# XOR Arbiter PUFs: an Empirical Approach to Input Transformations

Nils Wisiol · 30th Crypto Day · {Freie, Technische} Universität Berlin · nils.wisiol@fu-berlin.de

# Outline

PUF 101

Arbiter PUFs

Attacking Arbiter PUFs

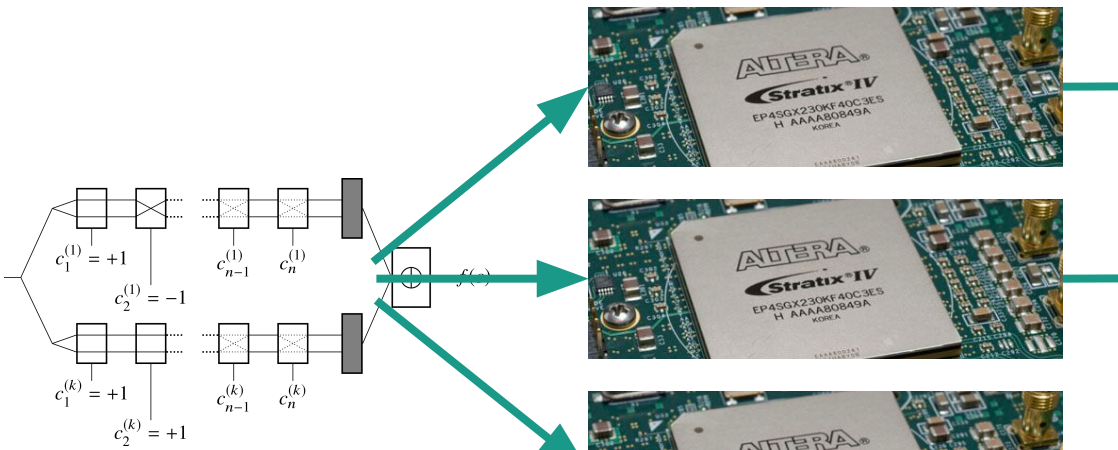
XOR Arbiter PUFs

Input Transformations

Attack Success Rates

Research Questions

# Physically Unclonable Functions







| Input | Output |
|---|---|
| 0101 0101 | 1100 1001 |
| 0101 0101 | 0111 1111 |
| 0101 0101 | 1001 0110 |

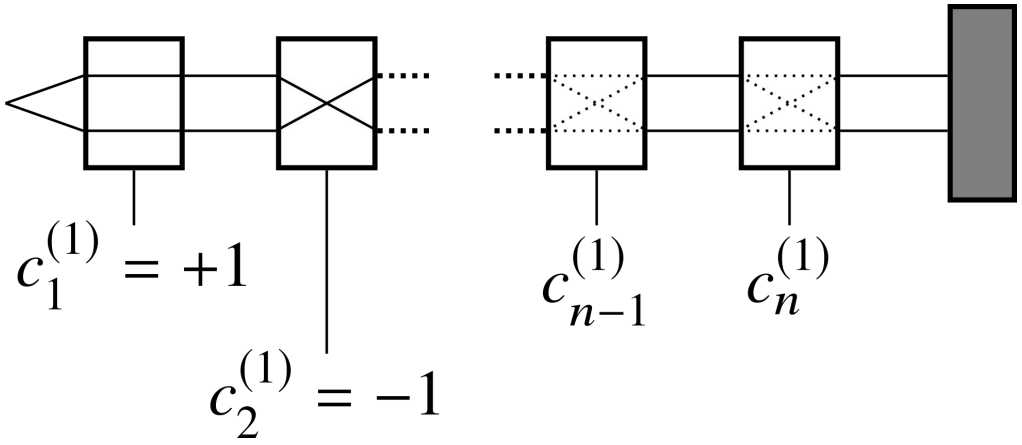
FPGA image credit: Olmaltr · CC BY 3.0 · Cropped
https://commons.wikimedia.org/wiki/File:Altera_StratixIVGX_FPGA.jpg

# Physically Unclonable Functions (PUFs)

- Uniform circuit design
- Chip-individual behavior
  - Formalized by input-output (challenge-response) behavior
  - Caused by manufacturing imperfections
- Use for identification, even authentication?
  - By comparing recorded behavior
  - Depends on unclonability

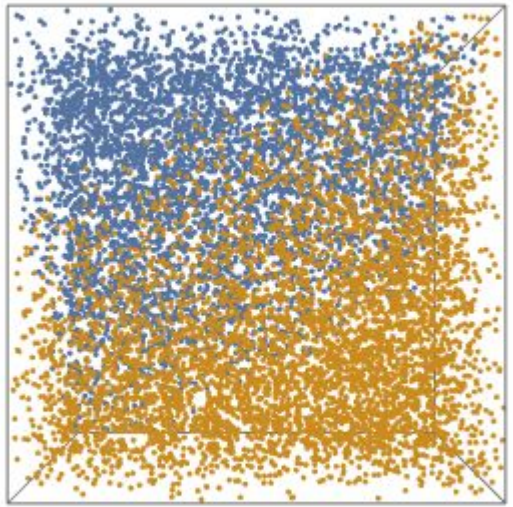
# Arbiter PUF

- A signal race
- Delay depends on manufacturing imperfections
- Challenge sets path
- Output depends on signal delays



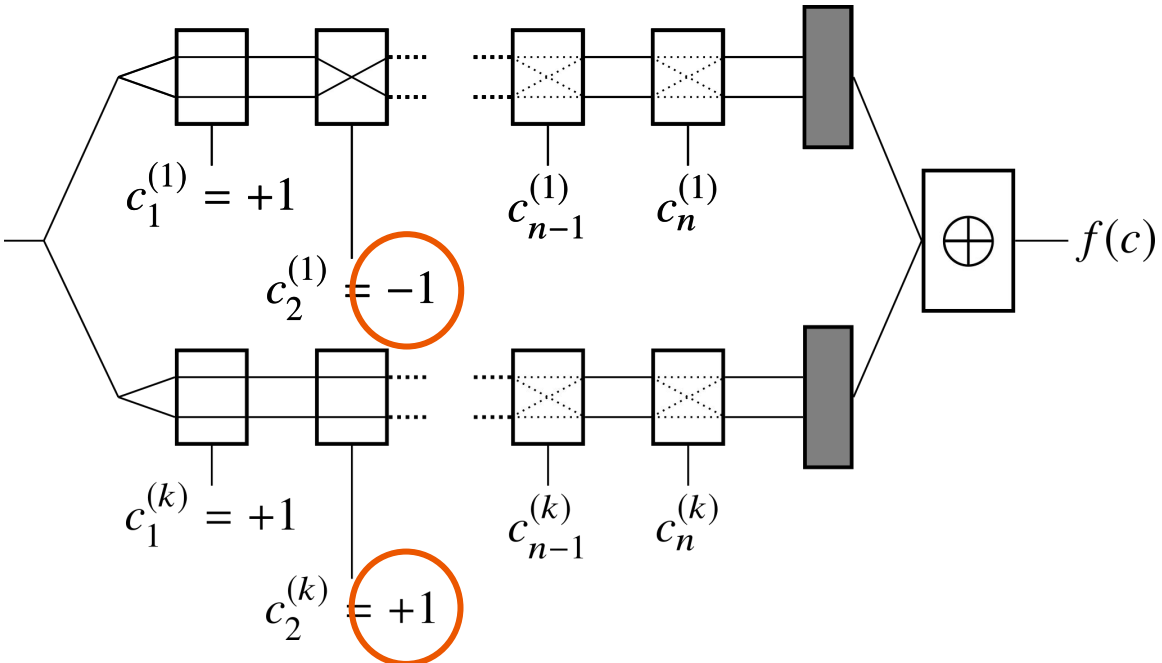
# Modelling Arbiter PUF Behavior

1. Record challenge-response behavior of target
2. **Compute feature vector**
3. Define loss function
4. Gradient descent on loss function



# 64-bit Arbiter PUFs are predictable after seeing just 1000 examples!

# XOR Arbiter PUF and Input Transformations



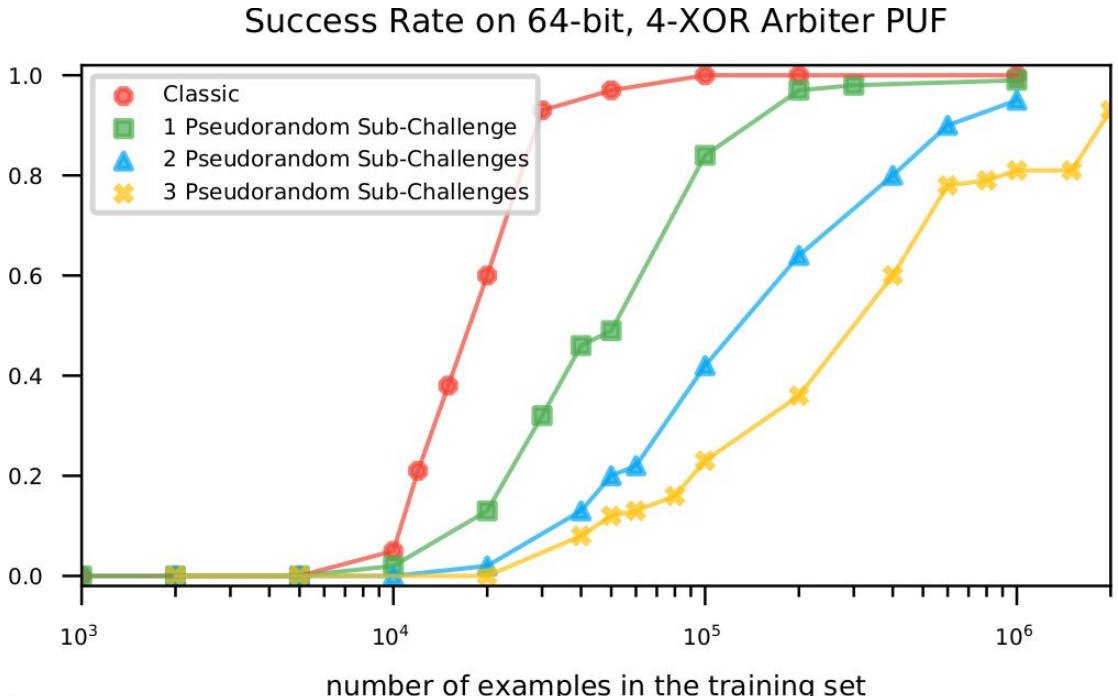

- Modelling more involved
- More examples required
- Easy to implement

# XOR Arbiter PUF Input Transformations

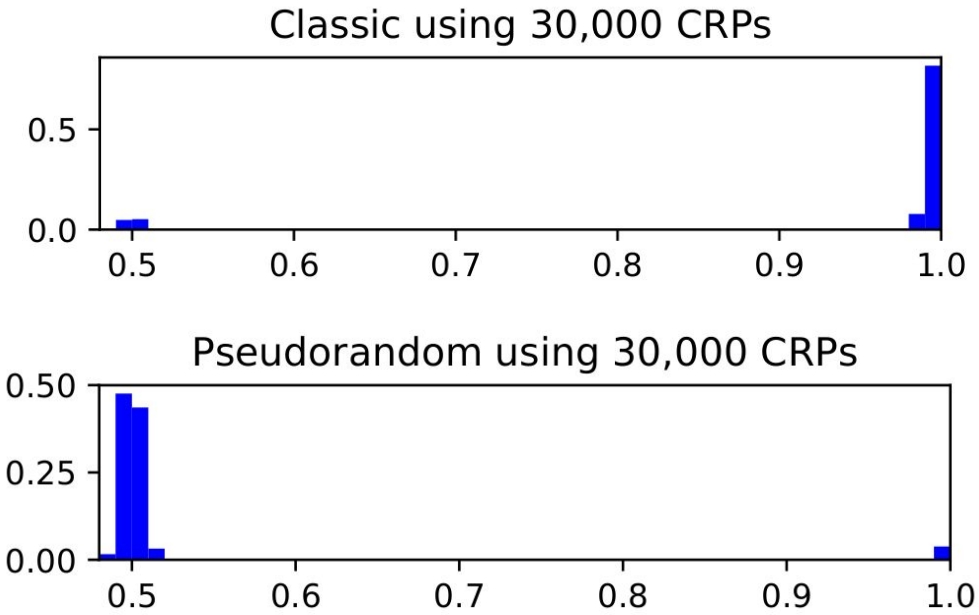
- None (“Classic”): same challenge for every arbiter chain
- Lightweight Secure (Majzoobi et al., 2008): designed for avalanche criterion
- New: Pseudorandom Generator
- New: Permutation of master challenge

# Do Input Transformations Influence the Success of Attacks on XOR Arbiter PUFs?

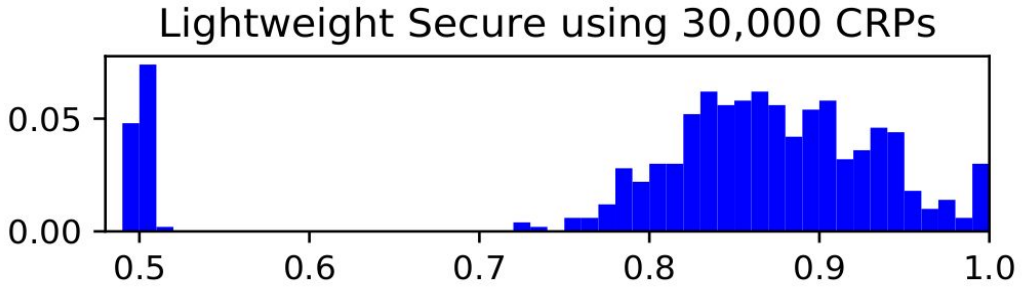
# Classic And Pseudorandom Input Transformations

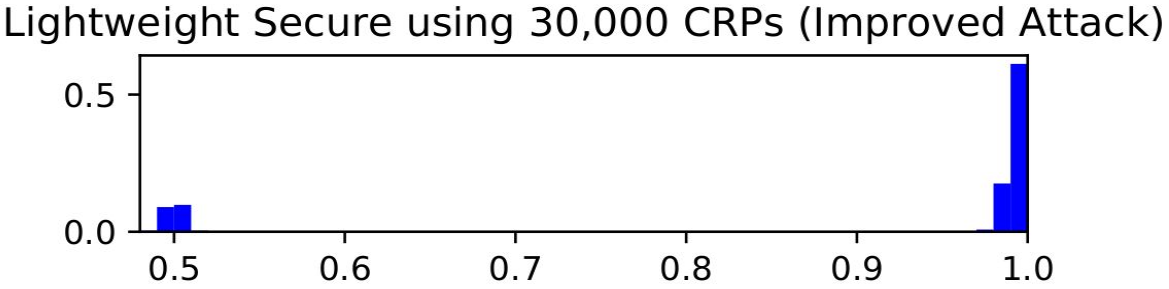


# Prediction Accuracy Distribution

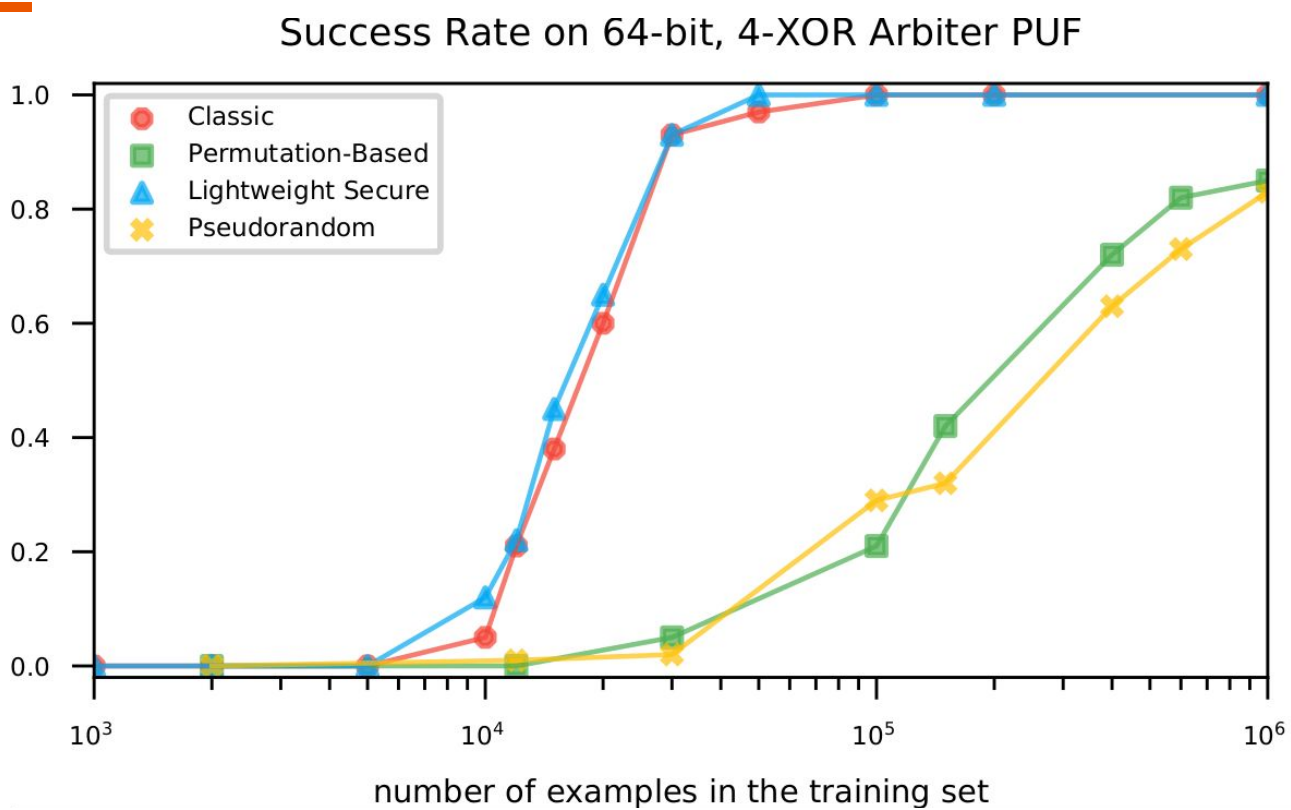


# Lightweight Secure Input Transformation



# Which Input Transformations are Hard To Learn?


Success Rate on 64-bit, 4-XOR Arbiter PUF
Classic
Permutation-Based
Lightweight Secure
Pseudorandom
1.0
0.8
0.6
0.4
0.2
0.0
$10^3$
$10^4$
$10^5$
$10^6$
number of examples in the training set

# Open Research Questions

- Which input transformations can be learned using known techniques?
  - Some positive results as shown in this presentation
- Are certain input transformations PAC Learning learnable?
  - “Classic” XOR Arbiter PUFs are PAC learnable for constant k.

nils.wisiol@fu-berlin.de · github.com/nils-wisiol/pypuf