Queensland University of Technology
Brisbane Australia

This is the author's version of a work that was submitted/accepted for publication in the following source:

Al Mashrafi, Mufeed, Koon-Ho Wong, Kenneth, Simpson, Leonie, Bartlett, Harry, & Dawson, Edward (2011) Algebraic analysis of the SSS stream cipher. In *Proceedings of the 4th international conference on Security of information and networks*, ACM, Macquarie Graduate School of Management, Sydney, NSW, pp. 199-204.

This file was downloaded from: http://eprints.qut.edu.au/48792/

© Copyright 2011 ACM

Notice: *Changes introduced as a result of publishing processes such as copy-editing and formatting may not be reflected in this document. For a definitive version of this work, please refer to the published source:*

http://dx.doi.org/10.1145/2070425.2070457

Algebraic analysis of the SSS stream cipher

Mufeed AlMashrafi[1], Kenneth Wong[2], Leonie Simpson[2], Harry Bartlett[2], Ed Dawson[2]
Information Security Institute
Queensland University of Technology
GPO Box 2434, Brisbane Qld 4001, Australia
[1] almashrafi@student.qut.edu.au
[2] (lr.simpson,kk.wong,h.bartlett,e.dawson)@qut.edu.au

ABSTRACT

Both the SSS and SOBER-t32 stream cipher designs use a single word-based shift register and a nonlinear filter function to produce keystream. In this paper we show that the algebraic attack method previously applied to SOBER-t32 is prevented from succeeding on SSS by the use of the key-dependent substitution box (SBox) in the nonlinear filter of SSS. Additional assumptions and modifications to the SSS cipher in an attempt to enable algebraic analysis result in other difficulties that also render the algebraic attack infeasible. Based on these results, we conclude that a well-chosen key-dependent substitution box used in the nonlinear filter of the stream cipher provides resistance against such algebraic attacks.

Keywords

Algebraic Attack, Stream Ciphers, SSS, eSTREAM, Multivariate Equations.

1. INTRODUCTION

SSS [11] and SOBER-t32 [12] are two designs from the SOBER family of stream ciphers [13]. Each cipher design is based on a single 17-stage word-based shift register, and uses a nonlinear filter function (NLF) to generate the keystream outputs. In both cases the word-based NLF makes use of a Substitution Box (SBox), and both modular and binary addition (XOR) are performed over the word size. Although the structures are similar, there are some important differences between these two designs. SOBER-t32 is a synchronous stream cipher that uses 32-bit words, a 256-bit key (8 words) and a linear feedback function for the shift register. SSS is a self-synchronous stream cipher that uses 16-bit words, a 128-bit key (also 8 words) and a nonlinear feedback function for the shift register. Although both designs apply a nonlinear filter to stages of the shift register in producing keystream, there are differences in the components of this function. For SOBER-t32, the SBox used is fixed, and it is applied once in the NLF, whereas the SBox used for SSS is key-dependent, and it is used twice in the NLF. A final difference is that for SOBER-t32, the output of the NLF is *stuttered* before it is used as a keystream. For SSS the output of NLF is used directly as the keystream. Besides keystream generation, SSS is also used to generate a 128-bit Message Authentication Code (MAC), although this functionality is not explored in this paper.

The SOBER-t32 cipher, with the stuttering function removed, has been attacked using an algebraic attack to recover the initial states of the shift register [3]. The basic idea of this attack is to develop a set of equations for the NLF of SOBER-t32 that relate the inputs to the NLF (taken from the shift register stages) with the keystream outputs. For a known keystream segment, solving the set of equations permits the recovery of the initial state of the shift register.

Previously, the SSS cipher was analysed using a chosen-ciphertext attack [9] to recover the secret key. Self-synchronous ciphers are vulnerable to this style of attack, as the attacker has control over some inputs to the keystream generator. The attack presented in [9] uses less than 10 kB of a single chosen-ciphertext stream to recover the contents of the key-dependent SBox, taking on average ten seconds to perform using a PC. The authors of the attack claim that this attack is practical, and the time and keystream requirements for their simulated attacks support that statement. However, the SSS specification [11] states that the cipher designers assume that the result of decrypting altered ciphertext will not be made available to the attacker, since then the attacker has complete control over the state of the cipher through the self-synchronising mechanism. Hence, the existing attack may be considered successful only if it is possible to access the cipher in a way prohibited by the cipher designers.

In this paper, we investigate the application of a different type of attack to SSS; one that does not require chosen-ciphertext. As the algebraic attack on SOBER-t32 without stuttering [3] was successful, and the two ciphers have similar designs (and SSS does not employ stuttering), we have attempted to modify the SOBER-t32 attack and apply it to SSS.

The outline of this paper is as follows. Section 2 gives a brief description of the keystream generators for SOBER-t32 and SSS. An outline of the algebraic attack method is provided in Section 3. Section 4 shows how we applied an algebraic attack to SSS. Section 5 discusses the results obtained and

presents some concluding remarks regarding the security of nonlinearly filtered keystream generators against algebraic attacks.

2. DESCRIPTION OF KEYSTREAM GENERATORS

2.1 Notation

The following notation will be used throughout this paper:

- $\oplus$: Exclusive OR.
- $(\ggg b)$: Rotation right by b bit positions.
- $+$: Addition modulo the word size:(2^{32}) or (2^{16}) for SOBER-t32 or SSS respectively.
- W^t: word at time t.
- W_i^t: The i-th bit of word W at time t.
- $S[i]_j^t$ and $R[i]_j^t$: The j-th bit position of register stage i at time t.
- $S[i]_{j\to k}^t$ and $R[i]_{j\to k}^t$: A consecutive bit stream from the j-th to k-th bits of stage i at time t.

This notation may be applied to other symbols. For example we use P^t , Z^t and C^t to represent the plaintext, keystream and ciphertext words respectively at time t.

2.2 Description of SOBER-t32 keystream generator

The SOBER-t32 keystream generator consists of a 17-stage Linear Feedback Shift Register (LFSR) and a nonlinear filter function (NLF) as shown in Figure 1. Each register stage $S[0], \ldots, S[16]$, contains a 32-bit word. Thus, the total internal state size is 544 bits.

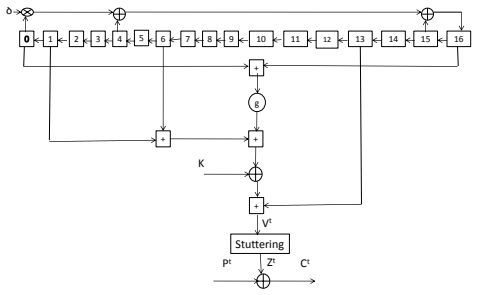

Figure 1: SOBER-t32 keystream generator

The state update function for the LFSR is as follows:

$$S[i]^{t+1} = \begin{cases} S[i+1]^t, & \text{for } i = 0, \ldots, 15, \\ S[15]^t \oplus S[4]^t \oplus \delta \cdot S[0]^t, & \text{for } i = 16. \end{cases}$$

where δ = 0xc2db2aa3 and $\cdot$ represents multiplication over $GF(2^{32})$.

The output function for SOBER-t32 is performed in two stages. Firstly, the intermediate value V^t is obtained as a nonlinear combination of the contents of five stages of the LFSR and the key-dependent constant K. The value of the constant K, is determined during the initialization of the LFSR and is retained for the entire session. As shown in Figure 1, V^t is given by the following equation:

$$V^t = ((g(S[0]^t + S[16]^t) + S[1]^t + S[6]^t) \oplus K) + S[13]^t$$

The function g uses a fixed SBox with an 8-bit input and 32-bit output. We let D denote the input to the function g, D_H denote the 8 most significant bits and D_L denote the 24 least significant bits. In [12] the role of the SBox in g is given by:

$$g(D) = SBox(D_H) \oplus (0||D_L) \quad (1)$$

If we expand the function g using the expression in equation 1, we obtain an equivalent structure for the NLF, as shown in Figure 2.

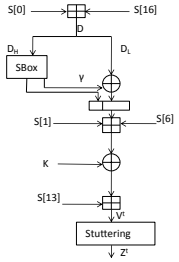

Figure 2: NLF of SOBER-t32

The second stage of the output function is irregular decimation of the output of the NLF. (In [12], this is referred to as stuttering). The sequence of keystream words Z^t is the irregular decimation of the sequence of words V^t. This is intended to make correlation attacks infeasible. The ciphertext word C^t at time t is generated by XORing the generated keystream word Z^t with the plaintext word P^t as shown in Figure 1. The algebraic attack is applied to SOBER-t32 with the stuttering mechanism removed. Therefore, we do not elaborate on this operation here. For stuttering details, the reader is referred to the SOBER-t32 specification [12].

2.3 Description of SSS keystream generator

The SSS keystream generator consists of a 17-stage shift register and a NLF as shown in Figure 3. Each register stage $R[0], \ldots, R[16]$, contains a 16-bit word. Thus, the total internal state size is 272 bits.

The state update function for the shift register is given as follows:

$$R[i]^{t+1} = \begin{cases} C^t & \text{for } i = 16 \\ R[15]^t + f(c^t \ggg 8) & \text{for } i = 14 \\ f(R[13]^t) & \text{for } i = 12 \\ (R[2]^t \ggg 8) & \text{for } i = 1 \\ R[i+1]^t & \text{for } i = 0, 2, \ldots, 11, 13, 15. \end{cases}$$

The keystream word Z^t is obtained as a nonlinear combination of the contents of five stages of the shift register as shown in Figure 3, and given by following equation:

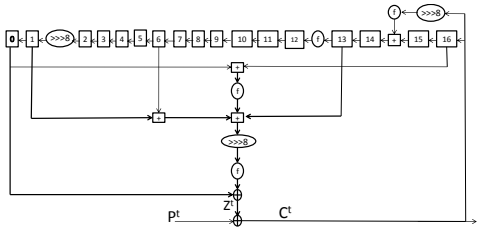

Figure 3: SSS keystream generator

$$Z^t = f(f(R^t[0] + R^t[16]) + R^t[1] + R^t[6] + R^t[13]) \ggg 8) \oplus R^t[0]$$

An important component of SSS is the nonlinear function f. The function f makes use of a key-dependent SBox which has an 8-bit input and a 16-bit output. Similar to SOBER-t32, we consider the 16 bit intermediate word X as the concatenation of the two eight bit words, X_H and X_L. In [11] the role of the SBox in f is given by

$$f(X) = \text{SBox}(X_H) \oplus X \tag{2}$$

If we expand the function f using the expression in equation 2, we obtain an equivalent structure for the NLF, as shown in Figure 4.

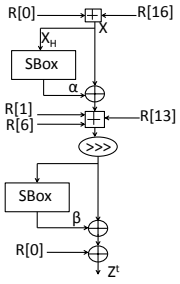

Figure 4: Non-Linear Filter (NLF) of SSS

The ciphertext word C^t at time t is formed by XORing the generated keystream word Z^t with the plaintext word P^t . As SSS is a self-synchronous stream cipher, the ciphertext word C^t is fed back to the stages of shift register as shown in Figure 3.

3. OUTLINE OF ALGEBRAIC ATTACKS

Algebraic attacks on symmetric ciphers were applied first to block ciphers in 2002 [8] and then to stream ciphers in 2003 [7]. Algebraic attacks on stream ciphers work by finding equations describing relations between the secret key or the internal state and the output keystream. There are two main steps in performing algebraic attacks. The first step is to find valid relationships between the internal state bits or the key bits and the keystream output bits. These are expressed as a set of equations. The second step is to solve this system of equations efficiently. 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. At the bit level, the cipher functions are described in terms of the XOR and AND bit operations (addition and multiplication over GF(2) respectively).

The relationships between initial state bits or key bits and known keystream 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 step is a system of multivariate equations of a certain degree.

In order to simplify solving the system of equations, a prefix operation can be applied before beginning step two. The aim of this operation is to reduce the degree of the equations. This can be achieved by finding low degree multiples of the output nonlinear function that generates the keystream. Alternatively, relabelling can be used where tradeoffs are made between the number of variables and the degree of the equations. Also, guessing some key or state bits that occur in many high degree terms could reduce the degree of the equations, but at the same time, it may increase the total complexity of the attack. Once the system of equations is prepared, an attempt is made to find a solution. The output of this step is the recovery of the unknowns, namely the key bits or the initial state bits of the register.

There are several approaches to solving the system of nonlinear equations. One method is known as linearization. This 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. Another method is to use Gröbner bases [10]. The idea behind this method is to transform the set F of polynomials that describe the problem into another set G of polynomials with certain properties such that F and G generate the same output but with low degree in function G. Another approach to solve the nonlinear equation system is the XL algorithm [4].

Let T be the number of monomials in the system of n variables with maximum degree d, then T is given by the following equation:

$$T = \sum_{i=0}^{d} \left(\begin{array}{c} n \\ i \end{array} \right)$$

A practical algorithm to solve this system is Strassen's algorithm [14] . This algorithm requires about $7 \cdot T^{\log_2 7}$ operations. As we are working over GF(2), the processor of a modern CPU can handle 64 such operations at each cycle. Thus, the total complexity C using this algorithm will be:

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

4. APPLYING THE ALGEBRAIC ATTACK TO SSS

In this section, we modify the algebraic attack applied to SOBER-t32 [3], and attempt to apply it to SSS. We consider a word as 16 individual bits and we start by building multivariate equations for the NLF. These equations relate the contents of the shift register stages with the keystream output.

4.1 Constructing multivariate equations for NLF

The NLF of SSS at time t takes input words from five stages of the shift register to generate one word of keystream Z^t as shown in Figure 4. The relation between the words of the shift register stages $R[0]$, $R[16]$ and the intermediate word X can be described at time t with the following equation:

$$X^t = R[0]^t + R[16]^t$$

The first two bits of X, namely X_0 and X_1, can be obtained from the following equations which use binary addition modulo 2 (exclusive OR) instead of integer addition:

$$\begin{aligned} X_0^t &= R[0]_0^t \oplus R[16]_0^t \\ X_1^t &= R[0]_1^t \oplus R[16]_1^t \oplus R[0]_0^t R[16]_0^t \end{aligned}$$

Taking a similar approach, we can determine relations among the SBox outputs α and β, the keystream output Z, and several register states, as follows:

$$\begin{aligned} Z_8^t &= X_0^t \oplus R[1]_0^t \oplus R[6]_0^t \oplus R[13]_0^t \oplus \beta_8^t \oplus R[0]_8^t \oplus \alpha_0^t \\ Z_9^t &= X_1^t \oplus R[1]_1^t \oplus R[6]_1^t \oplus R[13]_1^t \oplus \beta_9^t \oplus R[0]_9^t \oplus \alpha_1^t \oplus \\ &\quad (\alpha_0^t \oplus X_0^t)(R[1]_0^t \oplus R[6]_0^t \oplus R[13]_0^t) \oplus \\ &\quad R[1]_0^t (R[6]_0^t \oplus R[13]_0^t) \oplus R[6]_0^t R[13]_0^t \end{aligned} \quad (3)$$

Alternatively we can build other relations as follows. If we look into the first modular addition $X = R[0] + R[16]$, we can divide this addition into two separate additions. The first addition is the addition of the least significant byte which is the addition between bits $0 \rightarrow 7$ of stages $R[0]$ and $R[16]$. The second addition is the most significant byte which is used as an input to the SBox. In the second addition there will be a carry s at the 8-th position from the previous addition and this carry should be considered in this addition process. So the two additions can be described as follows:

$$\begin{aligned} X_{0\rightarrow 7} &= (R[0]_{0\rightarrow 7} + R[16]_{0\rightarrow 7}) \bmod 2^8 \\ X_{8\rightarrow 15} &= (R[0]_{8\rightarrow 15} + R[16]_{8\rightarrow 15} + s) \bmod 2^8 \end{aligned}$$

4.2 Constructing the SSS combiner

The NLF of SSS is too complex for the output keystream to be modeled directly as functions of register states. However, indirect relationships may be computed in a similar manner to the algebraic attack on SOBER-t32 by treating the NLF as a combiner with memory. The combiner for SSS is shown in Figure 5. It can be observed that it is not possible to determine α_0, α_1 given the inputs, since the SBox is key-dependent. Therefore, this combiner cannot be used to find valid relations for SSS.

Following the discussion from [6] for key-dependent combiners, the unknown output bits from using the key-dependent

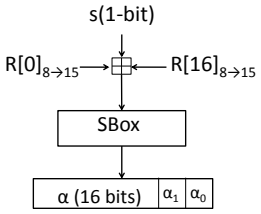

Figure 5: Combiner structure of SSS

SBox twice can be treated as memory bits in the combiner. This means that these unknown bits would not appear in the relations obtained from the combiner.

We extend the combiner to cover the entire NLF, where the output is the keystream bit Z_8, described in equation 3. Let α and β represent the intermediate values which are the output of the first use of SBox and second use of SBox respectively. The Z_8 keystream bit depends on β_8, $R[0]_8$, $R[13]_0$, $R[6]_0$, $R[1]_0$, α_0, $R[0]_0$ and $R[16]_0$. As mentioned before, we treat bits of α and β as memory bits. In total there would be 6 bits of input, one bit output, and 2 memory bits. This combiner is shown in Figure 6.

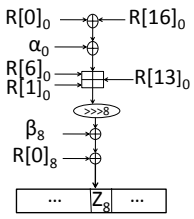

Figure 6: Possible valid combiner structure for NLF of SSS

If we construct the matrix M for this combiner, it is necessary to have 2^8 rows in M to cover all possibilities of the input bits and memory bit. On the other hand, there are only 7 variables from the input and the output for the columns. To guarantee a valid relation between the inputs and outputs, we require that

$$\sum_{i=0}^{k} \binom{7}{i} > 2^8$$

This inequality cannot be satisfied, since the maximum value for the binomial sum is 2^7. This means that valid relations cannot be guaranteed, since there would be always more rows then columns in the matrix M. Nevertheless, it may be possible to still find valid relations if the rank of M happens to be lower than the number of columns, but this occurs with a very low probability.

Since the matrix M is of a practical size, we have constructed M using Magma 2.12 [2] . All possible monomials up to the maximum degree 7 are used, giving 2^7 columns, which is the highest possible for the 7 bits of chosen inputs and outputs in the combiner. It has been found that M has rank 2^7, which means that it has trivial nullspace, and so

no linear dependencies among the columns. Therefore, it is not possible to obtain a valid relation for this combiner.

4.3 Conditional analysis of SSS

From Section 4.2, it is clear that a valid relation cannot be obtained by the combiner method. However, suppose that another method could be used to find these relations for this specific NLF. Such a method may not be systematic and may be difficult to discover, and we leave it as an open problem. However, we continue our analysis of SSS for the sake of completeness of the analysis under the assumption that a method for forming these relations exists.

4.3.1 Algebraic analysis assuming valid relations exist

Assume that a valid relation can actually be found for the NLF among the register states and the output keystream using an alternative method, despite the difficulties discussed previously. This means there is an equation

$$F(R[0]_0, R[16]_0, \ldots, Z_8) = 0$$

of maximum degree k in 7 variables that is valid for all possible register inputs and keystream outputs. In the worst case we have $k = 7$. The relation can then be used to generate a system of equations by evaluating the relation for a series of time steps. Under a known plaintext scenario, the keystream bits Z_j^t are assumed to be known at all times t, and these values can be substituted into the relevant equations.

If the underlying register is linear, as it is in the case of SOBER-t32, the register state bits $R[i]_j$ would be linear combinations of the initial state bits of the shift register. These linear combinations of initial state bits can simply be substituted into the equations relating the register contents to the keystream. Clocking the register does not increase the degree of the equations relating the contents of the register stages to the initial values, they remain linear. The resulting equation system would have variables representing all bits of the initial states of the shift register, and it would be of maximum degree $k \leq 7$, since all substitutions made have been linear. Solving the system means recovering these initial states.

In the case of SSS, some stages of the shift register are updated nonlinearly, so it is not possible to use only linear combinations of the initial values of the register states to express the contents of the register stages at later times. Furthermore, the updated register contents contain values obtained from the key-dependent SBox, so it is not possible to determine the register states for successive clocks without knowledge of either the key or the SBox or without introducing an excessive number of new variables. This prevents us from generating a set of equations from the relations found for the NLF, and the algebraic attack fails.

4.3.2 Algebraic analysis of modified SSS

We consider the application of the algebraic attack outlined in Sections 4.2 to a modified version of SSS, where the shift register is updated linearly using a slightly modified version of the feedback function taken from the shift register in SOBER-t16, so that the cipher remains self-synchronous. The modification consists of XORing the ciphertext word with the feedback word from the shift register in updating the register contents, as shown in Figure 7.

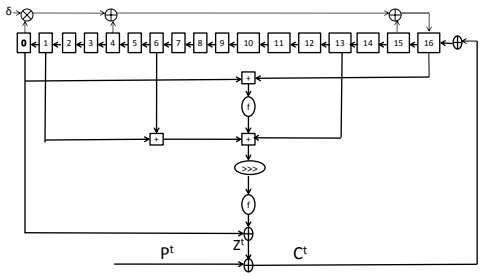

Figure 7: Modified structure of SSS keystream generator

For this modified version of SSS, we attempt to develop a system of equations for use in an algebraic attack. Since the register states are updated linearly, each state at time t can be represented as linear expressions in the initial states variables. Therefore, the system has maximum degree $k \leq 7$. Solving this system of equation permits the recovery of the initial states of the shift register. In the worst case, the equations are of degree up to 7 as mentioned in Section 4.2. We have 272 unknowns in this system representing the initial state of the 16 bits in each of the 17 stages of the shift register. This system can be solved by the XL algorithm [5].

The number of monomials T in this system is given by:

$$T = \sum_{i=0}^{7} \binom{272}{i} \approx 2^{44}$$

Therefore, we would need to generate at least T equations for all initial states to be recovered. We would also need 2^{44} observations of Z_8 from a single key, which would be obtained from at least 2^{44} keystream words generated from the key. This keystream requirement falls within the maximum of 2^{80} words of keystream output allowed for a single key IV pair in the SSS specification. The complexity of this attack using the XL algorithm would be:

$$(7/64) \cdot T^{\log_2 7} \approx 2^{121}$$

Note that the complexity of this algebraic attack on the modified SSS is less than the complexity for exhaustive key search of SSS, which costs 2^{128}. This shows that modifying the register so that the feedback function is linear, does allow the algebraic attack to be launched, under the condition that a method for forming valid relations for the NLF can be found.

5. DISCUSSION AND CONCLUSION

The algebraic attack on SOBER-t32 viewed the NLF as a combiner with two output bits (α_0 and α_1) and only one memory bit (the carry bit). The matrix constructed for this combiner had a greater number of columns than the number of rows, so there exists a multivariate equation that relates the input and the output of the combiner without the memory bit.

As the SSS cipher has a similar structure, we view the NLF of SSS as a combiner. However, the combiner constructed for the NLF of SSS has a greater number of memory bits than output bits. This is due to the fact that the SBox is key-dependent and is used twice in the NLF of SSS. As shown in Section 4.2, to construct a combiner for one bit of keystream output from the NLF, we require two memory bits. If we include more output bits in the combiner, we also have to include more memory bits that correspond to the output bits (the number of memory bits is twice the number of output bits). Therefore, when constructing a matrix for this combiner, there will always be more rows than columns. This means that linear dependencies cannot be guaranteed, so finding a valid relation for the NLF using this matrix method is unlikely. For a combiner with one bit of output, we showed that it is not possible to obtain a valid relation because the matrix is of full rank.

This result supports the claim by the SSS authors in their eSTREAM submission document [11] that SSS is secure from algebraic attack. The use of a key-dependent SBox results in the relations describing the NLF being unknown, which prevents valid relations being found among the register states and the keystream outputs using the combiner method. The situation is further compounded by the fact that the SBox is used twice in the NLF, which doubles the number of memory bits needed for the combiner. This renders a successful algebraic attack on SSS using this strategy very improbable, if not infeasible.

Our conditional analysis shows that the use of a nonlinear shift register update function also contributes to the resistance of SSS to algebraic attacks. Even if valid relations of low degree could be found for the NLF, the degree of such equations will sharply increase during the equation generation stage. Finally, we considered a modified version of SSS such that the shift register state update function is linear. We show that this makes solving the resulting system of equations feasible. This algebraic method can be launched successfully on SSS if both a valid equation can be formed for NLF and the updated function for state is linear. In such a case we estimate the complexity of the attack to be 2^{121} operations and require 2^{44} keystream bits.

In conclusion, using a key-dependent SBox in the NLF contributes to the resistance of SSS against algebraic attack using the combiner method. The use of a nonlinear update function (which also makes use of the key-dependent SBox for the shift register) also increases the resistance against this type of attack. This research indicates that the use of a key-dependent SBox may be a worthwhile strategy in designing secure keystream generators for stream ciphers.

6. REFERENCES

[1] F. Armknecht and M. Krause. Algebraic attacks on combiners with memory. In E. Biham, editor, *Advances in Cryptology CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 162–175. Springer, 2003.

[2] W. Bosma, J. Cannon, and C. Playoust. The Magma Algebra System I: The User Language. *Journal of Symbolic Computation*, 24(3–4):235–265, 1997.

[3] J. Y. Cho and J. Pieprzyk. Algebraic attacks on SOBER-t32 and SOBER-t16 without stuttering. In W. M. Bimal K. Roy, editor, *Fast Software Encryption*, volume 3017 of *Lecture Notes in Computer Science*, pages 49–64. Springer, 2004.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] J. Daemen, J. Lano, and B. Preneel. Chosen Ciphertext Attack on SSS. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/044, 2005. http://www.ecrypt.eu.org/stream.

[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] P. Hawkes, M. Paddon, G. G. Rose, and M. W. de Vries. Primitive Specification for SSS. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/028, 2005. http://www.ecrypt.eu.org/stream.

[12] P. Hawkes and G. Rose. Primitive specification and supporting documentation for Sober-t32. NESSIE project submission, 2000.

[13] G. G. Rose. SOBER: A stream cipher based on linear feedback over GF(2^8). Technical report, QUALCOMM Australia, Suite 410, Birkenhead Point, Drummoyne NSW 2137, Australia,, 1998. http://www.home.aone.net.au/ qualcomm.

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