

# A KNOWN-PLAINTEXT ATTACK ON THE NONLINEAR - FEEDFORWARD GENERATOR

A. M. Youssef                      [youssefa@yahoo.com](mailto:youssefa@yahoo.com)  
M. A. Azer                         [marazer@yahoo.com](mailto:marazer@yahoo.com)  
M.S. El Soudani                 [melsoudani@menanet.net](mailto:melsoudani@menanet.net)  
*Faculty of Engineering, Cairo University, Egypt.*

**Abstract:**                      The nonlinear feedforward generator is commonly used to generate stream ciphers. In this paper, we describe a technique for cryptanalyzing this generator using a modification of the method described by Bedi and Pially. Given  $2L$  bits of the plaintext, where  $L$  is the length of the linear feedback shift register, our attack enables us to determine the initial state of the linear feedback shift register. This attack could be applied on a system having a linear feedback shift register of length 130.

**Keywords:**                      Cryptanalysis, Nonlinear Feedforward Generator, Attacks, Stream ciphers.

## 1. INTRODUCTION

Stream ciphers represent an important class of encryption algorithms. In stream ciphers the characters of a plain message are encrypted one at a time using an encryption transformation. This transformation is variable with time. Stream ciphers are suitable when buffering is limited or when characters must be individually processed as they are received. Because they have limited or no error propagation, stream ciphers may also be advantageous in situations where transmission errors are highly probable.

Feedback shift registers are considered as one of the basic keystream generators building blocks. The linear feedback shift registers LFSRs are

widely used in the keystream generation as they can produce sequences of large period with good statistical properties. Although the LFSR has the advantage of being simple, it is not secure. This is because one can use the linear properties of the LFSR to guess its initial state. To destroy the linearity of the stream cipher nonlinear Boolean functions are used. For a Boolean function to be used in stream cipher applications, the following requirements stated in [1] must be fulfilled:

- (1) Balance.
- (2) Good correlation immunity.
- (3) High nonlinearity.
- (4) High algebraic degree.
- (5) Simple hardware implementation.

There are three LFSR based stream ciphers generators : The nonlinear combining generator NLCG, the non linear feedforward generator NLFF and the clock control generator. We will focus our attention on the NLFF. In the NLFF generator, a single LFSR is used for sequence generation, but its linearity is destroyed using a nonlinear filtering function  $f$ . The structure of the NLFF is shown in Figure 1.1.

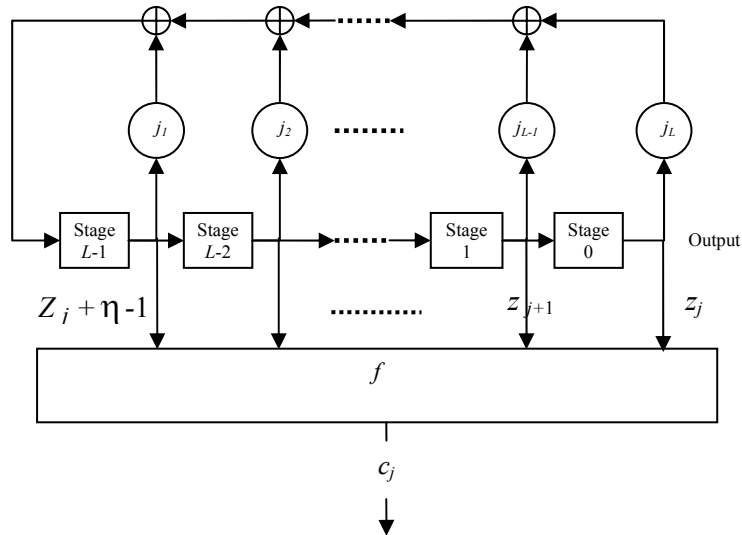


Figure 1.1. The Nonlinear Feed forward Generator

In the previous categories of generators the clock controlling all LFSRs was the same. In the clock controlled generators, the output of one LFSR is used to control another LFSR. This irregular clocking has the target of introducing a high nonlinearity in the generated keystream. The shrinking

generator [2] is an example for such a type of generators. The generators presented in this section were LFSR based. For ease of software implementation, other types of generators that replaced the LFSR with other sequence generators were proposed. From these types, we mention the SEAL (Software-optimized Encryption Algorithm) [2] and the RC4 [3].

When producing a ciphertext, the plaintext is bitwise xored with a pseudo random sequence called keystream. According to [4], for a cipher to be unbreakable, this pseudorandom sequence must be truly random. In some situations, only a part of the ciphertext is available. To decrypt all messages, the cryptanalyst must find the secret key. The main idea of the correlation attack is as follows. Assume that we have observed a keystream sequence of length  $N$ . It is generated from a generator with  $\lambda$  different LFSRs. If the cryptanalyst can find a correlation between the keystream and the output of one of the  $\lambda$  LFSRs, then he/she can find the initial state of this LFSR. This is done assuming that the feedback polynomial of the target LFSR is known. The original correlation attack was first introduced by Siegenthaler in [5]. It was described as a ciphertext-only attack.

To reduce the computational complexity of the exhaustive search in correlation attacks, we use the fast correlation attacks. Meier and Staffelbach [6] made use of some parity check equations created from the feedback polynomial of the LFSR and developed two algorithms ( $A$  and  $B$ ) for fast correlation attacks.

To improve the fast correlation attacks presented by Meier and Staffelbach [7]. The methods used for the improvement were classified into two classes. The methods of the first class were used to increase the number of obtained parity check equations and the methods of the second class try to improve the fast correlation attacks by using more powerful iterative decoding techniques.

For finding parity check equations, Mihaljevic and Golic [8] suggested a method based on the matrix representation of the LFSR. Other methods were presented in [9, 10]. It has also been proposed to use other iterative decoding methods. These iterative methods make use of the obtained low weight parity check equations. An overview of some of these iterative methods is given in [11, 12]. A more detailed overview for cryptanalysis of stream ciphers is given in [13].

The correlation attacks described above fail if the output of the LFSR is not correlated to the output sequence. Siegenthaler [14] formally proposed the concept of correlation immunity for this condition as follows. Assume  $f(x)$  is balanced and its components are  $\eta$  binary and independent random

variables. Then a Boolean function  $f(x): F_2^n \rightarrow F_2$  is called  $D^{\text{th}}$  order correlation immune, if  $f(x)$  is statistically independent of any  $\eta$  input components. For a memoryless combining function to be  $D^{\text{th}}$  order correlation immune, a necessary condition was given by Siegenthaler [14]. The condition states that for a Boolean function to be  $D^{\text{th}}$  order correlation immune, it shouldn't have any term of degree more than  $\eta - D$ , where  $\eta$  is the number of the input variables.

In [15] Siegenthaler gives a sufficient condition for the correlation immunity of a finite state machine combiner (a combiner with internal state which will be updated with time). It states that any  $\eta$  inputs and internal state are jointly independent of the output bit.

The paper is organized as follows. In Section 2 we give some preliminaries on the decoding model that is used for cryptanalysis, and in Section 3 we shortly introduce the method used for the solution of non linear Boolean equations. In Section 4 two types of equations are derived, in section 5 the solution of the system of equations is explained. In section 6 the attack results are presented, and finally, in Section 7 we give some conclusions and possible extensions.

## 2. ATTACK DESCRIPTION

This attack is a modification of the algorithm described in [16]. The NLFF generator under consideration consists of two parts, as shown in Figure 1.1. The first part is a LFSR of length  $L$ , which generates an output sequence  $z$  depending on its connection polynomial. The second part is a NLFF function, whose input sequence is  $\eta$  bits output from the LFSR and its output sequence is a keystream sequence  $c$  of length  $2L$ . This output sequence  $c$  will then be used to recover the original settings of the LFSR. The locations from which the output of the LFSR is taken are called tap points or more simply taps.

In our attack two sets of equations are generated: linear and nonlinear equations. The method that will be used for the solution of the nonlinear set of equations is called the local reduction technique [17].

### 3. SOLUTION OF A SYSTEM OF NONLINEAR BOOLEAN EQUATIONS

Only exponential time algorithms are available for solving an arbitrary set of Boolean equations. In our case we shall use the local reduction technique developed by Zakrevkij and Vasilkova [17]. This technique can solve large number of equations provided that the following two conditions are satisfied:

- (1) There is a relatively small number of variables in each equation.
- (2) There is a large overlap in the set of variables.

There are two main stages in this technique, the reduction of the given system of equations described in section 3.1 and the removal of common factors is described in section 3.2.

#### 3.1. Local Reduction Technique

The local reduction technique consists of two steps. The first is the local reduction, and the second is the reduction of the system of equations.

##### *Onset Representation of a Boolean Equation*

The onset representation of a Boolean equation contains all the possible solutions of that equation. For example, given a Boolean equation in the form  $x.y + z = 1$ , where the “.” sign denotes the “AND” function and the “+” sign denotes the “OR” function, the solution set will be  $\{x, y, z : 001, 011, 101, 110, 111\}$ . The solution set of a Boolean equation presented in this form is called onset. The onset representation is the first step in the solution of a system of equations.

##### *Reduction of the System of Equations*

If we have a system of equations, the solution of each equation should be obtained first and written in the onset representation. Now after solving each equation independently in a set of equations we have to find the common solution for all equations. For example if we have the following group of onsets, each representing the solution of an equation in a system of equations, with non empty intersection of variables.

$$\begin{array}{lll} f(h, i, j), & \text{onset}_3 = \{100, 010\} & 1 \\ f(i, j, k), & \text{onset}_4 = \{110, 001\} & 2 \end{array}$$

To solve this system of equations we start by finding the common variables between each two equations. From Equation 1 and Equation 2, the common variables are  $(i, j)$ , and  $s_1=\{00,10\}$   $s_2=\{11, 00\}$ , where  $s_1, s_2$  are called the projection of the onsets of Equation 1, Equation 2 on  $(i, j)$ . It follows that  $s_{1,2}=s_1 \cap s_2=\{00\}$ , where “ $\cap$ ” denotes the intersection between two sets. After discarding the non valid solutions in all equations, the new solution set becomes

$$f(h, i, j, k) \quad \text{solution onset}=\{1001\}.$$

The complexity of the local reduction increases significantly with the increase of the number of variables per equation, hence we need a small number of variables in each equation.

### 3.2. The Removal of Common Factors

Sometimes we have a repeated solution for some variables in the onset solution, for example:

$$f(x, y, z, w, h) \quad \text{onset}=\{11001, 11111, 11010\} \quad 1$$

It can be noticed that the values of the variables  $x$  and  $y$  are always the same in all the subsets of the onset, as the variables  $x$  and  $y$  are always equal to one in the onset of Equation 3. Hence, we can deduce that those values are valid for the whole system of equations. Therefore, it is preferable to substitute those two variables by their values as if we were factoring out  $x$  and  $y$  from all equations. This will reduce the number of variables, and hence the complexity of each equation.

## 4. THE DERIVATION OF THE EQUATIONS

From the Block diagram of the NLFF generator, shown in Figure 1.1, we can deduce that there are two different types of equations in this systems:

- Linear equations
- Nonlinear equations

### The Linear Equations

These equations describe the relation between the output sequences of the LFSR. The linear equations are derived using the connection polynomial of the LFSR. If the connection polynomial has the following form

$$g(x) = 1 + j_1x + j_2x^2 + j_3x^3 + \dots + j_Lx^L, \quad 2$$

where  $L$  is the LFSR length

Then, the linear equations which relate the generated sequence  $z_j$  with the previously generated sequence have the form

$$z_j = j_1z_{j-1} \oplus j_2z_{j-2} \oplus j_3z_{j-3} \oplus \dots \oplus j_Lz_{j-L}, \quad j \geq L \quad 3$$

As seen from Equation 5, the LFSR initial settings  $z_0, z_1, z_2, \dots, z_{L-1}$  must be available to generate the rest of the  $z$  sequence.

### The Non-Linear Equations

The nonlinear Boolean function uses the LFSR sequence  $z$  to generate the keysequence  $c$ . The nonlinear Boolean function takes  $\eta$  inputs from the LFSR tap points, as shown in Figure 1.1. If we assume, without loss of generality, that the nonlinear function input is taken from the last  $\eta$  taps of the LFSR, we obtain the following equations:

$$\begin{aligned} c_0 &= f(z_S, z_{S+1}, \dots, z_{S+\eta-1}) \\ c_1 &= f(z_{S+1}, z_{S+1+1}, \dots, z_{S+1+\eta-1}) \\ &\vdots \\ &\vdots \\ c_{2L-1} &= f(z_{S+2L-1}, z_{S+2L-1+1}, \dots, z_{S+2L-1+\eta-1}) \end{aligned} \quad 4$$

or more generally

$$c_j = f(z_{S+j}, z_{S+j+1}, \dots, z_{S+j+\eta-1}) \quad 5$$

where  $V$  is the first tap point from which the input is taken to the nonlinear function and is equal to  $L-\eta$ ,  $z$  is the LFSR output sequence, and  $c$  is the sequence of length  $2L$  generated from the feedforward generator and is function in  $\eta$  LFSR output sequences.

Now, if the nonlinear function input is taken from any tap points of the LFSR, not necessarily the last, and if these tap points are distant from  $V$  by a position  $i$ , the output will be as follows:

$$\begin{aligned}
c_0 &= f(z_S, z_{S+1}, \dots, z_{S+\eta-1}) \\
c_1 &= f(z_{S+1}, z_{S+1+1}, \dots, z_{S+1+\eta-1}) \\
&\vdots \\
c_{2L-1} &= f(z_{S+2L-1}, z_{S+2L-1+1}, \dots, z_{S+2L-1+\eta-1})
\end{aligned} \tag{6}$$

A more general form is given as follows:

$$c_j = f(z_{S+j}, z_{S+j+1}, \dots, z_{S+j+\eta-1}) \tag{7}$$

where  $S=L-\eta-i$ ,  $0 \leq j \leq 2L-1$ ,  $0 \leq i \leq V < L$  and  $c$  is the sequence of length  $2L$  generated from the feedforward generator and is function in  $\eta$  LFSR output sequences.

It can be deduced from Equation 9 that the maximum length of the sequence  $z$  needed to generate the keystream  $c$  is  $3L-i-1$ . This comes from the fact that the last needed  $z$  is the one used for generating  $c_{2L-1}$ , which is  $z_{S+2L-1+\eta-1}$  i.e.,  $z_{S+2L-1+\eta-1}=z_{3L-i-2}$ . Given that the sequence  $z$  starts at  $z_0$ , then the total length of the generated sequence  $z$  is  $3L-i-1$ .

## 5. SOLUTION OF THE SYSTEM OF EQUATIONS

The set of equations derived in the previous section can be solved simultaneously using the following steps:

- (1) Find all possible solutions for the nonlinear function for  $f=0$  and for  $f=1$ , in the form of onsets.
- (2) For each nonlinear equation write the onset solution for all variables.
- (3) Arrange the onset solutions in their proper location in a table, and the absent variables of each equation are considered as don't cares "x".
- (4) This is the solution step. To find the solution of the nonlinear set of equations, we follow a tree search together with the local reduction technique. The main idea of the local reduction technique is that when two onsets have common variables, the values of the common variables are compared. If no contradiction is found, then a new onset is formed from the two onsets. In our case, since we are solving the two different types of equations together, we need to test at early stages that the found solutions verify the nonlinear set of equations as well as

the linear set of equations. Therefore, when the difference between the first and the last index of the sequence  $z$  is greater than or equal to the LFSR length  $L$ , we substitute in the LFSR equation to verify that the obtained solutions satisfy both types of equations. If no contradiction occurs we continue through the same tree path. If not, a trace back is needed to get new valid solutions.

- (5) In this step all values ranged from  $z_{L-\eta-i}$  till  $z_{3L-2-i}$ , where  $i$  is the shift from the last  $\eta$  taps, are available. The values of  $z_{L-i-1}$  till  $z_0$ , which are considered as the LFSR initial state are still unknown. These values can be deduced from the LFSR linear equations using a backward substitution in the form

$$z_{\psi-L} = j_0 \bullet z_{\psi} \oplus j_1 \bullet z_{\psi-1} \oplus j_2 \bullet z_{\psi-2} \oplus \dots \oplus j_{L-1} \bullet z_{\psi-(L-1)} \quad 8$$

or more generally

$$z_{\psi-L} = \bigoplus_{y=0}^{y=L-1} j_y \bullet z_{\psi-y} \quad 9$$

where  $L \leq \psi < 2L$  and  $j_y$  are the LFSR connection polynomial coefficients.

### Example

The LFSR shown in Figure 5.1 has a length  $L=5$ , a connection polynomial  $g(x) = 1 \oplus x^3 \oplus x^5$ , and a known initial state  $z_0, z_1, z_2, z_3, z_4 = \{1, 0, 1, 0, 1\}$ . The input of the nonlinear function is taken from the last two taps, i.e.  $\eta = 2$ , and the nonlinear function's output is calculated as follows:

$$c_j = z_{j+3} \oplus z_{j+4} \oplus z_{j+3} \cdot z_{j+4} \quad 10$$

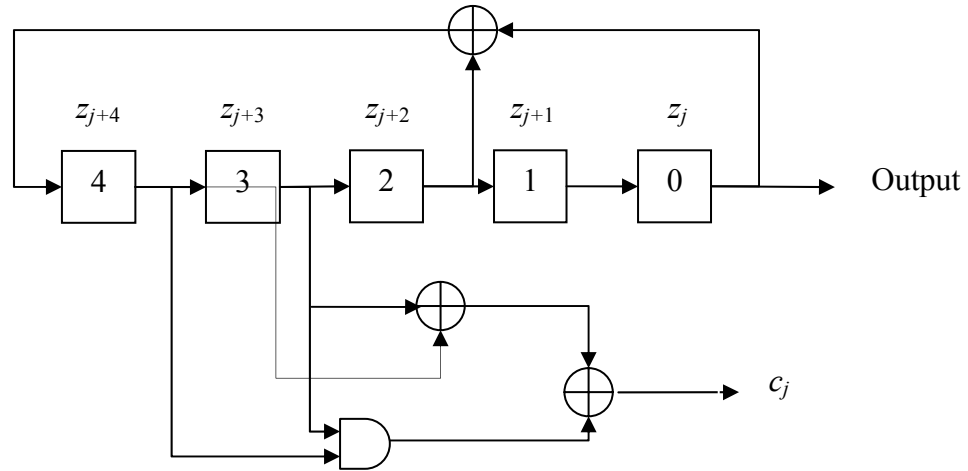


Figure 5.1. The nonlinear feedforward generator of the example.

According to Equation 5, the LFSR output sequence  $z$  is equal to:  
 $z_0, z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8, z_9, z_{10}, z_{11}, z_{12}, z_{13} = \{1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0\}$ .  
 From the obtained sequence  $z$  it is now possible, using the given nonlinear function's equation, to calculate the values of  $c_j$ , which are found to be:  $c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9 = \{1, 1, 0, 0, 0, 1, 1, 0, 1, 1\}$ . The onset solution for  $c=1$  is  $\{10, 01, 11\}$ . The onset solution for  $c=0$  is  $\{00\}$ . After Deriving the non linear equations using Equation 8 and Equation 12, and mapping the solutions in a table as shown in Table 1, we start solving the two types of equations as it was described in the previous section. The solution steps are illustrated pictorially in Figure 5.2. To derive the initial state of the LFSR, we use the solution set together with a backward substitution in Equation 12.

Table 1. The onset solution of the nonlinear equations.

$j$	$z_0$	$z_1$	$z_2$	$z_3$	$z_4$	$z_5$	$z_6$	$z_7$	$z_8$	$z_9$	$z_{10}$	$z_{11}$	$z_{12}$	$z_{13}$
Sol <sub>0</sub>	X	X	X	1	0	X	X	X	X	X	X	X	X	X
	X	X	X	0	1	X	X	X	X	X	X	X	X	X
	X	X	X	1	1	X	X	X	X	X	X	X	X	X
Sol <sub>1</sub>	X	X	X	X	1	0	X	X	X	X	X	X	X	X
	X	X	X	X	0	1	X	X	X	X	X	X	X	X
	X	X	X	X	1	1	X	X	X	X	X	X	X	X
Sol <sub>2</sub>	X	X	X	X	X	0	0	X	X	X	X	X	X	X
Sol <sub>3</sub>	X	X	X	X	X	X	0	0	X	X	X	X	X	X
Sol <sub>4</sub>	X	X	X	X	X	X	X	0	0	X	X	X	X	X
Sol <sub>5</sub>	X	X	X	X	X	X	X	X	1	0	X	X	X	X
	X	X	X	X	X	X	X	X	0	1	X	X	X	X
	X	X	X	X	X	X	X	X	1	1	X	X	X	X
Sol <sub>6</sub>	X	X	X	X	X	X	X	X	X	1	0	X	X	X
	X	X	X	X	X	X	X	X	X	0	1	X	X	X
	X	X	X	X	X	X	X	X	X	1	1	X	X	X
Sol <sub>7</sub>	X	X	X	X	X	X	X	X	X	X	0	0	X	X
Sol <sub>8</sub>	X	X	X	X	X	X	X	X	X	X	X	1	0	X
	X	X	X	X	X	X	X	X	X	X	X	0	1	X
	X	X	X	X	X	X	X	X	X	X	X	1	1	X
Sol <sub>9</sub>	X	X	X	X	X	X	X	X	X	X	X	X	1	0
	X	X	X	X	X	X	X	X	X	X	X	X	0	1
	X	X	X	X	X	X	X	X	X	X	X	X	1	1

where Sol<sub>j</sub> is the solution of the  $j^{th}$  nonlinear equations.

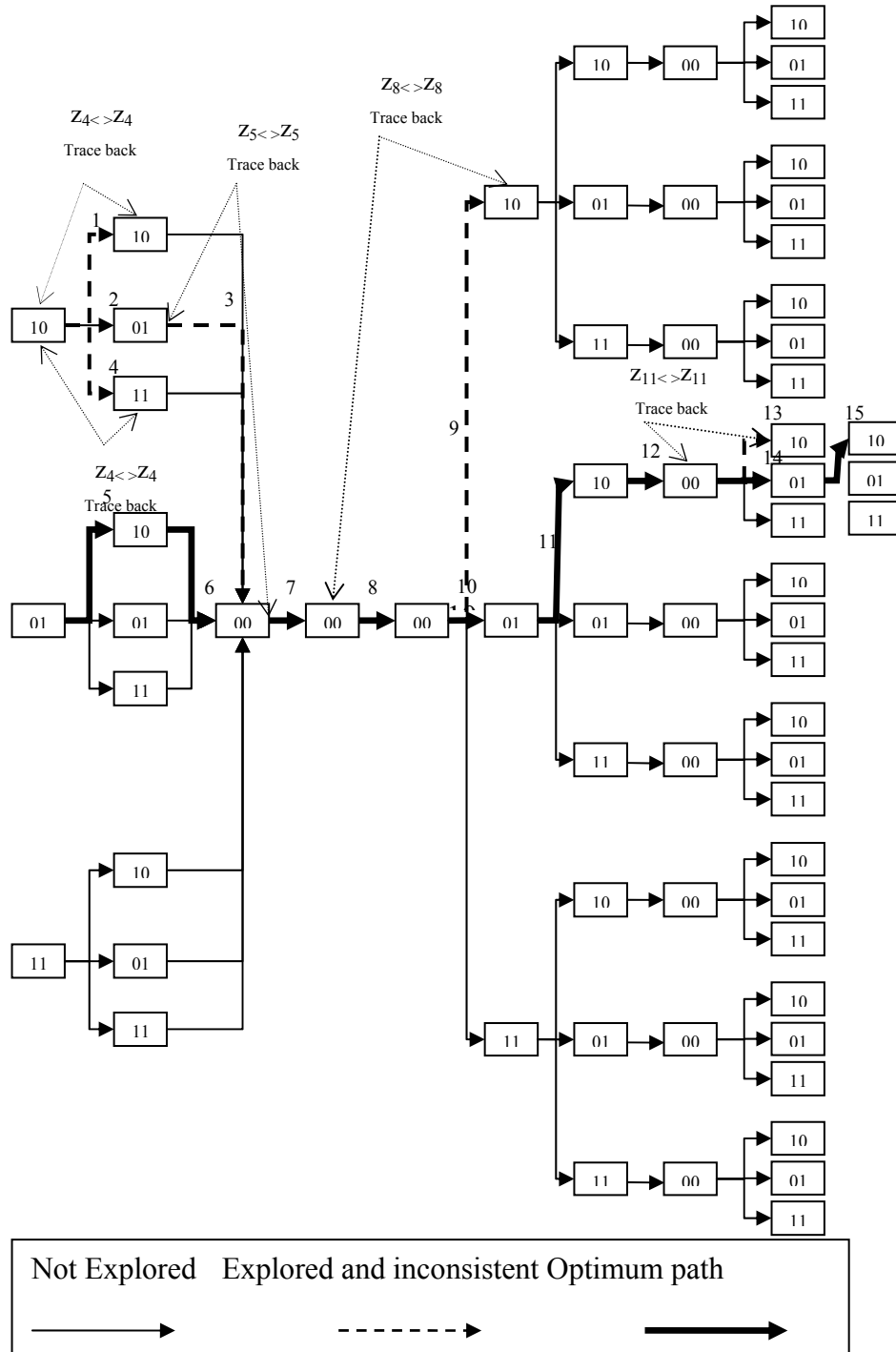


Figure.5.2. A pictorial representation of the used tree search method

## 6. SIMULATION RESULTS

This attack was applied on several LFSRs having different lengths using nonlinear functions with different degrees. Simulation results are presented in Table 2. It is clear that the time required for the attack increases significantly with the increase of the number of inputs,  $\eta$ , of the nonlinear function for the same LFSR length. Whereas when the LFSR length increases, keeping the same value of  $\eta$ , there is no large increase in the attack time. Hence, the complexity increases more significantly with the increase of the number of inputs to the nonlinear function. For example if we consider the LFSR whose length  $N=128$ , we find that the simulation time varies according to the chosen  $\eta$ . When  $\eta$  exceeded 11, the simulation time increased significantly, whereas for  $\eta=10$  there was no large increase in time when the LFSR length increased from 128 to 130.

Table 2. The simulation results of the cryptanalysis of the NLFF.

$N$	$\eta$	$G(X)$	Simulation Time
64	8	$1+x+x^3+x^4+x^{64}$	2 sec.
71	8	$1+x^6+x^{71}$	4 sec.
128	8	$1+x+x^2+x^7+x^{128}$	2 sec.
128	10	$1+x+x^2+x^7+x^{128}$	3 sec.
128	11	$1+x+x^2+x^7+x^{128}$	7 sec
128	12	$1+x+x^2+x^7+x^{128}$	253 sec.
128	13	$1+x+x^2+x^7+x^{128}$	1092 sec.
130	10	$1+x^3+x^{130}$	37 sec.

## 7. SUMMARY AND CONCLUSIONS

In this paper we described a method for the cryptanalysis of the NLFF generator. In our attack we have assumed that the connection polynomial and  $2L$  bits of the plaintext were available. This attack was based on two types of equations, the LFSR linear equations and the nonlinear function equations. These equations were derived and solved simultaneously using the local reduction technique together with a tree search method. A pictorial representation for the tree search method was presented. The tree search method was found to be very efficient in the solution of nonlinear equations and economic memory wise as only the explored path is stored. The used tree search method was a depth first. Other tree search algorithms such as the best first or the breadth first might minimize the attack time. The LFSR initial

states were found by a back substitution in the LFSR linear equations. Experimental results have shown that the complexity of this method increases significantly with the increase of the number of inputs to the nonlinear Boolean function for the same LFSR length. On the other hand, the increase of the LFSR length did not cause a large increase in the simulation time provided that the number of inputs to the nonlinear Boolean function was kept constant. This attack is found to be valid if the input of the nonlinear function is taken from any  $\eta$  adjacent tap points of the LFSR.

## 8. REFERENCES

- [1] Tarannikov, Y.(1997) "On Resilient Boolean Functions with Maximum Possible Nonlinearity." Progress in Cryptology-Indocrypt'2000, LNCS Vol., Springer-Verlag 2000, pp. 19-30.
- [2] Menezes, A, Van Oorschot, P., and Vanstone, S.(1997), "Handbook of Applied Cryptography." CRC Press, Boca Raton.
- [3] Schneier, B.(1996) "Applied Cryptography: Protocols, Algorithms, and Source Code in C." John Wiley and Sons, 2nd Edition.
- [4] Beth, T., Dai, Z., "On the Complexity of Pseudo-Random Sequences – Or: If You Can Describe a Sequence It Can't Be Random ," Advances in Cryptology-Eurocrypt'89, LNCS Vol. 434, Springer-Verlag, pp. 533-543.
- [5] siegenthaler, T.(1990), "Decrypting a Class of Stream Ciphers Using Ciphertext Only." IEEE Transactions on Computers, Vol. 34, pp. 81-85.
- [6] sandarajan, D.(2001), "The Discrete Fourier Transform: Theory Algorithms and Applications." World Scientific Pub. Co.
- [7] Meier, W., Staffelbach, O.(1989), "Fast Correlation Attacks on Certain Stream Ciphers." Journal of Cryptology, Vol. 1, pp. 159-176.
- [8] Mihaljevic, M., Golic, J.(1990), "A Fast Iterative Algorithm for a Shift Register Initial State Reconstruction Given the Noisy Output Sequence." Advances In Cryptology-Auscrypt 90, LNCS Vol. 453, Springer-Verlag.
- [9] Chepyzhov, V., Smeets, B.(1991), "On Fast Correlation Attacks on Certain Stream Ciphers." Advances In Cryptology-Eurocrypt, LNCS Vol. 547, Springer-Verlag, pp. 176-185.

- [10] Penzhorn, W. ( 1996), "Correlation Attacks on Certain Stream Ciphers: Computing Low Weight Parity Checks Based on Error Correcting Codes." Fast Software Encryption'96, LNCS Vol. 1039, Springer-Verlag, pp. 159-172.
- [11] Clark , A., Golic, J., and Dawson, E.(1991), "A Comparison of Fast Correlation Attacks." Fast Software Encryption'96, LNCS Vol. 1039, Springer-Verlag.
- [12] Fossierier, M., Mihaljevic, M., and Imal, H.(1999), "Critical Noise for Convergence of Iterative Probabilistic Decoding with Belief Propagation in Cryptographic Applications." Applied Algebra , algebraic Algorithms and Error Correcting Code-aec 13,. LNCS Vol. 1719, Springer-Verlag, pp. 282-293.
- [13] Jiang, S., Gong, G., "Cryptanalysis of Stream Ciphers-A Survey." CORR, 2002-29,.www.cacr.math.uwaterloo.ca/technical reports.
- [14] Siegenthaler, T.(September 1985), "Correlation Immunity of Nonlinear Combining Functions for Cryptographic Applications." IEEE Transactions on Information Theory, Vol. IT-30, No. 5.
- [15] Siegenthaler, T.(1986), "Design of Combiners to Prevent Divide and Conquer Attacks." Advances in Cryptology-CRYPTO'85, H. C. Williams, Editor, LNCS 218, Springer-Verlag, pp. 273-279.
- [16] Bedi, S., Pillai, N.(2001), "Cryptanalysis of the Nonlinear Feed Forward Generator", indocrypt'01, LNCS Vol. 2247, pp. 188-194.
- [17] Zakrevskij, A., Vasilkova, I.( 21-22 Sep. 2000), "Reducing Large Systems of Boolean Equations." Fourth International Workshop on Boolean problems.
- [18] Zakrevskij, A., Vasilkova, I.(21-22 Sep. 2000), "Reducing Large Systems of Boolean Equations." Fourth International Workshop on Boolean problems.