)^i s^{m-i} - ps^i(1-s)^{m-i}\right). \quad (8)$$
Then, calculate p_{thr} and N_{thr} by using h_{max} as follows:
$$p_{thr} = \frac{1}{2}\left(p^*(p,m,h) + p^*(p,m,h+1)\right), \quad (9)$$
$$N_{thr} = U(p,m,h) \cdot N. \quad (10)$$
p^* is calculated according to the Eq. 5. U is the probability that at most h of m equations are satisfied and is calculated as follows:
$$U = \sum_{i=0}^{h} \binom{m}{i} \left(ps^i(1-s)^{m-i} + (1-p)(1-s)^i s^{m-i}\right). \quad (11)$$
3. Initialize the iteration counter $i = 0$.
4. For every bit of z, compute the new probability p^* given in Eq. 5. Determine the number N_w of the bits with $p^* < p_{thr}$.
5. If N_w is smaller than the expected number of the bits with $p^* < p_{thr}$ or $i < \alpha$ increment i and go to 4.
6. Complement the bits of z with $p^* < p_{thr}$ and set the probability of each bit to the original probability p.
7. If there are still bits not satisfying the linear relation of LFSR sequence a, go to 3.
8. Terminate with $a = z$.

C. Modified Algorithm B

For some correlation probabilities close to 0.5 Algorithm B needs longer output sequences. If the length of the output sequence N is not long enough, Algorithm B fails to find a. In this case all p^* values are greater than p_{thr}, so N_w, which is the number of bits to be complemented, becomes 0. On the other hand, a cannot be obtained by the algorithm, because there are still bits not satisfying the recurrence relations of the LFSR. If this is the case, we propose to make a small change in Algorithm B as follows:

4′. For every bit of z, compute the new probability p^* with respect to the number of the equations satisfied. Determine the number N_w of the bits with $p^* < p_{thr}$. If $N_w = 0$, decrease α by 1. If $\alpha = 0$ algorithm fails.

Figure 2 shows the comparison of N_w values for the original and the modified versions of Algorithm B for $p = 0.56$ and

$N = 70000$. As it is seen from this figure, Algorithm B oscillates in N_w values after $N_w = 0$. This situation results in the failure of Algorithm B to find a. On the other hand, whenever $N_w = 0$ the modified algorithm starts to decrease α and change the value of N_w at each step and thus finds a.

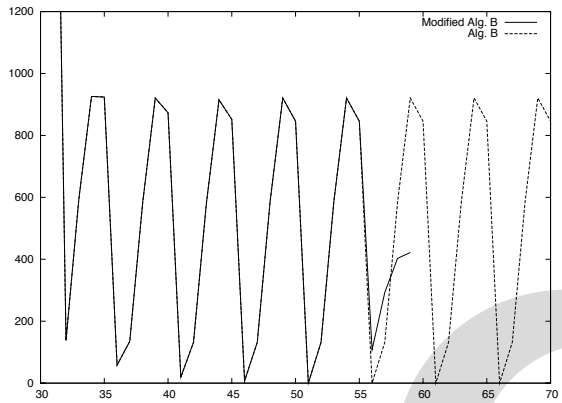

Fig. 2. Comparison of N_w values for the original and the modified Algorithm B for $p = 0.56$ and $N = 70000$

Table I compares the original and the modified versions of Algorithm B with respect to the number of bits that become true or false after flipping at each step for the correlation probability $p = 0.56$ and $N = 70000$. The first column in the table is the number of steps executed. The second and the third column is the number of bits that become false and true, respectively, after flipping in the original Algorithm B. The fourth and the fifth columns are the corresponding columns for the modified version of Algorithm B. As it is seen from Table I, after step 14 Algorithm B cannot find any bits to flip, whereas the modified algorithm continues the process and finds the LFSR sequence at step 16.

IV. Experimental Results

In the numerical experiments the following primitive feedback polynomial is chosen for L_A,

$$c(X) = 1 + X^2 + X^{35}. \tag{12}$$

The length of the LFSR is 35 and the number of tabs is 2.

Figure 3 and Figure 5 show the correlation between a and z at each step for different keystream lengths for the initial correlation probabilities $p = 0.56$ and $p = 0.60$, respectively, with the original Algorithm B. As it is seen in Figure 3, if $N \leq 70000$, the original Algorithm B fails to find sequence a for $p = 0.56$. Figure 5 shows that if $N \leq 9500$, it fails to find sequence a for $p = 0.60$.

Figure 4 and Figure 6 show the correlation between a and z at each step for different keystream lengths when using the modified algorithm. In the figures the initial correlation is taken as $p = 0.56$ and $p = 0.60$, respectively. As it is seen from the Figure 4, for $p = 0.56$ the modified version successfully finds the sequence a for $N > 30000$. For $N = 30000$ and

TABLE I
Number of bits that become true or false after flipping for $p = 0.56$ and $N = 70000$

Step Number	Algorithm B		Modified Alg. B	
	False	True	False	True
1	2073	2182	2073	2182
2	2153	2384	2153	2384
3	3206	3479	3206	3479
4	3047	3474	3047	3474
5	1993	2935	1993	2935
6	1335	4512	1335	4512
7	539	5514	539	5514
8	233	8484	233	8484
9	87	8616	87	8616
10	0	2674	0	2674
11	0	58	0	58
12	0	19	0	19
13	0	7	0	7
14	0	0	0	0
15	0	0	8	838
16	0	0	0	422
17	0	0		
18	0	0		
19	0	0		

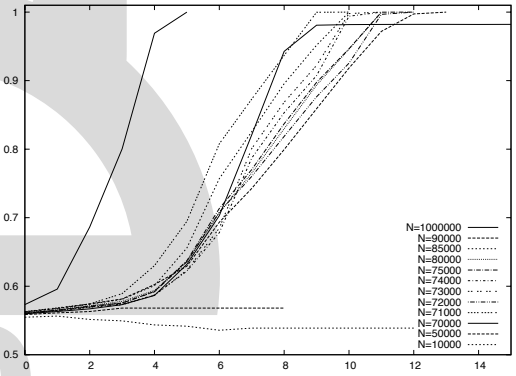

Fig. 3. The correlation between a and z at each step for different keystream lengths N and the initial correlation $p = 0.56$ with the original Algorithm B, number of tabs is 2, $\alpha = 5$.

$N = 20000$ the modified algorithm finds a sequence which is not equal to a. We conclude that decreasing α, while being able to find sequences that the standard algorithm fails to find, is not applicable in all cases. Figure 6 shows that the modified version can find the sequence a for $N > 4250$, for $p = 0.60$.

V. Convergence Speed of Algorithm B

In this section, we give the experimental results of the original Algorithm B to show the relationship between the number of steps executed to obtain the LFSR sequence a and the length of the output sequence N, and the correlation probability p.

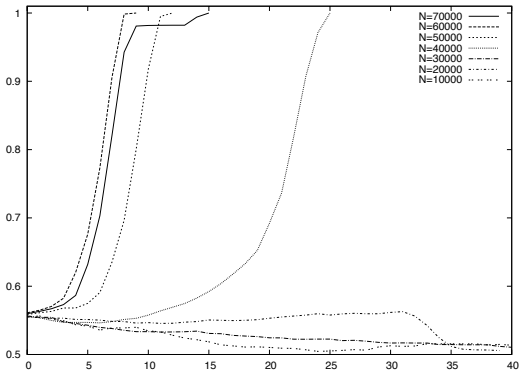

Fig. 4. The correlation between a and z at each step for different keystream lengths N and the initial correlation $p = 0.56$, number of the tabs is 2 with the modified algorithm

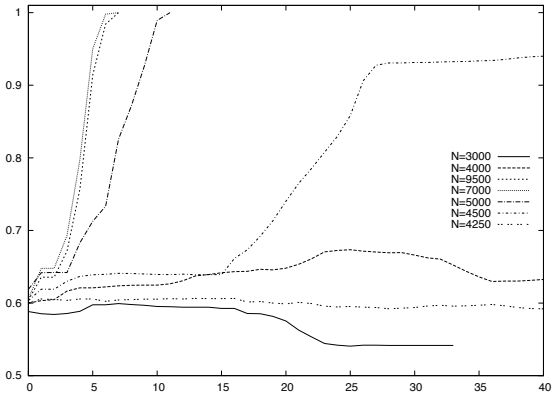

Fig. 6. The correlation between a and z at each step for the initial correlation $p = 0.60$ and different keystream lengths N with the modified algorithm

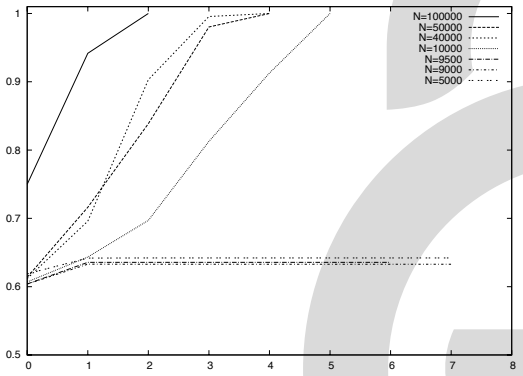

Fig. 5. The correlation between a and z at each step for the initial correlation $p = 0.60$ and different keystream lengths N with the original Algorithm B

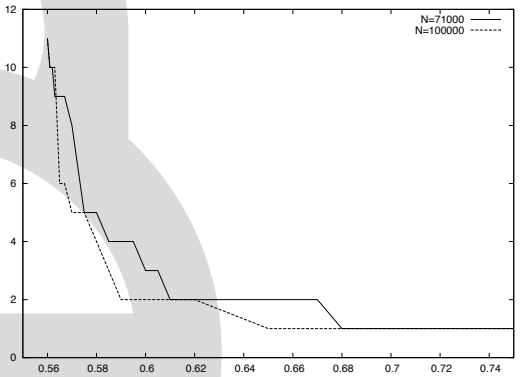

Fig. 7. Number of steps for different correlation probabilities in the original Algorithm B for the output lengths N=71000 and N=100000

A. The Effect of the Keystream Length on Convergence

If the length of z is increased, there will be more equations to be checked. Thus, the decision that $a_n = z_n$ will be more accurate. So, we expect that if the length of the output sequence is increased in the original Algorithm B, the number of steps executed to obtain the LFSR sequence a will be decreased. This is confirmed in Figure 7.

Figure 8 and Figure 9 show that for fixed correlation probabilities $p = 0.56$ and $p = 0.60$, respectively, the number of steps decreases while the length of the keystream increases.

B. The Effect of the Correlation Probability on Convergence

If tests are done on sequences with larger correlation probabilities, the number of steps will decrease, since $\Pr(a_n = z_n)$ will become larger at each step. Figure 10 shows the number of steps for $p = 0.56$ and $p = 0.60$ for different keystream lengths.

Figure 11 and Figure 12 show that for fixed keystream lengths $N = 71000$ and $N = 100000$, respectively, the number of steps decreases when the correlation probability increases.

VI. Conclusion

We analyze Algorithm B in Meier-Staffelbach's fast correlation attack. For some keystream lengths and correlation probabilities this algorithm fails to find the LFSR sequence. We propose to decrease the iteration number α in the algorithm. We show that decrease in α makes the attack successful for shorter keystream lengths. However, in some cases this modification may result in a completely different sequence than the LFSR sequence. We also analyze the speed of the algorithm under fixed correlation probabilities and keystream lengths. For higher correlation probabilities or longer keystream lengths a decrease in the steps executed to find the LFSR sequence is observed as it is expected.

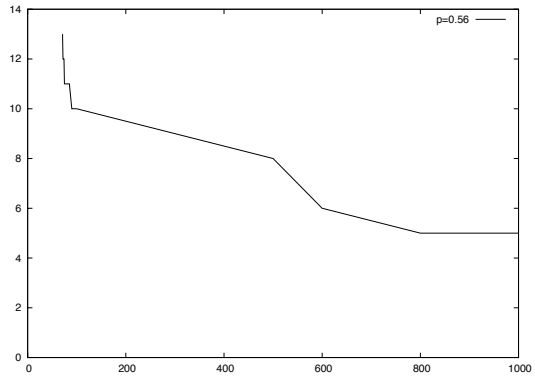

Fig. 8. Number of steps for different output lengths (in thousand) in the original Algorithm B for the correlation probability $p = 0.56$

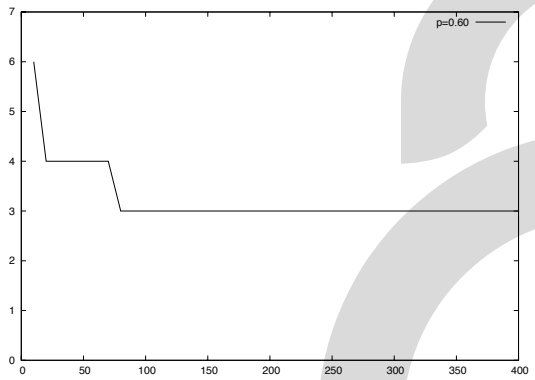

Fig. 9. Number of steps for different output lengths (in thousand) in the original Algorithm B for the correlation probability $p = 0.60$

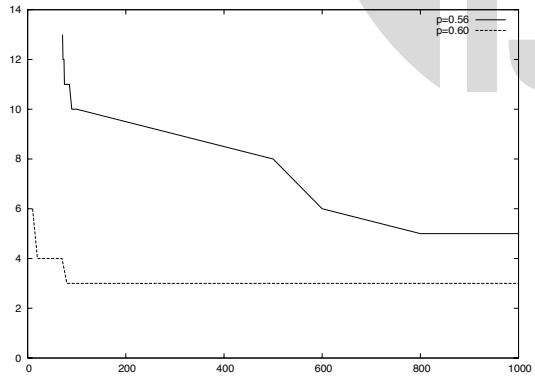

Fig. 10. Number of steps for different output lengths (in thousand) in the original Algorithm B for the correlation probabilities $p = 0.56$ and $p = 0.60$

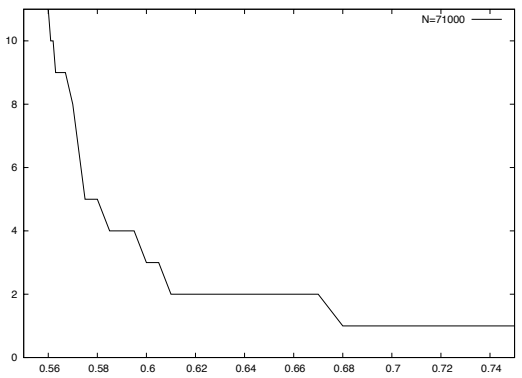

Fig. 11. Number of steps for different correlation probabilities in the original Algorithm B for the output length N=71000

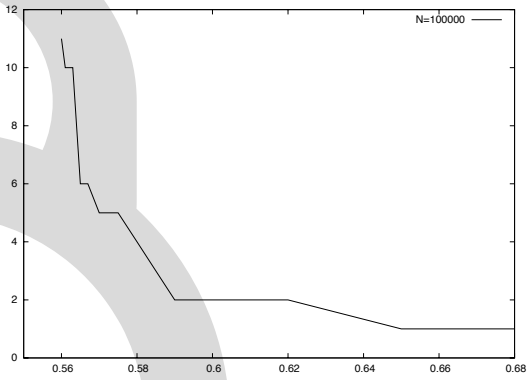

Fig. 12. Number of steps for different correlation probabilities in the original Algorithm B for the output length N=100000

ACKNOWLEDGMENTS

We would like to thank our colleague Cevat Manap for his help during this work.

REFERENCES

[1] W. Meier and O. Staffelbach, "Fast Correlation Attacks on Certain Stream Ciphers", *Journal of Cryptology*, Vol.1, 1989, pp. 159–176.

[2] T. Siegenthaler, "Decrypting a Class of Stream Ciphers Using Ciphertext Only", *IEEE Transactions on Computers*, Vol. C-34, No. 1, 1985, pp. 81–85.

[3] A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, 1st Edition, USA, CRC Press, 1997, pp. 205–208.

[4] T. Johansson and F. Jönsson, "Improved Fast Correlation Attacks on Stream Ciphers via Convolutional Codes", *Advances in Cryptology - EUROCRYPT'99, Lecture Notes in Computer Science*, Vol. 1592, 1999, pp. 347–362.

[5] F. Jönsson and T. Johansson, "Theoretical analysis of a correlation attack based on convolutional codes", in *Proc. IEEE International Symposium on Information Theory 2000*, Sorrento, Italy, 2000, p. 212.

[6] T. Johansson and F. Jönsson, "Fast correlation attacks based on turbo code techniques ", *Advances in Cryptology - CRYPTO'99, Lecture Notes in Computer Science*, Vol. 1666, 1999, pp. 181–197.

[7] P. Ekdahl and T. Johansson, "Another Attack on A5/1", *IEEE Transaction on Information Theory*, Vol. 49, No.1, 2003, pp. 284–289.

[8] A. Maximov and T. Johansson, "An Improved Correlation Attack on A5/1 ", *Advances in Cryptology - SAC'04, Lecture Notes in Computer Science*, Vol. 3357, 2004, pp. 1–18.
[9] F. Jönsson and T. Johansson, "A Fast Correlation Attack on LILI-128", *Information Processing Letters*, Vol. 81, No. 3, 2002, pp. 127–132.
[10] M. Mihaljevic and J. Dj. Golić, "A Fast iterative Algorithm for a Shift Register Initial State Reconstruction Given the Noisy Output Sequence", *Advances in Cryptology - AUSCRYPT'90, Lecture Notes in Computer Science*, Vol. 483, 1990, pp. 165–175.
[11] V. Chepyzhov and B. Smeets, "On a Fast Correlation on Stream Ciphers",*Advances in Cryptology–EUROCRYPT'91, Lecture Notes in Computer Science*, Vol. 547, 1991, pp. 176–185.
[12] Y. Lu and S. Vaudenay, "Faster Correlation Attack on Bluetooth Keystream Generator E0", *Advances in Cryptology - CRYPTO'04, Lecture Notes in Computer Science*, Vol. 3152, 2004, pp. 407–425.
[13] C. Berbain, H. Gilbert and A. Maximov, "Cryptanalysis of Grain", *Advances in Cryptology - FSE'06, Lecture Notes in Computer Science*, Vol. 4047, 2006, pp. 15–29.
[14] E. Barkan and E, Biham, "Conditional estimators:An Effective Attack on A5/1", *Advances in Cryptology - SAC'05, Lecture Notes in Computer Science*, Vol. 3897, 2006, pp. 1–19.
[15] W. Penzhorn, "Correlation Attack on Stream Ciphers: Computing Low Weight Parity Checks Based on Error Correcting Codes", *Advances in Cryptology - FSE'96, Lecture Notes in Computer Science*, Vol. 1039, 1996, pp. 159–172.
[16] V. Chepyzhov, T. Johansson and B. Smeets, "A Simple Algorithm for Fast Correlation Attacks on Stream Ciphers", *Advances in Cryptology - FSE'2000, Lecture Notes in Computer Science*, Vol. 1978, 2001, pp. 124–135.