A Probabilistic Algebraic Attack on the Grain Family of Stream Ciphers

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

Abstract:- In 2005, Hell, Johansson and Meier submitted a stream cipher proposal named Grain v1 to the estream call for stream cipher proposals and it also became one estream finalists in the hardware category. The output function of Grain v1 connects its 160 bits internal state divided equally between an LFSR and an NFSR, using a non-linear filter function in a complex way. Over the last years many cryptanalyst identified several weaknesses in Grain v1. As a result in 2011 the inventors modified Grain v1 and published a new version of Grain named Grain-128a which has a similar structure as Grain v1 but with a 256 bits internal state with an optional authentication is the latest version of Grain family resisting all known attacks on Grain v1. However both these ciphers are quite resistant against the classical algebraic attack due to the rapid growth of the degree of the key-stream equations in subsequent clockings caused by the NFSR. This paper presents a probabilistic algebraic attack on both these Grain versions. The basic idea of our attack is to develop separate probabilistic equations for the LFSR and the NFSR bits from each key-stream equations. Surprisingly it turns out that in case of Grain-128a our proposed equations hold with all most sure probability, which makes the sure retrieval of the LFSR bits. We also outline a technique to reduce the growth of degree of the equations involving the NFSR bits for Grain v1. Further we high light that the concept of probabilistic algebraic attack as proposed in this paper can be considered as a generic attack strategy against any stream cipher having similar structure of the output function as in case of the Grain family.
Keywords:- Boolean Function, Grain v1, Grain-128a, Algebraic Attack, Probabilistic Algebraic Attack.

1 Introduction

Grain v1 [11] is one of the finalist in the hardware category of the estream project. This cipher is based on an 80 bits LFSR, an 80 bits NFSR and a non-linear filter function. This stream cipher was introduced by Hell, Johansson and Meier [11] in 2005. The key-stream generation function combines some particular state bits of the LFSR as well as the NFSR using the non-linear Boolean function in a complex way. Detail specification of this cipher is described in Section 3.

Grain-128a [2] is the latest modified version of Grain family. This cipher is based on an 128 bits LFSR , an 128 bits NFSR and a non-linear filter function and two different modes of operations:

with or without authentication. This cipher is proposed by Agren, Hell, Johansson and Meier in 2011. Design specification of this cipher is described in Section 4.

Algebraic attack, introduced by N.T. Courtois and W. Meier [6], is a well studied cryptanalytic technique against stream ciphers. The basic principle of this attack is to express the relation between some internal state (may be the secret key itself) and some known key-stream bits (may not be consecutive) as a large system of multivariate polynomial equations and attempt to solve this system in order to retrieve that secret internal state subsequently. There are some existing algorithm e.g. re-linearization, XL algorithm [5] and Gröbner bases [13], [9], [10] etc. algorithm for solving systems of multivariate polynomial equations. However, the efficiency of these algorithms strongly depends on the algebraic degree of the equations.

In 2003 Courtois et.al. [6] proved the existence of low degree multiple of Boolean functions and showed how to use them to generate low degree multivariate equations. In [6] Courtois and Meier proved that finding low degree multiple of Boolean function f is actually equivalent to finding low degree annihilators of f or $1 + f$.

However, this classical version of algebraic attack is well suited for combiner or filter models purely based on LFSR, this method cannot be applied directly to stream ciphers involving NFSR such as Grain v1 [11] or Grain-128a [2]. This is due to the non-linear feedback function and presence of bits from NFSR bits in non-linear filter function, the degree of the generated equations increases quite rapidly in successive clockings.

In 2005 An Braeken and Bart Preneel [4] introduced one type of probabilistic variant of algebraic attack for LFSR based stream ciphers combined with non-linear Boolean function. Their method involves finding probabilistic equations of certain low degree by determining approximate low degree annihilators of the combiner or filter function or its boolean complement with high probability. But this method is again unsuitable for Grain family due to the structure of its output function.

In many papers in the literature they have given different types of attack on Grain v1 [1], [12] [14], [3]. The only paper concerning the algebraic attack on Grain v1 is by Mehreen Afzal and Ashraf Masood [1]. But in [1], the author described algebraic attack on Grain v1 based on computer simulations. But there is no proper explanation of the results obtained. In case of Grain-128a, however, no such attempt has yet been made.

In this paper we attempt to develop a probabilistic algebraic attack on Grain family. The basic idea of our attack is to generate two simultaneous probabilistic equations, one involving only LFSR bits and the other only NFSR bits, from each output equations. Then we can construct two separate probabilistic systems corresponding to LFSR and NFSR bits respectively. Now the classical algebraic attack strategy can be applied on the LFSR part. For the NFSR part of Grain v1 we show that by knowing half of the NFSR state bits provides sufficient number of low degree equations which may helpful for algebraic attack. In this connection we would also like to mention that, though the basic process of generation of the probabilistic equations in both the cases is the same, for Grain-128a we obtain overwhelming probability (close to 1) of the generated equations which makes our attack more effective for this case. We also show that the probability of obtaining the correct internal state of LFSR part for Grain family following our strategy. Our approach in this paper can be viewed as a general attack strategy against stream cipher having similar structure of the output function as in case of the Grain family. To the best of our knowledge, this type of probabilistic approach in algebraic attack has not been attempted in the literature previously. The details of our attack is described from Section 7.

The rest of the paper is organised as follows: in Section 2 we discuss some basic definitions. Algebraic attack for LFSR based stream cipher is discussed in Section 5. Low degree multiple of NFSR feedback function of Grain v1 is given in Section 6. Then in Section 7 we discuss our probabilistic algebraic attack for Grain v1. Low degree multiple for LFSR bits equations of Grain v1 is discussed in Section 8. Construction of low degree probabilistic equations for the NFSR part of Grain v1 is given in Section 9. In Section 10 we discuss some observations on the NFSR bits equations for Grain v1. Probabilistic algebraic attack on the LFSR of the Grain-128a stream cipher is discussed in Section 11. In Section 12 we discuss about the success probability of obtaining the LFSR states of Grain family. Finally the paper is concluded in Section 13.

2 Definitions

Definition 1. ***Boolean function***
A Boolean function of n variables is a mapping from $\{0,1\}^n$ to $\{0,1\}$. The set of all Boolean functions of n- variables is denoted by $\mathcal{B}_n$

Definition 2. ***Algebraic normal form***
The algebraic normal form of a Boolean function f is the multivariate polynomial expression of that function. The polynomial expression is given by

$$f(x_1, ..., x_n) = a_0 \oplus \bigoplus_{1\leq i\leq n} a_i x_i \oplus \bigoplus_{1\leq i<j\leq n} a_{ij} x_i x_j \oplus ..... \oplus a_{12...n} x_1 x_2 .... x_n.$$

Definition 3. ***Degree of Boolean function***
The algebraic degree or simply degree of a Boolean function f is defined as the number of variables present in the highest order product term of its ANF.

Definition 4. ***Linear and Non-linear Boolean function***
A Boolean function f is affine if the degree of that function is one. The set of all affine functions of n variables is denoted by $A(n)$. Any affine function can be written as $l_{u,b}(x) =< u, x > +b$, where $u = (a_1, ..., a_n) \in \mathbb{F}_2^n$ and $b = a_0 \in \mathbb{F}_2$. An affine function with constant term equal to zero ($i.e.$ $a_0 = 0$) is called a linear function and the set of all n variable linear functions is denoted by C_n. Any n variable Boolean function which lies outside A_n is called non-linear Boolean function.

3 Design specification of Grain v1 stream cipher

Grain v1 [11] stream cipher is based on one 80 bits LFSR, one 80 bits NFSR and one non-linear filter function of 5 variables. The initial LFSR's bits are denoted by $s_i,\ i = 0, 1, ..., 79$ and the initial NFSR's bits are denoted by $b_i,\ i = 0, 1, ..., 79$. The feedback polynomial of the LFSR is denoted by $f(x)$ and it is a primitive polynomial in $\mathbb{F}_{2^{80}}$. The feedback polynomial of the NFSR is denoted by $g(x)$. And the non-linear filter function is denoted by $h(x)$. The design of Grain v1 is given in Fig.1 where $f(x) = 1 + x^{18} + x^{29} + x^{42} + x^{57} + x^{67} + x^{80}$. So the update function of the LFSR is $s_{i+80} = s_{i+62} + s_{i+51} + s_{i+38} + s_{i+23} + s_{i+13} + s_i$. The feedback polynomial of the NFSR is

$$\begin{aligned} g(x) =& 1 + x^{18} + x^{20} + x^{28} + x^{35} + x^{43} + x^{47} + x^{52} + x^{59} + x^{66} + x^{71} + x^{80} \\ &+ x^{17}x^{20} + x^{43}x^{47} + x^{65}x^{71} + x^{20}x^{28}x^{35} + x^{47}x^{52}x^{59} + x^{17}x^{35}x^{52}x^{71} \\ &+ x^{20}x^{28}x^{43}x^{47} + x^{17}x^{20}x^{59}x^{65} + x^{17}x^{20}x^{28}x^{35}x^{43} + x^{47}x^{52}x^{59}x^{65}x^{71} \\ &+ x^{28}x^{35}x^{43}x^{47}x^{52}x^{59} \end{aligned}$$

So the form of the state update function of NFSR will be

$$\begin{aligned} b_{i+80} =& s_i + b_{i+62} + b_{i+60} + b_{i+52} + b_{i+45} + b_{i+37} + b_{i+33} + b_{i+28} + b_{i+21} + b_{i+14} \\ & + b_{i+9} + b_i + b_{i+63}b_{i+60} + b_{i+37}b_{i+33} + b_{i+15}b_{i+9} + b_{i+60}b_{i+52}b_{i+45} + \\ & b_{i+33}b_{i+28}b_{i+21} + b_{i+63}b_{i+45}b_{i+28}b_{i+9} + b_{i+60}b_{i+52}b_{i+37}b_{i+33} + \\ & b_{i+63}b_{i+60}b_{i+21}b_{i+15} + b_{i+63}b_{i+60}b_{i+52}b_{i+45}b_{i+37} + \\ & b_{i+33}b_{i+28}b_{i+21}b_{i+15}b_{i+9} + b_{i+52}b_{i+45}b_{i+37}b_{i+33}b_{i+28}b_{i+21}. \end{aligned} \tag{1}$$

These contents of LFSR and NFSR are the current states of the registers. From these states 5 variables are taken as the input for the non-linear function $h(x)$. This $h(x)$ is a non-linear function of 5 variables of non-linearity 12. Among 5 inputs in $h(x)$ 4 are coming from LFSR and one input is coming from NFSR, where, $h(x) = x_1 + x_4 + x_0x_3 + x_2x_3 + x_3x_4 + x_0x_1x_2 + x_0x_2x_3 + x_0x_2x_4 + x_1x_2x_4 + x_2x_3x_4$, where the variable x_4 is coming from NFSR and the other variables x_0,x_1,x_2,x_3 are coming from the LFSR, where x_0, x_1, x_2, x_3 correspond to s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64} and the variable x_4 corresponds to b_{i+63}. The expression of the key stream bit is $z_i = \sum\limits_{k \in A} b_{i+k} + h(s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}, b_{i+63})$ where $A = \{1, 2, 4, 10, 31, 43, 56\}$. The $h(x)$ can be rewritten as $h(x) = x_4 \cdot u(x_0, x_1, x_2, x_3) + v(x_0, x_1, x_2, x_3)$, where $u(x_0, x_1, x_2, x_3) = 1 + x_3 + x_0x_2 + x_1x_2 + x_2x_3$ and $v(x_0, x_1, x_2, x_3) = x_1 + x_0x_3 + x_2x_3 + x_0x_1x_2 + x_0x_2x_3$ [3].

3.1 Key initialization of Grain v1

Before producing any key-stream bits, the cipher must be initialized with the secret key K and initial value IV. Let the secret key bits be $k_i,\ 0 \le i \le 79$ and the initial value bits be $IV_i,\ 0 \le i \le 63$. The cipher is initialized by the following procedure:

- First load the secret key bits by $b_i = k_i,\ 0 \le i \le 79$ in the NFSR .
- Then load the initial value by $s_i = IV_i,\ 0 \le i \le 63$ in the LFSR for first 64 positions, fill the remaining positions by 1 i.e. $s_i = 1,\ 64 \le i \le 79$.

After that the cipher is clocked 160 times without generating any key-stream bits. Whatever the output is coming is fed-back an XOR-ed with the input of the LFSR and the NFSR. The description is given in the following figure.

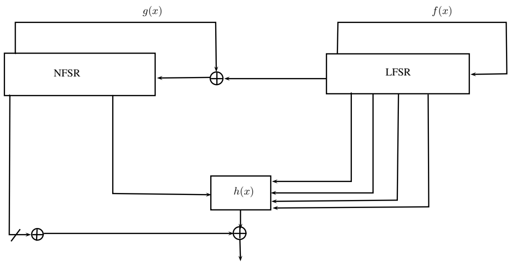

Figure 1: Design specification of Grain v1

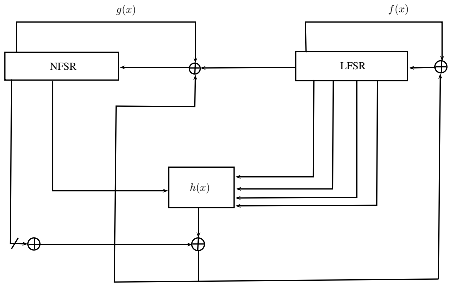

Figure 2: Key initialization of Grain v1

4 Design specification of Grain-128a

Grain-128a [2], latest modified version of Grain family of stream ciphers is based on one 128 bits LFSR, one 128 bits NFSR and one non-linear filter function. It consists of a procedure which produces a pre-output stream and two different modes of operation: with or without authentication. The state of the LFSR is denoted by s_i $0 \leq i \leq 127$ and the state of the NFSR is denoted by b_i, $0 \leq i \leq 127$. The primitive polynomial of the LFSR is denoted by $f(x)$, where $f(x) = 1+x^{32}+x^{47}+x^{58}+x^{90}+x^{121}+x^{128}$. So the update function of the LFSR is $s_{i+128} = s_i+s_{i+7}+s_{i+38}+s_{i+70}+s_{i+81}+s_{i+96}$. The non-linear feedback function for the NFSR is $g(x)$, where

$$\begin{aligned} g(x) = 1 + x^{32} + x^{37} + x^{72} + x^{102} + x^{128} + x^{44}x^{60} + x^{61}x^{125} + x^{63}x^{67} + x^{69}x^{101} \\ + x^{80}x^{88} + x^{101}x^{111} + x^{115}x^{117} + x^{46}x^{50}x^{58} + x^{103}x^{104}x^{106} + x^{33}x^{35}x^{36}x^{40} \end{aligned}$$

The state update function of the NFSR is

$$\begin{aligned} b_{i+128} =& s_i + b_i + b_{i+26} + b_{i+56} + b_{i+91} + b_{i+96} + b_{i+3}b_{i+67} + b_{i+11}b_{i+13}+ \\ & b_{i+17}b_{i+18} + b_{i+27}b_{i+59} + b_{i+40}b_{i+48} + b_{i+61}b_{i+65} + b_{i+68}b_{i+84}+ \\ & b_{i+88}b_{i+92}b_{i+93}b_{i+95} + b_{i+22}b_{i+24}b_{i+25} + b_{i+70}b_{i+78}b_{i+82} \end{aligned}$$

The non-linear filter function h is defined by $h(x) = x_0x_1 + x_2x_3 + x_4x_5 + x_6x_7 + x_0x_4x_8$. Among the 9 variables 2 variables are coming form NFSR state and 7 variables are coming from LFSR state. The variables $x_0, x_1, ..., x_8$ correspond to $b_{i+12}, s_{i+8}, s_{i+13}, s_{i+20}, b_{i+95}, s_{i+42}, s_{i+60}, s_{i+79}$ and s_{i+94} respectively. The output function of the cipher is given by $y_i = h(x) + s_{i+93} + \sum_{j \in A} b_{i+j}$ where $A = \{2, 15, 36, 45, 64, 73, 89\}$. The design description of the output generator is given by the following figure.

4.1 Key and IV initialization of Grain-128a

Before producing any key-stream by the cipher must be initialized with the key and the IV. The secret key is denoted by k_i, $0 \leq i \leq 127$ and the IV bits IV_i, $0 \leq i \leq 95$. First 128 bits of the NFSR is loaded by the secret key bits, $b_i = k_i, 0 \leq i \leq 127$, and the first 96 bits of the LFSR are filled by the IV bits, $s_i = IV_i$, $0 \leq i \leq 95$. The last 32 bits are filled by ones and zeros as $s_i = 1$, $96 \leq i \leq 126$, $s_{127} = 0$. After that the cipher is clocked 256 without producing any key-stream. Instead the output is XORed with the input of the NFSR and LFSR. The design description is given by the following figure.

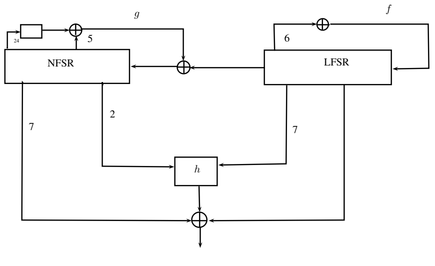

Figure 3: Design specification of Grain-128a

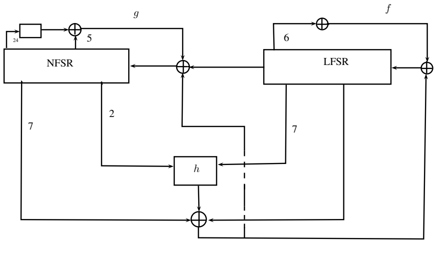

Figure 4: Key initialization of Grain-128a

4.2 Key-stream generation of Grain-128a

Grain-128a[2] is based on two different modes of operations, with and without authentication. Authentication is in on mode when $IV_0 = 1$ and when $IV_0 = 0$ then authentication is in off mode. When $IV_0 = 1$, then the expression of the key-stream bits is $z_i = y_{i+64}$, i.e. we will get even position output as key-stream bits after 64 outputs of the cipher. These 64 bits and the odd positions outputs will be used for authentication purpose.

When $IV_0 = 0$, then the expression of the key-stream bits will be $z_i = y_i$. Details of the cipher can be found in [2].

5 Algebraic attack on stream cipher

Algebraic attack on LFSR based stream cipher was first introduced by Courtois and Meier in 2003 [6]. The basic strategies of this attack are-

- First find algebraic system of equations over the unknown secret key or internal state of the cipher.
- Then try to solve this system to get the secret key.

Suppose the attacker knows some plain-text bits and the corresponding cipher-text bits. Then the attacker will XOR these plain-text bits and the cipher-text bits to get the corresponding key-stream bits. After knowing these key-stream bits, the attacker will try to construct an algebraic system of equations over the unknown secret key, then try to solve this system to get the secret key. In case of LFSR based stream cipher algebraic attack can be implemented efficiently due to the linearity of the LFSR. But in case of the NFSR based stream cipher it is not so easy because of the non-linearity and degree of the feedback function of the NFSR. At each clocking the degree of the expression of the key-stream bits in terms of the secret internal state variables will increase rapidly. In 2003 N.T. Courtois [6] proved a theorem for existence of low degree multiple of a given boolean function. The statement of that theorem is as follows

Theorem 5.1. *For any $f \in \mathcal{B}_n$, there is a non-zero $g \in \mathcal{B}_n$ of degree at most $\lceil \frac{n}{2} \rceil$ such that fg is of degree at most $\lceil \frac{n}{2} \rceil$.*

If the degree of the equation of the key-stream is high then we can multiply that expression by a low degree boolean function to make it a low degree equation. Previous theorem ensures the existence of such low degree multiple of a Boolean function. In case of NFSR based stream cipher we may get low degree multiples of different degrees for each equations as the degree of the key-stream expressions changes with different clockings.

6 Existence of low degree multiple of non-linear feedback function of Grain v1

The expression of the key-stream bit of Grain v1 stream cipher is
$z_i = \sum_{k \in A} b_{i+k} + h(s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}, b_{i+63}), \qquad A = \{1, 2, 4, 10, 31, 43, 56\}$.

Where the b_i's are coming from the NFSR and s_i's are coming from the LFSR. As the degree of the NFSR feedback function is 6 then the degree of the expression of the key-stream bit will change in multiples of 6. The b_i's present in the expression of the key-stream bit equations are either linear or it can be expressed in terms of the previous state variables by a sixth degree equation. Now if we multiply $g(x)$ by $x^{20'}x^{52'}$ we will get

$$g(x) \cdot x^{20'}x^{52'} = x^{20'}x^{52'}[1 + x^{18} + x^{28} + x^{35} + x^{43} + x^{47} + x^{59} + x^{66} + x^{71} + x^{80} + x^{43}x^{47} + x^{65}x^{71}]$$

So $g(x) \cdot x^{20'}x^{52'} = U(x) \cdot V(x)$, where, $U(x) = x^{20'}x^{52'}$ and $V(x) = 1 + x^{18} + x^{28} + x^{35} + x^{43} + x^{47} + x^{59} + x^{66} + x^{71} + x^{80} + x^{43}x^{47} + x^{65}x^{71}$. So, if we multiply $g(x)$ by $x^{20'}x^{52'}$ then we are getting $g(x) \cdot x^{20'}x^{52'}$ as a multiplication of two low degree Boolean functions of degree 2. Now if we multiply the state update function by $b'_{i+28} \cdot b'_{i+60}$ then we will also get same form of $b_{i+80} \cdot (b'_{i+28} \cdot b'_{i+60})$. The expression will be,

$$\begin{aligned} b_{i+80} \cdot (b'_{i+28} \cdot b'_{i+60}) =& b'_{i+28} \cdot b'_{i+60}[s_i + b_{i+62} + b_{i+52} + b_{i+45} + b_{i+37} + b_{i+33} + \\ & b_{i+21} + b_{i+14} + b_{i+9} + b_i + b_{i+37}b_{i+33} + b_{i+15}b_{i+9}]. \end{aligned}$$

Using this technique we can reduce the degree of the equations of the key-stream bits.

7 Probabilistic algebraic attack on Grain v1

In this section we shall introduce a new probabilistic algebraic attack on the stream cipher Grain v1. The key-stream expression of the Grain v1 stream cipher has two components, one is a linear combination of the NFSR bits and the other is the output of the non-linear filter function. Precisely, $z_i = A_i + b_{i+63}u(\cdot) + v(\cdot)$, where $A_i = \sum_{k \in A} b_{i+k}$ where $A = \{1, 2, 4, 10, 31, 43, 56\}$. Now for $z_i = 0$ we are getting $A_i + (b_{i+63}u(\cdot) + v(\cdot)) = 0$, similarly for $z_i = 1$ we will get $1 + A_i + (b_{i+63}u(\cdot) + v(\cdot)) = 0$. First we consider the equation for $z_i = 0$, i.e. $A_i + (b_{i+63}u(\cdot) + v(\cdot)) = 0$. Now multiply this equation by $u'(\cdot) = 1 + u(\cdot)$. After this multiplication we get $u'(\cdot)A_i + u'(\cdot)v(\cdot) = 0$. This equation is of the form $X_i + Y_i = 0$ where $X_i = u'(\cdot)A_i$ and $Y_i = u'(\cdot)v(\cdot)$ (exactly in the same way we can construct similar type of equation for $z_i = 1$ with $X_i = (1 + A_i)u'(\cdot)$, $Y_i = u'(\cdot)v(\cdot)$).

We will discuss the case for $z_i = 0$, the discussion will be similar for $z_i = 1$. Now for $z_i = 0$ we are getting $X_i + Y_i = 0$. From this equation we can easily tell that there are only two possible cases for X_i, Y_i either (i) $X_i = 0, Y_i = 0$ or (ii) $X_i = 1, Y_i = 1$. Let $p = Pr[X_i = 0, Y_i = 0]$ and $q = Pr[X_i = 1, Y_i = 1]$. Now obviously we will choose the case corresponding to the higher one between p and q in order to increase our success probability. In fact we will prove shortly that $p > q$ and similarly for $z_i = 1$.

Thus in this way, for each clocking we get two probabilistic equations with certain probability. Now consider the equations corresponding to Y_i's. This equations will be of the form $u'(\cdot)v(\cdot) = Y_i$, where $Y_i \in \mathbb{F}_2$. This is a probabilistic system of equations involving only LFSR bits, which we can solve using the classical algebraic attack technique to obtain probabilistic values of the LFSR state bits. The low degree multiple of these equations, as described in Section 8 can be useful in this respect.

After solving the previous system involving the LFSR state bits we will put all LFSR bits values to the original equations of z_i. Then the form of these equations will be $z_i = \sum_{k \in A} b_{i+k} + b_{i+63} \cdot u(\cdot) + v(\cdot)$ where $u(\cdot)$ and $v(\cdot)$ are known as all bits of the LFSR state are now known, only NFSR bits are now unknown variables in these equations. The procedure for solving this system of NFSR state variables will be considered in more detail in the Section 10.

Now we will calculate $p = Pr[X_i = 0, Y_i = 0]$. Firstly we note that here we assume that the probability of each of the initial state variables being 0 or 1 after the key initialization phase is $\frac{1}{2}$. Moreover as clearly mentioned in [11] the NFSR feedback function is balanced, so $Pr[\sum\limits_{k \in A} b_{i+k} = 0 \; or \; 1]$ is $\frac{1}{2}$. Also the functions $u'(\cdot)$ and $v(\cdot)$ both are balanced Boolean functions, but $Y_i = u'(\cdot)v(\cdot)$ is not balanced in fact $Pr[Y_i = 0]$ is $\frac{3}{4}$. Truth table of these functions are given in the appendix A. Now,
$p = Pr[X_i = 0, Y_i = 0] = Pr[X_i = 0|Y_i = 0] \cdot Pr[Y_i = 0] = \frac{3}{4} \cdot Pr[X_i = 0|Y_i = 0]$ Now we need to find $Pr[X_i = 0|Y_i = 0]$. Clearly,

$X_i = 0 \implies (i) \; u'(\cdot) = 0, \sum_{k \in A} b_{i+k} = 0; \; or \; (ii) \; u'(\cdot) = 0, \sum_{k \in A} b_{i+k} = 1;$
$or \; (iii) \; u'(\cdot) = 1, \sum_{k \in A} b_{i+k} = 0$
Also, $Y_i = 0 \implies (i) \; u'(\cdot) = 0, v(\cdot) = 0; \; or \; (ii) \; u'(\cdot) = 0, v(\cdot) = 1;$
$or \; (iii) \; u'(\cdot) = 1, v(\cdot) = 0$
Total number of cases when $u'(\cdot) = 0, v(\cdot) = 0$ and $u'(\cdot) = 0, v(\cdot) = 1$ is 8 as $u'(\cdot)$ is balanced function, under these cases $\sum_{k \in A} b_{i+k}$ can take any values to make $X_i = 0$. Hence total number of favourable cases under above scenario will be 8×2^7. Total number of cases when $u'(\cdot) = 1, v(\cdot) = 0$ is 4, under this case $\sum_{k \in A} b_{i+k}$ can take only 0 value to make $X_i = 0$. Hence total number of favourable cases under this scenario will be 4×2^6. Hence total number of favourable cases will be $8 \times 2^7 + 4 \times 2^6$. The number of cases when $Y_i = 0$ is 12. Now total number of possible cases will be 12×2^7. Hence the probability $Pr[X_i = 0|Y_i = 0] = \frac{(8 \times 2^7 + 4 \times 2^6)}{12 \times 2^7} = \frac{5}{6}$. Hence the required probability is $Pr[X_i = 0, Y_i = 0] = \frac{3}{4} \times \frac{5}{6} = \frac{5}{8}$

Thus we see that for $z_i = 0$; $X_i = 0, Y_i = 0$ has the higher probability of occurrence. A similar study can be done for $z_i = 1$. Now due to the independence of the equations corresponding to different clockings, the probability of the total system of equations will be obtained by multiplying all the probabilities corresponding to all the clockings considered.

8 On the solution of the equations involving LFSR bits of Grain v1

By the discussion of the Section 3 the expression of the key-stream bit is

$$\begin{aligned} z_i &= \sum_{k \in A} b_{i+k} + h(s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}, b_{i+63}) \\ &= \sum_{k \in A} b_{i+k} + b_{i+63} \cdot u(s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}) + v(s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}). \end{aligned}$$

Now after multiplying the above equation by $u'(\cdot)$ we will get an equation of the form $X_i + Y_i = 0$. Now by the discussions of the previous section we will choose the equations $X_i = 0, Y_i = 0$ for $z_i = 0$ (similar study for $z_i = 1$). Now $Y_i = 0$ means $u'(\cdot)v(\cdot) = 0$. This is an equation involving only LFSR bits. The expression of the function $u'(\cdot)v(\cdot)$ is $x_0x_3 + x_1x_3 + x_1x_2 + x_0x_1x_2$. So the degree of each equations involving LFSR bits will be 3 and remains constant for all clockings. As $u'(\cdot)v(\cdot)$ is a third degree function of 4 variables, by the theorem in Section 5 we can surely tell that $u'(\cdot)v(\cdot)$ has low degree multiple such that the degree of the resulting function becomes at most

2. So finally we have a probabilistic system of equations of degree at most 2 involving only LFSR bits as unknowns. This system can be solved by using any existing polynomial system solving algorithms [5], [13], [9], [10]. After solving this system we will get the probabilistic LFSR state bits.

Time complexity for solving this system:- The degree of each equation for the LFSR of Grain-v1 is finally 2. The number of monomials of degree 2 of 80 state variables is $T = \binom{80}{2} = 3160$. So we need 3160 number of variables to linearize the system. After linearization we will solve the system by Gauss elimination method. As modern CPU can handle 64 operations in one single clock of CPU. Hence the complexity of Gauss reduction for this system will be $\frac{7 \cdot T^{log_2 7}}{64} \approx 2^{30}$ CPU clocks which is less than exhaustive search.

9 Existence of low degree probabilistic equations for the NFSR of Grain-v1

In this section we will discuss the existence of low degree probabilistic equations for the NFSR part of Grain-v1. The expression of the key-stream bit of Grain-v1 is $z_i = \sum\limits_{k \in A} b_{i+k} + b_{i+63} \cdot u(\cdot) + v(\cdot)$ where $u(\cdot),\ v(\cdot)$ are balanced Boolean function. If we multiply both sides of the expression by b'_{i+63} we will get, $b'_{i+63} \cdot [z_i + \sum_{k \in A} b_{i+k}] + b'_{i+63} \cdot v(\cdot) = 0$. Now consider $A_i = b'_{i+63} \cdot [z_i + \sum\limits_{k \in A} b_{i+k}]$ and $B_i = b'_{i+63} \cdot v(\cdot)$. i.e $A_i + B_i = 0$. From this equation we can tell either (i) $A_i = 0, B_i = 0$ or (ii) $A_i = 0, B_i = 1$. As b'_{i+63}, $z_i + \sum\limits_{k \in A} b_{i+k}$ and $v(\cdot)$ are balanced by same (previously described) procedure we will get $Pr[A_i = 0, B_i = 0] = \frac{5}{8}$. Now $A_i = 0$ is an equation involving only NFSR bits of Grain-v1. So for each clocking we can construct these types of probabilistic low degree equations involving NFSR bits only.

10 Some observations on the degree of the equations involving NFSR bits of Grain v1

In this section we will discuss some observations on the NFSR bits' equations. In the Section 8 we have discussed how to tackle the LFSR bits equations using classical algebraic attack technique. However due to the non-linear feedback function, the degree of the equations involving the NFSR bits does not remain fixed, rather it increases quite rapidly. In this section we will try to control this rapid increase in the degree of the equations and will finally show that if we know half of the NFSR bits, then we obtain sufficient number of equations of quite low degree (at most degree 4) which is feasible to solve by the existing algorithms.

The equation involving the NFSR bits are of the form $\sum\limits_{k \in A} b_{i+k} + b_{i+63} u(\cdot) + v(\cdot) = 0$ where $A = \{1, 2, 4, 10, 31, 43, 56\}$. Note that unless $i + 63 = 80$ i.e. $i = 17$ no equation of above form will involve any of the bits that are derived from the non-linear feed back. Thus we will obtain 17 linear equations involving the NFSR bits. Similarly one can observe that the next 7 equations will involve derived bits in only one position, next 13 equations with feedback in two positions, next 12 equations with feedback in three positions, next 21 equations with feedback in four positions, next

6 equations with feedback in five positions and next 2 equations with feedback in six positions and next equation with feedback in seven positions and next onwards we will have derived bits in all the eight positions. Now consider the NFSR feedback expression 1. Note that this feedback function has the following low degree multiple of degree 2.

$$\begin{aligned} b_{i+80} \cdot (b'_{i+28} \cdot b'_{i+60}) = b'_{i+28} \cdot b'_{i+60} [s_i + b_{i+62} + b_{i+52} + b_{i+45} + b_{i+37} + b_{i+33} + \\ b_{i+21} + b_{i+14} + b_{i+9} + b_i + b_{i+37} b_{i+33} + b_{i+15} b_{i+9}] \end{aligned}$$

Now consider the following strategy:
For all derived bits present in the output equation, we will multiply the equation by appropriate variables in view of reducing the degree of the equation. We furnish below some examples which will make our strategy more clear. Consider the 18-th equation (considering $z_i = 0$, similarly for other case) $b_{19} + b_{20} + b_{22} + b_{28} + b_{49} + b_{61} + b_{74} + b_{80} \cdot u(\cdot) + v(\cdot) = 0$

Note that this equation, when expressed in terms of initial NFSR bits, has degree 6. Now consider the following,

$$\begin{aligned} b'_{28}.b'_{60}[b_{19} + b_{20} + b_{22} + b_{28} + b_{49} + b_{61} + b_{74} + u(\cdot)(s_0 + b_{62} + b_{52} + b_{45} + b_{37} \\ + b_{33} + b_{21} + b_{14} + b_9 + b_0 + b_{37} b_{33} + b_{15} b_9) + v(\cdot)] = 0 \end{aligned}$$

Thus we get now an equation of degree 4.

However in order to solve 80 variables in polynomial time by any of the available method, we must have an over determined system i.e. more than 80 equations. In the following, we demonstrate, what will happen when we will apply this strategy to the 80-th equation

$$b_{81} + b_{82} + b_{84} + b_{90} + b_{111} + b_{123} + b_{136} + b_{143} u(\cdot) + v(\cdot) = 0 \tag{2}$$

Now the term $b_{81} + b_{82} + b_{84} + b_{90}$ is of degree at most 6, as the all terms have at most 6^{th} degree expressions. Now for b_{111} term we need to multiply by $b'_{59} b'_{91}$ both sides of the previous equations for b_{111} element. We can find one 4^{th} degree expression for b_{91} by using its low degree multiple. Also in b_{111} there is one term b_{83}, similarly it also has 6^{th} degree expression. So the final degree of the term $b_{81} + b_{82} + b_{84} + b_{90} + b_{111}$ becomes 11. Similarly we can find the degree of the term $b_{81} + b_{82} + b_{84} + b_{90} + b_{111} + b_{123}$ and the degree will be 21 . Now consider the term b_{136}. The low degree multiple of b_{123} is $b'_{84} b'_{116}$. So we need to multiply the whole equation by this low degree multiple $b_{136} b'_{84} b'_{116} = b'_{84} b'_{116} [\textit{ linear part} + b_{93} b_{89} + b_{71} b_{65}]$. Now the degree of the term b'_{84} is 4 by its low degree expression. Also the degree of the term $b_{93} b_{89} + b_{71} b_{65}$ is 8. So we can see that the degree of the equation 2 becomes strictly greater than 33. So the degree of the 80^{th} equation is strictly greater than 33. We need over defined system for 80 unknown variables. Then the degree of the system will even much higher than 33, which is difficult to solve by usual techniques. Thus although this strategy is able to reduce the growth of the degree of the resulting system, but still it has a very high solving complexity. Next we are going to present a new strategy to find low degree equations.

Consider the scenario when half of the NFSR bits are known. In the following we assume that all the bits in the odd positions are known. The argument will be similar for even position bits also. Note that now we have only 40 unknown bits and therefore now we will require much less number of equations, closed to 40. Now as we have mentioned earlier in this case also we have 24 linear equations over the NFSR bits. Now look at the non-linear feedback equation. The distribution of odd and even position bits in the terms of degree ≥ 3 are as follows:

Term	Degree	Odd position	Even position
$b_{i+60}b_{i+52}b_{i+45}$	3	1	2
$b_{i+33}b_{i+28}b_{i+21}$	3	2	1
$b_{i+63}b_{i+45}b_{i+28}b_{i+9}$	4	3	1
$b_{i+60}b_{i+52}b_{i+37}b_{i+33}$	4	2	2
$b_{i+63}b_{i+60}b_{i+21}b_{i+15}$	4	3	1
$b_{i+63}b_{i+60}b_{i+52}b_{i+45}b_{i+37}$	5	3	2
$b_{i+33}b_{i+28}b_{i+21}b_{i+15}b_{i+9}$	5	4	1
$b_{i+52}b_{i+45}b_{i+37}b_{i+33}b_{i+28}b_{i+21}$	6	4	2

Thus when all the odd bits are known and the expression of a derived bit in terms of the NFSR feedback function does not contain any other previously derived bit, the degree of the expression is 2 if it is an even bit and 4 if it is an odd bit. Note that in the non-linear feedback equation, the highest variable present within the non-linear terms is b_{i+63}, next b_{i+60} and next is b_{i+52}. Thus the first 17 derived bits will not have any previously derived bits among the non-linear terms in their expression, the next 3 has only one derived bit and the next 8 has only two. Now as noted earlier after the first 17 linear equations, the next 7 equations have derived bits in only one position namely b_{i+63} and the next 13 in two positions namely b_{i+63} and b_{i+56} and the next 12 equations in three positions namely b_{i+63}, b_{i+56} and b_{i+43}. So from the above discussion it is clear that following the first 17 linear equations we will get 7 equations which will be of degree 2 and degree 4 alternatively, the next 12 equations are of degree 2 or 4 and the next equation is of degree at most 6 and next 8 equations are of degree at most 8. Thus upto this point we have obtained an over defined system of 45 equations in 40 unknown variables of which 17 linear, 19 quadratic or bi-quadratic one equation is of degree 6 and 8 equations of degree 8. We observe that the degree of the equations involving NFSR bits is not increasing so fast after using 40 known values of states, which may be helpful for algebraic attack on the NFSR part.

11 Probabilistic algebraic attack on the LFSR part of Grain-128a

In this section we will discuss the probabilistic algebraic attack on the LFSR part of Grain-128a. The expression of the key-stream bits of Grain-128a has two parts one is the linear combinations of some NFSR bits and one LFSR bit other one is the output of the non-linear function $h(\cdot)$. Precisely $z_i = A_i + h(\cdot)$ (when $IV_0 = 0$, similar study can be done for $IV_0 = 1$), where $A_i = \sum_{k \in A} b_{i+k} + s_{i+93}$ where $A = \{2, 15, 36, 45, 64, 73, 89\}$. For $z_i = 0$ we are getting $A_i + h(\cdot) = 0$, similarly for $z_i = 1$ we will get $1 + A_i + h(\cdot) = 0$. We will discuss the case for $z_i = 0$ when $IV_0 = 0$, similar study can be done for other cases as degree of the LFSR part remains same for all cases. For the operation with authentication we will consider the initial internal state of the cipher to be the one of the clocking when the authentication register has also been initialized. So for $z_i = 0$ we are getting $A_i + x_0x_1 + x_2x_3 + x_4x_5 + x_6x_7 + x_0x_4x_8 = 0$. Where $x_0, x_1, ..., x_8$ correspond to $b_{i+12}, s_{i+8}, s_{i+13}, s_{i+20}, b_{i+95}, s_{i+42}, s_{i+60}, s_{i+79}$ and s_{i+94} respectively. Now multiply the equation by $x'_1 \cdot x'_5 \cdot x'_8$ then we will get $x'_1 \cdot x'_5 \cdot x'_8 \cdot [x_2 \cdot x_3 + x_6 \cdot x_7] + x'_1 \cdot x'_5 \cdot x'_8 \cdot A_i = 0$. Let $u_1(x) = x'_1 \cdot x'_5 \cdot x'_8$ and $u_2(x) = x_2 \cdot x_3 + x_6 \cdot x_7$. i.e. we are getting $u_1(\cdot) \cdot u_2(\cdot) + u_1(\cdot) \cdot A_i = 0$, where $u_1(\cdot), u_2(\cdot)$ are two functions involving only LFSR bits $s_{i+8}, s_{i+13}, s_{i+20}, s_{i+42}, s_{i+60}, s_{i+79}$ and s_{i+94} respectively.

Let's take $X_i = u_1(\cdot) \cdot u_2(\cdot)$ and $Y_i = u_1(\cdot) \cdot A_i$. i.e. for $z_i = 0$ implies $X_i + Y_i = 0$. From this equation we can easily tell that there are only two possible cases for X_i, Y_i, either (i) $X_i = 0, Y_i = 0$

or (ii) $X_i = 1, Y_i = 1$. Let $p = Pr[X_i = 0, Y_i = 0]$ and $q = Pr[X_i = 1, Y_i = 1]$. Now we will choose the case where probability will be high. In fact we will prove that $p > q$ and similarly for other cases ($IV_0 = 1,\ z_i = 1$). Indeed this probability p is quite overwhelming. Thus in this way we get probabilistic system of equations $X_i = u_1(\cdot) \cdot u_2(\cdot)$ and $Y_i = u(\cdot) \cdot A_i$ where $X_i, Y_i \in \{0, 1\}$. Now $X_i = u_1(\cdot) \cdot u_2(\cdot)$ is an equation involving only LFSR bits only. In this way we can construct a probabilistic system of equations with high probability involving only LFSR bits, which we can solve by using any existing algorithm in literatures [5], [13], [9], [10] to obtain the LFSR bits with high probability.

Now we will find the probability $p = Pr[X_i = 0, Y_i = 0]$. Firstly we will assume that the probability of each of the initial state variables being 0 or 1 after key initialization is $\frac{1}{2}$. Moreover as clearly mentioned in [2] that the NFSR feedback function is balanced, so $Pr[\sum_{k\in A} b_{i+k} + s_{i+93} = 0\ or\ 1]$ is $\frac{1}{2}$. From the truth table of $u_1(\cdot)$ and $u_2(\cdot)$ it has been observed that $Pr[X_i = 0] = \frac{61}{64}$. Truth table of these functions are given in the appendix B. Now, $p = Pr[X_i =, Y_i = 0] = Pr[X_i = 0] \cdot Pr[Y_i = 0|X_i = 0] = \frac{61}{64} \cdot Pr[Y_i = 0|X_i = 0]$. Now, we need to calculate $Pr[Y_i = 0|X_i = 0]$. Clearly,
$X_i = 0 \Rightarrow (i)\ u_1(\cdot) = 0,\ u_2(\cdot) = 0;\ \ or\ \ (ii)\ u_1(\cdot) = 0,\ u_2(\cdot) = 1;$
$or\ (iii)\ u_1(\cdot) = 1,\ u_2(\cdot) = 0;$
Also, $Y_i = 0 \Rightarrow (i)\ u_1(\cdot) = 0,\ A_i = 0;\ \ or\ \ (ii)\ u_1(\cdot) = 0,\ A_i = 1;$
$or\ (iii)\ u_1(\cdot) = 1,\ A_i = 0;$

Total number of cases when $u_1 = 0,\ u_2 = 0;\ u_1 = 0,\ u_2 = 1;\ u_1 = 1,\ u_2 = 0$ are 70, 42 and 10 respectively. Under the cases $u_1 = 0,\ u_2 = 0$ and $u_1 = 0,\ u_2 = 1$ A_i can take any values to make $Y_i = 0$. So the total number of favourable cases under above scenario will be $(70 + 42) \times 2^8$. Now under the case $u_1 = 1,\ u_2 = 0$; A_i can take only 0 to make $Y_i = 0$. Hence the total number of cases under this scenario will be 10×2^7 (as A_i is balanced). Hence the total number of favourable cases will be $(70+42)\times 2^8 + 10\times 2^7$. The total number of cases when $X_i = 0$ is 122. Now total number of possible cases will be 122×2^8. Hence the probability $Pr[Y_i = 0|X_i = 0] = \frac{(70+42)\times 2^8 + 10\times 2^7}{122\times 2^8} = \frac{117}{122}$. Hence the required probability $Pr[X_i = 0, Y_i = 0] = \frac{61}{64} \times \frac{117}{122} = 0.914$, which is quite overwhelming.

Hence we see that for $z_i = 0$; $X_i = 0, Y_i = 0$ has the higher probability of occurrence. Now we will choose $X_i = 0$ for $z_i = 0$ to construct a probabilistic system of equations (similarly for $z_i = 1$) involving LFSR bits only, then by using classical algebraic attack technique described in Section 5 we can get the probabilistic LFSR bits after key initialization step of Grain-128a.

Note:- If we take $A_i = \sum_{k\in A} b_{i+k}$ and $u_2 = s_{i+13} \cdot s_{i+20} + s_{i+60} \cdot s_{i+79} + s_{i+93}$ then the above probability $Pr[X_i = 0, Y_i = 0]$ will be 0.906 which is slide lesser than 0.914. For this reason we are not involving s_{i+93} in u_2.

Time complexity for solving the system involving LFSR bits only:- The degree of each equation involving LFSR bits is 4 (after reducing by its low degree multiple). The number of terms of degree 4 of 128 variables is $T = \binom{128}{4}$. We need T number of variables to linearize the system. After the linearization part is over we will solve this system by Gauss elimination method. As the modern system can perform 64 operations in each CPU clock the complexity of solving this system will be $\frac{7\cdot T^{log_2 7}}{64} \approx 2^{63}$ CPU clocks which is less than exhaustive search.

12 Note on the success probability of the attack for the LFSR of Grain family

In this section we analyse the success probability of our attack on LFSR part of Grain family. Let for the LFSR part we require approximately k equations. Thus in effect we need approximately k key-stream bits equations. Thus our probabilistic system of equations have probability approximately $(\frac{1}{2}+p)^k$. Let $p_1 = Pr[Solution\ is\ exact\ solution]$

$$\begin{aligned} p_1 &= Pr[Solution\ is\ exact\ and\ system\ is\ exact] \\ &\qquad\qquad + Pr[Solution\ is\ exact\ and\ system\ is\ not\ exact] \\ &\geq Pr[Solution\ is\ exact\ and\ system\ is\ exact] \\ &= Pr[Solution\ is\ exact|system\ is\ exact]Pr[system\ is\ exact] \\ &= Pr[system\ is\ exact] = \left(\frac{1}{2}+p\right)^k \end{aligned}$$

Hence the solution obtain from it will have probability greater than $\left(\frac{1}{2}+p\right)^k$ of matching with the exact solution. Note that as per the discussion in the previous sections we are able to construct a probabilistic system of algebraic equations involving LFSR bits only from the key-stream bits equations of Grain family with significant probability. Now to find the values of the state bits we need to solve this non-linear system. There are many existing algorithms in literature for solving this kind of over defined non-linear system some of them are XL [5], Gröbner bases [13], [9], [10]. The time complexity for solving this system by linearization method is already discussed in Section 8, 11

13 Conclusion

In this paper we have described a feasible probabilistic algebraic attack on the LFSR part of the Grain family of stream ciphers. Note that Grain v1 and Grain-128a has been designed so as to restrict the classical form of algebraic attack on stream ciphers. This is mainly because of the fact that due to the presence of the NFSR bits in the output function, the degree of the algebraic equations increases rapidly in state of remaining fixed as was the case for classical algebraic attacks on simple LFSR based stream cipher. Our approach in this paper has two significant features. Firstly by our method we are able to separate out the equations involving the LFSR and the NFSR bits for Grain v1 and Grain-128a. Then we are able to recover whole LFSR state of Grain v1 and Grain-128a with significant probabilities (surprisingly quite higher in case of Grain-128a). We have also observed some properties of algebraic equations involving NFSR bits of Grain v1 which may be helpful for algebraic attack. Secondly our approach may be considered as a generic version of probabilistic algebraic attack on stream cipher with similar structure of the output function as in case of Grain family.

References

[1] Afzal, M., Masood, A.: Algebraic cryptanalysis of a nlfsr based stream cipher. In: Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd

International Conference on. pp. 1–6. IEEE (2008)

[2] Ågren, M., Hell, M., Johansson, T., Meier, W.: A new version of grain-128 with authentication. In: Symmetric Key Encryption Workshop (2011)

[3] Banik, S., Maitra, S., Sarkar, S.: A differential fault attack on the grain family of stream ciphers. In: Cryptographic Hardware and Embedded Systems–CHES 2012, pp. 122–139. Springer (2012)

[4] Braeken, A., Preneel, B.: Probabilistic algebraic attacks. In: Cryptography and Coding, pp. 290–303. Springer (2005)

[5] Courtois, N., Klimov, A., Patarin, J., Shamir, A.: Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In: Advances in CryptologyEUROCRYPT 2000. pp. 392–407. Springer (2000)

[6] Courtois, N., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. Advances in CryptologyEUROCRYPT 2003 pp. 644–644 (2003)

[7] Crama, Y., Hammer, P.L.: Boolean models and methods in mathematics, computer science, and engineering (2010)

[8] Cusick, T.W., Stănică, P.: Cryptographic Boolean functions and applications. Academic Press (2009)

[9] Faugre, J.C.: A new efficient algorithm for computing Gröbner bases (F4). Journal of Pure and Applied Algebra 139(1–3), 61–88 (June 1999), http://www-salsa.lip6.fr/~jcf/Papers/F99a.pdf

[10] Faugre, J.C.: A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In: Proceedings of the 2002 international symposium on Symbolic and algebraic computation. pp. 75–83. ISSAC '02, ACM, New York, NY, USA (2002), http://www-salsa.lip6.fr/~jcf/Papers/F02a.pdf

[11] Hell, M., Johansson, T., Meier, W.: Grain: a stream cipher for constrained environments. International Journal of Wireless and Mobile Computing 2(1), 86–93 (2007)

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

[13] Segers, A.: Algebraic attacks from a gröbner basis perspective. Master's Thesis (2004)

[14] Zhang, H., Wang, X.: Cryptanalysis of stream cipher grain family. IACR E-print Archiv, Report 109 (2009)

A Truth Table of $u'(\cdot),\ v'(\cdot)$

Truth table of the functions $u'(\cdot), v(\cdot)$ and $u'(\cdot)v(\cdot)$ where $u'(x_0, x_1, x_2, x_3) = x_3 + x_0x_2 + x_1x_2 + x_2x_3,\ v(x_0, x_1, x_2, x_3) = x_1 + x_0x_3 + x_2x_3 + x_0x_1x_2 + x_0x_2x_3$ and $u'(\cdot)v(\cdot) = x_0x_3 + x_1x_3 + x_1x_2 + x_0x_1x_2$ is given by,

$$u'(\cdot) \Rightarrow 0\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0, \quad v(\cdot) \Rightarrow 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 1,$$
$$u'(\cdot)v(\cdot) \Rightarrow 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0$$

B Truth Table of $u_1(\cdot)$, $u_2(\cdot)$

Truth table of the functions $u_1(x_0, x_1, ..., x_6)$ and $u_2(x_0, x_1, ..., x_6)$ are given below. Where $u_1(x_0, x_1, ..., x_6) = x'_0 \cdot x'_3 \cdot x'_6$ and $u_2(x_0, x_1, ..., x_6) = x_1 \cdot x_2 + x_4 \cdot x_5$. Truth table of $u_1(\cdot)$ is given by,

1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0
0 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Truth table of $u_2(\cdot)$ is given by,

0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1
1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0
0 0 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0