An Improved Correlation Attack on A5/1

Alexander Maximov[1], Thomas Johansson[1], and Steve Babbage[2]

[1] Dept. of Information Technology, Lund University, Sweden
[2] Vodafone Group R&D, UK

Abstract. A new approach to attack A5/1 is proposed. The proposed attack is a refinement of a previous attack by Ekdahl and Johansson. We make two important observations that lead to a new attack with improved performance.

1 Introduction

The security of GSM conversation is based on usage of the A5 family of stream ciphers. Many hundred million customers in Europe are protected from the over-the-air piracy by the stronger version in this family, the A5/1 stream cipher. Other customers on other markets use the weaker version A5/2. The approximate design of A5/1 was leaked in 1994, and in 1999 the exact design of both A5/1 and A5/2 was discovered by Briceno [1]. As the result, a lot of investigations of the A5 stream ciphers were done.

The first analysis of the A5/1 cipher resulted in "Guess-and-Determine" type of attacks [2]. Then a time-memory trade-off attack was proposed by Biryukov, Shamir, and Wagner [3], which in some cases can break A5/1 in seconds. Unfortunately, it needs to use a huge precomputational time and about 4×73Gb of hard memory. The attack complexity grows exponentially depending on the length of the LFSRs in the design of the cipher. Another attack was presented by Biham and Dunkelman [4]. Their attack breaks the cipher within $2^{39.91}$ A5/1 clocking assuming $2^{20.8}$ bits of keystream available. This attack has expensive assymptotic behaviour. In 2002, Krause, [5] presented a general attack on LFSR-based stream ciphers, called BDD-based cryptanalysis. This attack requires computation complexity of $n^{O(1)}2^{an}, a < 1$ polynomial time operations, where a is a constant depending on the cipher and n is the combined shift registers length. For A5/1, the attack achieves $a = 0.6403$, so the complexity is again exponential in the shift registers length.

A completely different way to attack A5/1 was proposed by Ekdahl and Johansson in 2001 [6]. The attack needs a few minutes for computations, and 2-5 minutes of conversation (plaintext). The idea behind the attack came from correlation attacks. This is the only attack for which the complexity does not grow exponentially with the shift register length.

Finally, very recently Barkan, Biham and Keller [7] investigated the usage of the A5 ciphers in GSM. They demonstrated an active attack where a false base station can intercept a conversation and perform a man in the middle attack. By asking for usage of the weak A5/2 algorithm in the conversation with the

H. Handschuh and A. Hasan (Eds.): SAC 2004, LNCS 3357, pp. 1–18, 2005.

base station and then breaking it, the false base station finds the session key which is also used in the A5/1 protected conversation with the mobile unit. In [7] the authors also propose the passive memory-time trade-off ciphertext only attack. As one of the examples, if 5 minutes of conversation is available, then the attack needs one year of precomputations with 140 computers working together, 22×200GBs hard discs. Then the attack can be done in time 2^{28} by one PC. Obviously, the authors did not try to implement the attack and the complexity was just estimated.

In this paper a new approach to attack the A5/1 stream cipher is proposed. We consider the Ekdahl-Johansson attack as the basis, and apply several new improvements. As the result, the new attack now needs only less then 1 minute of computations, and a few seconds of known conversation. It does not need any notable precomputation time, and needs reasonable space of operation memory.

For the case of a ciphertext-only attack on A5/1, we use the fact that some redundancy is part of the plaintext. There are at least two kinds of redundancy that are explicit and may be used in an attack where only ciphertext is available. *The first* kind is the fact that coding is done before encryption, which results in linear relationships in the plaintext since the parity check symbols are also encrypted. This observation was used in [7]. *The second* kind of redundancy is the fact that during silence, a special frame including a large number of zeros is sent [8]. Silence occurs very often, but unfortunately these frames used for silence are transmitted less frequently, one to initialise a period of silence and then two each second. The attack that we propose can be considered in a ciphertext-only scenario, in which case we use this redundancy during silence to get some known outputs from the cipher.

Although several of the previous attacks are sufficient to break A5/1 in a known plaintext attack, we believe that further progress is very important. The A5/1 stream cipher is perhaps the most used cipher in the world, and from the wireless communication channel interception of the communication is very easy. Mobile base stations are not expensive to buy and they can be used to record GSM conversations.

The paper is organized as follows. In Section 2 a short description of the cipher A5/1 is given. The basic Ekdahl-Johansson attack on A5/1 is briefly described in Section 3. Then, in Section 4, we give new ideas to improve the attack in general. The details and particulars of the attack simulations are described in Section 4.2. Then in Section 5 the results of our simulations are presented.

2 Description of A5/1

A GSM conversation between A and B is a sequence of frames, each sent in about 4.6 milliseconds. Each frame consists of 228 bits – 114 bits of which is the message from A to B, and the second half bits are representing communication from B to A. One session is encrypted with a secret *session key* K. For the jth frame the running key generator is initialised with mixture of K and the publicly known *frame counter*, denoted by F_j. It then generates 228 bits of running key

for the current frame. The ciphertext is a binary xor of the running key and the plaintext.

A5/1 consists of 3 LFSRs of lengths 19, 22, and 23, which are denoted R_1, R_2, and R_3, respectively. The LFSRs are clocked in an irregular fashion. Each of them has one tap-bit, C_1, C_2, and C_3, respectively. In each step, 2 or 3 LFSRs are clocked, depending on the current values of the bits C_1, C_2, and C_3. Thus, the clocking control device implements the majority rule, shown in the table on the right. Note, for each step the probability that an individual LFSR is being clocked is 3/4.

values of			clocking		
C_1	C_2	C_3	R_1	R_2	R_3
$1 \oplus c$	c	c	×	√	√
c	$1 \oplus c$	c	√	×	√
c	c	$1 \oplus c$	√	√	×
c	c	c	√	√	√

After the initialisation procedure for the LFSRs, 228 bits of running key are produced, using irregular clocking. In each step one bit of the running key is calculated as the binary xor of the current output bits from the LFSRs.

The initialisation process uses the session key K and the known frame counter F_n. First the LFSRs are initialised to zero. They are then clocked 64 times, ignoring the irregular clocking, and the key bits of K are consecutively xored in parallel to the feedback of each of the registers. In the second step the LFSRs are clocked 22 times, ignoring the irregular clocking, and the successive bits of F_n are again xored in parallel to the feedback of the LFSRs. Let us call the state of LFSRs at this time the *initial state* of the frame. In the third step the LFSRs are clocked 100 times *with* irregular clocking, but ignoring outputs. Then, the LFSRs are clocked 228 times with the irregular clocking, producing 228 bits of the running key. For a more detailed description of A5/1 we refer to [1].

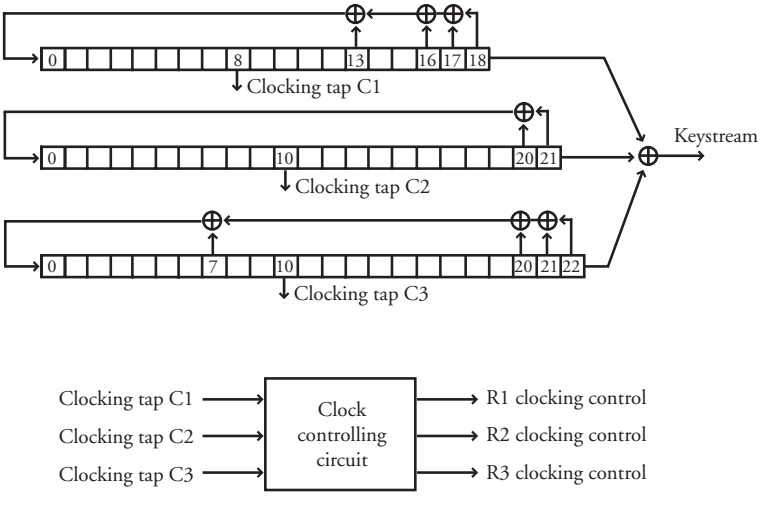

Fig. 1. The structure of A5/1 cipher

3 A Short Description of the Ekdahl-Johansson Attack on A5/1

This attack was proposed in 2002 by Ekdahl and Johansson. The idea behind the attack came from correlation attacks, and is based on the linearity of the initialisation procedure. The attack needs a set of m frames (about 20000-50000 in their attack), during one session, i.e., when the session key K is not changed.

For notation purposes, let the key $K = (k_1, \ldots, k_{64})$, and the frame counter $F_j = (f_1, \ldots, f_{22})$, where $k_i, f_j \in \mathbf{F}_2$, $i = 1..64, j = 1..22$. Denote by $u_1^j(l_1)$, $u_2^j(l_2)$, and $u_3^j(l_3)$ the output bits of LFSRs, if they are independently *clocked* l_1, l_2, and l_3 times, respectively, *after* the LFSRs being in the initial state, and when the current frame is number j. The 228 bits of the running key are then denoted as $v^j(101), \ldots, v^j(100+228)$, and every $v^j(t) = u_1^j(l_1) \oplus u_2^j(l_2) \oplus u_3^j(l_3)$, for *some unknown* l_1, l_2, l_3.

Note, that $u_1^j(l_1)$ is a linear combination of K and F_j bits, since all operations before the initial state are linear. I.e., $u_1^j(l_1)$ can be represented as $u_1^j(l_1) = X_{1,l_1}(F_j) + Y_{1,l_1}(K)$, where $X_{1,l_1}(F_j)$ is a known fixed value and $Y_{1,l_1}(K) = \sum_{i=1}^{64} y_{1,l_1,i} \cdot k_i$ is a linear function with known coefficients $y_{1,l_1,i} \in \mathbf{F}_2$.

With the same arguments we define

$$u_1^j(l_1) = X_{1,l_1}(F_j) + Y_{1,l_1}(K),$$
$$u_2^j(l_2) = X_{2,l_2}(F_j) + Y_{2,l_2}(K),$$
$$u_3^j(l_3) = X_{3,l_3}(F_j) + Y_{3,l_3}(K),$$

where $X_{a,l_a}(F_j)$ and the coefficients $y_{a,l_a,i} \in \mathbf{F}_2$, for $a = 1, 2, 3$, $l_a = 0, 1, \ldots, 100+228$, $i = 1, \ldots, 64$ are precomputed and fixed. Let us write

$$s_1(l_1) = Y_{1,l_1}(K), \qquad s_2(l_2) = Y_{2,l_2}(K), \qquad s_3(l_3) = Y_{3,l_3}(K). \tag{1}$$

Our target is to estimate 19 bits from the first LFSR $s_1(0), \ldots, s_1(18)$, 22 bits from the second LFSR $s_2(0), \ldots, s_2(21)$, and 23 bits from the third LFSR $s_3(0), \ldots, s_3(22)$. These 64 bits map one-to-one to 64 bits of the key K, if the frame counter F_j is given.

For notation purposes we write $E \stackrel{p}{=} \hat{E}$, when $\hat{E}$ appears to be an estimator for E, such that $\Pr\{E = \hat{E}\} = p$, for some probability p. $\hat{E}$ can be derived from accessible data, or assumed (guessed).

One can think about the data we have access to as a binary table of m frames in the form

$$\begin{pmatrix} v^1(101) & v^1(102) & \ldots & v^1(100+228) \\ v^2(101) & v^2(102) & \ldots & v^2(100+228) \\ & & \vdots & \\ v^m(101) & v^m(102) & \ldots & v^m(100+228) \end{pmatrix}.$$

The idea behind the attack is to observe that $v^j(101) \stackrel{p}{=} s_1(l_1) + s_2(l_2) + s_3(l_3) + X_{1,l_1}(F_j) + X_{2,l_2}(F_j) + X_{3,l_3}(F_j)$ for some $p \neq 1/2$, if l_1, l_2, l_3 are chosen properly. The probability $p = \frac{1}{2} + \frac{1}{2}\Pr\{(l_1, l_2, l_3) \text{ at time } t\}$, where $\Pr\{(l_1, l_2, l_3) \text{ at time } t\}$ is the probability that at time 101 the LFSRs were

regularly clocked exactly l_1, l_2, l_3 times, respectively. The probability that at time $t \in \{101 \dots 100 + 228\}$, the LFSRs have been clocked (l_1, l_2, l_3) times is

$$\Pr\{(l_1, l_2, l_3) \text{ at time } t\} = \frac{\binom{t}{t-l_1}\binom{t-(t-l_1)}{t-l_2}\binom{t-(t-l_1)-(t-l_2)}{t-l_3}}{4^t}. \quad (2)$$

Let us now define the known value $\hat{O}^j_{l_1,l_2,l_3}(t) = v^j(t) \oplus X_{1,l_1}(F_j) \oplus X_{2,l_2}(F_j) \oplus X_{3,l_3}(F_j)$. Then

$$\hat{O}^j_{l_1,l_2,l_3}(t) \stackrel{p}{=} s_1(l_1) \oplus s_2(l_2) \oplus s_3(l_3). \quad (3)$$

The case when $\hat{O}^j_{l_1,l_2,l_3}(t)$ is equal to the value $s_1(l_1) \oplus s_2(l_2) \oplus s_3(l_3)$ can happen only in two ways,

a) The LFSRs are really clocked l_1, l_2, l_3 at time t, happening with probability $\Pr\{(l_1, l_2, l_3) \text{ at time } t\}$. If so, the expression will be true with probability 1.
b) If the condition in a) is not fulfilled, the expression will still be true with probability 1/2.

This means that the relation (3) is biased ($p > 1/2$).

From the given frames we can estimate many of the linear combinations $s_1(l_1) \oplus s_2(l_2) \oplus s_3(l_3)$ for different triples (l_1, l_2, l_3). But we only need 64 correct estimates in order to recover the key K uniquely.

To minimise the amount of frames m and perform the estimation with low probability of error, Ekdahl and Johansson suggested to use the values of $v^j(101), \dots, v^j(164)$ for all j for better estimation of $s_1(l_1) \oplus s_2(l_2) \oplus s_3(l_3)$. They used the expression

$$\begin{aligned}\Pr\{s_1(l_1) \oplus s_2(l_2) \oplus s_3(l_3) = 1, \text{ for the frame } j\} = p^j_{(l_1,l_2,l_3)} = \\ = \sum_{t \in \{101 \dots 164\}} \Pr\{(l_1, l_2, l_3) \text{ at time } t\} \cdot \left[\hat{O}^j_{l_1,l_2,l_3}(t) = 0\right] \\ + 1/2 \cdot \left(1 - \sum_{t \in \{101 \dots 164\}} \Pr\{(l_1, l_2, l_3) \text{ at time } t\}\right).\end{aligned}$$

This probability gives the estimation of the corresponding linear combination for one frame j. We will increase the possibility to estimate the value of $s_1(l_1) + s_2(l_2) + s_3(l_3)$ correctly, when m frames (samples) $v^1(101 \dots 328), \dots, v^m(101 \dots 328)$ are given, as each of them provides some small contribution. By calculating the likelihood ratio

$$\Lambda_{l_1,l_2,l_3} = \sum_{j=1}^{m} \log_2 \left[\frac{p^j_{(l_1,l_2,l_3)}}{1 - p^j_{(l_1,l_2,l_3)}}\right]$$

we achieve a likelihood value (estimate) which is taken over all m frames. This can be turned into a binary estimate by

$$s_1(l_1) \oplus s_2(l_2) \oplus s_3(l_3) \stackrel{p}{=} \begin{cases} 0 \text{ if } \Lambda_{l_1,l_2,l_3} \geq 0 \\ 1 \text{ if } \Lambda_{l_1,l_2,l_3} < 0 \end{cases},$$

where $p > 0.5$ depends mainly on m. In [6] the authors finally examine different strategies for implementing the recovery of the key bits as efficient as possible.

4 Explaining the New Attack

In this section we describe our discovered improvements in general. Our main purpose is to reduce the number of frames m, which is needed for the attack.

4.1 Statistical Analysis of m Frames

We mentioned before that we have identified two general ideas for improving the previous results. The first is the fact that it is beneficial to study the derivative sequences instead of the sequences themselves. Assume that at time t the LFSRs are clocked l_1, l_2, and l_3 times, respectively. Then we also assume that at time $t+1$ the third LFSR is not clocked. In this case we have the equalities,

$$\begin{aligned} \hat{O}^j_{l_1,l_2,l_3}(t) &= s_1(l_1) \oplus s_2(l_2) \oplus s_3(l_3), \\ \hat{O}^j_{l_1+1,l_2+1,l_3}(t+1) &= s_1(l_1+1) \oplus s_2(l_2+1) \oplus s_3(l_3). \end{aligned} \tag{4}$$

Then the probability $\Pr\{\hat{O}^j_{l_1,l_2,l_3}(t) \oplus \hat{O}^j_{l_1+1,l_2+1,l_3}(t+1) = s_1(l_1) \oplus s_2(l_2) \oplus s_1(l_1+1) \oplus s_2(l_2+1)\} = \frac{1}{4} \cdot \Pr\{(l_1,l_2) \text{ at time } t\}$, where

$$\Pr\{(l_1,l_2) \text{ at time } t\} = \frac{\binom{t}{t-l_1}\binom{l_1}{t-l_2}}{2^{3t-(l_1+l_2)}}.$$

Note, that $\frac{1}{4} \cdot \Pr\{(l_1,l_2) \text{ at time } t\} > \Pr\{(l_1,l_2,l_3) \text{ at time } t\}$ so it gives us a larger bias when estimating the value of linear combinations of $s_i(l_i)$'s. Below is a comparison of these probabilities.

$(l_1,l_2,l_3),t$	$\Pr\{(l_1,l_2,l_3) \text{ at } t\} \cdot 10^4$	$\frac{1}{4}\Pr\{(l_1,l_2) \text{ at } t\} \cdot 10^4$
$(76,76,76),101$	9.7434	22.1207
$(79,79,79),105$	9.2012	21.2840
$(80,80,80),105$	6.6388	19.3778
$(79,80,81),106$	8.3858	20.8899
$(82,82,82),109$	8.7076	20.5083

The first idea to improve the attack is then to consider two consecutive expressions (4). Their sum only depends on two LFSRs, and the probability of the event is higher than before. We also note that we can similarly assume that LFSR-1 and LFSR-2 are not clocked at some time t. This gives us 3 *cases*. We define

$$\begin{aligned} {}_1\hat{z}^j_{l_2,l_3}(t) &= \hat{O}^j_{l_1,l_2,l_3}(t) \oplus \hat{O}^j_{l_1,l_2+1,l_3+1}(t+1) \stackrel{p}{=} s_2(l_2) \oplus s_3(l_3) \oplus s_2(l_2+1) \oplus s_3(l_3+1), \\ {}_2\hat{z}^j_{l_1,l_3}(t) &= \hat{O}^j_{l_1,l_2,l_3}(t) \oplus \hat{O}^j_{l_1+1,l_2,l_3+1}(t+1) \stackrel{p}{=} s_1(l_1) \oplus s_3(l_3) \oplus s_1(l_1+1) \oplus s_3(l_3+1), \\ {}_3\hat{z}^j_{l_1,l_2}(t) &= \hat{O}^j_{l_1,l_2,l_3}(t) \oplus \hat{O}^j_{l_1+1,l_2+1,l_3}(t+1) \stackrel{p}{=} s_1(l_1) \oplus s_2(l_2) \oplus s_1(l_1+1) \oplus s_2(l_2+1). \end{aligned} \tag{5}$$

The case when ${}_3\hat{z}^j_{l_1,l_2}(t)$ is equal to the value $s_1(l_1) \oplus s_2(l_2) \oplus s_1(l_1+1) \oplus s^j_2(l_2+1)$ can happen in two ways,

a) The first and the second LFSRs are indeed clocked l_1, l_2 times at time t occuring with probability $\Pr\{(l_1, l_2) \text{ at time } t\}$, **AND** at time $t+1$ the third LFSR *is not clocked*, with probability 1/4. The expression is always true in this case.
b) If the condition in a) is not fulfilled the expression will still be true with probability 1/2.

The second idea is to consider d consecutive estimators jointly as one d-dimension estimator. If we look at the sequence of d estimators of the form ${}_3\hat{z}^j_{l_1,l_2}(t), \ldots, {}_3\hat{z}^j_{l_1+d-1,l_2+d-1}(t+d-1)$, then we note that they are dependent on each other. To use this fact we suggest to consider not binary expressions, but vectors of d bits. Introduce a new d-bits vector, derived from the frame j,

$$ {}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t) = \begin{pmatrix} {}_3\hat{z}^j_{l_1,l_2}(t) \\ {}_3\hat{z}^j_{l_1+1,l_2+1}(t+1) \\ \vdots \\ {}_3\hat{z}^j_{l_1+d-1,l_2+d-1}(t+d-1) \end{pmatrix} \tag{6} $$

$$ = \begin{pmatrix} v^j(t) & \oplus\, v^j(t+1) \oplus & X_{1,l_1}(j) & \oplus & X_{2,l_2}(j) & \oplus\, X_{1,l_1+1}(j) \oplus X_{2,l_2+1}(j) \\ v^j(t+1) & \oplus\, v^j(t+2) \oplus & X_{1,l_1+1}(j) & \oplus & X_{2,l_2+1}(j) & \oplus\, X_{1,l_1+2}(j) \oplus X_{2,l_2+2}(j) \\ & & \vdots & & & \\ v^j(t+d-1) \oplus v^j(t+d) \oplus & X_{1,l_1+d-1}(j) & \oplus & X_{2,l_2+d-1}(j) & \oplus & X_{1,l_1+d}(j) \oplus X_{2,l_2+d}(j) \end{pmatrix}. $$

Define the d-dimension vector ${}_3\mathcal{S}_{l_1,l_2}$ (which is unknown for the attacker) as

$$ {}_3\mathcal{S}_{l_1,l_2} = \begin{pmatrix} s_1(l_1) & + & s_2(l_2) & + s_1(l_1+1) + s_2(l_2+1) \\ s_1(l_1+1) & + & s_2(l_2+1) & + s_1(l_1+2) + s_2(l_2+2) \\ & & \vdots & \\ s_1(l_1+d-1) + s_2(l_2+d-1) & + & s_1(l_1+d) + s_2(l_2+d) & \end{pmatrix}. \tag{7} $$

Then, from (5) it follows that

$$ {}_3\mathcal{S}_{l_1,l_2} \stackrel{p}{=} {}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t), \tag{8} $$

with some biased probability p. Note that the symbols are now of alphabet size 2^d.

Examining this in more detail, consider d consecutive irregular steps. The total number of possible scenarios is 4^d, since in each step one of four types of irregular clockings can be chosen, according to the bits C_1, C_2, C_3. If we assume that at time t the first and the second LFSRs are clocked exactly l_1, l_2 times, then we can classify the bits of the vector ${}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t)$. They can be either **C***orrect* (i.e., the next clocking is the required one so the bit has the same value as the corresponding bit in the vector ${}_3\mathcal{S}_{l_1,l_2}$), or **R***andom* (i.e., the bit can be 0 or 1, with probability 1/2). For each possible pattern $\{\mathbf{C}orrect, \mathbf{R}andom\}^d$ we calculate the corresponding number of scenarios out of 4^d possible, by exhaustively trying all the scenarios. For example, when $d = 4$, we have the following distribution:

Condition	${}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t)$				Probability	Event
	${}_3\hat{z}^j_{l_1,l_2}(t)$	${}_3\hat{z}^j_{l_1+1,l_2+1}(t+1)$	${}_3\hat{z}^j_{l_1+2,l_2+2}(t+2)$	${}_3\hat{z}^j_{l_1+3,l_2+3}(t+3)$		
Assumption is NOT correct	**R***andom*	**R***andom*	**R***andom*	**R***andom*	$1-P_0$	E_R
Assumption is correct	**C***orrect*	**C***orrect*	**C***orrect*	**C***orrect*	$P_0 \cdot 1/2^8$	E_0
	R*andom*	**C***orrect*	**C***orrect*	**C***orrect*	$P_0 \cdot 1/2^8$	E_1
	C*orrect*	**R***andom*	**C***orrect*	**C***orrect*	$P_0 \cdot 1/2^8$	E_2
	R*andom*	**R***andom*	**C***orrect*	**C***orrect*	$P_0 \cdot 1/2^8$	E_3
	C*orrect*	**C***orrect*	**R***andom*	**C***orrect*	$P_0 \cdot 1/2^8$	E_4
	R*andom*	**C***orrect*	**R***andom*	**C***orrect*	$P_0 \cdot 1/2^8$	E_5
	C*orrect*	**R***andom*	**R***andom*	**C***orrect*	$P_0 \cdot 1/2^8$	E_6
	R*andom*	**R***andom*	**R***andom*	**C***orrect*	$P_0 \cdot 1/2^8$	E_7
	C*orrect*	**C***orrect*	**C***orrect*	**R***andom*	$P_0 \cdot 3/2^8$	E_8
	R*andom*	**C***orrect*	**C***orrect*	**R***andom*	$P_0 \cdot 3/2^8$	E_9
	C*orrect*	**R***andom*	**C***orrect*	**R***andom*	$P_0 \cdot 3/2^8$	E_{10}
	R*andom*	**R***andom*	**C***orrect*	**R***andom*	$P_0 \cdot 3/2^8$	E_{11}
	C*orrect*	**C***orrect*	**R***andom*	**R***andom*	$P_0 \cdot 11/2^8$	E_{12}
	R*andom*	**C***orrect*	**R***andom*	**R***andom*	$P_0 \cdot 11/2^8$	E_{13}
	C*orrect*	**R***andom*	**R***andom*	**R***andom*	$P_0 \cdot 43/2^8$	E_{14}
	R*andom*	**R***andom*	**R***andom*	**R***andom*	$P_0 \cdot 171/2^8$	E_{15}

where $P_0 = \Pr\{(l_1, l_2) \text{ at time t}\}$ and the assumption is that the first two LFSRs have clocked (l_1, l_2) at time t.

Let us assume that we have received the vector ${}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t) = (0,1,1,0)^T$ at time t from the frame j. If we consider the hypothesis that ${}_3\mathcal{S}_{l_1,l_2} = (0,0,1,1)$, then the error pattern is $\mathcal{E}_d = {}_3\mathcal{S}_{l_1,l_2} \oplus {}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t) = (0,1,0,1)$. This *error pattern* $\mathcal{E}_d$ can be the result of one of the following events: $E_R, E_{10}, E_{11}, E_{14}, E_{15}$. Thus, the conditional probability $\Pr\{{}_3\mathcal{S}_{l_1,l_2} = (0,0,1,1) | {}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t) = (0,1,1,0)\} = \Pr\{\mathcal{E}_d = (0,1,0,1)\} = \frac{\Pr\{E_R\}}{2^4} + \frac{\Pr\{E_{10}\}}{2^2} + \frac{\Pr\{E_{11}\}}{2^3} + \frac{\Pr\{E_{14}\}}{2^3} + \frac{\Pr\{E_{15}\}}{2^4} = (1-P_0)/2^4 + P_0 \cdot 275/2^{12}$. Continuing in this way, the complete table for $\Pr\{\mathcal{E}_d\}$ can be derived. The distribution for $d = 4$ is given as in the table on the right.

$\mathcal{E}_d = {}_3\mathcal{S}_{l_1,l_2} \oplus {}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t)$	
$\mathcal{E}_d$	$\Pr\{\mathcal{E}_d\}$
(0, 0, 0, 0)	$(1-P_0)/2^4 + P_0 \cdot 431/2^{12}$
(1, 0, 0, 0)	$(1-P_0)/2^4 + P_0 \cdot 229/2^{12}$
(0, 1, 0, 0)	$(1-P_0)/2^4 + P_0 \cdot 293/2^{12}$
(1, 1, 0, 0)	$(1-P_0)/2^4 + P_0 \cdot 183/2^{12}$
(0, 0, 1, 0)	$(1-P_0)/2^4 + P_0 \cdot 341/2^{12}$
(1, 0, 1, 0)	$(1-P_0)/2^4 + P_0 \cdot 199/2^{12}$
(0, 1, 1, 0)	$(1-P_0)/2^4 + P_0 \cdot 263/2^{12}$
(1, 1, 1, 0)	$(1-P_0)/2^4 + P_0 \cdot 173/2^{12}$
(0, 0, 0, 1)	$(1-P_0)/2^4 + P_0 \cdot 377/2^{12}$
(1, 0, 0, 1)	$(1-P_0)/2^4 + P_0 \cdot 211/2^{12}$
(0, 1, 0, 1)	$(1-P_0)/2^4 + P_0 \cdot 275/2^{12}$
(1, 1, 0, 1)	$(1-P_0)/2^4 + P_0 \cdot 177/2^{12}$
(0, 0, 1, 1)	$(1-P_0)/2^4 + P_0 \cdot 323/2^{12}$
(1, 0, 1, 1)	$(1-P_0)/2^4 + P_0 \cdot 193/2^{12}$
(0, 1, 1, 1)	$(1-P_0)/2^4 + P_0 \cdot 257/2^{12}$
(1, 1, 1, 1)	$(1-P_0)/2^4 + P_0 \cdot 171/2^{12}$

For each frame j and for each vector $(b_0, \ldots, b_{d-1})^T$ we calculate

$$\begin{aligned}
&\Pr\{{}_3\mathcal{S}_{l_1,l_2} = (b_0, \ldots, b_{d-1})^T \text{ in } j\text{th frame}\} = {}_3p^j_{l_1,l_2}(b_0, \ldots, b_{d-1}) \\
&= \sum_{t \in \{101 \ldots 164\}} \Pr\{(l_1, l_2) \text{ at time } t\} \cdot \Pr\{\mathcal{E}_d = {}_3\hat{\mathcal{Z}}^j_{l_1,l_2}(t) \oplus (b_0, \ldots, b_{d-1})^T\} \\
&\qquad + \frac{1}{2}\left(1 - \sum_{t \in \{101 \ldots 164\}} \Pr\{(l_1, l_2) \text{ at time } t\}\right).
\end{aligned} \tag{9}$$

All the m frames give us a more precise estimation:

$$\begin{aligned}
\Pr\{{}_3\mathcal{S}_{l_1,l_2} = (b_0, \ldots, b_{d-1})^T\} &= {}_3p_{l_1,l_2}(b_0, \ldots, b_{d-1}) \\
&= \prod_{j=1}^{m} {}_3p^j_{l_1,l_2}(b_0, \ldots, b_{d-1}) = 2^{\sum_{j=1}^{m} \log_2 ({}_3p^j_{l_1,l_2}(b_0, \ldots, b_{d-1}))}.
\end{aligned} \tag{10}$$

In this formula the last two values should both be divided by a factor equal to their sum over all possible values of $(b_0, \ldots, b_{d-1})$. This factor has been left out because we are really interested in the relative values of the probabilities for different values of $(b_0, \ldots, b_{d-1})$. To simplify numerical calculations, ${}_3p_{l_1,l_2}(b_0, \ldots, b_{d-1})$ can be normalised through division by any constant.

We have just found the way how to calculate the probability $\Pr\{{}_3\mathcal{S}_{l_1,l_2} = (b_0, \ldots, b_{d-1})^T\}$, for every d-dimension value $(b_0, \ldots, b_{d-1})^T$. In a similar fashion, based on the equation (5), we can derive the d-dimension vectors ${}_1\hat{\mathcal{Z}}^j_{l_2,l_3}(t)$ and ${}_2\hat{\mathcal{Z}}^j_{l_1,l_3}(t)$, and then define the vectors ${}_1\mathcal{S}_{l_2,l_3}$ and ${}_2\mathcal{S}_{l_1,l_3}$. The formulas to calculate $\Pr\{{}_1\mathcal{S}_{l_2,l_3} = (b_0, \ldots, b_{d-1})^T\}$ and $\Pr\{{}_2\mathcal{S}_{l_1,l_3} = (b_0, \ldots, b_{d-1})^T\}$ are similar to equations (9) and (10).

Finally, we have a set of h tables like $\Pr\{{}_r\mathcal{S}_{l_i,l_j} = (b_0, \ldots, b_{d-1})\}$. If we "guess" the key $\hat{K}$, then in each such distribution table one row (probability) can be selected, corresponding to $\hat{K}$. *The measure of likelihood acceptance of $\hat{K}$ is the product of the selected probabilities through all the h tables.*

Our task is then to select a set of "guessed" keys $\hat{K}$ with maximum probabilities, and then perform a test whether the real key K can be one of the selected. More details depend on the exact structure of simulations, which we discuss in the next section.

4.2 Creating Candidate Tables of $s(l)$-Sequences

In the previous subsection we have found how to create a distribution table for d-dimension random variables ${}_r\mathcal{S}_{l_i,l_j}$. If we have h such distributions, then a "guessed" key $\hat{K}$ is measured by its probability, as described above. We are now faced with the problem of how to select the most likely $\hat{K}$'s in an efficient way. For this purpose we partly use the idea that was introduced in the Ekdahl-Johansson attack, but in a modified way. In this section we show the technical details of searching for the best $\hat{K}$'s, and focus on computation aspects.

The idea is that first we choose some interval $\mathcal{I}_1 = [I_{1,a} \dots I_{1,b}]$ and then we construct h_1 distribution tables for ${}_3\mathcal{S}_{l_1,l_2}$, where $l_1, l_2 \in \mathcal{I}_1$. I.e., the number of distribution tables will be $h_1 = (I_{1,b} - I_{1,a} + 1)^2$, and the number of $s_1(l)$'s and $s_2(l)$'s that are involved in the linear expressions for ${}_3\mathcal{S}_{l_1,l_2}$ is $2 \cdot (I_{1,b} - I_{1,a} + 1 + d)$, see formula (7).

Let us consider some choice of values for $s_1(I_{1,a}), \dots, s_1(I_{1,b}+d), s_2(I_{1,a}), \dots, s_2(I_{1,b}+d)$ to be a pair of vectors $(\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1})$ (note, the vector of interest ends with $I_{1,b} + d$, rather then $I_{1,b} + d - 1$; the reason can be seen from (7), where $l_1, l_2 \in \mathcal{I}_1$), i.e.,

$$(s_1(I_{1,a}), \dots, s_1(I_{1,b}+d), s_2(I_{1,a}), \dots, s_2(I_{1,b}+d)) \stackrel{p}{=} (\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1}). \tag{11}$$

The measure of the choice is the probability mass defined as

$$\prod_{l_1,l_2 \in \mathcal{I}_1} \Pr\{{}_3\mathcal{S}_{l_1,l_2} | (\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1})\}. \tag{12}$$

Now, by exhaustive search the most likely r pairs $(\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1})$ form a set ${}_3\mathfrak{T}_{\mathcal{I}_1} = \{(\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1})\}$. The size of the exhaustive search is $2^{2 \cdot (I_{1,b} - I_{1,a} + 1 + d)}$. In a similar way we can perform the same exhaustive search to create the sets ${}_1\mathfrak{T}_{\mathcal{I}_1} = \{(\mathcal{S}_{2,\mathcal{I}_1}, \mathcal{S}_{3,\mathcal{I}_1})\}$ and ${}_2\mathfrak{T}_{\mathcal{I}_1} = \{(\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{3,\mathcal{I}_1})\}$, each containing the r most likely candidates.

To understand better how the exhaustive search for ${}_3\mathfrak{T}_{\mathcal{I}_1}$ is done, one can think of the matrix multiplication:

$$\left(\begin{array}{c|c} \begin{array}{cccc} 1\,1\,0 & \dots & 0\,0 & \dots \\ 0\,1\,1 & & 0\,0 & \\ \vdots & \ddots & \vdots & \\ 0\,0\,0 & \dots & 1\,1 & \dots \end{array} & \begin{array}{cccc} 0\,0\,0 & \dots & 0\,0 & \dots \\ 0\,0\,0 & & 0\,0 & \\ \vdots & & \vdots & \\ 0\,0\,0 & \dots & 0\,0 & \dots \end{array} \\ \hline \begin{array}{cccc} 1\,1\,0 & \dots & 0\,0 & \dots \\ 0\,1\,1 & & 0\,0 & \\ \vdots & \ddots & \vdots & \\ 0\,0\,0 & \dots & 1\,1 & \dots \end{array} & \begin{array}{cccc} 0\,1\,1 & \dots & 0\,0 & \dots \\ 0\,0\,1 & & 0\,0 & \\ \vdots & \ddots & \vdots & \\ 0\,0\,0 & \dots & 0\,1 & \dots \end{array} \\ \hline \begin{array}{cccc} 0\,0\,0 & \dots & 1\,1\,0 & \dots \\ 0\,0\,0 & & 0\,1\,1 & \\ \vdots & & \vdots & \ddots \\ 0\,0\,0 & \dots & 0\,0 & \dots \end{array} & \begin{array}{cccc} 0\,0\,0 & \dots & 0\,0 & \dots \\ 0\,0\,0 & & 0\,0 & \\ \vdots & & \vdots & \\ 0\,0\,0 & \dots & 1\,1 & \dots \end{array} \end{array}\right) \cdot \begin{pmatrix} s_1(I_a) \\ \vdots \\ s_1(I_b+d) \\ s_2(I_a) \\ \vdots \\ s_2(I_b+d) \end{pmatrix} = \left(\begin{array}{c} {}_3\mathcal{Z}_{I_a,I_a} \\ \hline {}_3\mathcal{Z}_{I_a,I_a+1} \\ \hline \vdots \\ \hline {}_3\mathcal{Z}_{I_b,I_b} \end{array}\right),$$

where for every "guessed" vector $(\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1})$ (exhaustive search) the set of vectors ${}_3\mathcal{S}_{l_1,l_2} \stackrel{p}{=} {}_3\mathcal{Z}_{l_1 b,l_2}$ is determined uniquely by the matrix multiplication. We can then calculate the value of our choice by formula (12). After that, the most likely r pairs are selected and stored in the list (or table) ${}_3\mathfrak{T}_{\mathcal{I}_1}$.

Recall that to recover the key K uniquely, we need to have 64 bits: 19 bits of $s_1(l)$'s, 22 bits of $s_2(l)$'s, and 23 bits of $s_3(l)$'s. It means that for $d = 4$ it might be enough to have only one interval $\mathcal{I}_1$ of size 19. When we try to reduce the number of frames m needed for the attack, then there are two reasons for why this simple scenario is not working:

a) to create one likelihood table ${}_3\mathbf{T}_{\mathcal{I}_1}$ the exhaustive search will be of size $2^{2\cdot(19+4)} = 2^{46}$ – this is practically impossible;
b) when the number of frames m is reduced, then the number of candidates r must be increased significantly, so that the correct pairs are present in the tables ${}_1\mathbf{T}_{\mathcal{I}_1}, {}_2\mathbf{T}_{\mathcal{I}_1}$, and ${}_3\mathbf{T}_{\mathcal{I}_1}$. Otherwise, the joint intersection of these sets will not give us the correct triple $(\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1}, \mathcal{S}_{3,\mathcal{I}_1})$.

To overcome these problems, we could take $\mathcal{I}_1$ of a short size, and introduce one more interval, $\mathcal{I}_2 = [I_{2,a} \dots I_{2,b}]$, and then we construct two kinds of tables ${}_*\mathbf{T}_{\mathcal{I}_1}$ and ${}_*\mathbf{T}_{\mathcal{I}_2}$ each of size r. We need to take $\mathcal{I}_2$ such that it intersects $\mathcal{I}_1$, otherwise the intersection would be r^2, and, hence, r cannot be large. Now in a similar way we can create the sets ${}_1\mathbf{T}_{\mathcal{I}_2} = \{(\mathcal{S}_{2,\mathcal{I}_2}, \mathcal{S}_{3,\mathcal{I}_2})\}$, ${}_2\mathbf{T}_{\mathcal{I}_2} = \{(\mathcal{S}_{1,\mathcal{I}_2}, \mathcal{S}_{3,\mathcal{I}_2})\}$, and ${}_3\mathbf{T}_{\mathcal{I}_2} = \{(\mathcal{S}_{1,\mathcal{I}_2}, \mathcal{S}_{2,\mathcal{I}_2})\}$, each containing the r most likely pairs, the measure of which is calculated similar to the formula (12). Due to the intersection

$$\mathcal{S}_{i,\mathcal{I}_1} \times \mathcal{S}_{i,\mathcal{I}_2} = \begin{cases} \mathcal{S}_{i,\mathcal{I}_1 \cup \mathcal{I}_2}, & \text{if the end of } \mathcal{S}_{i,\mathcal{I}_1} \text{ corresponds to the beginning of } \mathcal{S}_{i,\mathcal{I}_2} \\ \emptyset, & \text{otherwise} \end{cases}$$

the intersection of these two sets is

$${}_3\mathbf{T}_{\mathcal{I}_1 \cup \mathcal{I}_2} = {}_3\mathbf{T}_{\mathcal{I}_1} \cap {}_3\mathbf{T}_{\mathcal{I}_2} = \left\{ (\mathcal{S}_{1,\mathcal{I}_1 \cup \mathcal{I}_2}, \mathcal{S}_{2,\mathcal{I}_1 \cup \mathcal{I}_2}) : \begin{cases} (\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1}) \in {}_3\mathbf{T}_{\mathcal{I}_1} \\ (\mathcal{S}_{1,\mathcal{I}_2}, \mathcal{S}_{2,\mathcal{I}_2}) \in {}_3\mathbf{T}_{\mathcal{I}_2} \\ \mathcal{S}_{1,\mathcal{I}_1} \times \mathcal{S}_{1,\mathcal{I}_2} \neq \emptyset \\ \mathcal{S}_{2,\mathcal{I}_1} \times \mathcal{S}_{2,\mathcal{I}_2} \neq \emptyset \end{cases} \right\}.$$

The larger the intersection of the intervals $\mathcal{I}_1$ and $\mathcal{I}_2$, the smaller the intersection set, i.e. $|{}_3\mathbf{T}_{\mathcal{I}_1 \cup \mathcal{I}_2}| \ll |{}_3\mathbf{T}_{\mathcal{I}_1}| \cdot |{}_3\mathbf{T}_{\mathcal{I}_2}| = r^2$. Let us call this type of intersections as *horizontal* intersection. Similar horizontal intersections are ${}_1\mathbf{T}_{\mathcal{I}_1 \cup \mathcal{I}_2}$ and ${}_2\mathbf{T}_{\mathcal{I}_1 \cup \mathcal{I}_2}$.

By *vertical* intersection we call the intersections of the form:

$${}_{1,2}\mathbf{T}_{\mathcal{I}_i} = {}_1\mathbf{T}_{\mathcal{I}_i} \cap {}_2\mathbf{T}_{\mathcal{I}_i} = \left\{ (\mathcal{S}_{1,\mathcal{I}_i}, \mathcal{S}_{2,\mathcal{I}_i}, \mathcal{S}_{3,\mathcal{I}_i}) : \begin{cases} (\mathcal{S}_{2,\mathcal{I}_i}, \mathcal{S}_{3,\mathcal{I}_i}) \in {}_1\mathbf{T}_{\mathcal{I}_i} \\ (\mathcal{S}_{1,\mathcal{I}_i}, \mathcal{S}_{3,\mathcal{I}_i}) \in {}_2\mathbf{T}_{\mathcal{I}_i} \end{cases} \right\},$$

and ${}_{2,3}\mathbf{T}_{\mathcal{I}_i}$, ${}_{1,3}\mathbf{T}_{\mathcal{I}_i}$ are defined in a similar way. One more *triple vertical* intersection is defined as

$${}_{1,2,3}\mathbf{T}_{\mathcal{I}_i} = {}_1\mathbf{T}_{\mathcal{I}_i} \cap {}_2\mathbf{T}_{\mathcal{I}_i} \cap {}_3\mathbf{T}_{\mathcal{I}_i} = \left\{ (\mathcal{S}_{1,\mathcal{I}_i}, \mathcal{S}_{2,\mathcal{I}_i}, \mathcal{S}_{3,\mathcal{I}_i}) : \begin{cases} (\mathcal{S}_{2,\mathcal{I}_i}, \mathcal{S}_{3,\mathcal{I}_i}) \in {}_1\mathbf{T}_{\mathcal{I}_i} \\ (\mathcal{S}_{1,\mathcal{I}_i}, \mathcal{S}_{3,\mathcal{I}_i}) \in {}_2\mathbf{T}_{\mathcal{I}_i} \\ (\mathcal{S}_{1,\mathcal{I}_i}, \mathcal{S}_{2,\mathcal{I}_i}) \in {}_3\mathbf{T}_{\mathcal{I}_i} \end{cases} \right\}.$$

4.3 Design of Intervals

Let us take one interval $\mathcal{I}_1' = [87 \dots 97]$. Two extreme situations are when $(l_1, l_2) = (87, 87)$ and $(l_1, l_2) = (97, 97)$. In each frame j there are only 228 bits are accessible $v^j(101), \dots, v^j(100 + 228)$. In Figure 2 we see that the probability $\Pr\{(l_1, l_2) \text{ at time } t\}$ for this interval gets its maximum value on around

$t \approx (116 \dots 129)$. Hence, the bits around $v(116) \dots v(129)$ give us the most information about the d-dimension vectors, when $l_1, l_2 \in \mathcal{I}_1$. We can also say that for this interval the informative bits are around $v(105) \dots v(145)$, because for any other v's the probability is almost 0.

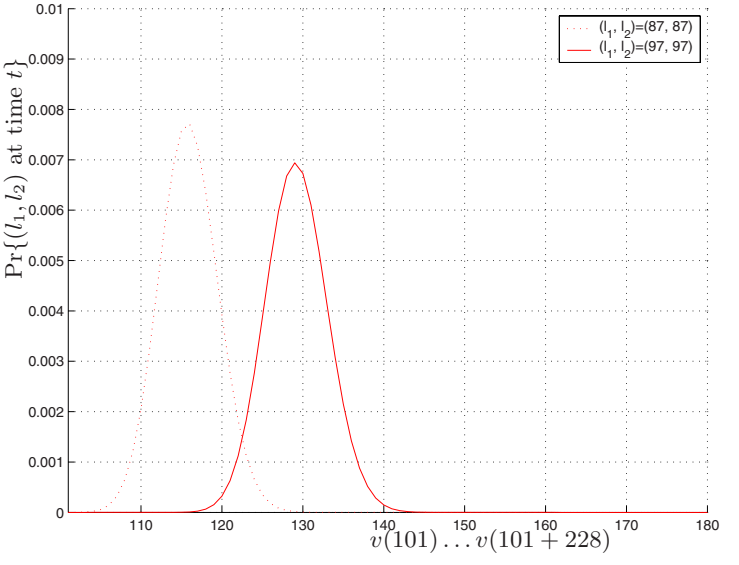

Fig. 2. The density of $\Pr\{(l_1, l_2) \text{ at time } t\}$ when $(l_1, l_2) = (87, 87)$ and $(l_1, l_2) = (97, 97)$

Let us now consider three more intervals $\mathcal{I}'_2 = [63 \dots 73]$, $\mathcal{I}'_3 = [165 \dots 175]$, and $\mathcal{I}'_4 = [231 \dots 241]$. In Figure 3 the bounded densities for each interval are shown. The interval $\mathcal{I}'_2$ is moved to the left below $t < 101$, where the valuable v's are unaccessible for us. It means that this choice is not appropriate. On the other hand, the interval $\mathcal{I}'_4$ is moved to the right and very close to the right border of accessible v's. This interval can be considered as the last appropriate interval. Also note that as the interval is moved to the righ the amplitute decreases, i.e. the error probability of the random variables estimation is higher.

In our simulations we decided to choose the size of each interval to be 11. Independently of the parameter $d \geq 1$ in each table ${}_3\mathfrak{T}_{\mathcal{I}_i}$ we store only the pairs $(\mathcal{S}_{1,\mathcal{I}_i}, \mathcal{S}_{2,\mathcal{I}_i})$ of vectors each of size 12 bits only. The schematical structure of intervals is depicted below in Figure 4.

Two neighbour intervals intersect in 6 positions, whereas the last $d-1$ positions are assumed to be badly estimated (tail bits). I.e., any horizontal intersection of two tables ${}_3\mathfrak{T}_{\mathcal{I}_i}$ and ${}_3\mathfrak{T}_{\mathcal{I}_{i+1}}$ will be done by 12 bits (6 bits are $s_1(\mathcal{I}_{i+1})$, $\dots$, $s_1(\mathcal{I}_{i+1}+5)$, and similar 6 bits are $s_2(\mathcal{I}_{i+1})$, $\dots$, $s_2(\mathcal{I}_{i+1}+5)$). Also note that any vertical intersection will be done in 12 bits also. The choice of this structure of the intervals allowed us to introduce several efficient strategies to intersect the tables.

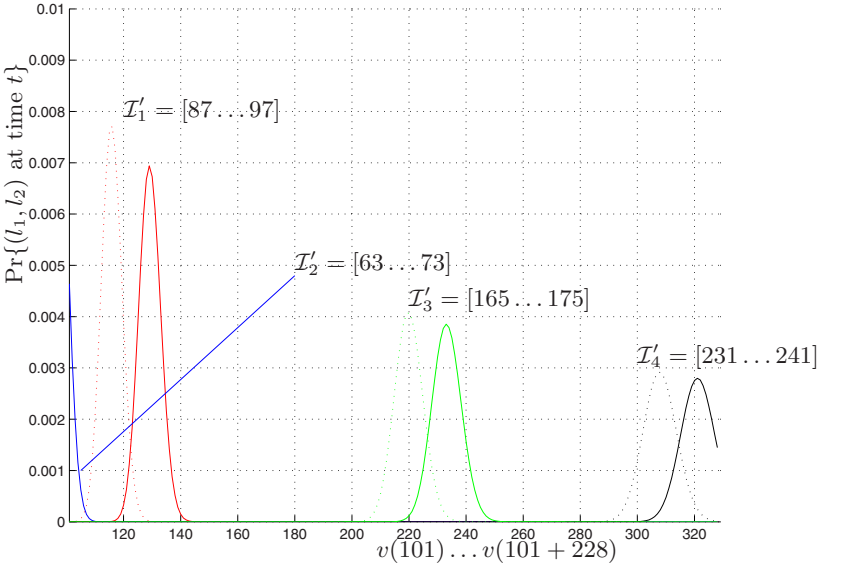

Fig. 3. The bounded densities for $\mathcal{I}'_1 = [87 \dots 97]$, $\mathcal{I}'_2 = [63 \dots 73]$, $\mathcal{I}'_3 = [165 \dots 175]$, and $\mathcal{I}'_4 = [231 \dots 241]$

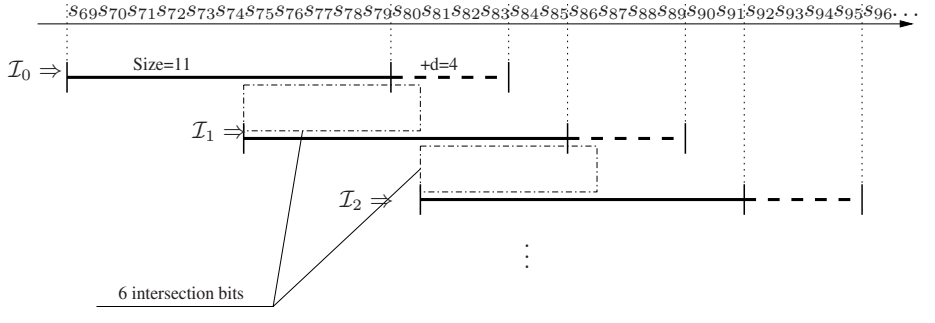

Fig. 4. The structure of intervals used in simulations

Since the size of each interval is 11, it means that the number of distribution tables of ${}_*\mathcal{S}_{l_i,l_j}$-random variables is $11^2 = 121$. When $d = 4$, the number of variables involved in ${}_*\mathcal{S}_{l_i,l_j}$'s is $2 \cdot (11 + 4) = 30$. Hence, to create one ${}_*\mathfrak{T}_{\mathcal{I}_i}$-set of the r most likelihood pairs, we need to perform an exhaustive search of size 2^{30}. The number of such sets ${}_*\mathfrak{T}_{\mathcal{I}_i}$ is 9 (3 intervals times 3 cases for '*').

In our simulations we have considered 28 intervals:

$$\begin{cases} \mathcal{I}_0 = [69 \dots 79] \\ \mathcal{I}_k = 6 \cdot k + \mathcal{I}_0 \quad \text{for } k = 1, 2, \dots, 27. \end{cases} \tag{13}$$

So, the last interval is $\mathcal{I}_{27} = [231 \dots 241]$ (see also Figure 3). When for a chosen interval $\mathcal{I}_i$ we estimate the probability $\Pr\{{}_3\mathcal{S}_{l_1,l_2} = (b_0, \dots, b_{d-1})^T\}$ with the

formula (9), then we only need to look through the bits v^j that are valuable for $\mathcal{I}_i$. Let us set the "window" of valuable bits to be of size 64, then, for example, for the interval $\mathcal{I}_1$ on the Figure 3 the "window" is $t_1 = [101 \dots 164]$, for $\mathcal{I}_3 \Rightarrow t_3 = [203 \dots 266]$, and for $\mathcal{I}_4 \Rightarrow t_4 = [266 \dots 329]$. Actually, the "window" can be less, but 64 bits completely cover the most valueable v's for any interval $\mathcal{I}_i$.

The likelihood sets ${}_*\mathfrak{T}_{\mathcal{I}_i}$, each containing r pairs, can be presented in the following table:

	$\mathcal{I}_0$	$\mathcal{I}_1$	...	$\mathcal{I}_{27}$
Case 1	${}_1\mathfrak{T}_{\mathcal{I}_0} = (\mathcal{S}_{2,\mathcal{I}_0}, \mathcal{S}_{3,\mathcal{I}_0})$	${}_1\mathfrak{T}_{\mathcal{I}_1} = (\mathcal{S}_{2,\mathcal{I}_1}, \mathcal{S}_{3,\mathcal{I}_1})$		${}_1\mathfrak{T}_{\mathcal{I}_{27}} = (\mathcal{S}_{2,\mathcal{I}_{27}}, \mathcal{S}_{3,\mathcal{I}_{27}})$
Case 2	${}_2\mathfrak{T}_{\mathcal{I}_0} = (\mathcal{S}_{1,\mathcal{I}_0}, \mathcal{S}_{2,\mathcal{I}_0})$	${}_2\mathfrak{T}_{\mathcal{I}_1} = (\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1})$		${}_2\mathfrak{T}_{\mathcal{I}_{27}} = (\mathcal{S}_{1,\mathcal{I}_{27}}, \mathcal{S}_{2,\mathcal{I}_{27}})$
Case 3	${}_3\mathfrak{T}_{\mathcal{I}_0} = (\mathcal{S}_{1,\mathcal{I}_0}, \mathcal{S}_{2,\mathcal{I}_0})$	${}_3\mathfrak{T}_{\mathcal{I}_1} = (\mathcal{S}_{1,\mathcal{I}_1}, \mathcal{S}_{2,\mathcal{I}_1})$		${}_3\mathfrak{T}_{\mathcal{I}_{27}} = (\mathcal{S}_{1,\mathcal{I}_{27}}, \mathcal{S}_{2,\mathcal{I}_{27}})$

The time complexity to form these data is $O(3 \cdot 28 \cdot (11^2 \cdot 2^d \cdot m \cdot 64 + 2^{22+2d}))$. This is because there are 84 sets ${}_*\mathfrak{T}_i$; to create each set requires 11^2 distribution tables of size 2^d; to calculate each value in the table requires $m \cdot 64$ operations; and the exhaustive search complexity for each set is 2^{22+2d}.

4.4 Strategies for Intersection of the Tables ${}_*\mathfrak{T}_{\mathcal{I}_i}$

When the first part of the attack is done, the second part is just intersection of the sets until we get the set of triples ${}_{1,2,3}\mathfrak{T}_*$ of appropriate size. Here are several strategies that we can follow to achieve our goal:

I. *Intersection of 9 tables, large r.* Try all triples of intervals $(\mathcal{I}_k, \mathcal{I}_{k+1}, \mathcal{I}_{k+2})$, for $k = 0, 1, \dots, 25$. The intersection of 9 tables gives us the table ${}_{1,2,3}\mathfrak{T}_{\mathcal{I}_k \cup \mathcal{I}_{k+1} \cup \mathcal{I}_{k+2}}$ of triples $(\mathcal{S}_{1,\mathcal{I}_k \cup \mathcal{I}_{k+1} \cup \mathcal{I}_{k+2}}, \mathcal{S}_{2,\mathcal{I}_k \cup \mathcal{I}_{k+1} \cup \mathcal{I}_{k+2}}, \mathcal{S}_{3,\mathcal{I}_k \cup \mathcal{I}_{k+1} \cup \mathcal{I}_{k+2}})$. Each $\mathcal{S}$ contains 24 bits, but we need only 19, 22, and 23 bits for LFSR-1, LFSR-2, and LFSR-3, respectively. We can do first vertical intersections and get ${}_{1,2,3}\mathfrak{T}_{\mathcal{I}_i}$, and then perform horizontal intersection. Since any of the intersections is done by 12 bits, the number of the most likely pairs in ${}_*\mathfrak{T}_{\mathcal{I}_i}$ can be quite large. For this strategy we can safely use $r \approx 50000$;

II. *Intersection of 6 tables, medium r.* The same as Strategy I, but for each interval one table is discarded. We just assume that the discarded tables do not contain the correct pairs. Then perform the intersection of the remaining 6 tables. The number of assumptions is 3^3. The parameter r is about $r \approx 30000$.

III. *Intersection of 4 tables, small r.* Try all pairs of intervals $(\mathcal{I}_k, \mathcal{I}_{k+2})$, for all $k = 0, 1, \dots, 25$. We assume also that one of the tables ${}_*\mathfrak{T}_{\mathcal{I}_k}$ and one of ${}_*\mathfrak{T}_{\mathcal{I}_{k+2}}$ do not contain the correct pair. The number of assumptions is 3^2. For the remaining 4 tables we perform the intersection. Note, there is no horizontal intersection, but only 2 vertical intersections, one for $\mathcal{I}_k$ and one for $\mathcal{I}_{k+2}$. Due to this the critical value for the parameter r is about

$r \approx 10000$. The appropriate choice of the intersection scheme made this strategy work.

IV. *Intersection of 4 tables, small r, version 2.* The same as Strategy III, but the pairs of intervals $(\mathcal{I}_{k_1}, \mathcal{I}_{k_2})$ can be so that $k_1 = 0, 1, \ldots, 25$, and $k_2 = k_1 + 2, \ldots, 28$. Unfortunately, it can happen that *some* outputs from LFSR's in the second interval $\mathcal{I}_{k_2}$ will be a linear combination of $s(l)$'s from $\mathcal{I}_{k_1}$. For LFSR-1, the size of which is 19, it is not very critical because we achieve 24 bits of information. It means that even if 5 bits will depend on others, we still have a full rank in translation from $s(l)$'s to 19 bits of the key K. It is more critical for LFSR-3, which is of length 23. Anyway, if the system will not be of full rank, then some bits we can just guess. That makes this strategy work in general (implementation is then more complicated).

V. *Heuristic procedure, r is dynamic.* Can be introduced in the following way: If in some step for some intersection $\mathfrak{T}' \cap \mathfrak{T}''$ we get $\emptyset$, or a very small set, then increase the value r for $\mathfrak{T}'$ and $\mathfrak{T}''$ selectively, until their intersection give us a set of size at least r_0, for some threshold value. Thus, we can start creating the sets ${}_*\mathfrak{T}_{\mathcal{I}_i}$ with a small value of r, and then increase it selectively, when necessary.

So, here is a wide choice to choose a strategy. In our simulations we have tried several of them.

5 Simulation Results

The attack can basically be divided into three steps,

1) Statistical analysis of m frames,
2) Decoding process and generating the tables ${}_*\mathfrak{T}_{\mathcal{I}_i}$,
3) Intersection of the tables and check estimated keys $\hat{K}$.

For the first two steps we present the actual time. The attack was implemented on Pentium-4, CPU 2.4GHz, 256Mb RAM, OS Windows XP Pro SP1.

1st step/ 2nd step	m=2000	m=5000	m=10000
d=1	11 sec/ 18 sec	26 sec / 18 sec	58 sec / 18 sec
d=2	14 sec/ 8 min	32 sec / 8 min	72 sec / 8 min
d=4	40 sec/ 7 hrs	94 sec / 7 hrs	190 sec / 7 hrs

The measure of "goodness" of the attack can be expressed in terms of the number of frames m needed and its success rate. The attack was successfully implemented on a usual PC-computer, and it performs the attack from several seconds to several minutes, depending on the choice of strategy, and parameters m, d, and r.

Success rate of the attack depends on the choice of the design parameters d and r, and the strategy that is used. For some values of m and d here we present in Figure 5 the plots for the probabilities:

$$\Pr\{ \text{ the correct vector } \textit{is in } {}_*\mathfrak{T}_{\mathcal{I}_i}, \text{ for given parameter } r\}.$$

When the tables are constructed, in the intersection process it is very important that the correct pair is present in the corresponding table. Otherwise, the intersection will never give us a correct key.

In Figure 5 we show the real estimated success rates for different strategies, with different number of frames m and the attack design parameter d. In Figure 5 (upper left) consider the curve corresponding to $d = 1$ and to Strategy I, when $m = 10000$ frames. For $r = 15000$ we have the success rate of the attack around 58%, whereas for Strategies II-IV the success rate is almost 100%.

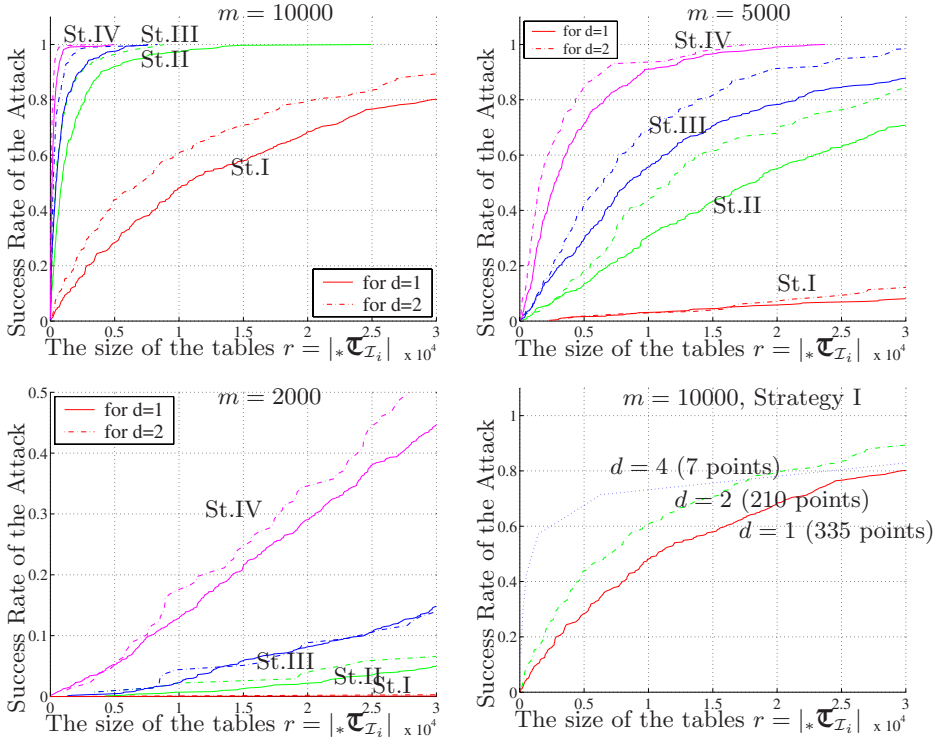

Fig. 5. Strategies comparison for $m = 10000$ (top left), $m = 5000$ (top right), and $m = 2000$ (down left). The effect of d on the success rate on the example when $m = 10000$ and Strategy I is applied (down right)

From the plots below the Strategy IV looks the most attractive. In this strategy we need to intersect only 4 tables, but the disadvantage is that there is no horizontal intersection. And then after two vertical intersections we need to try all possible combinations of elements in two tables. One more disadvantage is that we could get some equation dependencies between two intervals, so then the actual time complexity will grow. On the contrary, strategy III looks the next the most attractive, and there are no problems with intervals. Since there are no horizontal intersections in these strategies, this forces us to reduce the

parameter r significantly. The critical value of this parameter is $r_{cr} = 10000$, and the optimal is $r_{opt} = 2000$ from the computational and memory points of view. Strategy II avoids such problems mostly because of the presence of vertical intersections, which are intersecting on 12 bits.

A practical solution to overcome the time-memory problems related to intersections of the tables can be the use of the Heuristic Strategy V, combined with one of the previous strategies. The idea of Heuristic is to control the size of the intersection. If the size is likely to be increased by some threshold criteria, then try to increase the initial parameter r until the limit is reached, or solution is found. Heuristic can also control the size of the tables independently, and this will give the best performance of the attack.

Dramatic advantage of use the proper design parameter d is seen in the same Figure 5. To make the advantage clearer, the bottom right subplot shows how much we gain when d is 1, 2, and 4. When $r = 15000$, the change of the parameter d from $d = 1$ to $d = 2$ significantly increases the success rate from 58% to 70%. These simulations were done for $m = 10000$ frames, and with the application of Strategy I.

Finally, we show the advantage of our attack in comparison with the previous Ekdahl-Johansson attack in the following two tables:

Success Rate/ (Time of the Attack)	**Ekdahl-Johansson Attack (2002)** **Number of Frames/(time of GSM conversation in min/sec)**		
Configuration	**30000** (2m30s)	**50000** (3m45s)	**70000** (5m20s)
3 Intervals of size 7	0.02/(1min)	0.13/(2min)	0.49/(3min)
3 Intervals of size 8	0.02/(2min)	0.20/(3min)	0.57/(4min)
2 Intervals of size 9	0.03/(3min)	0.33/(4min)	0.76/(5min)

Success Rate/ (Time of the Attack)	**Our Proposed Attack** **Number of Frames/(time of GSM conversation in min/sec)**		
Configuration	**2000** (9sec)	**5000** (43sec)	**10000** (46sec)
St.I, d=2, r=10K	0.01/(8min)	0.05/(8min)	0.60/(8min)
St.II, d=1, r=5K	0.01/(29sec)	0.15/(44sec)	0.93/(76sec)
St.III, d=2, r=5K	0.02/(8min)	0.40/(8min)	0.99/(8min)
St.IV, d=2, r=5K	0.05/(10min)	0.85/(10min)	0.9999(10min)

6 Conclusions

We have demonstrated how two new ideas provide improved performance for a correlation attack against A5/1. In simulation we get a high success rate for only 2000-5000 frames, using very little computation. But there is still deviation in performance depending on the strategies we choose, which means that there may very well be further improvements to come if we can find the best attack strategies. Another interesting topic is to examine how small m can be made if

we allow a substantial increase in attack complexity. If m can be decreased a bit further, ciphertext only attack may be practically possible, as discussed briefly in the introduction of the paper.

Acknowledgments

We thank Eli Biham for his useful comments on the paper.

References

1. M. Briceno, I. Goldberg, and D. Wagner. A pedagogical implementation of A5/1. Available at http://jya.com/a51-pi.htm, Accessed August 18, 2003, 1999.
2. J.D. Golić. Cryptanalysis of alleged A5 stream cipher. In W. Fumy, editor, *Advances in Cryptology—EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 239–255. Springer-Verlag, 1997.
3. A. Biryukov, A. Shamir, and D. Wagner. Real time cryptanalysis of A5/1 on a PC. In B. Schneier, editor, *Fast Software Encryption 2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, 2000.
4. E. Biham and O. Dunkelman. Cryptanalysis of the A5/1 GSM stream cipher. In B. E. Roy and E. Okamoto, editors, *Progress in Cryptology—INDOCRYPT 2000*, volume 1977 of *Lecture Notes in Computer Science*, pages 43–51. Springer-Verlag, 2000.
5. M. Krause. BDD-based cryptanalysis of keystream generators. In L.R. Knudsen, editor, *Advances in Cryptology—EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 222–237. Springer-Verlag, 2002.
6. P. Ekdahl and T. Johansson. Another attack on A5/1. In *Proceedings of International Symposium on Information Theory*, page 160. IEEE, 2001.
7. E. Barkan, E. Biham, and N. Keller. Instant ciphertext only cryptanalysis of GSM encrypted communication. In D. Boneh, editor, *Advances in Cryptology—CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 600–616. Springer-Verlag, 2003.
8. ETSI EN 300 963 v8.0.1 (2000-11) Standard. Digital cellular telecommunications system (Phase 2+) (GSM); Full rate speech; Comfort noise aspect for full rate speech traffic channels (GSM 06.12 version 8.0.1 Release 1999), 2000.

[0] The work described in this paper has been supported in part by Grant VR 621-2001-2149, in part by the Graduate School in Personal Computing and Communication PCC++, and in part by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT.