In The Name of God

Fruit: ultra-lightweight stream cipher with shorter internal state

Vahid Aminghafari, Honggang Hu
Key Laboratory of Electromagnetic Space Information, Chinese Academy of Sciences,
University of Science and Technology of China, Hefei, China, 230027
vahidaming@mail.ustc.edu.cn, hghu2005@ustc.edu.cn

Abstract

In eSTREAM project a few lightweight stream cipher for hardware was introduced (2008) and then in FSE 2015 Sprout was proposed. Sprout introduced a new idea, design of stream cipher with shorter internal state by using key not only in initialization but also in keystream generation, but it was insecure. Fruit stream cipher is successor of Grain and Sprout stream ciphers that we show is secure and ultra-lightweight cipher. Internal state of Fruit is only 80 bits and also length of key and IV is 80 bits for 80-bit security. It is noticeable that internal state size is equal to amount of security while for resistance stream cipher against Time-Memory-Data trade-off attack, internal state should be at least twice of security level. For compensate of this we use some new ideas in design.

Keywords: Stream ciphers, Lightweight, Grain, Sprout, dynamic initialization, Cryptographic Primitive

Introduction

Nowadays need to secure lightweight symmetric cipher is obviously more than eSTREAM project time (this is provable by a lot of paper in design and cryptanalysis of light cipher). WSN and RFID and mobile phone are instances lead us to accept important of design new and secure lightweight ciphers.

Three Stream ciphers have been introduced in the hardware profile of portfolio of eSTREAM project. They are Trivium and MICKEY 2.0 and Grain-v1 [7, 8, 9]. Grain-v1 use NFSR and LFSR together, the linearity section would guarantee good statistical properties and guarantee about period, while the nonlinearity part would protect against attacks that can be mounted against a linear cryptosystem. Grain-128 was introduced in 2006 [10] and some attacks proposed to it [11, 12, 13, 14, 15, 16, 17]. Indeed Grain-128 was not secure as expected (such as Grain-v1). Grain-128a [18] was proposed in 2011. Although some attacks have been applied to Grain-128a [19, 20, 21], due to practicality point of view, still it is stand-up.

Sprout is stream cipher with shorter internal state that was introduced in FSE 2015 [1]. A short while after Sprout was introduced many attacks was published against it [3, 6, 4, 23, 2, 5]. Although it has become clear that Sprout is insecure, but it had a great new idea that help to design stream cipher with smaller area size. The new idea is to effect secret key not only in initialization but also during key generation. Actually this idea help to extend internal state to secret key. Due to in the most of application we should save key for reuse by different IV, the idea help to us have bigger internal sate (therefor we can design stronger ciphers). On the other hand, we need to save key in a fix memory in some applications (in this case one fix key is sufficient for ever) and it is known that save fix bits need less area size in compare to save bits in temporary memory (e.g. bourn fix key in fuse). Thus we can design stream ciphers with shorter internal state [1]. Fix key was not used in a suitable way in the design of Sprout and in the papers about cryptanalysis of Sprout mentioned that design of stream cipher (by the new idea) is too hard [3, 5] and in other paper, authors told it is fascinate [23] and other authors predict very soon secure cipher will be proposed by this new idea [2].

Necessary condition for stream ciphers to be resistance against time memory data trade-off attack is internal state size that should be at least twice of his security (while secret key only use in initialization stage). For example in Trivium and MICKEY 2.0 and Grain-v1, portfolio of eSTREAM project, we can see this fact. But we know that in some application (such as RFID, WSN) there are less resource and we need new stream cipher with minimal internal sate. We think that this is new generation in design of stream cipher and we define ciphers with less than 900 GE (gate equivalents) ultra-lightweight stream ciphers (Sprout is ultra-lightweight stream cipher).Here we propose another reduced internal state stream cipher, Fruit, that is successor of Grain and Sprout that we will show is secure and ultra-lightweight cipher. Really in nature after a while grain becomes sprout and in finally we have fruit. We think that Grain and Sprout were new generation in design of stream ciphers and now Fruit is mature in stream ciphers.

In continue of the paper, we explain about design of Fruit and next design criteria. Then we show Fruit is resistance to known attacks. Finally we tell about hardware implementation of Fruit.

Design Details of Fruit

Internal state consists of 43-bit LFSR ($l_t, \dots, l_{t+42}$), 37-bit NFSR ($n_t, \dots, n_{t+36}$), 14-bit counter, C ($c_t^0, \dots, c_t^{13}$). Inputs for cipher are 80-bit secret key, K (k_i ,$0 \leq i \leq 79$) and 80 bits public Initial Value, IV (v_i ,$0 \leq i \leq 79$). Note that maximum number of stream bits that can be produced from one key and IV is 2^43 bits and every key should use less than 2^15 times with different IVs. It is not acceptable to reuse IV, i.e. use identical IV with different keys.

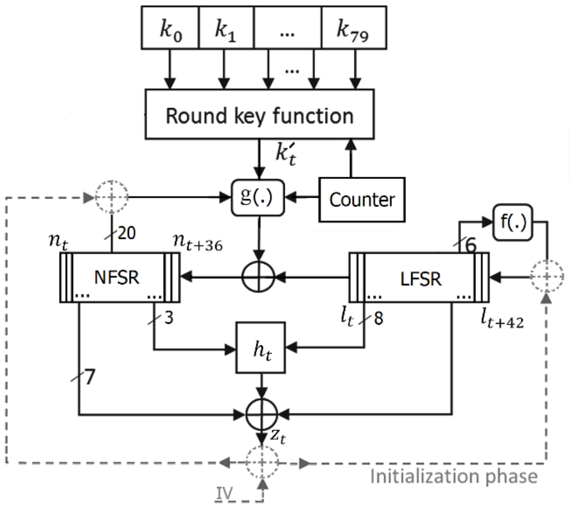

Fig. 1: Block Diagram of Fruit

Now we explain every part of cipher in detail:

-**Round key function**: we separate 80 bits key to 3 strings such as follow.

First key string = k_0 ,k_1 ,k_2 ,…, k_{15}

Second key string = k_{16} ,k_{17} ,k_{18} ,…, k_{47}

Third key string = k_{48} ,k_{49} ,k_{50} ,…, k_{79}

We combine 3 first bits of key strings to obtain first bit of round key, i.e. k_0'= $k_0 \oplus k_{16} \oplus k_{48}$. For next clock we should consider next bit in every key string in a cycle, i.e. k_1'= $k_1 \oplus k_{16} \oplus k_{48}$. We can produce round keys as follow. We need 14 bit registers for implementation of this function.

```
For $i = 0$ to $i = 15$
    For $j = 16$ to $j = 47$
        For $m = 48$ to $m = 79$
            $k_t' = k_i \oplus k_j \oplus k_m$
```

-**g function**: we use one bit from LFSR and 4 bits of the counter and k_t' and 20 bits of NFSR as a variables of g function for clocking of NFSR.

$$n_{t+37} = (k'_t \oplus 1) \,.\, (c_t^0 \oplus c_t^2 \oplus n_{t+10} \oplus n_{t+20} \oplus n_{t+12}.\, n_{t+3})$$
$$\oplus\, (k'_t) \,.\, (c_t^1 \oplus c_t^3 \oplus n_{t+6} \oplus n_{t+21} \oplus n_{t+11}.\, n_{t+16}) \oplus l_t \oplus n_t \oplus n_{t+14}.\, n_{t+25}$$
$$\oplus\, n_{t+8}.\, n_{t+18} \oplus n_{t+5}.\, n_{t+23}.\, n_{t+31} \oplus n_{t+28}.\, n_{t+30}.\, n_{t+32}.\, n_{t+34}$$

-f function: feedback function in LFSR is primitive. Thus maximum string will be produced by following function.

$$l_{t+43} = l_t \oplus l_{t+8} \oplus l_{t+18} \oplus l_{t+23} \oplus l_{t+28} \oplus l_{t+37}$$

-h function: this function produce pre-output stream from LFSR and NLFSR sates.

$$h_t = n_t.\, l_{t+15} \oplus l_{t+1}.\, l_{t+22} \oplus n_{t+35}.\, l_{t+27} \oplus n_{t+33}.\, l_{t+11} \oplus l_{t+6}.\, l_{t+33}.\, l_{t+42}$$

-running key: output will be produced by 7 bits from NLFSR and 1 bit from LFSR and output of h function.

$$z_t = h_t \oplus n_{t+1} \oplus n_{t+7} \oplus n_{t+13} \oplus n_{t+19} \oplus n_{t+24} \oplus n_{t+29} \oplus n_{t+36} \oplus l_{t+38}$$

-Initialization of cipher: We split bits of the key into 4-bits and XOR first 10 segment together and also last 10 segment together. Therefor we have two 4-bits word, I and I' as follow (I and I' are 4-bit digit).

$$I = \oplus_{i=0,4,8,12,16,20,24,28,32,36}\ (k_i k_{i+1} k_{i+2} k_{i+3})$$
$$I' = \oplus_{i=40,44,48,52,56,60,64,68,72,76}\ (k_i k_{i+1} k_{i+2} k_{i+3})$$

Now we extend IV to $I + I' + 82$ bits by concatenating two vectors to the first and last bits of IV. we add one number 1 and I number 0 to the first of IV and I' number 0 and one number 1 to the end of IV and we call it $IV'(v'_i, 0 \le i \le I + I' + 81)$ as follow.

$$IV' = \ 10 \ldots 00 v_0 v_1 v_2 \ldots v_{77} v_{78} v_{79} 00 \ldots 01$$

In initialization round, key bits is loaded to NFSR and LFSR respectively from LSB to MSB (k_0 to n_0 , k_1 to n_1 …and k_{37} to l_0 , k_{38} to l_{41} …). We clock $I + I' + 82$ times the cipher and before each clock, XOR output with IV' bits and also feedback output stream to NFSR and LFSR, i.e. $z_i \oplus v'_i$, $0 \le i \le I + I' + 81$ (as show in fig. 1). Now, we put $l_t = 1$, $t = I + I + 82$ for preventing all zeroes in LFSR. Then cipher should clock 80 times without feedback output in LFSR and NFSR (i.e. during last 80 clocks we disconnect feedback of z_t to LFSR and NFSR). Thus, the cipher doesn't produce any output stream in the $I + I' + 162$ initial clocks. Note that for initial clocks we use 8 bits of free counter registers.

Design criteria

-Limitation for producing output stream: due to length of LFSR we can produce 2^43 bits in maximum in every initialization (period of internal state is at least 2^43 bits). We think that 1 terabyte is almost sufficient for all application because our cipher is special for hardware application (e.g. WSN and RFID).

-**Round key function**: we produce 2^{14} different key from original key. Attacker can (with guessing internal states and known output keystream) obtain some of k'_t, but due to the number of initial clock is unknown (dynamic), it is not to easy solve linear equations system.

-g function: The function that produces n_{t+37} is in 20 variables of NFSR due to light implementation in hardware in compare to Grain-v1 and Sprout. If we suppose $k'_t = c_t^0 \oplus c_t^2 \oplus l_t = 0$, the nonlinearity of g function will be $2^8 \times 3760$ and resiliency 2. Variables for high degree term is chosen from n_t with $t > 27$ that cause the degree of variables very soon hit maximum possible degree in NFSR.

-f function: feedback polynomial is primitive, so the period of produced string by LFSR with non-zero initial is maximum. Due to after fed back output bit to LFSR we put l_0=0, we are sure that period of LFSR and NFSR is at least 2^43 bits. Some attacks was proposed to Grain and Sprout from this weakness [22, 23].

-Output function: The nonlinearity of h function is 976. We add 8 linear terms in order to increase nonlinearity to 249856 and also to make function with 7 resiliency. The best linear approximation of output function has 8 terms with $2^{-5.415}$ bias.

Note that we use n_{t+36} and $n_t$in output function for preventing produce keystream in next clock and previous clock with unknown k'_t.

Resistance against known attacks

Security level of fruit is 80 bits, thus here we show that computational complexity of some main attacks is more than exhaustive search attack to 80 bits.

-Linear Approximation Attack: this attack was applied to Grain-v0 [24]. In [24] discussed that if NFSR and output function choose with high nonlinearity and suitable resiliency, it will be strength to Linear

Approximations attack. We choose NFSR and output function with high nonlinearity and good resiliency and also in Fruit a nonlinear function of key is involved on NFSR. Therefor our cipher is strength to this attack.

-Guess and determine attack: due to shorter internal sate in Sprout and Fruit, this attack is very important to us (Sprout was weak against this attack [23]). If attacker guess all bits of internal sate in the Sprout, he can clock 2 times forward and one time backward (with unknown key) and in every clock he can decrease the wrong candidates of internal sate. In the next clocks attacker obtain one bit of key or decrease the wrong candidates of internal sate. We strengthen round key function and use n_{t+36} and $n_t$in output function for preventing produce keystream in next clock and previous clock with unknown k'_t and we use more bits of counter in internal state (to efficiently increase internal state size). If attacker can obtain some of k'_t, due to unknown number of initial clocks (the number of initial clocks depend on all bits of key) it is too hard attacker can solve linear equations and obtain key. Thus, Fruit is resistant against this attack.

-Time-Memory-Data Trade-off Attack: in literature is famous that if the size of internal state is not at least twice of security level, cipher is weak to this attack. In Fruit we use fix key as an internal state and also some bits of counter, therefor there is no problem from this view.

-Related-key Attack: There is weakness in initialization process of Grain-128a [21] and Sprout [3]. Designers of Sprout ruled out related key attack. They believe that due to key is fix in ultra-lightweight ciphers, this attack is not workable on Sprout [1]. Nevertheless we propose new scheme in initialization stage to strength against this attack. We do not load IV bits directly in internal sate and so do not combine IV and key bits straightforward together. Also we use asymmetric padding with IV and dynamic number of clock in initialization, thus we are sure there is not weakness to this attack.

-Cube attack: due to the dynamic number of clocks in initialization stage and suitable clock number, it is too hard to find any low degree multiplicative expression (of some bits of IV) base on key in Boolean function of output. Therefor our design is resistant against to all type of Cube attack.

-Algebraic attack: this attack have not been applied to Grain family but combination of this attack was applied to Sprout [4]. Short internal state (or we can tell weak round key function) in Sprout has been caused this weakness. We strengthen round key function and use more bits of counter in internal state, so we think Fruit in more resistant.

Hardware implementation cost

Design of lightweight cipher is very important in industries while we need to light ciphers in many filed such as communication and WSN and RFID and etc. due to stream ciphers are lighter than block ciphers and very lighter than public ciphers, it is obvious that design of ultra-lightweight stream cipher is very important. Our goal was design strength cipher with less than 900 GE. We use gate count for fair compare of hardware implementation cost between Grain-v1 and Sprout and Fruit proposed in [10, 18]. Due to hardware implementation cost is dependent on many factors such as hardware used in implementation and techniques of implementation and etc., and also due to it is not possible in clear way to show provable results of implementation, we compare gate count of ciphers according to Table 1 [10, 18]. Note that this compare is not consist of all parts such as multiplexers needed in order to switch between key/IV loading, initialization, but it consist of main parts of every ciphers. We don't dedicate any gate to key bits, because in every cipher should save key for reuse with different IVs.

Table 1. The gate count used for different functions

Function	Gate Count
NAND2	1
NAND3	1.5
NAND4	2
XOR2	2.5
2-1 MUX	5
Flip flop	8

Table 2. The compare of estimated gate count for implementation of Grain-v1 and Sprout and Fruit

	Grain-v1	Sprout	Fruit
LFSR	8*80	8*40	8*43
NFSR	8*80	8*40	8*37

f function	2.5*5	2.5*5	2.5*5
g function	75	56.5	49.5
Output	53	38	38
Round key function	0	85.5	117
etc.	64#	0	0
Total gate count	1484.5	832.5	857

#: Related to counter in initialization stage

Conclusion

Fruit stream cipher in compare with Grain-v1 is very lightweight in hardware implementation and due to Grain-v1 is lightest candidate in eSTREAM project, it is known that design of secure stream ciphers such as Fruit is very interesting. We show that Fruit unlike Sprout cipher is secure with some new idea. In most application of symmetric cipher secret key (for reuse with other IVs) should save (in a memory) and here we show how we can exploit it in the design. We prove that suitable use of fixed secret key as an internal state cause to save in area size while we can design secure cipher.

Acknowledgement

This work has been supported by CAS-TWAS President's Fellowship for International PhD students.

References

[1] Armknecht, F., Mikhalev, V.: On lightweight stream ciphers with shorter internal states. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 451–470. Springer, Heidelberg (2015)
[2] Esgin, M.F., Kara, O.: Practical Cryptanalysis of Full Sprout with TMD Tradeoff Attacks. http://eprint.iacr.org/2015/289.pdf
[3] Hao, Y.: A Related-Key Chosen-IV Distinguishing Attack on Full Sprout Stream Cipher. http://eprint.iacr.org/2015/231.pdf
[4] Maitra, S., Sarkar, S., Baksi, A., Dey, P.: Key Recovery from State Information of Sprout: Application to Cryptanalysis and Fault Attack. http://eprint.iacr.org/2015/236.pdf
[5] Zhang, B. ,Gong, X. : Another Tradeoff Attack on Sprout-Like Stream Ciphers, ASIACRYPT 2015, Part II, LNCS 9453, pp. 561–585, 2015
[6] Lallemand, V., Naya-Plasencia, M.: Cryptanalysis of full sprout. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology – CRYPTO 2015. LNCS, vol. 9215, pp. 663–682. Springer, Heidelberg (2015)
[7] De Canni`ere, C. Preneel, B. "Trivium - a Stream Cipher construction inspired by block cipher design principles. estream, ecrypt Stream Cipher" Technical report, of Lecture Notes in Computer Science.
[8] Babbage, S., Dodd, M. "The Stream Cipher MICKEY 2.0" ECRYPT Stream Cipher Project Report. http://www.ecrypt.eu.org/stream/ p3ciphers/mickey/mickey_p3.

[9] Hell, M., Johansson, T., Meier, W. “Grain—A Stream Cipher for constrained environments” New Stream Cipher Designs, 2008 Lecture Notes in Computer Science, vol. 4986, pp. 179–190, available at http://www.ecrypt.eu.org/stream/p3ciphers/grain/Grain_p3.pdf.

[10] Hell, M., Johansson, T., Maximov, A., Meier, W. “A Stream Cipher Proposal: Grain-128,” in International Symposium on Information Theory—ISIT 2006. IEEE, 2006

[11] Küçük, Ö, “Slide resynchronization attack on the initialization of Grain¨1.0,” eSTREAM, ECRYPT Stream Cipher Project, Report 2006/044, 2006, http://www.ecrypt.eu.org/stream.

[12] Dinur, I., Shamir, A. “Breaking grain-128 with dynamic cube attacks,” FSE 2011, Lyngby, Denmark, 2011, pp. 167–187.

[13] Dinur, I. ,Güneysu, T., Paar, C., Shamir, A., Zimmerman, R. “An experimentally verified attack on full grain-128 using dedicated reconfigurable hardware,” Proc. ASIACRYPT 2011, Seoul, South Korea, 2011, pp. 327–343.

[14] J. M. Miodrag, G. Sugata, P. Goutam, and I. Hideki, “Generic cryptographic weakness of K-normal Boolean functions in certain stream ciphers and cryptanalysis of grain-128,” Periodica Mathematica Hungarica, vol. 65, no. 2, pp. 205–227, Dec. 2012.

[15] J. P. Aumasson, I. Dinur, L. Henzen, W. Meier, and A. Shamir, “Efficient FPGA implementations of high-dimensional cube testers on the stream cipher grain-128,” IACR Cryptology ePrint Archive, pp.218–218, 2009

[16] P. Stankovski, “Greedy distinguishers and nonrandomness detectors,” INDOCRYPT 2007, Hyderabad, India, 2010, pp. 210–226.

[17] S. Knellwolf, W. Meier, and M. Naya-Plasencia, “Conditional differential cryptanalysis of NLFSR-based cryptosystems,” in Proc. ASIACRYPT 2010, Singapore, 2010, pp. 130–145.

[18] M. Ågren, M. Hell, T. Johansson, and W. Meier, “A new version of Grain-128 with authentication,” in Proc. Symmetric Key Encryption Workshop, Lyngby, Denmark, Feb. 2011 [Online]. Available: http://skew2011.mat.dtu.dk/

[19] Banik, S., Maitra, S., Sarkar, S.: A differential fault attack on Grain-128a using MACs. SPACE 2012. LNCS, vol. 7644, pp. 111–125. Springer, Heidelberg (2012)

[20] Banik, S., Maitra, S., Sarkar, S.: A differential fault attack on the Grain family of stream ciphers CHES 2012. LNCS, vol. 7428, pp. 122–139. Springer, Heidelberg (2012).

[21] L. Ding and J. Guan, "Related Key Chosen IV Attack on Grain-128a Stream Cipher", IEEE Trans.Inform.Forensic Secur., vol. 8, no. 5, pp. 803-809, 2013.

[22] H. Zhang and X. Wang, “Cryptanalysis of stream cipher Grain family,” Cryptology ePrint Archive, Report 2009/109, 2009, http://eprint.iacr.org/.

[23] Banik Subhadeep., Some Results on Sprout. http://eprint.iacr.org/2015/327.pdf

[24] A. Maximov, “Cryptanalysis of the “Grain” family of stream ciphers”, in ACM Symposium on InformAtion, Computer and Communications Security (ASIACCS’06), 2006, pp. 283–288.