Scan Based Side Channel Attack on Grain v1

Sonu Kumar Jha
Applied Statistics Unit, Indian Statistical Institute
203 B. T. Road
Kolkata-700108, India
Email: jhasonu1987@yahoo.com

***Abstract*—In this paper we study a scan based side channel attack against the Grain family of stream ciphers. The attack works because scan chain test of circuits can be transformed into a powerful cryptographic attack due to the properties of scan based technique. So as a result the attack targets the test circuitry. We show how the attacker gains the knowledge about the locations of internal state bits of the NFSR and the LFSR and how he finds the secret key.**

Keywords- Scan-based side channel attack; Grain v1; LFSR; NFSR; Stream ciphers.

I. INTRODUCTION

Grain Version 1 is a stream cipher which was designed and submitted to eSTREAM [1] project by Martin Hell, Thomas Johansson and Willi Meier in 2005 [2]. It is a synchronous bit oriented stream cipher which is designed primarily for restricted hardware environments i.e. it is designed so as to require low hardware complexity. A more descriptive explanation of Grain family of stream ciphers will be given in the next section. A number of potential weaknesses in this stream cipher have been discovered and, as a result, analysis of Grain has become an area of interest for cryptologists. For detailed information on cryptanalysis of Grain, one may go through [3-5, 8-17, 18].

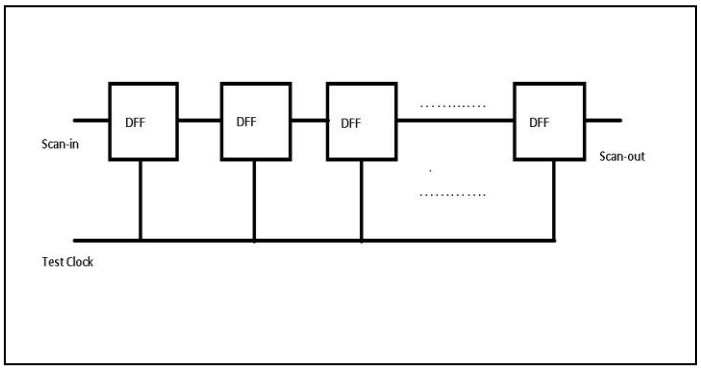

Figure 1. Design of a scan chain

Side channel attack is a class of cryptanalysis based on the information gained by the physical structure and implementation of a cryptosystem. Fault attacks, Timing attacks, Power monitoring attacks etc. are some general classes of side channel attacks. Scan based side channel attack can be very efficient in analysis of a cryptosystem because it can be performed in a very low cost and one can get the information about the internal state of the cryptosystem after getting the scan chain bits by scanning in the data through the scan-in pin into the system, running the system in normal mode and then scanning out the pattern chain. A simple scan chain is illustrated in Fig. 1. This method is generally used for testing purpose of a VLSI chip to find out the faulty registers inside the chip, if any. But one can see that this method is equally effective on the cryptanalysis point of view. Generally in IC's, Flip-Flops are connected as a chain of registers and the states of those FFs can be scanned out easily through that chain. Proceeding in this way, one can find out which bit in the scanned out chain corresponds to the state of which internal register of the cryptosystem. Once correspondence between the scanned out bits in the scan chain and the internal registers of the system is determined, one can find out the initial state of the cryptosystem and hence also the secret key. Some examples of scan chain analysis on stream ciphers are given in [6, 7]. The organization of the paper is as follows: In section 2 we give a description of the Grain family of stream ciphers, section 3 describes the scan chain based analysis of Grain v1 where as section 4 concludes the paper.

II. DESCRIPTION OF GRAIN FAMILY OF STREAM CIPHERS

Grain consists of an n-bit Linear Feedback Shift Register or LFSR, an n-bit Non-linear Feedback Shift Register or NFSR and an output function. An exact structure of Grain is explained in Fig. 2. The internal state of the cipher consists of these $2n$ bits. Keystream is produced by taking certain bits of both the shift registers as the input to a combining Boolean function. The content of the LFSR is denoted by $L = [l_0, l_1, \dots, l_{n-1}]$ and the content of the NFSR is denoted by $X = [x_0, x_1, \dots, x_{n-1}]$. The update function of the LFSR is given by the equation $l_{n-1} = f(L)$, where f is a linear function on the LFSR state bits obtained from a primitive polynomial in $GF(2)$ of degree n. The update function of

NFSR is given as $x_{n-1} = l_0 + g(X)$, where g is a non-linear function of the NFSR state bits.

The output keystream is produced by combining the LFSR and NFSR bits as $k = h'(X, L) = \bigoplus_{a \in A} x_a + h(X, L)$ where A is some fixed subset of $\{0,1,2, \ldots, n-1\}$.

A. Key Loading Phase

The Grain family uses an n-bit key K, and an m-bit initialization vector IV, with $m < n$. The key is loaded in the NFSR and the IV is loaded in the 0^{th} to $(m-1)^{th}$ bits of the LFSR. The remaining m^{th} to $(n-1)^{th}$ bits of the LFSR are loaded with some fixed pad $P \in \{0,1\}^{n-m}$. Hence at this stage, the $2n$ bit initial stage is of form $K \parallel IV \parallel P$.

B. Key Scheduling Phase

After the key loading phase, for the first $2n$ clocks, the key stream produced at the output point of the function h' is XOR-ed to both the LFSR and NFSR update functions. The update equations are given as $l_{n-1} = k + f(L), x_{n-1} = l_0 + k + g(X)$.

C. Pseudo-Random keystream Generation Phase

After key scheduling is done, k is no longer added to the LFSR and the NFSR but is used as the pseudo-random keystream bit. So in this phase, the LFSR and the NFSR are updated as $l_{n-1} = f(L), x_{n-1} = l_0 + g(X)$.

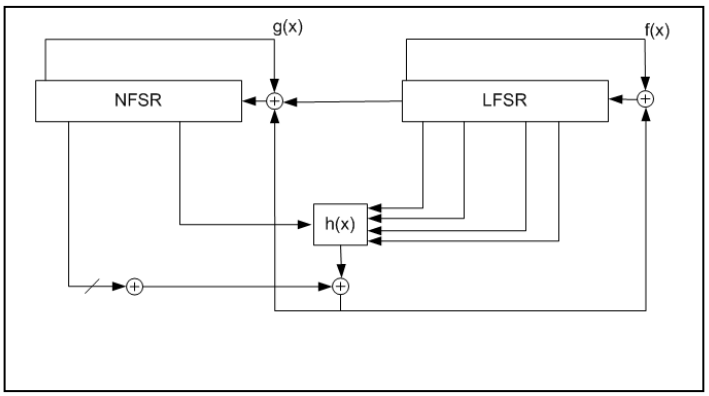

Figure 2. Structure of stream cipher in Grain family

It is notable that given any arbitrary state and the information about the number of clocks in phase B and phase C, one can backtrack to the corresponding state, say S_0^K, at the beginning of the phase B. This can happen because state update functions in both phases B and C are one-to-one and invertible.

In Grain v1, size of the key is $n = 80$ bits and size of the IV is $m = 64$ bits. The pad used in phase A is $P = 0xFFFF$. LFSR update rule is given as $l_{79} = l_{62} + l_{51} + l_{38} + l_{23} + l_{13} + l_0$. Update rule for NFSR is given as $x_{79} = l_0 + g(x_{63}, x_{62}, x_{60}, x_{52}, x_{45}, x_{37}, x_{33}, x_{28}, x_{21}, x_{15}, x_{14}, x_9, x_0)$ where,

$g(x_{63}, x_{62}, x_{60}, x_{52}, x_{45}, x_{37}, x_{33}, x_{28}, x_{21}, x_{15}, x_{14}, x_9, x_0) =$
$x_{62} + x_{60} + x_{52} + x_{45} + x_{37} + x_{33} + x_{28} + x_{21} + x_{14} + x_9 + x_0 + x_{63}x_{60} + x_{37}x_{33} + x_{15}x_9 + x_{60}x_{52}x_{45} + x_{33}x_{28}x_{21} + x_{63}x_{45}x_{28}x_9 + x_{60}x_{52}x_{37}x_{33} + x_{63}x_{60}x_{21}x_{15} + x_{63}x_{60}x_{52}x_{45}x_{37} + x_{33}x_{28}x_{21}x_{15}x_9 + x_{52}x_{45}x_{37}x_{33}x_{28}x_{21}$.

The output keystream is produced by combining LFSR and NFSR bits as $k = \bigoplus_{a \in A} x_a + h(l_3, l_{25}, l_{46}, l_{64}, x_{63})$, where $A = \{1,2,4,10,31,46,56\}$ and $h(s_0, s_1, s_2, s_3, s_4) = s_1 + s_4 + s_0s_3 + s_2s_3 + s_3s_4 + s_0s_1s_2 + s_0s_2s_3 + s_0s_2s_4 + s_1s_2s_4 + s_2s_3s_4$.

III. Scan Chain Based Analysis of Grain v1

In this section we will describe the scan based analysis on Grain v1. Grain v1 consists of an 80-bit LFSR and an 80-bit NFSR. The LFSR and the NFSR are updated by the feedback functions f and g respectively and the output keystream bit at each round is generated by an output function which is a function of certain locations from both the LFSR and the NFSR. Apart from this, the device representing the Grain v1 cryptosystem has an 8-bit counter for keeping track of the rounds. So in total, the number of scanned out bits will be $80 + 80 + 8 = 168$.

We break down the attack description in two steps. In the first step we show how the adversary obtains the knowledge of actual locations of the counter bits, the NFSR bits and the LFSR bits by observing the scanned out chain. Once the attacker gains the knowledge about the correspondence between the actual bits and the pattern he has scanned out, he then tries to obtain the secret key.

A. Deducing the bit correspondence

At first, the attacker tries to find the location of counter bits by looking at the scanned out pattern. In order to do so he exploits the Key-IV input pattern. As per design specifications of Grain and also the phase A discussed in section 2, the user needs to load the input as $K \parallel IV \parallel P$, where K is 80-bit key loaded in the NFSR, 64-bit IV is loaded in the LFSR and the rest of the bits are of the pad P. The attacker inputs the pattern of all zeros, i.e. 0^{160} or sequence of 160 zeros. The attacker now runs the system in normal mode for $(2^7 - 1)$ cycles. As the internal state of the cipher is set to all zeros, according to the update rule of the LFSR and the NFSR of Grain v1 discussed in section 2, running the system in normal mode will not bring any change in the internal state bits. The only thing which will change is the 8-bit counter. Since the attacker has run the system for $(2^7 - 1)$ clocks, 7 bits of the counter will be set to 1. The attacker then scans out the pattern and deduces that the bits which are 1 are the locations of the 7 counter bits in the chain. The attacker now resets the counter and scans in again the sequence of 160 zeros and runs the system for 2^7 clock cycles. He then scans out the resulting pattern. This time the MSB of the counter bits is set to 1 and hence the attacker gains the information about the original location of all the counter bits.

Now it is the time to gain the knowledge of original locations of the internal 160 state bits. Performing this task is straight forward. Observing the way in which the bits are circulated around the registers by looking at the update functions of Grain v1, we see that bits are shifted from right to

left after each cycle. We explain the general ideas used in the attack with the help of the following lemma:

Lemma 1. *Let x_m be the linear tap of the function g in the NFSR state update equation. If $x_m = 1$ and all other inputs to g are 0, then the output of g is 1.*

Proof. Update rule for NFSR is given as $x_{79} = l_0 + g(x_{63}, x_{62}, \dots, x_0)$, where the function g is defined in section 2.

For example x_{45} is a linear tap of the function g. If we set $x_{45} = 1$ and rest of the inputs of g to zero, then we have $g(0,0,0,0,1,0,0,0,0,0,0,0,0) = 1$. Similar argument holds for all linear taps.

Similar lemma holds for the update function f of the LFSR state update equation.

1. Correspondence of the NFSR state bits

Let x_m denote the NFSR state bit. Consider the following cases:

a. *x_m is a linear tap of the function g*

If we look at the state update equation of the NFSR for Grain v1 in section 2, we observe that the function g has certain linear taps. For example x_0,x_9, x_{14}, x_{21},x_{62} etc. are the linear taps of the function g. We also know that a left shift of bits occurs after each cycle in Grain v1. So in general, if x_m is the linear tap of the function g, and the attacker sets x_m to 1 and rest of the 159 bits to zero in the input pattern, then according to lemma 1, x_{79} will be set to 1 and due to the left shift operation after each cycle, x_{m-1} will also be set to 1. In case when $x_0 = 1$ is set by the attacker, only x_{79} is set to 1 after running the system for 1 cycle. Hence the attacker gets the bit correspondence of x_{79} after observing the scanned out pattern. So having the bit correspondence of x_{79} in hand, in cases of all other x_m set to 1 by him, where $m \neq 0$, he knows the correspondence of x_{m-1} after observing the scanned out pattern.

b. *x_m is not a linear tap of the function g*

This case is fairly straight forward. The attacker sets $x_m = 1$ and rest of the bits of the internal state registers to zero in his input pattern. He runs the system in normal mode for 1 clock cycle and then scans out the bit pattern and due to the left shift operation of the shift registers, x_{m-1} is set to 1 in the scanned out pattern. Hence he knows the bit correspondence of x_{m-1}.

2. Correspondence of the LFSR state bits

Let l_m denote the LFSR state bit. Consider the following cases:

a. l_0 is a linear tap of the functions g and f

Note that l_0 is the linear tap of both the functions f and g of the LFSR and the NFSR state update equations. If the attacker sets $l_0 = 1$ and rest of the bits to zero in his input pattern, then according to Lemma 1, after running the system for 1 clock cycle will set l_{79} and x_{79} to 1. The attacker knows the position of x_{79} already, hence he also finds out the bit correspondence of l_{79} after observing the scanned out pattern.

b. l_m is a linear tap of function f (where $m \neq 0$)

In this case, when the attacker sets $l_m = 1$ and rest of the bits to zero in the input pattern, then according to Lemma 1, l_{79} will be set to 1 after running the system for a clock cycle. Due to the left shift operation of the internal state registers after each clock cycle, l_{m-1} will also be set to 1. Since the attacker already knows the bit correspondence of l_{79}, he ascertains the bit correspondence of l_{m-1} too, after observing the scanned out pattern.

c. l_m is not a linear tap of the function f

Since l_m is not a linear tap in f, setting $l_m = 1$ and rest of the bits to zero in the input pattern, will set $l_{m-1} = 1$ after running the system for 1 clock cycle. So in this case attacker gets the bit correspondence of l_{m-1} after observing the scanned out pattern.

Following all the above cases, the attacker now has the knowledge of the bit correspondence of every state bit of the NFSR and the LFSR.

B. Finding the secret key

After the locations corresponding to all the NFSR and the LFSR bits are known, the attacker now attempts to find the secret key. He lets the cipher initialize with an unknown Key-IV. After 160 clocks he stops the normal mode of operation and scans out the contents of the device. Since the attacker already knows what position of the scanned out vector corresponds to which locations of the NFSR and the LFSR, he can perfectly reconstruct the internal state of the cipher after 160 rounds of the key scheduling phase (Phase B, section 2). Now all that remains to be done is to find the secret key from the knowledge of the internal state. He can do this by using the KSP^{-1} routine as described below.

Given the primitive polynomial of the Grain LFSR, the feedback function f is of the form $f(L) = l_0 + f'(L')$, where $L' = [l_1, \dots, l_{n-1}]$ is an $(n-1)$-bit vector obtained from L by removing the first term l_0. Similarly the update function g is of the form $g(X) = x_0 + g'(X')$ where $X' = [x_1, \dots, x_{n-1}]$ is an $(n-1)$-bit vector obtained from X by removing the first term x_0. This implies that the non-linear function g' does not depend on the term x_0. Similarly the linear function f' does not depend on the term l_0. This is necessary and sufficient condition for the state update functions of the NFSR and the LFSR to be one-to-one [19]. Due to this, the state update maps of the Grain family of ciphers during both the phase B and C described in section-2, are one-to-one and invertible, i.e. given any particular state, during any iteration of the phase B or C of section-2, it is possible to determine the previous state.

Given the NFSR and the LFSR state after the completion of phase B of section-2, Algorithm-1 will determine the NFSR and the LFSR state at the beginning of the phase B.

a. Algorithm-1: KSP^{-1}

Input: State $S_0 = x_0, \dots, x_{n-1}, l_0, \dots, l_{n-1}$
Output: State $S_0^K = x_0, \dots, x_{n-1}, l_0, \dots, L_{n-1}$
for $2n\ clocks$ **do**
$y_j = l_{n-1}$ and $n_j = x_{n-1}$
$l_i = l_{i-1}$ and $x_i = x_{i-1}$ for $i = n-1, n-2, \dots, 1$
$k = \bigoplus_{a \in A} x_a + h(x_0, \dots, x_{n-1}, l_0, \dots, l_{n-1})$
$l_0 = z + y_j + f'(l_1, \dots, l_{n-1})$
$x_0 = z + n_j + l_0 + g'(x_1, \dots, x_{n-1})$
end

Following the above algorithm, the attacker gets in possession with the secret key in case of Grain v1. Hence the attack on Grain v1 is successfully established.

IV. CONCLUSION

In this paper, we have shown that hardware designs of stream ciphers can be attacked when testing using scan chains. We have demonstrated such an attack on Grain v1 stream cipher which is in the hardware profile of the eSTREAM portfolio. We demonstrated the attack in two parts of which first part showed the methods of ascertaining the locations of the internal state bits of the cipher and the second part showed the algorithms to get in the possession of the secret key.

REFERENCES

[1] The ECRYPT stream cipher project. eSTREAM portfolio of stream ciphers. Revised on September 8, 2008.

[2] M. Hell, T. Johansson and W. Meier. Grain - A Stream Cipher for Constrained Environments. ECRYPT Stream Cipher Project Report 2005/001, 2005. Availableat http://www.ecrypt.eu.org/stream.

[3] J. P. Aumasson, I. Dinur, L. Henzen, W. Meier, and A. Shamir. E_cient FPGA Implementations of High-Dimensional Cube Testers on the Stream Cipher Grain-128.In SHARCS - Special-purpose Hardware for Attacking Cryptographic Systems,2009.

[4] C. Berbain, H. Gilbert and A. Maximov. Cryptanalysis of Grain. In FSE 2006,LNCS, Vol. 4047, pp. 15-29, 2006.

[5] A. Berzati et-al. Fault analysis of Grain-128 in: IEEE Workshop on Hardware Oriented Security and Trust, pp. 7-14, 2009.

[6] D.R. Chowdhury, A. Rijmen and A.das. Scan Based Side Channel Attacks on Stream Ciphers and Their Counter-Measure. In Indocrypt 2008,LNCS 5365, pp. 226-238, 2008.

[7] B.yang, K. Wu, R. Karri: Scan based side channel attack on dedicated hardwarre implementations of data encryption standard. In: ITC 2004: Proceedings of the International Test Conference,Washington. DC, USA, PP. 339-344. IEEE Computer Society,Los Alamitos (2004).

[8] Subhadeep Banik, Subhamoy Maitra, Santanu Sarkar: A Diffrential Fault Attack on Grain family of Stream Ciphers. In CHES 2012: 122-139.

[9] Subhadeep Banik, Subhamoy Maitra, Santanu Sarkar: A differential Fault Attack on Grain family under Reasonable Assumptions. In INDOCRYPT 2012: 191-208.

[10] Subhadeep Banik, Subhamoy Maitra, Santanu Sarkar: Some Results on Related Key-IV Pairs of Grain. SPACE 2012: 94-110.

[11] Subhadeep Banik, Subhamoy Maitra, Santanu Sarkar: A Diffrential Fault Attack on Grain-128a using MACs. SPACE 2012: 111-125.

[12] T.E. Bjorstad. Cryptanalisis of Grain using time/memory/data tradeoffs (v 1.0/2008-02-25). Available at http://www.ecrypt.eu.org/stream.

[13] C. De Canniere, O. Kucuk and B. Preneel. Analysis of Grain initialization algorithm. In AFRICACRYPT 2008,LNCS vol 7073,pp. 276-289,2008.

[14] I. Dinur, T. Guneysu,C. Paar, A. Shamir,R. Zimmerman. An experimentally verified attack of full Grain-128 using dedicated reconfigurable hardware.In ASICRYPT 2011, LNCS vol. 7073, pp. 327-343,2011.

[15] I. Dinur , A. Shamir. Breaking Grain-128 with dynamic cube attacks. In FSE2011, LNCS,Vol. 6733,pp.167-187,2011.

[16] H. Englund, T. Johansson and M S Turan. A framework for chosen IV statistical analysis of stream ciphers. In INDOCRYPT 2007,LNCS,VOl.4859,pp. 268-281,2007.

[17] S. Fischer, S. Khazaei and W. Meier. Chosen IV statistical analysis for key recovery attacks on stream ciphers. In AFRICACRYPT 2008,LNCS, Vol. 5023, pp. 236-245,2008.

[18] S. Knellwolf, W.Meier and M. Naya-Plasencia. Conditional differential cryptanalysis of NLFSR- Based cryptosystems. In ASIACRYPT 2010, LNCS,Vol. 6477,pp.130-145,2010.

[19] H. Fredricksen.A survey of full length non-linear shift register cycle algorithms, SIAM Rev.,24(1982),pp.195-221, 1982.