Synchronization fault cryptanalysis of A5/1

M. Gomułkiewicz, M. Kutyłowski, Th. Vierhaus, P. Wlaź

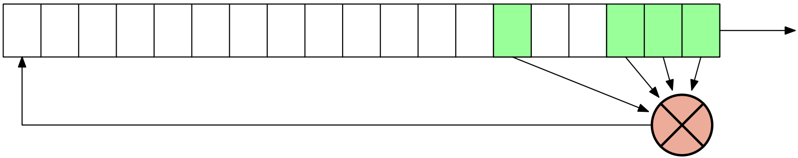
Wrocław University of Technology, Brandenburgische Technische Universität, Lublin University of Technology

4th International
Workshop on Efficient and Experimental Algorithms

A5/1

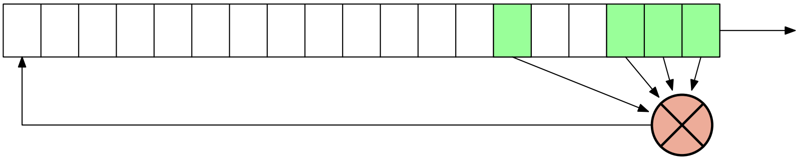
- cheap pseudo-random string generator for encryption in GSM
- possible applications:
  - lightweight cryptography for weak devices
    - sensor networks,
    - Bluetooth like
  - a component for self-testing circuits of crypto hardware

LFSR -linear shift register



- in a step:
  - the rightmost bit = the current output bit,
  - all bits move one position to the right,
  - the leftmost bit obtained as a linear combination of bits from certain positions

LFSR -linear shift register



- in a step:
  - the rightmost bit = the current output bit,
  - all bits move one position to the right,
  - the leftmost bit obtained as a linear combination of bits from certain positions
- despite a long period it is a very weak cryptographically: breaking by building a system of linear equations

How to make LFSR's stronger?

How to make LFSR's stronger?

- combine the output of a few different LFSR's

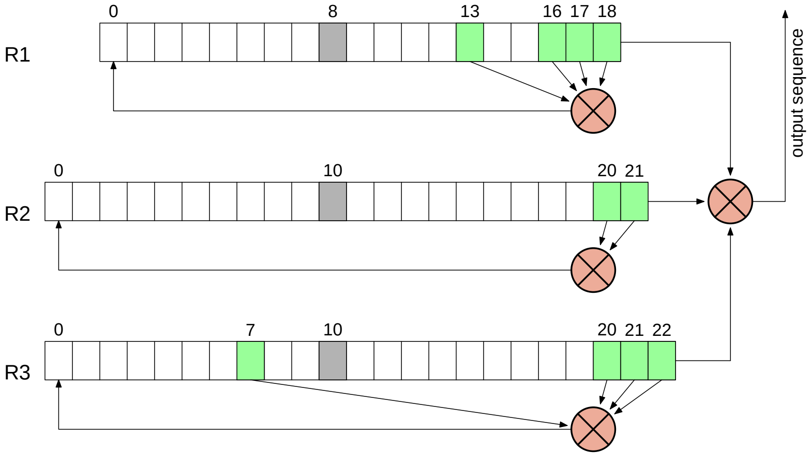
How to make LFSR's stronger?

- combine the output of a few different LFSR's
  - with XOR as a combining function - again easy to break

How to make LFSR's stronger?

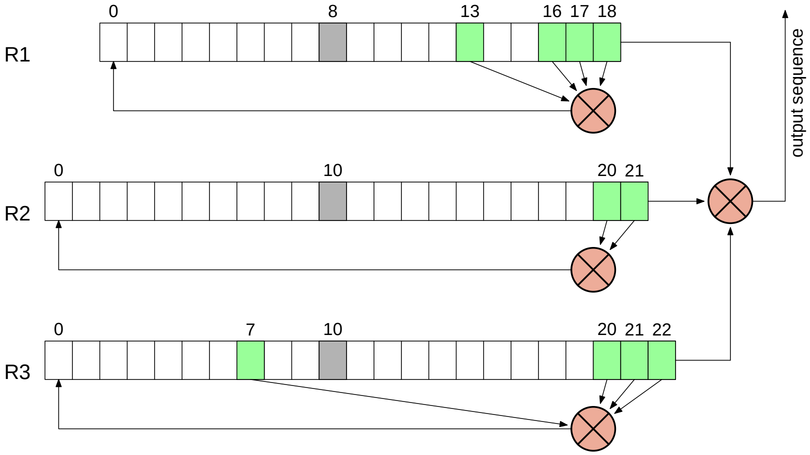
- combine the output of a few different LFSR's
  - with XOR as a combining function - again easy to break
  - inserting some nonlinear operation

A5/1



- 3 LFSR's
- their output XOR-ed

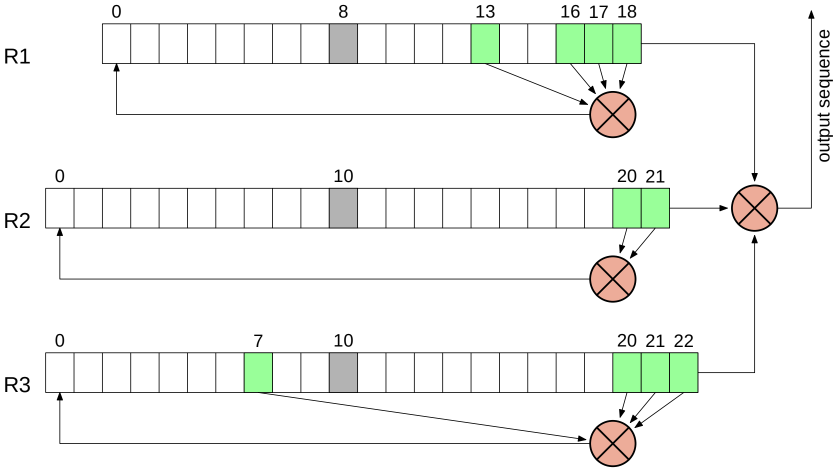
A5/1



- 3 LFSR's
- their output XOR-ed
- but: one out of three LFSR's might be stopped from shifting at each step

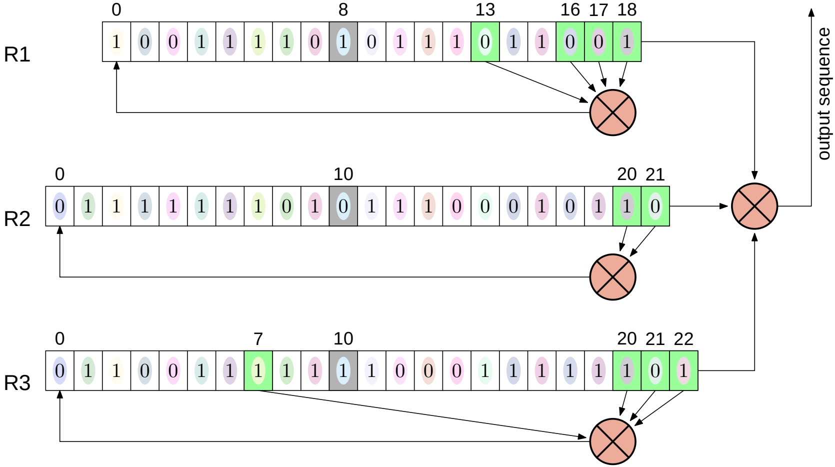
Clocking

- bits at positions 8, 10, 10, respectively, are considered,
- if an LFSR i has a bit b and the remaining 2 LFSR's have bit $1 - b$, then this LFSR is not active at this step



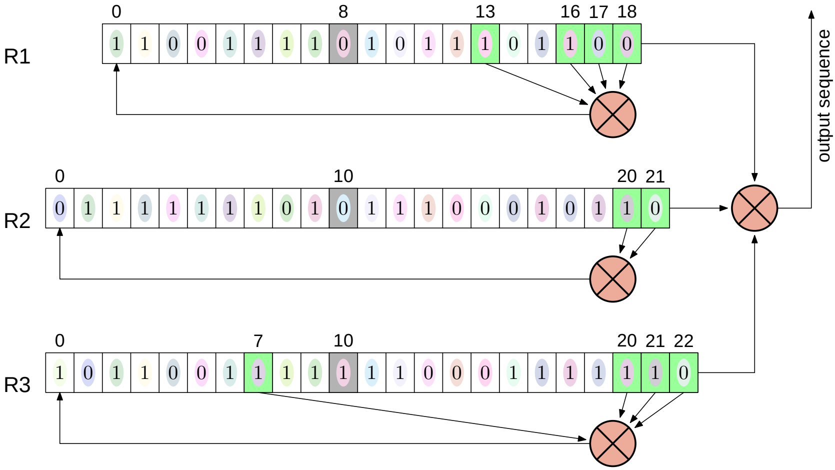
Clocking

- bits at positions 8, 10, 10, respectively, are considered,
- if an LFSR *i* has a bit *b* and the remaining 2 LFSR's have bit $1 - b$, then this LFSR is not active at this step



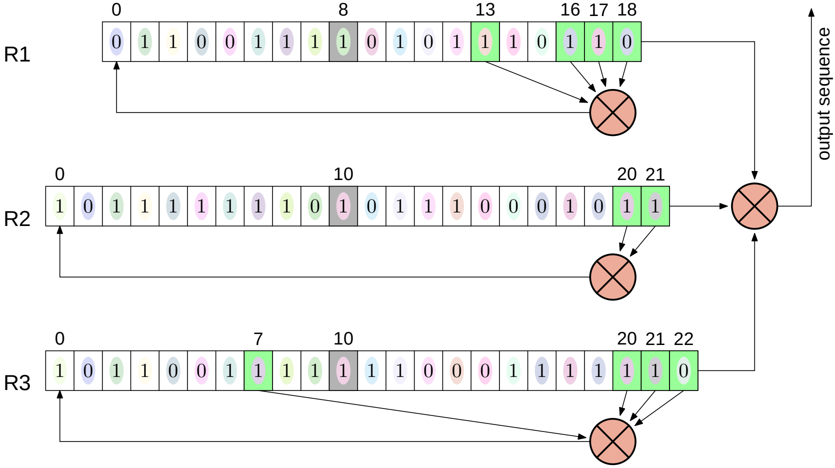
Clocking

- bits at positions 8, 10, 10, respectively, are considered,
- if an LFSR *i* has a bit *b* and the remaining 2 LFSR's have bit $1 - b$, then this LFSR is not active at this step



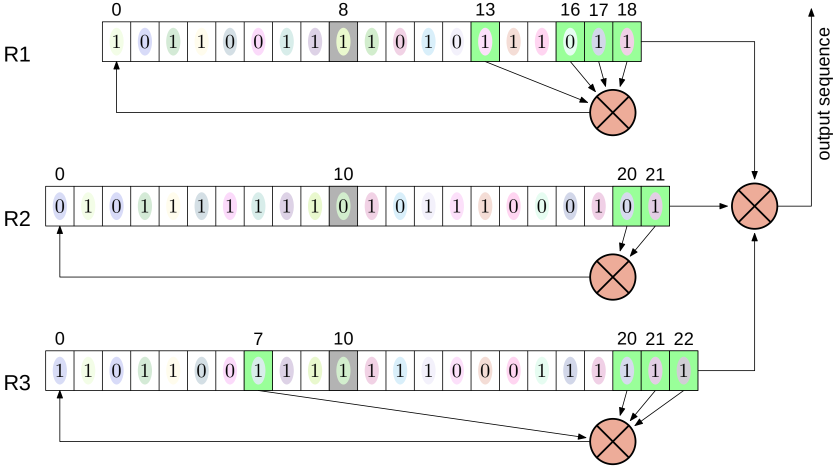
Clocking

- bits at positions 8, 10, 10, respectively, are considered,
- if an LFSR *i* has a bit *b* and the remaining 2 LFSR's have bit $1 - b$, then this LFSR is not active at this step



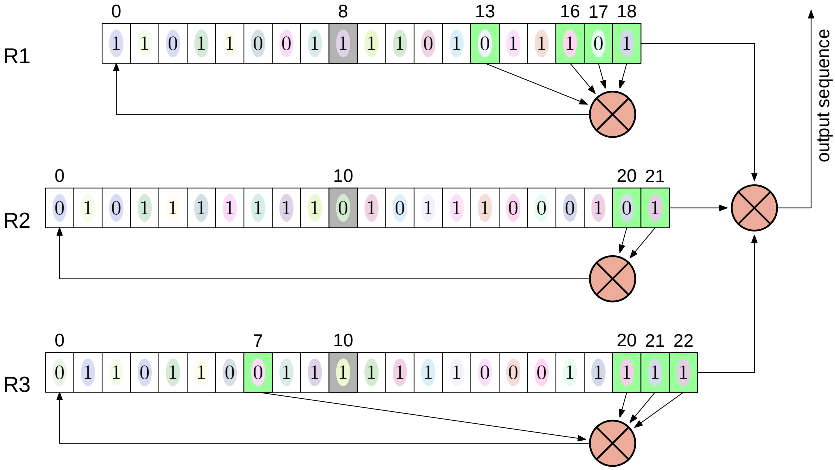
Clocking

- bits at positions 8, 10, 10, respectively, are considered,
- if an LFSR i has a bit b and the remaining 2 LFSR's have bit $1 - b$, then this LFSR is not active at this step



Clocking

- bits at positions 8, 10, 10, respectively, are considered,
- if an LFSR i has a bit b and the remaining 2 LFSR's have bit $1-b$, then this LFSR is not active at this step



Attacks on A5/1

- via switching to a weak A5/2 (GSM specific)
- statistical analysis plus backtracking to the moment when the secret is in the registers

  very much dependent on the length of LFSR's and the feedback function

Fault Cryptanalysis

- classical cryptanalysis: only output (and input) considered

Fault Cryptanalysis

- classical cryptanalysis: only output (and input) considered
- fault cryptanalysis – a tamper proof device holding secret keys inside
  goal – reconstruct the keys
  method – generate faults and analyze the output

Our Attack

we show that the clever choice of shifting rule of A5/1 might be dangerous due to fault attacks

Attack idea

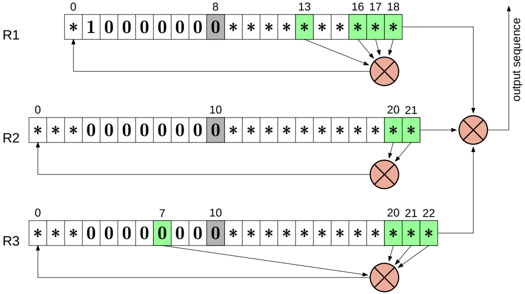
- run a device twice with the same frame number
  - once without fault
  - once with a fault that prevents one of the LFSR's from shifting

Attack idea

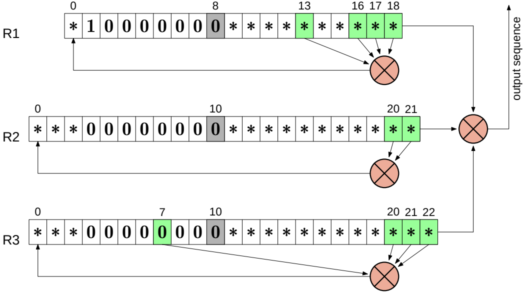
- run a device twice with the same frame number
  - once without fault
  - once with a fault that prevents one of the LFSR's from shifting
- typically the outputs get completely different from the moment of injecting a fault
- **but sometimes it is the same after a certain number of steps**
- **the reason: accidentally the pattern of moves in the faulty case catches up the correct computation**

Re-synchronization -Example

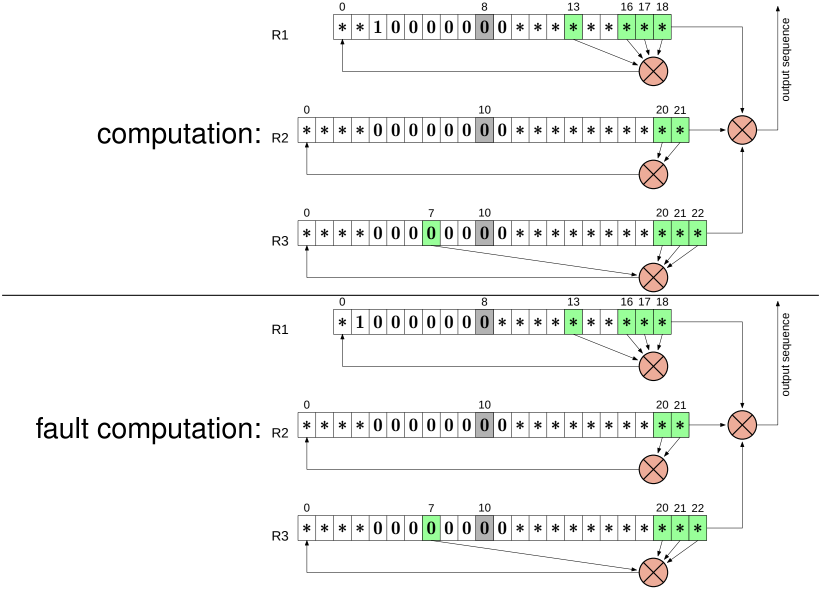
computation:



fault computation:

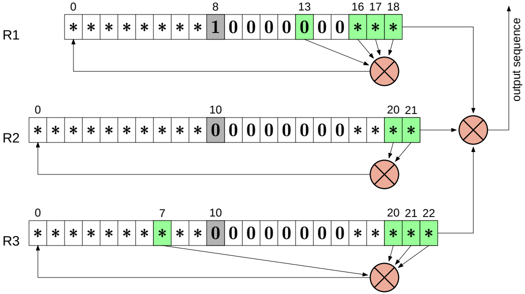


Re-synchronization -Example

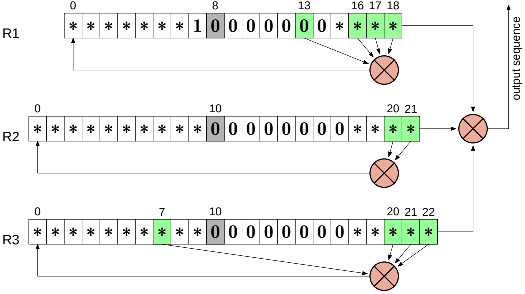


Re-synchronization -Example

computation:

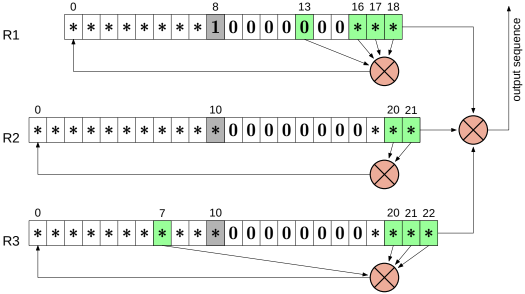


fault computation:

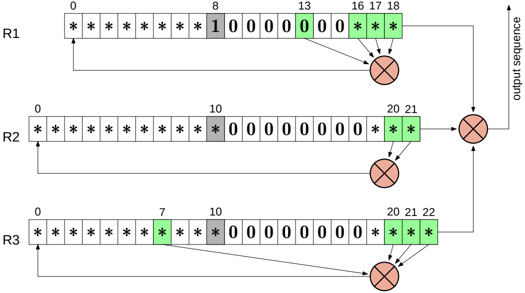


Re-synchronization -Example

computation:



fault computation:



Re-synchronization patterns

- we look for such “catching up” within 5 to 9 steps

Re-synchronization patterns

- we look for such “catching up” within 5 to 9 steps
- it does not occur frequently, but if it occurs we have only a few hundred candidates for blocks consisting of several bits

Re-synchronization patterns

- we look for such “catching up” within 5 to 9 steps
- it does not occur frequently, but if it occurs we have only a few hundred candidates for blocks consisting of several bits
- the patterns of bits causing re-synchronization after k steps are called
  re-synchronization patterns of length k
  or RSPk.

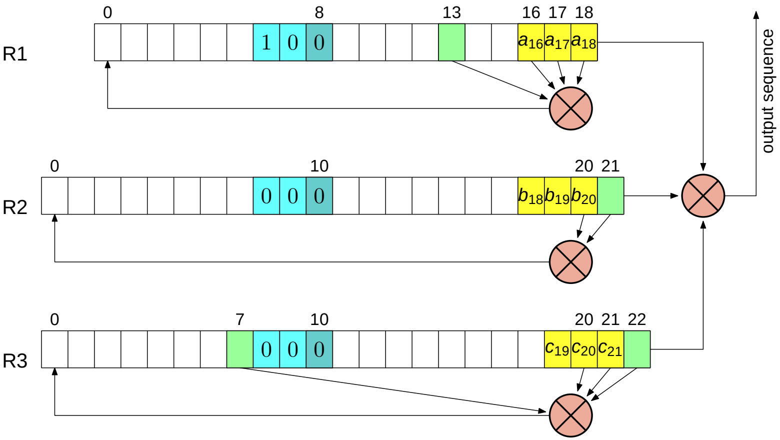
Re-synchronization patterns

- we look for such "catching up" within 5 to 9 steps
- it does not occur frequently, but if it occurs we have only a few hundred candidates for blocks consisting of several bits
- the patterns of bits causing re-synchronization after k steps are called
  re-synchronization patterns of length k
  or RSPk.
- there are 30 for RSP5, 112 for RSP6, 480 for RSP7, 2068 for RSP8, and 8992 for RSP9.

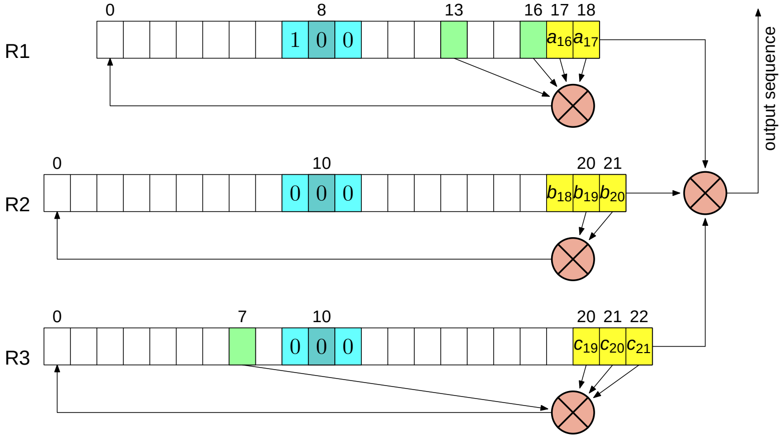
Re-synchronization patterns

- we look for such “catching up” within 5 to 9 steps
- it does not occur frequently, but if it occurs we have only a few hundred candidates for blocks consisting of several bits
- the patterns of bits causing re-synchronization after k steps are called
  re-synchronization patterns of length k
  or RSPk.
- there are 30 for RSP5, 112 for RSP6, 480 for RSP7, 2068 for RSP8, and 8992 for RSP9.
- chances for re-synchronization of length 5–9 are about 1.5$\%$ (assuming independency of bits – and experiments confirm the figure)
- “output re-synchronization” after 5–8 steps gives 90% chances for re-synchronization after 5–9 steps

Linear Equations for Patterns – an example

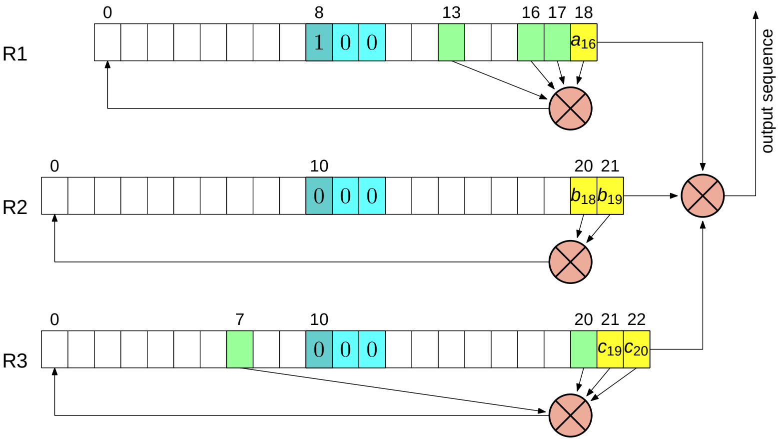


Linear Equations for Patterns – an example



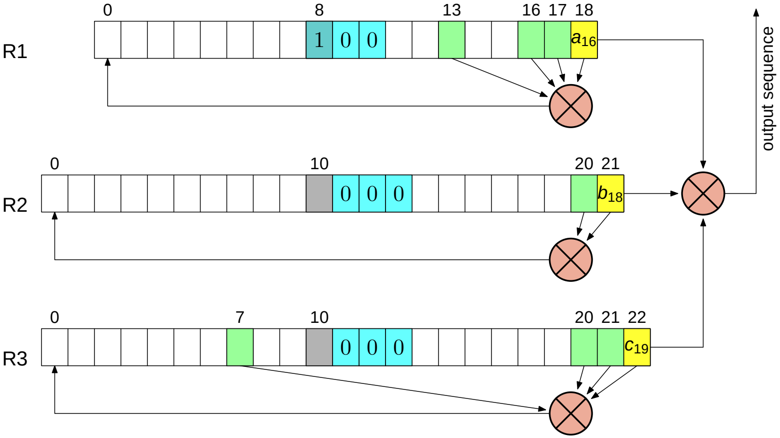
proper computation: $a_{17} + b_{20} + c_{21} = x_1$

Linear Equations for Patterns – an example



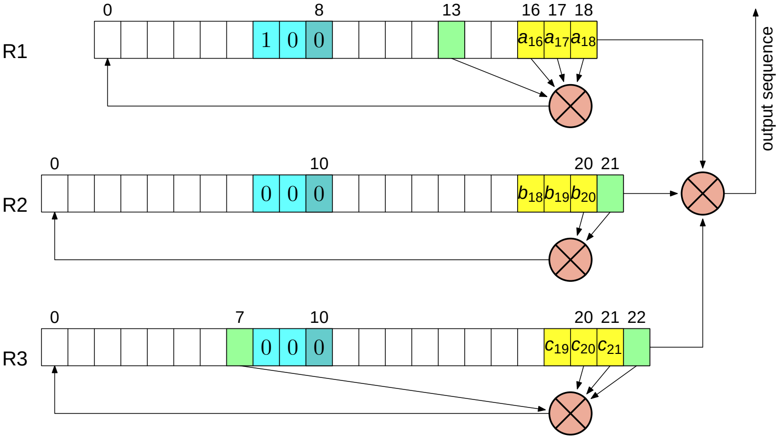
proper computation: $a_{17} + b_{20} + c_{21} = x_1$, $a_{16} + b_{19} + c_{20} = x_2$

Linear Equations for Patterns – an example



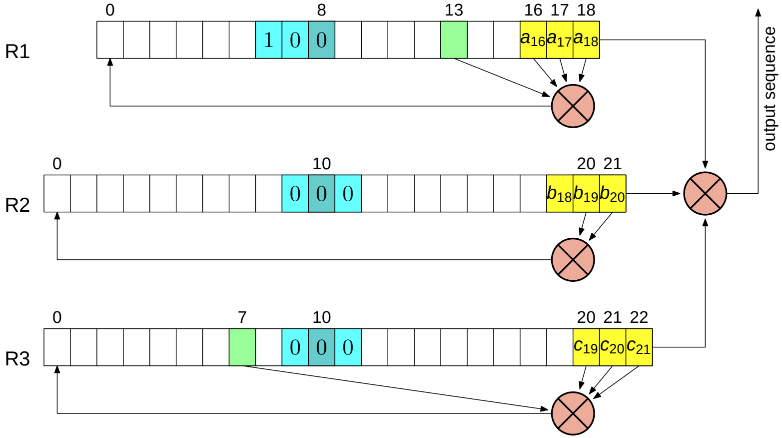
proper computation: $a_{17} + b_{20} + c_{21} = x_1$, $a_{16} + b_{19} + c_{20} = x_2$, $a_{16} + b_{18} + c_{19} = x_3$

Linear Equations for Patterns – an example



proper computation: $a_{17} + b_{20} + c_{21} = x_1$, $a_{16} + b_{19} + c_{20} = x_2$, $a_{16} + b_{18} + c_{19} = x_3$

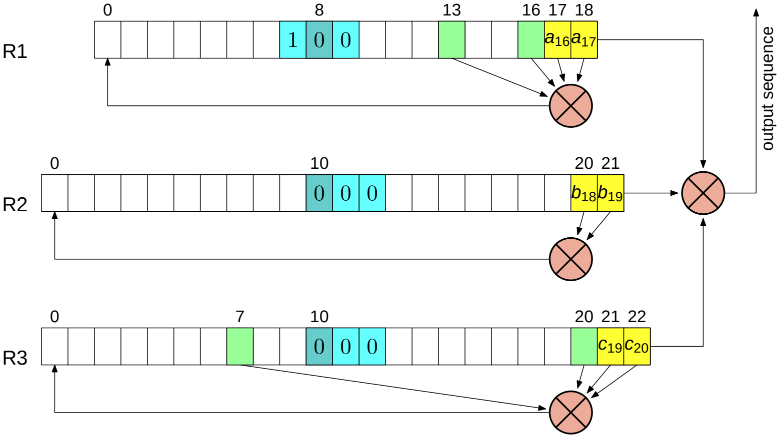
Linear Equations for Patterns – an example



proper computation: $a_{17} + b_{20} + c_{21} = x_1$, $a_{16} + b_{19} + c_{20} = x_2$, $a_{16} + b_{18} + c_{19} = x_3$

fault computation: $a_{18} + b_{20} + c_{21} = y_1$

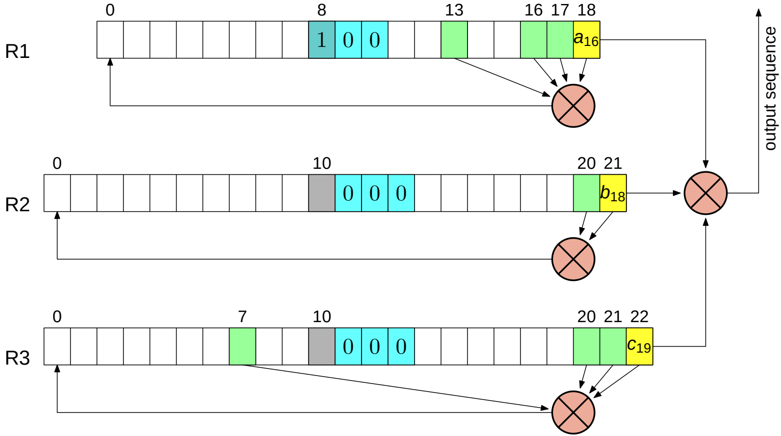
Linear Equations for Patterns – an example



proper computation: $a_{17} + b_{20} + c_{21} = x_1$, $a_{16} + b_{19} + c_{20} = x_2$, $a_{16} + b_{18} + c_{19} = x_3$

fault computation: $a_{18} + b_{20} + c_{21} = y_1$, $a_{17} + b_{19} + c_{20} = y_2$

Linear Equations for Patterns – an example



proper computation: $a_{17} + b_{20} + c_{21} = x_1$, $a_{16} + b_{19} + c_{20} = x_2$, $a_{16} + b_{18} + c_{19} = x_3$

fault computation: $a_{18} + b_{20} + c_{21} = y_1$, $a_{17} + b_{19} + c_{20} = y_2$, $a_{16} + b_{18} + c_{19} = y_3$

Linear Equations for Patterns – an example

We solve the system (all 5 equations are independent in this case)

$$\begin{cases} a_{16} = b_{20} + c_{20} + x_1 + x_2 + y_2 \\ a_{17} = b_{20} + c_{21} + x_1 \\ a_{18} = b_{20} + c_{21} + y_1 \\ b_{18} = b_{20} + c_{19} + c_{20} + x_1 + x_2 + x_3 + y_2 \\ b_{19} = b_{20} + x_1 + y_2 \end{cases}$$

store the solution;

Linear Equations for Patterns – an example

We solve the system (all 5 equations are independent in this case)

$$\begin{cases} a_{16} = b_{20} + c_{20} + x_1 + x_2 + y_2 \\ a_{17} = b_{20} + c_{21} + x_1 \\ a_{18} = b_{20} + c_{21} + y_1 \\ b_{18} = b_{20} + c_{19} + c_{20} + x_1 + x_2 + x_3 + y_2 \\ b_{19} = b_{20} + x_1 + y_2 \end{cases}$$

store the solution;
after the output is known, we have to guess 4 unknowns, and easily calculate the other 5.

Linear Equations for Patterns

- On average, more than 70% of patterns are excluded

Linear Equations for Patterns

- On average, more than 70% of patterns are excluded
- Gains in the number of bits from considering RSP are

RSP#	5	6	7	8	9
gain	16.93	19.31	21.45	23.63	25.80

Next Steps

- gradually guess the values of unknown bits needed for the clocking mechanism,

Next Steps

- gradually guess the values of unknown bits needed for the clocking mechanism,
- emulate a move of the system,

Next Steps

- gradually guess the values of unknown bits needed for the clocking mechanism,
- emulate a move of the system,
- construct a linear equation with current rightmost bits of the registers and the output bit.

Next Steps

- gradually guess the values of unknown bits needed for the clocking mechanism,
- emulate a move of the system,
- construct a linear equation with current rightmost bits of the registers and the output bit.

- about 2^{34} systems of linear equations to be considered

Remarks and Conclusions

- No matter what is the length of LFSR's we **always** get some gain - we reduce the number of unknown bits in the LFSRs.

Remarks and Conclusions

- No matter what is the length of LFSR's we **always** get some gain - we reduce the number of unknown bits in the LFSRs.
- feedback not confined to the values in the same LFSR would make this attack infeasible.

Remarks and Conclusions

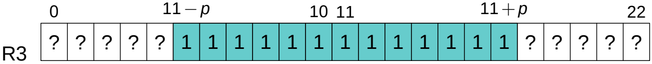
- No matter what is the length of LFSR's we **always** get some gain - we reduce the number of unknown bits in the LFSRs.

- feedback not confined to the values in the same LFSR would make this attack infeasible.
- similar re-synchronization attack when injecting single random faults

Other Models

Marcin Gomułkiewicz, Mirosław Kutyłowski, Paweł Wlaź, *Fault Cryptanalysis for Breaking A5/1*, to appear in Tatra Mountains Mathematical Publications, 2005

- The attacker can set "continuous" area in the center of one of register to ones in given moment



- only one (fault) output needed
- about $2^{40-1.6p}$ systems + $400 \cdot 2^{23}$ frame runs on a simulator

Thanks for your attention!