The need for Hardware roots of trust

Ingrid Verbauwhede
KU Leuven
imec-COSIC

Slide acknowledgement:
former and current PhD students

ingrid.verbauwhede@esat.kuleuven.be
Outline

• Motivation: security and trust is required everywhere

• Design methodology

• Essential roots of trust

• Conclusion
Internet of Everything – IOT – Industry 4.0
E-...

- Internet of things
- E-health, e-commerce
- E-voting, e-...
- Smart grid
- Big data

Anything E- or Smart needs security
A root of trust is a component at a lower abstraction layer, upon which the system relies for its security.
How the crypto protocol paper sees it:

Some calculations are on the arrows?

Protocol relies on secrets and randoms

How to store a secret?

Permanently: e.g. for a master key
- Fuses: large, visible, limited numbers
- Non-volatile memory: extra processing
- Battery-backed SRAM, cumbersome, battery can die
- PUFs: physically unclonable functions

= a cost-efficient replacement technology for secure non-volatile memory (NVM)

[PhD Jeroen Delvaux]
Silicon PUFs - Variability

- Silicon Biometrics
- Variability in transistors and interconnect
- In general undesired, except for PUFs
- Random dopant fluctuation
- Line edge/width roughness
- Crucial design challenge with CMOS downscaling (Moore’s law)

Pelgrom’s law: $\sigma^2 \sim 1/\text{WL}$ (Marcel Pelgrom, Dutch engineer)
The ideal PUF?

Chip-dependent binary function with noisy output

Evaluation 1

≈ 1-15% noise

Evaluation 2

IDEAL PUF is without noise
PUF (F = Function)

- Dream 1: IDEAL PUFS don‘t exist...
- Two design methodologies for PUF circuits

Most strong PUFs broken: focus on weak PUFs for key generation
Weak PUF

- An array of identically designed circuit elements
- Each producing 1 (or a few) response bit(s)
- High-quality response bits, i.e., high entropy
- Limited number of bits, e.g., a few 1000s
- Weak because of limited response size, but the best in reality
- E.g., SRAM PUF, spot-break-down PUF

- Typical application: key generation

E.g. 128-bit AES
Strong PUF

- Finite number of physical building blocks combined with mathematical operations
- E.g., sum of delays, currents, voltages etc.
- Can produce a gazillion of response bits ($2^{128}$) \(\Rightarrow\) Strong
- Low-quality bits: highly correlated, low-entropy

- E.g., arbiter PUF
- Typical application: IC authentication
Digital Clone of a Strong PUF

- A full CRP read-out might not be feasible
- A gazillion of bits \(2^{128}\), computationally infeasible
- However, the bits are highly correlated
- Model building, using a small CRP training set
- Machine Learning: Artificial Neural Networks, Evolution Strategies, etc.
Arbiter PUFs: XOR Variant

- Arbiter PUF: original MIT work
- UNIQUE project result

Switch Block

Challenge:

Arbiter

0/1

Response:

0/1

Unique ASIC results

49%

≈7%

Temp./Volt. variation

Arbiter

Arbiter

Arbiter

Arbiter

Response: 0/1

6%

49%

3%

47%

≈7%

46%
Arbiter PUF – XOR Variant

- XOR the response of multiple chains
- More resistant against machine learning
  - # CRPs in training set ↑
  - Training time ↑
- Unfortunately, noise amplification as well
- Example: Becker et al. at CHES 2015

<table>
<thead>
<tr>
<th>ML Method</th>
<th>CRP Source</th>
<th>Pred. Rate</th>
<th>No. of XORs</th>
<th>CRPs ($\times 10^3$)</th>
<th>Training Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>FPGA</td>
<td>&gt; 99%</td>
<td>3, 4, 5</td>
<td>19.5, 39, 78</td>
<td>51.5 sec, 139 sec, 39 min</td>
</tr>
<tr>
<td>LR</td>
<td>ASIC</td>
<td>&gt; 99%</td>
<td>3, 4, 5</td>
<td>19.5, 39, 78</td>
<td>26 sec, 63.5 sec, 18:09 min</td>
</tr>
</tbody>
</table>

[Ruhrmair, IEEE TIFS 2013]
Dream or future research?

Wish a strong PUF:

- Finite number of elements
- Gazillion Challenge Response Pairs
- Non-linear combination to resist modeling attacks: ideally cryptographic functions
- BUT: noise amplification makes output not useful

Dream: strong PUF from finite number of elements, resistant to modeling, noise tolerant

Maybe: computational security?
Weak SRAM PUF: Basics

6T CMOS SRAM Cell

Guajardo et al. 2007, FPGA SRAM temp./volt. var.

Holcomb et al. 2009, Commercial SRAM

Holcomb et al. 2007, Embedded SRAM
PUF behavior of SRAM in commodity micro-controller

Black box approach (off the shelf micro-controllers)

- PIC16F1825
- STM32F100R8

Average bit value (%)

Within and between class HD (%)

Within Class

[PhD Anthony VH]
PUF behavior of SRAM in commodity micro-controller

Black box approach (off the shelf micro-controllers)

- PIC16F1825
- STM32F100R8

Needs post-processing to create key!
PUF Usage

[Strong PUF: Entity authentication]
Weak PUF: Key Generation
Light weight solution

- PUF is promised as ‘light-weight’ key generation for Internet of Things, RFID tags, etc.
- Key generation is larger than lightweight algorithm??

Research:
secure lightweight key generation!
Root of trust: True Random Number Generator

True Random Number Generators – compliant with AIS 31 and NIST

Entropy Source → Entropy Extractor → Post Processing

Raw bit → Random bit

Application

Prototype evaluation

On-line Testing

Entropy Source Testing

[PhD Vladimir Rozic, Bohan Yang]
TRNG: the old school

NIST statistical tests:

Random Number Generator

10110...

Statistical Tests

PASS / FAIL
Step 1: randomness source
Entropy source: Timing Jitter

\[ \text{Delay} = d_0 + \Delta d \]

More Oscillators
More Transitions
Efficient Entropy Extraction
FPGA Entropy extraction

TRNG based on Carry4 primitive

Better sampling resolution

Higher throughput

ES-TRNG at CHES 2018

DC TRNG at DAC 2015
Step 2: Post-processing
Step 3: On-line testing: TOTAL

Online testing design method

- HW (SW) Statistical Tests from standards
- Stochastic model (jitter model)
- TOTAL
- Canary numbers

TRNG On-the-fly Testing for Attack detection using Lightweight hardware

- Experiment oriented design methodology
- Tests tailored for entropy source and attacks

Canary Numbers: Design for Light-weight Online Testability of True Random Number Generators

- Explore online testability

Hot Topic
TOTAL: TRNG On-the-fly Testing for Attack detection using Lightweight hardware

1. Data Collection (normal operation and under attack)
2. Preliminary selection of useful features
3. Feature verification
4. Attack impact analysis
5. HW implementation
6. HW verification
Canary online testing:
Conclusion:
Root of trust: one level down!

Protocol: low power authentication protocol design

Algorithm: public key, secret key, hash algorithms, post-quantum

Architecture: Co-design, HW/SW, SOC

Micro-Architecture: arithmetic, co-processor design

Circuit: Circuit techniques to combat side channel analysis, TRNG’s PUFs,

At the bottom: PUFs and TRNGs