Freie Universität Berlin

Attacking RO-PUFs with Enhanced Challenge-Response Pairs

Nils Wisiol and Marian Margraf
{firstname.lastname}@fu-berlin.de

Outline

1. Physically Unclonable Functions
2. Ring Oscillator PUF with Enhanced Challenge-Response Pairs
3. Attack
4. Discussion
5. Future Work
6. Q/A

I. Physically Unclonable Functions

Physically Unclonable Functions

- Identical circuit design
- Behavior different on each chip
  - Formalized by a challenge-response schema
- Hard to clone, physically or otherwise
- How many challenges does it have?
  - “Weak” PUF
  - “Strong” PUF

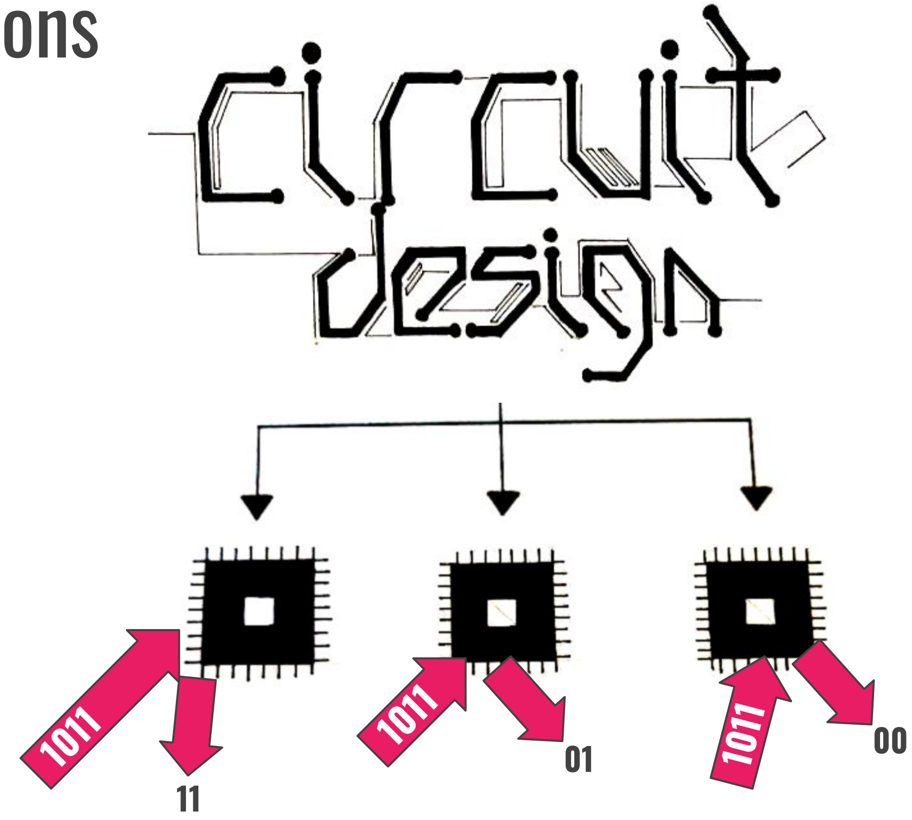

Image credit: Zhenni Li

Ring Oscillator Physically Unclonable Functions

- Cheap and effective method for implementation of PUFs on FPGAs
- Ring of inverters
- Oscillates with hardware-intrinsic frequency
- One PUF has an array of *n* oscillators
- Challenge selects two, response tells us which one has higher frequency
- “Weak”, i.e. small number of challenge-response pairs

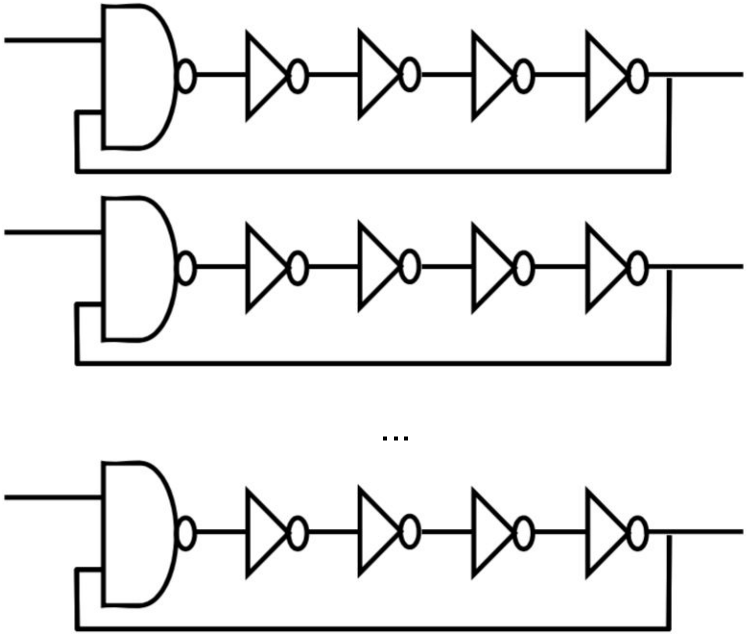

Image credit: Maiti, Abhranil, and Patrick Schaumont. "Improved ring oscillator PUF: an FPGA-friendly secure primitive." Journal of cryptology 24.2 (2011): 375-397.

II. RO-PUF with Enhanced Challenge-Response Pairs

Delavar, Mahshid, Sattar Mirzakuchaki, and Javad Mohajeri. "A Ring Oscillator-based PUF with enhanced challenge-response pairs." Canadian Journal of Electrical and Computer Engineering 39.2 (2016): 174-180.

Enh-RO-PUF: Setup

- Choose an instance-specific **seed** S of n−1 random bits
- n ring oscillators have **frequencies** f_i
- The **comparison vectors** $\varphi(i)$ indicate for each ring, if the other rings oscillate faster or slower

The RO-PUFs secret

Enh-RO-PUF: Challenge and Response

- **Challenge** C is any subset $\{c_1,c_2,\ldots,c_k\}$ of $\{1,2,\ldots,n\}$
- For each challenge C, we shift the seed S by $c_1+c_2+\ldots+c_k$ bit. For the **shifted seed** we write $\rho(C)$

  Note that $\rho(C) = \rho(C \cup \{n-1\})$
- Finally, the **response** for challenge C is

$$\mathrm{res}(C) = \underbrace{\varphi(c_1) \oplus \ldots \oplus \varphi(c_k)}_{\text{XOR of all the comparison vectors for rings selected by the input}} \oplus \underbrace{\rho(C)}_{\text{Shifted seed intended to mask the output}}$$

III. Attack

Attack Step One: Recover $\varphi(n\text{-}1)$

using
2 Queries

- Shift operation ρ of seed S is cyclic
  - $\rho(C) = \rho(C \cup \{n-1\})$
- Choose challenges $C_1 = \{1\}$, $C_2 = \{1, n-1\}$

$$\mathrm{res}(C_1) = \varphi(1) \oplus \rho(C_1)$$

$$\mathrm{res}(C_2) = \varphi(1) \oplus \varphi(n-1) \oplus \rho(C_2)$$

$$\mathrm{res}(C_1) \oplus \mathrm{res}(C_2) = \varphi(n-1) \oplus \underbrace{\rho(C_1) \oplus \rho(C_2)}_{=\,0}$$

Attack Step Two: Recover Seed *S*

- Choose challenges $C_3 = \{n-1\}$

$$res(C_3) = \varphi(n-1) \oplus \rho(C_3) = \underbrace{\varphi(n-1)}_{\text{Known from attack step one}} \oplus S$$

using 1 Query

Attack Step Three: Recover All Other Comparison Vectors

- $\varphi(n-1)$ known from step one
- $\varphi(1)$ known after step two: we had $res(C_1) = \varphi(1) \oplus \rho(C_1)$
- To recover $\varphi(i)$, Choose challenge $C = \{i\}$

$$res(C) = \varphi(i) \oplus \underbrace{\rho(\{i\})}_{\text{Known from attack step two}}$$

using
n-2 queries

All secrets recovered after n+1 chosen queries

IV. Discussion

Security Implications

- We only break one proposed design choice of Delavar et al.
- Other design choices are secured by additional crypto primitives and hence out of scope
- Attack shown for attacker-chosen challenges, but can be extended to passive attacks
- Breaks all protocols based on the primitive

How did This Happen?

- Some assumptions used in the security analysis do not hold, e.g.
  Different challenges are not xored with unique random vectors, but with shifted versions of a single random vector
- Important design choices left open, e.g.
  Seed generation once or every time?
- Some conclusions used in the security analysis are not sound, e.g.
  High uniqueness does not imply unclonability

Future Work

How to Build Secure Strong PUFs?

- Still no secure strong PUF known
- Failed attempts:
  - Arbiter PUF by Gassend and Lim (attack also by Gassend and Lim)
  - XOR Arbiter PUF by Suh and Devadas (attack by Rührmair et al.)
  - Bistable Ring PUF by Chen et al. (attack by Xu et al.)
  - Ring Oscillator Sum PUF by Yu and Devadas (attack by Becker et al.)
- Not yet failed attempts:
  - Majority Vote XOR Arbiter PUF by myself (2017)
  - (modified) Arbiter PUF once more by Mispan et al. (2018)
  - Coin-Flipping PUF by Tanaka et al. (2018)
  - Dual-Mode PUF by Wang et al. (2018)
- Let’s turn to cryptographic constructions!

Questions & Answers

Attacking RO-PUFs with
Enhanced Challenge-Response
Pairs

Nils Wisiol
Marian Margraf

Freie Universität Berlin

http://idm.mi.fu-berlin.de
firstname.lastname@fu-berlin.de

DOI: 10.1007/978-3-319-99828-2

24th IFIP World Computer
Congress, TC-11 SEC, 18. Sep
2018, Poznan, Poland