Journal of Universal Computer Science, vol. 15, no. 15 (2009), 2981-2998
submitted: 1/2/09, accepted: 29/8/09, appeared: 1/9/09 © J.UCS

Graph-Based Approach to the Edit Distance Cryptanalysis of Irregularly Clocked Linear Feedback Shift Registers

Pino Caballero-Gil
(University of La Laguna, La Laguna, Spain
pcaballe@ull.es)

Amparo Fúster-Sabater
(Institute of Applied Physics (C.S.I.C.), Madrid, Spain
amparo@iec.csic.es)

Candelaria Hernández-Goya
(University of La Laguna, La Laguna, Spain
mchgoya@ull.es)

Abstract: This paper proposes a speed-up of a known-plaintext attack on some stream ciphers based on Linear Feedback Shift Registers (*LFSRs*). The algorithm consists of two basic steps: first, to guess the initial seed value of one of the *LFSRs*, and then to use the resulting binary sequence in order to deduce useful information about the cipher parameters. In particular, the proposed divide-and-conquer attack is based on a combination of graph-based techniques with edit distance concepts. While the original edit distance attack requires the exhaustive search over the set of all possible initial states of the involved *LFSR*, this work presents a new heuristic optimization that avoids the evaluation of an important number of initial states through the identification of the most promising branches of the search graph. The strongest aspects of the proposal are the facts that the obtained results from the attack are absolutely deterministic, and that many inconsistent initial states of the target *LFSRs* are recognized and avoided during search.
Key Words: symmetric cryptography, attack, linear feedback shift register
Category: D.4.6, E.3, K.6.5

1 Introduction

Stream ciphers are based in the generation of long pseudorandom keystream sequences from short keys in such a way that it is not possible to rebuild the short keys from the knowledge of the keystream sequences. This paper deals with stream ciphers based on Linear Feedback Shift Registers (*LFSRs*), and in special with the Shrinking [Cop 1994] and the Alternating Step [Gün 1988] generators. The keystream sequences produced by each generator have high linear complexity, long period and good statistical properties [Gol 1989].

Attacks on stream ciphers are usually performed under a known-plaintext hypothesis, that is to say, it is assumed that the attacker has direct access to the keystream output from the generator [Jia 2002]. The computational complexity of such attacks is then compared with the one of the exhaustive search, and the cipher is said to be broken if the former is smaller. This theoretical definition might look useless, but in fact is

very important for the analysis of the security of stream ciphers because it can reveal weaknesses that might lead to practical attacks.

This paper proposes a deterministic improvement of a known-plaintext three-step divide-and-conquer attack first proposed in [Gol 1991] by means of a theoretical model and a distance function known as Levenshtein or edit distance. The three steps of the attack are:

1. Guess the initial state of an *LFSR* component of the generator.
2. Try to determine the other variables of the cipher based on the intercepted keystream.
3. Check that the guess was consistent with observed keystream sequence.

This work proposes an approach that may be seen as an extension of the constrained edit distance attack to clock-controlled *LFSR*-based generators presented in [Kan 2003] and generalized in [Cab 2005]. Our main goal here is to investigate whether the number of initial states that have to be analyzed can be reduced. This feature was pointed out in [Gol 1998] as one of the most interesting problems in the cryptanalysis of stream ciphers.

According to the original method, the attacker needs to traverse an entire search tree including all the possible *LFSR* initial states. However, in this work the original attack is improved by simplifying the search tree in such a way that only the most efficient branches are retained. In order to achieve such a goal, cut sets are defined in certain graphs that are here used to model the original attack. This new approach produces a significant improvement in the computing time of the original edit distance attack since it implies a dramatic reduction in the number of initial states that need to be evaluated. This quantitative improvement is well established through implementation for the specific cases of Shrinking and Alternating Step Generators. Furthermore, it is remarkable that, unlike previous attacks, the results obtained with the proposal of his work are fully deterministic.

The structure of the present work is as follows. Section 2 introduces basic definitions of Shrinking and Alternating Step Generators and essential concepts regarding edit distances. In Section 3, some ideas for an efficient initial state selection method are given that allow describing a method for deducing a threshold value for the computation of the edit distance. Section 4 presents the full description of the proposed attack. Finally, Section 5 contains specific details for the cases of Shrinking and Alternating Generators and Section 6 provides some results from experimental implementations. Finally, some conclusions and open questions complete the paper.

2 Background

2.1 Shrinking and Alternating Step Generators

The Shrinking Generator (*SG*) is a nonlinear combinator based on two *LFSRs*, introduced in 1993 by Coppersmith, Krawczyk and Mansour [Cop 1994]. In this generator

the bits produced by one *LFSR*, denoted by *S*, are used to determine whether the corresponding bits generated by the second *LFSR*, denoted by *A*, are used as part of the overall keystream or not.

The Alternating Step Generator (*ASG*) is a nonlinear combinator based on three *LFSRs*, proposed in 1987 by G*ü*nther [G*ü*n 1988]. According to this generator each bit produced by one of the three *LFSRs*, denoted by *S*, is used to determine the output from the other two *LFSRs A* and *B*. In this work we use a modified and equivalent version of the original *ASG* defined as follows. If *S* produces a 1, then the output bit of the generator is the output bit from the *LFSR A*, which is clocked. Otherwise, if *S* produces a 0, then the output bit of the generator is the output bit from the *LFSR B*, which is clocked.

The notation used within this work is as follows. The lengths of the *LFSRs S*, *A* and *B* are denoted respectively by L_S, L_A and L_B. Their characteristic polynomials are respectively $P_S(x)$, $P_A(x)$ and $P_B(x)$, and the sequences they produce are denoted by $\{s_i\}$, $\{a_i\}$ and $\{b_i\}$. The output keystream is $\{z_j\}$.

Both generators have been the object of many different attacks during the last years [Ekd 2003] [Mol 2004] [Zen 2004] [Eng 2006] [Jun 2006] [Kha 2007] [Gom 2008]. However, despite the simplicity of their design and the high number of attacks, both generators remain being considered remarkably resistant to practical cryptanalysis because there is no known attack that may be considered efficient enough when the *LFSRs* are too long for exhaustive search. Indeed, each of the above references may be described either as a theoretical attack because it requires hard hypothesis and the obtained results are probabilistic, or as an attack launched only against the hardware implementation of the generator.

2.2 Edit Distance

The edit or Levenshtein distance is the minimum number of elementary operations (insertions, deletions and substitutions) required to transform one sequence X of length N into another sequence Y of length M, where $M \leq N$. Some applications of the edit distance are file checking, spell correction, plagiarism detection, molecular biology and speech recognition. The dynamic programming approach (like the shortest-distance graph search and Viterbi algorithm) is a classical solution for computing the edit distance matrix where the distances between prefixes of the sequences are successively evaluated until the final result is achieved.

When applying an edit distance attack on a clock-controlled stream cipher, the objective is to compute the initial state of a target *LFSR* that is a component of the attacked generator. As in Viterbi search, this problem has the property that the shortest path to a state is always part of any solution of which such a state is part. We will be able to see this fact quite clearly by the formalization of the algorithm as a search through a graph.

When a restriction exists on a maximum number of times that the register may be clocked before an output bit is produced, clock-controlled registers are said to work

with constrained clocking. For these registers, attacks based on a so-called constrained edit distance have been proposed and analyzed in [Gol 1991]. In this work, two different possible models for the attacked generator, shown respectively in Fig. 1 and Fig. 2, are considered. In both cases it is assumed that the feedback polynomial of the target *LFSR* is known.

In the first model it is assumed that $Y = \{y_n\}$ is an intercepted keystream segment of length M, which is seen as a noisy decimated version of a segment $X = \{x_n\}$ of length N produced by a target *LFSR*. On the other hand, according to the second model, it is assumed that $X = \{x_n\}$ is an intercepted keystream segment of length N, which is seen as a noisy widened version of a segment $Y = \{y_n\}$ of length M produced by a target *LFSR*. In this latter case, insertions in the sequence Y are indicated by two sequences S and B so that S points the locations where the bits of B must be inserted. The simplest examples represented by these theoretical models are *SG* and *ASG*, respectively. Furthermore, in both cases it is not necessary to consider noise to model the generator.

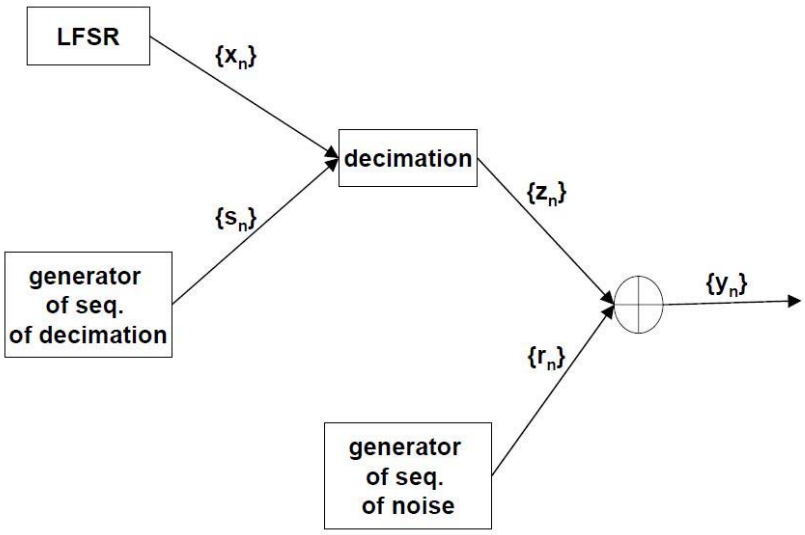

Figure 1: Theoretical model with decimations

According to these theoretical models, the main objective of the attack is to deduce some initial state of the target *LFSR* that allows producing an intercepted keystream sequence through decimation or insertion, respectively, without knowing the decimation

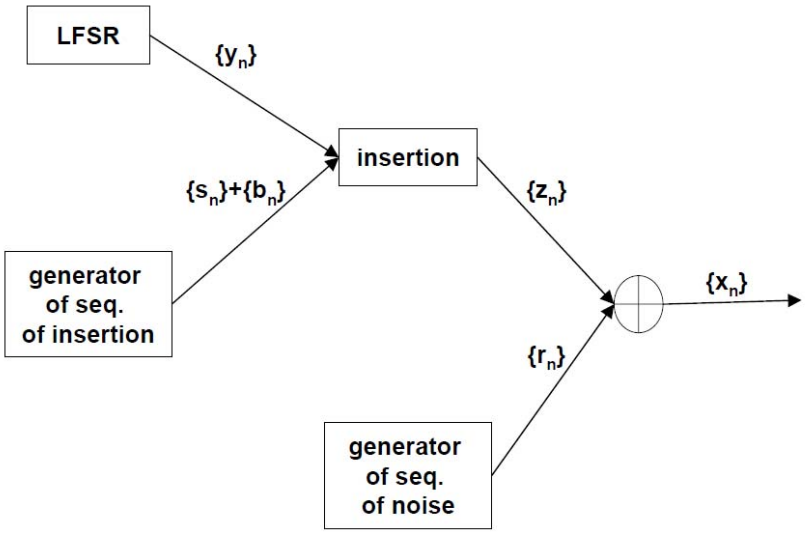

Figure 2: Theoretical model with insertions

or insertion sequences. In either case, the attack is considered successful if only a few initial states are identified.

Note that the use of these models implies that the known-plaintext attack is applicable not only to those generators that fit exactly such a model but also to other sequences produced by more complex generators that generalize the given description. In this latter case, the attack would provide a simpler equivalent description of the original attacked generator.

An essential step in the edit distance attacks is the computation of edit distance matrices $W = (w_{i,j}), i = 0,1,\ldots,N-M, j = 1,2,\ldots,M$ associated each one with a couple of sequences X and Y where Y is the intercepted keystream sequence and X is a $LFSR$ sequence produced by one possible initial state.

It is remarkable the fact that the computation of the edit distance matrix requires choosing between both models the one that better fits the attacked generator. If it is the first model, the intercepted sequence is Y while X is the candidate sequence. Otherwise, if it is the second model, then the intercepted sequence is X while Y is the candidate sequence. Also note that from the computation of the edit distance between X and Y, the edit sequences that are computed in the first case correspond to decimation sequences while in the second case they correspond to insertion sequences.

Some of the parameters of such a matrix are described below. Firstly, its dimension

is $(N-M+1)\cdot M$. Furthermore, its last column gives the edit distance between X and Y thanks to the value $\min_{i=0,\ldots,N-M}\{w_{i,M}+N-M-i\}$. Lastly, each element of the matrix but the last column $w_{i,j},\ i=0,\ldots,N-M,\ j=1,\ldots,M-1$ corresponds exactly to the edit distance between prefix sub-sequences $x_1,x_2,\ldots,x_{i+j}$ and $y_1,y_2,\ldots,y_j$. The edit distance between prefix sub-sequences $x_1,x_2,\ldots,x_{i+M}$ and Y are given by $w_{i,M}+N-M-i,\ i=0,\ldots,N-M$.

In the edit distance attack here analyzed only deletions and substitutions are allowed. Consequently, each element $w_{i,j}$ of the edit distance matrix W may be recursively computed from the elements of the previous columns according to the formulas in Equation (1), which depend exclusively on the coincidence or difference between the two bits x_{i+j} and y_j.

$$\begin{gathered} w_{i,1}=P_i(x_{i+1},y_1),\ i=0,\ldots,N-M \\ w_{0,j}=w_{0,j-1}+P_0(x_j,y_j),\ j=2,\ldots,M \\ w_{i,j}=\min_{k=0,\ldots,i}\{w_{i-k,j-1}+P_k(x_{i+j},y_j)\},\ i=1,\ldots,N-M,\ j=2,\ldots,M \\ P_k(x_{i+j},y_j)=\begin{cases} k & if\ x_{i+j}=y_j \\ k+1 & if\ x_{i+j}\neq y_j \end{cases},k=0,...,i \end{gathered} \tag{1}$$

$P_k(x_{i+j},y_j)$ gives the cost of the deletion of k bits previous to x_{i+j} plus its substitution by its complementary if $x_{i+j}\neq y_j$. Note that a maximum length k of possible runs of decimations is assumed for constrained edit distance matrices. It is also remarkable that at each stage the minimum has to be obtained in order to extend the search at a next stage, which implies the need to maintain a record of the search in the same way that Viterbi algorithm saves a back pointer to the previous state on the maximum probability path.

2.3 Graph-Based Approach

In order to avoid the computation of the edit distances for all possible initial sequences, in this paper we propose a graph-theoretic approach so that the computation of edit distances may be seen as a search through a basic graph. Such a basic graph, shown in Fig. 3, is a directed graph where each vertex denoted by $(i+j,j), i=0,1,,N-M; j=1,2,,M$, indicates a correspondence between the bits x_{i+j} and y_j and each edge indicates either a deletion of the bit x_{i+j} when $j=0$, or a possible transition due to a deletion (D) or a substitution (S), in the remaining cases. In this way, the computation of edit distances consists in finding the shortest paths through such a graph.

For the description of our improvement, we define a weighted directed graph, here called induced graph, where the costs of shortest paths come directly from the elements of the matrix W. This induced graph is computed from the basic graph shown in Fig. 3 as follows. If we eliminate vertical edges in the graph of Fig. 3 by computing the partial transitive closure of every pair of edges of the form $((i+j-2,j-1),(i+j-1,j-1))$

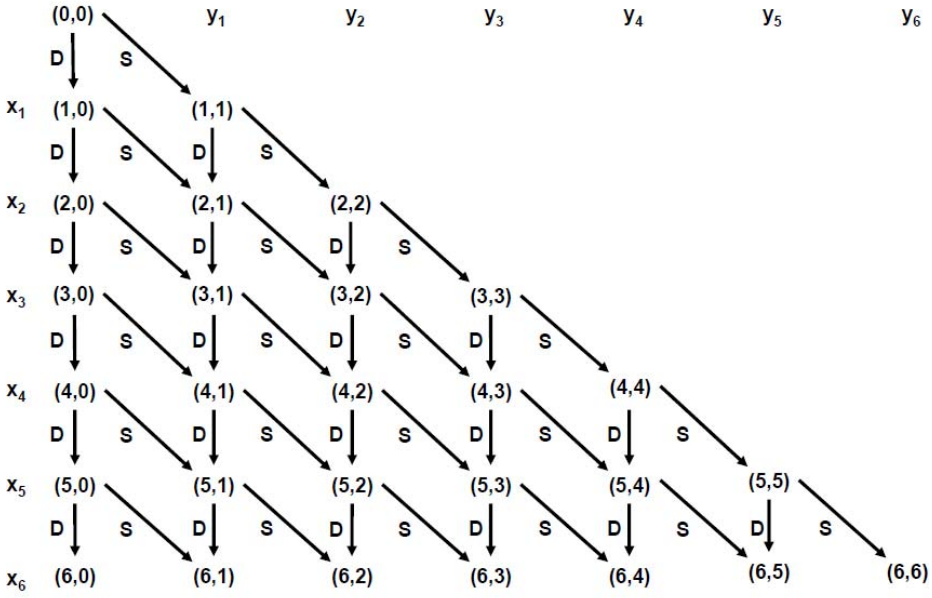

Figure 3: Basic graph

and $((i+j-1,j-1),(i+j,j))$ and substituting them by the edge $((i+j-2,j-1),(i+j,j))$, we get the graph shown in Fig. 4, which is here called induced graph.

In this graph there are as many vertices as elements in the matrix W, plus an additional source and an additional sink. On the other hand, the directed edges in this induced graph are defined from the computation of the edit distances described in Equation (1), plus additional edges joining the source with the vertices associated to the first column of W and additional edges joining the vertices associated to the last column of W with the sink.

For instance, the induced graph corresponding to a constrained edit distance matrix with runs of decimations of maximum length 1 has $(N-M+1)\cdot(2M-N+2)$ vertices and $2\cdot(N-M+1)\cdot(2M-N+2)-M-3$ edges. Moreover, as described in Equation (1), the edges in the induced graph have different costs depending on the specific pair of sequences X and Y, and, in particular, on the coincidences between the corresponding bits of both sequences. Note that in the induced graph, the shortest paths between the source and the sink provide us with the solution of the cryptanalytic attack through the specification of both the decimation and the noise sequences that can be extracted from them.

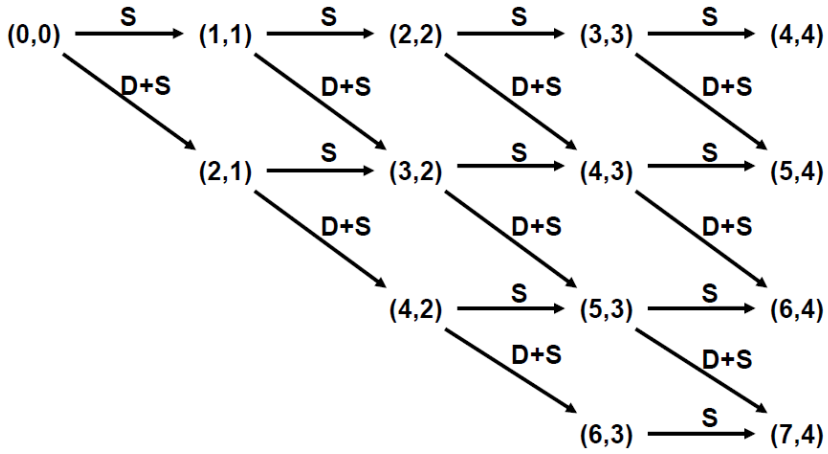

Figure 4: Induced graph

Example : For an intercepted keystream sequence Y:1010111 of length M=7 and a candidate sequence X:1001011111 of length N=10, the constrained edit distance matrix with runs of decimations of maximum length 1 is:

$$W = \begin{pmatrix} 0 & 0 & 1 & 2 & 3 & - & - \\ 2 & 1 & 1 & 1 & 1 & 1 & - \\ - & 4 & 3 & 3 & 2 & 2 & 2 \\ - & - & 5 & 5 & 4 & 3 & 3 \end{pmatrix}.$$

The graph induced by this matrix is shown in Fig. 5 where there are exactly 24 vertices and 38 edges of which the ones belonging to the 12 shortest paths between the source and the sink are remarked in bold.

For each one of those 12 optimal paths, we obtain a possible solution to the crypt-analysis. The computation of 12 decimation sequences S may be deduced from the above graphical representation in the following way. A horizontal edge in an optimal path is interpreted as a 0 in a deduced decimation sequence (that is to say, no deletion of the corresponding bit) whereas each oblique edge in the path gives a 1 and a 0 as two decimation bits (corresponding to the deletion of one bit).

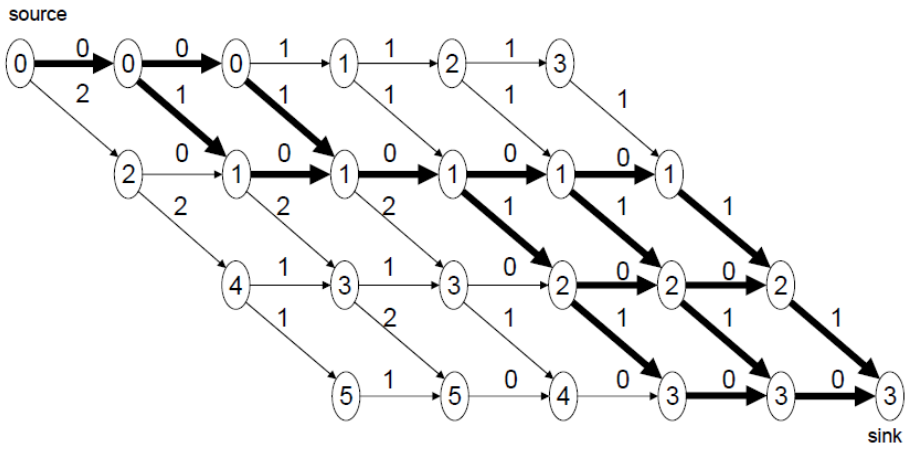

Figure 5: Shortest paths

$$S = \{s_n\} : \begin{cases} 0010000101: \mathit{Solution}1 \\ 0010001001: \mathit{Solution}2 \\ 0010001010: \mathit{Solution}3 \\ 0010010001: \mathit{Solution}4 \\ 0010010010: \mathit{Solution}5 \\ 0010010100: \mathit{Solution}6 \\ 0100000101: \mathit{Solution}7 \\ 0100001001: \mathit{Solution}8 \\ 0100001010: \mathit{Solution}9 \\ 0100010001: \mathit{Solution}10 \\ 0100010010: \mathit{Solution}11 \\ 0100010100: \mathit{Solution}12 \end{cases}$$

3 Pruning Search Graphs

The main idea behind the method shown in this Section comes directly from the association between bits x_{i+j} and edges of the induced graph. Since the calculation of the minimum edit distance implies the computation of some shortest path in such a graph, cut sets between the source and the sink in the induced graph may be useful in order to define a set of conditions for candidate sequences so that it is possible to establish a minimum threshold edit distance. In this way, once an intercepted sequence fulfills some of those stated conditions, the cost of the corresponding cut set can be guaranteed to be minimal for some possible candidate sequence, what has direct consequences on the costs of the shortest paths, that is to say, on the edit distances.

In this way, as soon as an intercepted sequence fulfills some specific condition defined below, and this fact allows the description of a candidate and feasible initial sequence, we will know that such an initial sequence will provide us with a useful upper threshold for the edit distance and even in many cases, such a sequence will be a minimum edit distance sequence.

The specific cut sets that we have used for the numerical results shown in this work are defined as follows. Each cut set $C_{i+j}, 2 \leq i+j \leq N-1$ contains:

1. The set of all the arcs corresponding to the vertex x_{i+j}.
2. All those edges corresponding to bits x_w with $w > i+j$ whose output vertex is one of the output vertices of the former set.

For the first model, these cut sets may be characterized by several independent conditions on the intercepted sequence Y that may be used to guarantee a decrease on the edit distances of different candidate sequences X. After having checked each hypothesis separately, the tools used to check both sets of conditions on candidate sequences X are described in terms of a pattern that is made out of independent bits of X according to the formulas in Equation (2).

$$\begin{aligned}
&\text{If } \forall j: 2,3,\ldots,N-M+1;\ y_1 = y_2 = \cdots = y_j \text{ then } y_j = x_1 = x_2 = \cdots = x_{j+N-M}\\
&\text{If } \forall j: N-M+2, N-M+3,\ldots,M; y_{M-N+j} = \cdots = y_{j-1} = y_j \text{ then } y_j = x_j = x_{j+1} = \cdots = x_{j+N-M}\\
&\text{If } \forall j: M+1, M+2,\ldots,N-1;\ y_{M-N+j} = \cdots = y_{M-1} = y_M \text{ then } y_M = x_j = x_{j+1} = \cdots = x_N
\end{aligned} \tag{2}$$

For the second model, the cut sets may be characterized by different independent conditions on the intercepted sequence X that may be used to guarantee a decrease on the edit distances of candidate sequences Y. After having checked each hypothesis separately, the tools used to check both sets of conditions on candidate sequences Y are described in terms of a pattern that is made out of independent bits of Y according to the formulas in Equation (3).

$$\begin{aligned}
&\text{If } \forall j: 2,3,\ldots,N-M+1;\ x_1 = x_2 = \cdots = x_{j+N-M} \text{ then } x_1 = y_1 = y_2 = \cdots = y_j\\
&\text{If } \forall j: N-M+2, N-M+3,\ldots,M; x_j = x_{j+1} = \cdots = x_{j+N-M} \text{ then } x_j = y_{M-N+j} = \cdots = y_{j-1} = y_j\\
&\text{If } \forall j: M+1, M+2,\ldots,N-1; x_j = x_{j+1} = \cdots = x_N \text{ then } x_j = y_{M-N+j} = \cdots = y_M
\end{aligned} \tag{3}$$

For checking previous equations (2) and (3), it is necessary to determine the value of N, which depends on k that is the maximum length of possible runs of decimations. For example, if $k = 1$, then $N = 3M/2$, which is the mathematical expectation of N in such a case.

Note that the checking procedure of hypothesis described with the previous Equations, applied on the intercepted sequence takes polynomial time as it implies a simple verification of runs. The previous patterns allow discovering promising initial states producing sequences with a low edit distance. In fact, such a pattern provides a good quality threshold for the method that will be described in the following Section.

4 Attack Algorithm

The threshold obtained through the pattern described in the previous Section is a fundamental ingredient of the general attack described below. The algorithm here developed also makes use of a new concept, the so-called *stop column*, which leads to a considerable saving in the computation of the edit distance matrices. Indeed, a *stop column* with respect to a threshold T may be defined as a column j_0 of the edit distance matrix W such that each one of their elements fulfills the Equation (4).

$$w_{i,j_0} > T - (N - M - i), \forall i \tag{4}$$

Once a minimum edit distance threshold has been obtained, we may use such a threshold to stop the computation of any matrix W as soon as a *stop column* has been detected. This is due to the fact that the edit distance corresponding to the candidate initial state will be worse than the threshold. In this simple way, two new improvements on the original attack may be achieved. On the one hand, as yet mentioned, the computation of any matrix may be stopped as soon as a *stop column* is obtained. On the other hand and thanks to the association between bits x_{i+j} and edges of the graph, we may define an anti-pattern on the initial states of the target $LFSR$, the so-called $IS-anti-pattern$. This new parameter allows us to discard the set of initial states fulfilling such an $IS-anti-pattern$ when an early *stop column* has been detected. This is so because once a *stop column* has been obtained, it is possible to discard directly all the initial states whose first bits coincide with those that produce the *stop column*. In order to take full advantage of *stop columns*, it is convenient to have some efficient way of obtaining a good threshold. That is exactly the effect of the pattern described in the previous Section.

Since it is possible that the described pattern correspond only to sequences that may not be produced by the target $LFSR$, in practice it is convenient to restrict the pattern to the length of the target $LFSR$. So, the pattern obtained from the first formulas of Equations (2) and (3) limited to the length of the target $LFSR$ is what we call $IS-pattern$. On the other hand, although sequences generated through the $IS-pattern$ have minimum edit distance, it is possible that the corresponding obtained decimation

or insertion sequences and noise sequences are not consistent with the description of the attacked generator. This is the reason why the proposed algorithm includes a process of hypothesis relaxation, which implies the successive complementation of bits of the $IS-pattern$ until getting a positive result.

Finally, since the $IS-pattern$ is determined by the runs at the beginning of the intercepted sequence, if no long run exists at the beginning of the sequence, initially the algorithm discards the first bits in the intercepted sequence before a long run, and use those discarded bits to confirm the result of the attack. This idea is expressed within the algorithm by a parameter $H \in [0,L]$, chosen by the attacker depending on its computational capacity (the greater capacity, the fewer H).

The full description of the proposed general edit distance attack is as follows.

Algorithm

Input: The intercepted keystream sequence and the feedback polynomial of the target $LFSR$ of length L.

Output: The initial states of the target $LFSR$ producing sequences with a low edit distance with the intercepted sequence, and the corresponding decimation or insertion sequence and noise sequence.

1. Verification of hypothesis on the intercepted sequence described in Equation (2) or (3).
2. While fewer than H hypothesis are fulfilled, discard the first bit and consider the resulting sequence as new intercepted sequence.
3. Definition of the $IS-pattern$ according to the first L formulas in Equation (2) or (3).
4. Initialization of the threshold $T = N$.
5. For each initial state fulfilling the $IS-pattern$, which has not been previously rejected:
   (a) Computation of the edit distance matrix, stopping after detecting a *stop column* according to threshold T and Equation (4).
   (b) Definition of the $IS-anti-pattern$ and rejection of all initial states fulfilling it.
   (c) Updating of the threshold T.
6. For each initial state producing a sequence with minimum edit distance:
   (a) Computation of the shortest paths from the graph induced by the edit distance matrix.

(b) Translation from each shortest path into decimation or insertion sequences and noise sequences.

(c) Checking that the obtained decimation or insertion sequences, and noise sequences are consistent with the attacked generator. Otherwise, updating of the $IS-pattern$ by complementing one of the bits in the original $IS-pattern$.

Note that if the output is not the minimum edit distance sequence, the obtained edit distance can be used as threshold for the stop column method in order to find such a sequence quickly.

5 Cryptanalysis of Shrinking and Alternating Step Generators

In this Section a specific implementation of the general attack presented in the previous Section for the cases of the *SG* and the *ASG* is considered.

One of the first questions that have to be taken into account in both cases is the limitation on the number of consecutive deletions because the longest run of consecutive deletions in X to get Y is always shorter than the length L_S of the selector register S. This restriction implies that the equation (1) corresponding to the computation of the edit distance matrix should be modified in the following way:

$$\begin{gathered} w_{i,1} = P_i(x_{i+1}, y_1),\ i = 0,\ldots,L_S \\ w_{0,j} = w_{0,j-1} + P_0(x_j, y_j),\ j = 2,\ldots,M \\ w_{i,1} = \infty,\ i = L_S+1,\ldots,N-M \\ w_{i,j} = \min_{k=0,\ldots,L_S-1}\{w_{i-k,j-1} + P_k(x_{i+j}, y_j)\},\ i = 1,\ldots,N-M,\ j = 2,\ldots,M \\ P_k(x_{i+j}, y_j) = \begin{cases} k & \text{if } x_{i+j} = y_j \\ k+1 & \text{if } x_{i+j} \neq y_j \end{cases} k = 0,...,L_S-1 \end{gathered} \tag{5}$$

Equations (2) and (3) corresponding to the definition of the pattern in the first and the second model, respectively must be also adapted to the *SG* and the *ASG*, producing the Equations (6) and (7) respectively:

$$\begin{aligned} &\text{If } \forall j : 2,3,\ldots,N-M+1; y_{1+j/L_S} = \cdots = y_{j-1} = y_j \text{ then } y_j = x_{L_S(j/L_S)} = x_{L_S(j/L_S)+1} = \\ &\quad \cdots = x_{L_S(j/L_S)+L_S-1} \\ &\text{If } \forall j : N-M+2, N-M+3,\ldots,M; y_{M-N+j} = \cdots = y_{j-1} = y_j \text{ then } y_j = x_j = x_{j+1} = \\ &\quad \cdots = x_{j+L_S-1} \\ &\text{If } \forall j : M+1, M+2,\ldots,N-1;\ y_{M-N+j} = \cdots = y_{M-(N-j)/L_S} \text{ then } y_M = x_j = x_{j+1} = \\ &\quad \cdots = x_{\min(j+L_S-1,N)} \end{aligned} \tag{6}$$

$$\begin{aligned}
&\text{If } \forall j : 2,3,\ldots,N-M+1; x_{L_S(j/L_S)} = x_{L_S(j/L_S)+1} = \cdots = x_{L_S(j/L_S)+L_S-1} \text{ then } x_{L_S(j/L_S)} = \\
&\quad y_{1+j/L_S} = \cdots = y_{j-1} = y_j \\
&\text{If } \forall j : N-M+2,N-M+3,\ldots,M; x_j = x_{j+1} = \cdots = x_{j+L_S-1} \text{ then } \; x_j = y_{M-N+j} = \\
&\quad \cdots = y_{j-1} = y_j \\
&\text{If } \forall j : M+1,M+2,\ldots,N-1; \; x_j = x_{j+1} = \cdots = x_{\min(j+L_S-1,N)} \text{ then } x_j = y_{M-N+j} = \\
&\quad \cdots = y_{M-(N-j)/L_S}
\end{aligned} \tag{7}$$

Finally, the process of hypothesis relaxation explained in the last Section must also be used for the cases of *SG* and *ASG* when the minimum obtained edit distance is greater than $N-M$ since it corresponds to the presence of noise.

The following toy example is used simply to show some of the most remarkable aspects of the proposal.

Example :

Let us assume that an attacker has intercepted the keystream sequence Y:1011110 of length M=7 produced by a *SG*, and knows the following parameters:

- $L_S = 3$ and $L_A = 7$
- $P_S(x) = 1+x+x^3$ and $P_A(x) = 1+x+x^7$

If we consider $H = 3$, after the verification of hypothesis from Equation (2) on the intercepted sequence we find that none of them is fulfilled, so we discard the first bit and check the hypothesis on the resulting sequence. In the sequence of length 6 only two hypotheses are fulfilled, so again we discard the first bit and consider the resulting sequence as new intercepted sequence Y:11110 of length M=5 because it fulfills 3 hypothesis.

If we consider N=10, the $IS-pattern$ according to the first formulas in Equation (2) corresponding to the cut sets shown in Fig. 6 for this new sequence X is given by $IS-pattern$: 111111x

Given Y:11110, for each initial state fulfilling the $IS-pattern$, the edit distance matrix is computed:

- Initial state: 1111110, X:1111110101 and $W = \begin{pmatrix} 0 & 0 & 0 & - & - \\ 1 & 1 & 1 & 1 & - \\ 2 & 2 & 2 & 2 & - \\ - & 3 & 3 & 4 & 4 \\ - & 4 & 5 & 4 & 4 \\ - & - & 5 & 6 & 6 \end{pmatrix}$.

The threshold is updated by the edit distance $T = 5$

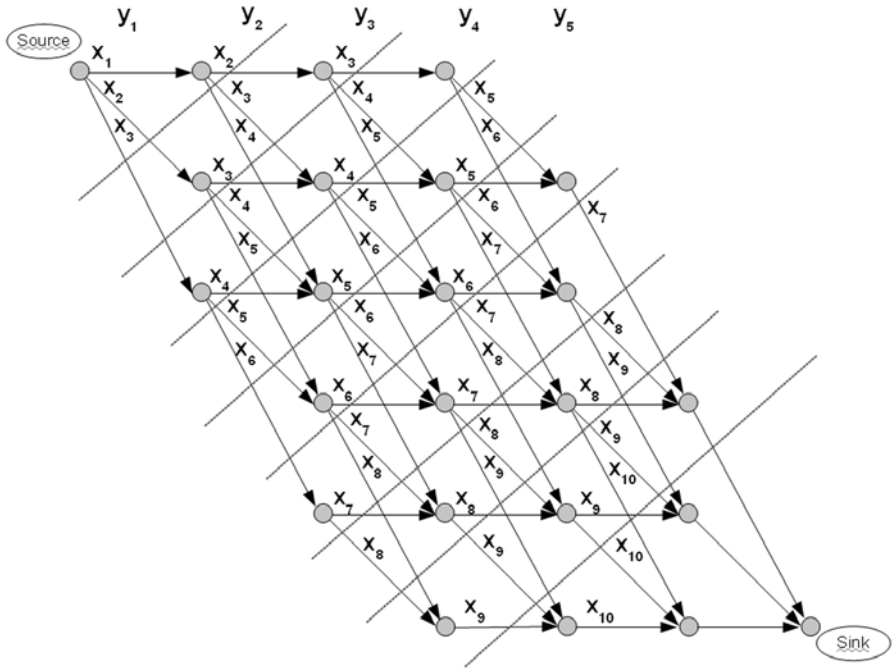

Figure 6: Cut sets

– Initial state: 1111111,X:1111111010 and $W = \begin{pmatrix} 0 & 0 & 0 & - & - \\ 1 & 1 & 1 & 1 & - \\ 2 & 2 & 2 & 2 & - \\ - & 3 & 3 & 3 & 3 \\ - & 4 & 4 & 5 & 5 \\ - & - & 6 & 5 & 5 \end{pmatrix}$.

For both initial states we get the same minimum edit distance $N - M$=5. However, when we recover the shortest paths from the source to the sink in the graphs induced by the edit distance matrices, we get the following. On the one hand, in the first case, the shortest paths correspond to non-possible decimation sequences according to the parameters of the attacked generator. On the other hand, the second initial state provides fifty-four shortest paths, and only two of them correspond to possible decimation sequences S: 0011101001 and S:1001110100 that are consistent with P_S. However, if we try to confirm these solutions with the first two discarded bits of the intercepted sequence, we get that none of them are consistent with those bits.

Consequently, according to the algorithm, the $IS - pattern$ has to be updated by complementing any of the bits in the original $IS - pattern$. For instance, consider the

new $IS-pattern$: 111110x. For this new $IS-pattern$ again there exist two initial states fulfilling the $IS-pattern$ for which minimum edit distances equal to 5 are computed. However, only the second initial state 1111101 provides a valid decimation sequence S: 0011101001 that is consistent both with the parameters of the generator and with discarded bits, so this solution is accepted as a valid solution for the cryptanalysis. On the other hand, if we update the $IS-pattern$ by complementing for instance the first bit instead of the last bit, we obtain another new $IS-pattern$: 011111x, and the resulting initial state and decimation sequence are 0111111 and S:0100111010, respectively.

In conclusion, the attack of this example has been successful alter the evaluation of only 4, out of 128, promising initial states.

6 Experimental Implementation

The next table shows some results for experimental sequential implementations of the algorithm against shrunken sequences. The columns denoted Seq.p. display the number of sequences that fulfill the $IS-pattern$. Cases marked with * indicate the existence of initial states fulfilling the $IS-pattern$ and producing sequences X that are solutions. Thres. and Dist. are the columns where the obtained threshold and the minimum edit distance are shown.

From these randomly generated examples, we may deduce a general classification of inputs into several cases. The best ones correspond to $IS-patterns$ which directly identify solutions. On the contrary, bad cases are those 'missing the event' cases in which the $IS-pattern$ fails to identify any correct initial state. Such cases are generally associated with long runs at the beginning of the sequences Y. Finally, the medium cases are those for which, despite the non existence of solutions fulfilling the pattern, a good threshold is obtained. Such cases allow a good percentage of saving in computing thanks to the detection of many early *stop columns*.

From the obtained results and the relationship between L_A and Seq.pat. we may deduce that the proposed algorithm produces the solution in $O(2^{L_A/2})$ time instead of the $O(2^{L_A})$ time corresponding to the exhaustive search that implied the original attack [Pet 2004]. Furthermore, it is clear that the worst outputs appear when the initial results in steps 1 to 4 are not adequate as there are no initial states fulfilling the $IS-pattern$. However, even in these cases that require more computation, it is guaranteed that the solution is always obtained.

Note that as aforementioned, the proposed algorithm not always output the minimum edit distance sequence (cases that are here denoted by an *) but however, since the hypothesis on Y are independent, the groups of bits in the $IS-pattern$ are also independent and consequently, the conditions might be considered separately in such way that we might define in this way a relaxed $IS-pattern$ which might lead to sequences that fulfill them. In addition, empirical results have shown that intercepted sequences Y with short runs at the beginning cause a greater improvement in the time complexity of

(N,M)	L_A	2^{L_A}	Seq.p.	Thres.	Dist.
(20,15)	7	128	0	-	5
(30,20)	9	512	1	11	10
(33,22)	7	128	2*	12	12
(75,50)	7	128	8	29	27
(150,100)	9	512	32	57	55
(300,200)	11	248	128	166	164
(300,200)	13	8192	128	115	114
(450,300)	13	8192	128	176	171
(450,300)	14	16384	1024	173	171
(450,300)	16	65536	256	173	171
(750,500)	14	16384	1024*	291	291

Table 1: Results of experimental implementations

the attack. Thus, another way to avoid a bad behaviour of the original algorithm is by choosing sub-sequences from the intercepted sequence Y that have no too long runs at the beginning, and by applying the algorithm to each one of these sub-sequences.

7 Conclusions

A new graph-based approach to a known-plaintext cryptanalysis of *LFSR*-based stream ciphers has been proposed in this paper. In particular, an improvement of the edit distance attack on certain irregularly clocked Linear *LFSR* has been described.

Our divide-and-conquer proposal has a lower computational complexity than the original attack because it does not require the exhaustive search over all the possible initial states of the target *LFSR*. The proposed heuristic optimization is based on the characterization of certain directed graphs where optimal paths provide cryptanalytic results. Also the definition of cut sets on such graphs allows getting useful thresholds to identify the most promising branches of the search graph.

The strongest aspects of our proposal are the facts that the obtained results from the attack are completely deterministic, and that many inconsistent initial states of the target *LFSRs* are recognized and avoided during search. The proposed idea of using cut sets to improve edit distance attacks might be also used against generalized clock-controlled *LFSR*-based generators.

Acknowledgements

This work was supported by the Spanish Ministry of Science and Innovation and European FEDER Fund under Project TIN2008-02236/TSI as well as by CDTI (Spain)

and the companies INDRA, Unin Fenosa, Tecnobit, Visual Tool, Brainstorm, SAC and Technosafe under Project Cenit-HESPERIA.

References

[Cab 2005] Caballero-Gil, P., Fúster-Sabater, A.: "Improvement of the Edit Distance Attack to Clock-Controlled LFSR-Based Stream Ciphers"; Lect. Notes Comp. Sci. 3643, Springer, Berlin (2005) 355-364.

[Cop 1994] Coppersmith, D., Krawczyk, H., Mansour, H.: "The Shrinking Generator"; Lect. Notes Comp. Sci. 773, Springer, Berlin (1994) 22-39.

[Ekd 2003] Ekdahl, P., Meier, W., Johansson, T.: "Predicting the Shrinking Generator with Fixed Connections"; Lect. Notes Comp. Sci. 2656, Springer, Berlin (2003) 330-344.

[Eng 2006] Englund, H., Johansson T.: "Three ways to mount distinguishing attacks on irregularly clocked stream ciphers"; International Journal of Security and Networks 1 Is. 1/2 (Sep 2006).

[Gol 1998] Golic, J.D.: "Recent Advances in Stream Cipher Cryptanalysis"; Publication de l'Institut Mathematique 64 Is. 78 (1998) 183-204.

[Gol 1991] Golic, J.D., Mihaljevic, M.J.: "A Generalized Correlation Attack on a Class of Stream Ciphers Based on the Levenshtein Distance"; Journal of Cryptology 3 No. 3 (1991) 201-212.

[Gol 1989] Gollmann, D., Chambers, W.C.: "Clock-Controlled Shift Registers: A Review"; IEEE Transactions on Selected Areas in Communications SAC-7 (May 1989) 525-533.

[Gom 2008] Gomulkiewicz, M., Kutylowski, M., Wlaz, P.: "Random Fault Attack against Shrinking Generator"; Lect. Notes Comp. Sci. 5389, Springer, Berlin (2008) 87-99.

[G*ü*n 1988] G*ü*nther, C.G.: "Alternating Step Generators Controlled by De Bruijn Sequences"; Lect. Notes Comp. Sci. 304, Springer, Berlin (1988) 5-14.

[Jia 2002] Jiang, S., Gong, G.: "On Edit Distance Attack to Alternating Step Generator"; Technical Report Corr2002-28 (2002) University of Waterloo.

[Jun 2006] Juntao, G., Xuelian, L., Yupu, H.: "Fault attack on the balanced shrinking generator"; Wuhan University Journal of Natural Sciences 11 N. 6 (Nov 2006) 1773-1776

[Kan 2003] Kanso, A.: "Clock-Controlled Shrinking Generator of Feedback Shift Registers"; Lect. Notes Comp. Sci. 2727, Springer, Berlin (2003) 443-451.

[Kha 2007] Khazaei, S., Fischer, S., Meier, W.: "Reduced Complexity Attacks on the Alternating Step Generator"; Lect. Notes Comp. Sci. 4876, Springer, Berlin (2007) 1-16.

[Mol 2004] Molland, H., Helleseth, T.: "An Improved Correlation Attack Against Irregular Clocked and Filtered Keystream Generators"; ; Lect. Notes Comp. Sci. 3152, Springer, Berlin (2004) 373-389.

[Pet 2004] Petrovic, S., Fúster, A.: "Clock Control Sequence Reconstruction in the Ciphertext Only Attack Scenario"; Lect. Notes Comp. Sci. 3269, Springer, Berlin (2004) 427-439.

[Zen 2004] Zenner, E.: "On Cryptographic Properties of LFSR-based Pseudorandom Generators", PhD Thesis, University of Mannheim, Germany (2004).