Appendix B. Design of Stream Ciphers

In this appendix, we introduce stream ciphers in practice including A5/1 in GSM cellular system, w7, an analogue of A5/1, E0 in Bluetooth and RC4 in WEP (wire equivalent privacy) and two stream cipher candidates from ECRPTO.

1 Examples of Stream Ciphers in Practice

In this section, we introduce four stream ciphers which are used in practical systems. Three of them are LFSR based design, and one is a transposed permutation based design.

1.1 A5/1 in GSM System

A5/1 stream cipher key generator is for securing GSM conversations. A GSM conversation is sent as a sequence of frames per 4.6 millisecond, and each frame contains 228 bits. The key stream generator of the stream cipher used in GSM is shown in Figure 1.

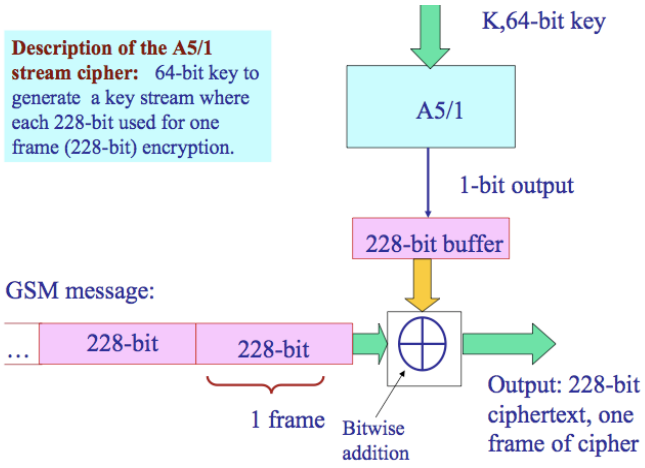

Figure 1: A Diagram of Encryption in GSM using A5/1

A. A5/1 key Stream Generator

Parameters:

1. Three LSFRs which generate m-sequences with periods $2^{19}-1$, $2^{22}-1$, and $2^{23}-1$, respectively.

Table 1: Majority Function in A5/1

(x_1, x_2, x_3)	$f(x_1, x_2, x_3)$ $= (y_1, y_2, y_3)$
000	111
111	
001	110
110	
011	011
100	
101	101
010	

2. Let $f_i(x)$ be the primitive polynomial for $LFSR_i, i = 1, 2, 3$ which generate their respective m-sequences $\mathbf{a} = \{a(t)\}$, $\mathbf{b} = \{b(t)\}$, and $\mathbf{c} = \{c(t)\}$ where

$$\begin{aligned} f_1(x) &= x^{19} + x^5 + x^2 + x + 1 \\ f_2(x) &= x^{22} + x + 1 \\ f_3(x) &= x^{23} + x^{16} + x^2 + x + 1. \end{aligned}$$

3. Tap positions: $d_1 = 11, d_2 = 12$ and $d_3 = 13$.

4. Majority function $f(x_1, x_2, x_3) = (y_1, y_2, y_3)$ is defined by the following table. That is, $y_i = 1$ if x_i is in majority where $(x_1, x_2, x_3) = (a(t + 11), b(t + 12, c(t + 12))$.

The output sequence is given by $\mathbf{u} = \{u(t)\}$ which computes at time t

$$u(t) = a(i_1) + b(i_2) + c(i_3), t = 0, 1, \cdots$$

where i_1, i_2, and i_3 are determined in a stop-and-go clock controlled model by the majority function f. A diagram of A5/1 generator is given in Figure 6.

For example, at time t, if

$$f(a(t + 11), b(t + 12), c(t + 13)) = (1, 1, 0),$$

i.e., $(y_1, y_2, y_3) = (1, 1, 0)$, then LFSR 1 and LFSR 2 are clocked and LFSR 3 has no clock pulse. Thus, at this state, the output bits is the exclusive-or of three bits for which a bit from both LFSR 1 and LFSR 2 are current bits, and the bit from LFSR 3 is the previous bit.

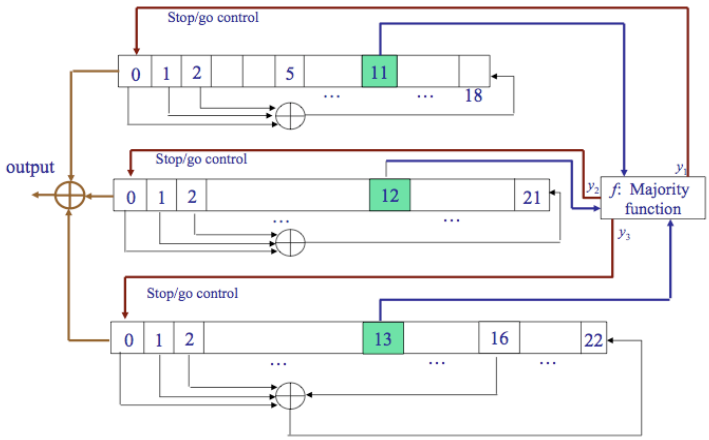

Figure 2: A Diagram of A5/1 Generator

A session key or seed consists of initial states for three LFSRs, i.e., a total of 64 bits.

Remark 1 The first original A5 algorithm was renamed as A5/1. Other algorithms include A5/0, which means no encryption at all, and A5/2, a weaker over-the-air privacy algorithm. Generally, the A5 algorithms after A5/1 have been named as A5/x. Most of the A5/x algorithms are considerably weaker than the A5/1. A5/3 is available from the Working Group of wireless communications.

B. What does A5/1 suffer ?

1. It can be broken with few hours by a PC.
2. Short period problem: Without stop/go operation, the period of sum of the three LFSRs is given by
$$(2^{19} - 1)(2^{22} - 1)(2^{23-}1).$$
However, the experiement shows that the period of A5/1 is arround
$$(4/3)(2^{23} - 1).$$
3. Collision problem: different seeds (i.e., different initial states of three LFSRs) may result in the same key stream and only about 70% seeds produce different key streams).
4. The maority function is the worst function in terms of correlation with all affine functions.

C. Possible Attacks on A5/1

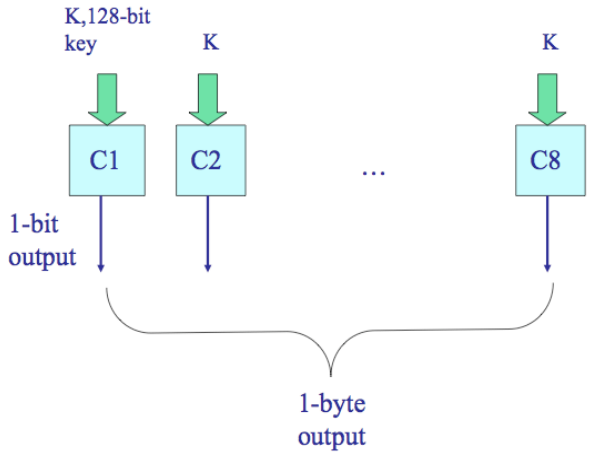

Figure 3: A Diagram of w7 Key Stream Generator

1. Brute-Force Attack against A5: If we have a Pentium III class chip with approximately 20 million transistors and the implementation of one set of LSFRs (A5/1) would require about 2000 transistors, we would have a set of 10,000 parallel A5/1 implementations on one chip.

2. If the chip was clocked to 600 MHz, we could try approximately 2M keys per second per A5/1 implementation. A key space of 2^{54} keys would thus require about 900,000 seconds or 250 hours, with one chip.

3. Alex Biryukov and Adi Shamir (co-inventor of the RSA) claim to be able to penetrate the security of a A5/1 ciphered GSM call in less than one second using a PC with 128 MB RAM and large hard drives.

1.2 w7 - an Analogue Cipher of A5/1

The w7 stream cipher algorithm is proposed by S. Thomas, D. Anthony, T. Berson, and G. Gong published as an INTERNET DRAFT, April 2002. The w7 algorithm is a byte-wide, synchronous stream cipher optimized for efficient hardware implementation at very high data rates. It is a symmetric key algorithm supporting key lengths of 128 bits. It contains eight similar models, $C1, C2, \cdots, C8$. A diagram of w7 key stream generator is shown in Figure 3 and the detailed design of the block $C2$ is shown in Figure 4.

This is an example of a stream cipher with slightly better security than A5/1. Compared with A5/1, the key size is increased from 64 bits to 128 bits, each LFSR sequence is filtered by a cubic boolean function, 8 parallel identical structures for outputting one byte instead of one bit.

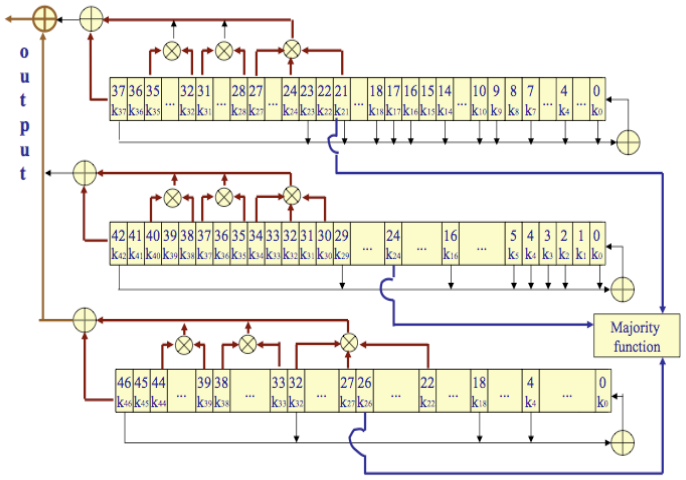

Figure 4: A Diagram of C2 Block in w7 Cipher

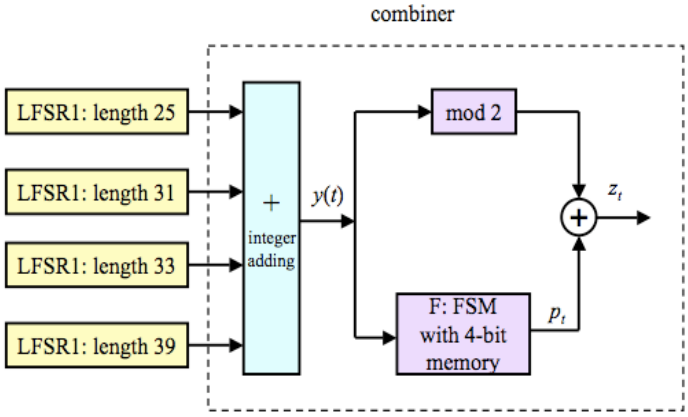

Figure 5: A Diagram of E0 Generator

1.3 E0 in Bluetooth Standard

Bluetooth is the new emerging technology for wireless communication. It was developed by a group called Bluetooth Special Interest Group (SIG), formed in May 1998. The founding members were Ericsson, Nokia, Intel, IBM and Toshiba. Since then, almost all of the biggest companies in the telecommunications business (e.g. 3Com, Microsoft, Motorola) have joined the Bluetooth SIG and the number of the participating companies is now over 1,500.

E0 is the keystream generator used in the Bluetooth wireless interface. However, it suffers algebraic attacks. A diagram of E0 key stream generator is shown in Figure 5.

Key Stream Generator of E0: E0 is a combinatorial generator which consists of $k = 4$ regularly

clocked LFSRs, a total of 128-bit initial values, and a combiner function F with 4-bit memory which is a finite state function with 4-bit memory.

1. The characteristic polynomials of four LFSRTs are given by

$$\begin{aligned} f_1(x) &= x^{25} + x^{17} + x^{13} + x^5 + 1 \\ f_2(x) &= x^{31} + x^{19} + x^{15} + x^7 + 1 \\ f_3(x) &= x^{33} + x^{29} + x^9 + x^5 + 1 \\ f_4(x) &= x^{39} + x^{35} + x^{11} + x^3 + 1 \end{aligned}$$

   Let $x_t = (x_{1,t}, x_{2,t}, x_{3,t}, x_{4,t})$ where $\{x_{i,t}\}$ is the output of the $LFSR_i$.

2. The combiner function F is a 4-bit finite state machine. Let $c_t = (q_t, p_t, q_{t-1}, p_{t-1})$ be 4-bit registers of F. Let $l(t) = x_{1,t} + x_{2,t} + x_{3,t} + x_{4,t}$ where the addition is the integer addition.

   - Output of F is p_t, i.e., $p_t = F(x_t, c_t)$.
   - The state change is given by

$$\begin{aligned} c_{t+1} &= (q_{t+1}, p_{t+1}, q_t, p_t) \\ &= (S_{1,t+1} + q_t + p_{t-1}, S_{0,t+1} + p_t + q_{t-1}, q_t, p_t) \end{aligned}$$

     where

$$S_{t+1} = (S_{1,t+1}, S_{0,t+1}) = \lfloor \frac{\pi(t) + 2q_t + p_t}{2} \rfloor$$

     where the addition in the above formula is the integer addition. $(S_{1,t+1}, S_{0,t+1})$ is the binary representation of the right hand number where $S_{1,t+1}$ and $S_{0,t+1}$ are its respective most significant bit and least significant bit.

3. The output of E0 is

$$z_t = (l(t) \bmod 2) + p_t$$

   where the addition is exclusive or addition, i.e., modulo 2.

1.4 RC4 in WEP

The 802.11 standard describes the communication that occurs in wireless local area networks (LANs). The Wired Equivalent Privacy (WEP) algorithm is used to protect wireless communication from eavesdropping. A secondary function of WEP is to prevent unauthorized access to a wireless network; this function is not an explicit goal in the 802.11 standard, but it is frequently considered to be a feature of WEP.

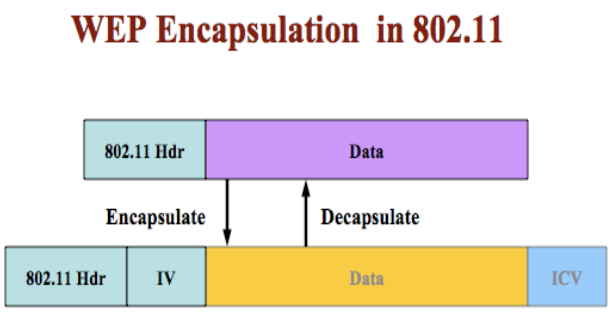

Figure 6: Format of Cipher in WEP

WEP relies on a secret key that is shared between a mobile station (e.g. a laptop with a wireless ethernet card) and an access point (i.e., a base station). The secret key is used to encrypt packets before they are transmitted, and an integrity check is used to ensure that packets are not modified in transit. The standard does not discuss how the shared key is established. In practice, most installations use a single key that is shared between all mobile stations and access points. More sophisticated key management techniques can be used to help defend from the attacks. WEP uses the RC4 encryption algorithm, which is a stream cipher.

Summary of WEP (Wired Equivalent Privacy) Encapsulation:

- Encryption algorithm is RC4.
- Per-packet encryption key is 24-bit IV concatenated to a pre-shared key.
- WEP allows IV to be reused with any frame.
- Data integrity provided by CRC-32 of the plaintext data ("ICV").
- Data and ICV are encrypted under the per-packet encryption key.

RC4 was designed by Ron Rivestin 1987 which is designated for software implementation. Its design kept secretly until 1994, when it was reversely engineered. It is used to protect Internet traffic using the SSL protocols, integrated into Microsoft windows, Lotus Notes, Apple AOCE, Oracle Secure SQL etc.. RC4 has an N stage register, which holds a permutation S of all the $N = 2^n$ possible n-bit integers, where n is typically chosen as 8. The initial state is derived from a key (whose typical size is between 40 and 256 bits) by a Key-Scheduling Algorithm (KSA), and then the Pseudo-Random Generation Algorithm (PRGA) alternately modifies the state (by exchanging two out of the N integers) and produces an output (by picking one of the state in S).

RC4 Key-Scheduling Algorithm and the Pseudorandom Generation Algorithm

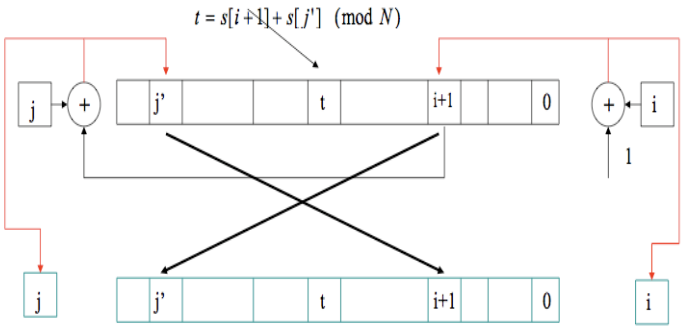

Figure 7: State Transition of RC4

KSA (K)
Initialization
For $i = 0$ to $2^n 1$
$S[i] = i$
Scrambling
$j = 0$
For $i = 0$ to $2^n 1$
$j = j + S[i] + K[i(\mathrm{mod} l)]$
$\mathrm{swap}(S[i], S[j])$

PRGA (S)
Initialization
$i = 0$, $j = 0$
Generation Loop
$i = i + 1$
$j = j + S[i]$
$\mathrm{swap}(S[i], S[j])$
$t = S[i] + S[j](\mathrm{mod} N)$
Output $z = S[t]$

2 Stream Cipher Candidates from ECRYPT

In this section, we introduce two stream cipher candidates submitted to ECRYPTO, WG steam cipher, by Yassir Nawaz and Guang Gong, 2005, which is the only one with desired randomness properties, and Grain 2, by Martin Hell, Thomas Johansson and Willi Meier, 2005, which has fastest implementation in the pool of the submissions and currently is in the third phase evaluation of eSTREAM project.

2.1 WG Stream Cipher

Outline of WG

- Synchronous stream cipher submitted in Profile 2 (for hardware applications)
- Key lengths of 80, 96, 112 and 128 bits

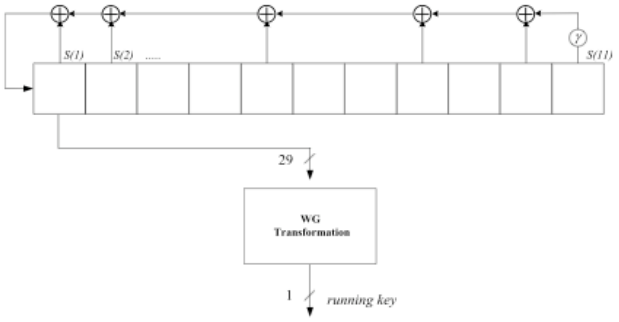

Figure 8: A Diagram of WG Generator

- IVs of 32, 64 bits and also the same lengths as the key are allowed
- Estimated strength 2^{128} (exhaustive search)
- Based on Welch-Gong (WG) transformation sequences, well studied in sequence design for communications
- The keystream possesses the cryptographic properties of WG sequences

The WG cipher is a filtering generator, which is shown in Figure 8.

A. Description of 29-bit WG Cipher

The LFSR generates an m-sequence over the extension field $GF(2^{29})$ of degree 11. And then the elements of m-sequence are filtered by a WG transformation: $GF(2^{29}) \to GF(2)$, to produce a keystream sequence. The 29-bit WG transformation is given by

$$WG(x) = Tr(t(x+1)+1) \text{ where } t(x) = x + x^{q_1} + x^{q_2} + x^{q_3} + x^{q_4}$$

where

$$q_1 = 2^{19} + 2^9 + 1, \quad q_2 = 2^{19} - 2^9 + 1$$
$$q_3 = 2^{19} + 2^{10} - 1, \quad q_4 = 2^{10} + 1$$

(see [**?**] for the formulae of q_i's). WG transformation operations can be implemented using the normal basis in the finite field $GF(2^{29})$. The 29-bit WG transform is shown in Figure 9.

The implementation of the WG is shown in Figure 10 in which the operator $\times$ is the multiplier over $GF(2^{29})$ under the normal basis, the operator $+$ is bit wise addition, and $\pm$ is modulo 2 addition of 29 input bits (29 bit XOR gate).

B. Key Initialization

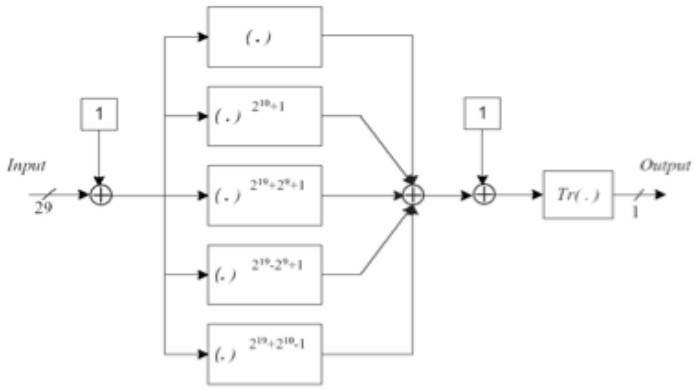

Figure 9: A Diagram of the 29-bit WG Transformation

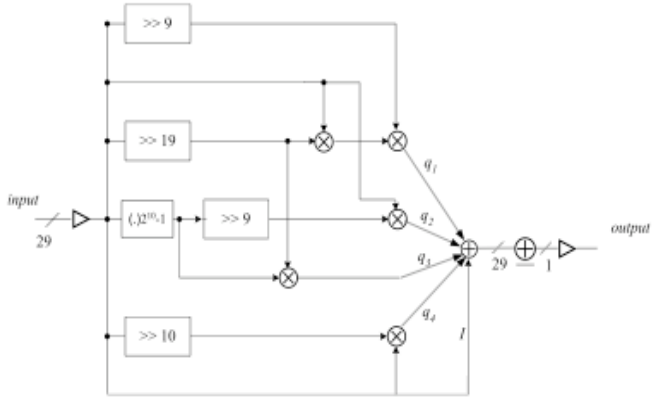

Figure 10: A Diagram of Implementation of 29-bit WG Transformation

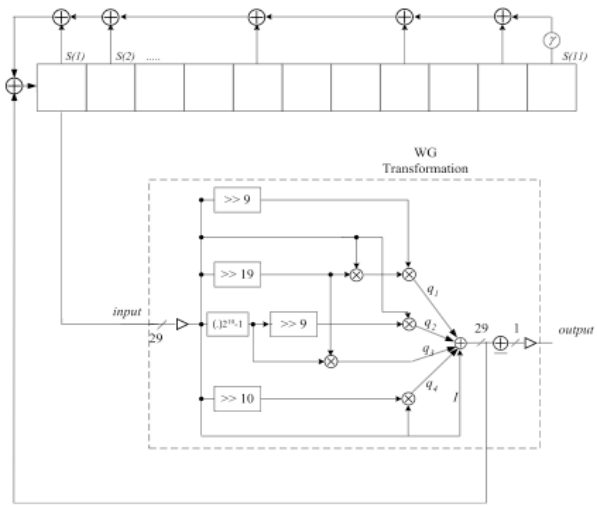

Figure 11: A Diagram of Key Scheduling Algorithm of 29-bit WG

An initial state of the LFSR contains 339 bits where each register holds 29 bits. For a 128-bit key and 128-bit IV (initial vector), the rule for loading to the LFSR is as follows.

Registers	29-bit Format
0,2,4,6,8	16 bits from the key||8 bits from IV||padding zeros
1,3,5,7,9	8 bits from the key||16 bits from IV||padding zeros
10	8 bits from the key||8 bits from IV||padding zeros

In the above table, $x||y = (x_0, \cdots, x_{t-1}, y_0, \cdots, y_{r-1})$, the concatenation of two vectors $x = (x_0, \cdots, x_{t-1})$ and $y = (y_0, \cdots, y_{r-1})$. Once LFSR has been loaded with the key and IV, the key stream generator is run for 22 clock cycles. This is the key-initialization phase of the cipher operation. During this phase the 29 bit vector of the output of the WG transformation is added to the feedback of the LFSR which is then used to update the LFSR, which is shown in Figure 11.

C. Security of the Cipher

The WG stream cipher produces a key stream sequences with good correlation and large span. The cipher is resistant to all the known attacks. We list those randomness properties as follows.

- Randomness Properties of Keystream
  - Period is $2^{319} - 1$
  - Balanced

  - 2-level autocorrelation
  - Ideal t-tupledistribution ($1 \leq t \leq 11$)
  - Linear complexity $\approx 2^{45.04}$

- Cryptographic Properties of WG Transformation
  - 1-order resilient
  - Algebraic degree 11
  - Nonlinearity $= 2^{28} - 2^{14} = 268419072$
  - Additive autocorrelation between $f(x+a)$ and $f(x)$ has three values: $0, 2^{15}$
  - 1-order propagation

Security against Known Attacks

- Time/Memory/Data Tradeoff Attacks: size of internal state is 2^{319}. Thus this attack is not applicable.
- Algebraic Attacks: the number of linear equations $\approx \begin{pmatrix} 319 \\ 11 \end{pmatrix}$; and the attack complexity $\approx 2^{182}$.
- Correlation Attacks: WG transformation is 1-order resilient, and nonlinearity is very high $2^{28} - 2^{14}$. Thus the correlation between WG transformation and any linear or affine function is very small.

Thus we have the following features of WG stream ciphers:

- Guaranteed keystreamrandomness properties.
- Secure against time/memory/data tradeoff attacks, algebraic and correlation attacks.
- Can be implemented in hardware with reasonable complexity.

2.2 Grain 2

The stream cipher Grain is a filtering sequence for which a filtering function is applied to two FSRs where one is an LFSR and the other is an NLFSR. The filtering function is the sum of two functions where one is a nonlinear function in 5-variables which takes four taps from the LFSR and one from

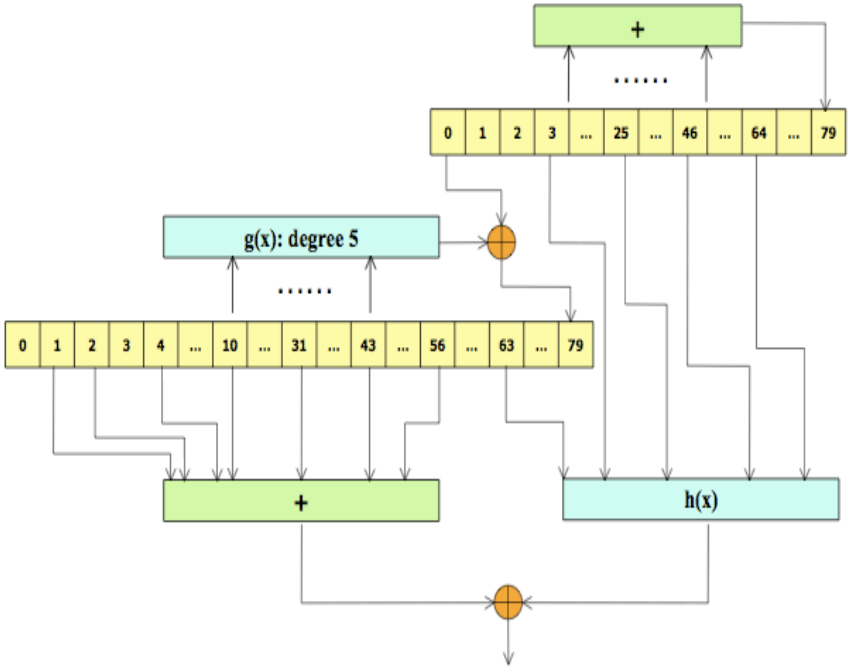

Figure 12: A Diagram of Grain 2 Stream Cipher

the NLFSR, and the other is a linear function which takes 7 taps from the NLFSR. In other words, the output is the sum of the output of the nonlinear function and the sum of the sequences from 7 tap positions at the NLFSR. Both LFSR and NLFSR have 80 stages. A diagram of Grain 2 is shown in Figure 12.

A. Description of Grain 2 Key Stream Generator

1. Let $\mathbf{a} = \{a_i\}$ be the output of the LFSR with the characteristic polynomial $t(x)$

$$t(x) = x^{80} + x^{62} + +x^{51} + x^{38} + x^{23} + x^{13} + 1.$$

The linear recursive relation is given by

$$a_{i+80} = a_{i+62} + a_{i+51} + a_{i+38} + a_{i+23} + a_{i+13} + a_i, i \geq 0.$$

2. Let $\mathbf{b} = \{b_i\}$ be the output of the NLFSR with the following feedback boolean function $g(\underline{x})$

where $\underline{x} = (x_0, x_1, \cdots, x_{79})$.

$$\begin{aligned} g(x_0, x_1, \cdots, x_{79}) &= x_{62} + x_{60} + x_{52} + x_{45} + x_{37} + x_{33} + x_{28} + x_{21} \\ &+ x_{14} + x_9 + x_0 + x_{63}x_{60} + x_{37}x_{33} + x_{15}x_9 + x_{60}x_{52}x_{45} + x_{33}x_{28}x_{21} \\ &+ x_{63}x_{45}x_{28}x_9 + + x_{60}x_{52}x_{37}x_{33} + x_{63}x_{60}x_{21}x_{15} \\ &+ x_{63}x_{60}x_{52}x_{45}x_{37} + x_{33}x_{28}x_{21}x_{15}x_9 + x_{52}x_{45}x_{37}x_{33}x_{28}x_{21}. \end{aligned}$$

The feedback bit of the NLFSR is masked by the output of the LFSR, i.e., the recursive relation is given by

$$b_{i+80} = a_i + g(b_i, b_{i+1}, \cdots, b_{i+79}), i = 0, 1, \cdots.$$

3. The filtering function $f(x_0, x_1, \cdots, x_{11}) = h(x_0, x_1, x_2, x_3, x_4) + \sum_{j=5}^{11} x_j$ where $h(x)$ is given by

$$\begin{aligned} h(x_0, x_1, x_2, x_3, x_4) &= x_1 + x_4 + x_0x_3 + x_2x_3 + x_3x_4 \\ &+ x_0x_1x_2 + x_0x_2x_3 + x_0x_2x_4 + x_1x_2x_4 + x_2x_3x_4. \end{aligned}$$

The tap positions are $(d_1, d_2, d_3, d_4, r_1, \cdots, r_8)$ where

$$\begin{aligned} &d_1 = 3, d_2 = 25, d_3 = 46, \text{ and } d_4 = 64 \text{ from the LFSR} \\ &r_1 = 63, \{r_i\}_{i=2}^{8} = \{1, 2, 4, 10, 31, 43, 56\} \text{ from the NLFSR.} \end{aligned}$$

Let $\mathbf{u} = \{u_i\}$ be the output of $h(x)$ whose elements are given by

$$u_i = h(a_{i+3}, a_{i+25}, a_{i+46}, a_{i+64}, b_{i+63}), i = 0, 1, \cdots,$$

i.e., four inputs from the LFSR and one from the NLFSR.

4. The output sequence of the generator, denoted by $\mathbf{s} = \{s_i\}$, is defined as

$$s_i = \sum_{j=2}^{8} b_{i+r_j} + u_i, i = 0, 1, \cdots$$

i.e., summation over sequences from seven tap positions of the NLFSR and the output of $h(x)$.

B. Key Initialization

Before any keystream is generated the cipher must be initialized with the key and the IV, the initial vector. Let the key $K = (k_0, k_1, \cdots, k_{79})$ (80-bits) and the $IV = (IV_0, IV_1, \cdots, IV_{63})$ (64-bits). The initialization of the key is done as follows. First load the key K as an initial state of the

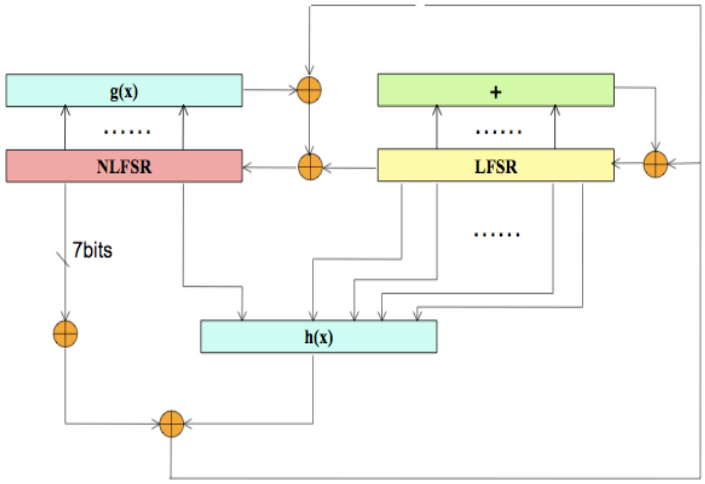

Figure 13: Key Initialization of Grain 2

NLFSR and load $T = IV||\underbrace{(1,\cdots,1)}_{14}$ as an initial state of the LFSR where $x||y$ is the concatenation of x and y. Then the generator is clocked 160 times without producing any running key. Instead the output function is fed back and xored with the input, both to the LFSR and to the NFSR, see Figure 13.

C. Property of the Key Stream

- The boolean function g is 2-resilient since x_0, x_{14}, x_{62} occur only in the linear terms.
- Filtering function is bent in 5-variables with nonlinearity 12 which is maximum.
- Period of the output sequence is at least $2^{80} - 1$.
- The exhaustive search space is 2^{80}.

Notes

Security of a stream cipher is determined by randomness of key stream generators, see Appendix A for those discussions. Descriptions on A5/1, w7, E0 and RC4 are documented in [2] [14] [4] [13], respectively. For cryptanalysis and attacks on those stream ciphers, see [3] for A5/1, [1] [5] for E0, and [10] [6] [12] for RC4. An extension of RC4 is proposed in [7], which is the fastest software based stream cipher currently known. The two candidates submitted to eSTREAM, WG and Grain 2, can be found in [8], and a WG family is discussed in [11].

References

[1] F. Armknecht and M. Krause, Algebraic attacks on stream combiners with memory, *Advances in CryptologyCRYPTO 2003*, Lecture Notes in Computer Science, No. 2729, pp. 162-176, Springer-Verlag, 2003.

[2] R. Anderson, and M. Roe. A5, 1994. Available at http://jya.com/crack-a5.htm

[3] A. Biryukov, A. Shamir and David Wagner, Real time cryptanalysis of A5/1 on a PC, *Fast Software Encryption 2000,* B. Schneier (Ed.), LNCS, vol.1978, Springer-Verlag, 2001, pp. 1-18.

[4] Bluetooth CIG, Specification of the Bluetooth system, Version 1.1, February 22, 2001. Available from www.bluetooth.com.

[5] Nicolas Courtois, Fast algebraic attacks on stream ciphers with lienar feedback, *Advances in Cryptology-Crypto'2003*, Lecture Notes in Computer Science, vol. 2729, pp. 176-194, Springer-Verlag, 2003.

[6] I. Mantin. Predicting and distinguishing attacks on RC4 keystream generator. *Advances in Eurocrypt2005*, LNCS, vol. 3494, Springer-Verlag, 2005, pp. 491-506.

[7] G. Gong, K. C. Gupta, M. Hell, andY. Nawaz, Towards a general RC4-like keystream generator, SKLOIS Conference on Information Security and Cryptology (CICS05), December15-17, Beijing, China. Springer-Verlag, 2006. (Download: http://comsec.uwaterloo.ca.)

[8] eSTREAM - *The ECRYPT Stream Cipher Project*, http://www.ecrypt.eu.org/stream/

[9] I. Mantin and A. Shamir. A practical attack on broadcast RC4, *Fast Software Encryption* 2001. *LNCS*, vol. 2355, Springer-Verlag, 2001, pp. 152-164.

[10] S. Mister and S. Tavares. Cryptanalysis of RC4-like Ciphers. *SAC '98, vol. 1556 of LNCS, pp. 131-143, Springer-Verlag, 1999.*

[11] Y. Nawaz and G. Gong, WG: A family of stream ciphers with designed randomness properties, *Information Sciences*, Vol. 178, No. 7, April 1, 2008, pp. 1903-1916.

[12] S. Paul and B. Preneel. A New Weakness in the RC4 Keystremeam Generator and an Approach to Improve the Security of thr Cipher. Fast Software Encryption 2004. *Vol. 3017 of LNCS, pp. 245-259, Springer-Verlag, 2004.*

[13] R. Rivest, RC4, 1987, and reverse engineered in 1994.

[14] A. Thomas, T.A. Berson, D.J. Anthony and G. Gong, System and method for securing a communication channel over an optical network, http://www.ietf.org/ietf/1id-abstracts.txt, October 2002.