A different algebraic analysis on ZUC stream cipher

Mufeed AlMashrafi
Information Security Institute
Queensland University of Technology
GPO Box 2434, Brisbane Qld 4001, Australia
almashrafi@student.qut.edu.au

ABSTRACT

Preliminary algebraic analysis of the ZUC cipher indicates that the cipher may be vulnerable to algebraic attack. In this paper we present an alternative algebraic analysis method for the ZUC stream cipher, using a combiner to represent the nonlinear function. This approach can recover the initial state of ZUC from an observed 2^{97} words of keystream, with a complexity of 2^{282} operations. This method is particularly successful when applied to a modified version of ZUC, where the number of output words per clock is increased.

Keywords

Algebraic Attack, Stream Ciphers, ZUC, 128-EEA3, 128-EIA3, Multivariate Equations.

1. INTRODUCTION

The ZUC stream cipher [8] is designed for use in China as a standard for fourth generation mobile phones. It is a word based stream cipher with a word size of 32-bits. The keystream generator is based on a single 16-stage register, where each stage contains 31 bits. The inputs to the cipher are a key (k) and Initialization Vector (IV), each of size 128-bits (4 words), and the output is a 32-bit keystream word. The structure of the ZUC keystream generator consists of three layers: a linear feedback shift register (LFSR), a bit reorganization layer and a nonlinear function f. The second and third layers can be considered as a nonlinear filter applied to the LFSR. Besides generating keystream to be used for encryption, the keystream of ZUC can also be used to generate a 32-bit Message Authentication Code (MAC). The algorithms making use of the ZUC keystreams for encryption and for authentication are referred to as 128-EEA3 and 128-EIA3, respectively [7, 9].

Algebraic analyses of ZUC are given in [9, 1, 11]. The algebraic approach taken in these papers is straightforward and involves finding direct relationships between the internal state bits of ZUC and the observed keystream bits. In this paper, we take a different approach and establish relationships between the internal state bits and the keystream bits indirectly. This analysis involves considering the nonlinear filter as a combiner with four input words and two output words: two input words from the LFSR, and two inputs words from the internal memory states. The algebraic attack based on these relationships requires 2^{97} words of observed keystream and 2^{282} operations to recover the initial state bits of the shift register.

The paper is organized as follows. A brief description of the algebraic attack is presented in Section 2. Section 3 describes ZUC stream cipher. The existing algebraic attack analysis on ZUC is described in Section 4. Our algebraic attack analysis of ZUC is given in Section 5. Section 6 concludes this paper.

2. ALGEBRAIC ATTACK

Algebraic attacks on symmetric ciphers were first applied to block ciphers in 2002 [6] and then to stream ciphers in 2003 [5]. On stream ciphers, the attacks involve two main stages [4]. In the first stage, the analyst establishes multivariate relations of low degree between either the key or the internal state values and the output keystream. In the second stage, the observed keystream is used to replace the variables denoting the keystream outputs $z_0, z_1, ...$ to obtain a system of multivariate equations. Finally, solving this system of equations permits recovery of the key or internal state values, respectively. The complexity of solving the system of equations depends on the amount of keystream required, which is determined by the number of equations generated.

For ease of analysis, algebraic attacks are often performed by describing the cipher state and keystream in terms of bit values, regardless of whether the cipher is bit-based or word-based. The cipher functions, at the bit level, are described in terms of the XOR and AND bit operations (addition and multiplication over GF(2) respectively). This approach has been applied in this paper.

The relationship between either key bits or internal state bits and the output bits can be obtained directly or indirectly for the nonlinear components that generate the keystream output. Direct relationships are obtained by building mathematical models of the nonlinear components, where the output bits are presented as functions of the input bits. Indirect relationships are obtained by considering the input and output bits of the nonlinear components together and finding valid relationships among them. In either case, the output of this first stage is a system of multivariate equations of a certain degree.

In the second stage, solving the equations, a prefix operation can be applied. The aim of this operation is to reduce the degree of the equations to increase the efficiency of finding a solution. A reduction in degree can be achieved by finding

low degree multiples of the output nonlinear function that generates the keystream. Alternatively, relabeling of variables can be used, although this reduces the degree of the equations at the expense of increasing the number of variables. Linearization is a technique which involves replacing each high degree term in an equation with a single term, and the introduction of these new variables enables the equation system to be translated to a linear system. Guessing some key or state bits that occur in many high degree terms can also be used to reduce the degree of the equations, but this simultaneously increases the total complexity of the attack. In any case, once the system of equations is prepared, an attempt is made to find a solution. The output of this stage is the recovery of the unknown key or internal state bits, respectively.

Solving the system of equations may be simple if they are linear, or easily linearized. The eXtended Linearization (XL) algorithm [2] described below may be applied to obtain this. Another approach is to use Gröbner bases [10]; essentially transforming the set F of polynomials that describe the relationships into another set G of polynomials with certain properties such that F and G generate the same output, but where G is a low degree function.

2.1 XL Algorithm

The XL algorithm was proposed in 2000 as a means for solving overdefined quadratic systems [2], and adapted for equations of higher degree in 2002 [3]. The idea of the XL algorithm is to multiply the initial system of equations in n variables with maximum degree d by all possible monomials of degree up to d. If the degree of a resulting equation is equal or less than d, then this new equation is added to the system. After this the linearization approach is applied to the resulting system of equations. Let T denote the number of monomials in the system up to degree d. Then T is given by $T = \sum_{i=0}^{d} \binom{n}{i}$.

The complexity of the XL algorithm depends on the elimination technique used. Strassen's algorithm [12] is widely applied for this purpose, requiring about $7 \cdot T^{\log_2 7}$ operations.

3. DESCRIPTION OF ZUC STREAM CIPHER

The internal state of ZUC [8] consists of a 16 stage LFSR $(s_0, \ldots, s_{15})$ and two internal memory states R_1 and R_2, with each register stage containing 31 bits and each memory state containing 32 bits. Thus the total internal state size is 560 bits. The structure of ZUC can be considered as three layers: the LFSR, a bit reorganization layer which takes 16 bits from each of 8 different stages of the LFSR to form four 32-bit words, and a nonlinear function f, which uses two 32-bit words from the bit reorganization layer and the contents of the two memory stages R_1^t and R_2^t to generate two new values R_1^{t+1} and R_2^{t+1}. The second and third layer can be considered together as a NonLinear Filter (NLF) applied to the LFSR. The structure of the ZUC stream cipher is shown in Figure 1. The two memory words are combined with a word from the bit reorganization, X_0, to generate a word W used only during the initialization phase to update the shift register. In the keystream generation phase these two memory words are combined with two words from the bit reorganization layer, X_0 and X_3 to generate the keystream word Z.

3.1 Notation

The following notation will be used throughout this paper:

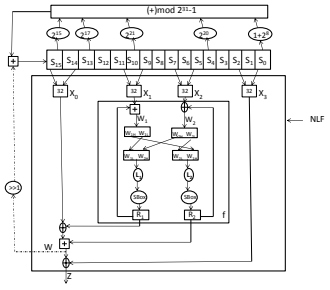

Figure 1: Structure of ZUC stream cipher

- $\oplus$: Exclusive OR.
- $(\ll b)$: Rotation left by b bit positions.
- $(\gg b)$: Shift right by b bit positions.
- $+$: Addition modulo(2^{32}).
- s_i^t: The stage i at time t.
- $||$: The concatenation of two bytes to form a 4-byte (32-bit) word.

3.2 Linear Feedback Shift Register (LFSR)

The ZUC keystream generator uses a 16-stage shift register S, where each stage contains a 31 bit value. The values of each stage are restricted to $1, 2, 3, \ldots, 2^{31} - 1$. The state update function for the LFSR differs depending on whether the cipher is in initialisation or keystream generation phase. In the initialisation phase, the update function of the LFSR is not autonomous but uses the output word W as shown in Figure 1 (including the dashed line) as discussed in Section3.3). The updated process in initialisation phase is as follows:

1. $u = (2^{15}s_{15}+2^{17}s_{13}+2^{21}s_{10}+2^{20}s_4+(1+2^8)s_0) \mathrm{mod}(2^{31}-1)$
2. $s_{16} = (u + (W \gg 1)) \mathrm{mod}(2^{31} - 1)$
3. If $s_{16} = 0$, set $s_{16} = 2^{31} - 1$
4. $(s_1, s_2, \ldots, s_{15}, s_{16}) \rightarrow (s_0, s_1, \ldots, s_{14}, s_{15})$

In the keystream generation phase, the LFSR is autonomous with state update function as follows:

1. $s_{16} = (2^{15}s_{15}+2^{17}s_{13}+2^{21}s_{10}+2^{20}s_4+(1+2^8)s_0) \mathrm{mod}(2^{31}-1)$
2. If $s_{16} = 0$, set $s_{16} = 2^{31} - 1$
3. $(s_1, s_2, \ldots, s_{15}, s_{16}) \rightarrow (s_0, s_1, \ldots, s_{14}, s_{15})$

3.3 The NonLinear Filter (NLF)

The NLF takes as input four 32-bit words from the LFSR denoted (X_0, X_1, X_2, X_3) as shown in Figure 1, and two words from memory states R_1 and R_2. There are two layers in NLF: bit reorganization and the nonlinear function f.

3.3.1 Bit reorganization

The bit reorganization layer extracts 128-bits from the state of the LFSR to form four 32-bit words as follows: $X_0 = s_{15H}||s_{14L}$, $X_1 = s_{11L}||s_{9H}$, $X_2 = s_{7L}||s_{5H}$ and $X_3 = s_{2L}||s_{0H}$, where s_{iH} and s_{iL} are the 16 most significant bits and 16 least significant bits of stage i respectively. The two words X_1^t and X_2^t are used by the nonlinear function f to update the two 32-bit words R_1^t and R_2^t. The other two words X_0^t and X_3^t are used by NLF to generate the output word Z.

3.3.2 The nonlinear function f

The nonlinear function f uses four words as input: X_1^t, X_2^t, R_1^t and R_2^t. Two output words R_1 and R_2 are produced as shown in Figure 1. The two intermediate words are used W_1 and W_2 in the nonlinear function f as follow: $W_1 = R_1 + X_1$, $W_2 = R_2 \oplus X_2$, $R_1 = S(L_1(W_{1L}||W_{2H}))$ and $R_2 = S(L_2(W_{2L}||W_{1H}))$, where S is a (32×32) SBox and L_1 and L_2 are linear transformations.

The (32×32) SBox used in ZUC is composed of four (8×8) Sboxes $S = (S_0, S_1, S_2, S_3)$, where $S_0 = S_2$ and $S_1 = S_3$. In order to use this construction the input word to the SBox is divided into four 8 bit subwords. Each of these is input to one of Sboxes.

These two Sboxes S_0 and S_1 have the following properties. The algebraic degree for S_0 and S_1 are 5 and 7 respectively. The highest probability of differential characteristic for S_0 and S_1 are 2^{-5} and 2^{-6} respectively. The highest bias of linear characteristic for S_0 and S_1 are 2^{-3} and 2^{-4} respectively. The number of linear independent quadratic equations that can be establish by S_0 and S_1 are 11 and 39 respectively [1, 11].

The linear transforms L_1 and L_2 transform a 32-bit word in the following way:

$L_1(X) = X \oplus (X \lll 2) \oplus (X \lll 10) \oplus (X \lll 18) \oplus (X \lll 24)$

$L_2(X) = X \oplus (X \lll 8) \oplus (X \lll 14) \oplus (X \lll 22) \oplus (X \lll 30)$

3.4 Initialisation phase

The initialisation phase takes as input the 128-bit secret key k, a 128-bit IV v and a 240-bit constant value D. Consider both k and v as the concatenation of 16 bytes, so $k = k_0||k_1||\ldots||k_{15}$ and $v = v_0||v_1||\ldots||v_{15}$, respectevely. Similarly, consider D as the concatenation of 16 15-bit values as $D = d_0||d_1||\ldots||d_{15}$. To begin, the register and memories are loaded as follows.

1. Set $s_i = k_i||d_i||v_i$ for $0 \leq i \leq 15$
2. Set the memory cells R_1 and R_2 to 0.
3. Following this, 32 iterations of the initialisation state update function (as outlined in Section 3.2) are performed.

3.5 Keystream generation phase

Once initialisation phase is completed, keystream for keystream generation phase is generated one word at time t, Z^t, as follows:

$$Z^t = ((X_0^t \oplus R_1^t) + R_2^t) \oplus X_3^t \quad (1)$$

The LFSR state update function during keystream generation is as outlined in Section 3.2. The first word of the keystream generated is discarded and other words are used to encrypt the plaintext or decrypt the ciphertext or for authentication purposes to generate a 32-bit MAC tag.

4. EXISTING ALGEBRAIC ANALYSIS OF ZUC

In this section we review the work in [1, 9, 11] that applies direct relationship to emphasis the algebraic relations between the initial state bits and the output keystream bits. From the description of the ZUC stream cipher, the three different main operations in the structure of ZUC are addition and multiplication in the finite field $\mathbb{F}_p$, where $p = 2^{31} - 1$, addition in the ring $\mathbb{Z}_{2^{32}}$, and the word XORing. Besides these, the cipher also makes use of the SBox and the linear transformation on the bit level.

4.1 Relating modular addition to the XOR operation

Let $z = x + y \bmod p$, be elements of the field $\mathbb{F}_p$, where $p = 2^{31} - 1$, and $x = x_{30}x_{29}\ldots x_1x_0$, $y = y_{30}y_{29}\ldots y_1y_0$, $z = z_{30}z_{29}\ldots z_1z_0$ be its binary expression. Then we have the following:

$$\begin{aligned} z_0 &= x_0 \oplus y_0 \oplus c_0 \\ z_1 &= x_1 \oplus y_1 \oplus c_1 \\ &\vdots \\ z_{30} &= x_{30} \oplus y_{30} \oplus c_{30} \end{aligned}$$

where $c_i = x_{i-1}y_{i-1} \oplus (x_{i-1} \oplus y_{i-1})c_{i-1}$ for $(1 \leq i \leq 31)$ are the carry bits and c_0 is calculated by substituting i as 31 instead of 0. Then to have a full description of the addition in the field $\mathbb{F}_p$, the carry bits are eliminated from the above equations, to have the following equations:

$$\begin{aligned} z_0 &= x_0 \oplus y_0 \oplus x_{30}y_{30} \oplus (x_{30} \oplus y_{30})(x_{30} \oplus y_{30} \oplus z_{30}) \\ z_1 &= x_1 \oplus y_1 \oplus x_0y_0 \oplus (x_0 \oplus y_0)(x_0 \oplus y_0 \oplus z_0) \\ &\vdots \\ z_{30} &= x_{30} \oplus y_{30} \oplus x_{29}y_{29} \oplus (x_{29} \oplus y_{29})(x_{29} \oplus y_{29} \oplus z_{29}) \end{aligned}$$

The above equations involve the input and output bits in the quadratic terms. To have the output bits z_i in linear form, the degree of the equations will increase as the indexes i increase.

Similarly, let $z = x + y \bmod (2^{32})$ be elements of the ring $\mathbb{Z}_{2^{32}}$, and $x = x_{31}x_{30}\ldots x_1x_0$, $y = y_{31}y_{10}\ldots y_1y_0$, $z = z_{31}z_{30}\ldots z_1z_0$ be its binary expression. Then we have the following:

$$\begin{aligned} z_0 &= x_0 \oplus y_0 \\ z_1 &= x_1 \oplus y_1 \oplus c_1 \\ &\vdots \\ z_{31} &= x_{31} \oplus y_{31} \oplus c_{31} \end{aligned}$$

where $c_1 = x_0y_0$ and $c_i = x_{i-1}y_{i-1} \oplus (x_{i-1} \oplus y_{i-1})c_{i-1}$ for $(1 < i \leq 31)$ are the carry bits. Then to have a full description of the addition in the ring $\mathbb{Z}_{2^{32}}$, the carry bits are eliminated from the above equations, to have the following equations:

$$\begin{aligned} z_0 &= x_0 \oplus y_0 \\ z_1 &= x_1 \oplus y_1 \oplus x_0y_0 \\ z_2 &= x_2 \oplus y_2 \oplus x_1y_1 \oplus (x_1 \oplus y_1)(x_1 \oplus y_1 \oplus z_1) \\ &\vdots \\ z_{31} &= x_{31} \oplus y_{31} \oplus x_{30}y_{30} \oplus (x_{30} \oplus y_{30})(x_{30} \oplus y_{30} \oplus z_{30}) \end{aligned}$$

In similar way, the algebraic equations can be established for the case of modulo p addition for c inputs. In the case of ZUC, there are six $(c = 6)$ inputs involved in the feedback of the LFSR which are the states $s_{15}, s_{13}, s_{10}, s_4, s_0$ (two times s_0 is input to the feedback) and so the output state s_{16} have algebraic degree 6.

The two SBoxes used in the nonlinear function f have algebraic immunity of order 2. To establish equations between the input and the output, the degree of the equations will be 2.

The total internal state of ZUC is 560 bits and the cipher generates at each clock 32 bits of output. Each output bit raises an equation between the input internal state and the output keystream bits. To recover the secret internal states bits using an algebraic attacks, a system of equations should be developed. The minimum keystream requirement to construct this system will be 18 keystream words $(\lceil 560 \setminus 32 \rceil = 18)$.

4.2 Constructing equations over $\mathbb{F}_2$

From the specification of the ZUC stream cipher, we can see that the update function of the LFSR involves five stages and raises algebraic equations of degree 6 for updating state s_{16}, since the state S_0 appears twice. This state (s_{16}) is input to the nonlinear function f through the variable X_1 after four clocks. There are two nonlinear operations in the nonlinear function: addition modulo 2^{32} and the application of the SBox.

If s_{16} is eliminated from the system of equation, then the degree of equations of the whole system will increase to 8. The number of variables T in the system of equations to recover the secret states will be:

$$T = 16 \times 31 + 18 \times 2 \times 32 = 1648$$

This system of equations can be solved using the linearization method with complexity of:

$$\left(\begin{array}{c} T \\ 8 \end{array} \right)^{2.37} \approx 2^{166}$$

If the output of the LFSR feedback computation is considered as new variables, and the algebraic equations for the LFSR states are of degree 6, then the total number of variables T under the assumption that only 18 words of keystream will be used in this system will be:

$$T = 16 \times 31 + 17 \times 31 + 18 \times 2 \times 32 = 2175$$

The complexity of solving this system using linearization method will be:

$$\left(\begin{array}{c} T \\ 6 \end{array} \right)^{2.37} \approx 2^{135}$$

The last analysis is that if the new variables are introduced and a system of equation of algebraic degree 2 is developed for the states. Then the computation of LFSR feedback will be as follows:

$$y_1 = (1 + 2^8)s_0 \text{mod} p$$

$$y_2 = 2^{20}s_4 + y_1 \text{mod} p$$

$$y_3 = 2^{21}s_{10} + y_2 \text{mod} p$$

$$y_4 = 2^{17}s_{13} + y_3 \text{mod} p$$

$$s_{16} = 2^{15}s_{15} + y_4 \text{mod} p$$

From the above equations, 93 independent algebraic equations can be generated from each equation of degree 2. Thus, the total number of independent algebraic equations of degree 2 that can be generated from the LFSR feedback computation will be $93 \times 5 = 465$. For the nonlinear function f, it is possible to generate algebraic equations of degree 2. It can be establish that 93 independent algebraic equations from the modular addition operation and the SBoxes S_0 and S_1 can generate 11 and 39 independent equation of degree 2 respectively. Thus, ZUC stream cipher can form a system of algebraic equation of degree 2 and the number of variables that can be establish under the assumption of observing only 18 words of keystream will be:

$$T = 16 \times 31 + 2 \times 32 - 1 + 17(5 \times 31 + 3 \times 32 - 2) = 4792$$

The number of equations m of degree 2 will be:

$$m = 93 + 17(93 \times 5 + 2 \times 93 + 39 + 11) = 12010$$

This system can be solved using XL algorithm, since the linearisation method is not effective, and the estimated complexity using XL algorithm will be:

$$\left(\begin{array}{c} T \\ \frac{T}{\sqrt{m}} \end{array} \right)^{2.37} \approx 2^{830}$$

Practically, the number of independent equations is much smaller than the number of variables if linearization method is used, and hence the keystream words requirement to solve the system is more than 18. So this will increase the number of intermediate variables and also will increase the complexity of solving the system. Thus solving the equations in this way is far worse than the exhaustive key search which is cost 2^{128} operations.

5. OUR ALGEBRAIC ATTACK ANALYSIS ON ZUC

In this section we apply a different type of algebraic attack to ZUC. This attack investigates the relationship between the input states of the cipher with the output using indirect relations. A direct relation cannot be easily formed for this cipher due to the complexity of the nonlinear function f. However, it may be possible to find indirect relations through the use of a combiner. Valid relation can be found by using a matrix to represent the combiner. The rows of the matrix represent the input bits of the combiner and the memory bits if any. The columns represent all the monomials in the variables of the inputs and outputs of the combiner, up to certain degree without the memory bits. Then the output relation is applied to T number of clocks without increasing the degree of the equation due to the linearity of the update function. These relations will form a system of equations which is used to recover the unknowns (the initial state bits in our case) by observing corresponding values of the keystream bits and substituting their values in the system of equations.

5.1 Building relations for NLF

The NLF of ZUC at time t takes four 32-bit words from different stages of the shift register to generate one 32-bit word of keystream Z^t during the keystream generation, as shown in Figure 1. Two words X_1 and X_2 are inputs to the nonlinear function f. The output of f is the two words R_1 and R_2 which are used with the other two words of the input stages X_0 and X_3 to generate the output keystream word Z^t as shown in equation 1.

We start our analysis by considering the NLF as a combiner with enclosed dash line shown in Figure 2. The aim of this combiner is to find relationship between the input bits from the states of the shift register and the output keystream bits. There are four input words to the combiner which are from the states of shift register and one output word which is the keystream word.

To find a valid relation we have to construct a matrix for the combiner. The rows of the matrix represent the input bits of the words X_0^t, X_1^t, X_2^t and X_3^t. The columns represent all monomials up to degree d in the 128-bits of state input and the 32 bits of the output word Z^t. To find the degree d, equation 2 should be satisfied such that the minimum value of degree d is selected.

$$2^{Mm} \cdot \sum_{i=0}^{d} \left(\begin{array}{c} Mc \\ i \end{array} \right) > 2^{Mc+l} \qquad (2)$$

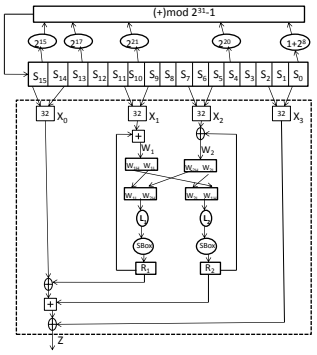

Figure 2: NLF combiner

where M represents consecutive steps of forming the combiner, c, m, and l represent number of the input bits, the output bits and the memory bits respectively.
The minimum degree that satisfies equation 2 is 30 for M =1. The value of the degree is very high and it is not practical. So we have to see other combiner to minimize the degree of the equation.

Now we will consider the nonlinear function f as a combiner with memory. This combiner has two words of input X_1^t and X_2^t, two words of memory R_1^{t-1} and R_2^{t-1} and two words of output R_1^t and R_2^t, as shown in Figure 3.

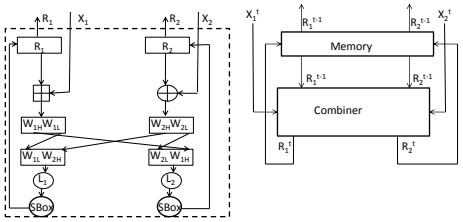

Figure 3: ZUC NonLinear function f

To find a valid relation we have to construct a matrix for the combiner. The rows of the matrix represent the input bits of the words X_1^t and X_2^t and the memory bits of the words R_1^{t-1} and R_2^{t-1}. The columns represent all the monomials in the variables of the inputs and outputs, up to certain degree without the memory bits. The problem here is that the memory bits are the same as the output bits. If we eliminate the memory bits by using the output of the matrix relations then in this case we also eliminate the output bits which we need to find a valid relation between the input and the output. This type of combiner is also not working due to using the memory bits as the output bits.
To deal with this problem, we will consider the memory bits as an input bits to the combiner. In this case there are four input words to the combiner X_1^t, X_2^t, R_1^{t-1} and R_2^{t-1}, and two words of the output R_1^t and R_2^t as shown in Figure 4.

To find a valid relation between the input bits and the output bits of the combiner, a matrix A is constructed. The rows represent all possible input bits to the combiner which are 2^{128}. The columns represent all monomials up to degree d in

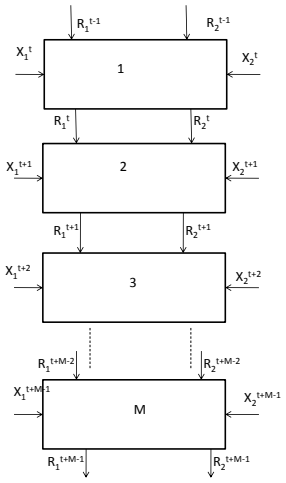

Figure 4: Combiner of ZUC

the 128-bits of register inputs and the 64 bits of the output. To find the degree d, equation 2 should be satisfied such that the minimum value of degree d is selected.
The minimum degree that satisfies equation 2 is 16 for M = 1. So we have to consider all product monomials that can be constructed for all degrees from 0 to 16. The number of columns of this matrix A will be:

$$2^{64} \cdot \sum_{i=0}^{16} \binom{128}{i} \cong 2^{131} > 2^{128}$$

Since the number of columns is greater than the number of rows, at least one column must be a linear combination of other columns. The linear dependencies can be efficiently found by Gaussian elimination. So, we can construct equations of up to degree 16 that relate the input bits and the output bits to the combiner.
The output equations relate the input words $X_1^t, X_2^t, R_1^{t-1}, R_2^{t-1}$ to the output R_1^t, R_2^t of the combiner. Then we use equation 1 to get the direct relationship between the input words of the shift register and the output keystream words.
An attack will be considered significant if it require less than 2^{560} operations, since the size of the internal states of ZUC is $n = 560$ bits. We have 560 unknowns in this system representing the initial state of the 31 bits in each of the 16-stages of the shift register and the two words of the memories R_1 and R_2. This system can be solved by the XL algorithm [3].
The number of monomials T in this system is given by:

$$T = \sum_{i=0}^{16} \binom{560}{i} \approx 2^{102}$$

Therefore, we would need to generate at least T equations for all initial states to be recovered. This means that 2^{102} keystream word observations from a single key is required.
The complexity of this attack is

$$C = (7/64) \cdot T^{\log_2 7} \approx 2^{282}$$

5.2 Algebraic attacks on modified version of ZUC

In this section we analysis the modified version of the ZUC. The modified version of ZUC considers the same inputs but only the amount of the output is increased.

The new structure can be done by different ways, for example, by considering the output of each SBox as an output word after XORing it by one of the free words of the bit reorganization word. We repeat the analysis for different values of m and then calculate the minimum degree d that satisfies equation 2, the keystream requirement and the complexity of the attack.

Table 1 shows all the specifications of the attack for the modified version of the ZUC stream cipher. In table 1, time (step 1) represents the time requires for finding valid equations. Time (step 2) represents the time requires for solving these equations effectively.

n= 496 , M=1, c=128(4 words)						
m	32	64	65	80	100	120
d	30	16	15	11	5	2
Keystream bits	2^{162}	2^{99}	2^{95}	2^{74}	2^{38}	2^{17}
Time(step 1)	2^{357}	2^{357}	2^{357}	2^{357}	2^{357}	2^{357}
Time(step 2)	2^{453}	2^{278}	2^{265}	2^{206}	2^{106}	2^{47}

Table 1: Algebraic attack on ZUC

We see that the time required to find equations is same for all numbers of output bits m. However, the keystream bits requirement and the time require to solve the equations decrease when the number of output bits m increased. This means that the security of the ZUC cipher collapse when we increase the number of output bits in order to increase the speed of the cipher.

If we compare the results given in Table 1 with the results done for Snow 2.0 in Table 3 in [4], we can see that the two results are same. This means that the ZUC and Snow cipher designs are optimal for the design output which is resistance to this type of algebraic attack.

6. CONCLUSION

This paper presents an alternative method for applying an algebraic attack to the ZUC stream cipher. An existing method finds direct relations between the input and the output keystream, but has been shown to be infeasible. The method presented in this paper uses an indirect method and considers the keystream generator as a combiner with memory to form the equations relating the inputs to the outputs. Of the possibilities considered, only one is effective in determining equations for the nonlinear function f of the NLF.

The algebraic analysis is also applied to a modified version of the ZUC stream cipher, where more output bits are produced at each clocking. This increases the throughput of the cipher, but the analysis shows that the security of the cipher is greatly reduced as a result.

7. REFERENCES

[1] C. Cid, S. Murphy, F. Piper, and M. Dodd. ZUC Algorithm Evaluation Report. Technical report, Codes & Ciphers Ltd, 7 May 2010.

[2] N. Courtois, A. Klimovand, J. Patarin, and A. Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In B. Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000: International Conference on the Theory and Application of Cryptographic Techniques*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Springer-Verlag, 2000.

[3] N. T. Courtois. Higher order correlation attacks, XL algorithm and cryptanalysis of Toyocrypt. In C. H. L. Pil Joong Lee, editor, *Information Security and Cryptology*, volume 2587 of *Lecture Notes in Computer Science*, pages 182–199. Springer, 2002.

[4] N. T. Courtois. Algebraic attacks on combiners with memory and several outputs. In S. Park, Choonsik; Chee, editor, *Information Security and Cryptology*, volume 3506 of *Lecture Notes in Computer Science*, pages 3–20. Springer, 2004.

[5] N. T. Courtois and W. Meier. Algebraic Attacks on Stream Ciphers with Linear Feedback. In E. Biham, editor, *Advances in cryptology EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345 – 359. Springer-Verlag, 2003.

[6] N. T. Courtois and J. Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In Y. Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002: 8th International Conference on the Theory and Application of Cryptology and Information Security*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer-Verlag, 2002.

[7] ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 1: 128-EEA3 and 128-EIA3 Specification. Technical report, ETSI, 4th January 2011. http://gsmworld.com/documents/EEA3_EIA3_specification_v1_5

[8] ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 2: ZUC Specification. Technical report, ETSI, 4th January 2011. http://gsmworld.com/documents/EEA3_EIA3_ZUC_v1_5.pdf.

[9] ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 4: Design and Evaluation Report. Technical report, ETSI, 18th January 2011. http://gsmworld.com/documents/EEA3_EIA3_Design_Evaluatio

[10] J. Faugère. A new efficient algorithm for computing Grobner bases without reduction to zero (F5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, pages 75–83. ACM, 2002.

[11] L. R. Knudson, B. Preneel, and V. Rijmen. Evaluation of ZUC. Technical report, ABT Crypto, 9 May 2010.

[12] V. Strassen. Gaussian Elimination is not Optimal. In *Numerical Mathematics*, pages 13:354–356, 1969.