

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,100

Open access books available

167,000

International authors and editors

185M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Recent Results on Some Word Oriented Stream Ciphers: SNOW 1.0, SNOW 2.0 and SNOW 3G

Subrata Nandi, Srinivasan Krishnaswamy and Pinaki Mitra

Abstract

In this chapter, we have studied three word-oriented stream ciphers SNOW 1.0, SNOW 2.0 and SNOW 3G in a detailed way. The detailed description includes the working principles of each cipher, security vulnerabilities and implementation issues. It also helps us to study the challenges in each cipher. As SNOW 3G is used as a confidentiality and integrity component in 3G, 4G and 5G communications, the after study of this article may instigate the reader to find the fixes from different cryptanalysis and also find a new suitable design in Mobile telephony security.

Keywords: finite field, pseudorandomness, Boolean function, attacks

1. Introduction

In modern era of communication, mobile devices, tablets, computers are developed with huge processing power, memory and storage. When one mobile device communicates with a remote server or another mobile device, the communication always takes place in a secret way. For any bank transaction or any online purchase through PC, we always require communication link between the source and the server which ensures the authentication, confidentiality and integrity of the channel. Before the year 1999, Block ciphers was the only way to provide confidentiality in any kind of communication in word oriented environment. AES, DES, 3-DES, BlowFish, Serpent, Twofish are some of the common used block cipher algorithms. The problems associated with block ciphers are mainly processing power, throughput in comparison to stream cipher. Stream cipher works very efficiently in hardware due to its simple design and good statistical properties. But it lacks in software based applications. But is it possible to make a word-oriented cipher which will be faster than Block cipher, at least gives security of AES [1] (Advance Encryption Standard) and suits in software as well as hardware? This initiate the design and analysis of word-oriented stream cipher. The basic building block of word oriented stream cipher is designed by LFSR (Linear Feedback Shift Register) with multi input multi output (MIMO) delay blocks and a Nonlinear function. In this kind of design, LFSR plays the role of generating sequence with uniform distribution. It generates m -sequence. As, sequence from LFSR can be easily cryptanalyzed by Berlekamp Massey Algorithm [2], Nonlinear maps are used along with LFSR to increase the linear complexity as well as nonlinearity of the output

sequence. Generally, S-box, Addition modulo 2^n (\boxplus), Subtraction modulo 2^n (\boxminus) are used as Nonlinear function. Word-based Cipher acts as a pseudorandom number generator (PRNG) which produces vector in each clock cycle as keystreams. Plaintext block and keystream block are encrypted with bitwise EXOR operator and it creates cipher text block. In the receiving end, cipher text block and the same keystream generator produces the plaintext block using the same bitwise exor operator. In the next subsection, we discuss about some existing word-based LFSR.

1.1 Related work

The research on word-based stream cipher was coined by Bart Preneel in FSE 1994. In NESSIE (New European Schemes for Signature, Integrity, and Encryption) competition (2000–2003), six stream ciphers (BMGL, Leviathan, LILI-128, SNOW [3], SOBER-t16 [4] and SOBER-t32 [5]) were submitted. Among which SNOW 1.0, SOBER-t16 and SOBER t-32 were found as word oriented stream ciphers. In 2002, SNOW, SOBER-t16 and SOBER t-32 were found with security flaws with certain cryptographic attacks (Distinguishing attack, Guess and Determine attack and Linear cryptanalysis). In 2002, SNOW 2 [6] was proposed by Ekdahl and Johansson, the same author published SNOW 1.0. But two cryptographic attacks, Algebraic attack and Linear Distinguishing attack made SNOW 2.0 vulnerable. After SNOW 2.0, in 2006 SNOW 3G and in 2008 Sosemanuk [7] (as a Estream finalist) came into the literature. SNOW 3G was selected as 3GPP Confidentiality and Integrity Algorithms UEA2 and UIA2. It was also analyzed by Fault Analysis attack [8] in 2009. Sosemanuk cipher is an modified of version of SNOW 2.0. There are some attacks on Sosemanuk like Linear masking method [9], byte based guess and determine attack [10]. Ekdahl et al. recently proposed SNOW-V [11] stream cipher with the feature of 256-bit security and huge throughput in 5G environment. Still, we find fast correlation attack [12] with $2^{251.93}$ complexity and improved guess and determining attack [13] with 2^{406} complexity on SNOW-V.

In this literature, we are trying to present detailed study of SNOW 1.0, SNOW 2.0 and SNOW 3G and discuss the basic problems related to it.

2. Symbols used and their meaning

The following mathematical symbols will be used in this article (**Table 1**).

3. Preliminaries

In this section, we present some definitions and related concepts useful to understand the context of next sections.

Primitive polynomial: A polynomial that generates all elements in an extension field over a base field is called Primitive Polynomial. It is also irreducible polynomial. There are $\frac{\phi(q^n-1)}{n}$ primitive polynomials of degree n in $\mathcal{GF}(q)[X]$, where ϕ is the Euler phi function.

Example 1. $x^4 + x + 1$ and $x^4 + x^3 + 1$ are two primitive polynomials of $\mathcal{GF}(2^4)$.

Linear feedback shift register (LFSR): LFSR is an important source of PRNG in stream cipher design. It is very fast and easy to implement in hardware. It consists of some D flip flops and a feedback polynomial. If f is the primitive polynomial of degree n and $\{x_1, x_2, \dots, x_n\}$ where each $x_i \in \mathbb{F}_2$, is the state of the LFSR, the state update function of the LFSR L is defined:

Symbol	Meaning
\mathbb{F}_{b^n}	Finite field of cardinality p^n , where b is a prime number
\mathbb{F}_p^n	n -dimensional vector space over \mathbb{F}_p
$GF(P)$	Galois field with elements $\in \{0, 1, \dots, p - 1\}$
v^T	A vector v with transposed form $v = (v_1, v_2, \dots, v_n)$
\oplus	XOR
\boxplus	Addition modulo 2^{32}
$M^{n \times n}$	Matrix M with n rows and n columns
$x \lll 7$	Cyclic shift of x to 7 step left
$M_m(F_2)$	Matrix Ring of $m \times m$ matrices over Finite Field F_2

Table 1.
Symbol and their meaning.

$$L : \{x_1, x_2, \dots, x_n\} \rightarrow \{f(x_1, x_2, \dots, x_n), x_2, \dots, x_{n-1}\} \tag{1}$$

If the feedback polynomial of a LFSR is primitive, it can generate all the nonzero states in its period. But LFSR based PRNG is vulnerable to Barleycamp Massey attack. It finds the initial state and the feedback polynomial of the LFSR if $2 \times n$ keystream can be accessed from the LFSR state. So, various forms of nonlinear feedback shift register (NLFSR) like Nonlinear combiner generator, Nonlinear feedforward generator, Clock control generator are used as a keystream generator to resist BMA attack. But LFSRs are slow in smartphone, PC, embedded system applications with respect to word oriented operation. So word oriented PRNG's like RC4, SOBER, SNOW, SNOW 2.0 etc. came to the market to serve the purpose of PRNG. The important factor in word oriented LFSR is primitive polynomial over extension field. These papers [14–16] are a good source of materials to study primitive polynomials over extension field.

Let b be the number of m input output delay blocks (D_0, D_1, \dots, D_b where each $D_i \in \mathbb{F}_2^m$) and gain matrices $B_0, B_1, \dots, B_{b-1} \in \mathbb{F}_2^{m \times m}$ of a multi-input multi-output LFSR (MIMO LFSR). Initial state of the MIMO LFSR is of mb bits. The state update function of a σ -LFSR, A_{mb} is defined as:

$$A_{mb} = \begin{bmatrix} 0 & I & 0 & \cdots & 0 \\ 0 & 0 & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & I \\ B_0 & B_1 & B_2 & \cdots & B_{b-1} \end{bmatrix} \in \mathbb{F}_2^{mb \times mb}$$

where $0, I \in \mathbb{F}_2^{m \times m}$ are all zero and identity matrix respectively. The characteristic polynomial of A_{mb} ,

$$f(x) = x^n + B_{b-1}x^{n-1} + B_{b-2}x^{n-2} + \cdots + B_0 \tag{2}$$

is called a primitive polynomial over \mathbb{F}_{2^m} if periodicity of the polynomial is $2^{mb} - 1$. Primitive MIMO LFSR is a good PRNG as the keystreams generated from it satisfy balancedness, span- n , 2-level autocorrelation property according to Golomb's

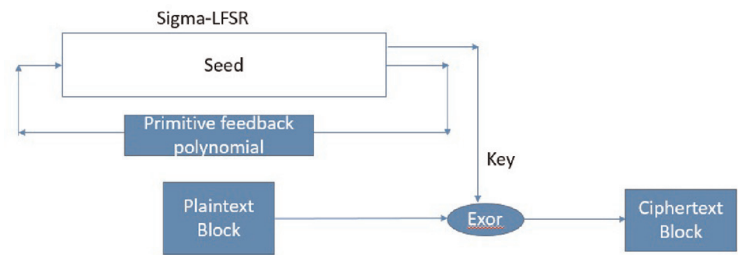


Figure 1.
Word oriented LFSR based Encryption.

randomness criterion. But only LFSR cannot be used as a PRNG due to small linear complexity. Linear complexity is the length of the smallest linear feedback shift register which can generate the sequence. To increase the linear complexity, nonlinear functions are used in PRNG along with LFSR (**Figure 1**).

Substitution Box (S-Box): An S-Box or substitution box f is a vectorial Boolean function [17] which is defined as follows:

$$f : \mathbb{F}_{2^n} \rightarrow \mathbb{F}_{2^n}$$

It is nothing but the permutation of n elements from one set to another. We can represent f as (f_1, f_2, \dots, f_n) where each f_i is the component Boolean function [18] of the S-box.

$$f_i : V_n \rightarrow \mathbb{F}_2$$

There are $n!$ S-box'es for a set of n elements. We can categorize S-boxes into two section.

1. **Affine S-box:** If all the component functions are affine functions.
2. **Non Affine S-box:** If at least one component function is nonlinear function.

In cryptology, researchers are interested on Non affine S-boxes whose all component functions are nonlinear. S-box should have good cryptographic characteristics such as balancedness, good nonlinearity, resiliency, optimal algebraic immunity, good differential uniformity [19].

Example 2. One of the S-boxes used in DES(Data Encryption Standard) is:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	

There are four boolean function with respect to this S-box. Their Algebraic Normal Form(ANF) are:

$$1.f_1 : y_0 * y_1 * y_3 + y_0 * y_2 + y_0 * y_3 + y_1 + y_3$$

$$2.f_2 : y_0 * y_1 + y_0 * y_2 * y_3 + y_0 * y_2 + y_0 * y_3 + y_0 + y_1 * y_2 * y_3 + y_1 * y_2 + y_1 * y_3 + y_1 + y_2 * y_3 + 1$$

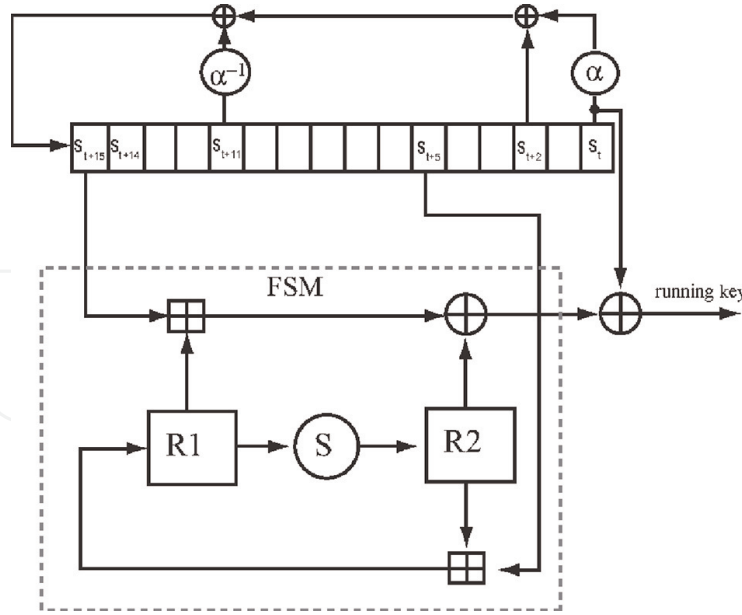


Figure 3.
Block Diagram of SNOW 2.0.

The S-box S operation acts as follows (**Figure 3**). The input y is broken into 4 bytes. Each of the bytes are changed to another byte by a nonlinear mapping from 8 to 8 bits. The nonlinear mapping is

$$x = Y^7 \oplus \beta^2 \oplus \beta \oplus 1 \quad (7)$$

where x is output of nonlinear map, Y is the input and both considered to be representing elements $\in \mathbb{F}_{2^8}$ with the polynomial basis $\{\beta^7 \cdots \beta, 1\}$. β is the root of the irreducible polynomial $h(y) = y^8 + y^5 + y^3 + y + 1$ such that $h(\beta) = 0$. The nonlinear mapping followed by a permutation of the bits in the output word.

4.1 SNOW 1.0 algorithm

In this section, we demonstrate the working nature of SNOW 1.0. It starts with Initialization, LFSRupdate, FSMupdate and finally ends with SNOW 1.0 algorithm.

Algorithm 1: Initialization().

Input: Key $K = (k_1, \dots, k_8) \in \mathbb{F}_{2^{256}}$ where each $k_i \in \mathbb{F}_{2^{32}}$ and Initialization vector $IV = (IV_2, IV_1) \in \mathbb{F}_{2^{64}}$ where each $IV_2, IV_1 \in \mathbb{F}_{2^{32}}$.

- 1: $(St_t, \dots, St_{t+15}) \leftarrow (k_1 \oplus IV_1, k_2, k_3, k_4 \oplus IV_2, k_5, k_6, k_7, k_8, k_1 \oplus 1, k_2 \oplus 1, k_3 \oplus 1, k_4 \oplus 1, k_5 \oplus 1, k_6 \oplus 1, k_7 \oplus 1, k_8 \oplus 1)$
- 2: $(Reg1_t, Reg2_t) \leftarrow (0, 0)$
- 3: **for all** $t \leftarrow 1$ to 32 **do**
- 4: $Fm_t \leftarrow (Reg1_t \boxplus St_t) \oplus Reg2_t$
- 5: $FSMupdate()$
- 6: $LFSRupdate()$
- 7: $(St_t, \dots, St_{t+15}) \leftarrow (St_t, \dots, St_{t+15}) \oplus Fm_t$
- 8: **end for**

<p>Algorithm 2: FSMupdate().</p> <hr/> <p>Input: St_t, Fm_t Output: Output of the FSM Fm_t at time t. [19] 1: $Fm_t \leftarrow (Reg1_t \boxplus St_t) \oplus Reg2_t$ 2: $Reg1_{t+1} = ((Fm_t \boxplus Reg2_t) \lll) \oplus Reg1_t$ 3: $Reg2_{t+1} = S(Reg1_t)$ 4: $Reg1_t = Reg1_{t+1}$ 5: $Reg2_t = Reg2_{t+1}$</p> <hr/>	
<p>Algorithm 3: LFSRupdate()</p> <hr/> <p>1: $temp = \alpha(St_t \oplus St_{t+3} \oplus St_{t+9})$ 2: $(St_t, \dots, St_{t+15}) \leftarrow (temp, St_t, \dots, St_{t+14})$</p> <hr/> <p>Algorithm 4 SNOW 1.0(). Input: K, IV Output: Keystream z_t at time t. 1: $Initialization(K, IV)$ 2: $t \leftarrow 1$ 3: while $t \leq GivenNumber$ do 4: $z_t = Fm_t \oplus St_t$ 5: $FSMupdate()$ 6: $LFSRUpdate()$ 7: Output z_t 8: $t \leftarrow t + 1$ 9: end while</p> <hr/>	

4.2 Weaknesses in SNOW 1.0

1. **Guess and determine attack:** It is one type of key recovery attack. It [20] utilizes the relationship between internal values (recurrence relation in a shift register) and the relationship used to establish the key-stream values from the registers values. In this attack, value of some registers are guessed and then the relationships are utilized to find other internal values.
 The problem in SNOW 1.0 is the recurrence relation

$$St_{t+16} = \alpha(St_t \oplus St_{t+3} \oplus St_{t+9}) \tag{8}$$

If we square Eq. (1),

$$St_{t+32} = \alpha^2(St_t \oplus St_{t+6} \oplus St_{t+18}) \tag{9}$$

We can find out the distance of three words between St_t and St_{t+3} , and the distance of 6 words between St_{t+3} and St_{t+9} . So the attacker can use $(St_{t+i} \oplus St_{t+6+i})$ as a single input to both the equation. Another aspect the use of \lll operator (Circular shift operator) helps in finding relation between FSM and Reg2 (Table 2).

	Data complexity	Process complexity
Guess-and-determine attack (method 1)	$\mathcal{O}(2^{64})$	$\mathcal{O}(2^{256})$
Guess-and-determine attack (method 2)	$\mathcal{O}(2^{95})$	$\mathcal{O}(2^{100})$

Table 2.
GD attack complexity

2. Distinguishing attack: In this kind of attack linear approximation of the nonlinear part is done first and combined with the linear part. Coppersmith et al. [21] observed that only α present which in $\mathbb{F}_{2^{32}}$. Using Frobenious automorphism ($\phi : y \rightarrow y^{2^{32}}$) they eliminated α and gave a new linear relation over $GF(2)$.

$$p(y)^{2^{32}} + p(y) = y^{16 \times 2^{32}} + y^{13 \times 2^{32}} + y^{7 \times 2^{32}} + y^{16} + y^{13} + y^7 \quad (10)$$

The best linear approximation of the two consecutive round input outputs of the FSM from the following.

$$\delta = (St_t)_{15} \oplus (St_t)_{16} \oplus (St_{t+1})_{22} \oplus (St_{t+1})_{23} \oplus (Fm_t)_{15} \oplus (Fm_{t+1})_{23} \quad (11)$$

where $(St_t)_k$ signify k th bit of St_0 state of the LFSR at time t . The bias of the linear approximation evaluated was at least $2^{-9.3}$. And the author calculated $2^{101.6}$ rounds keystream requirement for distinguishing the output sequence from SNOW 1.0 and the sequences from true random bit generator.

5. SNOW 2.0 KSG

This section discusses all about SNOW 2.0 Keystream generator. We also mention about some cryptographic attacks on SNOW 2.0.

SNOW 2.0 is the updated KSG over SNOW 1.0. Here, the primitive polynomial over $GF(2^{32})$ is chosen by studying the weakness of the primitive polynomial in SNOW 1.0. Let δ be the generating element of the primitive polynomial $f(y) = y^8 + y^7 + y^5 + y^3 + 1$, such that $f(\delta) = 0$ and α be the generator of the primitive polynomial $g(y) = y^4 + \delta^{23}y^3 + \delta^{245}y^2 + \delta^{48}y + \delta^{239}$ such that $g(\alpha) = 0$. We can represent each element in $\mathbb{F}_{2^{32}}$ with the help of the basis $\{\alpha^3, \alpha^2, \alpha, 1\}$. Using the above 2 extension fields the generator polynomial of SNOW 2.0

$$H(y) = \alpha y^{16} + y^{14} + \alpha^{-1}y^5 + 1 \in \mathbb{F}_{2^{32}}[Y] \quad (12)$$

is calculated and the recurrence relation of $H(y)$ is as follows:

$$St_{t+15} = \alpha^{-1}St_{t+11} + St_{t+2} + \alpha St_t \quad (13)$$

where $St_t \in \mathbb{F}_{2^{32}}$ is the state of the first delay block in clock time t .

The FSM part of SNOW 2.0 is same as SNOW 1.0, except St_{t+5} is used as a input to the FSM. It makes more dependency of state vectors to the FSM. We can evaluate the FSM Fm_t as:

$$Fm_t = (St_{t+15} \boxplus Reg1_t) \oplus Reg2_t \quad (14)$$

and the keystream z_t is given by

$$z_t = Fm_t \oplus St_t \quad (15)$$

The updation of registers $Reg1_{t+1}$, $Reg2_{t+1}$ from $Reg1_t$, $R2_t$ are related as follows:

$$Reg1_{t+1} = St_{t+4} \boxplus Reg2_t \quad (16)$$

$$Reg2_{t+1} = S(Reg1_t) \quad (17)$$

Here S is the S-box which takes 4 bytes (b_0, b_1, b_2, b_3) as input and uses AES S-box followed by mixcolumn operation to output 4 bytes.

$$\begin{bmatrix} b_0^{t+1} \\ b_1^{t+1} \\ b_2^{t+1} \\ b_3^{t+1} \end{bmatrix} = \begin{bmatrix} X & X+1 & 1 & 1 \\ 1 & X & X+1 & 1 \\ 1 & 1 & X & X+1 \\ X+1 & X & 1 & 1 \end{bmatrix} \begin{bmatrix} S(b_0^t) \\ S(b_1^t) \\ S(b_2^t) \\ S(b_3^t) \end{bmatrix} \quad (18)$$

In the above equation, the matrix used is for Mixcolumn operation where the value in $X \in \mathbb{F}_{2^8}$ and the S-box $S : \mathbb{F}_{2^8} \rightarrow \mathbb{F}_{2^8}$ is a permutation function used in SubByte step defined as:

$$S(y) = \begin{cases} 0, & \text{if } y = 0 \\ y^{-1}, & \forall y \in \mathbb{F}_2^8 - \{0\} \end{cases}$$

5.1 Key initialization

In SNOW 2.0 128 bits or 256 bits key (K) and a initialization vector IV (public) is used. The $IV \in \{0, 1, \dots, 2^{128} - 1\}$ and the two memory registers are set to 0. The cipher is then clocked 32 times where no keystream is produced and the FSM output is feeded as following:

$$St_{t+15} = \alpha^{-1} St_{t+11} \oplus St_{t+2} \oplus \alpha St_t \oplus Fm_t \quad (19)$$

The cipher is then switched into the normal mode, but the first output of the keystream is discarded. After 2^{50} keystream the cipher's key K is changed to a new value for resisting from cryptanalysis.

5.2 SNOW 2.0 algorithm

In this section, we describe the working principle of SNOW 2.0 algorithm which consists of Initialization, LFSRupdate, FSMupdate.

Algorithm 5: Initialization().

Input: Key $K = (k_0, \dots, k_7) \in \mathbb{F}_{2^{256}}$ where each $k_i \in \mathbb{F}_{2^{32}}$ and Initialization vector $IV = (IV_3, IV_2, IV_1, IV_0) \in \mathbb{F}_{2^{128}}$ where each $IV_i \in \mathbb{F}_{2^{32}}$.

```

1:  $(St_t, \dots, St_{t+15}) \leftarrow (k_0 \oplus 1, k_1 \oplus 1, k_2 \oplus 1, k_3 \oplus 1, k_4 \oplus 1, k_5 \oplus 1, k_6 \oplus 1, k_7 \oplus 1, k_0, k_1 \oplus IV_3,$   

 $k_2 \oplus IV_2, k_3, k_4 \oplus IV_1, k_5, k_6, k_7 \oplus IV_0)$   

2:  $(Reg1_t, Reg2_t) \leftarrow (0, 0)$   

3: for all  $t \leftarrow 1$  to 32 do  

4:  $Fm_t \leftarrow (Reg1_t \boxplus St_{t+15}) \oplus Reg2_t$   

5:  $FSMupdate()$   

6:  $LFSRupdate()$   

7:  $(St_t, \dots, St_{t+15}) \leftarrow (St_t, \dots, St_{t+15}) \oplus Fm_t$   

8: end for

```

Algorithm 6: FSMupdate()

(Input) St_{t+15}, St_{t+5}
Output: Output of the FSM Fm_t at time t .

```

1:  $Fm_t \leftarrow (Reg1_t \boxplus St_{t+15}) \oplus Reg2_t$   

2:  $Reg1_{t+1} = St_{t+5} \boxplus Reg2_t$   

3:  $Reg2_{t+1} = St(Reg1_t)$   

4:  $Reg1_t = Reg1_{t+1}$   

5:  $Reg2_t = Reg2_{t+1}$ 

```

Algorithm 7: LFSRupdate()

```

1:  $temp = \alpha^{-1}St_{t+11} \oplus St_{t+2} \oplus \alpha St_t$   

2:  $(St_{t+15}, \dots, St_t) \leftarrow (temp, St_{t+15}, \dots, St_{t+1})$ 

```

Algorithm 8: SNOW 2.0()

Input: K, IV
Output: Keystream z_t at time t .

```

1:  $Initialization(K, IV)$   

2:  $t \leftarrow 1$   

3: while  $t \leq 2^{50}$  do  

4:  $z_t = F_t \oplus S_t$   

5:  $FSMupdate()$   

6:  $LFSRupdate()$   

7: Output  $z_t$   

8:  $t \leftarrow t + 1$   

9: end while

```

5.3 Cryptographic attack on SNOW 2.0

5.3.1 Distinguishing attack

In this kind of attack a distinguisher algorithm is constructed to distinguish the output keystream from a PRNG and same length output from a true random number generator. If the distinguishing algorithm complexity is less than the brute force search algorithm, this is called an attack on the cipher.

1. Watanabe et al. [21] 2003 used linear masking method to distinguish the output of SNOW 2.0 from a TRNG. Basically it tries to find out linear relation between the output of the keystream with FSM and LFSR. So to serve this purpose we need to find a mask $T \in \mathbb{F}_{2^{32}}$ with high bias such that

$$TSt_{t+16} \oplus (T \cdot \alpha^{-1}) \cdot St_{t+11} \oplus TSt_{t+2} \oplus (T\alpha) \cdot St_t = 0 \quad (20)$$

holds. The 2 rounds approximation of FSM

$$T_0St_t \oplus T_1St_{t+1} \oplus T_5St_{t+5} \oplus T_{15}St_{t+15} \oplus T_{16}St_{t+16} = T_0z_t \oplus T_1z_{t+1} \quad (21)$$

with assumption that all the masks values are same, becomes possible of two nonlinear approximation such as S-box and the three \boxplus operator,

$$TS(X) = TX \quad (22)$$

$$T(X \boxplus y) = TX \oplus Ty \quad (23)$$

The bias of the total approximation can be found with complexity $\mathcal{O}(2^{-112.25})$. So we need about 2^{225} words to distinguish SNOW 2.0 from true random bit sequence.

2. In 2006 FSE, Nyberg et al. [22] improved this attack by approximating FSM (Finite state machine) and output of the cipher with different linear mask ($T, \lambda \in \mathbb{F}_2^{32}$)

$$TF(x) = \lambda x \quad (24)$$

$$T(z_{t+16} \oplus z_{t+2}) \oplus T\alpha \cdot z_t \oplus T\alpha^{-1} \cdot z_{t+11} \oplus \lambda(z_{t+17} \oplus z_{t+3}) \oplus \lambda\alpha \cdot z_{t+1} \oplus \lambda\alpha^{-1} \cdot z_{t+12} = 0 \quad (25)$$

measured the bias of the above relation with correlation (T, λ) which is defined as:

$$\text{correlation}(T, \lambda) = (\#\{x \in \mathbb{F}_2^{32} : TF(x) = \lambda x\} - \#\{x \in \mathbb{F}_2^{32} : TF(x) \neq \lambda x\}) \quad (26)$$

They also investigated the diffusion property of Mixcolumn, improved the search complexity of linear distinguishing attack.

5.3.2 Correlation attack

In correlation attack [23] over extension field, the correlation of output keystream with the LFSR output is calculated for a particular N (# available words). If the correlation or bias value is far greater than $\frac{1}{2^n}$, we find linear relation between input and output and also find out the initial state of the LFSR. It is also one kind of key recovery attack. Another kind of correlation attack is Fast Correlation attack [24] where the each output of a keystream z_i is written as:

$$z_i = u_i + e_i \quad (27)$$

u_i is the output of the LFSR and e_i is considered as error in the discrete memoryless channel which is the nonlinear function attached with LFSR. So, finding initial state of

the LFSR is equivalent of solving the decoding problem in error correcting code. We consider LFSR as (N, l) linear code, where l is the size of the LFSR.

1. In 2008, Lee et al. [9], in Asiacrypt 2008 presented more improved result with second LFSR derivation technique to mount correlation attack with time complexity $\mathcal{O}(2^{212.38})$, space complexity $\mathcal{O}(2^{202.83})$ bits and data complexity $\mathcal{O}(2^{197.77})$ bits to find out the initial state of SNOW 2.0 LFSR.
2. In 2015, Bin Zhang et al. [25] published paper on Fast correlation attack on SNOW 2.0 over extension field to find out the initial state of the LFSR with data complexity $\mathcal{O}(2^{163.59})$, time complexity $\mathcal{O}(2^{164.15})$ which is 2^{49} times faster than the previous one.
3. In 2018, Funabiki et al. [26] updated the FCA attack on SNOW 2.0 by using a MILP aided search for the linear mask very efficiently. It also uses the k-tree algorithm in [25] to find the key of SNOW 2.0 in data complexity $\mathcal{O}(2^{162.34})$ and time complexity $\mathcal{O}(2^{162.92})$.
4. In 2020, Gond et al. [27] published their work on FCA on SNOW 2.0 by slightly modifying the idea of k-tree algorithm. It finds the key of SNOW 2.0 in data complexity $\mathcal{O}(2^{159.62})$ and time complexity $\mathcal{O}(2^{162.86})$.

Theorem 6.1 *The carry bit c_i in the addition $X \boxplus Y = Z$ is equal to zero with probability $\frac{1}{2} + \frac{1}{2^{i+1}}$.*

5.3.3 Algebraic attack

Any stream cipher can be expressed with respect to algebraic equations where the variables of the equations are nothing but the initial state of the LFSR. We know that the challenge is to solving system of nonlinear equations with respect to the keystreams available to the us. It is well known to us that Solving such system of nonlinear equations over finite field is NP-Hard problem. But there are some approaches in the literature to mount algebraic attack like linearization, Grobner basis, Finding low degree annihilators [28] of a Boolean function etc.

In 2005, [29] Olivier Billet et. el cryptanalyzed SNOW 2.0 with algebraic attack like following:

1. Assuming \boxplus in the cipher as \oplus operation, the equations at time stamp t help in mounting algebraic attack:

$$Reg2_t = Reg2_0 + \sum_{i=0}^t z^i + \sum_{i=0}^t (St_{4+i} + St_{15+i} + St_i) \quad (28)$$

where only known information is output keystreams(z).

$$Reg2_{t+1} = S(Reg1_t) \quad (29)$$

$$= S(Reg2_t + z^t + St_{15+t} + St_t) \quad (30)$$

6. SNOW 3G KSG

SNOW 3G is an updated version of SNOW 2G. To get resistance from Algebraic Attack on SNOW 2.0, this cipher is proposed. It is used in mobile telephony for 4G and 5G communication as authenticated encrypted word oriented stream cipher in UEA2 and UIA2. The basic building block of SNOW 3G is as follows (**Figure 5**):

The LFSR configuration of SNOW 3G is as same as SNOW 2.0. The only changes are in the FSM configuration. One extra register R3 and extra Sbox is proposed by the designer. We discuss only the FSM construction of SNOW 3G in the following paragraph.

FSM: The FSM of SNOW 3G consists of two input words St_{15} and St_5 . It generates a 32-bit output word FW.

$$Fw = (St_{15} \boxplus R1) \oplus St_5 \quad (31)$$

On the next step, the registers are updated. To do so, we calculate one value r like following:

$$r = R2 \boxplus (R3 \oplus St_5) \quad (32)$$

Next, we update the registers

$$R3 = SBox2(R2) \quad (33)$$

$$R2 = SBox1(R1) \quad (34)$$

$$R1 = r \quad (35)$$

SBox1: This Sbox is same as the Sbox described in SNOW 2.0.

SBox2: The SBox2 (S2 in Eq. (23)) is constructed using the Dickson polynomial. It is defined as an element of $GF(2^8)$ which is generated by $x^8 + x^6 + x^5 + x^3 + 1$.

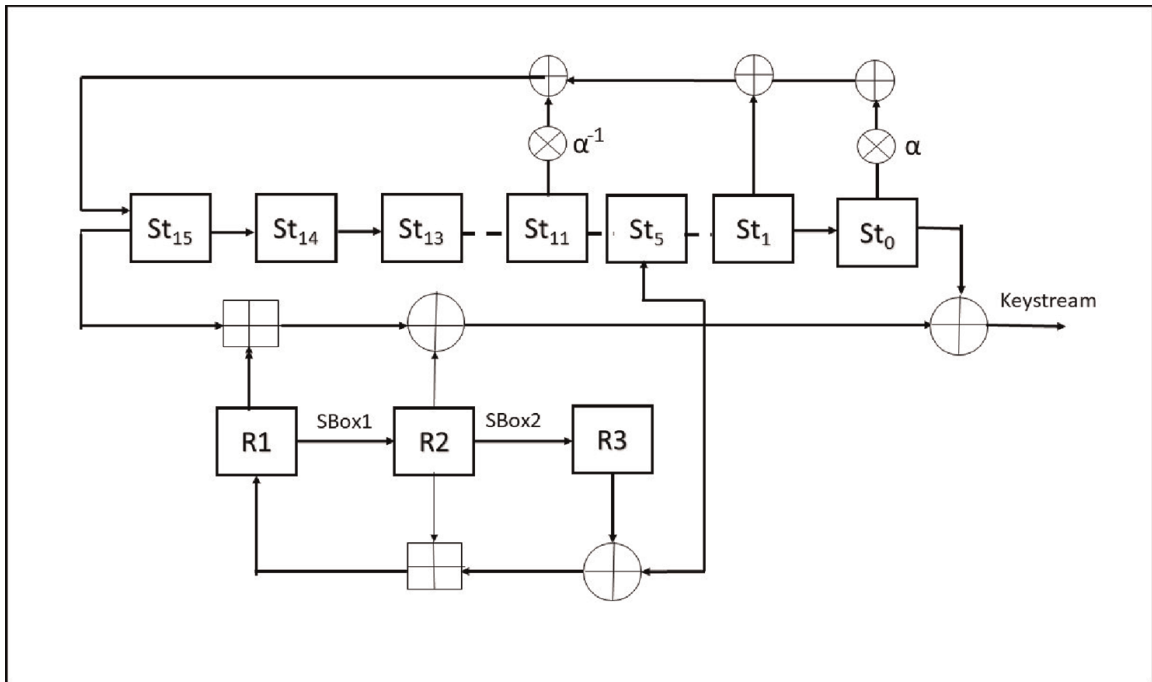


Figure 5.
Block diagram of SNOW 3G.

Suppose $b_i \in GF(2^8)$, using that we find an 32–bits inputs $b = b_1 || b_2 || b_3 || b_4$ which is given as a input to the SBox2. It works as the following:

$$\begin{bmatrix} b_0^{t+1} \\ b_1^{t+1} \\ b_2^{t+1} \\ b_3^{t+1} \end{bmatrix} = \begin{bmatrix} Y & Y+1 & 1 & 1 \\ 1 & Y & Y+1 & 1 \\ 1 & 1 & Y & Y+1 \\ Y+1 & Y & 1 & 1 \end{bmatrix} \begin{bmatrix} S2(b_0^t) \\ S2(b_1^t) \\ S2(b_2^t) \\ S2(b_3^t) \end{bmatrix} \tag{36}$$

Here, $Y \in GF(2^8)$ and the matrix of Y and 1 is used for mixcolumn transformation.

6.1 Cryptographic attack on SNOW 3G

6.1.1 Fault attack on SNOW 3G

In this attack [8, 34] the attacker has a access of the physical device and it can cause transient fault attack on the device. In this kind of fault attack, the attacker can reset the device to its original state and apply a fault to the same device to get new keystream. The attacker can find the secret key of the cipher by verifying the faulty keystream and the actual keystream.

In SNOW 3G KSG, fault is injected between two computations of the following registers:

- Register $R1^t$.
- Register $R2^t$.
- LFSR state St_0^t .

We need to keep track that we consider the memory locations of $R1^{t+1}$, $R2^{t+1}$, St_{15}^{t+1} after computation of a new word. Next we can find out $X_t \oplus X_{t'}$ which can give value either 1 or 0 for a certain fault injected positions, where X_t denotes the value the above mentioned registers in RAM and $X_{t'}$ is the faulty valued of the registers. We also verify the $Z_t \oplus Z_{t'}$ value to get the confirmation of the fault occurrence. Finally, the state of the LFSR of SNOW 3G can be found by at least 24 fault injections and solving 512 linear equation by Gaussian elimination.

Note: Besides, all the attacks on SNOW 2.0 written before except algebraic attacks are also performed on SNOW 3G. It results SNOW 3G as a 128-bit secure cipher.

In **Table 3**, we enlist the attacks possible on SNOW 1.0, SNOW 2.0 and SNOW 3G.

	Distinguishing attack	Algebraic attack	Fast correlation attack	Guess and determining attack	Fault attack
SNOW 1.0	✓	X	X	✓	X
SNOW 2.0	✓	✓	✓	✓	✓
SNOW 3G	✓	X	✓	✓	✓

Table 3.
Various attacks on SNOW 1.0, SNOW 2.0, SNOW 3G.

7. Implementation issues

7.1 Vector-vector multiplication over $\mathcal{F}_{2^{32}}$

Whole SNOW family of ciphers use LFSR with $y * \beta$ operation over $\mathcal{F}_{2^{32}}$ where β is a primitive element of $\mathcal{GF}(2^{32})$. SNOW 2.0 and SNOW 3G uses the following approach:

$$y \cdot \beta = (y \ll 8) \oplus \text{Table}(y \gg 24) \quad (37)$$

Here, $\text{Table}(x)$ function takes 8-bit vector as input and return 32-bit vector as output. It can be described as follows:

$$\text{Table}(x) = F_1(x, 23, 0xA9) \parallel F_1(x, 245, 0xA9) \parallel F_1(x, 48, 0xA9) \parallel F_1(x, 239, 0xA9) \quad (38)$$

where $0xA9$ is the hexadecimal representation of the primitive polynomial $x^8 + x^7 + x^5 + x^3 + 1$ over \mathcal{F}_2 , F_1 is a function which takes 16 bit inputs and positive integer i to 8-bit vector. $F_1(W, i, x) = W$ if $i = 0$ and $F_1(W, i, x) = F_2(F_1(W, i-1, x), x)$, otherwise, where $W, x \in \mathcal{F}_2^8$. And $F_2(W, x) = (W \ll_8 1) \oplus x$ if $\text{leftmostbit}(W) = 1$, else $F_2(W, x) = (W \ll_8 1)$. F_2 is a function which takes two 8-bit inputs and gives 8-bit output. This whole procedure reduces the time complexity of vector-vector multiplication over finite field to $\mathcal{O}(1)$ complexity.

7.2 Addition modulo $2^{32} (\boxplus)$

Let us take two vectors $P, Q \in \mathcal{F}_2^{32}$ and addition modulo 2^{32} is defined as follows:

$$P \boxplus Q = (P + Q) \pmod{2^{32}} = (P + Q) \& (0xFFFFFFFF) \quad (39)$$

Here, $+$ is normal addition and $\&$ is bitwise AND-operator.

8. Conclusion

In this article, we can observe that SNOW 1.0 has problems in its design regarding the \ll_7 shift operator. The interaction between the \ll_7 shift operator and the S-box is one of the reasons for the large correlation in the FSM. Besides, the S-box used in SNOW 1.0 is not up to the mark. So, changing the S-box with Rijndael improves its power [35] against Guess and Distinguishing attack. In this context, SNOW 2.0 is better than SNOW 1.0 with respect to the design issues and cryptographic attacks. Still, it is susceptible to several attacks. Among all the attacks, the best-known attack is Algebraic Attack [29] which is taken care in the upgraded version SNOW 3G. A new S-Box and another Register are introduced in SNOW 3G to circumvent the problems in SNOW 2.0. Besides, the only practical attack possible for SNOW 2.0 and SNOW 3G is Fault Attack [8, 34] which is possible if someone access the memory location of the SNOW algorithm. This results SNOW 3G to be used in mobile telephony with 128 bit security. Though, KDFC-SNOW is another approach which is also potential candidate to resist some known plaintext attacks on SNOW 2.0, it requires some research on the implementation issue in the initialization phase. Besides, KDFC-SNOW cannot resist

the recent fast correlation attack by Gond et al. [27] on SNOW 2.0. A possible research towards the improvement of SNOW family may be the finding an word LFSR with 64- or 128-bit block size with 256 bit key security which would be beneficial in 5G communication. Also, resistance of Fast Correlation Attack may be another kind of research which can be taken as future study.

Author details


Subrata Nandi¹, Srinivasan Krishnaswamy² and Pinaki Mitra^{1*}

1 Department of Computer Science and Engineering, IIT Guwahati, Assam, India

2 Department of Electronics and Electrical Engineering, IIT Guwahati, Assam, India

*Address all correspondence to: pinaki@iitg.ac.in

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Daemen J, Rijmen V. Rijndael: Aes proposal; 1999
- [2] Berlekamp ER. Algebraic coding theory. *Negacyclic Codes*. 1968:207-217
- [3] Ekdahl P, Johansson T. Snow—A new stream cipher. In: *Proceedings of First Open NESSIE Workshop*. KU-Leuven; 2000. pp. 167-168
- [4] Rose G. A stream cipher based on linear feedback over $gf(2^8)$. In: *Australasian Conference on Information Security and Privacy*. Springer; 1998. pp. 135-146
- [5] Rose G, Hawkes P. The t-class of sober stream ciphers. Unpublished manuscript. <http://www.home.aone.net.au/qualcomm>, 1999
- [6] Ekdahl P, Johansson T. A new version of the stream cipher snow. In: *International Workshop on Selected Areas in Cryptography*. Springer; 2002. pp. 47-61
- [7] Berbain C, Billet O, Canteaut A, Courtois N, Gilbert H, Goubin L, et al. Sosemanuk, a fast software-oriented stream cipher. In: *New Stream Cipher Designs*. Springer; 2008. pp. 98-118
- [8] Debraize B, Corbella IM. Fault analysis of the stream cipher snow 3g. In: *2009 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*. IEEE; 2009. pp. 103-110
- [9] Lee J-K, Lee DH, Park S. Cryptanalysis of sosemanuk and snow 2.0 using linear masks. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer; 2008. pp. 524-538
- [10] Feng X, Liu J, Zhou Z, Wu C, Feng D. A byte-based guess and determine attack on sosemanuk. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer; 2010. pp. 146-157
- [11] Ekdahl P, Johansson T, Maximov A, Yang J. A new snow stream cipher called snow-v. *IACR Transactions on Symmetric Cryptology*. 2019;2019(3): 1-42
- [12] Gong X, Zhang B. Resistance of snow-v against fast correlation attacks. *IACR Transactions on Symmetric Cryptology*. 2021:378-410
- [13] Yang J, Johansson T, Maximov A. Improved guess-and-determine and distinguishing attacks on snow-v. *Cryptology ePrint Archive*. 2021
- [14] Krishnaswamy S, Pillai HK. On the number of special feedback configurations in linear modular systems. *Systems and Control Letters*. 2014;66:28-34
- [15] Tsaban B, Vishne U. Efficient linear feedback shift registers with maximal period. *Finite Fields and Their Applications*. 2002;80(2):256-267
- [16] Zeng G, Han W, He K. High efficiency feedback shift register: Sigma-lfsr. *IACR Cryptology ePrint Archive*. 2007;114;2007
- [17] Carlet C. Boolean functions for cryptography and error correcting codes. *Boolean Methods and Models*. 2006
- [18] Cusick TW, Stanica P. *Cryptographic Boolean Functions and Applications*. Academic Press; 2017

- [19] Adams C, Tavares S. The structured design of cryptographically good s-boxes. *Journal of Cryptology*. 1990; **30**(1):27-41
- [20] Hawkes P, Rose GG. Guess-and-determine attacks on snow. In: *International Workshop on Selected Areas in Cryptography*. Springer; 2002. pp. 37-46
- [21] Watanabe D, Biryukov A, De Canniere C. A distinguishing attack of snow 2.0 with linear masking method. In: *International Workshop on Selected Areas in Cryptography*. Springer; 2003. pp. 222-233
- [22] Nyberg K, Wallén J. Improved linear distinguishers for snow 2.0. In: *International Workshop on Fast Software Encryption*. Springer; 2006. pp. 144-162
- [23] Siegenthaler T. Correlation attacks on certain stream ciphers with nonlinear generators. In: *IEEE International Symposium Information Theory, Saint Jovite, Canada*. 1983. pp. 26-29
- [24] Meier W, Staffelbach O. Fast correlation attacks on certain stream ciphers. *Journal of Cryptology*. 1989; **10**(3):159-176
- [25] Zhang B, Xu C, Meier W. Fast correlation attacks over extension fields, large-unit linear approximation and cryptanalysis of snow 2.0. In: *Annual Cryptology Conference*. Springer; 2015. pp. 643-662
- [26] Funabiki Y, Todo Y, Isobe T, Morii M. Several milp-aided attacks against snow 2.0. In: *International Conference on Cryptology and Network Security*. Springer; 2018. pp. 394-413
- [27] Gong X, Zhang B. Fast computation of linear approximation over certain composition functions and applications to snow 2.0 and snow 3g. *Designs, Codes and Cryptography*. 2020; **880**(11): 2407-2431
- [28] Dalai DK. Some necessary conditions of boolean functions to resist algebraic attacks [PhD thesis]. Indian Statistical Institute; 2006
- [29] Billet O, Gilbert H. Resistance of snow 2.0 against algebraic attacks. In: *Cryptographers' Track at the RSA Conference*. Springer; 2005. pp. 19-28
- [30] Ahmadi H, Eghlidos T. Heuristic guess-and-determine attacks on stream ciphers. *IET Information Security*. 2009; **30**(2):66-73
- [31] Nia MSN, Payandeh A. The new heuristic guess and determine attack on snow 2.0 stream cipher. *IACR Cryptology ePrint Archive*. 2014; **2014**: 619
- [32] Nandi S, Krishnaswamy S, Zolfaghari B, Mitra P. Key-dependent feedback configuration matrix of primitive σ -lfsr and resistance to some known plaintext attacks. *IEEE Access*. 2022; **10**:44840-44854
- [33] Krishnaswamy S, Pillai HK. On multisequences and their extensions. *arXiv preprint arXiv:1208.4501*. 2012
- [34] Armknecht F, Meier W. Fault attacks on combiners with memory. In: *International Workshop on Selected Areas in Cryptography*. Springer; 2005. pp. 36-50
- [35] Yilmaz E. Two versions of the stream cipher snow [PhD thesis]. Middle East Technical University. 2004