Fault Analysis on the Stream Ciphers LILI-128 and Achterbahn

Dibyendu Roy and Sourav Mukhopadhyay
Department of Mathematics,
Indian Institute of Technology Kharagpur,
Kharagpur-721302, India
{dibyendu.roy,sourav}@maths.iitkgp.ernet.in

Abstract:- LILI-128 is a clock controlled stream cipher based on two LFSRs with one clock control function and one non-linear filter function. The clocking of the second LFSR is controlled by the first LFSR. In this paper we propose a fault algebraic attack on LILI-128 stream cipher. We first recover the state bits of the first LFSR by injecting a single bit fault in the first LFSR. After that we recover the second LFSR state bits by following algebraic cryptanalysis technique. We also propose fault attack on Achterbahn stream cipher, which is based on 8 NLFSRs, 8 LFSRs and one non-linear combining function. We first inject a single bit fault into the NLFSR-A then observe the normal and faulty keystream bits to recover almost all the state bits of the NLFSR-A after key initialization phase. One can apply our technique to other NLFSR-B, C, D to recover their state bits also.
Keywords:- Stream ciphers, LFSR, NLFSR, LILI-128, Achterbahn, Fault attack.

1 Introduction

Fault analysis is a cryptanalysis technique for the stream ciphers. By this cryptanalysis technique attacker injects fault into the stream cipher to recover full set or few bits of the state of the cipher. There are some methods to inject fault in the cipher like as laser shot. Due to this fault, in some clockings the cipher produces faulty keystream bits. The attacker first observes the normal and faulty keystream bits to detect the fault position in each clocking. After that the attacker will try to recover the full or some bits of the state of the cipher by observing the normal and faulty keystream bits. Recently, in many literatures [5], [8], [6], [9], [1] authors have proposed fault attack on many recent ciphers.

LILI-128 is one of the clock controlled stream cipher, based on two LFSRs, one clock control function and one non-linear filter function. In 2000, Dawson et al. [3] proposed the design specification of LILI stream cipher. This cipher is based on two LFSRs but due to effect of the clock control function the state of the second LFSR updates in a non-linear way. In each clocking the state of the $LFSR_c$ updates in usual way. Then the clock control function takes two bits as input from the state of $LFSR_c$ to generate a number $c \in \{1, 2, 3, 4\}$. After that the $LFSR_d$ clocks c times. Then the non-linear filter function takes some bits from the state of the $LFSR_d$ as input to generate the keystream bits in that clocking. The detail of the cipher is described in Section 2.

In 2004, Hoch and Shamir [5] first proposed a fault attack on the LILI-128 stream cipher. They proposed the fault attack on this stream cipher by injecting a single bit fault in data register of the cipher. After injecting the fault in the data register they observed the normal and faulty keystream bits to recover two clock control bits of the clock control register. After that they found the state bits of the data register. The main difference between their work and our work is, in our work we inject fault in clock control register to recover the state bits of clock control LFSR then we perform algebraic attack on the data register to get back the state bits of the data register.

Achterbahn [4] is one of the stream cipher proposals submitted in the eSTREAM call for stream ciphers. Achterbahn [4] is a new type of stream cipher, which is based on 8 NLFSRs and 8 LFSRs and a non-linear combining function. Before generating any keystream bit the cipher first passes through key initialization phase. The key initialization phase is described in [4]. After the key initialization phase, the cipher passes through key generation phase. In each clocking of the key generation phase each NLFSR updates by their non-linear state update function. After that each linear filter function generates a single bit output by taking some bits of the NLFSR as input. Finally, the non-linear combining function takes these bits as input to generate the keystream bit. The detail of the keystream generation phase is described in Section 4.

In 2006, Johansson et al. [7] proposed a cryptanalysis on Achterbahn stream cipher. They proposed this cryptanalysis by observing low linear complexity of the nonlinear filter function of the cipher.

Our contribution:- In this paper we propose fault attack on this two stream ciphers. We introduce a new type of fault attack on the LILI-128 stream cipher [3] by injecting fault in the clock control LFSR. By injecting a single bit fault in the clock control LFSR we first recover the state bits of the first LFSR. After that we apply algebraic attack [2] technique to recover the state bits of the second LFSR. In this paper we also propose fault attack on the Achterbahn stream cipher [4] by injecting a single bit fault in the NLFSR-A. By injecting a single bit fault we are able to recover almost all the state bits of the NLFSR-A after key initialization phase. One can apply our method to other NLFSR-B, C, D to recover their state bits after the key initialization phase. This is our first attempt to analyse Achterbahn [4] by injecting fault.

Organization of the article:- The rest of the article is organized as follows: In Section 2 we discuss about the design specification of the stream cipher LILI-128. In the Section 3 we propose fault attack on the stream cipher LILI-128. In Section 4 we discuss the design specification of the stream cipher Achterbahn. The fault attack on the stream cipher Achterbahn is discussed in Section 5. Complexity of the fault attack on Achterbahn stream cipher is calculated in Section 5.2. Finally the article is concluded in Section 6.

2 Design specification of LILI-128 stream cipher

In this section we shall discuss about the design specification of LILI-128 stream cipher [3]. This cipher is based on two LFSRs and one clock control function and a non-linear filter function. First LFSR is named as $LFSR_c$ and other one is $LFSR_d$. The clocking of the second LFSR is controlled by the first LFSR. The design specification of this cipher is given in the following figure 1. First LFSR is of 39 bits and the second LFSR is of 89 bits. The state bits of the $LFSR_c$ are denoted by x_i's and the state bits of the $LFSR_d$ are denoted by d_i's. The state update function of $LFSR_c$ is L_c and the state update function of $LFSR_d$ is L_d. The clock control function is denoted by f_c and the non-linear filter function is denoted by f_d. The primitive polynomial corresponding to $LFSR_c$ is given by,

$$x^{39} + x^{35} + x^{33} + x^{31} + x^{17} + x^{15} + x^{14} + x^2 + 1.$$

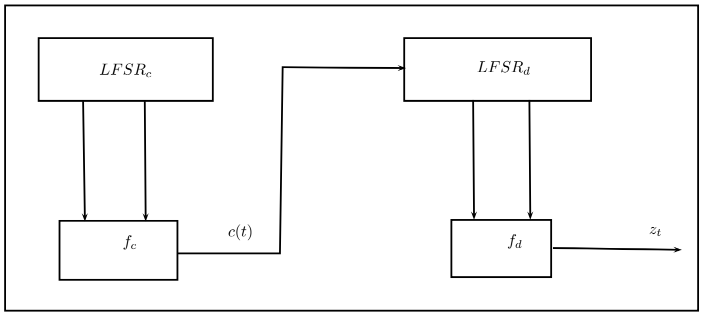

Figure 1: Design specification of LILI-128

The primitive polynomial corresponding to $LFSR_d$ is given by,

$$x^{89} + x^{83} + x^{80} + x^{55} + x^{53} + x^{42} + x^{39} + x + 1.$$

The clocking of the second LFSR is controlled by a function involving two state bits of $LFSR_c$. The expression of the clock control function is given by,

$$f_c(x_{12}, x_{20}) = 2x_{12} + x_{20} + 1 = c(t) \ \in \{1, 2, 3, 4\}.$$

The non-linear filter function (f_d) of the cipher is of degree 6. The description of the nonlinear filter function is given in [4]. In each clocking this function takes some state bits from the $LFSR_d$ as input to generate the keystream bits.

In each clocking the $LFSR_c$ clocks in normal procedure and the clock control function takes two state bits (namely x_{12}, x_{20}) from the $LFSR_c$ as input and generates a output $c(t) \in \{1, 2, 3, 4\}$. This $c(t)$ determines the number of clockings of the $LFSR_d$ in that clocking. i.e. suppose in t-th clocking the value of $c(t) = 2$ then in t-th clocking the $LFSR_d$ will clock 2 times before generating any keystream bit.

For this clock control function the state update function of the $LFSR_d$ becomes nonlinear. Let d_i^{t+1} denote the i-th bit of the $LFSR_d$ at $(t+1)$-th clocking. The algebraic expression of d_i^{t+1} will be,

$$d_i^{t+1} = (x_{12}^t + 1)(x_{20}^t + 1)d_{i-1}^t + (x_{12}^t + 1)x_{20}^t d_{i-2}^t + x_{12}^t(x_{20}^t + 1)d_{i-3}^t + x_{12}^t x_{20}^t d_{i-4}^t.$$

Where x_{12}^t, x_{20}^t are the clock control bits for the $LFSR_d$ at t-th clocking.

3 Fault analysis on LILI-128 by injecting fault in the $LFSR_c$

In this section we shall propose our new fault analysis on LILI-128 stream cipher [3], by injecting a single bit fault in the $LFSR_c$. After injecting a fault in the $LFSR_c$ we will first find the state bits of $LFSR_c$. After finding the state bits of the $LFSR_c$ we will find state bits of the $LFSR_d$ by using classical algebraic attack technique. Before injecting fault in the $LFSR_c$ we assume that the attacker has the following freedom,

- Attacker can inject fault in any arbitrary position of the $LFSR_c$
- After injecting fault the attacker can reset the cipher to the normal condition.

After injecting the fault in any arbitrary position of the $LFSR_c$, the attacker has to do the following job,

- The attacker has to detect the position of the fault
- After detecting the fault the attacker has to check the propagation of the fault in each clocking.

As the attacker is injecting fault in any arbitrary position of the $LFSR_c$ then after injecting the fault, depending upon the fault position fault may or may not affect the clock control function's output. After injecting fault to any arbitrary position the fault will start moving in each clocking, when it reaches to the position x_{12} or x_{20} the fault will affect in the clock control function's output i.e. the clock control function will give faulty output. Suppose the fault is at x_{20} then the expression of the output of the clock control function will be $\bar{y} = 2x_{12} + x'_{20} + 1$. The expression of the output of the clock control function without fault is $y = 2x_{12} + x_{20} + 1$. The difference between y and $\bar{y}$ will be $y + \bar{y} = 1$. Similarly if the fault is at x_{12} then the difference between y and $\bar{y}$ will be 2. If fault presents at both the x_{12} and x_{20} then the difference between y and $\bar{y}$ will be 3.

Now the attacker has to detect that, between y and $\bar{y}$ which one is higher. Clearly y and $\bar{y}$ are the number of clockings of the $LFSR_d$. Suppose $y > \bar{y}$ then for normal case the second $LFSR_d$ will take extra time to generate the keystream than for the faulty case. So by observing the time taken for generating the keystream bits of the $LFSR_d$ the attacker can find that which one is higher between y and $\bar{y}$.

So by observing the time difference for keystream generation for the normal and the faulty case the attacker can detect whether fault presents in the clock control bits or not. Suppose at t-th clocking the fault is at x_{20} then on that clocking the time difference will occur. After 8 clockings of the $LFSR_c$, the fault will move to x_{12} then at $(t + 8)$-th clocking the time difference will again occur. By observing this type of pattern the attacker can detect that whether fault present at x_{20} or not. Similarly, if at t-th clocking fault presents at x_{30} then the time difference of keystream generation will occur when the fault will move from x_{30} to x_{20} i.e. after 10 clockings of the $LFSR_c$ the fault will create the keystream generation's time difference. So, by observing these types of time differences the attacker can detect the fault position at certain clocking.

After performing above steps suppose the attacker detect that fault is at x_{20} and $\bar{y} > y$. That means $x_{20} = 0$. Similarly if fault presence at other position or in the feedback of the $LFSR_c$ then by observing the time difference between the generation of the normal keystream bits and the fault affected keystream bits the attacker can detect some state bits of the $LFSR_c$ or get some linear equations involving state bits of the $LFSR_c$, then the attacker solves the system to get the other bits. By following these procedures the attacker can recover the state bits of the $LFSR_c$.

After getting the state bits of the $LFSR_c$ the attacker will find the state bits of the $LFSR_d$. As all the bits of the $LFSR_c$ are known now, that implies the value of the output of the clock control function in each clocking is known now. The non-linear filter function f_d of the cipher is of degree 6. Courtois and Meier [2] have found a low degree multiple g_d of the function f_d such that $f_d g_d$ will be of degree 4. Then by using classical algebraic attack [2] technique we can find the state bits of the $LFSR_d$ with the 2^{57} CPU clocks.

We are not injecting fault in any fixed position. We are injecting fault in any arbitrary position after that we are detecting the fault position then we are recovering the state bits of the $LFSR_c$.
Remark:- If we consider the time difference of the keystream generation process for consecutive keystream bits then also we can construct some linear equations involving the state bits of $LFSR_c$.

4 Design specification of Achterbahn stream cipher

In this section we shall discuss about the design specification of the Achterbahn stream cipher [4]. Firstly the cipher follows KSG algorithm to initialize the states then it follows key-generation algorithm to generate keystream bits. The cipher is based on 8 NLFSRs and 8 linear filter functions $a(x), b(x), ..., h(x)$ and a non-linear combining function. These filter functions are determined by the secret key. The number of variables involved in the linear filter functions are 6, 7, 7, 8, 8, 9, 9, 10 respectively. The non-linear combining function R is of degree 3 involving 8 variables, which are the outputs of the linear filter functions. The design specification of the keystream generator of the cipher is given in the following figure 2. The algebraic normal form of the non-linear combining

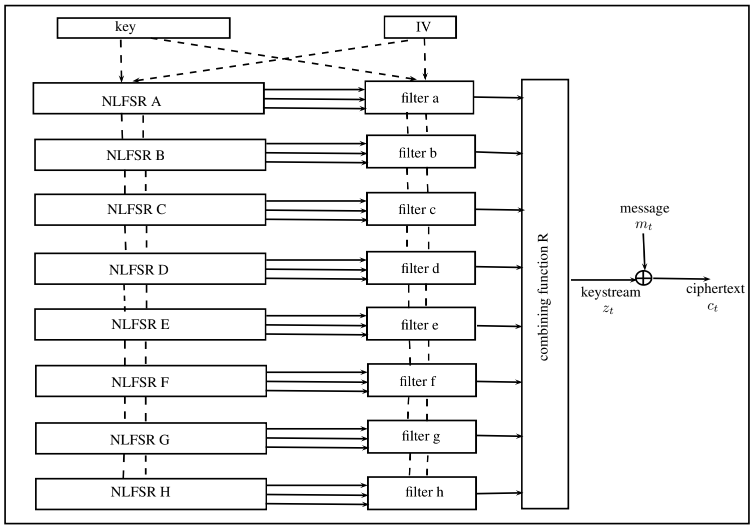

Figure 2: Design specification of Achterbahn

function R is,

$$R(y_1, y_2, ..., y_8) = y_1 + y_2 + y_3 + y_4 + y_5y_7 + y_6y_7 + y_6y_8 + y_5y_6y_7 + y_6y_7y_8.$$

The polynomial corresponding to the linear filter function of the filter-a is given by $a(x) = a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + 1$. Where a_1 is the first bit of the secret key, a_2 is the second bit of the secret key i.e. a_i is the i-th bit of the secret key. After KSG filter-a takes first 6 bits as input to generate a single bit output. Similarly the other filters also follow same types of procedures. The details of the cipher is given in [4]. The feedback function of the NLFSR-A is given by,

$$\begin{aligned} A(x_0, x_1, ..., x_{21}) =& x_0 + x_5 + x_6 + x_7 + x_{10} + x_{11} + x_{12} + x_{13} + x_{17} + x_{20} \\ &+ x_2x_7 + x_4x_{14} + x_8x_9 + x_{10}x_{11} + x_1x_4x_{11} + x_1x_4x_{13}x_{14}. \end{aligned}$$

The feedback functions of the other NLFSR- B, C, D, E, F, G, H are given in [4]. In the keystream generation phase each NLFSR updates by usual procedure. In each clocking 8 linear filter functions takes some bits of the current states of the corresponding NLFSRs' and generates 8 single bit output. After that the non-linear combining function takes these 8 bits as input to generate keystream bit.

5 Fault analysis on Achterbahn by injecting fault in the NLFSR-A

In this section we shall discuss about the fault analysis on Achterbahn stream cipher [4] by injecting a single bit fault in the first NLFSR. Before injecting fault to the cipher we will assume that the attacker has the following freedom,

- Attacker can inject fault in any arbitrary position of the NLFSR-A after the key initialization step
- After injecting fault the attacker can reset the cipher to normal position.

Firstly, we assume that the algebraic expression of the linear filter function filter-a is known to the attacker. We will denote the faulty keystream bit by z'_t and normal keystream bit by z_t. As all the NLFSRs are independent then if we inject a single bit fault in any arbitrary position of the first NLFSR that fault will never affects other NLFSRs in any clocking. If we inject a single bit fault in the first NLFSR then that fault will change the output of the filter-a at certain clocking. In that clocking we will get faulty y_1 (output of filter-a) which will generate a faulty keystream bit, as in the expression of R, y_1 presents as linearly. As y_1 presents in the combining function R as linear term then by observing the normal and faulty keystream bits attacker can find that whether the fault in the NLFSR-A is affecting the output of the filter function or not as the difference between faulty keystream bit and the normal keystream bit is equal to $z' + z = R'(\cdot) + R(\cdot) = y'_1 + y_1 = a'(\cdot) + a(\cdot)$. So by observing the difference between normal and faulty keystream bits the attacker can find that whether the injected fault presents in the bits of NLFSR-A which are involved in the linear filter function filter-a.

Now we discuss about the detection procedure of the fault position after injecting the fault in the first NLFSR. For example we consider the expression of the filter function filter-a is $f_a(x_0, x_1, ...x_5) = x_0 + x_1 + x_2 + x_3 + x_5$. Now we inject a fault in arbitrary position of the NLFSR-A. After injecting the fault attacker will start observing the normal keystream bits and the faulty keystream bits. Suppose after injecting fault the attacker observes that $z'_t + z_t = 1$ for first time for $t = 10$, then for $t = 12, 13, 14, 15$. Then we can tell that after injecting fault, next 9 keystreams are not affected by the fault but $10^{th}, 12^{th}, 13^{th}, 14^{th}, 15^{th}$ keystream bits are affected by the fault. As x_5 is the highest indexed variable involved in the filter function, so when the fault moves to 6^{th} position of the NLFSR-A it affects the output of the filter function filter-a. As the output of the filter function is affected then it will affect the output of the combining function R also i.e. the keystream bit. For $t = 10$ we are getting first time $z'_t + z_t = 1$. It means after injecting the fault the cipher will produce fault free keystream bits for 9 clockings and the attacker will get first time $z'_t + z_t = 1$ at 10^{th} clocking. That means initially fault was 16^{th} position. By following this procedure the attacker can detect the fault position in each clocking.

5.1 Determining the bits of NLFSR-A after key initialization phase

Now we shall discuss the procedure for finding the bits of the NLFSR-A. By the procedure explained in the previous section we can detect the position of the fault in each clocking. By following the fault detection procedure the attacker knows the fault position in each clocking. Now the attacker will use this information to get some bits of the NLFSR-A or to construct some low degree algebraic equations involving NLFSR bits only, which one can solve easily.

Suppose the fault is at x_{10}. The algebraic normal form of the feedback function of the NLFSR-A is,

$$\begin{aligned}A(x_0, x_1, ..., x_{21}) =& x_0 + x_5 + x_6 + x_7 + x_{10} + x_{11} + x_{12} + x_{13} + x_{17} + x_{20} \\ &+ x_2x_7 + x_4x_{14} + x_8x_9 + x_{10}x_{11} + x_1x_4x_{11} + x_1x_4x_{13}x_{14}.\end{aligned}$$

As the fault is at x_{10} then the expression of the feedback function will be,

$$\begin{aligned}\tilde{A}(x_0, x_1, ..., x_{21}) =& x_0 + x_5 + x_6 + x_7 + x'_{10} + x_{11} + x_{12} + x_{13} + x_{17} + x_{20} \\ &+ x_2x_7 + x_4x_{14} + x_8x_9 + x'_{10}x_{11} + x_1x_4x_{11} + x_1x_4x_{13}x_{14}.\end{aligned}$$

So the normal feedback is $A(\cdot)$ and the faulty feedback is $\tilde{A}(\cdot)$. Consider the difference between the normal feedback and the faulty feedback and the difference will be,

$$\begin{aligned}A(\cdot) + \tilde{A}(\cdot) &= x_{10} + x'_{10} + x_{10}x_{11} + x'_{10}x_{11} \\ &= 1 + x_{11}(x_{10} + x'_{10}) \\ &= 1 + x_{11}.\end{aligned}$$

By fault detection procedure the attacker can find the value of $A(\cdot) + \tilde{A}(\cdot)$ by observing the normal and faulty keystream bits. By using the value of $A(\cdot) + \tilde{A}(\cdot)$ the attacker can find the value of x_{11}.

Now if the fault is at x_8 then the faulty feedback of the NFSR-A is $\tilde{A}(\cdot)$. The expression of the faulty feedback will be,

$$\begin{aligned}\tilde{A}(x_0, x_1, ..., x_{21}) =& x_0 + x_5 + x_6 + x_7 + x_{10} + x_{11} + x_{12} + x_{13} + x_{17} + x_{20} \\ &+ x_2x_7 + x_4x_{14} + x'_8x_9 + x_{10}x_{11} + x_1x_4x_{11} + x_1x_4x_{13}x_{14}.\end{aligned}$$

So the normal feedback is $A(\cdot)$ and the faulty feedback is $\tilde{A}(\cdot)$. Now observe the difference between the normal feedback and the faulty feedback and the difference will be,

$$\begin{aligned}A(\cdot) + \tilde{A}(\cdot) &= x_8x_9 + x'_8x_9 \\ &= x_9(x_8 + x'_8) \\ &= x_9.\end{aligned}$$

By fault detection procedure the attacker can find the value of $A(\cdot) + \tilde{A}(\cdot)$ by observing the normal and faulty keystream bits. By knowing the value of $A(\cdot) + \tilde{A}(\cdot)$ the attacker can find the value of x_9. Similarly if the fault is at x_9 then we can get the value of x_8. By following the same procedure attacker can get the value of x_2 and x_7 also.

Suppose fault is at x_i other than above variables. When this x_i moves to x_{10} position then x_{i+1} will move to x_{11}. Then by above procedure we can find the value of x_{i+1}. By this way we can find values of some other bits also.

By the above observation one can find some bits of the NLFSR-A. Now we will try to construct some low degree algebraic equations involving the NLFSR-A bits only, by using normal and faulty keystream bits. By the fault detection procedure the attacker can detect the position of the fault. Suppose the fault is at position x_{13} then the expression of the faulty feedback of the NLFSR-A is,

$$\begin{aligned}\tilde{A}(x_0, x_1, ..., x_{21}) =& x_0 + x_5 + x_6 + x_7 + x_{10} + x_{11} + x_{12} + x'_{13} + x_{17} + x_{20} \\ &+ x_2x_7 + x_4x_{14} + x_8x_9 + x_{10}x_{11} + x_1x_4x_{11} + x_1x_4x'_{13}x_{14}.\end{aligned}$$

Now consider the difference between $A(\cdot)$ and $\tilde{A}(\cdot)$,

$$\begin{aligned} A(\cdot) + \tilde{A}(\cdot) &= (x_{13} + x'_{13}) + (x_1x_4x'_{13}x_{14} + x_1x_4x_{13}x_{14}) \\ &= 1 + x_1x_4x_{14}(x_{13} + x'_{13}) \\ &= 1 + x_1x_4x_{14}. \end{aligned}$$

By observing the difference between the normal keystream bits and faulty keystream bits we can find the value of $A(\cdot) + \tilde{A}(\cdot)$. Then the above equation will be $1 + x_1x_4x_{14} = a$, where a is a known value.

If the fault is at x_{11} then the expression of the faulty feedback of the NLFSR-A is,

$$\begin{aligned} \tilde{A}(x_0, x_1, ..., x_{21}) =& x_0 + x_5 + x_6 + x_7 + x_{10} + x'_{11} + x_{12} + x_{13} + x_{17} + x_{20} \\ &+ x_2x_7 + x_4x_{14} + x_8x_9 + x_{10}x'_{11} + x_1x_4x'_{11} + x_1x_4x_{13}x_{14}. \end{aligned}$$

Consider the difference between $A(\cdot)$ and $\tilde{A}(\cdot)$,

$$\begin{aligned} A(\cdot) + \tilde{A}(\cdot) &= (x_{11} + x_{10}x_{11} + x_1x_4x_{11}) + (x'_{11} + x_{10}x'_{11} + x_1x_4x'_{11}) \\ &= 1 + x_{10} + x_1x_4. \end{aligned}$$

By following the above mentioned procedure we can construct more low degree equations involving the state bits of NLFSR-A after the key initialization phase. After that we solve these equations by using the known bits (previously found) to get the values of the other bits. The following table describes the fault location vs bits obtained of the NLFSR-A.

Fault location	NLFSR bits obtained
x_2	x_7
x_7	x_2
x_{10}	x_{11}, x_5
x_8	x_9, x_3
x_9	x_8, x_4
x_{11}	x_{10}, x_6
x_{12}	x_{11}
x_{13}	x_{12}, x_{14}
x_{16}	x_{15}, x_{17}
x_{19}	x_{18}, x_{20}
x_{21}	x_{22}

Table 1: Fault location vs NLFSR bit obtained

From the above table we can see that attacker can recover almost all the bits of the NLFSR-A after key initialization phase by injecting single bit fault in the first NLFSR. One can apply same procedure to the NLFSR-B, C, D to recover some bits of the state after key-initialization phase.

Remark:- We are not injecting fault in a fixed position. We are injecting fault in any arbitrary position after that we are detecting the fault position by observing the normal and faulty keystream bits. After that we are detecting the NFSR bits.

5.2 Complexity of the attack

In this section we will discuss about the time complexity of this attack. The size of the first NLFSR is 22 bits. After injecting a single bit fault in the first NLFSR attacker may need to clock the cipher maximum 22 times to detect the fault position. After detecting the fault position the attacker can easily recover the NLFSR bit by observing the normal and faulty keystream bits. So the time complexity for detecting the fault position is $22 \times 22 \approx 2^9$. Hence the time complexity to recover the NLFSR state bits after key initialization phase is $\approx 2^9$ which is quite less than exhaustive search.

Remark:- Through out the article we have assumed that the filter function filter-a is known. But now we will discuss about how to get the algebraic normal form the linear filter function filter-a. Suppose attacker injects a single bit fault in any arbitrary position of the NLFSR-A. After injecting the fault the attacker will observe the normal and faulty key stream bits. Suppose at t_1-th clocking attacker first observes that $z_{t_1} + z'_{t_1} = 1$. After that at t_2-th clocking he/she observes that $z_{t_2} + z'_{t_2} = 1$. Now the attacker can easily find the value of $t_2 - t_1$. Next the attacker can construct equation involving t_2 and t_3. By following the same procedure finally he/she will get one equation $n - t_k = a$, where a is a known value and n is also known as first position bit is always involved in the expression of the linear filter function. For detail of the linear filter function see page no. 12 of [4]. Now the attacker can easily solve the systems involving t_i's using one known value $t_k = n - a$. Then by using this t_k the attacker can find t_{k-1}. After getting the values of t_i's the attacker can find the algebraic normal form the linear filter function filter-a.

6 Conclusion

In this paper we have introduced the fault analysis on the stream cipher LILI-128. Firstly, we have recovered the state bits of the $LFSR_c$ by injecting fault in the $LFSR_c$. After that we have recovered the state bits of the $LFSR_d$ by using classical algebraic attack technique. We have also described the fault analysis on the Achterbahn stream cipher by injecting fault in the first NLFSR. In this paper we have shown that by injecting fault into NLFSR-A attacker will able to recover almost all the state bits of the NLFSR-A after key initialization phase. We are also able to construct some low degree equations involving the state bits of the NLFSR-A only, which may be helpful for other cryptanalysis purpose on this cipher. Similarly, one can give same type of analysis for NLFSR-B, C, D also.

References

[1] Berzati, A., Canovas, C., Castagnos, G., Debraize, B., Goubin, L., Gouget, A., Paillier, P., Salgado, S.: Fault analysis of grain-128. In: Hardware-Oriented Security and Trust, 2009. HOST'09. IEEE International Workshop on. pp. 7–14. IEEE (2009)

[2] Courtois, N.T., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In: Advances in CryptologyEUROCRYPT 2003, pp. 345–359. Springer (2003)

[3] Dawson, E., Clark, A., Golic, J., Millan, W., Penna, L., Simpson, L.: The lili-128 keystream generator. In: Proceedings of first NESSIE workshop. Citeseer (2000)

[4] Gammel, B., Göttfert, R., Kniffler, O.: The achterbahn stream cipher. Submission to eSTREAM (2005)

[5] Hoch, J.J., Shamir, A.: Fault analysis of stream ciphers. In: Cryptographic Hardware and Embedded Systems-CHES 2004, pp. 240–253. Springer (2004)

[6] Hojsik, M., Rudolf, B.: Differential fault analysis of trivium. In: Fast Software Encryption. pp. 158–172. Springer (2008)

[7] Johansson, T., Meier, W., Muller, F.: Cryptanalysis of achterbahn. In: Fast software encryption. pp. 1–14. Springer (2006)

[8] Karmakar, S., Chowdhury, D.R.: Fault analysis of grain-128 by targeting nfsr. In: Progress in Cryptology–AFRICACRYPT 2011, pp. 298–315. Springer (2011)

[9] Kircanski, A., Youssef, A.M.: Differential fault analysis of rabbit. In: Selected Areas in Cryptography. pp. 197–214. Springer (2009)