RUHR-UNIVERSITÄT BOCHUM

RUB

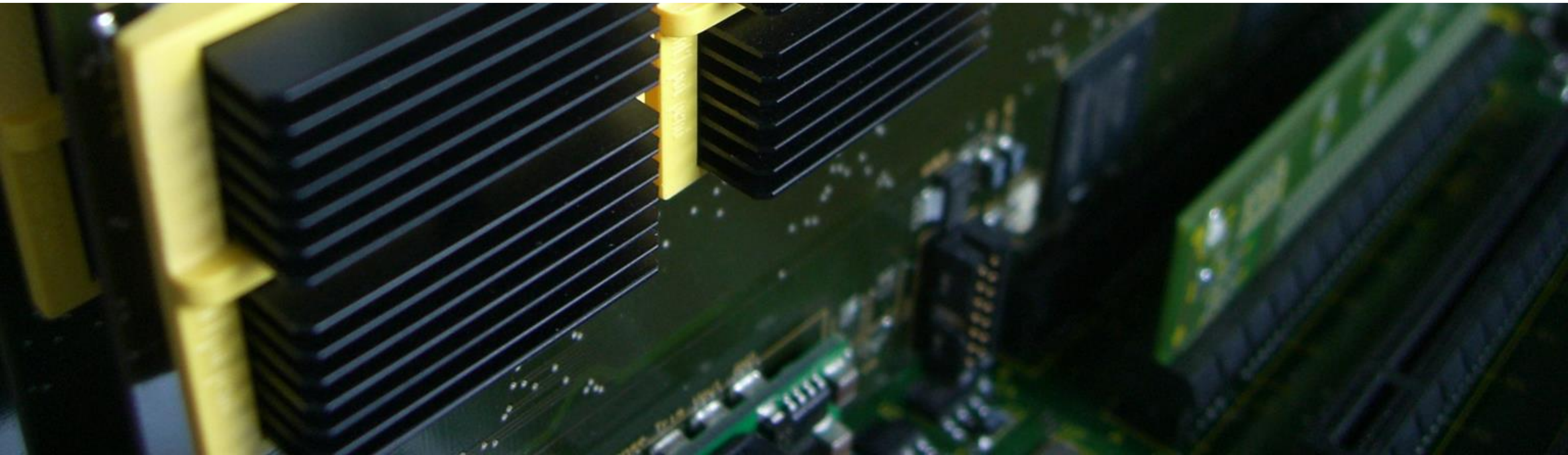
Fault Sensitivity Analysis Meets Zero-Value Attack

Oliver Mischke, Amir Moradi, Tim Güneysu
Horst Görtz Institute for IT-Security

secmobil

EUROPÄISCHE UNION
Investition in unsere Zukunft
Europäischer Fonds
für regionale Entwicklung

FDTC 2014, Busan, 2014-09-23



Motivation

- Zero-value vulnerability is a known issue (AES S-box)
  - Major weakness in multiplicative masking schemes
  - Also applicable to unmasked implementations
- Not only relevant to Side-Channel Analysis (SCA) but also to Fault Sensitivity Analysis (FSA)
- In fact: Weakness is so severe it can even be used to break several Concurrent Error Detection (CED) schemes

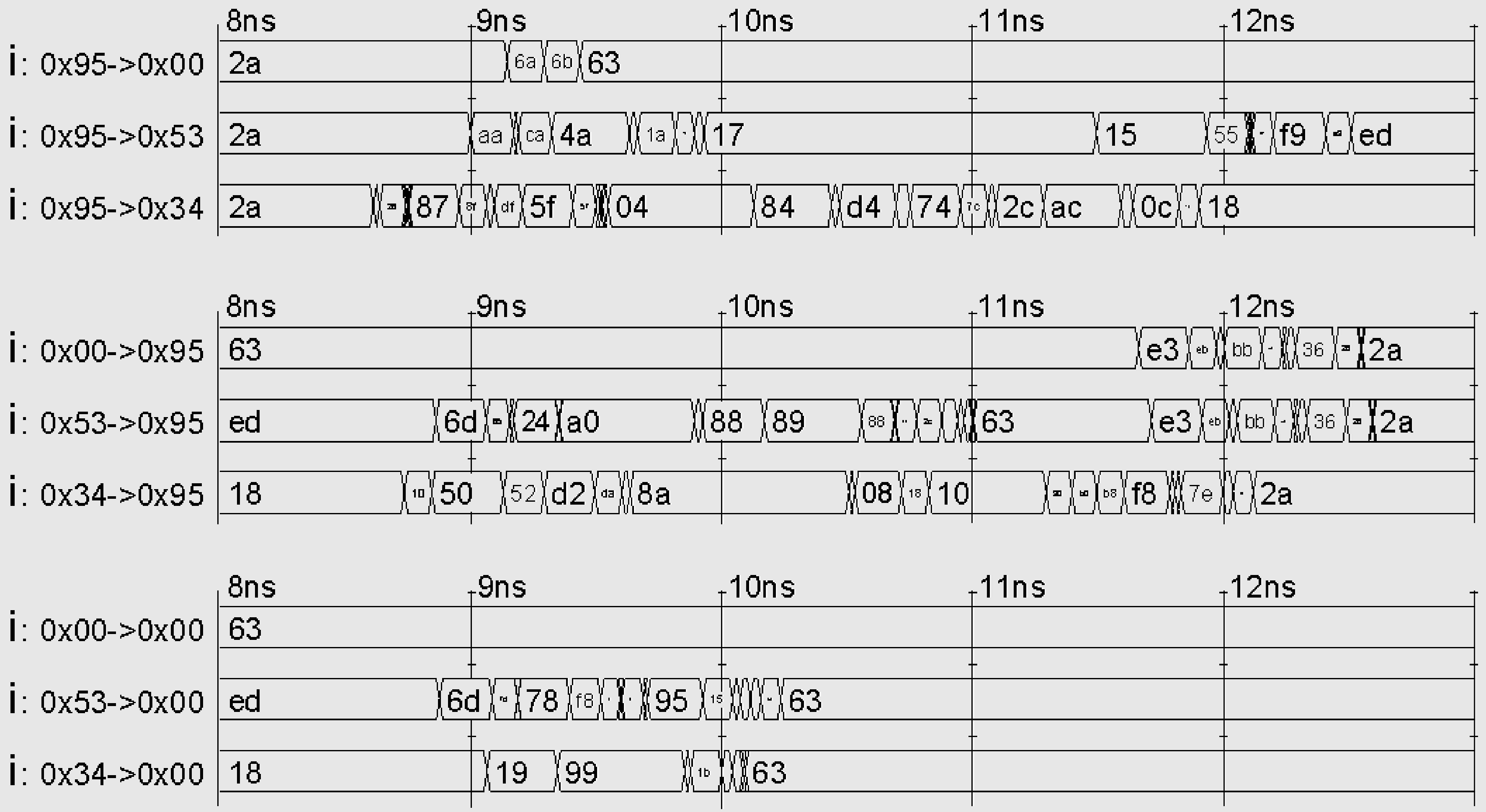
Outline

- Fault Sensitivity Analysis
- Zero-value vulnerability of composite field S-boxes
- Evaluation architecture
- Zero-value attack & results
- Conclusion

Fault Sensitivity Analysis

- Presented by Yang Li *et al.* CHES 2010
- Critical path delay of an AES S-box is input dependent
- Insert faults by clock glitches
- Showed that the critical path of a PPRM S-box correlates to the Hamming weight
- No use of faulty output but of byte-wise fault information/rate
- Extended in CHES 2011 by Correlation Collision Attack
- Model for other S-boxes?

Simulation Results Critical Path Delays of the Used S-box on SASEBO-G2 (Virtex 5)

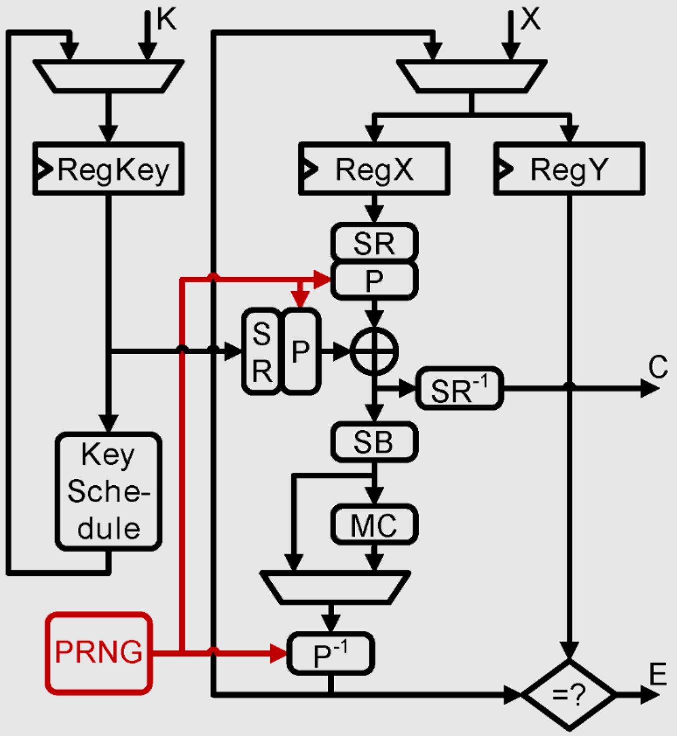


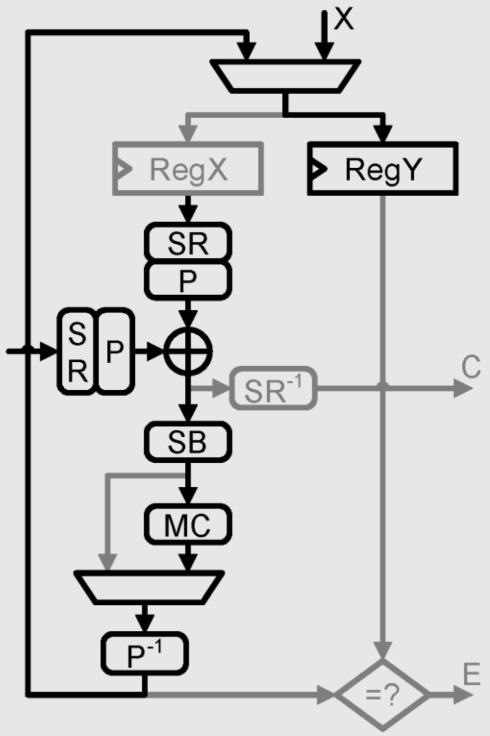
Zero-Value Vulnerability

- Very distinct weakness, clearly exploitable by *standard* FSA
- What if we use a CED scheme?
  - No byte or bit-wise fault information available
  - Key cannot be found directly!
- Indirect approach:
  - Instead of finding the correct key bytes exclude wrong candidates!

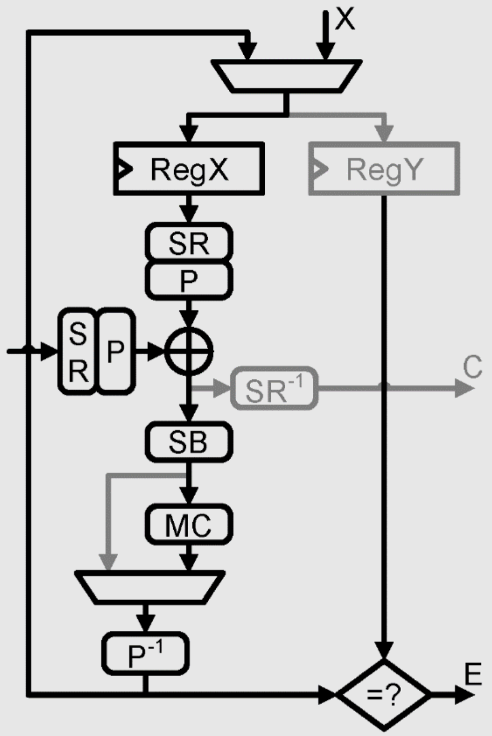
Architecture

- Round-based AES-128, two cycles per round



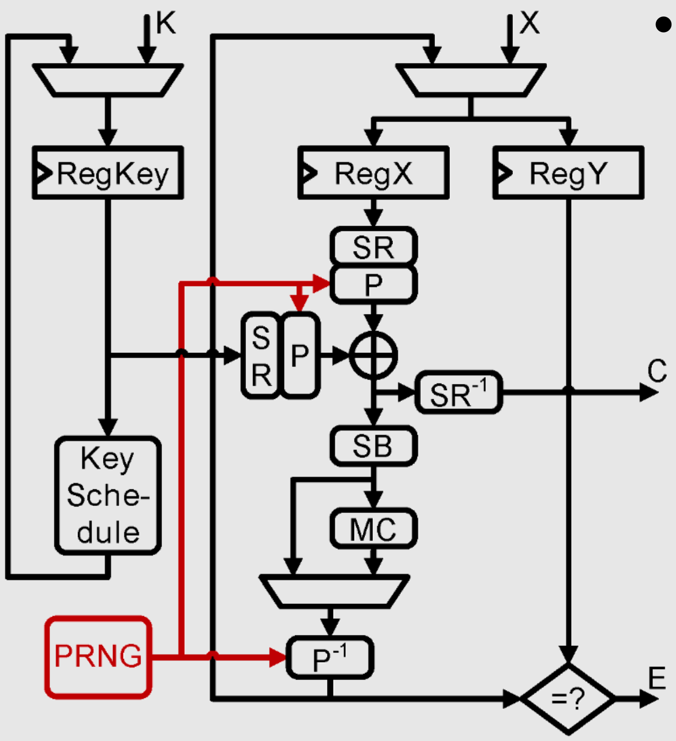


computation step



checking step

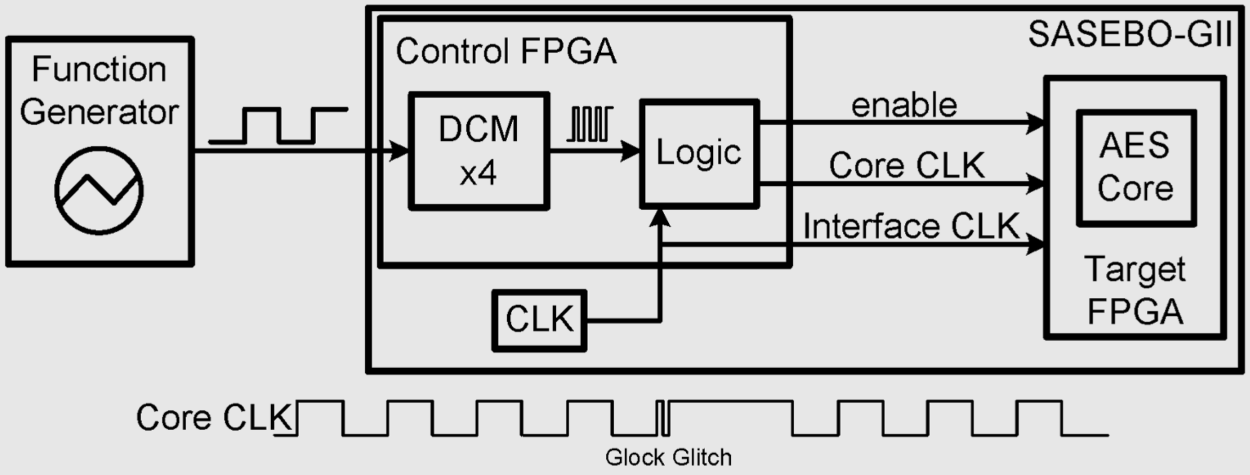
Architecture cont‘d

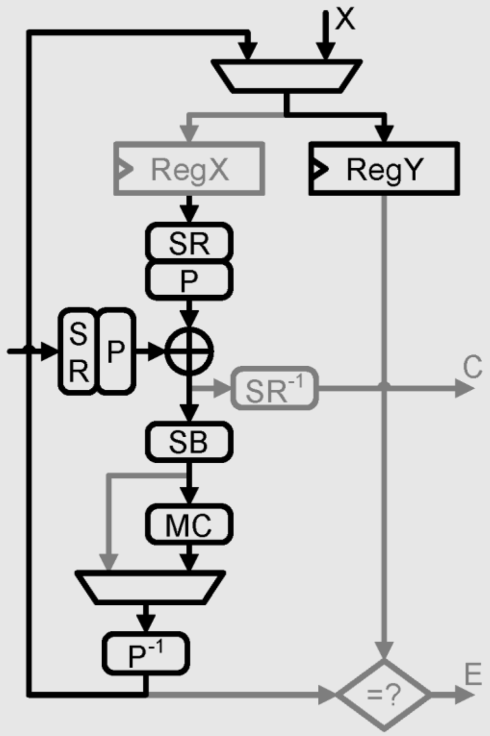


- Circuit can mimic different CED schemes
- Scheme dependent on configuration of **P**:
  - Profile A:
    - 1st and 2nd cycle: **P** = pass through
      - → Time redundancy CED
  - Profile B:
    - 1st cycle pass through, 2nd cycle fixed permutation
      - → Invariance-based CED (from DAC 2012)
  - Profile C:
    - Both cycles **P** as random column permutation (*shuffling)*

Setup

- SASEBO-G2 (Target: XC5VLX50)
- Agilent 33521A Function Generator



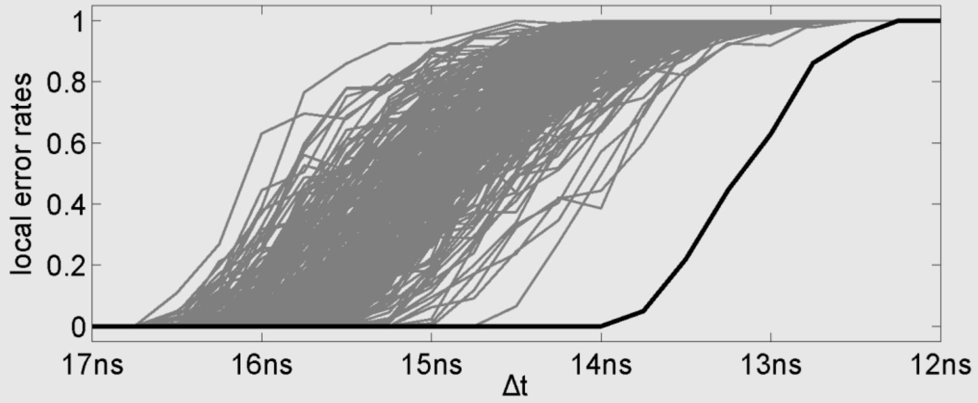


computation step

RUB

Profile A: Evaluation of a Single S-box

- 1st round zero input for S-box$_j$ → $p_j = k_j$
- Send *random* plaintext bytes (for target S-box)



→ vulnerability exists in full implementation

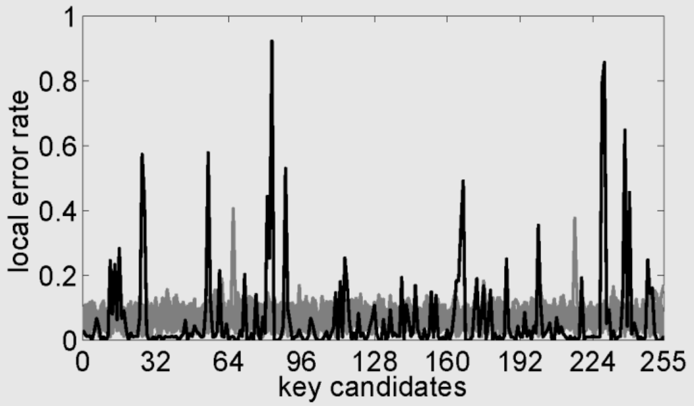
- *Similar* picture for all S-boxes
- But: No usable model besides zero-value

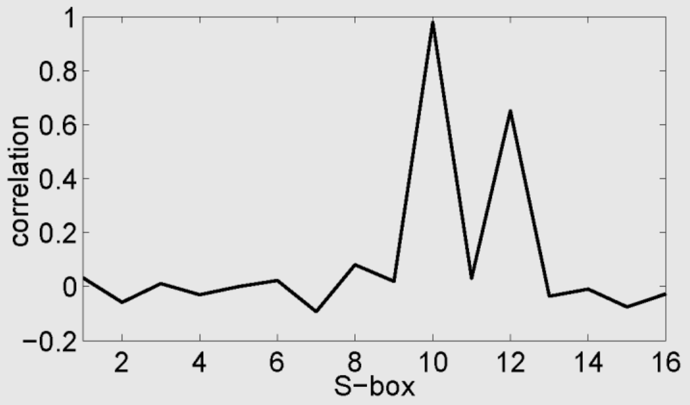
Profile B: Full Key Extraction from CED Protected Circuit

- Long term goal: find full plaintext X which has the shortest critical path

  → all plaintext bytes x_j are equal to their corresponding key bytes
- S-boxes have different critical paths because of placement/routing
- Clock glitch affects some S-boxes more than others
- Try to affect as few S-boxes/input values as possible!

Profile B: Full Key Extraction from CED protected Circuit (First Iteration)

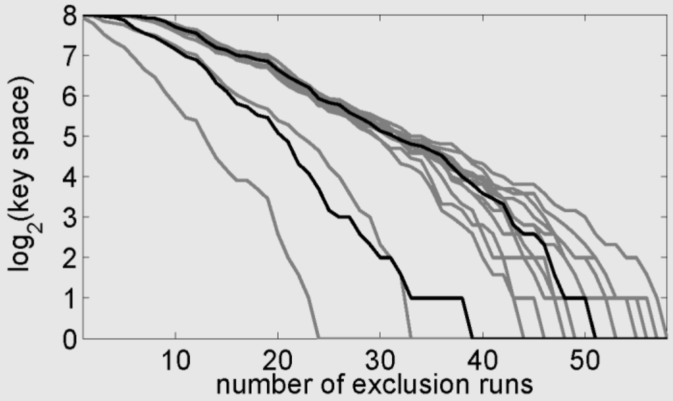


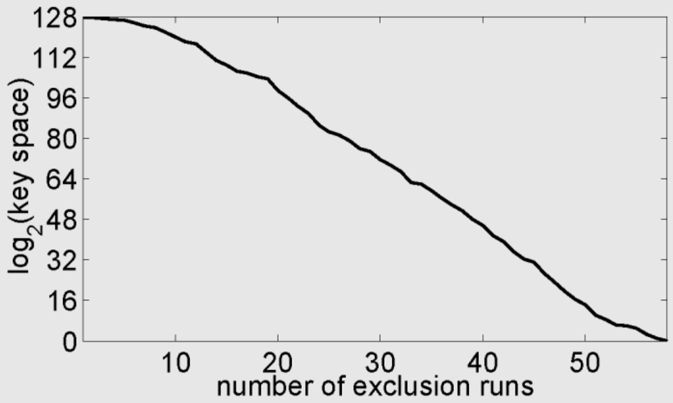


- Start with a clock glitch length which yields a low error rate
- Construct n plaintexts $X^{i\,\in\,\{1,\,\ldots\,,\,n\}} = (x_1{}^i, \ldots, x_{16}{}^i)$ from *remaining key sets* and send to the device
- Note *total error rate* and *local error rates*
  - Total error rate: faulty outputs/number of sent plaintexts
  - Local error rate: fault rate for each value of a certain plaintext byte x_j
- High local error rate for a value x_j of S-box$_j$ means that it is unlikely to be the correct key byte

  → discard value from key set k_j
- Repeat!
  - Construct new plaintexts X^i from remaining values of key sets k_j
  - Decrease glitch duration when total error rate gets too low

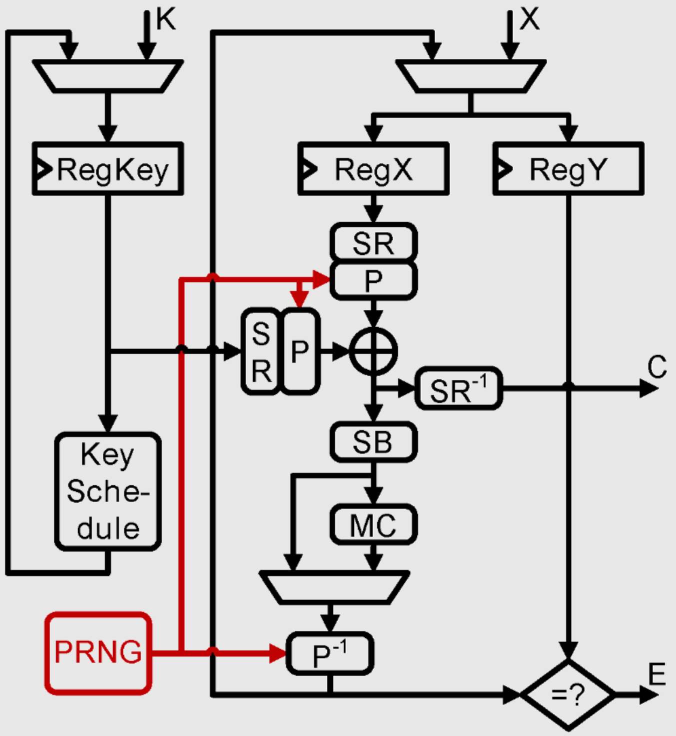
Profile B: Full Key Extraction from CED Protected Circuit (Results)





- Key sets can be systematically restricted
- Here: complete key recovery after < 60 runs (8 hours)
- Number of runs depends on *aggressiveness* of key exclusion
- Recovery possible if correct key byte got falsely excluded (see paper)

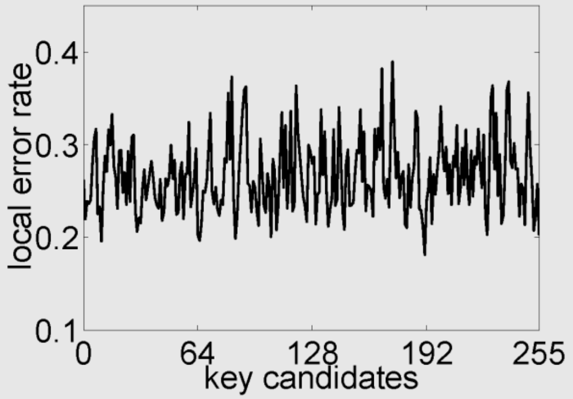
Profile C: Column Shuffling + CED (Idea)



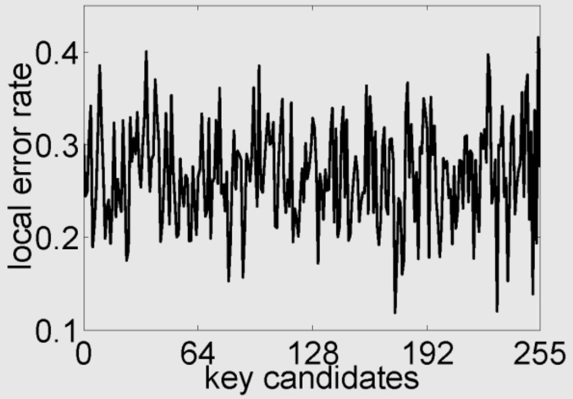
- Use different random permutations **P** in both the computation & checking step
- Original idea: increase attack difficulty
- Local error rates of a state row get mixed (different S-boxes are used)

Profile C: Column Shuffling + CED (Results)

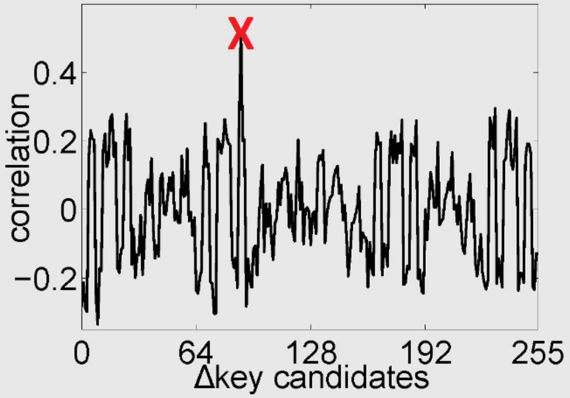
- Attack now easier…
- Since inputs to one row now behave the same collision attacks become possible
- Perform exclusion runs as before (ca. 20-30)
- Retrieve linear key differences for each row
- Brute force remaining key space (32bit)



byte 6



byte 10



result collision attack

Conclusion

- Practical proof that composite field S-boxes have a zero-value vulnerability exploitable by FSA
  - vulnerability is problematic for CED schemes if not mitigated
  - combination of CED with other (SCA) counter-measures can either be a mitigation (masking) or make the attack easier (shuffling)
- Attack also applicable to infection fault countermeasures
- Failproof implementation of CED is **tricky**
  - e.g., ensure comparison is not the critical path

RUHR-UNIVERSITÄT BOCHUM

Thanks!
Any questions?

Oliver Mischke, Amir Moradi, Tim Güneysu
Horst Görtz Institute for IT-Security

FDTC 2014, Busan, 2014-09-23