Posted on Authorea 30 Nov 2020 — The copyright holder is the author/funder. All rights reserved. No reuse without permission. — https://doi.org/10.22541/au.160677005.55609183/v1 — This a preprint and has not been peer reviewed. Data may be preliminary.

Hardware-Basierte Sicherheit, Homework 2

Alexander Uzikov[1], Zekarias Tiruneh[1], Nitin Varghese[1], Shifat Sahariar[1], and Anton Komar[1]

[1]Affiliation not available

November 30, 2020

Authors: Alexander Uzikov(ID 105825), Shifat Sahariar Bhuiyan(ID 86582), Anton Komar(105826), Zekarias Tiruneh(ID 106773) Nitin Varghese(ID 82840)

Task 1. The concept of Physical Unclonable Functions

1. PUF is a function that is embedded into a physical object. This function can be based on a physical form(like in optical PUFs), memory-based properties, electrical properties of an object. The core idea is that the function is able to reply on challenge with a response that is based on unclonable property of an object. It can be used for identification, authentification, remote attestation, key storage, random number generators and so on.
2. Unclonability(the object must be impossible to clone, even for manufacturer), Robustness(the same challenge x must always produce almost the same response y), unpredictability(it is impossible to predict response for x that was not seen before), Tamper-Evidence(if OUF has such property, it changes its behavior when attacked invasively)
3. Intrinsic PUF is a function that is already a part of device and do not have to be added to a design. Usually these are memory based PUFs.

Task 2

1. Optical PUF has a transparent material with (reflective) optical particles, scattered at random positions. A laser beam is focused on this material, which produces a pattern due to reflection of beam from particles. This pattern is then recorded and used. For example, if it is used for an verification, it can be compared to the pattern produced by the same beam before.

   The main drawback is that such PUF requires optical hardware(which add costs), can require a lot of space. It results in inability to use such PUFs in, for example, IOT. Another drawback is that measurement of responses must be precise.
2. An arbiter PUF has two signals that run through parallel chains of arbiters. Both chains have the same number of arbiters, therefore arbiters can be divided on pairs. Each pair receives a bit from a challenge. Depending on this bit, signals might either swap or stay at the same chain during running through a pair of arbiters. Since physical properties of arbiters are not exactly the same, signals will not come at the same time.

Posted on Authorea 30 Nov 2020 — The copyright holder is the author/funder. All rights reserved. No reuse without permission. — https://doi.org/10.22541/au.160677005.55609183/v1 — This a preprint and has not been peer reviewed. Data may be preliminary.

The problem of arbiter PUF is that design must be precisely "symmetric" to get random response. Another problem is that chains must be long, which results in signal delay. Results can also be predicted using machine learning.

3. Ring Oscillator PUF:Works on a number of free-running oscillators by counting the oscillation frequency of a pair of oscillators. And the oscillation frequency depends on signal delays and it is hard to control. The output bit corresponds to the oscillator that is faster than other oscillator. The main drawback is, only a few physical oscillators generate responses (dependence!).
4. SRAM PUFS are cross coupled inverters.It works on Infinite loop on startup.At a certain point of time due to production variations the larger inverter wins and draws values to a certain bit for instance 0 or 1. The disadvantage of SRAM puf is the high error rate between outputs of the same PUF
5. DRAM cell consists of a capacitor and a transistor. Due to manufacturing variations some cells decay faster than others which is explioted as a PUF.The main advantages of DRAM PUFs over SRAMS are low error bits since the decay depends on temperature same under different temperature and equivalent decay time same decay can be observed. Read out time of DRAM PUFS is in order of minutes which can be seen as a disadvantage(https://eprint.iacr.org/2016/253.pdf).

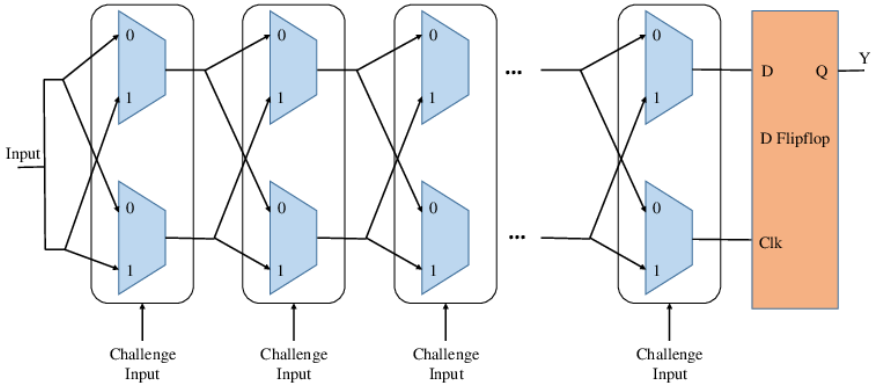

Figure 1: Arbiter PUF

Task 3. PUF Metrics

1. Hamming weight. Ideally, if PUF is not biased, relative value must be close to $\frac{1}{2}$. Otherwise, probabilities of different sequences are not equal.
2. Intra hamming distance of responses. If one bit in challenge is changes, it must result in change of approximately half of bits.
3. Inter Hamming distance of responses. Applying the same challenge to different PUFs must result in difference in approximately half of bits.

   Another metric for quantifying physical PUF responae is a Jaccard Index. $J(A,B) = \frac{A \cap B}{A \cap B}$
4. Intra Jaccard Index quantifies response on the same challenge. Ideally response must be robust and value must be close to 1.

Posted on Authorea 30 Nov 2020 — The copyright holder is the author/funder. All rights reserved. No reuse without permission. — https://doi.org/10.22541/au.160677005.55609183/v1 — This a preprint and has not been peer reviewed. Data may be preliminary.

5. Inter Jaccard Index measures response on different PUF challenges. Ideally it it must be must be close to 0 so that function is unclonable and unpredictable.

Task 4. Error correction and PUF applications

1. 1) Issue: Repeated writes to DRAM cells tend to introduce errors in memory. Location of errors are unique for each device.

   Solution: reserve memory area, write repeatedly and observe bit flips.

   2) Issue is that some PUF types can be: sensitive to supply voltage variations, influenced by temperature variations, responding to challenges with entropy/min-entropy.

   Solution: choose the optimal PUF type for certain conditions to minimize imperfections.

   3) Issue: for example, to derive a cryptographic key, we need a 100% stable fingerprint. Often this is not the case and errors need to be corrected.

   Solutions: brute-force enumeration of error patterns (it is not feasible for complex error patterns and only applicable in the authentication scenario), indices to stable bits in response (does not always yield an error-free response, because requires bits that are 100% stable, this is often not the case and indices to stable bits may have PUF characteristics themselves), code offset method (may be slow in hardware).Methods can be used together if code offset method cannot remove all errors.

2. PUFS can be used in such cases:

   1) Key storage the advantage of using PUF here is that the cryptographic key need not be in memory.

   2) Identification and authentication (device, client, server)

   3) Random Number Generators. Stable PUF response allows identification, noise is used for randomness generation

   4) Hardware-Software-Binding. Alter program behavior based on PUF (correct behavior only if correct PUF is in use)

   5) Remote Attestation

   6) Building block for crypto primitives (block ciphers, oblivious transfer)