Reduced Complexity Attacks on the Alternating Step Generator

Shahram Khazaei[1] Simon Fischer[2] Willi Meier[2]

[1]EPFL, Lausanne, Switzerland
[2]FHNW, Windisch, Switzerland

Selected Areas in Cryptography
Ottawa, Ontario, Canada
August 16 & 17, 2007

Outline

- Description of Alternating Step Generator (ASG)
- Previous Attacks on the ASG
  - Divide-and-Conquer Attacks
  - Reduced Complexity Attacks
- Our Contribution:
  - Closed Form Relations for previous Reduced Complexity Attacks
  - New Reduced Complexity Attack
- Summary

Outline

- Description of Alternating Step Generator (ASG)
- Previous Attacks on the ASG
  - Divide-and-Conquer Attacks
  - Reduced Complexity Attacks
- Our Contribution:
  - Closed Form Relations for previous Reduced Complexity Attacks
  - New Reduced Complexity Attack
- Summary

Outline

- Description of Alternating Step Generator (ASG)
- Previous Attacks on the ASG
  - Divide-and-Conquer Attacks
  - Reduced Complexity Attacks
- Our Contribution:
  - Closed Form Relations for previous Reduced Complexity Attacks
  - New Reduced Complexity Attack
- Summary

Outline

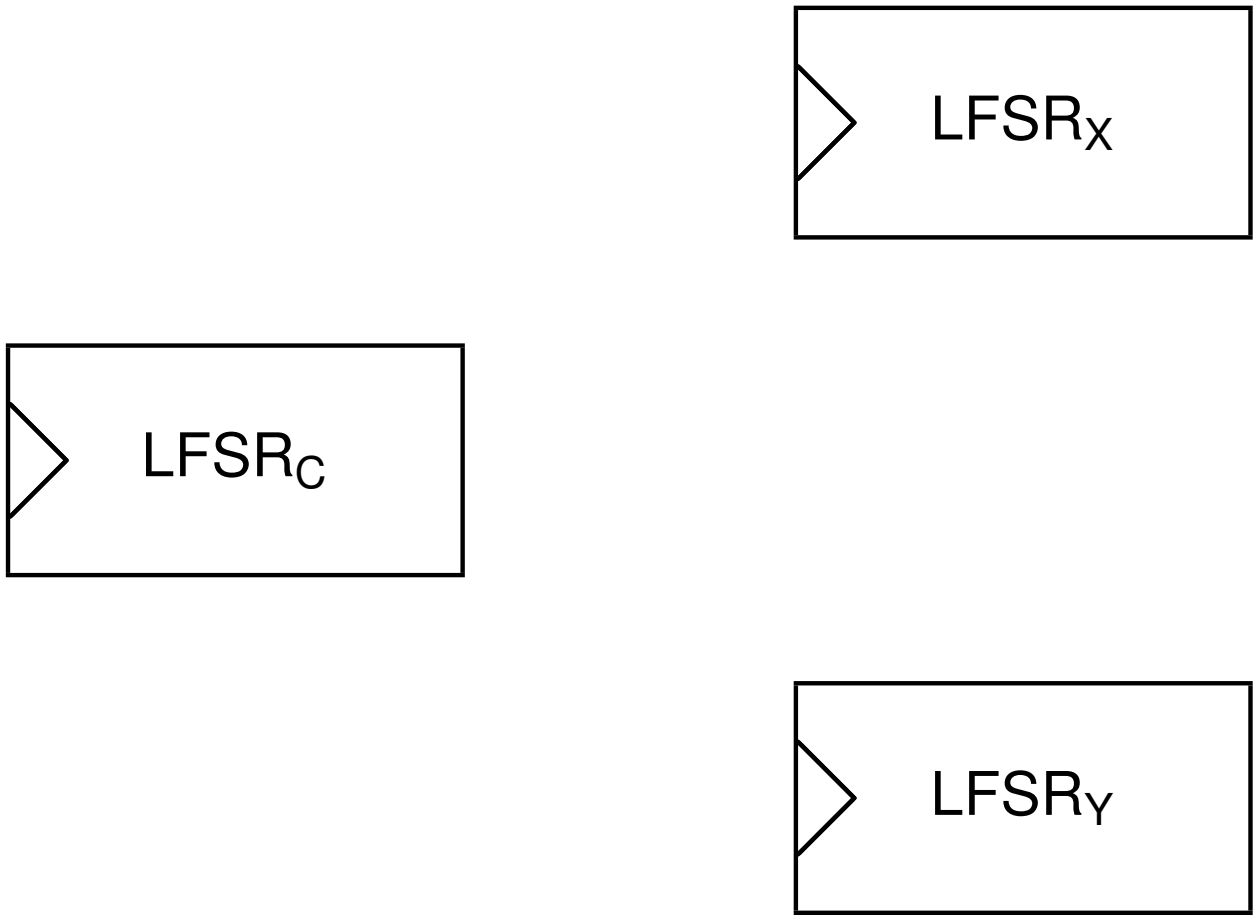
- Description of Alternating Step Generator (ASG)
- Previous Attacks on the ASG
  - Divide-and-Conquer Attacks
  - Reduced Complexity Attacks
- Our Contribution:
  - Closed Form Relations for previous Reduced Complexity Attacks
  - New Reduced Complexity Attack
- Summary

Outline

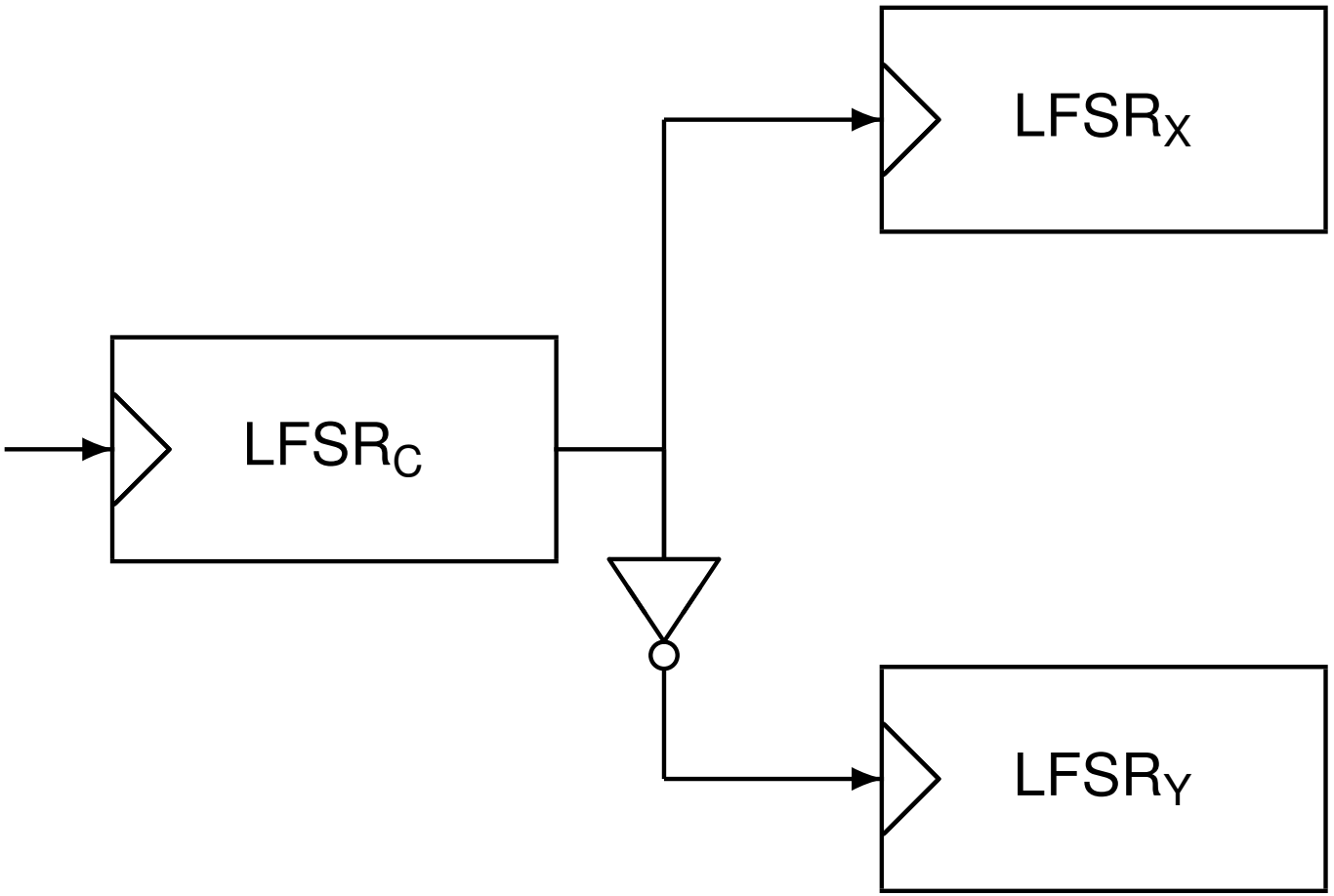
- Description of Alternating Step Generator (ASG)
- Previous Attacks on the ASG
  - Divide-and-Conquer Attacks
  - Reduced Complexity Attacks
- Our Contribution:
  - Closed Form Relations for previous Reduced Complexity Attacks
  - New Reduced Complexity Attack
- Summary

Part I
Description of ASG

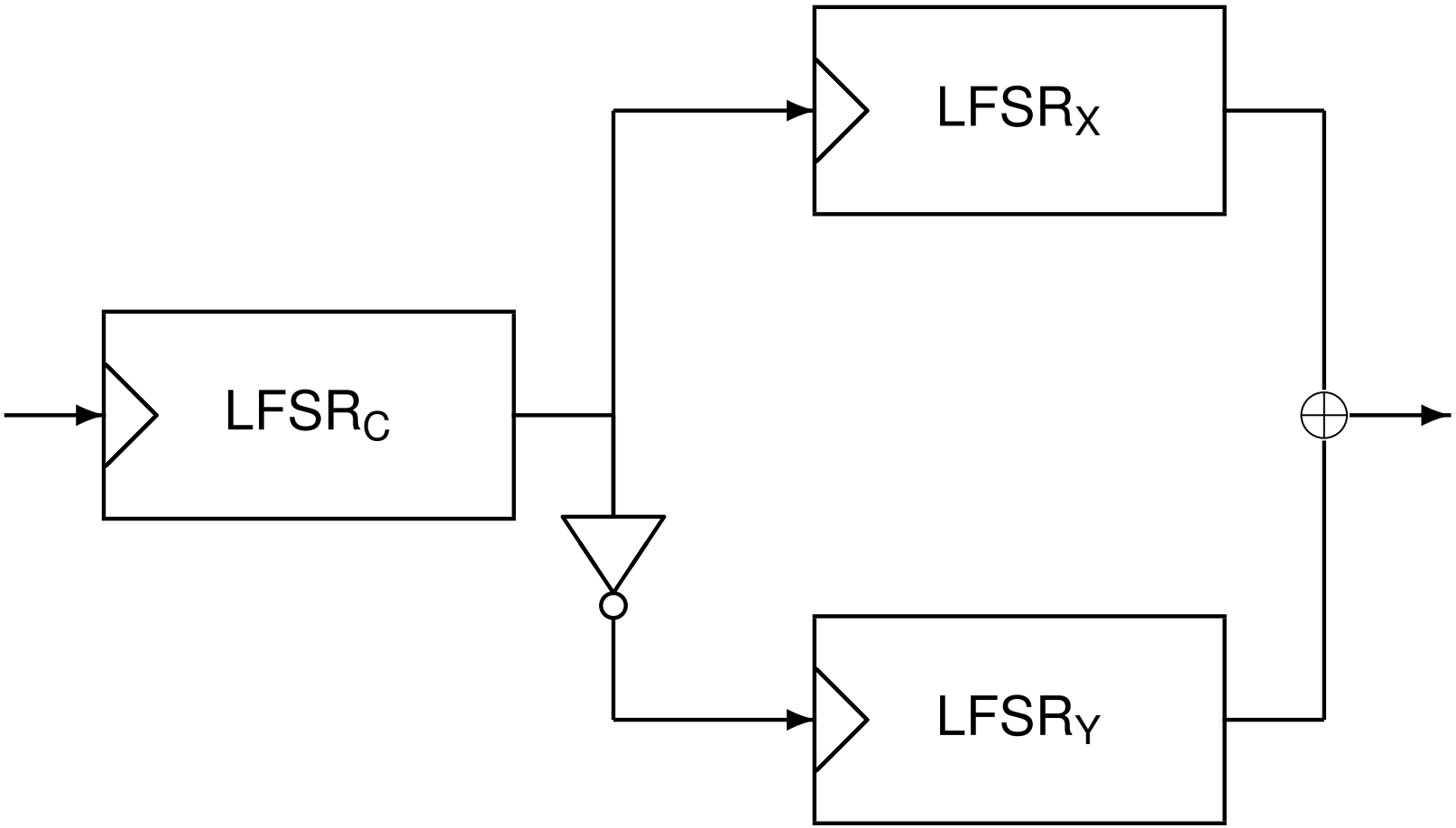
ASG (Original Description)



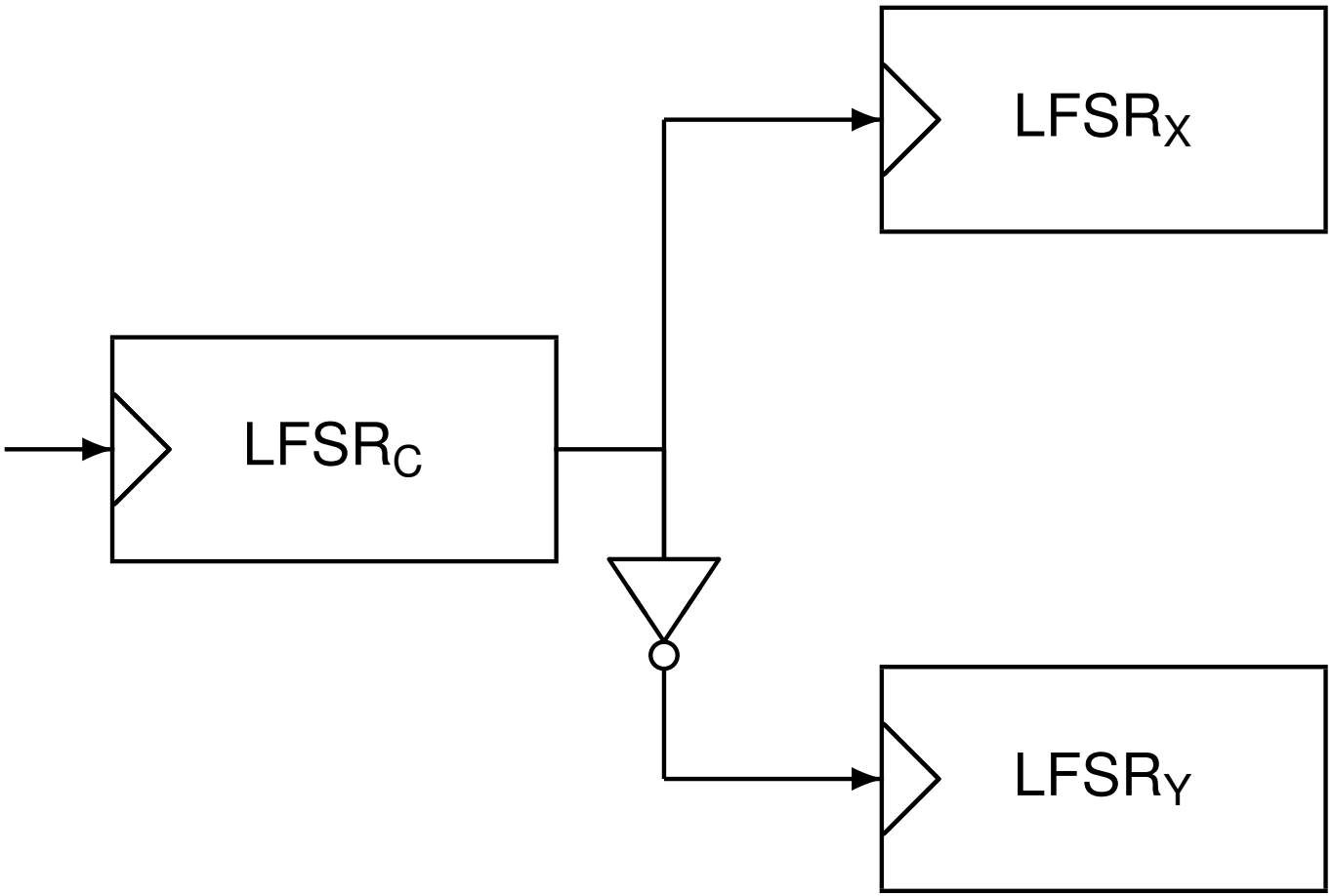
ASG (Original Description)



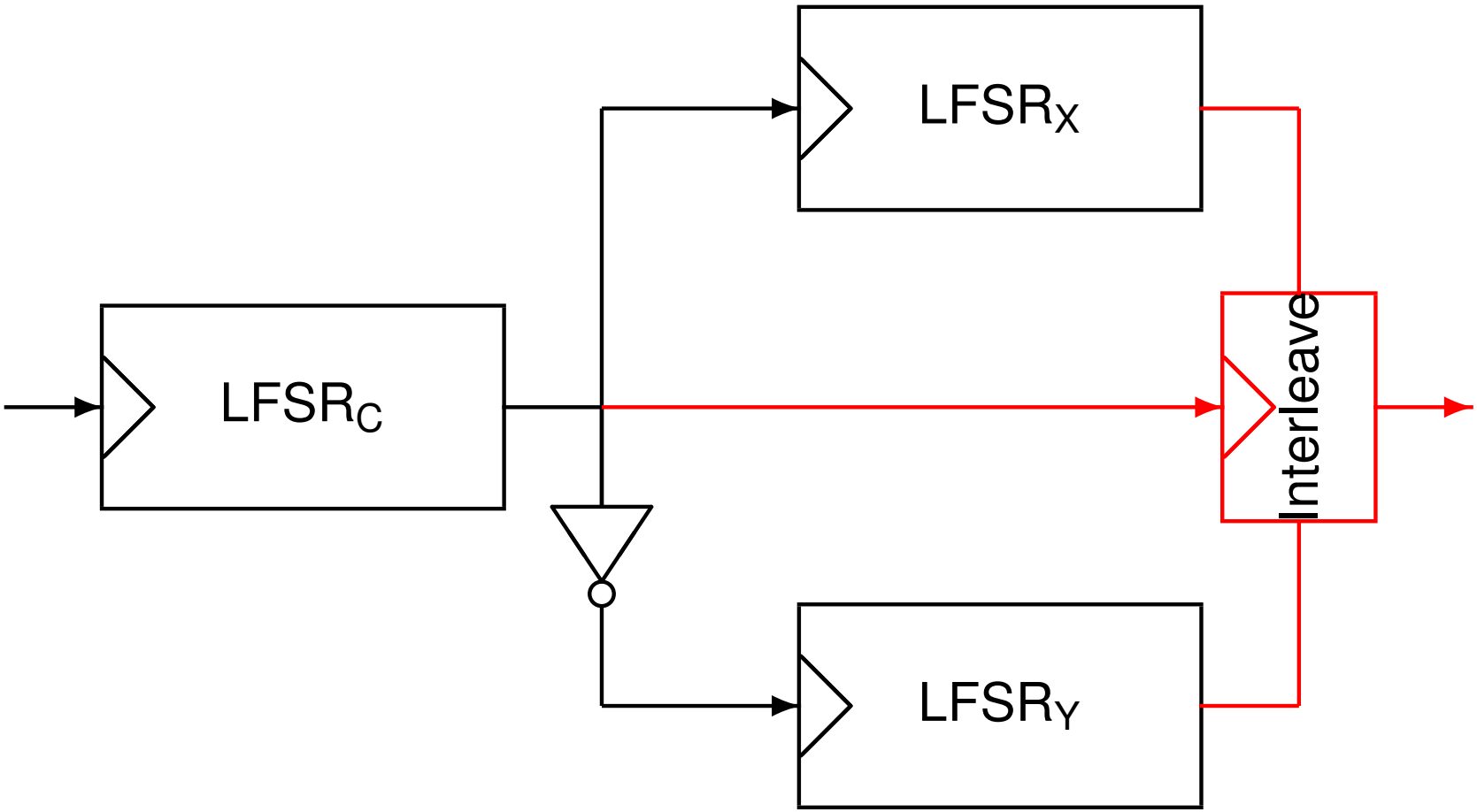
ASG (Original Description)



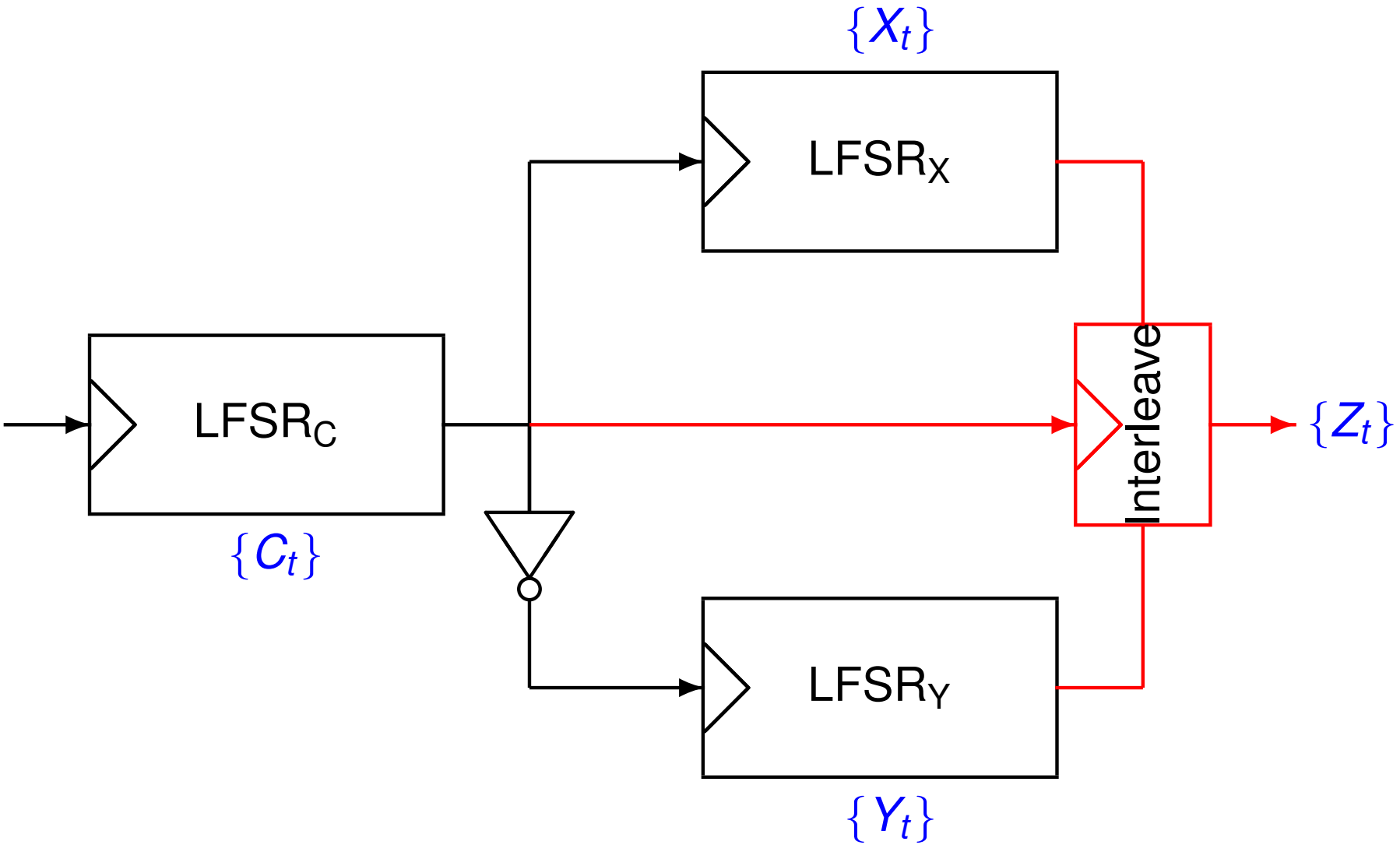
ASG (Another Description)



ASG (Another Description)



ASG (Another Description)



ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:	1	0	0	0	0	1	1	1	0	1	0	...
Y:	0	1	1	1	1	0	1	0	1	1	0	...
Z:												

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:	1	0	0	0	0	1	1	1	0	1	0	...
Y:	0	1	1	1	1	0	1	0	1	1	0	...
Z:												

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:	1	0	0	0	0	1	1	1	0	1	0	...
Y:		1	1	1	1	0	1	0	1	1	0	...
Z:	0											

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:	1	0	0	0	0	1	1	1	0	1	0	...
Y:		1	1	1	1	0	1	0	1	1	0	...
Z:	0											

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:		0	0	0	0	1	1	1	0	1	0	...
Y:		1	1	1	1	0	1	0	1	1	0	...
Z:	0	1										

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:		0	0	0	0	1	1	1	0	1	0	...
Y:		1	1	1	1	0	1	0	1	1	0	...
Z:	0	1										

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:			0	0	0	1	1	1	0	1	0	...
Y:		1	1	1	1	0	1	0	1	1	0	...
Z:	0	1	0									

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:			0	0	0	1	1	1	0	1	0	...
Y:		1	1	1	1	0	1	0	1	1	0	...
Z:	0	1	0									

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:			0	0	0	1	1	1	0	1	0	...
Y:			1	1	1	0	1	0	1	1	0	...
Z:	0	1	0	1								

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:			0	0	0	1	1	1	0	1	0	...
Y:			1	1	1	0	1	0	1	1	0	...
Z:	0	1	0	1								

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:			0	0	0	1	1	1	0	1	0	...
Y:				1	1	0	1	0	1	1	0	...
Z:	0	1	0	1	1							

ASG: Example

C:	0	1	1	0	0	1	0	0	0	1	1	...
X:						1	1	1	0	1	0	...
Y:							1	0	1	1	0	...
Z:	0	1	0	1	1	0	1	1	0	0	0	...

Part II
Previous Attacks on ASG

To simplify the notations we assume that
all registers have the same length L.

Previous Attacks on ASG

Attack	Target LFSR	Time Comp.	Data Comp.
Linear Consistency	C	$\mathcal{O}(L2^L)$	$\mathcal{O}(L)$
Edit Probability Correlation	X or Y	$\mathcal{O}(L^2 2^L)$	$\mathcal{O}(L)$

Previous Attacks on ASG
Reduced Complexity Attacks

Scenario I:
The attacker waits for a segment of M consecutive zeros (or ones) in the output sequence and assumes that:

- exactly $M/2$ of zeros come from LFSR_X
- Probability: $\beta = \binom{M}{M/2} 2^{-M}$

Previous Attacks on ASG
Reduced Complexity Attacks

Scenario I:
The attacker waits for a segment of M consecutive zeros (or ones) in the output sequence and assumes that:

- exactly $M/2$ of zeros come from LFSR_X
- Probability: $\beta = \binom{M}{M/2} 2^{-M}$

Attack	Target LFSR	Time Comp.	Data Comp.
Non-asymptotical	X or Y	$L^2 2^{L-M/2} \beta^{-1}$	$2^M \beta^{-1}/2$
Asymptotically optimum	X or Y	$\mathcal{O}(L^2 2^{\frac{2}{3}L})$	$\mathcal{O}(2^{\frac{2}{3}L})$

Previous Attacks on ASG

Reduced Complexity Attacks

Scenario II:

The attacker waits for a segment of length M containing at most ω ones (or zeros) and assumes that:

- half of the zeros come from the $\mathrm{LFSR_X}$
- all the ones and the remaining zeros come from the $\mathrm{LFSR_Y}$
- Probability: $\beta = 2^{-\omega}\binom{M-\omega}{(M-\omega)/2}2^{-(M-\omega)}$

Previous Attacks on ASG

Reduced Complexity Attacks

Scenario II:

The attacker waits for a segment of length M containing at most ω ones (or zeros) and assumes that:

- half of the zeros come from the LFSR_X
- all the ones and the remaining zeros come from the LFSR_Y
- Probability: $\beta = 2^{-\omega}\binom{M-\omega}{(M-\omega)/2}2^{-(M-\omega)}$

Attack	Target LFSR	Time Comp.	Data Comp.
Non-asymptotical	X or Y	$L^2 2^{L-(M-\omega)/2}\beta^{-1}$	$2^M\binom{M}{\omega}^{-1}\beta^{-1}/2$
Asymptotically optimum	X or Y	$\mathcal{O}(L^2 2^{0.69L})$	$\mathcal{O}(2^{0.64L})$

Correlation Based Attack

Proposed by Golic and Menicocci

The idea is to use the posterior probabilities of individual bits of the regularly clocked LFSR_X and LFSR_Y sequences, when conditioned on a given segment of the output sequence

$$p_i = \Pr\{x_i = 1 | Z^n\},\ 1 \leq i \leq n$$

These probabilities are quite far from 1/2.

Correlation Based Attack

Proposed by Golic and Menicocci

The idea is to use the posterior probabilities of individual bits of the regularly clocked LFSR_X and LFSR_Y sequences, when conditioned on a given segment of the output sequence

$$p_i = \Pr\{x_i = 1 | Z^n\},\ 1 \leq i \leq n$$

These probabilities are quite far from 1/2.

Our attack can be considered as a generalization of this idea.

Part III
Our Attack

Our Attack

Is a general framework applied to ASG

It benefits from:

- Low Sampling Resistance of the ASG
- Strong correlation between the output sequence and the initial state of the stop/go LFSR's

Sampling Resistance

The notion of sampling resistance was first introduced by Biham, Shamir and Wagner to speed up time/memory/data trade-off attacks on stream ciphers.

At FSE'07 it was noticed by Meier and Fischer that sampling may be useful along with other attacks in a unified framework.

Sampling Resistance

Any initial state (C^L, X^L, Y^L) of ASG which can produce Z^m is called a preimage of Z^m.

Sampling Resistance: 2^{-m} where m is the maximum value for which we can efficiently produce preimages of m-bit outputs

How to find T preimages of Z^m efficiently uniformly at random?

Sampling Algorithm for ASG

Input: Output sequence Z^m of m bits.
Output: Find $\mathcal{B}$ with T preimages of Z^m.

1: Initially, set $\mathcal{B} = \emptyset$.
2: **while** $|\mathcal{B}| < T$ **do**
3: choose a random non-zero initial states C^L
4: Set $\mathcal{X} = \mathcal{Y} = \emptyset$.
5: Compute C^m, a prefix of length m of the output sequence of $\mathrm{LFSR_C}$.
6: Based on C^m, split up Z^m into X^ω and $Y^{m-\omega}$, where $\omega = \mathrm{wt}(C^m)$.
7: Add all (non-zero) X^L to $\mathcal{X}$, if $\mathrm{LFSR_X}$ can generate X^ω.
8: Add all (non-zero) Y^L to $\mathcal{Y}$, if $\mathrm{LFSR_Y}$ can generate $Y^{m-\omega}$.
9: For all $X^L \in \mathcal{X}$ and $Y^L \in \mathcal{Y}$, add (C^L, X^L, Y^L) to the $\mathcal{B}$.
10: **end while**

Complexity of the Sampling Algorithm for ASG

Complexity of finding T preimages of a given Z^m is:

$$\begin{array}{cc} \mathcal{O}(T) & \text{if} \quad m \leq 2L \\ \mathcal{O}(T2^{m-2L}) & \text{if} \quad m > 2L \end{array}$$

Complexity of the Sampling Algorithm for ASG

Complexity of finding T preimages of a given Z^m is:

$$\begin{array}{ll} \mathcal{O}(T) & \text{if} \quad m \leq 2L \\ \mathcal{O}(T2^{m-2L}) & \text{if} \quad m > 2L \end{array}$$

The sampling resistance of ASG is about 2^{-2L}.

For a segment of length at most the total length of the two stop/go LFSR's, we can efficiently produce the preimages.

General idea of our attack:

For a given Z^m, find T preimages of Z^m hoping that it includes the correct initial state of the LFSR_X e.g.

General idea of our attack:

For a given Z^m, find T preimages of Z^m hoping that it includes the correct initial state of the LFSR_X e.g.

Question:

For a given output sequence Z^m, how large should T be so that our subset contains the correct initial state of LFSR_X?

General idea of our attack:

For a given Z^m, find T preimages of Z^m hoping that it includes the correct initial state of the LFSR_X e.g.

Question:

For a given output sequence Z^m, how large should T be so that our subset contains the correct initial state of LFSR_X?

The answer is related to the conditional distribution of different initial states of LFSR_X which can produce a given segment of length m of the output sequence of the ASG

Sample Size and Conditional Entropy

For a fixed initial state x of LFSR_X, let $p(x|Z^m)$ be the conditional probability that this initial state produces a given Z^m.

Sample Size and Conditional Entropy

For a fixed initial state x of $\mathrm{LFSR_X}$, let $p(x|Z^m)$ be the conditional probability that this initial state produces a given Z^m.

Using our sampling algorithm, we need to draw about $T = 2^{H_X}$ preimages of Z^m to include the correct initial state of $\mathrm{LFSR_X}$ where,

$$H_X(Z^m) = -\sum_x p(x|Z^m) \cdot \log_2 p(x|Z^m)$$

Description of the Attack

Input: Parameters T and m, and output Z^n ($n \approx 40L$).
Output: Recover the initial state of ASG with some success probability δ.

1: Given the segment Z^m, find T preimages using sampling algorithm.
2: Compute the edit probability between Z^n and the output sequence for each suggested initial state.
3: Choose the most probable candidates for $\mathrm{LFSR_X}$ resp. $\mathrm{LFSR_Y}$.
4: Recover $\mathrm{LFSR_C}$ and verify the validity.

Description of the Attack

Input: Parameters T and m, and output Z^n ($n \approx 40L$).
Output: Recover the initial state of ASG with some success probability δ.

1: Given the segment Z^m, find T preimages using sampling algorithm.
2: Compute the edit probability between Z^n and the output sequence for each suggested initial state.
3: Choose the most probable candidates for $\mathrm{LFSR_X}$ resp. $\mathrm{LFSR_Y}$.
4: Recover $\mathrm{LFSR_C}$ and verify the validity.

Description of the Attack

Input: Parameters T and m, and output Z^n ($n \approx 40L$).
Output: Recover the initial state of ASG with some success probability δ.

1: Given the segment Z^m, find T preimages using sampling algorithm.
2: Compute the edit probability between Z^n and the output sequence for each suggested initial state.
3: Choose the most probable candidates for $\mathrm{LFSR_X}$ resp. $\mathrm{LFSR_Y}$.
4: Recover $\mathrm{LFSR_C}$ and verify the validity.

Description of the Attack

Input: Parameters T and m, and output Z^n ($n \approx 40L$).
Output: Recover the initial state of ASG with some success probability δ.

1: Given the segment Z^m, find T preimages using sampling algorithm.
2: Compute the edit probability between Z^n and the output sequence for each suggested initial state.
3: Choose the most probable candidates for LFSR_X resp. LFSR_Y.
4: Recover LFSR_C and verify the validity.

Description of the Attack

Input: Parameters T and m, and output Z^n ($n \approx 40L$).
Output: Recover the initial state of ASG with some success probability δ.

1: Given the segment Z^m, find T preimages using sampling algorithm.
2: Compute the edit probability between Z^n and the output sequence for each suggested initial state.
3: Choose the most probable candidates for $\mathrm{LFSR_X}$ resp. $\mathrm{LFSR_Y}$.
4: Recover $\mathrm{LFSR_C}$ and verify the validity.

Description of the Attack

Input: Parameters T and m, and output Z^n ($n \approx 40L$).
Output: Recover the initial state of ASG with some success probability δ.

1: Given the segment Z^m, find T preimages using sampling algorithm.
2: Compute the edit probability between Z^n and the output sequence for each suggested initial state.
3: Choose the most probable candidates for $\mathrm{LFSR_X}$ resp. $\mathrm{LFSR_Y}$.
4: Recover $\mathrm{LFSR_C}$ and verify the validity.

Complexity of step 4 is $\mathcal{O}(2^{0.27L})$ (Golic and Menicocci)

Attack Complexity

Overall time complexity of the attack is about:

$$\begin{array}{ll} \mathcal{O}(L^2 2^H) & \text{if } m \leq 2L \\ \mathcal{O}(2^{H+m-2L}) & \text{if } m \gg 2L \end{array}$$

Parameters for the Conditional Entropy

What can affect H:

- the output prefix Z^m
- LFSR's feedback polynomials

Parameters for the Conditional Entropy

What can affect H:

- the output prefix Z^m
- LFSR's feedback polynomials

However, we noticed that the most influencing parameters are:

- L: the LFSR lengths
- m: the length of the output segment
- ω: the weight of the output segment Z^m or the weight ω of its first derivative

Examples:
Distribution of the Initial States

Setting: random feedback polynomials and $L = 20$

	Z^m
(I)	11101101101001010100001001001010110001 10

Z^m	m	ω	H_C	H_X	H_Y
(I)	40	21	17.49	17.32	17.34

Examples:
Distribution of the Initial States

Setting: random feedback polynomials and $L = 20$

	Z^m
(I)	11101101101001010100001001001010110001 10
(II)	11101101101001010100001001001010110001 1001

Z^m	m	ω	H_C	H_X	H_Y
(I)	40	21	17.49	17.32	17.34
(II)	42	22	16.26	16.46	16.45

Examples:
Distribution of the Initial States

Setting: random feedback polynomials and $L = 20$

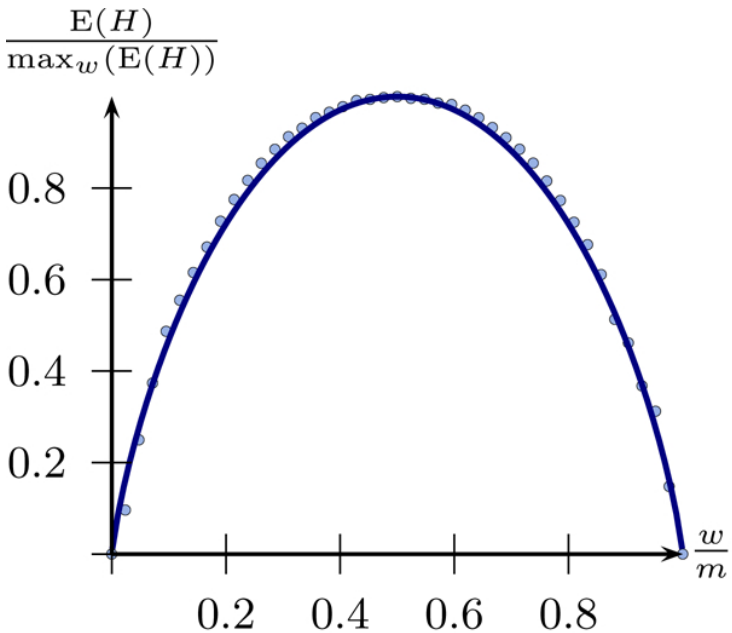
	Z^m
(I)	11101101101001010100001001001010110001 10
(II)	11101101101001010100001001001010110001 1001
(III)	00010100001000000001100000010001000000 00

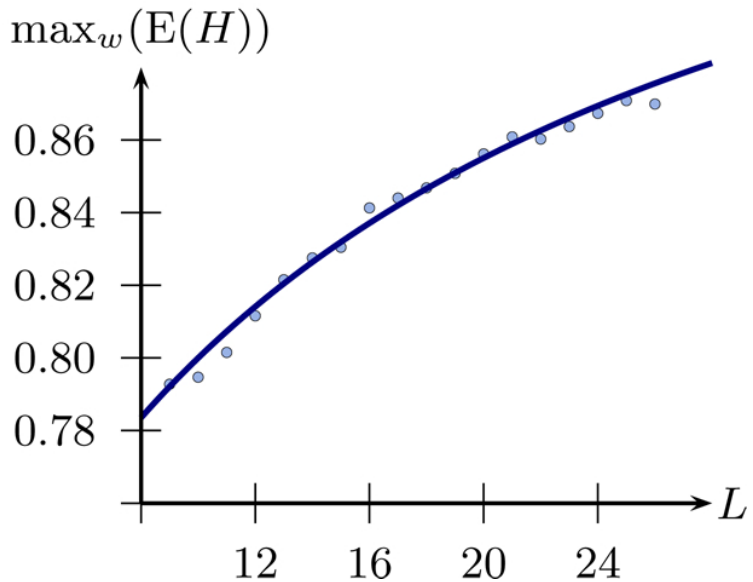
Z^m	m	ω	H_C	H_X	H_Y
(I)	40	21	17.49	17.32	17.34
(II)	42	22	16.26	16.46	16.45
(III)	40	7	17.39	12.24	12.63

Estimation of the Entropy
case $m = 2L$ and $\omega = \mathrm{wt}(Z^m)$

$$\mathrm{E}(H) \approx \gamma(L) \cdot L \cdot h\left(\frac{\omega}{m}\right)$$

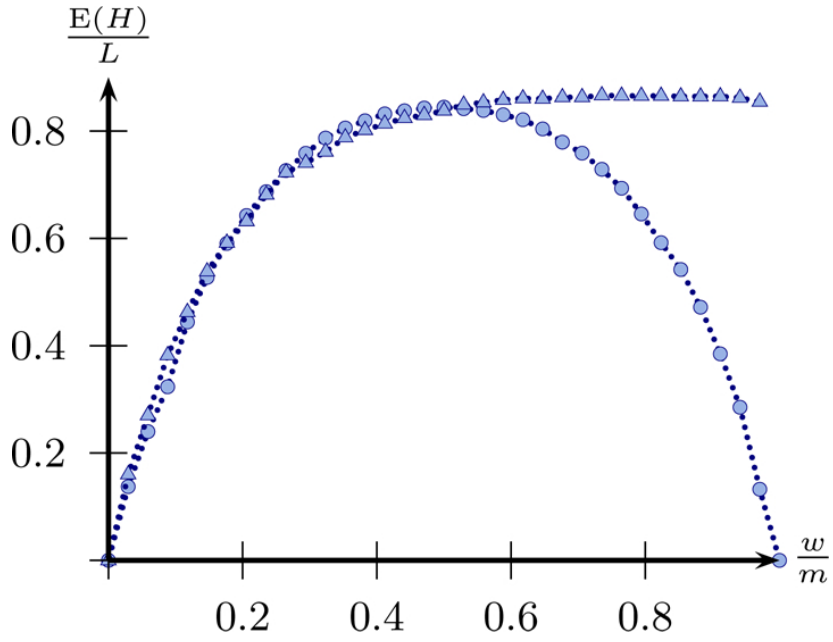
$$\gamma(L) \approx 1 - \frac{1}{0.19L + 3.1}$$





Estimation of the Entropy

$\omega = \mathrm{wt}(Z^m)$ versus $\omega = \mathrm{wt}(\dot{Z}^m)$



Time/Data Complexity of our Attack
case $m = 2L$

Using our estimation for the average value of the entropy:

$$\mathrm{E}(H) \approx \gamma(L) \cdot L \cdot h\left(\frac{\omega}{m}\right)$$

We have:

Attack	Target LFSR	Time Comp.	Data Comp.
Non-asymptotical	X or Y	$\mathcal{O}(L^2 2^{L \cdot h\left(\frac{\omega}{m}\right)})$	$2^m (3 \sum_{i=0}^{\omega} \binom{m}{i})^{-1}$
Asymptotically optimum	X or Y	$\mathcal{O}(L^2 2^{\frac{2}{3}L})$	$\mathcal{O}(2^{\frac{2}{3}L})$

Asymptomatic behavior is the same as Johansson's but ...

Comparison with Johansson's attack
Non-asymptotic behavior

For $L = 80$:

	C_T	C_D
Johansson's	$2^{69.4}$	$2^{55.2}$
Ours ($\omega = 0.42L$)	$2^{69.4}$	$2^{42.3}$

An improvement of a factor $2^{12.9}$ in data complexity!

Comparison
Short Keystream Attack

	C_T	C_D
Golic and Menicocci's	2^L	$40L$
Ours	$2^{\gamma L}$	$40L$

Again asymptotic behaviors are the same since $\gamma \longrightarrow 1$ when $L \longrightarrow \infty$.

Comparison
Short Keystream Attack

	C_T	C_D
Golic and Menicocci's	2^L	$40L$
Ours	$2^{\gamma L}$	$40L$

Again asymptotic behaviors are the same since $\gamma \longrightarrow 1$ when $L \longrightarrow \infty$.

But for moderate values there is some gain.

$L = 80 \Longrightarrow \gamma = 0.945 \Longrightarrow$ an improvement of a factor $2^{4.4}$.

A Large Scale Example of an Attack

Setting: $L = 42$ and random feedback polynomials

A Large Scale Example of an Attack

Setting: $L = 42$ and random feedback polynomials

$m = 84, \omega = 14 \Longrightarrow H = 24.16 \Longrightarrow T = 2^H = 18\,782\,717$

A Large Scale Example of an Attack

Setting: $L = 42$ and random feedback polynomials

$m = 84, \omega = 14 \Longrightarrow H = 24.16 \Longrightarrow T = 2^H = 18\,782\,717$

The correct initial state of LFSR_X is found in 84 out of 200 tries

$$\Longrightarrow P_{suc} \approx 0.42$$

Summary

Summary

- A quite general attack principle exemplified in the case of ASG
- Closed form for the complexity of the best previous attack on ASG
- New reduced complexity attack on ASG
  - More flexibility in keystream structure
  - Experimental confirmation of the attack