Guess-then-algebraic attack on the Self-Shrinking Generator

Blandine Debraize, Louis Goubin

Lausanne, February 12, 2008

Outline

1. Introduction
   - The Self-Shrinking Generator
   - Methods to Solve Algebraic Systems
   - Guessing Information

Outline

Outline

1. Introduction
   - The Self-Shrinking Generator
   - Methods to Solve Algebraic Systems
   - Guessing Information
2. Previous Work and Known Attacks
   - First Improved Attack
   - Mihaljević Attack
   - Hell-Johansson and Zhang-Feng Attack
3. Our Attack
   - First Method
   - Using More Keystream

Outline

Description of the self-shrinking Generator

SSG is :

- A pseudo random sequence generator
- Proposed by Meier and Staffelbach in 1994
- Derived from the Shrinking Generator
- Based on the irregular decimation of the output of one LFSR

Decimation principle:

LFSR sequence	01	11	10	00	01	11	10	00
		1	0			1	0	

When the first bit of the pair is 0, no output
when the first bit of the pair is 1, the second bit is the output

Algorithms to solve polynomial systems

Two main families

1. Linear algebra based systems:
   - Algorithms:
     - XL, XSL, T'
     - Gröbner Bases based algorithms (Buchberger, F4, F5).
   - No theory for non random systems.
   - Large matrices need huge memory.
2. SAT solvers, only for GF(2):
   - Recently proposed in algebraic cryptanalysis by Bard, Courtois and Jefferson.
   - Already used in cryptanalysis on Keeloq and Bivium.
   - One algorithm already used in crypto: MiniSAT.
   - No theory either.

SAT solvers Method

Method

- Converting the multivariate system into a CNF-SAT problem:
  - $a = xyz \Longleftrightarrow (x \vee \bar{a})(y \vee \bar{a})(z \vee \bar{a})(a \vee \bar{x} \vee \bar{y} \vee \bar{z})$
- Then applying a SAT-solver algorithm on it.
  - Choose a variable, try to assign it one value and then the other.
  - When some information is learned, new clauses are added to the system.

Important Parameters

- Number of clauses
- Total length of all the clauses
- Number of variables

Notations and Definitions

The length of the LFSR $\mathcal{L}$ is n, at clock t it outputs s_t.
The internal sequence at clock t is $S^t = s_0 s_1 \ldots s_t$.

Definition (Compression function)

C such that at clock t KG produces $C(S^t)$.
KG ouput sequence is $C(S^0)C(S^1)\cdots C(S^t)$.
The compression ratio η is the average number of keystream bits C outputs per internal bit.

Definition (Information Rate)

The keystream reveals about the first m bits of internal sequence the information rate per bit: $\alpha(m) = \frac{1}{m}\left(H(S^m) - H(S^m|Y)\right)$

First Attack on this type of PRNG

Method

Guess all the missing information.

Complexity

- For m output bits, the leakage of information given by the keystream is $\alpha m/\eta$.
- Then the entropy to recover m/η key bits is $H(S^m|Y) = (1-\alpha)\frac{m}{\eta}$.
- Final complexity $\mathcal{O}(2^{(1-\alpha)n})$.

On the SSG

- This is the first attack proposed on the SSG by Meier and Staffelbach.

How to improve this attack

Method and Complexity

- Decrease the amount of information we guess.
- Guess an amount of information h on the internal sequence per keystream bit, then the known information per keystream bit is $h + \alpha/\eta$.
- The ratio "guessed information" / "total information known per keystream" bit is

$$\frac{h}{h + \frac{\alpha}{\eta}}$$

- Final complexity of the guess is $\mathcal{O}(2^{\frac{h}{h+\frac{\alpha}{\eta}} n})$

Issue

Once the information is obtained, it has to be exploited to recover the key.

1 Introduction
- The Self-Shrinking Generator
- Methods to Solve Algebraic Systems
- Guessing Information

2 Previous Work and Known Attacks
- First Improved Attack
- Mihaljević Attack
- Hell-Johansson and Zhang-Feng Attack

3 Our Attack
- First Method
- Using More Keystream

4 Conclusion

First Improved Attack (Hell-Johansson 06)

Guess Method

- Instead of guessing all the internal bits, guess the even bits.
- It is equivalent to guessing the positions of the pairs $(1, e)$ in the internal sequence

Complexity

- The entropy per keystream bits for this information is $H(L) = \sum_{j=0}^{+\infty} \frac{j+1}{2^{j+1}} = 2$
- The complexity of the guess is then $\mathcal{O}(2^{\frac{2}{3}n})$
- The information is linear in the key bits, then a Gaussian elimination ($\mathcal{O}(n^3)$) is performed. Final complexity: $\mathcal{O}(n^3 2^{\frac{2}{3}n})$

Mihaljević Attack (96)

Method

- Look for the case when $\frac{n}{2}$ consecutive even internal bits are 1s.
- Then we know n internal bits.
- Time and Data complexity $\mathcal{O}(2^{\frac{n}{2}})$

Familly of attacks

Time/Data Tradeoff with

- Time complexity varying from $\mathcal{O}(2^{\frac{n}{2}})$ to $\mathcal{O}(2^{\frac{3}{4}n})$
- Data complexity varying from $\mathcal{O}(2^{\frac{n}{2}})$ to $\mathcal{O}(n)$ accordingly

Combining Attack [Hell-Johannson 06] and [Zhang-Feng 06]

Another tradeoff:

- Look for an internal sequence of length $l(\gamma)$ where the rate of 1s among the even bits is at least $\gamma > \frac{1}{2}$. l is computed such that it provides enough information (at least n bits).
- For each subsequence of length l guess the even bits compatible with rate of 1s $> \gamma$.
- Perform a Gaussian elimination on the linear equations provided by the known bits.
- Time complexity $\mathcal{O}(n^3 2^{\frac{n}{1+\gamma}})$.

Quadratic Attack

Method

- Still decrease the amount of information guessed.
- Instead of guessing the position of the even internal 1s, guess the position of one out of two.
- Consequence: if keystream sequence is $x_i, x_{i+1}, \cdots, x_{i+k}, \cdots$ we do not know the position of the internal pair $1x_{2i+1}$ but it ranges between pairs $1x_{2i}$ and $1x_{2i+2}$ positions.

Complexity of the Guess

- We guess size of "blocks" containing 2 even 1s.
- The entropy of the information guessed by keystream bit is: $H = -\frac{1}{2}\sum_{k\geq 0} \frac{\binom{k+1}{k}}{2^{k+2}} \log(\frac{\binom{k+1}{k}}{2^{k+2}}) \approx 1.356$
- The complexity of the guess is then $2^{\frac{1.356n}{1.356+1}} = 2^{0.575n}$

Quadratic Attack

Exploiting the information algebraically

Suppose the block contains k pairs beginning by 0. We have to describe the following information:

1. First and second bits of each block are known (linear)

Quadratic Attack

Exploiting the information algebraically

Suppose the block contains k pairs beginning by 0. We have to describe the following information:

1. First and second bits of each block are known (linear)
2. Only one pair among the remaining ones begins by 1:

Quadratic Attack

Exploiting the information algebraically

Suppose the block contains k pairs beginning by 0. We have to describe the following information:

1. First and second bits of each block are known (linear)
2. Only one pair among the remaining ones begins by 1:
   - There is at most one "1" among the even bits: $(s_{2i_j} = 1) \Rightarrow (s_{2i_l} = 0)$ gives $s_{2i_j} s_{2i_l} = 0$

Quadratic Attack

Exploiting the information algebraically

Suppose the block contains k pairs beginning by 0. We have to describe the following information:

1. First and second bits of each block are known (linear)
2. Only one pair among the remaining ones begins by 1:
   - There is at most one "1" among the even bits: $(s_{2i_j} = 1) \Rightarrow (s_{2i_l} = 0)$ gives $s_{2i_j} s_{2i_l} = 0$
   - There is at least one "1" among the even bits of the block: $\bigoplus_{j=1}^{k+1} s_{2i_j} = 1$

Quadratic Attack

Exploiting the information algebraically

Suppose the block contains k pairs beginning by 0. We have to describe the following information:

1. First and second bits of each block are known (linear)
2. Only one pair among the remaining ones begins by 1:
   - There is at most one "1" among the even bits: $(s_{2i_j} = 1) \Rightarrow (s_{2i_l} = 0)$ gives $s_{2i_j} s_{2i_l} = 0$
   - There is at least one "1" among the even bits of the block: $\bigoplus_{j=1}^{k+1} s_{2i_j} = 1$
3. The fact that the second bit e of the second pair beginning by "1" in the block is known : $(s_{2i_j} = 1) \Rightarrow (s_{2i_j+1} = e)$ equivalent to $s_{2i_j}(s_{2i_j+1} + e) = 0$.

Quadratic Attack

Exploiting the information algebraically

Suppose the block contains k pairs beginning by 0. We have to describe the following information:

1. First and second bits of each block are known (linear)
2. Only one pair among the remaining ones begins by 1:
   - There is at most one "1" among the even bits: $(s_{2i_j} = 1) \Rightarrow (s_{2i_l} = 0)$ gives $s_{2i_j} s_{2i_l} = 0$
   - There is at least one "1" among the even bits of the block: $\bigoplus_{j=1}^{k+1} s_{2i_j} = 1$
3. The fact that the second bit e of the second pair beginning by "1" in the block is known : $(s_{2i_j} = 1) \Rightarrow (s_{2i_j+1} = e)$ equivalent to $s_{2i_j}(s_{2i_j+1} + e) = 0$.

An amount of $\binom{k+1}{2} + k + 1$ quadratic equations and linear ones.

Quadratic Attack

Exploiting the information algebraically

- The system completely describes the key. But possible to find some other equations to make it overdefined.
- With SAT solvers, not very useful to generate overdefined systems.
- Results of the computations depends on the hamming weight of the feedback polynomial:

	$hw = 5$	$hw = 6$	$hw = 7$
$n = 128$	$0.02s$	$0.03s$	$0.05s$
$n = 256$	$0.025s$	$0.046s$	$62s$
$n = 512$	$0.127s$	$> 24h$	$> 24h$
$n = 1024$	$122.25s$	$> 24h$	$> 24h$

Generalization of the attack

Method

- Guess the position of one even internal one out of q.
- Entropy of this information by keystream bit is:
  $H(q) = -\frac{1}{q} \sum_{k \geq 0} \frac{\binom{q-1+k}{k}}{2^{q+k}} \log(\frac{\binom{q-1+k}{k}}{2^{q+k}})$.
- The complexity of the guess is then $2^{\frac{H(q)}{1+H(q)} n}$

Table: Average complexity of the guess for various values of q

	$q = 2$	$q = 3$	$q = 4$	$q = 5$
Complexity	$2^{0.575n}$	$2^{0.509n}$	$2^{0.458n}$	$2^{0.417n}$

Generalization of the attack

Exploiting the information algebraically

Suppose the block contains k pairs beginning by 0. We have to describe the following information:

1. First and second bits of each block are known (linear)
2. Exactly $q-1$ pairs among the remaining ones begins by 1:
   - $\binom{k-1}{q}$ degree q polynomials of the form $s_{2i_0}s_{2i_1}\cdots s_{2i_{q-1}} = 0$
   - One equation of degree $q-1$: $\sum s_{i_0}s_{i_1}\cdots s_{i_{q-2}} = 1$
3. The fact that each keystream bit e corresponding to this block follows an even 1 in the internal block is described by $\binom{k-1}{q-1}$ degree q equations of the form $s_{2i_0}s_{2i_1}\cdots s_{2i_{q-2}}(s_{2i_0+1} + e_0) = 0$.

Generalization of the attack

Exploiting the information algebraically

- If k is short, information can be described by lower degree equations.
- Also possible to find other equations.
- We fixed the Hamming weight of the feedback polynomial to 5.

Table: MiniSAT computations on quadratic systems of equations for q=3 and q=4

	$n = 128$	$n = 256$	$n = 512$
$q = 3$	$2.28s$	$80s$	$2716s$
$q = 4$	$14s$	$1728s$	$> 24h$

Method and Complexity

- Fix a value k and suppose each block contains at most k pairs beginning by 0.
- Compute the number of blocks l required to have all the necessary information.
- For each internal subsequence containing l blocks:
  - Guess the length of the l blocks.
  - Write the corresponding system of equations.
  - Solve the system by running MiniSAT on it.
- Time complexity of the guess: $\left(\frac{k-q+1}{\sum_{j=q}^{k}\frac{\binom{j-1}{q-1}}{2^j}}\right)^{\frac{n}{q+h}}$

  Data complexity: $\frac{1}{\left(\sum_{j=q}^{k}\frac{\binom{j-1}{q-1}}{2^j}\right)^{\frac{n}{q+h}}}$

Comparisons

Table: Total time complexity comparisons between Mihaljević attack, Hell *et al.* attack and our attack for the same data complexities

	$n = 256$				$n = 512$			
data	$2^{65.3}$	$2^{49.2}$	$2^{39.1}$	$2^{17.5}$	2^{128}	$2^{94.6}$	$2^{57.5}$	$2^{38.6}$
Miha	2^{145}	2^{152}	$2^{157.5}$	2^{174}	2^{288}	2^{302}	2^{322}	2^{336}
H-J, Z-F	$2^{160.2}$	$2^{164.8}$	$2^{167.8}$	$2^{176.4}$	2^{300}	$2^{308.3}$	2^{320}	2^{328}
Our att.	$2^{146.2}$	$2^{146.3}$	$2^{147.3}$	$2^{157.2}$	$2^{268.8}$	$2^{268.8}$	$2^{279.3}$	$2^{293.5}$

Conclusion

- New flexible attack on self-shrinking generator
  - When q increases, guess complexity decreases.
  - When k increases, data complexity decreases.
- Works only when the feedback polynomial hamming weight is low. In this case, it is the best Time/Data tradeoff.