A Guess-Then-Algebraic Attack on LFSR-Based Stream Ciphers with Nonlinear Filter

Xiao Zhong[1,2](✉), Mingsheng Wang[3], Bin Zhang[1,4], and Shengbao Wu[1,2]

[1] Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing, China
zhongxiao456@163.com,
{zhangbin,wushengbao}@tca.iscas.ac.cn

[2] Graduate School of Chinese Academy of Sciences, Beijing, China

[3] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
mingsheng_wang@aliyun.com

[4] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China

Abstract. This paper introduces a new parameter called 1-st minimum algebraic immunity $AI_{min}(f)$, along with some theoretical properties of $AI_{min}(f)$. Based on our results, we propose a guess-then-algebraic attack on LFSR-based stream ciphers with nonlinear filter Boolean function f. Our method takes some particular guessing strategy by taking advantage of the properties of $AI_{min}(f)$, and makes full use of each guessed bit to generate as many equations of degree $AI_{min}(f)$ as possible. The cost of time and data is less than that of the traditional algebraic attack under some condition. Our attack suggests that $AI_{min}(f)$ should not be too small, which is a new design criterion for the filter Boolean function f. We apply our method to a 80-stage keystream generator with a MAI Boolean function f as its filter, while $AI_{min}(f)$ is very small, which disobeys our criterion. The time and data complexities of the traditional algebraic attack are $O(2^{73.56})$ and $O(2^{24.52})$ respectively, with a memory cost of $O(2^{49})$. The time and data complexity of our method are $O(2^{56.86})$ and $O(2^{5.36})$ respectively, and the memory cost is $O(2^{10.72})$.

Keywords: Algebraic attack · Algebraic immunity · Guess-then-algebraic attack · LFSR-based stream ciphers · 1-st minimum algebraic immunity

1 Introduction

The research of the LFSR-based stream ciphers with nonlinear filter generators or combination generators has spawned many analytical methods and theoretical research. A most important cryptanalytic tool is algebraic attack.

Courtois, N.T. and Meier, W. proposed algebraic attack on stream ciphers with linear feedback in 2003 [3], which is a classical and efficient analytical method towards LFSR-based stream ciphers. The basic idea of the algebraic attack is divided into three steps:

Step 1: Construct a nonlinear equation system $\mathcal{A}$ with the secret key bits or initial vector values as its variables by using the keystream bits.

Step 2: Multiply a nonzero function g of low degree to each equation of the system, resulted to an equation system $\mathcal{B}$ with degree lower than that of $\mathcal{A}$.

Step 3: Solve the equation system $\mathcal{B}$.

Since algebraic attack was proposed, there have emerged many research issues related to it. Scholars try to explore the properties of the nonzero function g in Step 2, resulting to the research areas of algebraic immunity and annihilators for the Boolean functions [1,2,4,6,8,10,12,13,15,16]. Denote $AN(f) = \{g \in B_n | f \cdot g = 0\}$. Any function $g \in AN(f)$ is called an annihilator of f. The algebraic immunity of f, named $AI(f)$, is the minimum algebraic degree of nonzero annihilators of f or $f + 1$ [11], which measures the ability of the Boolean functions against algebraic attack. The maximum algebraic immunity for n-variable Boolean functions is $\lceil \frac{n}{2} \rceil$ [3], and a Boolean function with maximum algebraic immunity is called MAI Boolean function.

In this paper, for a Boolean function $f \in B_n$ (where B_n is the n-variable Boolean ring), we introduce a parameter called 1-st minimum algebraic immunity $AI_{min}(f)$. We prove that when $AI(f)$ is optimum, $AI_{min}(f) \leq AI(f)$, particularly, when n is odd, $AI_{min}(f) < AI(f)$. A simple algorithm is given to compute $AI_{min}(f)$.

When the filter of a LFSR-based stream cipher is a MAI Boolean function f, it can resist the algebraic attack to the greatest extent. Based on our theoretical results related to $AI_{min}(f)$, we propose a guess-then-algebraic attack on such kind of LFSR-based stream ciphers. The strategy of guessing is very important in the guess and determine attack. Our method takes some particular guessing strategy by taking advantage of the properties of $AI_{min}(f)$. First, we guess some initial state bits, the locations of which are determined by the specific structure of the LFSR and the properties of $AI_{min}(f)$. After guessing enough LFSR state bits, we derive equations of degree $AI_{min}(f)$ as many as possible by taking advantage of the specific structure of the LFSR and the filter taps, and then we solve the resulted equation system.

Our method has three special merits. (a) It makes full use of each guessed bit to derive equations of degree $AI_{min}(f)$ as many as possible. In fact, $AI_{min}(f)$ is strictly less than $AI(f)$ in most cases. (b) It utilizes not only the properties of f but also the tap positions, which helps to derive as much information as possible. The cost of time and data is less than that of the traditional algebraic attack given in [3] under some condition. (c) Our attack suggests that $AI_{min}(f)$ should not be too small, which is a new design criterion for the filter Boolean function f.

Moreover, we apply our method on a 80-stage keystream generator. Its filter generator is a 9-variable MAI Boolean function f, while $AI_{min}(f)$ is very small, which disobeys our design criterion. This model can resist the traditional algebraic attack given in [3] to the greatest extent. The time and data complexities of the traditional algebraic attack are $O(2^{73.56})$ and $O(2^{24.52})$ respectively, with a memory cost of $O(2^{49})$. While the time and data complexity of our method are $O(2^{56.86})$ and $O(2^{5.36})$ respectively, and the memory cost is $O(2^{10.72})$.

Notice that Debraize, B. and Goubin, L. proposed a guess-and-determine algebraic attack on the self-shrinking generator (SSG) with low Hamming weight feedback polynomial in [7]. It mainly aims to analyze the self-shrinking generators, while our method targets to the LFSR-based stream ciphers with filter Boolean functions of optimum algebraic immunity. They guess some information first, then write a system of polynomial equations and solve the system with SAT solver algorithm MiniSAT. While our method pays attention to the locations of the guessed bits by taking advantage of the theoretical properties of $AI_{min}(f)$, which is a new parameter in the academic sector of the stream cipher. Moreover, the degree of the equations derived after guessing some state bits is $AI_{min}(f)$, which is the most different character between our method and the guess-and-determine algebraic attack proposed in [7].

This paper is organized as follows: Sect. 2 introduces some basic knowledge related to our work. In Sect. 3, we introduce a new parameter called 1-st minimum algebraic immunity $AI_{min}(f)$, along with some theoretical analysis. We also give a simple algorithm to compute $AI_{min}(f)$. In Sect. 4, for LFSR-based keystream generators with nonlinear filter Boolean function, we propose a guess-then-algebraic attack by taking advantage of $AI_{min}(f)$ and give some design criterion of the LFSR-based keystream generators. In Sect. 5, we apply our method to a keystream generator which does not obey our criterion. Section 6 concludes this paper.

2 Preliminaries

Courtois, N.T. and Meier, W. proposed the algebraic attack on stream ciphers with linear feedback [3]. They mainly focused on the LFSR-based keystream generators with nonlinear filter Boolean function. Figure 1 shows the general model.

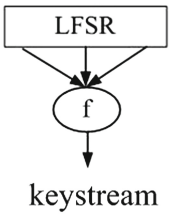

Fig. 1. LFSR-based keystream generator with nonlinear filter

First, we give a brief description for this model. Let the length of the linear feedback shift register be l. L is the "connection function" of the LFSR, and it is linear. Let the initial state of the LFSR be $s^0 = (s_0, s_1, ..., s_{l-1})$, then it generates a m-sequence $s_0, s_1, s_2....$ For sake of narrative convenience, we call this m-sequence as LFSR sequence. The state of the LFSR at time t is

$$s^t = (s_t, s_{t+1}, ..., s_{t+l-1}) = L^t(s_0, s_1, ..., s_{l-1}),$$

Table 1. Complexity of AA for the model in Fig. 1

Data	Memory	Complexity
$O(M)$	$O(M^2)$	$O(M^\omega)$

which is filtered by a balanced nonlinear Boolean function $f \in B_n$. The generator outputs one bit c_t at time t. For each c_t, we can construct an equation involving some key bits and initial value as its variables. Denote the output of the filter generator by $c_0, c_1, c_2, ...,$ where $c_i \in F_2$, then we can get the following equation system:

$$\begin{cases} c_0 = f \quad\ \ (s_0, s_1, ..., s_{l-1}) \\ c_1 = f(L\ (s_0, s_1, ..., s_{l-1})) \\ c_2 = f(L^2(s_0, s_1, ..., s_{l-1})) \\ \vdots \end{cases} \tag{1}$$

The problem of recovering the l initial state bits of the LFSR is reduced to solving the equation system (1).

Table 1 shows the complexity of the traditional algebraic attack (AA) in [3] on the model given in Fig. 1, where $M = \binom{l}{AI(f)}$, and ω is the parameter of the Gaussian elimination and in theory $\omega \leq 2.376$ [5].

While as the authors of [3] declare, the (neglected) constant factor in that algorithm is expected to be very big and they regard Strassen's algorithm [14] as the fastest practical algorithm. Then they evaluate the complexity of the Gaussian reduction to be $7 \cdot M^{log_2^7}/64$ CPU clocks. For convenience, scholars usually use $\omega = 3$ in Table 1 to estimate the time and data complexity of the traditional algebraic attack.

We can get that the complexity of the traditional algebraic attack on the model given in Fig. 1 is determined by l and $AI(f)$. When using the traditional algebraic attack given in [3], for each keystream bit, the lowest degree of equations that the analysts can obtain is $AI(f)$. In the next section, we try to find a way to see if we can decrease the time and data complexity by further exploring the properties of the nonlinear filter function and the structure of the filter tap positions.

3 1-st Minimum Algebraic Immunity $AI_{min}(f)$

In this section, for a Boolean function $f \in B_n$, we give a new definition called 1-st minimum algebraic immunity $AI_{min}(f)$. To begin with, we give the following definition.

Definition 1. *Given a Boolean function* $f(x_1, x_2, ..., x_n) \in B_n$*, for some fixed* i*,* $i \in [1, n]$*, define*

$$AI(f|_{x_i=0}) = AI(f(x_1, x_2, ..., x_{i-1}, 0, x_{i+1}, ..., x_n)).$$

$$AI(f|_{x_i=1}) = AI(f(x_1, x_2, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)).$$

Definition 2. *Given a Boolean function $f(x_1, x_2, ..., x_n) \in B_n$, for some fixed i, $i \in [1, n]$, define*

$$AI_i(f) = max\{AI(f|_{x_i=0}), AI(f|_{x_i=1})\}.$$

Definition 3. *For a Boolean function $f(x_1, x_2, ..., x_n) \in B_n$, define the 1-st minimum algebraic immunity of f as*

$$AI_{min}(f) = min\{AI_i(f) : i \in [1, n]\}.$$

Also define

$$N_{AI_{min}}(f) = \sharp\{i|AI_i(f) = AI_{min}(f) : i \in [1, n]\},$$

where "$\sharp$" denote the number of the elements in a set.
Let $G = \{g \in AN(f|_{x_i=c})|deg(g) = AI_{min}(f)\}$, and also denote $n_{i,c} = \sharp G$, where $c \in \{0, 1\}$.

Based on the above definitions, we derive Theorem 1:

Theorem 1. *For a Boolean function $f(x_1, x_2, ..., x_n) \in B_n$, if its algebraic immunity is optimum, then*

$$AI_{min}(f) \leq AI(f).$$

Particularly, if n is odd, then

$$AI_{min}(f) < AI(f).$$

Proof. Given $f(x_1, x_2, ..., x_n) \in B_n$, it can be expressed as:

$$f(x_1, x_2, ..., x_n) = x_i f_1(x_1, ..., x_{i-1}, x_{i+1}, ..., x_n) + f_2(x_1, ..., x_{i-1}, x_{i+1}, ..., x_n).$$

When x_i is fixed, there are only $n - 1$ variables left, then from Definitions 1–3, we can get that

$$AI_{min}(f) \leq AI_i(f) \leq \lceil \frac{n-1}{2} \rceil \leq AI(f).$$

Particularly, if n is odd, then

$$AI_{min}(f) \leq AI_i(f) \leq \lceil \frac{n-1}{2} \rceil < \lceil \frac{n}{2} \rceil = AI(f).$$

The following example verifies Theorem 1.

Example 1. The Boolean function $f = x_1x_2x_3x_5 + x_1x_2x_5 + x_1x_2 + x_1x_3x_4x_5 + x_1x_3x_4 + x_1x_3x_5 + x_1x_4x_5 + x_1x_4 + x_2x_3 + x_2x_4x_5 + x_2x_5 + x_3x_4 + x_4x_5 + 1$ is a 5-variable Carlet-Feng Boolean function. We can get that $AI(f) = 3$, which is optimum. The $AI_i(f), i \in [1, 5]$ of f are listed in Table 2.

We can see that $AI_{min}(f) = AI_i(f) = 2 < AI(f) = 3$, $N_{AI_{min}}(f) = 5 > 1$.

In fact, for a Boolean function $h \in B_n$, although $AI(h)$ is not maximum, it is possible that $AI_{min}(h)$ is strictly less than $AI(h)$. For instance, for the filter function f_d adopted by the stream cipher LILI-128 [9], $AI_{min}(f_d) = 3 < AI(f_d) = 4$, $N_{AI_{min}}(f_d) = 4$.

Table 2. Compute the $AI_i(f)$ of f

x_i	x_1	x_2	x_3	x_4	x_5
$AI_i(f)$	2	2	2	2	2

Algorithm 1 shows how to compute $AI_{min}(f)$ and $N_{AI_{min}}(f)$ for a Boolean function f.

Algorithm 1. Compute $AI_{min}(f)$ and $N_{AI_{min}}(f)$

```
Input: $I = \{1, 2, ..., n\}$, Boolean function $f \in B_n$.
Set $F = \emptyset$, $E = \emptyset$, $c \in \{0, 1\}$.
for $i \in I$ do
    Compute all the $AI_i(f)$ defined in Definition 2 and the corresponding $n_{i,c}$
    defined in Definition 3;
    if $AI_i(f) \leq AI(f)$ then
        $F = F \cup \{(x_i, AI_i(f), n_{i,0}, n_{i,1})\}$;
Denote the minimum $AI_i(f)$ in $F$ as $AI_{min}(f)$;
$E = E \cup \{(x_i, AI_i(f), n_{i,0}, n_{i,1}) \in D | AI_i(f) = AI_{min}(f), i \in I\}$;
$N_{AI_{min}}(f) = \sharp E$;
Output $E$, $N_{AI_{min}}(f)$.
```

4 A Guess-Then-Algebraic Attack on LFSR-Based Stream Ciphers via Our Theoretical Results

Designers often choose a MAI Boolean function as the filter of the LFSR-based keystream generators, which helps to resist the traditional algebraic attack to the greatest extent. Section 3 shows that for a Boolean function f of optimum algebraic immunity, $AI_{min}(f) \leq AI(f)$. Even if $AI(f)$ is not maximum, it is possible that $AI_{min}(f) < AI(f)$. We consider to take advantage of the properties of $AI_{min}(f)$ to recover the initial state of the LFSR.

The strategy of guessing is very important in the guess and determine attack. In this section, we would like to give a guess-then-algebraic attack, which takes some particular guessing strategy by taking advantage of the properties of $AI_{min}(f)$ proposed in Sect. 4. First we choose to guess some (initial) internal state bits of the LFSR, resulted to equations of degree $AI_{min}(f)$. Here we make use of each guessed bit to the greatest extent, that is, we would use each guessed bit to construct as many low-degree equations as possible. The positions of the guessed internal state bits of the LFSR should obey some rules according to the detailed structure of the LFSR, the properties of the filter Boolean function, and the filter tap positions. After guessing a suitable number of (initial) internal state bits, we can get an equation system of degree $AI_{min}(f)$.

In the following, we propose an attack by using our theoretical results in Sect. 3. We focus on the LFSR-based keystream generator with nonlinear filter shown in Fig. 1. With the same description in Sect. 2, we target to recover the initial state bits $s^0 = (s_0, s_1, ..., s_{l-1})$ by solving the following equation system:

$$\begin{cases} c_0 = f(s_0^1, s_0^2, ..., s_0^n) \\ c_1 = f(s_1^1, s_1^2, ..., s_1^n) \\ c_2 = f(s_2^1, s_2^2, ..., s_2^n) \\ \vdots \end{cases} \tag{2}$$

where $s_t^i, i = 1, 2, ..., n$ are the n LFSR state bits tapped as the input of the filter Boolean function $f(x_1, x_2, ..., x_n)$ at time t.

Guess-then-algebraic Attack:
Assume that the LFSR shifts right and it is regularly clocked.
Step I: For $f(x_1, x_2, ..., x_n) \in B_n$, compute $AI_{min}(f)$, E and $N_{AI_{min}}(f)$ via Algorithm 1. Suppose we get $E = \{(x_{i_1}, AI_{min}(f), n_{i_1,0}, n_{i_1,1}), ..., (x_{i_m}, AI_{min}(f), n_{i_m,0}, n_{i_m,1}) : 1 \leq i_1 < i_2 < \cdots < i_m \leq n\}$, $N_{AI_{min}}(f) = m$, $AI_{min}(f) = k$.
Step II: With the parameters derived from Step I, we do the following operation:
(1) At time t, denote the inputs of the filter function as $s_t^1, ..., s_t^n$, find the LFSR state bit corresponding to x_{i_1} in set E, and denote it as $s_t^{i_1}$.
(2) Guess $s_t^{i_1} = a$, where $a \in \{0, 1\}$, then we can derive an equation

$$f_t(s_t^1, s_t^2, ..., s_t^{i_1-1}, a, s_t^{i_1+1}, ..., s_t^n) = c_t.$$

From Algorithm 1 we get that $AI(f_t) = AI_{min}(f) = k$.
(3) For each clock, denote the distance of the locations for the LFSR state bits corresponding to the variables x_i and x_j of f as $d_{i,j}$.
Notice that $E = \{(x_{i_1}, AI_{min}(f), n_{i_1,0}, n_{i_1,1}), ..., (x_{i_m}, AI_{min}(f), n_{i_m,0}, n_{i_m,1}) : 1 \leq i_1 < i_2 < \cdots < i_m \leq n\}$, then we clock the stream cipher d_{i_1,i_2} clocks from time t. Find the LFSR state bit corresponding to x_{i_2} in the set E at time $t+d_{i_1,i_2}$, and denote it as $s_{t+d_{i_1,i_2}}^{i_2}$. With the keystream bit $c_{t+d_{i_1,i_2}}$ and the guessed value of $s_t^{i_1} = a$, we get another equation at time $t + d_{i_1,i_2}$ by substitute $s_{t+d_{i_1,i_2}}^{i_2}$ by a.

$$f_{t+d_{i_1,i_2}}(s_{t+d_{i_1,i_2}}^1, s_{t+d_{i_1,i_2}}^2, ..., s_{t+d_{i_1,i_2}}^{i_2-1}, a, s_{t+d_{i_1,i_2}}^{i_2+1}, ..., s_t^n) = c_{t+d_{i_1,i_2}}.$$

In the same way, we can get a group of m equations, and the algebraic immunity of the functions $f_t, f_{t+d_{i_1,i_2}}, ..., f_{t+d_{i_1,i_m}}$ is $AI_{min}(f) = k$.

$$\begin{cases} f_t(s_t^1, ..., s_t^{i_1-1}, a, s_t^{i_1+1}, ..., s_t^n) = c_t \\ f_{t+d_{i_1,i_2}}(s_{t+d_{i_1,i_2}}^1, ..., s_{t+d_{i_1,i_2}}^{i_2-1}, a, s_{t+d_{i_1,i_2}}^{i_2+1}, ..., s_{t+d_{i_1,i_2}}^n) = c_{t+d_{i_1,i_2}} \\ \vdots \\ f_{t+d_{i_1,i_m}}(s_{t+d_{i_1,i_m}}^1, ..., s_{t+d_{i_1,i_m}}^{i_m-1}, a, s_{t+d_{i_1,i_m}}^{i_m+1}, ..., s_{t+d_{i_1,i_m}}^n) = c_{t+d_{i_1,i_m}} \end{cases} \tag{3}$$

Until now, we get m equations by guessing one LFSR state bit. For each equation, we can derive $n_{i_j,c_{t+d_{i_j,i_k}}}$ (or $n_{i_j,c_{t+d_{i_j,i_k}}+1}$ when $c_{t+d_{i_j,i_k}} = 0$) equations of degree $AI_{min}(f)$, where $n_{i_j,c_{t+d_{i_j,i_k}}}$ is the number of annihilators defined in

Definition 3. We guess another LFSR state bit which is appropriately chosen, and get another group of equations, and so on.

We hope that the guessed bits are all the initial state bits of the LFSR, which can make the analysis much easier. If the guessed bits are not all the initial state bits, more careful analysis is needed.

Usually, we choose the parameter $t = 0$, for it can make the number of the guessed initial bits among all the guessed LFSR state bits as large as possible, which can make the analysis much easier.

Here we analyze the situation that the guessed bits are all the initial state bits of the LFSR. Suppose we guess r initial LFSR state bits, then we can get $r \cdot \sum_{1\leq j<k\leq m} n_{i_j,c_{t+d_{i_j,i_k}}}$ equations with the initial LFSR state bits as the variables by using the linear feedback recursion of the LFSR. Then we use the same method to analyze the complexity with the method mentioned in [3].

After guessing r LFSR initial state bits, we reduce the problem of solving the l initial LFSR state bits with a filter Boolean function f of algebraic immunity $AI(f)$ to solving $l-r$ unknown initial bits with a filter Boolean function of algebraic immunity $AI_{min}(f)$.

When the condition

$$r \cdot \sum_{1\leq j<k\leq m} n_{i_j,c_{t+d_{i_j,i_k}}} \geq \binom{l-r}{AI_{min}(f)} \tag{4}$$

is satisfied, the time complexity to solve the equation system derived from Step II is

$$T_1 = N^{\omega},$$

where $N = \binom{l-r}{AI_{min}(f)}$, and ω is the parameter of the Gaussian elimination, and we adopt $\omega = 3$ in this paper. The complexity to recover the initial state of the LFSR is

$$T = 2^r N^{\omega}.$$

The data that we need is

$$D = max(rm, \binom{l-r}{AI_{min}(f)}).$$

The key condition is that the inequality (4) is satisfied. In fact, when $AI_{min}(f)$ is small enough (especially when $AI_{min}(f) = 1$), the condition can be satisfied in most cases. Under this situation, we can directly see that the data complexity is better than that of the algebraic attack in [3]. While the improvement of the time complexity is not determined. It can be derived that the time complexity of our attack is less than that of the conventional algebraic attack given in [3] when r satisfies the following inequality:

$$\frac{r}{w} < log_2\binom{l}{d} - log_2\binom{l-r}{k}, \tag{5}$$

where r is the number of the guessed initial state bits, $d = AI(f)$, $k = AI_{min}(f)$, and ω is the parameter of the Gaussian elimination.

In fact, the number of the equations that can be derived by guessing r-bit initial state can be more than $r \cdot \sum_{1\leq j<k\leq m} n_{i_j, c_{t+d_{i_j}, i_k}}$ if we can make full use of the annihilators for $f|_{x_i=c}$. Let $G = \{g \in AN(f|_{x_i=c}) | deg(g) = AI_{min}(f)\}$, for an element $g \in G$, multiply monomials of degree less than $AI_{min}(f)$ to g, and we may get new polynomials that can be used to construct equations.

Our method suggests a new design criterion for the filter Boolean function f adopted by the LFSR-based stream ciphers, that is, the parameters of the keystream generator should not satisfy the inequality (4) and inequality (5) in the same time, which means that designers should pay attention that their keystream generator should satisfy that:

(a)$AI_{min}(f)$ should be large enough to resist our guess-then-algebraic attack.
(b)The number of variables corresponding to $AI_{min}(f)$ should not be too large.

Notice that our method can be applied to all kinds of LFSR-based stream ciphers with nonlinear filter. If the target stream cipher satisfies the inequalities (4) and (5), then we can decrease the time and data complexity.

5 Application of Our Method on a LFSR-Based Stream Cipher Overlooking Our Criterion

In this section, we would like to give an example to show that the $AI_{min}(f)$ should not be too small for the nonlinear filter Boolean function.

The target model is the same with the one shown in Fig. 1. The length of the LFSR is 80, and its initial state bits are $(s_0, s_1, ..., s_{79})$. The linear feedback polynomial is primitive. The filter Boolean function is a 9-variable Boolean function $f = f(x_1, x_2, ..., x_9)$, $AI(f)$ is optimum. Suppose the LFSR shifts left and it is regularly clocked. The 9 inputs to f are taken from LFSR according to this full positive difference set: (0,1,3,7,12,20,30,44,65).

From the above description we can get that this model can resist the traditional algebraic attack given in [3] to the greatest extent. In the following, we would show that although $AI(f)$ is optimum, our method works if the model disobeys our criterion mentioned in Sect. 4.

Suppose $AI_{min}(f) = 1$, and the corresponding variable is x_7, that is, $AI(f|_{x_7=0}) = AI(f|_{x_7=1}) = 1$, which means that this model disobeys our criterion.

According to the guess-then-algebraic attack given in Sect. 4, we can guess $r = 41$ initial state bits $(s_{38}, s_{31}, ..., s_{79})$. Then we can get at least 41 linear equations, which satisfy the inequality (4), that is $r \geq \binom{80-r}{AI_{min}(f)} = 39$. The time complexity of our attack is

$$T = 2^r \binom{80-r}{1}^{\omega} = 2^{56.86},$$

where m is the number of variables for f corresponding to $AI_{min}(f)$.
The data complexity is

$$D = max(r, \binom{80-r}{1} = 2^{5.36}.$$

The memory complexity is

$$M = 2^{10.72}.$$

Table 3 shows the comparison between our method (GA) and the traditional attack (AA) given in [3] on this model.

Table 3. Comparison between GA and AA in [3]

	T	D	M
Our method	$O(2^{56.86})$	$O(2^{5.36})$	$O(2^{10.72})$
Algebraic attack	$O(2^{73.56})$	$O(2^{24.52})$	$O(2^{49})$

Remark 1. This example verifies that although the target model obsesses a filter Boolean function of optimum algebraic immunity, if it disobeys our criterion, analysts can still use our guess-then-algebraic attack to recover the initial state with time and data complexities less than that of the traditional algebraic attack given in [3].

6 Conclusion

This paper introduces a new parameter called 1-st minimum algebraic immunity $AI_{min}(f)$ for the Boolean function f, along with some theoretical properties of $AI_{min}(f)$. Based on our results, we propose a guess-then-algebraic attack on the LFSR-based stream cipher with nonlinear filter Boolean function f by guessing some state bits in advance and then applying algebraic attack on it. Our method makes full use of each guessed bit to generate as many equations of degree less than or equal to $AI(f)$ as possible. The cost of time and data is less than that of the traditional algebraic attack in [3] under some condition. Our method suggests a new design criterion for the LFSR-based keystream generators with nonlinear filter. We apply our method to a 80-stage keystream generator with a MAI Boolean function f as its filter, while $AI_{min}(f)$ is very small, which disobeys our criterion. The time, memory and data complexities of our method are less than that of the traditional algebraic attack.

Acknowledgements. We are grateful to the anonymous reviewers for their valuable comments on this paper. This work was supported by the National Basic Research Program of China (Grant No. 2013CB834203, Grant No. 2013CB338002) and the National Natural Science Foundation of China (Grant No. 61379142, Grant No. 11171323, Grant No. 60833008, Grant No. 60603018, Grant No. 61173134, Grant No. 91118006, Grant No. 61272476), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDA06010701).

References

1. Armknecht, F., Carlet, C., Gaborit, P., Künzli, S., Meier, W., Ruatta, O.: Efficient computation of algebraic immunity for algebraic and fast algebraic attacks. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 147–164. Springer, Heidelberg (2006)
2. Carlet, C., Feng, K.: An infinite class of balanced functions with optimal algebraic immunity, good immunity to fast algebraic attacks and good nonlinearity. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 425–440. Springer, Heidelberg (2008)
3. Courtois, N.T., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 345–359. Springer, Heidelberg (2003)
4. Carlet, C., Zeng, X.Y., Li, C.L., Hu, L.: Further properties of several classes of Boolean functions with optimum algebraic immunity. Des. Codes Crypt. (DCC) **52**(3), 303–338 (2009)
5. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. J. Symb. Comput. **9**, 251–280 (1990)
6. Tang, D., Carlet, C., Tang, X.H.: Highly nonlinear boolean functions with optimal algebraic immunity and good behavior against fast algebraic attacks. IEEE Trans. Inf. Theor. (TIT) **59**(1), 653–664 (2013)
7. Debraize, B., Goubin, L.: Guess-and-determine algebraic attack on the self-shrinking generator. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 235–252. Springer, Heidelberg (2008)
8. Dalai, D.K., Maitra, S., Sarkar, S.: Basic theory in construction of Boolean functions with maximum possible annihilator immunity. Des. Codes Crypt. **40**(1), 41–58 (2006)
9. Dawson, E., Golić, J.D., Millan, W., Simpson, L.: The LILI-II keystream generator. In: Batten, L., Seberry, J. (eds.) ACISP 2002. LNCS, vol. 2384, pp. 25–39. Springer, Heidelberg (2002)
10. Limniotis, K., Kolokotronis, N., Kalouptsidis, N.: Secondary constructions of Boolean functions with maximum algebraic immunity. Crypt. Commun. (CCDS) **5**(3), 179–199 (2013)
11. Meier, W., Pasalic, E., Carlet, C.: Algebraic attacks and decomposition of Boolean functions. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 474–491. Springer, Heidelberg (2004)
12. Peng, J., Wu, Q.S., Kan, H.B.: On symmetric Boolean functions with high algebraic immunity on even number of variables. IEEE Trans. Inf. Theor. **57**(10), 7205–7220 (2011)
13. Rizomiliotis, P.: On the security of the Feng-Liao-Yang Boolean functions with optimal algebraic immunity against fast algebraic attacks. Des. Codes Crypt. (DCC) **57**(3), 283–292 (2010)
14. Strassen, V.: Gaussian elimination is not optimal. Numer. Math. **13**, 354–356 (1969)
15. Zeng, X., Carlet, C., Shan, J., Hu, L.: More balanced Boolean functions with optimal algebraic immunity and good nonlinearity and resistance to fast algebraic attacks. IEEE Trans. Inf. Theor. **57**(9), 6310–6320 (2011)
16. Chen, Y.D., Lu, P.Z.: Two classes of symmetric Boolean functions with optimum algebraic immunity: construction and analysis. IEEE Trans. Inf. Theor. **57**(4), 2522–2538 (2011)