CRYPTANALYSIS OF STREAM CIPHER—A SURVEY

Shaoquan Jiang and Guang Gong
Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario N2L 3G1, Canada
Email: {jiangshq, ggong}@calliope.uwaterloo.ca

Abstract. In this report, a survey on cryptanalysis of stream cipher is presented. For each significant method, a brief outline is provided and their advantages and drawbacks together with comparisons are presented as well.

1. Introduction

Stream cipher are commonly used in secure communications, especially in wireless communication. When producing a cipher, the plaintext string is bit-wise added to a pseudo-random sequence called *key stream.* It is known if this pseudo-random sequence is truly random, then the cipher is unbreakable, where truly random binary string is defined as i.i.d binary sequence with symbol 0 and 1 equally likely. However, a truly random sequence can not be generated [48]. A loose requirement is to construct a key stream generator, which is close to purely random in some sense. This closeness is in fact the strength of the stream cipher. It deserves mention that in different applications, the security requirements may be different. Accordingly, the cryptanalysis of a stream cipher can be classified as

1. Ciphertext Only Attack. This attack considers the possibility to break the stream cipher with cipher text only.
2. Known Plaintext Attack. In this attack, a piece of ciphertext together with its corresponding plaintext is known. For stream cipher, this is equivalent to say, a piece of key stream is known. The objective is to recover the private key that generates the whole key stream.
3. Chosen Plaintext Attack. A piece of arbitrary chosen plaintext and its corresponding ciphertext is known. In stream cipher, the key stream is independent of plaintext. Therefore, this attack is equivalent to the known palaintext attack. However, keep in mind that it is quite different for block cipher.
4. Distinguishing from Truly Random Sequence. The objective of this attack is to see whether it is really like a truly random sequence. This is not an attack in fact. It is just a strict

criterion for a good stream cipher. If we define a truly random sequence as an i.i.d source with each possible symbol equally likely, we can use the notion of distinguishing from truly random sequence below to define the success of this attack.

Definition 1.1. *Let G be a key stream generator. Given that a binary string C of length n is generated from model G or truly random model T equaly likely, a probabilistic algorithm $\mathcal{A}$ wants to decide C comes from G model or random model. It outputs 1 if it decides C is generated by G, outputs 0 otherwise. We say $\mathcal{A}$ can distinguish G from truly random sequence if there exists a constant $\epsilon > 0$ such that when $n > N_0$ for some positive number N_0,*

$$|\frac{1}{2}(P(\mathcal{A}(C) = 1|C \leftarrow G) + P(\mathcal{A}(C) = 0|C \leftarrow T) - \frac{1}{2})| \geq \epsilon, \quad (1)$$

where $C \leftarrow G$ means C is generated from G but $\mathcal{A}$ does not know this information and $C \leftarrow T$ means C is drawn according to purely random model T but $\mathcal{A}$ does not know this information. We say, $\mathcal{A}$ distinguish G from truly random sequence in worst (average) running time $T(n)$ if mathcalA makes decision in worst (average) time $T(n)$.

In known literature, almost all the research papers discuss only the known plaintext attack. Indeed, it is possible that people can known some plaintext by personal ways. For example, in cell phone communication it is not difficult to obtain a piece of cipher and its corresponding plaintext. Thus, this type of attack is very practical.

Distinguishing from truly random sequence attack is a newly proposed security criterion. Don Coppersmith [17] showed a distinguishing technique to a class of stream ciphers.

Ciphertext only attack is much weaker than known plaintext attack. In some situations, this attack is still practical.

In the following sections, we will consider kinds of cryptanalysis. And their advantages and drawbacks are presented. Related analysis are compared.

2. Ciphertext Only Attack

In some situations, the attacker can only get part of ciphertext. To decrypt all the massages, he has to find the secret key. That is, they have to some ciphertext only attacks to recover the secret key. Now we introduce the attack on multiple LSFRs [37] based combination generator. It is depicted as follows. Let $\{x_t^i\}_{t=0}^{\infty}, i = 1, 2, \cdots, m$ be m LFSR sequence and $f(y_1, \cdots, y_m)$ be a Boolean function. Then the key stream $\{z_t\}$ is output of this Boolean function where the inputs are the m LFSR sequences. And the cipher $\{c_t\}$ is produced by bitwise adding the plaintext $\{y_t\}$ to the key stream $\{z_t\}$. Now, suppose we are given

(1) Feedback polynomial for the m LSFRs.

(2) a segment of ciphertext C_1^N

(3) $P(y_t = 0) = p_0 > 1/2$, where p_0 is known.

(4) $P(z_t \oplus x_t^i) = q_i > 1/2, q_i, i = 1, \cdots, m$ is known.

The objective is to recover the initial states of all LFSRs. Siegenthaler [64] based on the initial idea of correlation concept [12], proposed a divide-and-conquer attack to the generator above. The key idea is to build correlation between LFSRi and ciphertext using the bias of plaintext and correlation probability between key stream and individual input LFSRi. We model both LFSRi and plaintext as i.i.d binary source. And 0 and 1 in LFSRi are equally likely. Let $\hat{X}^{(i)} = \{\hat{x}_n^{(i)}\}$ be the sequence obtained by guessing the initial state of LFSRi. Then

$$\begin{aligned} P(c_n = \hat{x}_n^{(i)}) &= P(y_n \oplus z_n = \hat{x}_n^{(i)}) \\ &= p_0 P(z_n = \hat{x}_n^{(i)}) + (1 - p_0)(1 - P(z_n = \hat{x}_n^{(i)})) \end{aligned}$$

Thus, if $\hat{X}^{(i)} = X^{(i)}$, define $p_i = P(c_n = \hat{x}_n^{(i)}) = p_0 q_i + (1 - p_0)(1 - q_i)$. Then $p_i - 1/2 = 2(p_0 - 1/2)(q_i - 1/2) > 0$. If $\hat{X}^{(i)} \neq X^{(i)}$, then $P(c_n = \hat{x}_n^{(i)}) = 1/2$. Therefore, if $\hat{X}^{(i)} = X^{(i)}$, then $\hat{X}^{(i)} \oplus C$ is i.i.d source with distribution $p_i > 1/2$. That is to say, correlation probabiltiy between the cipher string and input LFSRi is larger than one half. If $\hat{X}^{(i)} \neq X^{(i)}$, then $\hat{X}^{(i)} \oplus C$ is i.i.d source with distribution 1/2. Let $\beta = \sum_{n=1}^{N} \hat{x}_n^{(i)} \oplus c_n$, then it is well-known [69], the distribution can be approximated as normal distribution $N(N(1-p_i), N(1-p_i)p_i)$ if $\hat{X}^{(i)} = X^{(i)}$. Otherwise, it is approximated by $N(N/2, N/4)$. Thus, we can use this difference to distinguish this two classes. In details, one can choose a proper threshold T. If $\beta > T$, we make decision that $\hat{X}^i$ does not have the correct initial state and thus discard it. Otherwise, we make decision it has the correct initial state and select it as a candidate of the true sequence for LFSRi. It is shown if T is chosen properly, the false alarm probability and missing event probability can exponentially approach zero with the length N of known ciphertext. Thus when N is large enough but linear in length of LFSRi, then the above decision rule will result in unique candidate. If the length of LFSRi is r_i, the running time for this attack is $O(\sum_{i=1}^{s} 2^{r_i})$. Finally, if the feedback polynomial of LFSRs are unknown, we have to try all possible LFSRs. For example, if we know LFSRi is primitive, then we have to try all $\Phi(2^{r_i} - 1)/r_i$ primitive polynomials to the attack above. Thus the running time will change to $O(\sum_1^s 2^{2r_i})$. However, required ciphertext is still linear in r_i according to a formulae in [64].

From the attack procedure above, we can see it require plaintext has bias and key stream is correlated to the input sequences. The former condition is always satisfied in reality. However, the latter condition can be easily avoided by satisfying the first order correlation immunity condition [71, 63]. Furthermore, if for the correct initial sate and some wrong initial states, the corresponding βs are located at the same side of the threshold T, then it is possible to make wrong decision or no solution.

Palit and Roy [68] improved this by selecting the initial state corresponding to the lowest β. They call it as best-is-correct method. They further exploit this basic idea to the case where the combination function is unknown. In detail, they select the input of LFSRi as the correct initial state that maximizes $|\beta - N/2|$ (absolute sign is used because whether 0 has bigger probability is unknown). After obtaining all the initial state of input LFSRs, we try to find the combination function by finding the input-output truth table of this function. Since the combination function output is not available, we have to recover it before beuild this table. Notice the ciphertext is known and correlated to the input LFSRs. ALso the plaintext is assumed biased. For given each input tuple of the combination function, we collect all the cipher bits corresponding to this input tuple and compute string cipher occurence probability given the combination function output is 1 or 0 respectively, decide whether the output is 1 or 0 according which corresponds to the larger conditional probability.

Compared with Siegenthaler's method, Palit and Roys method has wider consideration and more accurate. However, it still assume the combination function is not correlation-immune. Furthermore, his probability calculation is not efficient in real computation. This computation is necessary for estimation of the length of required cipher text.

Eric [18] considered a decimation attack. In this attack, he still considers the divide-and-conquer attack to combination generator. In Siegenthalers divide-and-conquer attack, the required running time is $O(2^{n_i})$ in order to recover the initial state of input sequence i, where the n_i is the length of sequence LFSRi. Before running correlation attack, Eric run a d-decimation of the known cipher text. And then apply the same method as Siegenthaler's to the decimated key stream. Then it is proved the required running time is $(\hat{n}_i 2^{\hat{n}_i})$, where $\hat{n}_i$ the smallest number of n such that $2^n = 1 \bmod \frac{2^{n_i}-1}{gcd(d,2^{n_i}-1)}$. However, it is easy to see the required length of cipher text is at least $O(2^{(max\{n_i\})/2})$. In general, this is impractical. we would like to mention, it is impossible to get the whole initial state of LFSRi by analyzing one piece of d-decimation ciphertext since it only covers $\hat{n}_i$ information bits.

3. Sub-Key Guessing Attack

Zeng, *et, al* [73] noticed that although many stream ciphers have a long secret key, it is possible to determine the whole key by guessing only a sub key and running some consistency test. For LFSR based stream cipher, it is possible to find linear constraints on the key. Thus it is possible to guess a sub-key and then write a system of linear equations of form

$$A(K_1)x = b, \tag{2}$$

where the coefficient matrix $A(K_1)$ is determined by the generating algorithm and b is the known key stream. The solution x can be used to determine the remaining part of the private key. Zeng, *et, al* used this idea to successfully crack Jennings Generator [67], Multiple Speed Generator and Perfect Linear Cipher. We take Jennings Generator to explain how this type of attack works. This generator consists of two LFSRs, LFSR1 and LFSR2. The feedback polynomials of LSFR1 and LFSR2 $f(x), g(x)$, respectively are known. Suppose $deg f(x) = l, deg g(x) = n$. Then the generator is described as follows.

Step 1 LFSR1 is regulary clocked. Let its state at time t is $A(t) = (a_t, a_{t+1}, \cdots, a_{t+l-1})$. For fixed tap positions, $0 \leq n_1 < n_2 < \cdots < n_J$, output $k = a_{t+n_1} \cdots a_{t+n_J}$.

Step 2 $\theta : \{0,1\}^J \to \{0, 1, \cdots, n-1\}$ is a secret function. LSFR2 is also regulary clocked as LFSR1. let $\theta(k) = u$, then at time t, it outputs $b(t+u)$ as the key stream $c(t)$.

The secret key is the initial states of LFSR1 and LFSR2 together with the function θ. Let The attack procedure can be described as follows.

Step 1 For $0 \leq t \leq N$, write

$$x^t = r_{t,0} + r_{t,1}x + \cdots + r_{t,n-1}x^{n-1} \pmod{g(x)},$$

Then we have

$$b(t) = \sum_{i=0}^{n-1} r_{t,i} b(i) \tag{3}$$

Step 2 Guess the initial state for LFSR1 and write the following 2^J linear systems:

$$S_k : A_k x = c_k, 0 \leq k \leq 2^J - 1, \tag{4}$$

by using the equation (3) and J-tuple output of LFSR1. That is, if at time t, LFSR1 outputs k (look k in binary form), then put

$$c(t)=\sum_{i=0}^{n-1} r_{t,i}x_i \text{ to } S_k.$$

Although we do not know the output of the θ function, we indeed know the same input of θ corresponds to the same output. Thats, their corresponding tap positions of LFSR2 are the same. Thus if the initial state of LFSR1 is guessed correctly, S_k should be consistent; otherwise not. And thus check the linear consistency of all $S_k, k=0,1,\cdots,2^J-1$. If any S_k is consistent individualy , then a is reserved as candidate for the initial state of LFSR1. Otherwise, discard it. It is shown that the number of the reserved candidates is very small.

Step 3 For a reserved candidate initial state of LFSR1, solve the linear system $S_k, k=0,1,\cdots,2^J-1$, respectively. If this reserved initial state of LFSR1 is correct, then the initial state of LFSR2 should be one solution of S_k for some $k\in\{0,1,\cdots,2^J-1\}$. Notice the solution for $S_k, k=0,1,\cdots,2^J-1$ is different because their corresponding tap positions are different. However, it is only left to determine which S_k corresponds to the first tap position or the initial state of LFSR2. This can be done since LFSR2 sequences starting with the initial state being the solution of S_k, for each $k=0,\cdots,n-1$ are different by just small shifts.

Step 4 Use initial state of LFSR2 and $S_k, k=0,1,\cdots,2^J-1$, to recover the output sequence of θ. And we can form (input, output) pairs for function θ, which immediately determines the function θ.

Since for each initial state a of LFSR1, we need to run in $O(2^J)$ time, we have the total algorithm in $O(2^{l+J})$ time, where l is the length of LFSR1.

For other examples of sub-key guessing attack, the reader can refer to shrinking generator analysis [16]. Shrinking generator works as follows. Let $a(t), b(t)$ be outputs of LFSR1 and LFSR2. If $a(t)=1$, then $b(t)$ is the output of shrinking generator. Otherwise, discards $b(t)$. It is easy to see if we can correctly guess the initial state of $a(t)$, then half of $b(t)$ can be determined. By solving a linear system, we can find the initial state of $a(t)$ efficiently. Thus, the running time for this simple attack is $O(2^n m^3)$, where n, m are the degrees of known feedback polynomials of LFSR1 and LFSR2 respectively.

From the attacks above, we can see no matter how efficient our algorithm is, a large part of initial state has to be obtained by brute force search. Thus its efficiency is quite limited. Thus, more efficient algorithm should be proposed.

4. Correlation Attack

Ciphertext only attack to combining generator by Siegenthaler [64] is introduced in the last section. It essentially exploited the correlation probability $p_i > 1/2$ between the cipher text and the input LFSRi. That is, recovering the initial state of LFSRi only using $C = X^i + E$ with $P(e_i) = p_i > 1/2$. Therefore, it is enough to design an efficient algorithm for a general model: $Z = A + E$ with $P(z_i = a_i) = p > 1/2$, where A is an LFSR whose feedback polynomial is known. Meier and Staffelbach [59] proposed two algorithms: algorithm A and algorithm B to this problem. Different from the approach in [64], they do not need to carry out brute force search for the initial state for the LFSR. The key idea is (1) building a large set of feedback polynomial of A by using squaring approach on known feedback polynomial $g(x)$. (2) If we run parity check on A with these polynomials, then each check sum will be equal to zero. However, because the presence of noise E, the check sums on Z may not vanish. Let $L_1, \cdots, L_m$ denote the check sums involving bit z_i. Maximum likelihood tells us if most of the $L_i s$ are zero, then it is more likely $z_i = a_i$. Otherwise, it is more likely $z_i = \bar{a}_i$. Algorithm A of Meier and Staffelbach can be briefly described as follows. For each bit a_i, we can compute $P(z_i = a_i | L_1, \cdots, L_m)$. Find the k bits with highest $P(z_i = a_i | L_1, \cdots, L_m)$, where $k = degg(x)$. With them, find the initial state of A by solving linear system. If the solution of the linear system turns not to be the initial state, it is very possible only a small number of the selected k bits in $\{z_i\}$ are wrong. We correct them by trial. It is shown, this algorithm has average running time $O(2^{ck}), c < 1$.

In algorithm B, instead of considering only the k bits with highest posterior Probability, it updates $P(z_i = a_i)$ using $P(z_i = a_i | L_1, \cdots, L_m)$. It complements the bits with very low probability $P(z_i = a_i)$. Iteratively run this procedure until A is recovered. Algorithm B is more efficient and it is a polynomial time algorithm. Meiers method is good. However, they did not show the convergence possibility, the required length of key stream and the success probability. Also to make this attack efficient, it is required the key stream error probability should be small. It is easy to show, the probability that a check sum equals zero is exponentially approach 1/2 with the weight of known feedback polynomial of A. Thus, this method is efficient only if a low weight feedback polynomial is found. If the low weight polynomial has a high degree, the required key stream is longer, which in turn limits the attack efficiency.

Another approach to view the known plaintext attack to the model $B = A + X$ as a noisy decoding problem where a known key stream B of length N is looked the received vector, the first

N terms of LFSR A is a code-word of length N. X is the noise and $P(x(t)=1)=s_0<0.5$. Thus, recovering the initial state of A for a given segment of B is changed to decode A from the B. Now we review the methods in literature for this decoding problem. we first introduce linear syndrome method of Zeng, *et, al* [72, 74]. Key stream B is looked as sequence A corrupted by noise X. Zeng,*et, al* [72] proposed a linear syndrome algorithm to iteratively correct B. This algorithm is described as follows.

Step 1: Find a set of r-nomial multiples of feedback polynomial of A,$r \geq 3$, $g(x)=1+x^{i_1}+\cdots+x^{i_{r-1}}$ of the feedback polynomial of A.

Step 2: Compute an odd number, say $2m+1$, of syndromes

$$\sigma_{i,k}(g(x)) \stackrel{def}{=} \sum_{\rho=0}^{r-1} b(i+i_p-i_k) \tag{5}$$

If at least $m+1$ syndromes $\sigma_{i,k}$, $k=1,2,\cdots,2m+1$ are 1, then define $b(i)=\bar{b}(i)$. Otherwise, define $b'(i)=b(i)$.

Step 3: Use $\{b(t)\}$ replaces $\{b(t)\}$, repeat Step 2 until $b(t)$ can be generated by feedback polynomial of A.

To see the correctness of the above algorithm, it is enough to show $P(x(i)=1)$ will iteratively go to zero. Let s_i be the error rate of A after ith round. Since s_i is determined by s_{i-1} and m if r is fixed, then we can write $s_i=f_m(s_{i-1})$. Further, it is shown $s_i=p-(1-2p)\sum_{k=0}^{m-1}\left(\begin{array}{c}2k+1\\k\end{array}\right)(pq)^{k+1}$ where $p=\frac{1-(1-2s_{i-1})^{r-1}}{2}$, $q=1-p$. It is also shown that if $m>m_c$ for some critical number m_c, then $\{s_i\}$ will strictly decrease to zero. Otherwise, $\{s_i\}$ will strictly increase to 1/2. Thus LS algorithm will successful if and only if the size of set constructed in step 1 is bigger than a critical number m_c.

Zeng,*et, al* [74] improved the algorithm above, we denote as improved LS algorithm (ILS). Before going on, we introduce two concepts.

Definition 4.1. *Supercritical number m_{sc} is the smallest number m such that*

$$s_k=f_{m-k+1}(s_{k-1})<s_{k-1},\ \textit{for all}\ 1\leq k\leq m.$$

t-th cleansing number l_t is the smallest l such that $f_1^{(l)}(s_{m_{sc}})<10^{-t}$.

It is proved that such m_{sc} and l_t do exist for any $0\leq s_0<1/2$.

ILS algorithm is described as follows.

(1) Compute $g_i(x) = g_{i-1}^2(x), i = 1, 2, \cdots, \lceil \frac{2m_{sc}+1}{3} \rceil$, where $g_0(x) = g(x)$ be the feedfack polynomial of A and its weight is only 3.

(2) Compute syndrome using $g_i, i = 0, 1, 2, \cdots, \lceil \frac{2m_{sc}+1}{3} \rceil$, and apply ML rule to correct $b(i), L \leq b(i) \leq N - L - 1$, where $L = L + nL(m), L(m) = 2^{\lfloor \frac{4m-1}{6} \rfloor}$.

(3) $m = m - 1$, go to step 2 until $m = 0$.

(4) Apply LS algorithm to $b(i)$ for $L + n \leq b(i) \leq N - L - n - 1$, using $g(x)$ only. The number of iterative round is l_t.

The objective of step 2 and step 3 is to make LS algorithm with one test polynomial is apllicable. Recall we mention LS algorithm is applicable if and only if the number of parity check polynomial is bigger than the supercritical number m_{sc}. After the error rate drops below a threshold, one polynomial is enough for continuing decoding. This is idea of the algorithm above. Step 2 is to reduce the error rate below that threshold. Note after each round of step 2, the bits close to two ends are discarded since less parity checks are run on them. Step 4 runs the parity check using only one polynomial because of step 2.

It is shown the attack successful if the known key stream length $N \geq c(s_0, t)n$, where $c(s_0, t)$ is a function depends on initial error rate s_0 and a parameter t, n is the degree of known feedback polynomial of A. And the number of iterative decoding rounds does not depend on n. The failing probability of this algorithm will exponentially go to zero with t and computational complexity is linear in n. Although ILS only considers trinomial feedback polynomial of A, it is applicable to general case but indeed the required known key stream has to be longer since supercritical number m_{sc} will increase in this case. As an application, ILS can successfully crack Geffe generator and Beth-Piper generator.

Since when the weight of the known feedback polynomial is large, the bias involved in parity check equation in Zeng or Meier's methods exponentially tends to zero. As result, it is quite inefficient when the weight is large. And on the other hand, a low weight feedback polynomial is hard to find. Johansson and Jonsson [22] avoid this by proposing a new method based on a convolutional code. A theoretical analysis of this algorithm is presented in [24]. Their method still considers model $Z = U + E$, with $P(e_i = 1) = p < 1/2$. It first encodes U to a convolutional code V with memory size B, such that v_i is a sum of a small number t of bits in U. This is achieved by finding a special generating matrix. If R is the result obtained in the same procedure by substitute U by Z. It is immediate that the bias in $\{r_i\}$ is related to t bits of Z. Since t is small, this bias can be kept significantly large. Then apply Viterbi Algorithm to decode U from Z by maximizing the total values where the value on each edge of the trellis tree is defined as $P(r_n|v_n)$. The decoding computational complexity is essentially $O(m2^B)$, where B is memory size (determined by the attacker, the larger it is, the easier to find the special generating matrix needed above) and m be the number of parity check equations which are derived from the special generating matrix. To guarantee parity check equations exist, B can not be small compared to l, the length of LFSR U. However, its advantage is that we do not need to find small weight feedback polynomials. A fast correlation attack based iterative decoding algorithm [25] ot an improved algorithm [49] was propose by Canteau and Trabbia [11]. They claimed it has much better performance. If only the decoding algorithm is concerned, it really has a better performance. However, its precomputation to search for the low weight feedback polynomial is as expensive as $O(2^l)$, which is the brute force search cost. Even using time-memory trade-off is used, it is still very consuming. A recent application of fast correlation attack is the attack to shrinking generator by Golic [26]. It basically first computes the posterior base sequence. He observes the probability away from $1/2$ decreases slowly with length n, thus the attacker can apply hard decision and iterative probabilistic decoding algorithm. Some specific technologies such as sub sequence attack, reinitialization attack and composite attack are also applied in this attack.

5. Fruther Attack to Combination Generator and Filtering Generator

5.1. **Correlation-Immunity.** Recall the divide-and-conquer correlation attack. Notice it is based on the fact that the input component LSFR sequence is correlated to the output sequence. If this condition is not valid any more, this attack can not be successful. Siegentahler [63] formally proposed the concept of correlation immunity for this condition.

Definition 5.1. *Assume x is balanced and its components are n binary and independent random variables. Then a Boolean function $f(x) : Z_2^n \to Z_2$ is called m-th order correlation immune, if $Z = f(x)$ is statistically independent of any m input components.*

A necessary condition for the m-th order correlation immunity of memoryless combining function is given by Siegenthaler [63]. It states, if a Boolean function $f(x)$ is m-th order correlation-immune, then it does not have any term of degree more than $n - m$, where n is the number of the input sequences. Xiao and Massey [71] showed an equivalent condition for m-th order correlation-immunity: Wash transform $F(\omega)$ of $f(x)$ vanishes whenever $1 \le W(\omega) \le m$, where $W(\omega)$ is the hamming weight of ω. This is a direct result of their general Xiao-Massey lemma: $f(x)$ is independent of m input variables if and only if it is independent of arbitrary linear combination of m input variables.

Siegenthaler [66] gives a sufficient condition of m-th order correlation-immunity for finite state machine combiner (a combiner with internal state which will be updated with time): any m inputs and internal state are jointly independent of the output bit.

5.2. **Linear Span of Filtering Generator.** Similar to a combination generator is a filtering generator. Unlike the combination generator, the filtering generator inputs come from a single LFSR. It is not difficult to see if the filtering Boolean function is not chosen properly, the key stream may have a small linear span. Groth [39] construct a generator with its controllable linear span. Key [52] upper-bounded the linear span of k-th order nonlinlear filtering generator with input LFSR length L by $\sum_{j=1}^{k} \begin{pmatrix} L \\ k \end{pmatrix}$. He wrote the output of the generator into the trace representation: $\sum_{W(d_i) \leq k} Tr(c_i \alpha^{d_i L})$, where L is the length of input LFSR. Then it is immediate that the linear span of the generator is $\#\{c_i : c_i \neq 0\}n$. Garcia-Villaba [40] showed the k-th order filtering generator is one-one corresponding to the form $\sum_{W(d_i) \leq k} Tr(c_i \alpha^{d_i n})$. Rueppel [61] gave a root presence test to test whether $c_i \neq 0$. Consider a special case of this filtering generator where the Boolean function contains only one term of order k. For $k < L/2$, the upper-bound of linear span is decided mainly by his k-th order term. Thus to decide the linear span of this k-th order term is very important. In this case, instead of testing all the $c_i \neq 0$, Amparo and Pino [3] showed only d_i with uniformly weighted items have to be tested.

5.3. **Conditional Linear Approximation Attack to Filtering Generator.** Lee, *et, al* [54] proposed a conditional linear approximation attack to general filtering generator. This attack is based the original idea by Anderson [1]. Let $f(x_1, \cdots, x_n)$ be the filtering Boolean function. Let

$$F^m(x_1, \cdots, x_{m+n-1}) = (f(x_1, \cdots, x_n), \cdots, f(x_m, \cdots, x_{m+n-1}). \tag{6}$$

Then define conditional linear approximation of F^m as

$$\lambda_f^m(y, c) = |Pr(x \cdot c = 0 | F^m(x) = y) - 0.5|, c \in Z_2^{m+n-1}, y \in Z_2^m. \tag{7}$$

$$\wedge^m(f) = \mathrm{MAX}\{\lambda_f^m(y, c) | y \in Z_2^m, c \in Z_2^{m+n-1}\}. \tag{8}$$

Then

$$\sum_{i=1}^{m+n-1} c_i x_i = 0, \tag{9}$$

with probability $p+0.5$ for some $0.5 > p > 0$, provided a certain output y (for example, y achieves $\wedge^m(f)$)is fixed. Since the input is LFSR sequence, x_i can be expressed in terms of the initial state. Therefore, (9) can be reduced to $\sum_{i=1}^{k} t_i a_i = 0$ which holds with probability $p+0.5$, where $(a_1, \cdots, a_k)$ is the initial state of LFSR, t_i is the computed coefficients. It is shown $\wedge^m(f)$ is non-decreasing with m. The experiment shows $\wedge^m(f)$ is independent of k. Thus when k is large, we can find small m with $\wedge^m(f)$ close to 0.5. That is

$$\sum_{i=1}^{k} t_i a_i = 0 \tag{10}$$

holds with probability close to 1, for some fixed y. Occurence event of this special y can be scanned through the known key stream. Thus each event, we can derive the equation (10). Thus we can summarize the above as the following attack.

Step 0: Precomputation $D = \{y \in Z_2^m | \lambda_f^m(y, c_y) > p, \text{ for some } c_y \in Z_2^{m+n-1}\}$.
Step 1: For each $y \in D$, find all the relation $\sum_{i=1}^{k} t_i a_i = 0$, where k is the length of LFSR.
Step 2: Picked k equations from Step 2 and solve the initial state and try whether it is correct.
Step 3: Repeat Step 2 until the correct initial state is found.

Precomputation has running time $O(2^{m+n-1} \times 2^m \times 2^{n-1} = 2^{2n+2m-2})$. Take $n = m$, we have $O(2^{4n-2})$. If we model picking process in step 2 as random, then we need $(p+0.5)^{-k}$ trials in step 2 in average. Thus, the running time, is $O(2^{4n} + (p+0.5)^{-k})$, when n is the number of the input of filtering generator. Thus this method is efficient if the number of Boolean function inputs is small and p is large.

5.4. **Intrinsic weakness of Conbination Generator with Memory.** Now let us consider the security of combination generator with memory, which can be modelled as follows

$$S_t = F(X_{t-1}, S_{t-1}), t \geq 1$$
$$Y_t = f(X_t, S_t), t \geq 0,$$

where $F : GF(2)^{N+M} \to GF(2)^M$ is the next-state *vector Boolean function*, $f : GF(2)^{N+M} \to GF(2)$ is the output *Boolean function* and $S_t = (s_{1t}, \cdots, s_{Mt})$ is the state vector at time t. $X_t = (x_{1t}, \cdots, x_{Nt})$ is N-dimensional vector input binary sequences and Y_t is the output bit at time t.

Now we are going to introduce LSCA method [27] to approximate the key stream by linear model. Before going on, we first introduce several useful properties of Boolean function $f(x)$.

Proposition 5.2. *A vector Boolean function $F(X)$ is balanced if and only if all non-zero linear combination of its components are balanced.*

Proposition 5.3. *A vector Boolean function $F(X,Y)$ is independent of X if and only if each non-zero linear function of F is independent of each non-zero linear combination of X.*

Proposition 5.4. *A Boolean function $f(X)$ and a balanced Boolean function $g(X)$ are independent if and only if $f+g$ is balanced.*

Proposition 5.5. *A vector Boolean function $F(X,Y)$ is independent of X if and only if the sum of any nonzero linear combination of $F(X,Y)$ and the non-zero linear combination of X is balanced.*

Now let us come back to combination generator with memory. Let

$$(y_t,\cdots,y_{t-M}) = G(X_t,\cdots,X_{t-M},S_{t-M}),t \geq M). \tag{11}$$

Since by proposition 5.3, $(y_t,\cdots,y_{t-M})$ can not be balanced for any fixed $(X_t,\cdots,X_{t-M})$, we know there exists a linear function L_ω and L_W such that $L_\omega(y_t,\cdots,y_{t-M}) = L_W(X_t,\cdots,X_{t-M}) + \epsilon(X_t,\cdots,X_{t-M},S_{t-M})$, where ϵ is non-balanced Boolean function. If $\{X_t\}$ is i.i.d. balanced and $f(X,S)$ is balanced for each S, then $P(\epsilon=1)$ is a constant (it does not depend on t). Thus using linear sequential circuit approximation (LSCA) technique, we can get approximation of next state function and output function.

$$S_t = AS_{t-1} + BX_{t-1} + U(X_{t-1},S_{t-1}),t \geq 1,$$
$$y_t = CX_t + DS_t + E(X_t,S_t),t \geq 0,$$

where A is $M\times M$ matrix and B is the $M\times N$ matrix, C,D are $1\times N$ and $1\times M$ matrix, respectively. Using the formal power series method, we can solve the above equations:

$$\sum_{k=0}^{M}\phi_k y_{t-k} = \sum_{i=1}^{N}\sum_{k=0}^{M} h_{ik}x_{i,t-k} + \epsilon(X_t,\cdots,X_{t-M},S_{t-M}),t \geq M, \tag{12}$$

Based on U and E is non-balanced, it is highly possible that ϵ is non-balanced too. Thus the above equation can be used to recover the initial state of input LFSR by divide-and-conquer attack, if target input sequence is involved in the above equation.

Similar to the method above, Golic [28] showed the intrinsic weakness for a binary autonomous finite state machine generator, which is defined as follows.

$$S_{t+1} = F(S_t), t \geq 0,$$
$$Y_t = f(S_t), t \geq 0$$

This is the combiner with memory but no input. Thus the equation (12) is reduced to

$$\sum_{k=0}^{M} \phi_k y_{t-k} = \epsilon(S_{t-M}), t \geq M, \tag{13}$$

To distinguish the key stream from truly random sequence, we need $O(1/(P(\epsilon = 1) - 1/2)^2)$ length of key stream. Although the above LSCA approach can find the approximation, sometimes it is easy to find an approximation. Now let us present some linear equation for specific stream cipher.

(1) Clock control shift register (stop is possible). Let output y_k comes from input x_{i_k} and define $d = i_k - i_{k-1}$, then $P(d = 0)$ is the stop probability.

$$y_t + y_{t-1} = e_t, \tag{14}$$

If c is the correlation coeficient of e_t, then $c = P(0)$.

(2) Clock control shift register (no stop operator) if the feedback polynomial of state transition sequence is $f(z) = 1 + \sum_{k=1}^{w} z^{i_k}, 1 \leq i_1 < \cdots < i_w = r$, then $y_t + \sum_{k=1}^{w} y_{t-\hat{i}_k} = e_t, t \geq r$, where $\hat{f}(z) = 1 + \sum_{k=1}^{w} z^{\hat{i}_k}, 1 \leq \hat{i}_1 < \cdots < \hat{i}_w = r$, such that $\hat{i}_k - \hat{i}_{k-1} \leq i_k - i_{k-1}, 1 \leq k \leq w, \hat{i}_0 = i_0 = 0$. Let p be the deletion rate, then it is computed in [28], the correlation coefficient of e_t is approximated by $c \sim (1-p)\left(\frac{2\pi p}{1-p}\right)^{-w/2} \left(\prod_{k=1}^{w}(i_k - i_{k-1})\right)^{-1/2}$ and maximized as $(1-p)\left(\frac{2\pi p}{1-p}\right)^{-w/2} \left(\frac{r-w}{w}\right)^{-w/2}$.

(3) Memoryless combiner

If $y = AX + \epsilon$, and the correlation coefficient is c, then the overall correlation coefficient is c^w where w is the weight of the least common multiple of the input LFSR feedback polynomials.

(4) Shrinking generator.

Applying $p = 1/2$ in item (2), we get c is maximized as $c \sim \frac{1}{2}(2\pi)^{-w/2}(r/w - 1)^{-w/2}$.

6. Some Correlation to Decimation Like Generator

In the previous sections, we have introduced the attack methods for the correlation model $B = A + X$ with $P(x_i = 1) = p < 0.5$. Generators that suffer from this attack include combining generators with memory or without memory both, filtering generator. The common feather of these generators is that the input LFSRs are regularly clocked. Althouhgh some generators with irregularly clocked input LFSRs also suffer from this attack, one can not design a general fast correlation attack model to this type of generator. As a result, one can expect to design a generator with irregularly clocked input LSFR which does not suffer from a fast correlation attack. In general, this type of generator can not be reduced to the model $B = A + x$. Because of its irregularly clocked inputs, it is called clock control generator. A general review of clock control generator is done by Gollman [41]. Now, we introduce some attacks on this type of generator. Let us consider a general model. X is a LFSR sequence. $\{d_t\}$ is a positive integer sequence with $d_i \in [1, d]$ (defined as set $\{1, 2, \cdots, d\}$). For each d_i, its occurrence probability is $P(d_i)$. Then output key stream Y is defined as

$$y_i = x_{k_i}, k_i = \sum_{j=1}^{i} d_j, i = 1, 2, \cdots \tag{15}$$

Thus Y is in fact a decimation of input sequence X. The decimation pattern is defined by a specific cipher. First let us look at embedding correlation attack and probability correlation attack [34].

6.1. **Unconstrained Embedding Attack.** In this attack, for every initial state of X, we try to embed the known key stream Y of length n into a string of length m of X. Since the average jump steps is $\sum_{i=1}^{D} d_i P(d_i)$, thus the deletion rate is $P_d = 1 - (\sum_{i=1}^{d} d_i P(d_i))^{-1}$. Therefore, we should take $m = \frac{n}{1-p_d} + O(\sqrt{n})$, since in this case when the initial state is correct, Y_1^n can be emmebded into X_1^m with high probability. Unconstrained embedding attack is to select the initial state, starting from which X_1^m is generated. And X_1^m can generate the known key stream Y_1^n by some jump sequence $\{d_i\}_1^n$. To make this attack successful, the embedding probability from Y_1^n into a random sequence X_1^m should exponentially approach zero. We denote it as $P_Y(n, m)$. Golic proved $P_Y(n, m) = 1 - 2^{-m} \sum_{k=0}^{n-1} \left(\begin{array}{c} m \\ k \end{array} \right)$ for a general n, m. When consider the length set above, we have $P_Y(n, m) \approx 2^{\frac{n}{1-p_d}(H(p_d)-1)}$. Thus to make this successful, it requires $E[d] < 2$. As an application, shrinking generator, has a deletion rate 0.5. That is, $E[d] = 2$. Thus this attack is not successful on this cipher.

6.2. **Edit Probability Attack.** It is noted in unconstrained embedding attack, we did not use the distribution of the decimation sequence. Let $P_{XY}(e, s)$ denotes the probability that Y_1^s can be embedded into X_1^{e+s} by e deletions. It is easy to prove that

$$P_{XY}(e, s) = P_{XY}(e-1, s)p_d + P_{XY}(e, s-1)(1-p_d)\delta(x_{e+s}, y_s), \tag{16}$$

where $P_{XY}(e,0) = p^e$, $P_{XY}(-1,s) = 0$, $\delta(x_{e+s}, y_s) = \begin{cases} 0.5 & \text{if } x_{e+s} = y_s \\ 0 & \text{Otherwise.} \end{cases}$ Thus we can iteratively compute $P_{XY}(m-n,n)$ in $O((m-n)n)$ time. When the $m = \frac{n}{1-p_d} + O(\sqrt{n})$, the probability attack is to select X that achieves the a very large edit probability that can distinguish from the rest probabilities. Now let us consider the performance of the above method. We model the key stream as input LFSR, which can be looked as linear code, corrupted by independent synchronization error. Thus according to Shannon coding theorem for communication channel with synchronization error, the probability attack is successful if and only if the information rate is below the channel capacity, i.e,

$$\frac{r}{m} < C \text{ or } n > r\frac{1-p_d}{C}, \tag{17}$$

where r is the length of input LFSR. It is shown C is lower bounded by $1 - H(p_d)$, and it is conjectured $C = 1 - H(p_d/2)$. If it is, then the probability attack is successful as long as the length of known key stream is larger than a value linear in r.

6.3. **Constrained embedding attack.** In the above, we did not consider the maximum number d of consecutive deletion. Let denote $P_{d,Y}(n)$ as the probability that Y_1^n can be embedded into a random $X_1^{n(d+1)}$ with non more than d consecutive deletions, then it is proved

$$P_{d,Y}(n) < \left(\left(1 - \frac{1}{2^{d+2}}\right)^{2^{-d+2}2^{d+2}}\right)^n. \tag{18}$$

Thus for this attack to be successful, it is required the length of key stream $n = O(r2^d)$.

A similar probability attack called edit probability attack is successfully applied to Bilateral Stop/Go Generator [75, 76] by Menicocci and Golic [57]. This attack is also another example of divide-and-conquer attack. Another probabilistic correlation attack example (also divide-and-conquer attack) is the attack Shrinking generator by Simpson, *et, al.* [65].

6.4. **Distance Attack.**

6.4.1. *Lenshtein Distance Attack.* Let $X = \{x_k\}$ and $A = \{a_k\}$ be two LFSR sequences. Golic and Mihaljevic [33, 35] proposed a constrained Levenshtein distance (CLD) attack for the following noisy [1,2]-clocked stream cipher $Z = \{z_k\}$:

$$z_k = e_k + y_k, k = 1, 2, \cdots, \tag{19}$$

where $y_k = x_{f(k)}$, $f(k) = k + \sum_{j=1}^{k} a_j$, $P(e_k) = p < 0.5$. We call X as a *Base Sequence* and A as a *Control Sequence.* We always use X and A to represent the true base sequence and control sequence, respectively. To find the initial state of $\{x_k\}$ and $\{a_k\}$, when a partial key stream Z_1^N

and minimal polynomials of X, A are known, Golic and Mihaljevic use CLD between a random sequence $\hat{X}_1^M$ and Z_1^N to help find out the true initial state. Constrained Levenshtein Distance between $\hat{X}_1^M$ and Z_1^N in this stream cipher is defined as follows.

$$\begin{aligned} D(\hat{X}_1^M, Z_1^N) \quad = \quad & \text{minimal number of deletions and complementations required} \\ & \text{to produce } Z_1^N \text{ from } \hat{X}_1^M \text{ by model (19).} \end{aligned}$$

We give the following two remarks.

(1) since $f(N) = N + \sum_{j=1}^N a_j$ has a distribution $\binom{N}{f(N)-N}/2^N$, we have $f(N)$ can be as large as $2N$. Golic and Mihaljevic suggest taking $M = \frac{3N}{2}$, $\frac{3N}{2} + c\sqrt{N}$ or $2N+1$. Since $P(f(N) > \frac{3N}{2}) \approx \frac{1}{2}$, it is not good to take $M = \frac{3N}{2}$. Notice the distribution $\binom{N}{f(N)-N}/2^N$, can be approximated by Gaussian distribution $N(\frac{3N}{2}, \frac{N}{4})$, we have

$$P(f(N) > \frac{3N}{2} + c\sqrt{N}) = \Phi(\frac{c\sqrt{N}}{\sqrt{N/4}}) = \Phi(2c). \tag{20}$$

Thus if we take c large enough, the probability that the original Z_1^N can not be embedded into X_1^M is approaching zero. Thus it is good choice if c is large enough. Since $f(N) \le 2N$, $M = 2N + 1$ will be enough for sure. However it is not clear whether this choice will incur many solutions or not.

(2) Since M is fixed, the number of deletions for any embedding is constant $M - N$. (Note according to Golic and Mihaljevic, the arbitrary initial deletions are allowed.) Thus, statistic D is equivalent to the minimum number of complementations required to produce Z_1^N from $\hat{X}_1^M$ under model (19) with arbitrary initial deletions. Since allowing the arbitrary initial deletions is equivalent to allowing arbitrary tail deletions, we denote the minimal number of complementations from $\hat{X}_1^M$ to produce Z_1^N under model (19) with arbitrary tail deletions as $C(\hat{X}_1^M, Z_1^N)$. Golic and Mihaljevic hopes $D(\hat{X}_1^M, Z_1^N)$, (or equivalently, $C(\hat{X}_1^M, Z_1^N)$) is a sufficient or close to sufficient statistic for the initial state reconstruction problem. In this report, we show it is incorrect for $p > 0.25$. It is noticed that in $p = 0$ case no complementation is allowed and thus it is different from the case $p \neq 0$. Although this is considered in [77], it needs exponential time and exponential size of memory. The result by Zivkovic [77] or a better result by Golic [36] showed the embedding probability from binary string of length N into another randomly selected binary string of the length $2N$ will exponentially go to zero.

Now we will propose an algorithm [50] to show CLD is very possible not to be a sufficient statistic for the initial state reconstruction problem. Suppose Z_1^N is the known key stream and $\hat{X}_1^{2N}$ is a randomly selected binary string of length $2N$. The following embedding algorithm will present an embedding method that embeds Z_1^N into $\hat{X}_1^{2N}$ by model (19) with error free.

Embed$(\hat{X}_1^{2N}, Z_1^N)$
$i = 1; j = 1; C = 0;$
while $i \leq N$
  if $\hat{x}_j = z_i$, then $i++; j++;$
  else if $\hat{x}_{j+1} = z_j$; then $i++; j = j+2;$
    else $C = C+1; i++; j = j+2;$
Return C

Analysis: Since $\hat{X}_1^{2N}$ is i.i.d. source,

$$\begin{aligned}
& P(C \text{ increases at loop } i) && (21)\\
= & \sum_{j=1}^{2N} P(C \text{ increases at loop } i \text{ with corresponding parameter } j) && (22)\\
= & \sum_{j=1}^{2N} P(\hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i, \text{ parameter } j) && (23)\\
= & \sum_{j=1}^{2N} P(\text{ parameter } j | \hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i) P(\hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i) && (24)
\end{aligned}$$

Since changing the value of $\hat{x}_j$ and $\hat{x}_{j+1}$ does not affect the embedding process of $z_1, \cdots, z_{i_1}$, thus it does not affect the determination of j. Therefore, we have

$$P(\text{parameter } j | \hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i) = P(\text{parameter } j) \tag{25}$$

Therefore,

$$\begin{aligned}
P(C \text{ increase at loop } i) &= \sum_{j=1}^{2N} P(\text{ parameter } j) P(\hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i) && (26)\\
&= \sum_{j=1}^{2N} P(\text{ parameter } j) \cdot \frac{1}{4} && (27)\\
&= \frac{1}{4}, && (28)
\end{aligned}$$

because $\hat{X}_1^{2N}$ is randomly selected.
Define $c_i = \begin{cases} 1 & \text{if } C \text{ increases at loop } i; \\ 0 & \text{Otherwise.} \end{cases}$

Then

$$
\begin{aligned}
& P(c_i|c_1,\cdots,c_{i-1}) && (29)\\
= & \sum_{j=1}^{N} P(\hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i, \text{ parameter } j \text{ associated with loop } i|c_1,\cdots,c_{i-1}) && (30)\\
= & \frac{\sum_{j=1}^{N} P(c_1,\cdots,c_{i-1}, \text{ parameter } j|\hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i)P(\hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i)}{\sum_{j=1}^{N} P(c_1,\cdots,c_{i-1}, \text{ parameter } j)} && (31)
\end{aligned}
$$

Because of the same reason as (25), we have

$$P(c_1,\cdots,c_{i-1}, \text{ parameter } j|\hat{x}_j \neq z_i, \hat{x}_{j+1} \neq z_i) = P(c_1,\cdots,c_{i-1}, \text{ parameter } j). \quad (32)$$

Thus, $P(c_i = 1|c_1,\cdots,c_{i-1}) = \frac{1}{4} = P(c_i = 1)$. Therefore,

$$
\begin{aligned}
P(c_1,\cdots,c_n) &= \prod_{i=1}^{n} P(c_i|c_1,\cdots,c_{i-1}) && (33)\\
&= (1/4)^{w(\vec{c})}(3/4)^{n-w(\vec{c})} && (34)\\
&= \prod_{i=1}^{n} P(c_i), && (35)
\end{aligned}
$$

where $\vec{c} = (c_1,\cdots,c_n)$. Therefore, Distribution of variable C in the algorithm above is

$$\begin{pmatrix} N \\ C \end{pmatrix} (1/4)^C(3/4)^{N-C},$$

which can be approximated as Gaussian distribution $N(N/4, 3N/16)$. Let C' be the number of complementation if $\hat{X}_1^{2N} = X_1^{2N}$ and the embedding procedure is controlled by the true control sequence $\{a_k\}$. Then the distribution of C' is $\begin{pmatrix} N \\ C' \end{pmatrix} p^{C'}(1-p)^{N-C'}$ and it is approximated by $N(pN, p(1-P)N)$. The distribution of $C' - C$ is $N((p-1/4)N, (p(1-p)+3/16)N))$. Therefore, if $p > 1/4$, we have

$$P(C \leq C') = 1 - \Phi\left(\frac{(p-1/4)\sqrt{N}}{\sqrt{p(1-p)+3/16}}\right) \to 1 \quad (36)$$

when $N \to +\infty$. Although distance between X_1^{2N} and Z_1^N may be less than C, it is reasonable to believe their difference should be small. When $p > 0.25, C(\hat{X}_1^{2N}, Z_1^N)$ (also $D(\hat{X}_1^{2N}, Z_1^N)$ is very possible not to be a sufficient statistic for the initial state reconstruction problem.

6.4.2. *Edit Distance Attack.* A novel edit distance attack was proposed by Golic and Menicocci [38] to attack the alternating step generator(ASG) [42]. The success of this attack depends on whether the conditional zero distance probability goes to zero exponentially or not. Experimental data in [38] showed it is indeed the case. Jiang and Gong [51] show that both the minimal conditional zero distance probability and average conditional zero distance probability go to zero exponentially. They also prove that the maximum conditional zero distance probability will go to zero if there exists N such that the maximal conditional zero distance probability of length N is less than $\frac{1}{2(N+1)}$.

Alternating step generator was proposed by Gunther [42]. It consists of three LFSRs, $X = \{x_i\}, Y = \{y_i\}$ and $C = \{c_i\}$. The output key stream $Z = \{z_i\}$ is generated from X, Y, C as follows.

(1) Initially, $l = 0, t = 1$.

(2) If $c_t = 1$, then $l = l + 1$;

(3) Output $z_t = x_l \oplus y_{t-l}$;

(4) $t = t + 1$ and go to step (2).

We denote the above generator as $Z = ASG(X, Y; C)$ or $Z_1^n = ASG(X_0^n, Y_0^n; C_1^n)$ for $n \geq 1$. Here X and Y are called *base sequences* and C is called a *control sequence*. For any sequence $W = \{w_k\}$, we use W_i^n to denote the segment of $w_i, w_{i+1}, \cdots, w_n$. Unless a special mention, we use X and Y to denote the true base sequences.

Golic and Menicocci [38] proposed an edit distance correlation attack on this alternating step generator(ASG). They defined the edit distance between a random pair $(\hat{X}_0^n, \hat{Y}_0^n)$ and the known key stream segment Z_1^n as

$$D(\hat{X}_0^n, \hat{Y}_0^n; Z_1^n) = \mathrm{Min}_{C_1^n \in \{0,1\}^n} d_H(Z_1^n, \hat{Z}_1^n), \tag{37}$$

where $\hat{Z}_1^n = ASG(\hat{X}_0^n, \hat{Y}_0^n; C_1^n)$ and $d_H()$ is the Hamming Distance function. If the random pair $(\hat{X}_0^n, \hat{Y}_0^n) = (X_0^n, Y_0^n)$, then $D(\hat{X}_0^n, \hat{Y}_0^n; Z_1^n) = 0$. However, It is easy to see, given Z_1^n, most of the pairs $(\hat{X}_0^n, \hat{Y}_0^n)$ have $\hat{X}_0^n, \hat{Y}_0^n; Z_1^n) \neq 0$. But since $D(\hat{X}_0^n, \hat{Y}_0^n; Z_1^n) = D(\bar{\hat{X}}_0^n, \bar{\hat{Y}}_0^n; Z_1^n)$, the zero distance pair $(\hat{X}_0^n, \hat{Y}_0^n)$ is not unique. Golic and Menicocci hoped the probability of the zero pair $(\hat{X}_0^n, \hat{Y}_0^n)$ event conditional on the known key stream of length n approaches zero exponentially with n. If this is the case, one can easily determine n such that the number of the zero distance pairs $(\hat{X}_0^n, \hat{Y}_0^n)$, when it goes through all the possible initial states, is very small. We write this conditional zero distance probability given Z_1^n as $P_n(Z_1^n)$, *i.e.*

$$P_n(Z_1^n) = Pr((\hat{X}_0^n, \hat{Y}_0^n) : D(\hat{X}_0^n, \hat{Y}_0^n; Z_1^n) = 0 \text{ for given } Z_1^n). \tag{38}$$

Define

$$P_n^{max} = \mathrm{Max}_{Z_1^n} P_n(Z_1^n), \bar{P}_n = E[P_n(Z_1^n)], P_n^{min} = \mathrm{Min}_{Z_1^n} P_n(Z_1^n).$$

Experimental data in [38] show that $P_n^{max}, \bar{P}_n$ and P_n go to zero exponentially with n. In details,

$$P_n^{max} = 0.72 \cdot 0.915^n, P_n^{min} = 2.7 \cdot 0.562^n, \bar{P}_n = 0.83 \cdot 0.83^n. \quad (39)$$

Notice 0.915 and 0.83 are close to 1. The exhaustive search for P_n^{max} was only done for $n \leq 13$. For P_n^{min} and $\bar{P}_n$, it was only done for $n \leq 20$. Thus, it is not clear whether these probabilities exponentially approach zero with n. Jiang and Gong [51] show that $\bar{P}_n$ and P_n^{min} exponentially go to zero with n. They also prove that P_N^{max} will go to zero exponentially if there exists N such that $P_N^{max} < \frac{1}{2(N+1)}$.

7. Time/Memory/Data Trade-off Attacks

M. E. Hellman [47] proposed a memory-time trade-off technique for attacking block cipher (in fact it is also applicable for one-way functions). Golic [32] and Babbage [13] independently described a time/memory trade-off attack to stream cipher. Biryukov and Shamir [5] improved the their results based on Hellman's approach. We first introduce Hellman's approach and Golic's approach, then introduce Biryukov and Shamir's improvement.

Suppose P_0 is the fixed plaintext. S is the encryption algorithm and k is the encryption key. Define $f : f(k) = R(S_k(P_0))$, where R is a simple reduction such that $f(k)$ and P_0 have the same length. The Hellman trade-off method can be described as follows

(1) Precomputation: Randomly selects m plaintexts, $SP_1, SP_2, \cdots, SP_m$. Let $X_{i0} = SP_i$, define $X_{ij} = f(X_{i,j-1}), 1 \leq j \leq t$ and $EP_i = X_{it}$, then store $\{(SP_i, EP_i)\}_{i=1}^m$.

Suppose $C_0 = S_K(P_0)$ be the known cipher and K is the secret key we want to recover.

(2) $Y_1 = R(C_0) = f(K)$.
If $Y_1 \neq EP_i$ for any $1 \leq i \leq m$, then $K \neq X_{i,t-1}, i = 1, 2, \cdots, m$. If $Y_1 = EP_i$ for some i, then either $K = X_{i,t-1}$ or EP_i has more than one inverse image. This can be differentiated by evaluating $f(x)$ starting from SP_i till $X_{i,t-1}$ and checking whether $S_{X_{i,t-1}}(P_0) = C_0$. If $K \neq X_{i,t-1}$, then compute $Y_2 = f(Y_1)$ and run the same procedure as Y_1. If the key K is not found, then try $Y_3 = f(Y_2), \cdots, Y_t = f(Y_{t-1})$.

It is proved this method has success probability essentially lower bounded by $\frac{mt}{N}$, provided $mt \ll N$. Thus if we build up t storage tables as we do in step 1 by applying different reduction functions, it is highly possible that the method above will succeed if $mt^2 = N$.

Since in this case, the memory is $M = mt$, running time is essentially $T = t^2$, thus the time-memory trade-off curve is $M^2T = N^2, (1 \leq T \leq N)$. Golic [32] has trade-off curve $MT = N, 1 \leq T \leq D$ for stream cipher. He first builds M random (initial state, prefix-output) pairs, sorts and stores them. When carrying out real attack, he divides the known key stream of length $D + n$ into D block where n is the length of the initial state. For each block, look it up in the stored table. The attack is successful, if a block is found in the table. Using birthday paradox, we immediately get the trade-off claimed above.

Biryukov and Shamir [5] combined these two techniques. Different from block cipher, the stored table in stream cipher can apply to any known key stream prefix. Thus in order for a collision between storage and actual states, we only need to cover about N/D states, where D is the size of known key stream minus n. Therefore, if we consider Hellmans attack, we can reduce the number of tables to t/D. The total memory is reduced to $M = mt/D$, since each table needs still m memory. Precomputation $P = N/D$ since there are t/D tables and we assume mt^2 is about N. Running time $T = t/D \times t \times D = t^2$, unchanged. Thus we get trade-off curve $TM^2D^2 = N^2$, for $D^2 \leq T \leq N$, since $t \geq D$.

Biryukov and Shamir noticed a practical problem: random access to a hard disk is much more expensive than computation. In the above technique, the number of times for random access to hard disk is $T = t^2$. They apply Ronald Rivest's special technique to reduce the number of hard disk operations to t. In details, The output prefix component of each entry in the stored table is a special point (for example, the first k bits are zeros). When we run the actual attack, if a special point is not encountered, we do not need to check the table in the hard disk. For each stored pair, only one time hard disk access is required. Thus the total access number is t. In the above trade-off, we notice, the condition $T \geq D^2$ is a little strict if available data is large. Also it is not difficult to see the rate of the special point can be chosen freely. Biryukov and Shamir apply a BSW sampling technique [6] to reduce this cost. It still use the above tables. However, intermediate points are special points. Then the space of special output is mapped to input states easily. Thus, we can directly use the trade-off curve with data size DR and search space size NR, where R is the special points rate. That $TM^2(DR)^2 = (NR)^2$, with $T \geq D^2R^2$, i.e., $TM^2D^2 = N^2$ with $T \geq D^2R^2$. Thus the restriction is reduced to $T \geq D^2R^2$ and random access to hard disk is further reduced to tR.

8. Attack to A5/1

A5/1 is the Europian GSM encryption standard for cellular mobile phone. It has been caught cryptanlysts' attendions. Quite a few attacks are proposed for it. In this section, we are going to introduce some of them. A5/1 is composed of three LFSRs of length 19, 22, 23, respectively. Their feedback polynomials are all primitive. Their clocking rules are provided by a majority function. Detailed description is presented at [32]. Now we introduce some attacks by Golic. He first observed the brute force search space is only $2^{63.32}$ instead of 2^{64} since 3/8 of the states do not have preccessors when one consider the inversion of an internal state. Then he guesses n bits for each LFSR starting from the bit at clock-control tap position for each LFSR. Thus these $3n$ bits give $3n$ independent equations for the internal states, provided $n \leq 18$. And since in average the $3n$ bits provide $4n/3$ well-defined clocking, $1 + 4n/3$ additional equations are obtained. However, these $3n + 4n/3 + 1$ equations may not be independent. To avoid this problem, n should be no more than 12. Eventhough, when $n > 12$, the additional equations can provide the consistency test for the guessed $3n$ bites. Take $n = 10$, then only $63.32 - (3 \times 10 + 4 \times 10/3 + 1) = 19.02$ bits of internal states left unsolved. They can be determined by reversion tree method. Each node in the tree is an internal state. Nodes leaving a given node A are all the possible internal states which is one step before state A. Because of the clocking rule and the known key stream constraint, it can be shown the outgoing degree of the node in the reversion tree is 2.5 in average. Notice in average, each LFSR has $m = 19.02/3$ left to be solved, which requires $4m/3$ clocking, which, on the other hand, means the depth of tree is about $4m/3$ for the determination of the left internal states. Thus search tree requires $2.5^{4m/3} = 2^{11.16}$. Thus the total computation for the internal states needs $2^{30+11.16} = 2^{41.16}$. After deriving certain internal states of tree LFSRs ($S(101)$ in the description of A5/1), solving the initial states is simply done by guessing the clocking times of each LFSR starting from initial states to $S(101)$ and solving this linear system. This number is average 76 and standard deriviation is 4.35.

Except guessing attack, Golic proposed a Time-Memory trade-off attack. Recall we mentioned this in the last section. But that is for the general case. We describe it for A5 case. Golic noticed from K conversations, the attacker can get $102 \times K$ 64-bit blocks of the known key stream. If we build up M sorted (internal state, output block) pairs, then the kown $102 \times K$ blocks and M output blocks are very possible to intersect if $102K \times M \geq 2^{63.32}$. Since the time required to find a certain common block from M output blocks and $102K$ known block is $T = 102 \times K log M$. Thus time-memory trade-off curve is $TM \geq 2^{63.32}, T < 102 \times 2^{22}$, where $logM$ factor is ignored and upperbound of K is 2^{22} according to the description of A5.

Now we consider the internal state reversion attack via branching process. This attack is just to consider all the possibilities of $S(t-n)$ given $S(t)$. If the number of the possibilities of $S(t-1)$ is Z where the known key stream information is ignored, then it is shown $P(Z=0)=3/8, P(Z=1)=13/32, P(Z=2)=P(Z=3)=3/32, P(Z=4)=1/32$. The possibilities from $S(t)$ to $S(t-n)$ can be explained by building a reversion tree. It is well explained by the theory of branching process [2, 43]. Thus if we denote the size of depth n as Z_n, and the size till depth n as Y_n, then $Y_n = O(n^2), Z_n = O(n)$ with high probability. If we consider the known key stream information, it is shown $Y_n = O(n), Z_n < 2$ with high probability. Note the constant hidden in the big O is very small. Since Y_n represents the attacking cost to go through all the node no deeper than n and Z_n is the number of solutions at the depth n. It follows the attack is very efficient and solution is almost unique. Then this method can be used to find the initial state from $S(101)$ or find the session key from $S(0)$ and the public key. Note the former case can also be solved by the reversion attack via guessing clocking number of each LFSR presented before while the latter case can not be solved by it since a public key is involved when generating $S(0)$ from $S(-22)$.

Following Golics work, Biryukov, Shamir and Wagner [6] further attacked A5. In their attack, a special pattern is chosen and pre-process random samples. When running the real attack, they wait for the special pattern to occur in the known key stream. And then use the pre-process and further data to determine the internal states. As result, they need pre-computation $2^{42} \sim 2^{48}$. Biham and Dunkelman [4] improved the above two methods by waiting for a special event to occur and then using several guessing techniques to exploit this special event. As result, it need $2^{20.8}$ bits known key stream and $2^{39.91}$ running time in unit of one A5/1 clocking. Recently, Krause [53] propose a new approach based on a free binary decision diagram. His approach can apply to many stream cipher. As an example, he yields a running rime 2^{41}.

9. Distinguish from Truly Random Sequence

In this section, we will introduce distinguishing from truly random sequence attack. In this attack, a segment of binary string together with a key stream generating algorithm is known. This binary string is drawn equaly likely according to cipher model or according to truly random model. The objective of the attacker is to decide which model it is drawn according to. Note even optimal decision rule will have decision errors. The reason is, all the binary strings can be generated by both cipher model and truly random model. However, if this string is taken according to cipher model, the distribution of the binary string is highly different from that of random sequence. In general, the longer the binary string is, the more the difference is. Taking advantage of this difference, the attacker may learn some strategy to decide which model this string was drawn from. For a good decision rule, the probability of the correct decision should be better than that of coin toss decision. The correct decision probability minus one half is called the advantage of the attacker's distinguishing strategy. The attack is considered to be successful if his advantage is better than one half signicantly. Golic [28, 30, 31] presented a distinguishing attack to binary combiner with memory. Eksahl and Jonhansson showed a distinguishing attack [20] to SOBER-t16 [45] and SOBER-t32 [46]. Coppersmith, Halevi and Jutla [9] proposed a distinguishing attack to a class of stream cipher which has a linear masking. As examples, they applied this attack to newly proposed SNOW stream cipher [19] and Scream stream cipher [44].

In the following we will introduce the work by Coppersmith, *et al.* Their method essentially uses the statistical distance between cipher model and random model and then enlarge it such that the attacker can comfirm the given string is taken from cipher distribution or random distribution by considering an appropriate long key stream. We first define a notion of statistical distance.

Definition 9.1. *Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two distributions over some finite domain X. The statistical distance between $\mathcal{D}_1$ and $\mathcal{D}_2$ is defined as*

$$|\mathcal{D}_1 - \mathcal{D}_2| = \sum_{x \in X} |\mathcal{D}_1(x) - \mathcal{D}_2(x)| \tag{40}$$

Let $\mathcal{D}^N$ be the product distribution of $\mathcal{D}$ with components being term wise independent. It is shown in [70] that if $|\mathcal{D}_1 - \mathcal{D}_2| = \epsilon$ and take N properly large tetween $\Omega(1/\epsilon)$ and $O(1/\epsilon^2)$, then we can have $|\mathcal{D}_1^N - \mathcal{D}_2^N| \approx 1$. Also we have XOR-lemma: Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two distributions over $\{0,1\}^k$, let $\mathcal{D}_3 = \mathcal{D}_1 + \mathcal{D}_2$, and denote by $\mathcal{U}$ the uniform distribution over $\{0,1\}^k$, and $\epsilon_i = |\mathcal{U} - \mathcal{D}_i|$, then $\epsilon_3 \leq \epsilon_1\epsilon_2$. Now given a binary string x, a decision rule DR needs to decide it is generated according to distribution $\mathcal{D}_1$ or according to distribution $\mathcal{D}_2$. We define the statistical advantage of DR as

$$adv(DR) = \frac{1}{2}(Pr_{\mathcal{D}_1}(DR(x) = 1) + Pr_{\mathcal{D}_2}(DR(x) = 2)) - \frac{1}{2}. \tag{41}$$

It is easy to show if we apply maximum-liklihood rule:

$$ML(x) = \begin{cases} 1 & \text{if } \mathcal{D}_1(x) > \mathcal{D}_2(x); \\ 2 & \text{otherwise,} \end{cases}$$

then $adv(ML) = |\mathcal{D}_1 - \mathcal{D}_2|/4$. If we take $\mathcal{D}_1$ as Cipher distribution and $\mathcal{D}_2$ as Random distribution, then the distinguishing attack is looked as successful if $adv(DR) > s$, where s is a positive constant.

In the method of Coppersmith, *et al.*, the attacker first looks for a "good linear tranformation" for the consecutive outputs and then distinguish this linear transformation from truly random by collecting lots of data. Let us first define cipher distribution and random distribution, given a linear tranformation $l : \{0,1\}^{2n} \to \{0,1\}$.

Cipher distribution: $\mathcal{D}_c =< l(x_i + y_j, NF(x_j) + z_j) >_{j=1,2,\ldots}$, where x_j is independently and randomly taken from $\{0,1\}^n$ and $\{y_j\}$ and $\{z_j\}$ are taken from an appropriate linear subspace.

Random Distribution: $\mathcal{D}_r =< l(x_j, x'_j) >_{j=1,2,\ldots}$ x_j and x'_j are independently and randomly taken from $\{0,1\}^n$.

9.1. **Linear Attack.** In this attack, the attacker tries to find a linear approximation for the non-linear function. *i.e.*, he tries to function $l : \{0,1\}^{2n} \to \{0,1\}$, such that $l(x, NF(x))$ is biased. Let $\sigma_j = l(x_j + y_j, NF(x_j) + z_j)$. Since $\{y_j\}$ and $\{z_j\}$ are taken from linear space, we can find set J such that $\sum_{j\in J} y_j = \sum_{j \in J} z_j = 0$. Thus,

$$\sum_{j\in J} \sigma_j = \sum_{j\in J} l(x_j, NF(x_j)) + \sum_{j \in J} l(y_j, z_j) = \sum_{j\in J} l((x_j, NF(x_j)).$$

Thus if $l(x, NF(x))$ has a bias of ϵ, then $\sum_{j\in J}\sigma_j$ has a bias of $\epsilon^{|J|}$. Also it is proved in [9], $|\mathcal{D}_1 - \mathcal{D}_2| \le \sqrt{\sum_{r=1}^N A_N(r)\epsilon^{2r}}$. And because it is "smooth", it is expected $|\mathcal{D}_1 - \mathcal{D}_2| \approx \sqrt{0.8\sum_{r=1}^N A_N(r)\epsilon^{2r}}$. This result can be expained as each J provids a statistical evidence of weight $\epsilon^{2|J|}$. Thus all the J gives $\sum_{r=1}^N A_N(r)\epsilon^{2r}$ statistical evidence for distinguishing attack. If the sum is close to 1, then we immediately have the attacking advantage is close to $1/4$. The attack on SNOW is done by finding a linear approximation to nonlinear FSM. And apply the attack model above. As result, it needs 2^{95} words of key stream and work-load 2^{100}.

9.2. **Low diffusion attack.** Consider cipher model and random model below.

Cipher $\mathcal{D}_c =< (w_j = u_j + v_j, w'_j = u'_j + v'_j) >_{j=1,2,\ldots}$, $u'_j s$ are uniform and independent in $\{0,1\}^m$, $u'_j = f(u_j)$, $f : \{0,1\}^m \to \{0,1\}^{m'}$ is known and $v_1 v'_1 v_2 v'_2 \cdots$ are taken from a linear subspace.

Random $\mathcal{D}_r =< (w_j, w'_j) >_{j=1,2,\ldots}$, where w_j, w_j are uniform and independent random variables in $\{0,1\}^{m'}$.

Let $D_f^n =< d = \sum_{j=1}^n u_j, d' = \sum_{j=1}^n f(u_j) >, f : \{0,1\}^m \to \{0,1\}^{m'}$. Then it is shown $E_f[|D_f^n - R|] \leq c(n)2^{(m'-(n-1)m)/2}$, where $c(n)$ is a fixed function of n. Coppersmith, *et, al* expected it is approximated by $0.8c(n)2^{(m'-(n-1)m)/2}$.

Now let us describe the distinguishing procedure for the above two model.

(1) Precomputation: Build up a table with entry $< d, d', b >$ for all matched $< d, d' >$ defined above, where b is the bit information represents whether $P(d, d') > 2^{-m-m'}$.

(2) Given $< w_j, w'_j >_{j=1,2,\ldots}$, generate the corresponding sequence $\{b_j\}$ by looking up the probability information of $< w_j, w'_j >$. From the distance result above and basic statistical distance properties presented before, we have the required text $2^{(n-1)m-m'}/c(n)$.

(3) Apply a simple majority vote on sequence $\{b_j\}$ to determine whether it is from cipher model or from random model.

Now let us look at the attack cost. Precomputation needs $2^{m+m'}$ memory. Table building up needs $O((m+m')2^{m+m'}logn)$ time using Wash-Hadamard transform [62]. As an attack example, coppersmith, *et, al* get the result for attacking Scream [44], the attacker needs 2^{43} bytes key stream, 2^{80} time, and space 2^{50}.

10. Conclusion

In this report, we review the methods of stream cipher cryptanalysis in the literature. We explain each paper's key idea and some main techniques. Some evaluations, comparisons bewteen related papers are also given. Some other cryptanalysis of stream ciphers are not introduced here in detail but we list here for convenient reference. Fluhrer and McGrew [21] poposed a method for distinguishing RC4 from truly random sequence, which requires $2^{30.6}$ bytes key stream. Canteaut and Filiol [7] improved the ciphertext only attack to combination generator with unknown combination function introduced by Palit and Roy [68]. Mihaljevic, Fossorier and Imai [55] proposed a new fast correlation attack based a novel construction of parity check equations. Chepyzhov,*et. al* proposed a simple algorithm for fast correlation attack for stream ciphers. An algorithmic view of correlation attack is presented in [8]. The reader refers to [10, 60, 23, 14, 56, 58] for more fast correlation attacks. Golic [29] proposed an inversion attack to nonlinear filtering generator.

References

[1] R. J. Anderson, Searching for the Optimum Correlation Attack. *Fast Software Encryption 1994,* B. Preneel (Ed.), LNCS 1008, Springer-verlag, 1995, pp. 137-143.
[2] K. B. Athreya and P. E. Ney, *Branching Processes*, Berlin, Springer-Verlag, 1972.

[3] A. Fster-Sabater, P. Caballero-Gil, On the Linear Complexity of Nonlinear Filtered PN-sequences, *Advances in Cryptology-ASIACRYPT'94,* J. Pieprzyk and R. Safavi-Naini (eds.), LNCS 917, Springer-verlag, pp. 80-90, 1995.

[4] E. Biham, O. Dunkelman, Cryptanalysis of the A5/1 GSM Stream Cipher, *Progress in Cryptology-INDOCRYPT'00,* LNCS 1977, Springer- verlag, 2000, pp. 43-51.

[5] A. Biryukov and A. Shamir, Cryptanalytic Time/Memory/Data Tradeoffs for Stream Ciphers, *Advances in Cryptology-ASIACRYPT'00*, T Okamoto (Ed.), LNCS 1976, Springer-verlag, 2000, pp. 1-13.

[6] A. Biryukov, A. Shamir and David Wagner, Real Time Cryptanalysis of A5/1 on a PC, *Fast Software Encryption 2000,* B. Schneier (Ed.), LNCS 1978, Springer-verlag, 2001, pp. 1-18.

[7] A. Canteaut and E. Filiol, Ciphertext Only Reconstruction of Stream Ciphers Based on Combination Generators, *Fast Software Encryption 2000*, B. Schneier (Ed.), LNCS 1978, Springer-verlag, 2001, pp. 165-180.

[8] P. Chose, A. Joux and M. Mitton, Fast Correlation Attacks: An Algorithmic Point of View, *Advances in Cryptology-EUROCRYPT'02*, L. R. Knudsen (Ed.), LNCS 2332, Springer-verlag, 2002, pp. 209-221.

[9] D. Coppersmith, S. Halevi and C. Jutla, Cryptoanalysis of srteam ciphers with linear masking, to appear in Crypto2002. Available at http://eprint.iacr.org/2002/020

[10] A. Clark, J. D. Golic, E. Dawson, A Comparison of Fast Correlation Attacks, *Fast Software Encryption 1996*, D. Gollmann(Ed.), LNCS 1039, Springer-verlag, 1996, 145-157.

[11] A. Canteaut and M. Trabbia, Improved Fast Correlation Attacks Using Parity-Check Equations of Weight 4 and 5, *Advances in Cryptology- EUROCRYPT'00*, B. Preneel (ed.), LNCS 1807, pp. 573-588, 2000.

[12] W. Blaser and P. Heinzmann, New Cryptographic device with high security using public key distribution, IEEE student papers, pp. 150, 1979-1980.

[13] S. Babbage, A space/Time trade-off in exhaustive search attacks on stream cipher, *European Convention on Security and Detection,* IEEE Conference publication, No. 408, May 1995.

[14] V. Chepyzhov and B. Smeets, On A Fast Correlation Attack on Certain Stream Ciphers, *Advances in Cryptology-EUROCRYPT'91*, D. W. Davies (Ed.), LNCS 547, Springer-verlag, 1991, pp. 176-185.

[15] V. V. Chepyzhov, T. Johansson and B. Smeets, A Simple Algorithm for Fast Correlation Attacks on Stream Ciphers, *Fast Software Encryption 2000*, B. Schneier (Ed.), LNCS 1978, Springer-verlag, 2001, pp. 181-195.

[16] D. Coppersmith, H. Krawczys and Y. Mansour, The shrinking generator, *Advances in Cryptology-Crypto'93*, D. R. Stinson (ed.), LNCS 773, Springer-Verlag, 1994, pp. 22-39.

[17] D. Coppersmith, S. Halevi and C. Jutla, Cryptanalysis of stream ciphers with linear masking, to appear in crypto 2002.

[18] E. Filiol, Decimation Attack of Stream Ciphers, *Progress in Cryptology-INDOCRYPT'00,* LNCS 1977, Springer-verlag, 2000, pp. 31-42.

[19] P. Ekdahl and T.Johansson, Snow-a new stream cipher, NESSIE Submission, available at http://www.cosic.esat.kuleuven.ac.be/nessie/workshop/submissions.html

[20] P. Ekdahl and T. Johansson, Distinguishing Attacks on SOBER-t16 and t32, Fast Software Encryption 2002, J. Daemen and V. Rijmen(Eds.), Spinger-verlag, LNCS 2365, 2002, p. 210-224.

[21] S. R. Fluhrer and D. A. McGrew, Statistical Analysis of the Alleged RC4 Keystream Generator, *Fast Software Encryption 2000*, B. Schneier (Ed.), LNCS 1978, Springer-verlag, 2001, pp. 19-30.

[22] T. Johansson, F. Jnsson, Improved Fast Correlation Attacks on Stream Ciphers via Convolutional Codes, *Advances in Cryptology-EUROCRYPT'99,* J. Stern (ed.),LNCS 1592, pp. 347-362, 1999.

[23] T. Johansson, F. Jnsson, Fast Correlation Attacks through Reconstruction of Linear Polynomials, *Advances in Cryptology-CRYPTO'00*, M. Bellare (Ed.), LNCS 1880, Springer-verlag, 2000, pp. 300-315.

[24] T. Johansson, F. Jnsson, Theoratical Analysis of a Correlation Attack Based on Convolutional Codes, *IEEE Transactions on Information Theory*, Vol. 48, No. 8, August 2002, pp. 2173-2181.

[25] R. G. Gallager, Low-density parity-check codes, *IRE Trans. Inform. Theory,* IT-8, pp21-28, 1962.

[26] J. D. Golic, Correlation Analysis of the Shrinking Generator, *Advances in Cryptology-CRYPTO'01*, J. Kilian (ed.), LNCS 2139, Springer-Verlag, pp. 440-457, 2001.

[27] J. Dj. Golic, Correlation via Linear Sequential Circuit Approximation of Combiners with memory, *Advances in Cryptology- EUROCRYPT'92*, R. A. Rueppel (Ed.), LNCS 658, Springer-verlag, 1993, pp. 113- 123.

[28] J. Dj. Golic, Intrinsic Statistical Weakness of Keystream Generators, *Advances in Cryptology-ASIACRYPT'94*, J. Pieprzyk and R. Safavi-Naini (Eds.), LNCS 917, Springer-verlag, 1995, pp. 91-103.

[29] Jovan Dj. Golic: On the Security of Nonlinear Filter Generators, *Fast Software Encryption*, D. Gollmann(Ed.), LNCS 1039, Springer-verlag, 1996, 173-188.

[30] J. D. Golic, Correlation Properties of a General Binary Combiner with Memory, *Journal of Cryptology,* Vol. 9 No. 2, pp. 111-126, 1996.

[31] J. D. Golic, Linear Models for Keystream Generators, *IEEE Transactions on Computers,* Vol. 45, No. 1, pp. 41-49, 1996.

[32] J. Dj. Golic: Cryptanalysis of Alleged A5 Stream Cipher, *Advances in Cryptology-EUROCRYPT'97*, W. Fumy (Ed), LNCS 1233, Springer-verlag, 1997, pp. 239-255.

[33] J. D. Golic, M. J. Mihaljevic, A Noisy Clock-Controlled Shift Register Cryptanalysis Concept Based on Sequence Comparion Approach, *Advances in Cryptology - EUROCRYPT'90*, Ivan Damgard (ed.), LNCS 473, Springer-verlag, 1990, pp 487-491.

[34] J.D.Golic and L.O'Connor, Embedding and probability correlation attacks on clock-controlled shift registers, *Advances in Cryptology, Eurocrypto'94*, LNCS 950, A.D.Santis(Ed.), Spring- Verlag, pp.230-243, 1994.

[35] J.D. Golic and M.J. Mihaljevic, A generalized correlation attack on a class of stream ciphers based on the Levenshtein distance, *Journal of Cryptology*, 3(3): pp.201-212, 1991.

[36] J. D. Golic, Constrained Embedding Probability for Two Binary Strings, *SIAM Journal on Discrete Mathematics* Vol. 9, No.3, pp. 360-364, 1996.

[37] S. Golomb, *Shift Register Sequences,* 2nd edition, Aegean Park Press, 1982.

[38] J. D. Golic, Renato Menicocci, Edit Distance Correlation Attack on the Alternating Step Generator. *Advances in Cryptology-CRYPTO'97*, 499-512

[39] E. J. Groth, Generation of Binary sequences with controllable complexity, *IEEE Transactions on Information Theory,* IT-17, pp. 288-296, May 1971.

[40] L. J. Garca-Villalba and A. Fster-Sabater, On the General Classification of Nonlinear Filters of m-Sequences, *Information Processing Letters*, Vol. 69, No. 5, pp. 227-232, 1999.

[41] D. Gollman and W. G. Chambers, Clock-controlled Shift Register: A Review, *IEEE Journal on Selected Areas in Communications,* Vol. 7, pp. 525-533, May, 1989.

[42] C.G. Gunther, Alternating Step Generators Controlled by de Bruijn Sequences, *Advances in Cryptology-Eurocrypto'88*, S. Goldwasser (Ed.), LNCS 403, Springer-Verlag, 1988, pp. 88-92.

[43] T. H. Harris, *The Theory of Branching Processes,* Berlin, Springer- verlag, 1963.

[44] S. Halevi, D. Coppersmith, and C. Jutla, Scream: A Software-Efficient Stream Cipher, *Fast Software Encryption 2002,* J. Daemen and V. Rijmen(Eds.), Spinger-verlag, LNCS 2365, 2002, pp. 195-210.

[45] P. Hawkes and G. Rose, Primitive specification and supporting documentation for SOBER-t16, submission to NESSIE. In Proceedings of the First Open NESSIE Workshop, 13-14 November 2000, Heverlee, Belgium.

[46] P. Hawkes and G. Rose. Primitive specification and supporting documentation for SOBER-t32 submission to NESSIE. *the First Open NESSIE Workshop,* 13-14 November 2000, Heverlee, Belgium.

[47] M. E. Hellman, A Cryptanalytic Time-Memory Trade-Off, *IEEE Transactions on Information Theory,* Vol. IT-26, No. 4, July 1980, pp. 401-406.

[48] T. Beth, Z. D. Dai, On the Complexity of Pseudo-Random Sequences - Or: If You Can Describe a Sequence It Can't be Random, *Advances in Cryptology-EUROCRYPT'89*, J. Quisquater and J. Vandewalle (Eds.), LNCS 434, Springer-verlag, 1990, pp. 533-543.

[49] J. Hagenauer, E. Offer, and L. Papke, Iterative decoding of binary block and convolutional codes, IEEE Trans. Inform. Theory, vol. 42, No. 2, pp. 429- 445, 1996.

[50] S. Jiang and G. Gong, On Constrained Levenshtein Distance Attack to A stream Cipher, Manuscript.

[51] S. Jiang and G. Gong, On Edit Distance Attack to Alternating Step Generator, submitted.

[52] E. L. Key, Analysis of the Structure and Complexity of Nonlinear Binary Sequence Generators, *IEEE Transactions on Information Theory,* IT-22, pp. 732-736, Nov. 1976.

[53] M. Krause, BDD-Based Cryptanalysis of Keystream Generators, *Advances in Cryptology-EUROCRYPT'02*, L. R. Knudsen (Ed.), LNCS 2332, Springer-verlag, 2002, pp. 222-237.

[54] S. Lee, S. Chee, S. Park and S. Park, Conditional Correlation Attack on Nonlinear Filter Generators, *Advances in Cryptology- ASIACRYPT'96,*, K. Kim and T. Matsumoto (Eds.), LNCS 1163, Springer-verlag, pp. 360-367, 1996.

[55] M. J. Mihaljevic, M. P. C. Fossorier and H. Imai, A Low-Complexity and High-Performance Algorithm for the Fast Correlation Attack, *Fast Software Encryption 2000*, B. Schneier (Ed.), LNCS 1978, Springer-verlag, 2001, pp. 196-212.

[56] Miodrag J. Mihaljevic, Marc P. C. Fossorier, Hideki Imai: Fast Correlation Attack Algorithm with List Decoding and an Application, *Fast Software Encryption 2001*, M. Matsui (Ed.), LNCS 2355, Springer-verlag, 2002, pp. 196-210.

[57] R. Menicocci and J. Dj. Golic, Edit Probability Correlation Attack on the Bilateral Stop/Go Generator, *7th IMA Interantional Conference,* M. Walker (Ed.), LNCS 1746, Springer-verlag, 1999, pp. 201-212.

[58] R. Menicocci and J. Dj. Golic, Correlation Attacks on Up/Down and Stop/Go Cascasdes, *IEEE Transactions on Information Theory*, Vol. 45, No. 2, March 1999, pp. 486-498.

[59] W. Meier, O. Staffelbach, Fast Correlation Attacks on Certain Stream Ciphers, *Journal of Cryptology,* Vol. 1, No. 3, pp. 159-176, 1989.

[60] W. T. Penzhorn, Correlation Attacks on Stream Ciphers: Computing Low-Weight Parity Checks Based on Error-Correcting Codes, *Fast Software Encryption*, D. Gollmann(Ed.), LNCS 1039, Springer-verlag, 1996, 159-172.

[61] R. A. Rueppel, *Analysis and design of stream ciphers,* Springer- verlag, New York, 1986.

[62] D. sundararajan, *the Discret Fourier Transform: theory, algorithms and applications,* World Scientific Pub Co., 2001.

[63] T. Siegenthaler, Correlation-Immunity of Nonlinear Combining Functions for Cryptographic Applications, *IEEE transactions on information theory,* Vol. IT-30, No.5, September 1984.

[64] T. Siegenthaler, Decrypting a Class of Stream Cipher Using Ciphertext Only, *IEEE Transactions on Computers,* Vol. 34, No. 1, January,1985.

[65] L. Simpson, J. Dj. Golic and E. Dawson, A Probabilistic Correlation Attack on the Shrinking Generator, *ACISP'98,* C. Boyd and E. Dawson (eds.), LNCS 1438, Springer-verlag, 1998, pp. 147-158.

[66] T. Siegenthaler, Design of Combiners to Prevent Divide and Conquer Attacks, *Advances in Cryptology-CRYPTO'85*, H. C. Williams (Ed.), LNCS 218, Springer-verlag, 1986, pp. 273-279.

[67] S. M. Jennings, *A Special Class of Binary Sequences,* University of London, 1980, ph.D Thesis.

[68] S. Palit and K. Roy, Cryptanalysis of LFSR-Encryption Codes with Unknown Combining Function, A*dvances in Cryptology-ASIACRYPT'99,* K. Y. Lam, *et. al* (eds.), LNCS 1716, Srpinger-verlag, 1999, pp306-320.

[69] H. Cramer, *Mathematical methods of statistics,* Princeton University Press, Princeton, N.J. 1946.

[70] S. P. Vadhan, *A Study of Statistical Zero-Knowledge Proofs,* PhD thesis, MIT Department of Mathematics, August 1999, section 3.1.

[71] Z. Xiao and J. L. Massey, A special characterization of correlation-immunity combining functions, *IEEE transactions on information theory,* Vol. 34, No. 3, May 1988, pp 569-571.

[72] K. Zeng, M. Huang, On the Linear Syndrome Method in Cryptoanalysis, *Advances in Cryptology-CRYPTO'88,* S. Goldwasser(Ed.), LNCS403, Springer-verlag, 1990, pp469-478.

[73] K. Zeng, C. H. Yang, T. R. N. Rao, On the Linear Consistency Test (LCT) in Cryptanalysis with Applications, *Advances in Cryptology-CRYPTO'89,* G. Brassard (Ed.), LNCS435, Springer-verlag, 1990, pp164-174.

[74] K. Zeng, C. H. Yang, T. R. N. Rao, An Improved Linear Syndrome Algorithm in Cryptanalysis With Applications, *Advances in Cryptology-CRYPTO'90,* A. Menezes and S. A. Vanstone (Eds.), LNCS537, Springer-verlag, 1991, pp34-47.

[75] K. Zeng, C. H. Yang, T. R. N. Rao, Large Primes in Stream Cipher Cryptography, *Advances in Cryptology-AUSCRYPT'90*, J. Seberry and J. Pieprzyk (eds.), LNCS 453, Springer-veralg, 1990, pp. 194-205.

[76] K. Zeng, C. H. Yang, D. Y. Wei, T. R. N. Rao, Pseudorandom Bit Generators in Stream-Cipher Cryptography, *IEEE Trasactions on Computers*, Vol. 24, No. 2, pp. 8-17, 1991.

[77] M. V. Zivkovic, An Algorithm for the Initial State Reconstruction of the Clock-Controlled Shift Register, *IEEE Transactions on Information Theory*, Vol.37, pp. 1488-1490, Sep. 1991.